IMS
15.2.0

System Administration (2024-08-30 edition)

IBM

2024-08-30 edition.

This edition applies to IMS 15 (program number 5635-A06), IMS Database Value Unit Edition, V15.02.00 (program number 5655-DS5), IMS Transaction Manager Value Unit Edition, V15.02.00 (program number 5655-TM4), and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About this information

These topics provide guidance information for managing the administration and operations of a single IMS or one or more IMS systems that work as a unit (an IMSplex).The topics describe designing, documenting, operating, maintaining, and recovering an IMS system, as well as the Database Recovery Control (DBRC) facility, the Extended Recovery Facility (XRF), the IMSRSC Repository, and the Repository server. The topics also include information about the IMS Base Primitive Environment (BPE), IMS Common Queue Server (CQS), and IMS Common Service Layer (CSL), all of which can be part of an IMSplex, as well as information for sharing data and message queues in an IMSplex.

This information is available in IBM® Documentation.

Prerequisite knowledge

Before using this book, you should understand basic z/OS® and IMS concepts and your installation's IMS system. IMS can run in the following environments: DB Batch, DCCTL, TM Batch, DB/DC, DBCTL. You should understand the environments that apply to your installation. You should also have a basic understanding of database processing and the access methods used by DL/I. It is helpful to know the purpose of the different types of DL/I calls, the IMS application program structure, and the tasks associated with application program design.

To learn about z/OS, see z/OS Basic Skills. For more resources, see IBM Z Education and Training.

To learn about IMS, see the IBM Press publication *An Introduction to IMS*, the resources listed for IBM Information Management System, and the variety of options available in IBM Training.

How new and changed information is identified

For most IMS library PDF publications, information that is added or changed after the PDF publication is first published is denoted by a character (revision marker) in the left margin. The *Program Directory* and *Licensed Program Specifications* do not include revision markers.

Revision markers follow these general conventions:

- Only technical changes are marked; style and grammatical changes are not marked.
- If part of an element, such as a paragraph, syntax diagram, list item, task step, or figure is changed, the entire element is marked with revision markers, even though only part of the element might have changed.
- If a topic is changed by more than 50%, the entire topic is marked with revision markers (so it might seem to be a new topic, even though it is not).

Revision markers do not necessarily indicate all the changes made to the information because deleted text and graphics cannot be marked with revision markers.

Accessibility features for IMS 15.2

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility features

The following list includes the major accessibility features in z/OS products, including IMS 15.2. These features support:

- Keyboard-only operation.
- Interfaces that are commonly used by screen readers and screen magnifiers.

- Customization of display attributes such as color, contrast, and font size.

Keyboard navigation

You can access IMS 15.2 ISPF panel functions by using a keyboard or keyboard shortcut keys.

For information about navigating the IMS 15.2 ISPF panels using TSO/E or ISPF, refer to the *z/OS TSO/E Primer*, the *z/OS TSO/E User's Guide*, and the *z/OS ISPF User's Guide Volume 1*. These guides describe how to navigate each interface, including the use of keyboard shortcuts or function keys (PF keys). Each guide includes the default settings for the PF keys and explains how to modify their functions.

Related accessibility information

Online documentation for IMS 15.2 is available in IBM Documentation.

IBM and accessibility

See the *IBM Human Ability and Accessibility Center* at www.ibm.com/able for more information about the commitment that IBM has to accessibility.

How to send your comments

Your feedback is important in helping us provide the most accurate and highest quality information. If you have any comments about this or any other IMS information, you can take one of the following actions:

- Submit a comment by using the DISQUS commenting feature at the bottom of any IBM Documentation topic.
- Send an email to imspubs@us.ibm.com. Be sure to include the book title.
- Click the **Contact Us** tab at the bottom of any IBM Documentation topic.

To help us respond quickly and accurately, please include as much information as you can about the content you are commenting on, where we can find it, and what your suggestions for improvement might be.

Part 1. Introduction to IMS system administration

This topic introduces the IMS system. It also introduces the concepts that are central to administering an IMS system.

Chapter 1. Introduction to the IMS system

This topic outlines the administrative activities for an IMS system. It also introduces the concepts that are central to administering an IMS system.

Overview of administrative activities

To administer an online IMS system, you must design an IMS online system, establish operating procedures that meet application requirements, maintain a production system that is responsive to end users, and integrate new applications or major design changes into the current system.

To meet these responsibilities, you must coordinate many activities that occur during an application development cycle. Performance of these activities results in:

- Documentation of the IMS network
- Specifications for IMS system definition and execution control parameters
- Procedural controls for operation
- Strategies for monitoring and audit trails

After entry into production mode, your ongoing activities support:

- Auditing operations and end user service
- Monitoring and gathering production statistics
- Establishing procedures to control changes to the online system design
- Testing the online system after application and IMS changes

The following figure is an overview of the administration activities. The activities in the first column of the figure take place during the design phase, the second column's activities occur during development of application code, and those in the third column occur during test. The vertical center line marks the transition to production mode.

Before the start of production mode, administration activities lead to two major activities:

- Generating the system and preparing JCL (job control language)
- Developing operations procedures

The items to the right of the center line in the figure reflect the ongoing system administration activities. The column headed by PRODUCTION emphasizes awareness of the day-to-day operation and performance of the system. The next column's activities are concerned with maintenance. Minor application design changes, problem resolution, and any IMS maintenance are included in this category. The column on the far right identifies the activities needed to integrate additional applications or implement an application package. For a major addition, activities are similar to those on the left-hand side of the figure. Other design changes that do not involve terminal or network modifications can be handled without shutting down the IMS. For these simple design changes, the associated activities lead to a revision of operating procedures in maintenance mode.

After startup, revisions of IMS system definition and operating procedures become key activities. In this context, you might decide to change applications during online operation. The interpretation of system performance then becomes an important activity, supported by monitoring and performance analysis.

Each row in the figure represents a set of activities, each having its own characteristics:

- Analyzing user requirements involves examining the application documentation for the IMS function required and the expected workload.
- Collecting online requirements concerns specifications for IMS system definition, system data set allocation, and initial JCL.
- Preparing the IMS network involves interaction with system and network generation activities.

- Establishing security procedures encompasses the design and implementation of a security strategy.
- Developing an operations plan produces operations control documents and provides for audit of the control of production cycles.
- Forming a monitoring strategy results in monitoring the system and gathering performance data.
- Establishing criteria for performance leads to performance analysis and tuning activities.

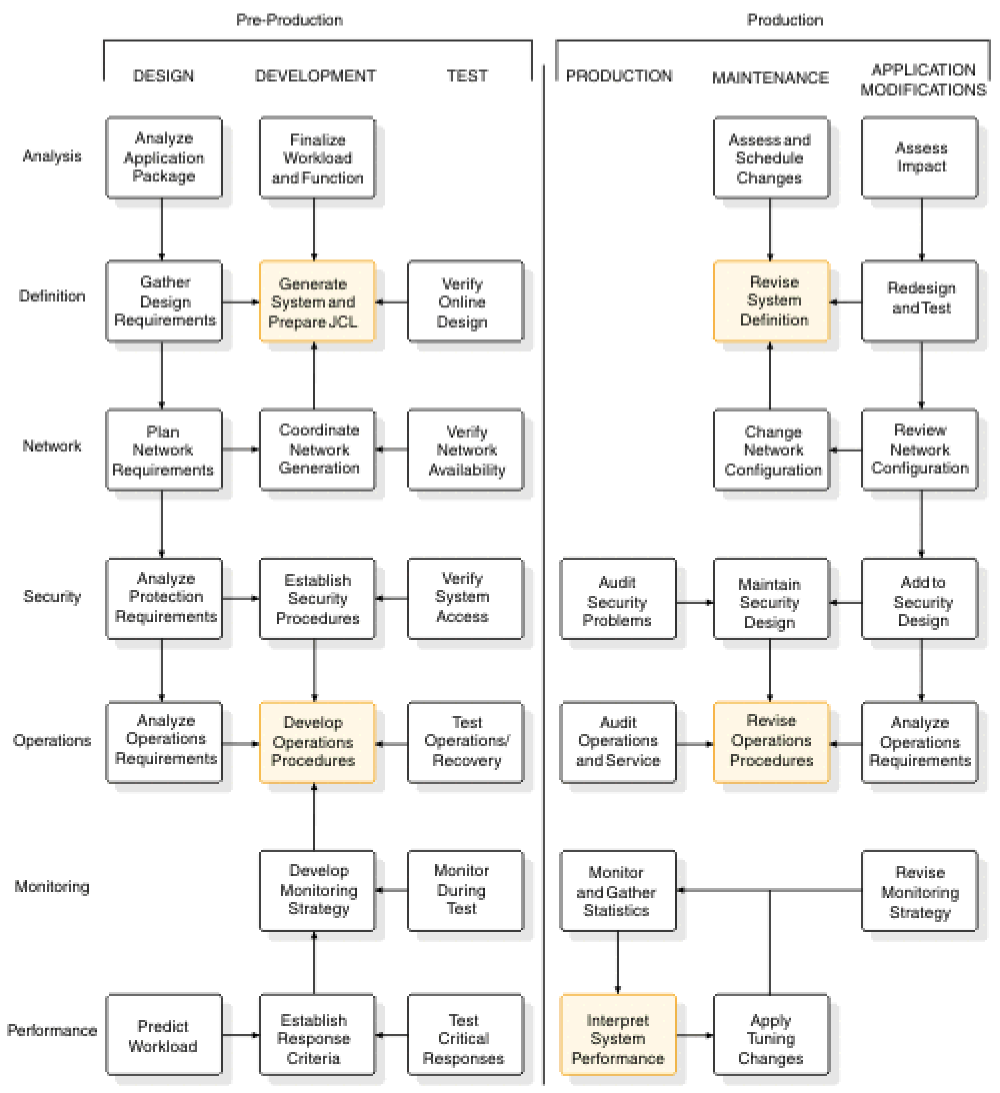

Pre-Production
Production
DESIGN
DEVELOPMENT
TEST
PRODUCTION
MAINTENANCE
APPLICATION MODIFICATIONS
Analysis
Definition
Network
Security
Operations
Monitoring
Performance
Analyze Application Package
Finalize Workload and Function
Assess and Schedule Changes
Assess Impact
Gather Design Requirements
Generate System and Prepare JCL
Verify Online Design
Revise System Definition
Redesign and Test
Plan Network Requirements
Coordinate Network Generation
Verify Network Availability
Change Network Configuration
Review Network Configuration
Analyze Protection Requirements
Establish Security Procedures
Verify System Access
Audit Security Problems
Maintain Security Design
Add to Security Design
Analyze Operations Requirements
Develop Operations Procedures
Test Operations/ Recovery
Audit Operations and Service
Revise Operations Procedures
Analyze Operations Requirements
Develop Monitoring Strategy
Monitor During Test
Monitor and Gather Statistics
Revise Monitoring Strategy
Predict Workload
Establish Response Criteria
Test Critical Responses
Interpret System Performance
Apply Tuning Changes

Figure 1. Chart of IMS system administration activities

Related reading: This publication does not address the detailed planning needed to establish operating procedures. For information on establishing operating procedures, see *IMS Version 15.2 Operations and Automation*.

IMS environments and configurations

IMS contains two major components that can be used together or separately: the IMS Database Manager (DB) and the IMS Transaction Manager (TM). When you use them together, they constitute the DB/DC environment.

Using the Database Manager alone, you can generate the batch environment and the database control (DBCTL) environment. Using the Transaction Manager alone, you can generate the data communication control (DCCTL) environment. Data sharing and the Extended Recovery Facility (XRF) are often considered environments, but they are special cases of the three environments listed here.

In addition to the three environments (DBCTL, DB/DC, and DCCTL), you can configure IMS as an *IMSplex*.

Use the major components of IMS to configure your environment as part of your system definition process, based on your business needs.

The DB/DC, DCCTL, and DBCTL environments are all considered *online* IMS systems.

Each of the IMS environments is a distinct combination of hardware and programs that supports distinct processing goals. The online environments and the goals they support are shown in the following table.

Table 1. IMS online environments

Environments	Data processing goals
DBCTL (See "DBCTL environment" on page 9)	• Process network transactions without the Transaction Manager—that is, use the Database Manager with a transaction management subsystem (for example, CICS®). • Run batch application programs using DB batch at certain intervals (for example, process a payroll or produce an inventory report). • Run database utilities using DB batch.
DB/DC (See "DB/DC environment" on page 5)	• Enable terminal users to retrieve data and modify the database with satisfactory real-time performance. (Some typical applications are banking, airline reservations, and sales orders.) • Ensure that retrieved data is current. • Distribute transaction processing among multiple CPUs in a communications network. • Run batch application programs using DB batch at certain intervals (for example, process a payroll or produce an inventory report). • Run database utilities using DB batch. • Run application programs that access external subsystems or access data in external subsystems, such as data in Db2 for z/OS.
DCCTL (See "DCCTL environment" on page 11)	• Process network transactions without the Database Manager by using the Transaction Manager with an external database management subsystem. • Maintain system log information for restart by using DBRC. • Run application programs that access external subsystems or access data in external subsystems, such as data in Db2 for z/OS.

Related concepts
"IMSplex overview" on page 16
An IMSplex is made up of IMS and z/OS components that work together.

DB/DC environment

In the DB/DC environment, data is centrally managed for applications that are being executed concurrently and made available to terminal users. Database Recovery Control (DBRC) facilities help to manage database availability, data sharing, and system logging.

The following figure shows an example DB/DC environment.

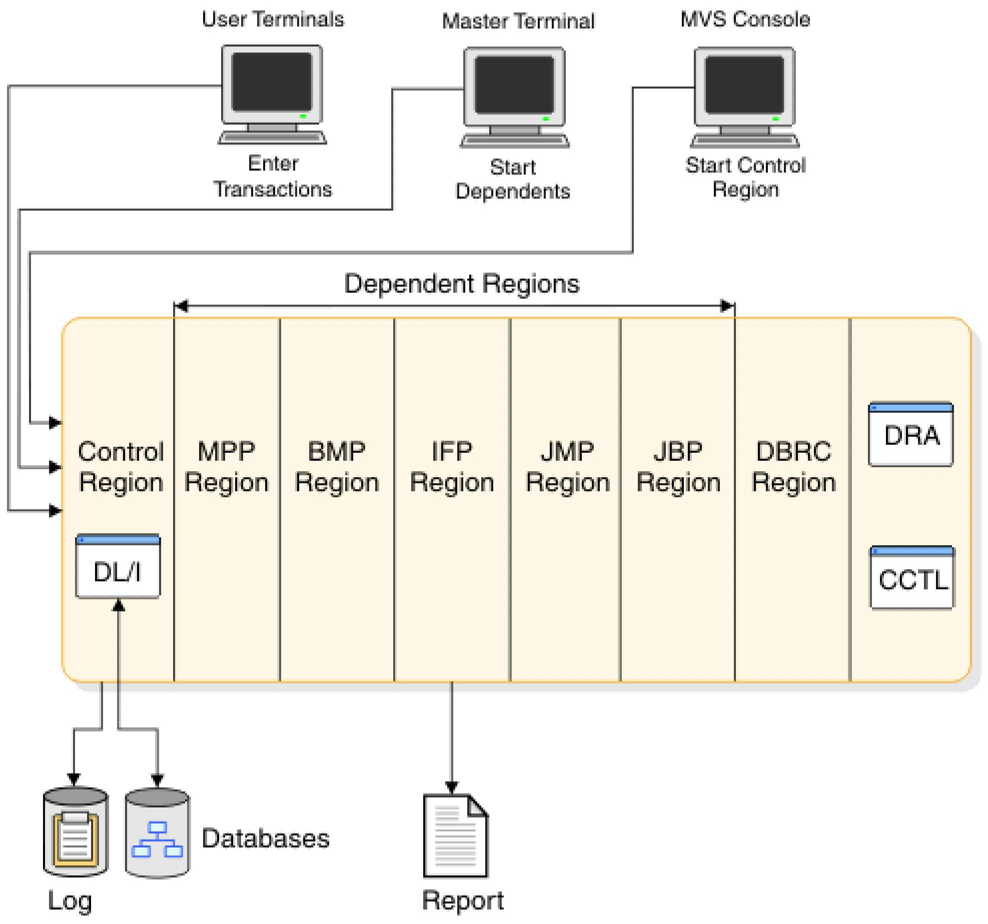


Figure 2. Example of a DB/DC environment

In the figure, DL/I is shown as part of the control region, but it does not need to run there. You can run DL/I in its own address space.

To understand the DB/DC Environment and how it works, it is important to understand the *transaction*. The basic unit of work in a DB/DC environment is the transaction. Transaction processing consists of:

- Receiving a request for work that has been entered at a terminal. The request is in the form of a transaction code, which identifies the kind of work to be performed and the data needed to do it.
- Invoking a program to do the work, and preparing a response for the terminal operator (for example, an acknowledgment of work performed or an answer to an inquiry).
- Transmitting the response to the terminal that requested the work.

The simplest kind of transaction involves two messages: an input message from the terminal user and an output message in return. Application programs can also send messages to terminals other than the input source, and they can generate transactions.

Related concepts

"Transaction flow in DB/DC and DCCTL environments" on page 374
A distinct sequence of events occurs during the processing of a transaction. Message-related processing is asynchronous within IMS, that is, not associated with a dependent region's processing.

Control region

The IMS control region holds the *control program*, which continuously runs in the control region address space and controls the processing in other regions.

The IMS control region services all DL/I calls either directly or through the DL/I separate address space (DLISAS). The IMS control region owns all the databases that can be accessed by online application programs and is responsible for all physical input/output to the databases. The IMS control region supervises the processing of messages and all the communication traffic for the connected terminals. The IMS control region also manages information for restart and recovery purposes and operates the IMS system log.

The IMS control region is normally started by using the z/OS **START** command. The IMS control region then automatically initiates the DBRC address space.

Database Recovery Control facility

The Database Recovery Control facility (DBRC) helps you control log and database recovery. It also controls the data sharing environment by allowing or preventing access to databases by the IMS systems that share those databases.

DBRC runs in its own address space, but is subordinate to the IMS control region. DBRC is required by all online IMS systems and any system that uses data sharing.

Related concepts

“Overview of DBRC” on page 473
A feature of the IMS Database Manager that facilitates easier recovery of IMS databases, DBRC maintains information that is required for database recoveries, generates recovery control statements, verifies recovery input, maintains a separate change log for database data sets, and supports sharing of IMS databases and areas by multiple IMS subsystems.

Dependent regions

The dependent regions are separate address spaces from the control region, which are dependent on IMS and in which IMS schedules the applications that process transactions. Dependent regions are initiated by a z/OS **START** command or by a `/START REGION` command from the IMS master terminal.

The following are different types of dependent regions:

- Message Processing Program (MPP) Region.

  MPP regions are started either by the master terminal operator or by JCL if the control program is running. The control program schedules application programs within the MPP regions. The application programs then run, accessing the online databases and obtaining their transaction input from the message queues. The application programs cannot access z/OS files or issue z/OS checkpoints. The application program output messages can be directed to LTERMs or to other application programs. An application program can remain scheduled in an MPP region even when there is no work to process for that region. The MPP region remains in a wait-state (wait-for-input mode) until there is more work for the region to process.

- Batch Message Processing (BMP) region.

  z/OS schedules the BMP regions. The application programs in those regions are determined by the JCL used to start each region, not by the control region. These application programs can access databases owned by the control region and z/OS data sets owned by their BMP regions. z/OS data sets include data entry databases (DEDBs) and main storage databases (MSDBs).

  Application programs in BMP regions can access input and output message queues; they can also execute in wait-for-input mode. To access the input message queues, you specify, in the JCL for a BMP region, a transaction code you want to access. Specifying this transaction code also gives you access to the output message queues using terminal program communications blocks (PCBs) in the application program's specification block (PSB). Even without access to input message queues, if you specify an output LTERM or transaction code in the JCL for the region, the application program can issue output messages.

- IMS Fast Path (IFP) region.

  Two types of programs run in IFP regions:

  - Application programs for processing Fast Path messages; these are called message-driven programs.
  - Utilities that process DEDBs; these are BMPs.

- Java™ Message Processing (JMP) region.

  JMP regions process messages with either applications written in Java or applications written in both Java and OO COBOL. JMP regions can load 31-bit or 64-bit Java virtual machine (JVM).

- Java Batch Processing (JBP) region.

  JBP regions process batch operations with either applications written in Java or applications written in both Java and OO COBOL. JBP regions can load 31-bit or 64-bit Java virtual machine (JVM).

Related reading:

- For more information on the types of Fast Path processing and the administration of a DB/DC environment that includes Fast Path, see *IMS Version 15.2 Database Administration*.
- For more information on writing applications that can run in JMP and JBP regions, see *IMS Version 15.2 Application Programming*.

The master terminal

The master terminal is the control center of the DB/DC environment. The Master Terminal Operator (MTO) must know all the operating aspects of the system and be familiar with the purpose and action of all the IMS commands that are entered.

Some characteristics of the master terminal are:

- It is used to enter commands that start, stop, and restart the system.
- As a logical terminal, it receives system messages.
- The primary control of the network and connecting terminals is performed through the master terminal. It can start and stop communication lines and assign logical terminals to physical terminal destinations.
- The status of the system can be displayed from the master terminal. Such items as the number of transactions to be processed, the number of programs and databases that are active, and the status of communication lines can be requested.
- If a program or database error occurs, commands can be entered from the master terminal to prevent further processing against the affected resource and to prevent input or output activity for terminals. These recovery actions can also apply to a recovery of the entire system after an abnormal termination occurs.
- If an exception condition occurs, the status of the Online Log Data Set (OLDS) can be displayed, and the function of the OLDS can be controlled from the master terminal.

If the master terminal becomes inoperable, the operating system console can be used as a backup. The MTO can either operate the IMS from the system console or assign the master terminal LTERM (logical terminal) to an alternate terminal. Messages continue to be routed to the old master terminal LTERM until the LTERM is assigned to the system console. The address of the system console is LINE 1 PTERM 1. If the system console is used to continue operation, terminal security is identical to that of the master terminal.

Databases supported in a DB/DC environment

The DB/DC environment supports all full-function databases (HSAM, SHSAM, HISAM, SHISAM, HDAM, PHDAM, HIDAM, PHIDAM, and PSINDEX).

BMP regions in a DB/DC environment can access GSAM databases. BMP regions can also access external subsystems (for example, Db2 for z/OS), because DB/DC supports the external subsystem interface.

Fast Path data entry databases (DEDBs) and main storage databases (MSDBs) are also supported.

Note: PHDAM, PHIDAM, and PSINDEX are database types added for High Availability Large Database (HALDB). They are partitioned equivalents of HDAM, HIDAM, and secondary indexes, respectively.

Fast DB Recovery region in DB/DC

The Fast DB Recovery region is a separate IMS control region that monitors an IMS, detects failure, and recovers any IMS-owned database resources that are locked by the failed IMS, making them available for other IMS systems without having to wait for the next full restart.

For database resources that are not IMS owned, such as Db2 for z/OS databases, the Fast DB Recovery region provides an optional exit routine, ESAF In-Doubt Notification exit routine (DFSFIDN0), for this purpose.

The Fast DB Recovery region is executed by the cataloged procedure that is supplied by IMS system definition. You must start the Fast DB Recovery region after you start the IMS that it tracks.

To enable a DB⁄DC IMS for Fast DB Recovery, specify the FDRMBR parameter in the IMS procedure. The FDRMBR parameter defines the DB⁄DC system as Fast DB Recovery-capable.

Data sharing in a DB/DC environment

Data can be shared among dependent regions and with other IMS systems. The other systems can be DB⁄DC or DBCTL.

If you intend to share data at the block level, the Internal Resource Lock Manager (IRLM) must be present in every environment that participates. IRLM runs in its own address space.

Running the Extended Recovery Facility

The Extended Recovery Facility (XRF) is a combination of programs that includes two DB⁄DC environments to provide a high level of IMS availability to end users.

One environment is active and is called the active system. The other environment continuously tracks the processing of the first and is called the alternate system. The alternate system is ready to take over if the active system fails, or if a planned takeover is initiated (to perform maintenance, for example).

DBCTL environment

The DBCTL environment is similar to the DB⁄DC environment; a DL⁄I region owns the databases to be processed. DL⁄I also exists in the DBCTL environment, although DL⁄I must run in its own address space. Database Recovery Control (DBRC) facilities, required for DBCTL, help to manage database availability, data sharing, system logging, and database recovery.

The greatest dissimilarity between DBCTL and DB⁄DC is that DBCTL does not support user terminals, a master terminal, or message handling. Therefore, no MPP regions exist. The BMP region is used only by batch applications and utilities. External program subsystems can, however, use an interface that does handle messages—a coordinator controller (CCTL). The same interface exists in the DB⁄DC control region, so it is possible to use a CCTL with a DB⁄DC environment. The interface between the CCTL and the control region is the database resource adapter (DRA). The DRA resides in the same address space as the CCTL.

The CCTL handles message traffic and schedules application programs, all outside the DBCTL environment. It passes database calls through the interface to the control region, which sends the calls to DL⁄I and passes results back through the interface to the CCTL.

The topics in this information that describe the IMS online system applies to both DB⁄DC and DBCTL. Exceptions are noted as not applicable to DBCTL.

The following figure shows an example of the DBCTL environment.

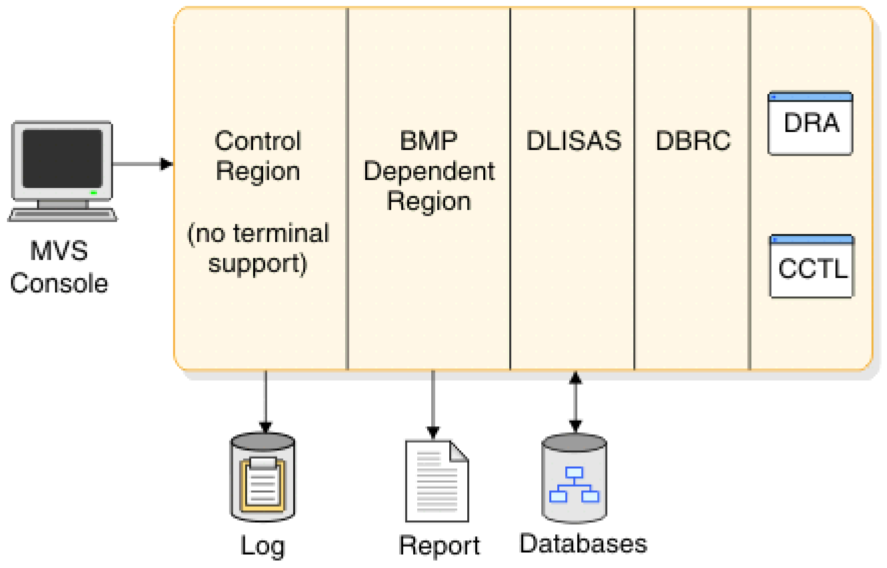


Figure 3. Example of a DBCTL environment

Even though the DBCTL environment includes no master terminal, you can still control the environment with IMS commands. The commands and command functions that control message processing are not operable here, but the remainder are. They can be entered through the MVS™ console or a secondary console. The control region recognizes commands by their first character, a slash (/). You can choose a different first character during system definition, or as an execution parameter.

Note: References to the master terminal operator (MTO) in this information refer to either the DB/DC MTO or to the DBCTL operator.

Output messages from a command are sent to the console that entered the command. You can also specify other consoles to receive unsolicited output. These consoles are those that fall into the category you define using the IMS-generating macro, IMSCTRL.

Databases supported by DBCTL

The DBCTL environment supports all full-function databases (HSAM, SHSAM, HISAM, SHISAM, HDAM, PHDAM, HIDAM, PHIDAM, and PSINDEX).

BMP regions in a DBCTL environment can access GSAM databases. BMP regions can also access external subsystems (for example, Db2 for z/OS), because DBCTL supports the external subsystem interface.

The DBCTL environment supports Fast Path data entry databases (DEDBs).

Note: PHDAM, PHIDAM, and PSINDEX are database types added for High Availability Large Database (HALDB). They are partitioned equivalents of HDAM, HIDAM, and secondary indexes, respectively.

Fast DB Recovery region in DBCTL

The Fast DB Recovery region is a separate IMS control region that monitors an IMS, detects failure, and recovers any IMS-owned database resources that are locked by the failed IMS, making them available for other IMS systems without having to wait for the next full restart.

For database resources that are not IMS owned, such as Db2 for z/OS, the Fast DB Recovery region provides the optional ESAF Indoubt Notification exit routine (DFSFIDN0) for this purpose.

The Fast DB Recovery region is executed by the cataloged procedure supplied by IMS system definition. You must start the Fast DB Recovery region after you start the IMS that it tracks.

To enable a DBCTL subsystem for Fast DB Recovery, you specify the FDRMBR parameter in the DBC procedure. The FDRMBR parameter defines the DBCTL system as Fast DB Recovery-capable.

Data sharing

As in the DB/DC environment, data can be shared between dependent regions and with other IMS systems. The other systems can be either DB/DC or DBCTL environments.

If you intend to share data at the block level, IRLM must be present in every environment that participates.

Alternate DBCTL environment

You cannot have XRF in a DBCTL environment, but you can still run two DBCTL environments—an active and an alternate—and thereby increase system availability.

However, the alternate DBCTL environment does not track the processing of the active environment. The console operator must use the emergency restart command (**/ERESTART**) against the alternate system in order to make it the active environment.

DCCTL environment

DCCTL is an IMS Transaction Manager subsystem that has no database components. A DCCTL environment is similar to the DB/DC environment. The primary difference is that a DCCTL control region owns no databases and does not service DL/I database calls.

DCCTL subsystems do not support the IMS catalog.

DCCTL, in conjunction with the IMS External Subsystem Attach Facility (ESAF) or the Db2 for z/OS Recoverable Resource Manager Services Attach Facility (RRMS), provides a Transaction Manager facility to external subsystems (for example, Db2 for z/OS). In a DCCTL environment, transaction processing and terminal management is identical to transaction processing and terminal management in a DB/DC environment. DCCTL contains the programming support necessary for:

- Master terminal support
- Terminal network support
- Data communication
- Message handling
- Transaction processing
- Application program execution
- IMS command execution

DCCTL also supports online change, Message Format Service (MFS), Multiple Systems Coupling (MSC), and Database Recovery Control (DBRC).

All IMS commands are supported in a DCCTL environment except database commands and database-related keywords.

DBRC is required, and is used to maintain system log information for restart. DBRC in a DCCTL environment maintains logs only for transactions. External database subsystems must maintain their own database logs.

DCCTL consists of three address space types:

- Control region
- DBRC
- Dependent regions (up to 999)

Dependent regions and DBRC are subordinate to the control region.

The DCCTL control region contains three structural components:

- A data communication manager, which controls terminal states and input/output message traffic. It also contains security controls that prevent unauthorized access to DC resources.

- A message manager, which is the read/write and I/O interface between terminal input from the data communication manager and the scheduling services of the Transaction Manager.
- A Transaction Manager, which manages MPPs, BMPs, and IFPs, schedules application programs in those dependent regions, and owns and responds to the application programming interface (API).

Each manager (data communication manager, Transaction Manager, and message manager) controls the use of its resources and the recoverability of its resources during a system failure. Like DB/DC dependent regions, MPP, BMP, IFP, JMP, and JBP dependent regions are used by the Transaction Manager to schedule application programs.

The following figure represents a DCCTL environment that is attached to an external subsystem.

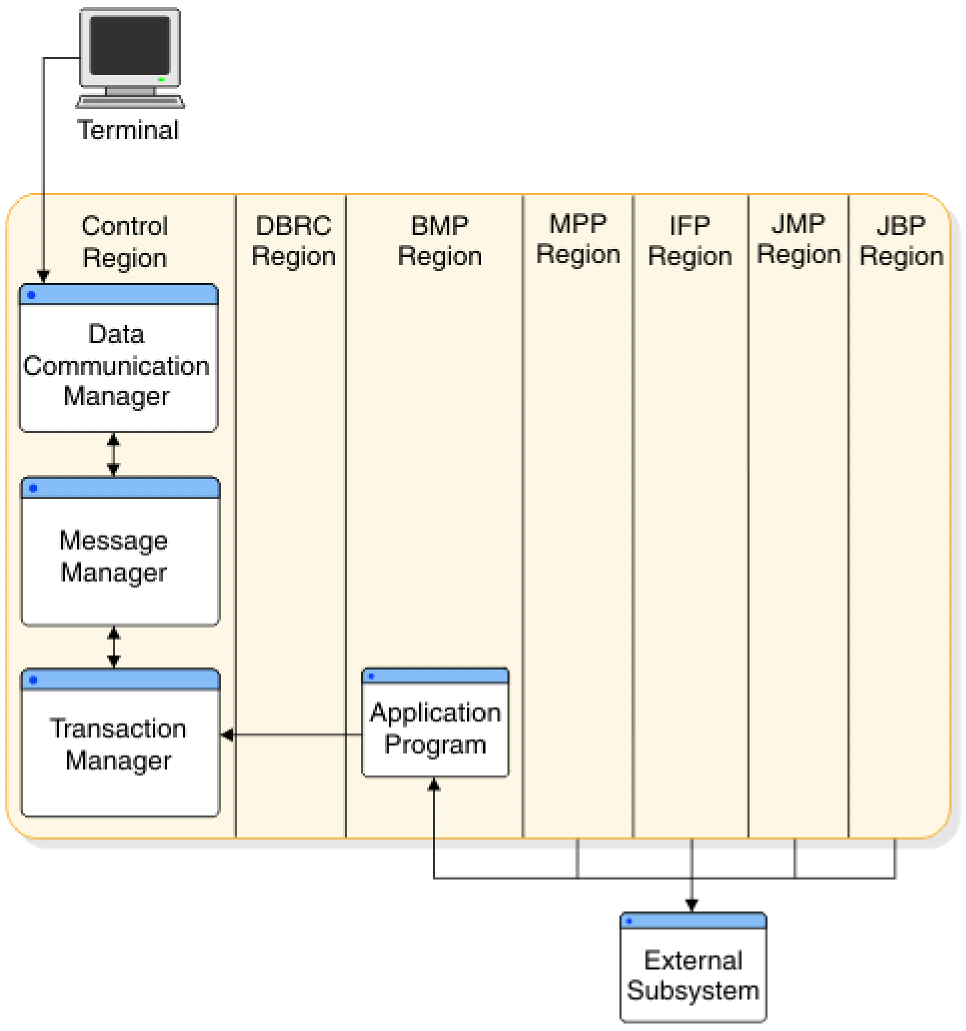


Figure 4. Example of a DCCTL environment and attached subsystem

DCCTL coordinates the sync point recovery process with the connected external subsystems. DCCTL ensures that database updates and terminal messages are committed when an application program reaches a sync point.

Two methods exist for connecting a DCCTL control region to another subsystem. You can have DCCTL use the control region EXEC parameter, SSM, to select the PROCLIB member. Or, you can use the **/START SUBSYSTEM SSM** command, which allows DCCTL to connect to other subsystems even though you did not request this option when you started IMS.

You can also specify the dependent region EXEC parameter, SSM, for dependent regions. The control region SSM member definition allows the dependent region to select one or more external subsystem connections. The SSM member can contain no definitions (null members) to prevent a connection to an external subsystem.

After you use the **/START SUBSYSTEM SSM** command, you must stop active dependent regions and then restart them if they require an external subsystem connection.

After IMS has established a connection between the dependent region and the external subsystem, a thread is created between the connected regions. The thread is used for subsequent application program calls, committing the data, or in failure situations, backing out the data.

The application programs managed by DCCTL are identical to those managed by the DC manager and TM manager in the DB/DC environment.

Related reading:

- For more information about the ESAF and RRSAF interfaces, see Chapter 30, "Accessing external systems from within IMS," on page 469.
- For more information on using the **/START SUBSYSTEM SSM** command, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Databases supported by DCCTL

The DCCTL configuration of IMS supports and is a compatible communications front end for database and dependent region combinations.

The following database and dependent region combinations are compatible communications front end for a DCCTL configuration of IMS:

- GSAM databases for BMP regions
- Db2 for z/OS databases for:
  - JMP, JBP, BMP, MPP, and IFP regions through the External Subsystem Attach Facility (ESAF)
  - JMP and JBP regions through the DB2® Recoverable Resource Services attachment facility (RRSAF)

Restriction: DCCTL does not support the following database types:

- Fast Path databases
- Full-function databases
- The IMS catalog, a HALDB full-function system database

With GSAM databases, DCCTL uses sequential, non-IMS data sets with a BMP. Application programs can also issue symbolic checkpoint (CHKP) and extended restart (XRST) calls against a GSAM data set using the I/O PCB. The ability to issue CHKP and XRST calls allows data set repositioning.

When DCCTL accesses Db2 for z/OS databases through the ESAF or RRSAF interfaces, the control region initiates contact with other subsystems. The other subsystems that DCCTL can access are defined in a subsystem member (SSM) of the IMS.PROCLIB data set that you provide. The subsystem definition contains the information DCCTL uses to communicate with the other subsystems.

Related reading:

- For more information about accessing external subsystems from IMS, see Chapter 30, "Accessing external systems from within IMS," on page 469.
- For more information about RRSAF, see *DB2 for z/OS Application Programming and SQL Guide*.

DCCTL handles transaction management for online IMS applications that need to access external subsystems.

Application calls supported by DCCTL

Application program calls passed to DCCTL receive an AD status code if the call function is not supported or if a database PCB is passed as part of the call list.

The following data communication calls are available to application programs in a DCCTL environment:

- AUTH
- CHNG

- CMD
- GCMD
- GN
- GU
- ICAL
- ISRT
- PURG
- SETO

The following system service calls are available to application programs in a DCCTL environment:

- APSB
- CHKP
- DPSB
- INIT
- INQY
- LOG
- ROLB
- ROLL
- ROLS
- SETS
- SETU
- SYNC
- XRST

DCCTL compared to DB/DC

DCCTL and the TM part of a DB/DC environment are very similar.

Similarities include:

- Control region and dependent region initialization and termination.
- System definition and generation.
- Restart.
- Stage 1 input without removing the database definitions. You will need to make some changes to define a DCCTL system. See *IMS Version 15.2 System Definition* for more information about analyzing macros for system definition.
- Diagnosis.

Exit routines

You do not need to change your IMS exit routines or your existing IMS application programs that access other subsystem resources in a DCCTL environment. However, application programs that contain a mixture of calls that access other subsystems and IMS databases require changes. All DL/I calls that use a database PCB receive status code AD.

Automated operator transactions

Automated operator transactions are started the same way as IMS transactions are started. Automated operator transactions run as IMS application programs with the authority to issue a subset of DCCTL commands using DL/I calls.

DB batch environment

The batch environment consists of a batch region—one address space—where an application program and DL/I routines reside. The batch job that runs here is initiated with JCL, like any operating-system job.

The following figure represents a batch environment in which the batch job is submitted through a TSO terminal, and an application program is run in order to read from an update file, write to a database, and produce a report. For example, an inventory application might be run—one that reads sales records (inventory reductions) and supply records (inventory increases), updates a database accordingly, and prints an inventory sales report.

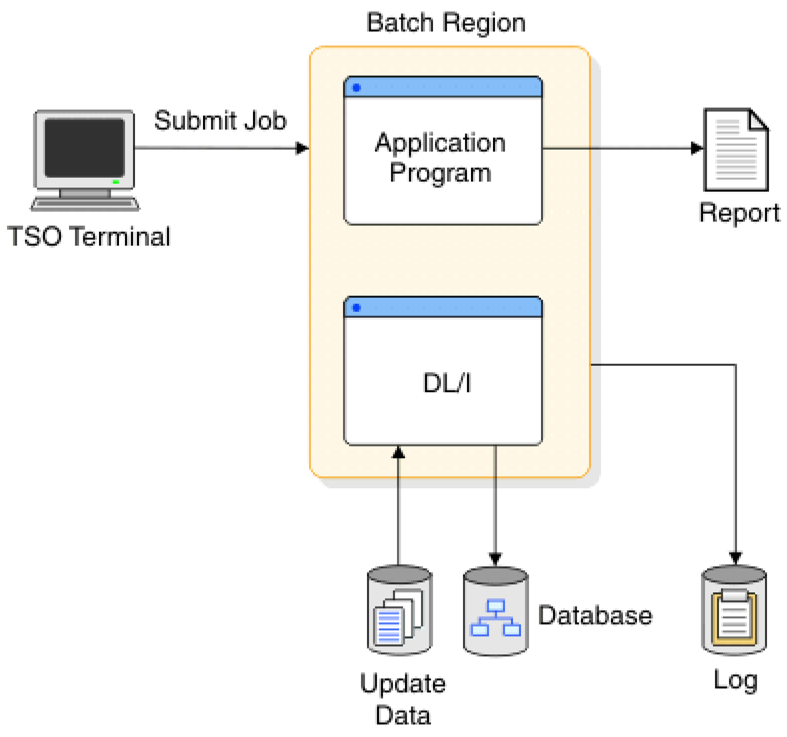


Figure 5. Example of a DB batch environment

TM batch environment

IMS TM supports a batch region for running application programs. IMS applications cannot use the ESAF to issue SQL calls in batch. This support is provided by an external subsystem.

You can connect Db2 for z/OS in an IMS TM batch environment in one of two ways. You can use the SSM parameter on the TM batch-region execution JCL and specify the actual name of the batch program on the MBR parameter. Alternatively, you can code the DDITV02 DD statement on the batch-region execution JCL and specify the name of the Db2 for z/OS module, DSNMTV01, on the MBR parameter.

Valid TM batch region types are DBB, DLI, or UPB. All other region types are not applicable to the TM batch environment.

You specify generated program specification blocks (GPSBs) for a TM batch environment using the PSB parameter in the DBBBATCH and DLIBATCH procedures.

The IEFRDER log DD statement is required in order to enable log synchronization with other external subsystems, such as Db2 for z/OS.

Related reading:

- If your external subsystem is Db2 for z/OS, see the *DB2 for z/OS Application Programming and SQL Guide* for a description of the steps required to allow batch programs to issue SQL calls.
- For additional options or requirements, see *IMS Version 15.2 System Definition*.

IMSplex overview

An IMSplex is made up of IMS and z/OS components that work together.

Connected systems such as Multiple Systems Coupling and replicated systems such as XRF are used in an IMSplex.

An IMSplex can be defined as:

- A set of IMS systems working together to share resources or message queues (or both) and workload.
- A single IMS system using the Common Service Layer (CSL) without the Resource Manager (RM) to have a single point of control. This configuration allows you to use IMS type-2 commands, which can be issued only through the Operations Manager (OM) APIs by an automated operator program (AOP). Optionally, IMSplexes of this type can also include the CSL Open Database Manager (ODBM).
- A single IMS system using the CSL with an RM.

Contrast the description of an IMSplex with a description of a Parallel Sysplex® (hereafter referred to as sysplex). A sysplex is multiple z/OS images working together, connected by a coupling facility. One or more IMSplex systems can be defined on one or more z/OS images; however, you do not have to have an IMS instance on every z/OS image in the sysplex.

The concept of data sharing is important in an IMSplex, specifically in an IMSplex that shares resources and workload.

This allows IMS batch jobs, online applications, or both, to run anywhere in the IMSplex. These batch jobs and applications can also access all of the data in the shared IMS databases. You can distribute your IMS workload according to your business requirements. After data sharing is enabled, you must create a plan for distributing batch and transaction workload across the IMSplex and then implement that plan. One technique is using shared queues (SQ). With SQ, any transaction can be entered from the network to any IMS system in the IMSplex and be executed by any IMS system in a shared-queues group. If one IMS system is unavailable, a different system can handle the work.

If you have multiple IMS systems sharing resources or message queues, administration and operation can be simplified by using the IMS Common Service Layer (CSL), which reduces complexity by providing a single-image perspective. With CSL, you can manage multiple IMS systems as if they were only one system. For example, instead of entering type-1 commands from multiple MTOs or a single point of control (SPOC) to perform local online changes, you can enter a type-2 command from one SPOC to perform global online change. The following figures contrast the relative complexity of managing an IMSplex with and without a CSL.

Managing an IMSplex using CSL provides:

- Improved systems management
- A single-system image
- Simpler operations through a single point of control
- Shared resources across IMS systems

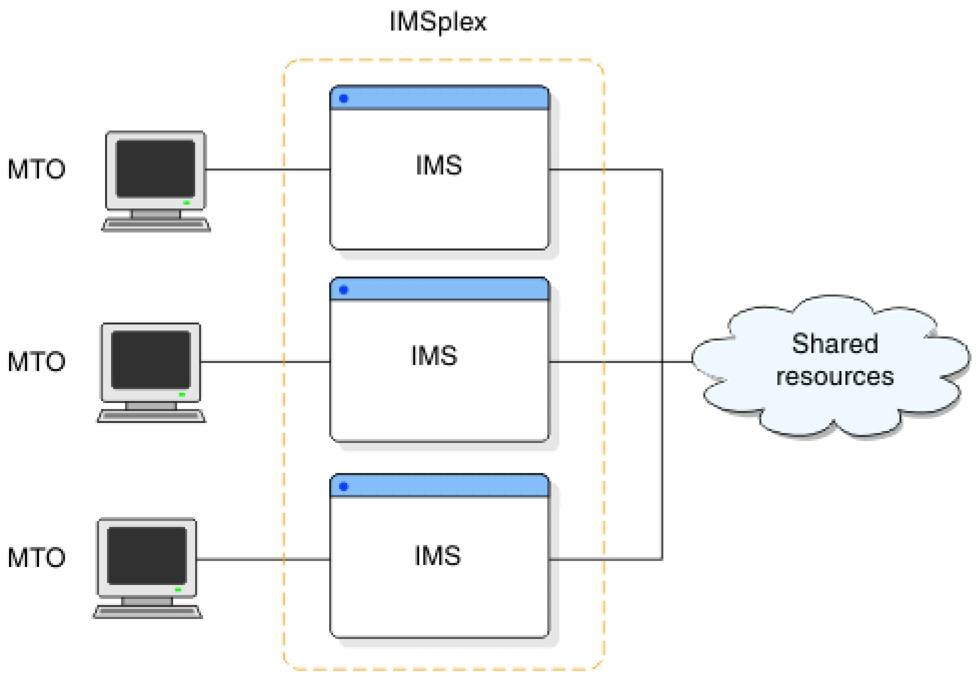


Figure 6. Complex management of an IMSplex without CSL

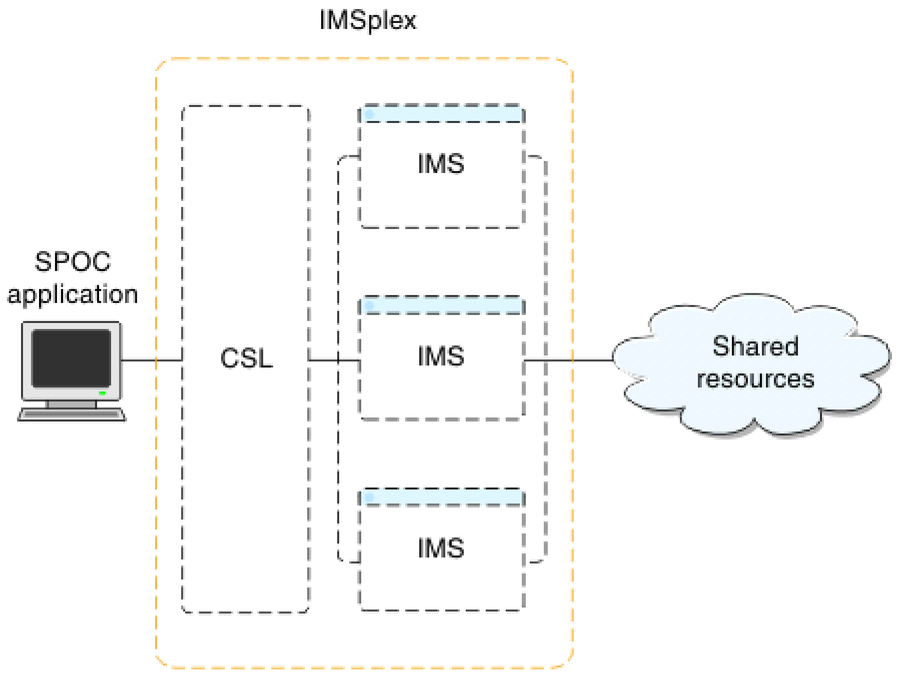


Figure 7. Simplified management of an IMSplex with CSL

When you manage an IMSplex with the CSL, the resulting environment is comprised of multiple *IMSplex components*. An IMSplex component either manages resources, manages operations, or facilitates communication among IMSplex components. The following are IMSplex components:

- Operations Manager (OM)
- Resource Manager (RM)
- Open Database Manager (ODBM)
- Structured Call Interface (SCI)
- IMS Connect
- Database Recovery Control (DBRC)

- Common Queue Server (CQS)
- Repository Server (RS)
- TSO Single Point of Control (SPOC)
- REXX SPOC API for automation programs
- Online control regions

Any program that registers with the SCI is considered an IMSplex component. Users and vendors can also write programs that register with SCI; these programs are also considered IMSplex components. Note that batch regions and dependent regions (MPR, BMP, and IFP) are not IMSplex components. Although DBRC code running within a batch region registers with SCI, the batch region itself is not a component. When an IMSplex component is initialized and joins the IMSplex, it becomes an *IMSplex member*.

Related concepts

"IMS environments and configurations" on page 4

IMS contains two major components that can be used together or separately: the IMS Database Manager (DB) and the IMS Transaction Manager (TM). When you use them together, they constitute the DB/DC environment.

z/OS components and system services used in an IMSplex

The z/OS components that are used in an IMSplex include the Coupling Facility and its cache, list, and lock structures.

An IMSplex takes advantage of several z/OS system services:

z/OS cross-system coupling facility

An IMSplex uses XCF for communication services. XCF services allow authorized programs on one system to communicate with programs on the same system or on other systems.

automatic restart management (ARM)

An IMSplex uses ARM for recovery services. ARM restarts subsystems, all the elements of a workload on another z/OS image after a z/OS failure, and a failed z/OS image.

cross-system extended services (XES)

An IMSplex uses XES for data sharing services and system-managed rebuild support.

IMS components and system services that are part of an IMSplex

An IMSplex can include several different IMS components: IMS subsystems, DBRC and RECON data sets, IMS Connect, and TSO SPOC. IMS uses IRLM for lock services.

In addition, a number of IMS address spaces provide system services in an IMSplex. These address spaces are all built using the Base Primitive Environment (BPE), which provides system services such as tracing, message formatting, parsing, storage management, sub-dispatching, and serialization.

The BPE system services are used by the Common Queue Server (CQS), and the CSL's Open Database Manager (ODBM), Operations Manager (OM), Resource Manager (RM), Structured Call Interface (SCI), Repository Server (RS), and, optionally, the Database Recovery Control facility (DBRC), each of which contributes other system services:

Common Queue Server

Receives, maintains, and distributes data objects from shared queues.

Common Service Layer

Includes the following components:

Open Database Manager

Provides access to IMS databases managed by IMS DB systems in DBCTL and DB/TM environments within an IMSplex. ODBM supports TCP/IP clients through IMS Connect and application servers running application programs that use the IMS ODBA interface, such as Db2 for z/OS or WebSphere® Application Server for z/OS.

Operations Manager
Routes commands, consolidates command responses, provides an API for command automation, and provides user exits for customization and security.

Resource Manager
Maintains global resource information, ensures resource consistency, and coordinates IMSplex-wide processes such as global online change.

The RM also provides the interface for and access to the IMSRSC repository.

Structured Call Interface
Routes messages, registers and deregisters members of an IMSplex, provides security authentication of IMSplex members, and more.

Repository Server
Manages the IMSRSC repository, which is a data storage facility that can be used to store resource and descriptor definitions.

IMS also uses IRLM for lock services.

Related concepts
Overview of the IMSRSC repository (System Definition)

Multiple Systems Coupling in an IMSplex

The Multiple Systems Coupling (MSC) function allows you to distribute and balance your workload across multiple systems.

MSC distributes transaction and message traffic between IMS subsystems. MSC extends its communication with IMS and scheduling capabilities, enabling you to perceive one virtual IMS system, when in fact transactions are being routed among several IMS subsystems.

An MSC network can coexist with an IMSplex using shared queues. A temporary MSC and IMSplex coexistence can be useful if you are migrating from an MSC network to an IMSplex configuration. Coexistence can also be permanent, such as when an MSC link connects an IMSplex to an IMS system outside of the IMSplex.

In both coexistence examples, you must consider the proper routing and processing of your transaction messages across both the MSC and IMSplex environments, because each environment uses a different routing method. Generally, MSC networks route transactions to specific IMS systems using SYSIDs. IMSplex systems with shared queues route transactions by making them available on the shared queue to any IMS system that registers an interest in the transactions. When an MSC network and an IMSplex with shared queues coexist, both of these methods of routing can apply to transactions.

For more information on the coexistence of MSC systems and IMSplex systems, see *IMS Version 15.2 Communications and Connections*.

Recovering systems in an IMSplex

The recovery of an IMS system can be managed by the Extended Recovery Facility (XRF). An IMS Fast Database Recovery region (FDBR) can also be used for recovery purposes.

XRF
Performs takeover on a system-by-system basis; XRF relies on the same physical databases and many of the same physical resources as the active IMS system.

XRF can be used in an IMSplex in the same way that it is used on a local system. The XRF alternate system needs to have a unique IMSID in the IMSplex, be defined in the IMSplex and have access to the CSL. An SCI must reside on the z/OS image where the XRF alternate resides. You cannot use XRF to recover an entire IMSplex, but you can use XRF to recover an individual IMS in an IMSplex.

An IMS Database Control (DBCTL) warm standby region and an IMS Fast Database Recovery region (FDBR) can also be used for recovery purposes.

FDBR
An FDBR region monitors an IMS subsystem and can automatically recover database resources (shared databases and areas) if that IMS subsystem fails.

DBCTL standby
While one DBCTL subsystem is active, you can start another DBCTL subsystem as a standby alternate. This alternate subsystem does not track the active subsystem (as it would in an XRF complex), but is a fully initialized IMS DBCTL subsystem that is waiting for a restart command.

An IMSplex can include XRF, FDBR, and DBCTL standby regions.

Related reading:

- For more information on XRF, see Part 4, "IMS system recovery," on page 567.
- For more information on FDBR, see *IMS Version 15.2 Operations and Automation*.

Defining and tailoring an IMSplex

You must establish the procedures required to start an IMSplex, and specify the appropriate parameters in the DFSCGxxx member or the CSL section of the DFSDFxxx member of the IMS PROCLIB data set.

The following topics describe how to initialize each of the necessary IMSplex components for an IMSplex with CSL.

These topics briefly describes the components that you must define in an IMSplex.

Related reading:

- For detailed information about IMSplex system definition, see *IMS Version 15.2 System Definition*.
- For more information about defining the IMS control region, the system definition macros, and the PROCLIB member data sets, see *IMS Version 15.2 System Definition*.

Related concepts
"Global TM resource management" on page 44
Using RM and a resource structure improves your ability to manage Transaction Manager resources in an IMSplex. IMS enforces name uniqueness for LTERMs (VTAM), nodes (VTAM single-session), user IDs (if the installation requests single-signon enforcement), and users.

Related tasks
"Defining a simplified IMSplex for the type-2 command environment" on page 22
To define a simplified IMSplex without RM either specify RMENV=N in the DFSCGxxx member of the IMS PROCLIB data set or specify RMENV=N in the CSL section of the DFSDFxxx PROCLIB member while defining an IMSplex.

Defining CQS

To define CQS, create the CQS and BPE PROCLIB member data sets and define the execution data sets.

The following steps summarize defining CQS.

1. Create a coupling facility resource management (CFRM) policy for the MSGQ, EMHQ and RSRC structures.
2. Define the applicable z/OS policies.
3. Activate the CFRM policy.
4. Create the CQS and BPE PROCLIB member data sets.
5. Define all CQS execution data sets.
6. Customize your CQS environment.
7. Authorize connections to CQS structures.
8. Update the z/OS program properties table.

Defining the IMS control region

To initialize IMS to function in an IMSplex, you must define the IMS control region to communicate with OM for command processing and RM for resource validation, status updates, and global online change.

Steps in the definition process include:

- Uniquely identifying each IMS system in the IMSplex by using the IMSID= keyword in the system definition macro
- Identifying the IMSplex to each IMS system by using the IMSPLEX= keyword in the DFSCGxxx PROCLIB member data set, or the CSL section of the DFSDFxxx PROCLIB member data set, of each IMS system

The PROCLIB member data sets that must be modified to correctly define the IMS system and the IMSplex include:

- DFSCGxxx
- DFSDFxxx
- DFSPBxxx
- DFSSQxxx
- DFSVSMxx

Defining ODBM

Use the CSLDIxxx member of the IMS.PROCLIB data set to specify parameters for initializing ODBM. Certain CSLDIxxx parameters can be overridden with ODBM execution parameters.

The IMSplex name that you specify in CSLDIxxx by using the IMSPLEX(NAME=*plxnm*) parameter must be the same as the IMSplex names specified in the CSLOIxxx, CSLRIxxx, CSLSIxxx, DFSDFxxx, and DFSCGxxx PROCLIB member data sets.

Use the CSLDCxxx PROCLIB member to specify the parameters that configure the ODBM connections to IMS data stores.

Defining OM

Use the CSLOIxxx member of the IMS.PROCLIB data set to specify parameters for defining OM. Certain CSLOIxxx parameters can be overridden with OM execution parameters.

The IMSplex name that you specify in CSLOIxxx by using the IMSPLEX(NAME=*plxnm*) parameter must be the same as the IMSplex names specified in the CSLDIxxx, CSLRIxxx, CSLSIxxx, DFSDFxxx, and DFSCGxxx PROCLIB members.

Defining RM

Use the CSLRIxxx member of the IMS.PROCLIB data set to specify parameters for defining RM. Certain CSLRIxxx parameters can be overridden with RM execution parameters.

The IMSplex name that you specify in CSLRIxxx by using the IMSPLEX(NAME=*plxnm*) parameter must be the same as the IMSplex names specified in the CSLDIxxx, CSLOIxxx, CSLSIxxx, DFSDFxxx, and DFSCGxxx PROCLIB members.

Defining SCI

Use the CSLSIxxx member of the IMS.PROCLIB data set to specify parameters for defining the SCI address space. Certain CSLSIxxx parameters can be overridden with SCI execution parameters.

The IMSplex name that you specify in CSLSIxxx by using the IMSPLEX(NAME=*plxnm*) parameter must be the same as the IMSplex names specified in the CSLDIxxx, CSLOIxxx, CSLRIxxx, DFSDFxxx, and DFSCGxxx PROCLIB members.

Defining a simplified IMSplex for the type-2 command environment

To define a simplified IMSplex without RM either specify RMENV=N in the DFSCGxxx member of the IMS PROCLIB data set or specify RMENV=N in the CSL section of the DFSDFxxx PROCLIB member while defining an IMSplex.

To bring up a CSL without RM, RMENV=N must be specified in the SCIPROC parameter in the DFSCGxxx or DFSDFxxx members of the IMS PROCLIB data set.

Related concepts

"Global online change" on page 42
The global online change function changes resources online for all IMS systems in an IMSplex.

"A simplified CSL configuration" on page 35
If your IMS configuration does not require RM services, you can still use OM and SCI to take advantage of type-2 IMS commands.

Related tasks

"Defining and tailoring an IMSplex" on page 20
You must establish the procedures required to start an IMSplex, and specify the appropriate parameters in the DFSCGxxx member or the CSL section of the DFSDFxxx member of the IMS PROCLIB data set.

Starting the CSL OM (Operations and Automation)

Starting the CSL SCI (Operations and Automation)

Related reference

"CSL configuration examples" on page 46
Typically, when an IMS control region (DL/I, DBRC, dependent regions) requires the use of the CSL, SCI, OM, and RM are all required. However, different configurations are possible for the CSL in an IMSplex. For example, you might include ODBM or omit RM in your CSL configuration.

DFSCGxxx member of the IMS PROCLIB data set (System Definition)

DFSDFxxx member of the IMS PROCLIB data set (System Definition)

Defining the IMSplex name for RECON loss notification or DBRC parallel RECON access

To enable automatic RECON loss notification or parallel RECON access, you must set an IMSplex name for the RECON data set using the **CHANGE.RECON IMSPLEX** command.

If automatic RECON loss notification is the only use of the CSL in an IMSplex, SCI is the only CSL address space that is required.

All DBRC instances accessing this RECON must specify the IMSplex name for the RECON data set by using the DBRC SCI Registration exit routine (DSPSCIX0) or the EXEC statement keyword parameter IMSPLEX=. To use SCI, DBRC must provide the name of the IMSplex in which the DBRC instance is running.

The DBRC SCI Registration exit routine, DSPSCIX0, is an exit routine that DBRC calls before it registers with SCI. If you have enabled parallel RECON access, this exit routine must supply the IMSplex name needed for SCI registration. If the exit routine is not used, DBRC continues as if the sample version of the exit routine was being used, which means that DBRC does not register with SCI. DSPSCIX0 must be found in an authorized library or in LINKLST. In a TSO environment, the library that contains DSPSCIX0 must be a TSO TASKLIB library.

If more than one set of RECONs is defined in the IMSplex, a *group_id* also needs to be defined either in the DSPSCIX0 exit routine or in the RECON data set. The EXEC statement keyword parameter for the *group_id* is DBRCGRP=xxxx. To define a *group_id* in the RECON data set, use the **CHANGE.RECON IMSPLEX**(*imsplex_name*,*group_id*) command.

Defining global online change

To enable global online change in an IMSplex that is defined with a CSL, the DFSCGxxx member or the CSL section of the DFSDFxxx member of the IMS PROCLIB data set must be specified.

The DFSCGxxx PROCLIB member and the CSL section of the DFSDFxxx PROCLIB member include parameters related to SCI, OM, and RM, which are common to all IMS systems in the IMSplex (except those systems defined without an RM). The OLC= parameter must specify GLOBAL for global online change or LOCAL for local online change.

OLC=GLOBAL means that the online change is coordinated across the IMSplex. Global online change is prepared and committed by the **INITIATE OLC** command. If global online change is enabled and a resource structure is defined, the MODBLKS, FORMAT, and ACBLIB data sets must be consistent across the IMSplex, unless resource consistency checking is omitted by the NORSCCC keyword.

OLC=LOCAL means the online change applies locally to each IMS. Local online change is prepared and committed by the **/MODIFY PREPARE** and **/MODIFY COMMIT** commands on each local IMS. The local online change must be manually coordinated across an IMSplex.

Specify the OLCSTAT data set using the OLCSTAT= parameter of the DFSCGxxx PROCLIB member data set or the CSL section of the DFSDFxxx PROCLIB member data set if global online change is enabled in an IMSplex with CSL. Run the Global Online Change utility (DFSUOLC0) to:

- Initialize the OLCSTAT data set before IMS cold starts for the first time.
- Recreate the OLCSTAT data set if an error makes the OLCSTAT data set unusable.
- Unlock the OLCSTAT data set.

OLCSTAT data set
The global OLCSTAT data set, which z/OS assigns to IMS, is a basic sequential access method (BSAM) data set that supports one record of variable size. You can allocate and deallocate the OLCSTAT data set dynamically. Unlike the MODSTAT data set, the OLCSTAT data set does not need to be continuously available when your IMS system is running.

OLCSTAT data set attributes

Before the OLCSTAT can be allocated, you must define the data set attributes. The following table shows the recommended attributes for the OLCSTAT data set, to allow up to 65 IMS records.

Table 2. Recommended attributes for the OLCSTAT data set

Data set attribute	Value
DSORG	Sequential
RECFM	V
LRECL	5204
BLKSIZE	5208

If you do not define these attributes, when IMS attempts to allocate the OLCSTAT data set, that allocation fails.

After you define these attributes, you must name the data set by using the OLCSTAT parameter in the DFSCGxxx PROCLIB data set member.

OLCSTAT data set format

The OLCSTAT data set consists of a header and any number of IMS records, with the following formats:

- OLCSTAT format for VERS=1

```
hlenhverivermstr  OLCINP modifyid MODBLKSz x IMSACBb x FORMATc x
ilenrveriverims   prepare_timestamp commit_timestamp
```

- OLCSTAT format for VERS=2

```
hlenhverivermstr  OLCINP modifyid MODBLKSz x IMSACBb x FORMATc x
MOLCstate MOLCtoken MOLCid
ilenrveriverims   prepare_timestamp commit_timestamp
```

The header information includes the master of the last online change phase, an online change in progress lock, the modify ID, the active MODBLKS DD name, the active IMSACB DD name, the active FORMAT DD name, and the libraries that were changed with the last online change. The header is initialized by the Global Online Change utility (DFSUOLC0).

hlen
The OLCSTAT data set header record length. The header record is the first line.

hver
The OLCSTAT data set header version.

iver
IMS version.

mstr
IMS ID of the IMS system that was the master of the last online change phase (prepare, commit, or abort.

ilen
The OLCSTAT data set IMS record length. The OLCSTAT data set contains zero, one or more OLCSTAT data set IMS records.

rver
The OLCSTAT data set IMS record version.

iver
IMS version.

ims
IMS ID.

OLCSTAT data set header

The following example shows a sample OLCSTAT data set header for Version 2.

```
RECORD SEQUENCE NUMBER - 1
000000  00000080 00000002 F1F0F1F0 40404040   40404040 40404040 40404040
000020  F0F0F0F1 40D4D6C4 C2D3D2E2 C140D540   C9D4E2C1 C3C2C140 D540C6D6
000040  C140D540 D4D6D3C3 C3D4D7E3 40BF88D7   940BE91E 82402006 283F2321
000060  028D4040 40404040 40404040 40404040   40404040 40404040 40404E40
000080  00000050 00000001 F1F0F1F0 40404040   C9D4E2F1 40F2F0F0 F6F2F8F3
0000A0  F0F5F2F7 4060F0F7 7AF0F040 F2F0F0F6   F2F8F340 F2F3F2F1 F0F3F440
0000C0  F0F04040 40404040 40404040 4040404E


F0F0F0F0   *........1010               0000*
D9D4C1E3   *0001 MODBLKSA N IMSACBA N FORMAT*
03465440   *A N MOLCCMPT ..P..Z.. ......... *
40404040   *..                        +     *
40F2F3F2   *...&....1010    IMS1 2006283 232*
60F0F77A   *0527 -07:00 2006283 2321034 -07:*
           *00            +                *


IDC0005I NUMBER OF RECORDS PROCESSED WAS 1
IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0
IDCAMS  SYSTEM SERVICES                                           TIME: 1
```

Each IMS record contains information about an IMS that is current with the online change libraries. The IMS either participated in the last online change, or cold started since the last online change. If no IMS has cold started yet, the OLCSTAT data set contains no IMS records.

You must also run the Online Change Copy utility (DFSUOCU0) as one step in the process of preparing an IMS or an IMSplex for an online change. The DFSUOCU0 utility copies a source library with your new definitions to a target library. In an IMSplex where IMS systems are not cloned and the libraries are not shared, you might need to run the DFSUOCU0 utility on every IMS system in the IMSplex. In an IMSplex where IMS systems are cloned and the libraries are shared, you might need to run the DFSUOCU0 utility only once on one IMS.SDFSRESL data set at the highest IMS level.

The DFSUOCU0 utility can copy the contents of the staging library to a target library that is specified as a parameter. The utility supports a target library value of G, which means that the library target is the active library determined by the utility, using the OLCSTAT data set. The target is the library that is not currently in use by the IMS systems in the IMSplex. If the new target library G is specified, the DFSUOCU0 utility reads the OLCSTAT data set to determine the target library. The global online change commands, **`INITIATE OLC PHASE(PREPARE)`**, followed by **`INITIATE OLC PHASE(COMMIT)`**, change the inactive library to an active library.

Related reference

Global Online Change utility (DFSUOLC0) (System Utilities)

Online Change Copy utility (DFSUOCU0) (System Utilities)

Dynamic resource definition in an IMSplex

An IMSplex can include IMS systems that have DRD enabled and others that do not have DRD enabled. This kind of IMSplex environment is referred to as a mixed-DRD IMSplex.

If a command that contains the MODBLKS parameter is issued in a mixed-DRD IMSplex, the MODBLKS parameter is ignored.

Two scenarios are provided to demonstrate how to add application programs in a mixed-DRD IMSplex.

Scenario 1: add an application program to a mixed-DRD IMSplex

In this IMSplex, IMS1 has DRD enabled, and IMS2 and IMS3 do not have DRD enabled.

1. Perform an ACBGEN for the PSB into the staging ACBLIB
2. Perform a MODBLKS generation to define the APPLCTN program in the staging MODBLKS data set for the IMS2 and IMS3 systems.
3. Create the application program dynamically by issuing the **`CREATE PGM`** command on the IMS1 or, if the program is to be scheduled in a BMP or JBP dependent region, by using the Program Creation user exit (PGMCREAT).
4. Copy the staging MODBLKS data set and ACBLIB to the inactive MODBLKS data set and ACBLIB.
5. Perform global online change for the MODBLKS data set and ACBLIB by issuing the following commands:

   ```
   INITIATE OLC PHASE(PREPARE) TYPE(ACBLIB,MODBLKS)
   INITIATE OLC PHASE(COMMIT)
   ```

The MODBLKS keyword is ignored by the IMS1 system; however, the IMSplex synchronizes and adds the program with one online change instance. The command that includes the MODBLKS keyword is not rejected; this would require another online change for the ACBLIB on the IMS1 system.

Scenario 2: add an application program and its MFS format in a mixed-DRD IMSplex that includes DBCTL IMS systems

IMS1 and DBCTL1 have DRD enabled. IMS2 and DBCTL2 do not have DRD enabled.

1. Perform an ACBGEN for the PSB into the staging ACBLIB.
2. Perform a MODBLKS generation to define the APPLCTN program in the staging MODBLKS data set for the IMS2 and DBCTL2 systems.
3. Run the MFS utility to create an MFS format in the staging FMTLIB for the IMS1 and IMS2 systems.
4. Create the application program dynamically by issuing the **`CREATE PGM`** command on the IMS1 and DBCTL1 systems or, if the program is to be scheduled in a BMP or JBP dependent region, by using the Program Creation user exit (PGMCREAT).
5. Copy the staging MODBLKS data set and ACBLIB to the inactive MODBLKS data set and ACBLIB.
6. Perform global online change for MODBLKS, FMTLIB, ACBLIB by issuing the following commands:

```
INITIATE OLC PHASE(PREPARE) TYPE(ACBLIB,FMTLIB,MODBLKS)
INITIATE OLC PHASE(COMMIT)
```

The MODBLKS keyword is ignored by the IMS1 and DBCTL1 systems. The FMTLIB keyword is ignored by the DBCTL1 and DBCTL2 systems. The ACBLIB keyword is processed on all four systems. This synchronizes the IMSplex and successfully adds the application program and its MFS format with one online change instance. The command that includes the MODBLKS keyword is not rejected; this would require additional online changes on IMS2, DBCTL1, and DBCTL2.

Related reading: For a detailed description of what you can do with DRD, see *IMS Version 15.2 System Definition*.

Related reference

CREATE PGM command (Commands)

PGMCREAT user exit routine type (Exit Routines)

Establishing IMSplex security

The way you protect the IMS online system does not change for an IMS system in an IMSplex. The resources that can be protected and the facilities available to protect them are similar. The types of security issues that exist for a standalone IMS system also exist for an IMS system within an IMSplex.

The basic decisions you must make when implementing security for IMS include:

- Determine what resources need to be protected.
- Determine which users need access to the resources.
- Determine the level of access to a resource that the users need.

In addition to the security checks made for standalone IMS systems, an IMSplex with a CSL performs additional security checks.

Chapter 2. Concepts for the system administrator

This topic presents concepts that are central to controlling the resources of an IMS. The other topics included in this information assume a thorough understanding of these concepts.

System support for application programs

Before you can run an application program under IMS, control blocks must be defined, generated, and placed into the following libraries: IMS.DBDLIB, IMS.PSBLIB, and IMS.ACBLIB.

Before executing an application program under IMS, you must:

- Describe that program and its use of logical terminals and logical data structures through a program specification block (PSB) generation, using the PSB Generation utility. The PSBs are maintained in one or more IMS system libraries called a PSB library.
- Create a database descriptor block (DBD) to have access to an IMS database. The DBD is assembled into a system library called a DBD library.
- Combine and expand the PSB and DBD control blocks into an internal format called application control blocks (ACBs). The Application Control Blocks Maintenance utility is used to create the ACBs and they are placed in an ACB library.

Application programmers and database administrators need to know the naming conventions for the PSB, DBD, and ACB library data sets.

The system administrator must decide:

- Whether to have the ACB library allocated by JCL or dynamically (using the DFSMDA macro)

  To begin using the DFSMDA macro for dynamic allocation of ACB library data sets, complete the following steps:

  1. Create DFSMDA members for the ACBLIBA and ACBLIBB data sets. DFSMDA members can be placed in either the data set specified in the IMS STEPLIB concatenation or in the IMSDALIB DD statement.
  2. Remove the IMSACBA and IMSACBB DD statements from the IMS and DL/I JCL procedures.
  3. Stop IMS and then restart it with the DFSMDA members.

- Whether to have the ACBs loaded into 64-bit storage

  At execution time, the ACBs needed by an application program must be in the 31-bit non-resident ACB storage pool. At scheduling time, IMS first searches the non-resident storage pools to see if the ACBs are already there. If they are not in the pool, IMS can load them into the 31-bit non-resident ACB storage pool from either the ACB library or from a 64-bit storage pool.

  To enable the use of a 64-bit storage pool for ACBs, complete the following steps:

  1. Calculate how much storage to allocate for the 64-bit storage pool by multiplying the total number of non-resident ACB members in the ACB library times the size of these members.
  2. Specify the size of the 64-bit ACB storage pool (in gigabytes) on the ACBIN64 parameter in the DATABASE section of the DFSDFxxx PROCLIB member. For most installations, specifying one or two gigabytes for the 64-bit ACB storage pool should be sufficient.
  3. Stop IMS and then restart it with the ACBIN64 parameter.

  After enabling the use of a 64-bit ACB storage pool, IMS populates this pool the first time an application program that need the ACBs is scheduled. Thereafter, when an application program is scheduled that needs these ACBs, IMS reads them from the 64-bit storage pool instead of from the ACB library.

  If you need to remove ACBs from both the 31-bit non-resident storage pool and the 64-bit storage pool, use the **`DELETE DB`** and **`DELETE PGM`** commands. You cannot remove ACBs from the pools using any command.

Related reading: For an expanded overview of generating PSBs, DBDs, and ACBs, see *IMS Version 15.2 Database Administration*.

Dynamic allocation with IMS

If data sets are specified with JCL in a control region procedure, they are initially allocated when the control region starts up. Some data sets can also be dynamically allocated.

You can specify that the following data sets be dynamically allocated when needed and deallocated when no longer in use.

Using the IMS macro DFSMDA, you declare those data sets that are subject to dynamic allocation and deallocation.

- Database data sets can be dynamically allocated explicitly with the **/START** command or implicitly when an application program is scheduled. Database data sets can be deallocated with the **/DBRECOVERY** command.
- For DEDB area data sets, an implicit allocation occurs at first access by an application program; the **/STOP** command also deallocates the data set.
- An IMS Monitor data set can be dynamically allocated at the time the IMS Monitor is started with the **/TRACE SET ON** command and deallocated by the **/TRACE SET OFF** command.
- RECON data sets, online log data sets (OLDSs), write-ahead data sets (WADSs), and system log data sets (SLDSs) that are required as input to restart can be dynamically allocated.
- ACB libraries can be dynamically allocated.
- For High Availability Large Databases (HALDBs), dynamic allocation allocates only the DBRC registered data sets. Allocation does not look for or process any DFSMDA members. The DD names allocated for HALDBs contain the letters A through J, X, or L, suffixed to the 7-byte HALDB partition name.

  If an allocation already exists with a DD name that matches the HALDB partition DD name generated, the data set name is compared to the DBRC registered data set name in online IMS environments. If the data set names do not match, an allocation failure occurs.

All data sets using dynamic allocation must be cataloged, except an IMS Monitor data set, which must not be cataloged. A data set that is initially allocated with JCL can be dynamically deallocated and reallocated during the execution of the control region.

Related reading: For more information on the IMS macro DFSMDA, see *IMS Version 15.2 System Definition*.

Base Primitive Environment overview

The IMS Base Primitive Environment (BPE) is a common system service base upon which many other IMS components are built. BPE provides services such as tracing, message formatting, parsing, storage management, sub-dispatching, and serialization.

The following IMS components use BPE:

- Common Queue Server (CQS)
- IMS Connect
- Open Database Manager (ODBM)
- Operations Manager (OM)
- Resource Manager (RM)
- Repository Server (RS)
- Structured Call Interface (SCI)

You can optionally have Database Recovery Control facility (DBRC) use BPE.

When an IMS component that uses BPE is started, the component loads a copy of the BPE service modules into its address space from the IMS program libraries. The IMS component's modules are specific to that component; however, the BPE service modules are common across the various address spaces. The base system service functions are therefore identical in every address space that uses BPE. The following figure shows the relationship of BPE, IMS components, and IMS program libraries.

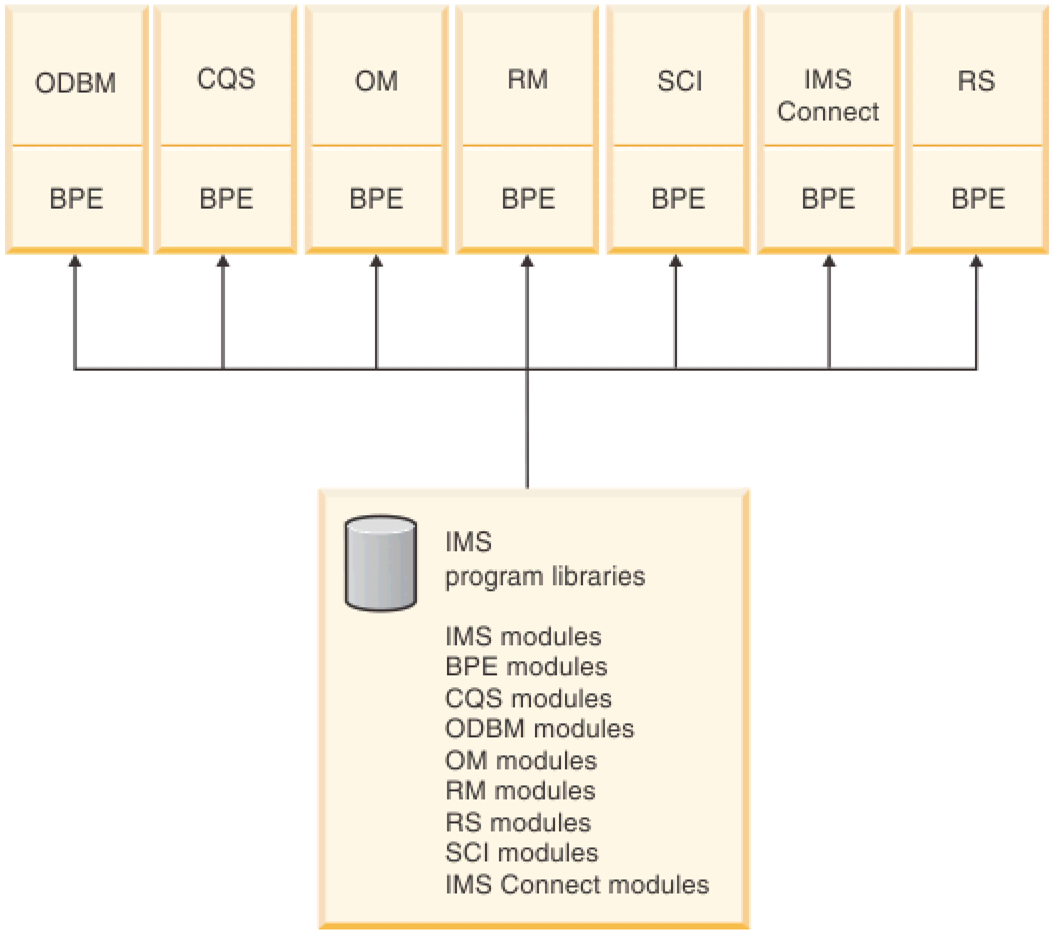


Figure 8. BPE and IMS components

Most of the time, BPE is a hidden layer in an IMS component address space. However, you can use the following external interfaces to BPE:

Configuration
: You can configure certain attributes about an address space that uses BPE at startup by using statements in BPE PROCLIB member data sets. For example, you can set the default level and size for BPE-managed trace tables. Information about defining BPE with the BPE PROCLIB member data set is in *IMS Version 15.2 System Definition*.

Commands
: You can use the small set of commands that BPE provides to operate on BPE-managed resources. For example, you can display and change attributes of BPE-managed user exit routines and trace tables. Information about BPE commands is in *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*.

User exit routines
: You can customize the operation of an IMS component address space by writing user exit routines. Components running with BPE can use the BPE user exit routine service to call component-specific user-supplied exit routines. BPE also has user exit routines of its own. User exit routines that are called through BPE receive control with a standard BPE user exit routine interface and are allowed to use BPE user exit routine callable services. Information about BPE user exits is in *IMS Version 15.2 Exit Routines*.

Messages and Abends

BPE has its own messages and abend codes. *IMS Version 15.2 Messages and Codes, Volume 4: IMS Component Codes* includes information about BPE abend and service return codes and BPE messages.

Tracing BPE components

You can enable tracing for BPE components (BPE, CQS, DBRC, IMS Connect, ODBM, OM, RM, RS, or SCI), and control whether the trace information is written either to memory, or to both memory and an external trace data set.

This topic describes the tasks associated with enabling tracing for BPE components:

Enabling BPE tracing

To enable BPE tracing so that it is always on for an IMS address space, you must define or modify the TRCLEV statements in the BPE configuration parameter member of the IMS.PROCLIB data set (specified by BPECFG=). You can also dynamically start BPE tracing by using the **UPDATE TRACETABLE** command.

The BPE configuration parameter member defines the BPE execution environment settings for the address space being started. BPE trace records can be written to internal (memory only) trace tables and to external data sets.

The default is to write to internal trace tables (EXTERNAL=NO). To write to an external data set, you must set EXTERNAL=YES on the TRCLEV statement and specify the EXTTRACE parameter in the BPE configuration parameter member. You must also define a generation data set group (GDG) for BPE to trace into. If you specify EXTERNAL=YES, trace data is written to both an external data set and internal trace tables.

You can dynamically change the external trace data set specification in the BPE configuration PROCLIB member and refresh the member while an address space is running. For example, if you are running without an external trace data set, you can edit your BPE PROCLIB member, add an external trace data set specification, and start using external trace without having to restart the address space.

Related reference

BPE UPDATE TRACETABLE command (Commands)

Writing to internal trace tables

The default BPE execution environment settings for the address space is to write to the internal trace table (EXTERNAL=NO).

To write BPE trace records to internal trace tables, specify the type, level, and, optionally, the number of storage pages allocated for the trace table on the TRCLEV parameter of the BPE configuration parameter member (BPECFG=) of the IMS.PROCLIB data set.

Related reference

BPE configuration parameter member of the IMS PROCLIB data set (System Definition)

Writing to external data sets

To write to an external data set, you must set EXTERNAL=YES on the TRCLEV statement and specify the EXTTRACE parameter in the BPE configuration parameter member.

To write BPE trace records to an external data set, you must:

1. Define the external trace data set Generation Data Group (GDG).
2. Define the GDG model.
3. Specify the EXTTRACE parameter in the BPE configuration parameter member (BPECFG=) of the IMS.PROCLIB data set. The EXTTRACE statement should specify the name of the GDG you defined in step 1.
4. Specify which trace tables you want to write to the BPE external trace data sets. You can do this by using either or both of the following methods:

- Include the EXTERNAL=YES parameter with the other TRCLEV options on the TRCLEV statements for the trace tables you want to externalize. The TRCLEV statements are coded in the BPE configuration parameter member (BPECFG=) of the IMS PROCLIB data set.

  Tip: Changes to TRCLEV statements are only processed when an address space is started. If your address space is running when you make the changes, they will not take effect until the address space is restarted.
- Issue the **`UPDATE TRACETABLE`** command, specifying the trace tables you want to externalize on the NAME parameter, and specifying EXTERNAL(YES). This command is used while an address space is running to dynamically turn external tracing on, regardless of the TRCLEV parameter specification.

5. Optional: If you want to start tracing to external data sets but you did not specify the EXTTRACE parameter when you started the address space:
   a) Add the EXTTRACE data set to the BPE configuration member of the IMS.PROCLIB data set while the address space is running.
   b) Issue the **`UPDATE TRACETABLE`** command and specify the OPTION(REREAD) option to cause the BPE to process the newly-updated EXTTRACE parameter.

Information about the EXTTRACE and TRCLEV statements and BPECFG= and the IMS.PROCLIB data set is in *IMS Version 15.2 System Definition*.

Defining an external trace data set
Use the z/OS **`DEFINE GENERATIONDATAGROUP`** command to define the GDG. The value that you specify for the NAME keyword in this command must match the data set name in the GDGDEF parameter of the EXTTRACE statement of the BPE configuration parameter member (BPECFG=) of the IMS PROCLIB data set.

Defining a GDG from an IDCAMS job

You can issue the z/OS **`DEFINE GENERATIONDATAGROUP`** command from an IDCAMS job, as shown in the following example.

```
//DEFGDG   JOB  ...
//STEP1    EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=A
//SYSIN    DD   *
   DEFINE GENERATIONDATAGROUP -
      (NAME(BPEEXTRC.GDG01) -
       NOEMPTY -
       SCRATCH -
       LIMIT(255))
```

Defining a GDG from a TSO session

You can issue the z/OS **`DEFINE GENERATIONDATAGROUP`** command from a TSO session, as shown in the following example.

```
DEFINE GENERATIONDATAGROUP (NAME(BPEEXTRC.GDG01) NOEMPTY SCRATCH LIMIT(255))
```

Defining a GDG from a TSO batch job

You can issue the z/OS **`DEFINE GENERATIONDATAGROUP`** command from a TSO command batch job, as shown in the following example.

```
//DEFGDG     JOB  ...
//STEP1      EXEC PGM=IKJEFT01
//SYSTSPRINT DD   SYSOUT=A
//SYSTSIN    DD   *
DEFINE GENERATIONDATAGROUP (NAME(BPEEXTRC.GDG01) NOEMPTY SCRATCH LIMIT(255))
```

For more information about the **`DEFINE GENERATIONDATAGROUP`** command, see *z/OS DFSMS Access Method Services for Catalogs*.

For additional information on trace data, see *IMS Version 15.2 Diagnosis*.

Defining the GDG model data set
You must provide a data control block (DCB) to be used when GDG entries are created. You can provide it when you define the generation data group (GDG) model data set. To define the GDG model data set, use the z/OS program IEFBR14.

Sample JCL to define a GDG model data set using IEFBR14

The following example provides sample JCL to define a GDG model data set, including the DCB information.

```
//* STEP1 - Define the GDG Base
//STEP2    EXEC PGM=IEFBR14
//BLDDSCB  DD DSN=BPEEXTRC.GDG01,
//            DISP=(NEW,KEEP),
//            UNIT=SYSDA,
//            VOL=SER=PAGE01,
//            SPACE=(TRK,(0)),
//            DCB=(DSORG=PS,LRECL=32756,RECFM=VB,BLKSIZE=32760)
```

You can also pass the DCB information using the DSN parameter on the EXTTRACE statement. However, you cannot override the data set attributes.

You must define the data set with RECFM=VB, and an LRECL that is 4 bytes less than the BLKSIZE. The minimum supported BLKSIZE is 8340.

Recommendation: Use a large block size (such as the maximum of 32 760). Using a large BLKSIZE ensures that multiple records are written per block. In addition, you would not have to change your BPE external trace definitions if the minimum supported block size is ever increased.

For more information on IEFBR14, see *z/OS MVS JCL Reference*.

Starting and stopping BPE external tracing

To start the BPE external trace, you must specify EXTERNAL=YES keyword on the TRCLEV= parameter of the BPE configuration parameter member (BPECFG=) of the IMS.PROCLIB data set or issue the UPDATE TRACETABLE command with the EXTERNAL(YES) parameter on it.

To see which trace tables have external tracing turned on, use the BPE **`DISPLAY TRACETABLE`** command.

If IMS encounters an I/O error while writing to one of the external trace data sets, IMS closes and deallocates the current data set and allocates and opens a new data set. If IMS cannot allocate or open a new data set, IMS disables BPE tracing for all components that use that GDG data set. Issue the BPE **`UPDATE TRACETABLE`** command to have IMS retry the allocation and open processing for the data set.

If the I/O capacity of the GDG data sets cannot support the volume of trace records being created, IMS skips writing trace records until the I/O system can catch up rather than waiting and potentially delaying other IMS work.

To stop BPE external tracing for a particular trace table type:

1. Issue the **`UPD TRTAB EXTERNAL(NO)`** command.
2. Specify the table type on the NAME parameter.

To stop BPE external tracing for all tables issue `UPD TRTAB NAME(*) EXTERNAL(NO)`. BPE closes and deallocates the last external trace data set when there are no more table types being externally traced.

Information about BPE commands is in *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*.

Formatting and viewing BPE external trace data sets

To format and view BPE trace entries, use the Interactive Problem Control System (IPCS). For information on using IPCS to view BPE trace entries, see *IMS Version 15.2 Diagnosis* .

Common Service Layer overview

The IMS *Common Service Layer* (CSL) is a collection of IMS manager address spaces that provide the infrastructure needed for systems management tasks in an IMSplex. The CSL address spaces include Open Database Manager (ODBM), Operations Manager (OM), Resource Manager (RM), and Structured Call Interface (SCI).

The CSL is built on the IMS Base Primitive Environment (BPE) layer. As a result, all BPE externals, such as commands, messages, abends, configurations, and user exit interfaces apply to all CSL manager address spaces.

The IMS CSL provides:

- Improved systems management
- A single system image
- Ease of use through a single point of control
- Shared resources across all IMS systems

A single CSL can service multiple IMS systems within an IMSplex, either on the same operating system or on multiple operating systems.

Recommendation: Activate more than one CSL manager of each type in the IMSplex. An IMS control region within an IMSplex that is defined with CSL cannot start unless at least one OM is active in the IMSplex and an SCI resides on every operating system in the IMSplex. This recommendation also applies to RM, although an RM is not required for an IMS control region to start.

There are advantages and disadvantages of having all CSL managers on each operating system. One advantage is better system performance. It is faster for an IMSplex member to communicate with a CSL manager on the same operating system than it is for the IMSplex member to communicate with a CSL manager on a different operating system. A disadvantage is the increased number of address spaces on each operating system. However, if only one instance of a CSL manager is defined in an IMSplex, whether it is ODBM, OM, or RM, no backup of that CSL manager exists if it should fail.

If you do not use shared queues or sysplex technology, you can take advantage of a simplified CSL configuration to issue *type-2 commands* through the CSL OM.

Related concepts

“A simplified CSL configuration” on page 35
If your IMS configuration does not require RM services, you can still use OM and SCI to take advantage of type-2 IMS commands.

CSL in an IMSplex

The IMS CSL reduces the complexity of managing multiple IMS systems by providing you with a single-image perspective in an *IMSplex*. An IMSplex is one or more IMS subsystems (control, manager, or server) that can work together as a unit.

Typically, but not always, these subsystems:

- Share either databases or resources or message queues (or any combination)
- Run in a z/OS sysplex environment
- Include an IMS CSL

Within an IMSplex, you can manage multiple IMS subsystems as if they were one system. For example, instead of entering commands on each IMS system during local online change, you can enter commands from one single point of control, and the commands run on each IMS system in the IMSplex. An IMSplex can also exist in a local, or non-sysplex, environment. Use of the CSL is optional.

An IMSplex component is an IMS-defined entity that typically runs in its own address space; it manages resources, manages operations, or facilitates communications between other IMS-defined entities. After the necessary started procedures for these components are initialized, they become *IMSplex members*. Examples of IMSplex components are:

- IMS subsystems (DB/DC, DBCTL, DCCTL)
- Resource Manager
- Operations Manager
- Open Database Manager
- Structured Call Interface
- IMS Connect
- A DLIBATCH or DBBBATCH region
- Repository Server (RS)

A DLIBATCH or DBBBATCH region is considered a special type of IMSplex component in that it does not interact with RMs and OMs.

A typical IMSplex environment including a CSL is shown in the following figure. There are three separate operating system (OS) images. Each OS has an IMS control region and an SCI. In addition, OS 1 has an OM and an RM, OS 2 has an OM and an ODBM, and OS 3 has an ODBM. All three OS images share a coupling facility, which has database sharing structures, a message queue structure, and a resource structure.

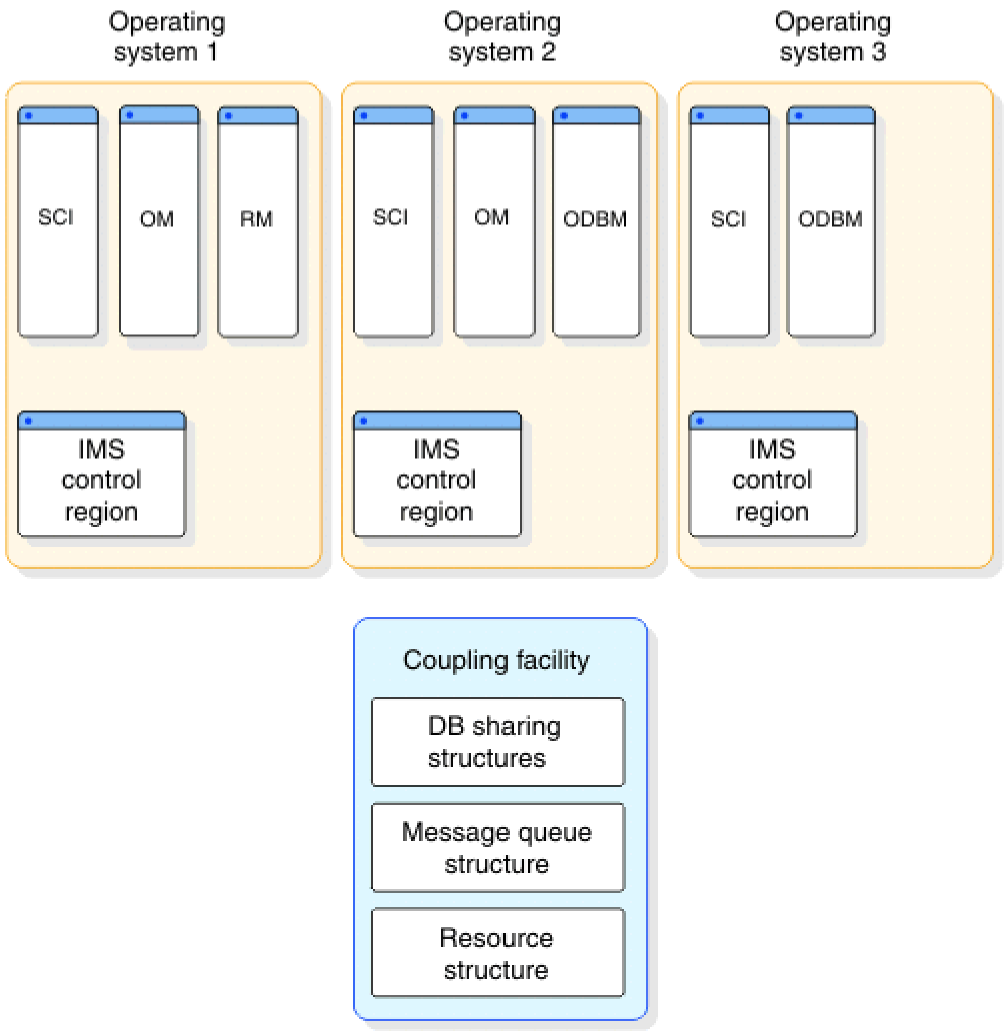


Figure 9. IMSplex environment including a CSL

The CSL includes the following IMS address spaces, some of which are optional:

- ODBM
- OM
- RM

- SCI

The address spaces that can participate in an IMSplex are:

- IMS control region address spaces
- IMS CSL manager address spaces (ODBM, OM, RM, SCI)
- IMS server address spaces (Common Queue Server)
- Non-IMS address spaces
- RS address spaces

The IMSIDs of IMS systems in the IMSplex must be unique.

A simplified CSL configuration

If your IMS configuration does not require RM services, you can still use OM and SCI to take advantage of type-2 IMS commands.

The commands can be issued using the TSO SPOC or any user-written or vendor-written automation program that uses the OM API.

You need CSL with RM to have IMS enabled with the IMSRSC repository.

The IMS Application Menu provides a common interface to enable you to start applications such as TSO SPOC, Syntax Checker, IVP, and more. For more information about the IMS Application Menu, see *IMS Version 15.2 Installation*.

Related concepts

"Common Service Layer overview" on page 33
The IMS *Common Service Layer* (CSL) is a collection of IMS manager address spaces that provide the infrastructure needed for systems management tasks in an IMSplex. The CSL address spaces include Open Database Manager (ODBM), Operations Manager (OM), Resource Manager (RM), and Structured Call Interface (SCI).

Related tasks

"Defining a simplified IMSplex for the type-2 command environment" on page 22
To define a simplified IMSplex without RM either specify RMENV=N in the DFSCGxxx member of the IMS PROCLIB data set or specify RMENV=N in the CSL section of the DFSDFxxx PROCLIB member while defining an IMSplex.

CSL managers

The CSL is a key component in the evolving architecture of IMS. Using the CSL in an IMSplex provides you with the infrastructure for improved systems management.

The CSL address spaces are also referred to as CSL managers:

- Open Database Manager
- Operations Manager
- Resource Manager
- Structured Call Interface

Overview of the CSL Open Database Manager

Open Database Manager (ODBM) provides distributed and local access to IMS databases that are managed by IMS DB systems configured for either the DBCTL or the DB/TM environments in an IMSplex.

Both independently and with IMS Connect, ODBM supports various interfaces to ease the development of application programs that access IMS databases from many different distributed and local environments. The interfaces that ODBM supports include:

- IMS Universal Database resource adapter

- IMS Universal JDBC driver
- IMS Universal DL/I driver
- The ODBA interface
- The ODBM CSLDMI interface for user-written ODBM client application programs

In addition to supporting the above interfaces, ODBM provides the following functions:

- Supports one-phase commit with or without requiring z/OS Resource Recovery Services as the sync point coordinator. If the ODBM parameter RRS=N is specified, the Database Resource Adapter (DRA) is used to communicate with IMS.
- Routes incoming database requests to IMS systems based on alias names. Alias names for IMS systems are defined to ODBM on the CSLDCxxx PROCLIB member and specified on incoming database requests by the client application programs.

  If the client application program does not specify an alias name on a database access request, ODBM routes the request among multiple IMS systems by using a round-robin method of distribution, in which ODBM routes each incoming request to the active IMS systems that are defined to ODBM.
- Enables ODBM client application programs to access databases from other logical partitions within an IMSplex
- Protects IMS control regions from the unexpected termination during DL/I processing of z/OS application programs that use the ODBA interface through ODBM.
- Functions as an RRMS resource manager capable of participating in RRS controlled sync point processes for ODBA applications that are configured to use ODBM.
- Functions as the resource manager and issues the necessary calls to RRS for single-phase commit processing for local RRS transactions.
- Functions together with IMS Connect as a complete Distributed Relational Database Architecture™ (DRDA) target server for client application programs that use the DRDA specification.

Because ODBM and IMS Connect support DRDA, you can develop your own DRDA source application server that communicates with IMS by using the distributed data management (DDM) Architecture commands that are a part of the DRDA specification.

ODBM leverages the z/OS System Management Facility (SMF) to perform logging of ODBM accounting information, such as CPU usage, and its retrieval. The logging of the ODBM address space is activated when the optional parameter LOGOPT=ACCOUNTING is specified in the ODBM initialization member, CSLDIxxx.

ODBM runs in a Common Service Layer (CSL) address space and is an optional IMSplex component. ODBM uses the Structured Call Interface (SCI) services of the CSL for communication and Operations Manager (OM) services of the CSL for command processing.

One ODBM instance must be defined in the IMSplex to use ODBM functions. Each z/OS image can have more than one ODBM. If multiple instances of ODBM are defined in the IMSplex, any ODBM instance can perform work from any z/OS image in the IMSplex.

The following figure shows an overview of an IMS configuration that includes ODBM.

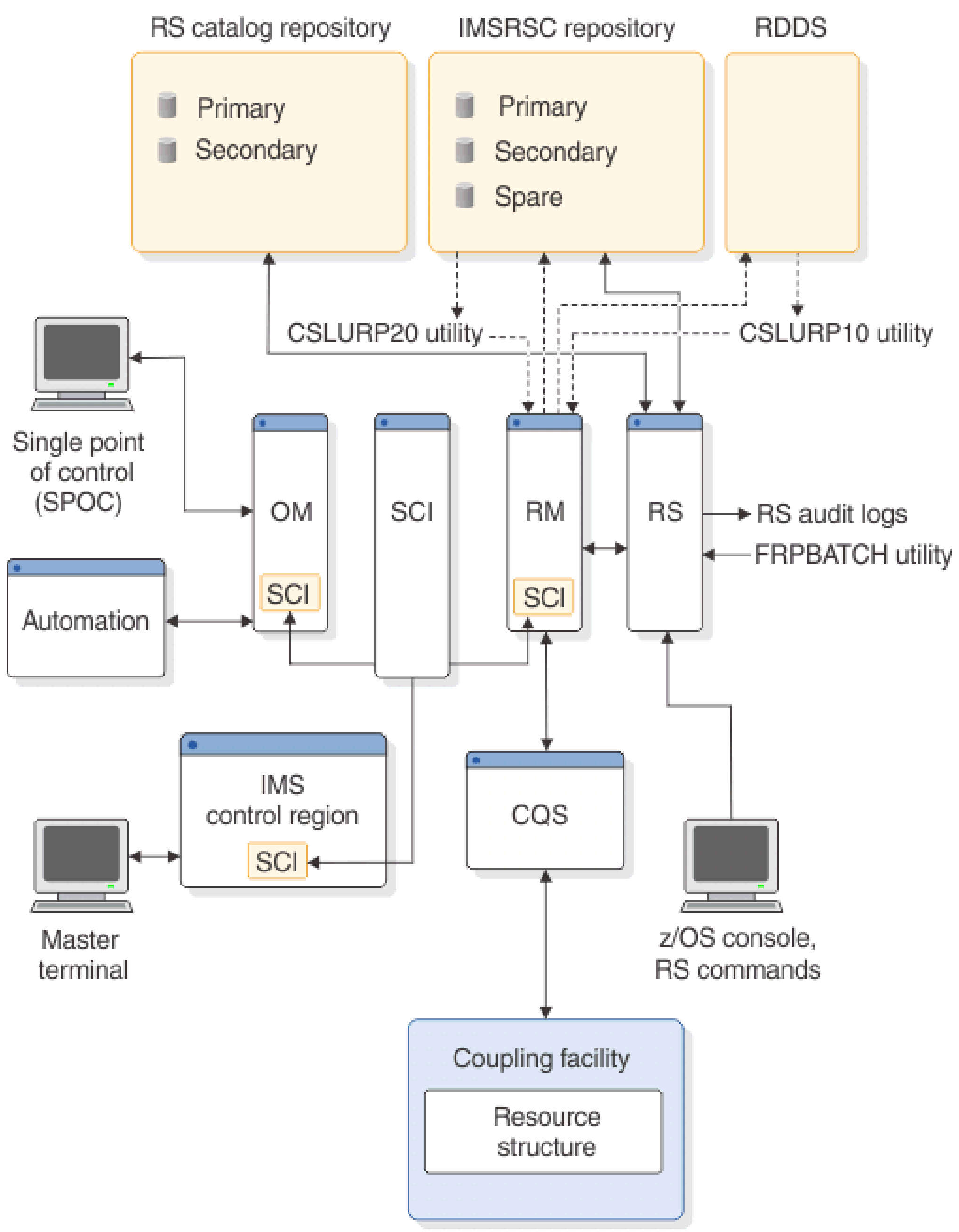


Figure 10. Overview of an IMS configuration that includes ODBM

Related concepts
"CSL ODBM administration" on page 129
The CSL Open Database Manager (ODBM) supports access to databases that are managed by IMS DB in DBCTL and DB/TM IMS systems in an IMSplex.

Programming with the IMS support for DRDA (Application Programming)

Overview of the Operations Manager

OM controls the operations of an IMSplex. OM provides an application programming interface (the OM API) through which commands can be issued and responses received. With a single point of control (SPOC) interface, you can submit commands to OM. The SPOC interfaces include the TSO SPOC and the REXX SPOC API. You can also write your own application to submit commands.

OM provides the following functions to an IMSplex:

- Routes IMS commands to IMSplex members registered for the command.
- Consolidates command responses from individual IMSplex members into a single response and provides that response to the originator of the command.
- Provides an API for automated operator commands.
- Provides a general use interface to register commands to support any command processing client.
- Provides user exits for command and response edit and command security.

OM log records can be written to a z/OS System Logger log stream by specifying the AUDITLOG= parameter of the IMSPLEX() keyword on the CSLOIxxx PROCLIB member. The log records are an audit trail of command input, associated command output, and unsolicited message output.

One OM must be defined in the IMSplex to use OM functions. Each z/OS image can have more than one OM. If multiple OMs are defined in the IMSplex, any OM defined can perform work from any z/OS image in the IMSplex.

Related reference

"CSL configuration examples" on page 46

Typically, when an IMS control region (DL/I, DBRC, dependent regions) requires the use of the CSL, SCI, OM, and RM are all required. However, different configurations are possible for the CSL in an IMSplex. For example, you might include ODBM or omit RM in your CSL configuration.

Overview of the Resource Manager

RM helps you manage resources that are shared by multiple IMS systems in an IMSplex. RM provides the infrastructure for managing global resource information and coordinating IMSplex-wide processes, such as global online change.

RM provides the following functions to an IMSplex:

- Maintains global resource information in a *resource structure*, which is a coupling facility list structure that all RMs in the IMSplex can access.
- Ensures resource consistency so that a resource defined as a transaction, lterm, or msname is defined as the same resource type for all IMS systems in the IMSplex.
- Supports resource services.
- Supports client services.
- Uses the Common Queue Server (CQS) to maintain global resource information.
- Coordinates IMSplex-wide processes (such as global online change).

With RM, the system administrator can manage resources that are shared by multiple IMS systems in an IMSplex. RM provides an infrastructure for managing global resources and coordinating processes across the IMSplex.

Recommendation: Although a resource structure is optional in an IMSplex, define a resource structure for improved recovery capabilities. However, if you are using shared queues and you want to manage serial applications in an IMSplex, at least one RM must be active with a defined resource structure.

At least one RM must be defined in an IMSplex to use RM functions. You can have one or more RMs on each z/OS image if a resource structure is defined. If no resource structure is defined, you can have only one RM. If you do not require RM functions, you can configure your IMS system without an RM, specifying RMENV=N on the DFSCGxxx PROCLIB member data set. Any RM can process work from any z/OS image within an IMSplex.

Related reference

"CSL configuration examples" on page 46

Typically, when an IMS control region (DL/I, DBRC, dependent regions) requires the use of the CSL, SCI, OM, and RM are all required. However, different configurations are possible for the CSL in an IMSplex. For example, you might include ODBM or omit RM in your CSL configuration.

Overview of the CSL Structured Call Interface

SCI allows IMSplex members to communicate with one another. Communication between IMSplex members can occur within a single z/OS image or among multiple z/OS images. The individual IMSplex members do not need to know where the other members reside or what communication interface to use.

SCI provides the following functions to an IMSplex:

- Routes messages and requests within an IMSplex.
- Registers and deregisters IMSplex members.
- Notifies IMSplex members when a member joins or leaves the IMSplex.
- Provides security authentication of members when they join the IMSplex.
- Provides a single call interface to isolate the client and server from the underlying communications technology.

Any IMSplex member that requires SCI services must have an SCI on its z/OS image. There can be at most one SCI address space per IMSplex on each z/OS image.

Note: DBRC automatic RECON loss notification and parallel RECON access use SCI, but not RM or OM. You can bring up an SCI address space without requiring the other CSL manager address spaces for automatic RECON loss notification.

Recommendation: Initialize each CSL manager address space in an IMSplex.

Related reference

"CSL configuration examples" on page 46

Typically, when an IMS control region (DL/I, DBRC, dependent regions) requires the use of the CSL, SCI, OM, and RM are all required. However, different configurations are possible for the CSL in an IMSplex. For example, you might include ODBM or omit RM in your CSL configuration.

A single point of control (SPOC) program in CSL

A single point of control (SPOC) is a program that allows you to manage operations of all IMS systems within an IMSplex instead of using a master terminal. With a SPOC, you can issue commands to all members of an IMSplex at once.

A SPOC communicates with a single OM. Through SCI, OM then communicates with all of the other IMS control regions in the IMSplex.

A SPOC is optional in an IMSplex. The existing command interfaces for the WTOR, MTO, and E-MCS console are supported only for type-1 commands. If you want to use type-2 commands, you must use a SPOC. You can use a SPOC instead of the z/OS master console, IMS master terminal, or user terminal for most command processing.

Types of SPOCs:

- On a 3270 TSO terminal, a TSO SPOC is a system management application with an ISPF panel interface. You can start the TSO SPOC using the IMS Application Menu. The TSO SPOC also displays the OM audit trail, which has a log of messages issued by IMS and commands issued by operators.
- The Batch SPOC utility uses OM services to submit IMS operator commands to an IMSplex. The utility accepts any commands that are supported by the OM API. The Batch SPOC utility is invoked by using standard JCL statements.
- The REXX SPOC API allows automation programs to use SPOC functions. You can use the REXX SPOC API to automate commands.
- Vendor- or user-written SPOC: A program that uses or accesses the OM API to perform SPOC functions.

There can be more than one type of SPOC in an IMSplex, and any number of SPOCs can be active.

A SPOC provides the following functions to an IMSplex:

- Single-system view of an IMSplex by allowing you to operate all IMS systems in the IMSplex from a single console
- Consolidated display of command responses from multiple IMS systems
- Message sending to IMS terminals connected to any IMS control region in the IMSplex

The following figure shows a SPOC application in an IMSplex. The IMSplex in this example has three identical OS images: each has an IMS control region, an IMS CQS address space, and an SCI, OM, and RM. All three OS images share a coupling facility, which includes database sharing structures, a message queue structure, and a resource structure. the IMSplex configuration also includes shared databases and RECON data sets.

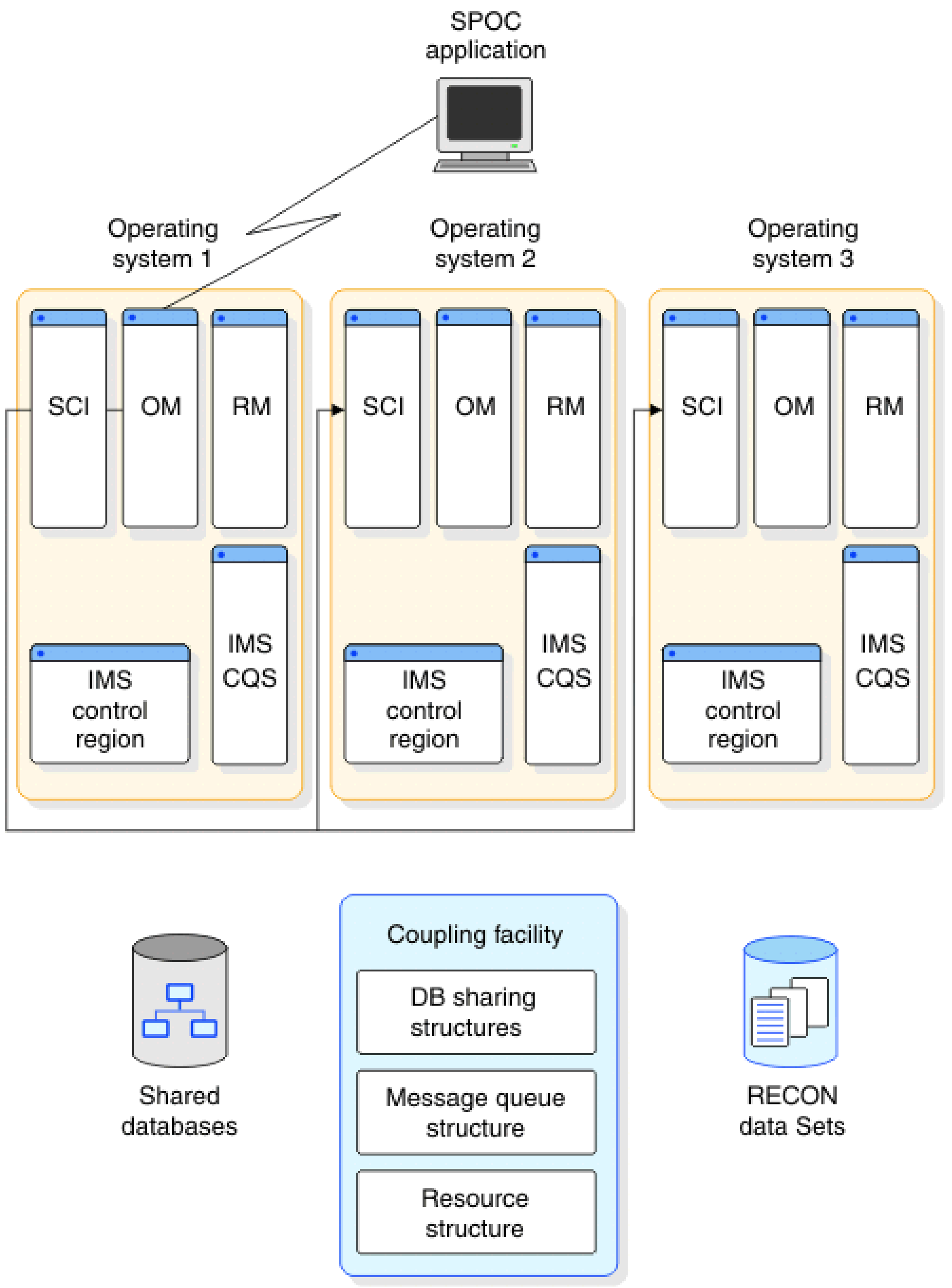


Figure 11. SPOC application in an IMSplex

The following figure shows an IMSplex environment with multiple SPOC users. The IMSplex environment includes four IMS control regions, each having its own SCI, and one OM. The multiple SPOC users include two SPOC TSO/ISPF applications, a REXX SPOC API, and a vendor-written SPOC program. Each SPOC can access the IMSplex environment.

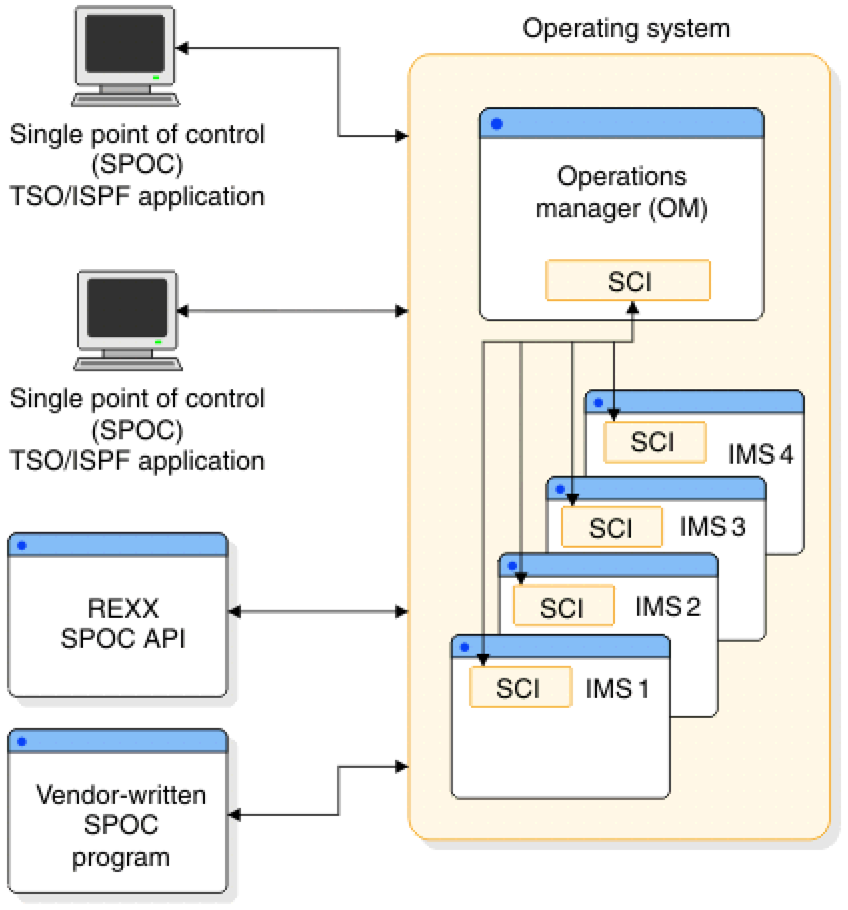

Operating system
Single point of control (SPOC) TSO/ISPF application
Operations manager (OM)
SCI
Single point of control (SPOC) TSO/ISPF application
SCI
IMS 4
SCI
IMS 3
SCI
IMS 2
REXX SPOC API
SCI
IMS 1
Vendor-written SPOC program

Figure 12. Multiple SPOC users in an IMSplex

Related concepts

Controlling IMS with the TSO SPOC application (Operations and Automation)

"REXX SPOC API" on page 41

The REXX SPOC API allows automated operator application programs written in the REXX programming language to submit IMS operator commands to the IMSplex and to retrieve the command responses.

Related tasks

Starting the IVP from the IMS Application Menu (Installation)

Related reference

IMS type-2 command format (Commands)

REXX SPOC API

The REXX SPOC API allows automated operator application programs written in the REXX programming language to submit IMS operator commands to the IMSplex and to retrieve the command responses.

The command responses are in the form of XML statements. REXX application programs can examine the command responses and perform further processing if needed.

For example, an REXX application program can issue the following command to check the queue count for a transaction:

```
QRY TRAN NAME(prod001) SHOW(QCNT)
```

If the response shows a high queue count, the REXX application program can issue other commands to start additional regions or to change the class of the transaction.

You can write REXX applications programs and run them in a TSO, a batch, or a Tivoli® NetView® for z/OS environment. You must start the SCI, the OM, and the IMS control region address spaces before the REXX application program can issue IMS operator commands.

Global online change

The global online change function changes resources online for all IMS systems in an IMSplex.

The benefits of global online change are as follows:

- Simplified process for changing resources online by issuing one instance of a command for online change for all IMS systems in the IMSplex
- Continuous processing so that each IMS is not stopped and manually coordinated with online changes.
- No situations where online change is committed on some IMS systems in the IMSplex and not committed on other IMS systems

With global online change, the master IMS control region coordinates online change of the following resources among all the IMS systems in the IMSplex:

- Databases (DMB in ACBLIB)
- Database directories (DDIR in MODBLKS)
- MFS formats (FMTLIB)
- Programs (PSB in ACBLIB)
- Program directories (PDIR in MODBLKS)
- Transactions (SMBs in MODBLKS)
- Routing codes (RCTEs in MODBLKS)

Restriction: The online change function is not supported for the following types of resources when dynamic definition of the resources is enabled:

- Database (DBD) and program view (PSB) resources when the IMS management of ACBs is enabled.
- MODBLKS resources when dynamic resource definition is enabled.

Global online change operates through a master IMS control region, as specified by the user or by OM. OM uses RM, if RM is active, to coordinate the phases of online change with the other IMS systems in the IMSplex. Start online change by issuing the command **`INITIATE OLC`** through the OM API. At that time you can specify the master IMS control region. If you do not choose a master IMS control region, OM selects one of the IMS systems as the master IMS control region.

If global online change is specified in the IMSplex, the following components are required:

- An IMSplex defined with CSL and at least one OM
- CQS, if there is a resource structure in the IMSplex

  The resource structure is used to save IMSplex-wide process status. It also provides more recovery capability during the online change, should any failures occur.
- OLCSTAT data set, which must be initialized by the Global Online Change utility (DFSUOLC0) before the first IMS cold start

  If you exclude RM from the IMSplex by specifying RMENV=N, each IMS system must have its own OLCSTAT data set and that OLCSTAT data set must contain the IMSID of the IMS system that owns it and no other IMSIDs.
- OLC=GLOBAL and OLCSTAT= parameters in the DFSCGxxx PROCLIB member data set
- CSLG= parameter in the IMS EXEC statement

At least one RM and a resource structure are recommended for global online change, but they are not required. Consider the following if you choose not to use RM or a resource structure. If you do not use an RM in your IMSplex, the OLCSTAT data set can contain only the IMSID of the IMS system that owns that OLCSTAT data set. Any attempt to restart an IMS system that contains an OLCSTAT data set with a different or multiple IMSIDs results in an abend. IMS rejects **INITIATE OLC** and **TERMINATE OLC** commands that are issued by an IMS system other than the IMS system that owns the OLCSTAT data set.

If you choose not to enable global online change, the MODSTAT data sets in the IMS procedure are optional. The IMS control region sample JCL includes DD statements for the MODSTAT data set and the MODSTAT2 data set (if XRF). If you specify the MODSTAT DD statements for an IMSplex where global online change is enabled, the MODSTAT and MODSTAT2 (if XRF) data sets must also exist.

Related tasks
"Defining a simplified IMSplex for the type-2 command environment" on page 22
To define a simplified IMSplex without RM either specify RMENV=N in the DFSCGxxx member of the IMS PROCLIB data set or specify RMENV=N in the CSL section of the DFSDFxxx PROCLIB member while defining an IMSplex.

ACB library member online change

If an IMS system uses an ACB library in an IMSplex environment, you can use the ACB member online change (OLC) function to add or change individual members of the ACB library, or the entire ACB library, and bring these new or changed members online without quiescing the IMSplex or refreshing the active ACB library.

The ACB member online change function does not apply when the IMS management of ACBs is enabled. In this case, individual ACB members can be changed online by submitting the changes to IMS by using either DDL or one of the IMS catalog population utilities and, if the change is not activated in the online system automatically, issuing the IMPORT DEFN SOURCE(CATALOG) command.

Members that are not affected by the member OLC are not quiesced.

You can make online changes to the ACB library in IMSplex environments in which:

- A single IMS system comprises the IMSplex. In this configuration, IMS must use the OLCSTAT data set with a CSL that consists of an OM and SCI. An RM is not required.
- Multiple IMS systems comprise the IMSplex. In this configuration, each IMS system must use the same shared OLCSTAT data set, or each system must use its own local OLCSTAT data set. The CSL must consist of an OM, an SCI, and an RM. Although a resource structure is recommended, it is not required.

Global online change for ACBLIB members can only be performed in an IMSplex.

You can specify whether the ACBLIB member is shared or dedicated in the IMSplex. The following example highlights how the ACBLIB member online change process can be coordinated among all IMS systems that share the OLCSTAT data set:

- OM chooses an IMS system to be the command master IMS.
- The command master IMS coordinates the member OLC process with other IMS systems sharing the OLCSTAT data set using RM.
- ACBSHR= must be specified as Y or N in the DFSCGxxx or DFSDFxxx PROCLIB member.
  - Y– Indicates that all of the IMS systems in the OLCSTAT are using the same active and inactive ACBLIB.
  - N– Indicates that each IMS in the OLCSTAT is using its own dedicated active and inactive ACBLIB.
- All IMS systems in the IMSplex that share the OLCSTAT data set must specify the value of ACBSHR=.
- The member OLC updates all the active ACBLIBs, whether or not they are shared.

The ACB member online change uses IMS type-2 commands only. The commands that are involved in performing an ACB library member online change are:

- **INITITATE OLC PHASE(PREPARE)**
- **INITIATE OLC PHASE(COMMIT)**
- **TERMINATE OLC**

- **QUERY MEMBER TYPE(IMS)**
- **QUERY OLC SHOW(RSCLIST)**

See "Changing or adding IMS.ACBLIB members online" on page 450 for the steps required to perform an ACB member online change. If you want to fall back to the previous version of the changed resource, perform a full online change process with a full library switch.

Global TM resource management

Using RM and a resource structure improves your ability to manage Transaction Manager resources in an IMSplex. IMS enforces name uniqueness for LTERMs (VTAM), nodes (VTAM single-session), user IDs (if the installation requests single-signon enforcement), and users.

IMS also ensures resource type consistency for message destinations and supports global callable services.

Related tasks
"Defining and tailoring an IMSplex" on page 20
You must establish the procedures required to start an IMSplex, and specify the appropriate parameters in the DFSCGxxx member or the CSL section of the DFSDFxxx member of the IMS PROCLIB data set.

Automatic RECON loss notification overview

SCI is the only CSL manager required for automatic RECON loss notification. If you are using the CSL solely for the automatic RECON lost notification, OM and RM are not required. However, SCI must be available on each z/OS image. If the IMS control region is started using the DFSCGxxx or the CSL section of the DFSDFxxx member, OM (and possibly RM) is required.

All DBRC instances using a given RECON must join the same IMSplex. You can have your IMS systems configured so that two or more IMS sharing groups are using a single RECON. This configuration is supported if the sharing groups are part of the same IMSplex. However, if each sharing group is identified as a unique IMSplex, then the RECON must be split into multiple RECONs, one per IMSplex. In addition, if the control region and DBRC address spaces both use SCI, they must also join the same IMSplex. You can also have your IMS systems configured so that two or more IMS sharing groups are part of the same IMSplex, but use unique RECONs.

Recommendation: Each RECON in the same IMSplex should have a unique DBRC sharing group ID. This is required if parallel RECON access is enabled for more than one RECON in the IMSplex.

Related reading: For detailed information about automatic RECON loss notification configuration, see *IMS Version 15.2 Database Administration*.

Configuring an IMSplex with CSL

An IMS control region that is running without CSL must be stopped and restarted to connect to CSL. When CSL is introduced into the IMSplex, the IMS control regions can be stopped and restarted one at a time. The entire IMSplex does not need to be shutdown to introduce CSL.

To configure an IMSplex with CSL, you must initialize at least one of each of the following IMSplex components:

- DB/DC, DCCTL, or DBCTL IMS control region
- DBRC for each IMS control region
- Operations Manager (OM)
- Structured Call Interface (SCI) on each operating system on which an IMSplex member is running
- CQS when message queues are shared or RM is used with the resource structure to manage global resources

It is also strongly recommended that you also initialize a Resource Manager (RM) and a resource structure. Although an RM and a resource structure are not required, you cannot share or manage resources globally.

Optionally, you can have the following in your IMSplex:

- CSL Open Database Manager, which provides distributed and local access to IMS databases that are managed by IMS DB systems configured for either the DBCTL or the DB/TM environments in an IMSplex.
- Data sharing of one or more databases in a DB/DC or DBCTL environment.
- One DL/I address space and 1 to 999 dependent regions for each IMS control region.
- Shared queues.

  Important: All IMS control regions should share message queues. Otherwise, shared queues should not be used. Running in a mixed environment where some of the queues are shared and others are not can result in an unpredictable or misleading output from type-2 commands, which are entered through the OM API.
- Any number of batch (DLIBATCH or DBBBATCH) regions. OM and RM do not interact with batch regions; therefore, they do not affect management of batch regions.
- IMSRSC repository for the shared resource definitions, along with the Repository Server (RS) address space.

To include CSL, an IMS control region must be started with the CSLG= parameter, or if the DFSDFxxx member of the IMS PROCLIB data set includes the COMMON_SERVICE_LAYER section , the DFSDF= parameter. The minimum configuration for the CSL is to have only one SCI on each operating system where an IMSplex component resides (including IMSplex components like TSO SPOC and CQS), one or more OMs anywhere in the IMSplex, and one or more RMs anywhere in the IMSplex.

How many RMs and OMs to have depends on your requirements. An outline of CSL configuration rules follows:

- Only one SCI can run on an operating system for a given IMSplex. Multiple IMS systems on the same operating system use the same instance of SCI to communicate within the same IMSplex. If one or more IMS systems on the same operating system are associated with multiple IMSplex systems, you must run a separate instance of SCI for each IMSplex.
- At least one OM must be active in an IMSplex. You can have one or more OMs active on one or more operating systems.
- An RM is not required, but at least one active RM is recommended in an IMSplex. You can have one RM active on any number of operating systems. If you have more than one RM, you must also have a resource structure.
- A CSL configuration with an RM is required if IMS is enabled to use the repository.

Configuring the resource structure for an IMSplex

The resource structure, supported by the Common Queue Server (CQS), is the repository for resources managed by RM. CQS manages uniquely named client resources that are shared within a single IMSplex by using the coupling facility list structures.

The resource structure is optional; however, if there is no resource structure certain IMSplex functions cannot be used.

Clients can use the resource structure to share resource information, control block information, and so on. The resource name is unique within the structure. Resources can be updated, deleted, or queried. A primary coupling facility list structure is used to contain the resources. Resource structures do not support an overflow structure.

CQS does not support structure recovery, structure checkpoint, or overflow processing for resource structures. Also, to improve performance, changes to resource structures are not logged. However, for recovery of resources, resource structures can be automatically duplexed by specifying so in the CFRM policy.

CQS defines the attributes of the resource structure the first time it connects to the structure. It defines the resource structure as a persistent, ENTRYKEY list structure and the list entries on the resource structure are defined to contain an adjunct area and a data entry. The resource structure is defined to permit SVC dumps, structure alter, and autorebuild.

Recommendation: Use multiple RMs and a resource structure in your IMSplex. Without a resource structure, you cannot use the global resource-sharing capabilities of RM. Additionally, without a resource structure, you can start only one RM. If that one RM goes down, you do not have a backup RM. However, if you do not have a resource structure and are in a DBCTL environment, you can still perform global online changes with one RM.

Configuring a simplified IMSplex for the type-2 command environment

If you want to use type-2 commands and you have a standalone IMS system or multiple IMS systems that only share databases, you can define an IMSplex without RM. This creates a simplified IMSplex with CSL.

Related reading:

- For information on restrictions for online change when an RM is not used, see "Global online change" on page 42.
- For information on defining an IMSplex without an RM, see:
  - "Defining a simplified IMSplex for the type-2 command environment" on page 22
  - *IMS Version 15.2 System Definition*
- For detailed information about CSL configuration, see "CSL configuration examples" on page 46.

CSL configuration examples

Typically, when an IMS control region (DL/I, DBRC, dependent regions) requires the use of the CSL, SCI, OM, and RM are all required. However, different configurations are possible for the CSL in an IMSplex. For example, you might include ODBM or omit RM in your CSL configuration.

The basic CSL components are the CSL managers, which are:

Open Database Manager
The Open Database Manager (ODBM) is optional in an IMSplex. You can define additional ODBMs in the IMSplex to enhance performance and availability.

Operations Manager
At least one OM must be available in the IMSplex. You can define additional OMs in the IMSplex to enhance performance and availability.

Resource Manager
If any IMS in the IMSplex requires RM services, at least one RM must be available when the IMSplex is initialized. You can define additional RMs in the IMSplex for performance and availability enhancements if a resource structure is used. However, only one RM can be started in an IMSplex if a resource structure is not used. You can configure your IMSplex without an RM and use type-2 commands.

If an IMSRSC repository is enabled in the IMSplex, a CSL configuration with RM is required.

Structured Call Interface
One SCI is required on each z/OS image where an IMSplex member is running. Only one SCI is allowed on each operating system image for a given IMSplex.

Sample IMSplex configuration with CSL

The following figure illustrates a sample IMSplex configuration that includes the CSL, a SPOC, and automated procedures.

- The OS image includes addresses spaces for ODBM, OM, SCI, RM, an IMS control region, IMS Connect, and IMS CQS.
- The OS image shares a coupling facility and databases.
- A SPOC application, an automation application, a master terminal, and an end user terminal all access the OS image.

- The Open database resource adapters and APIs are also shown accessing the OS image through IMS Connect, the TCP/IP gateway for IMS.

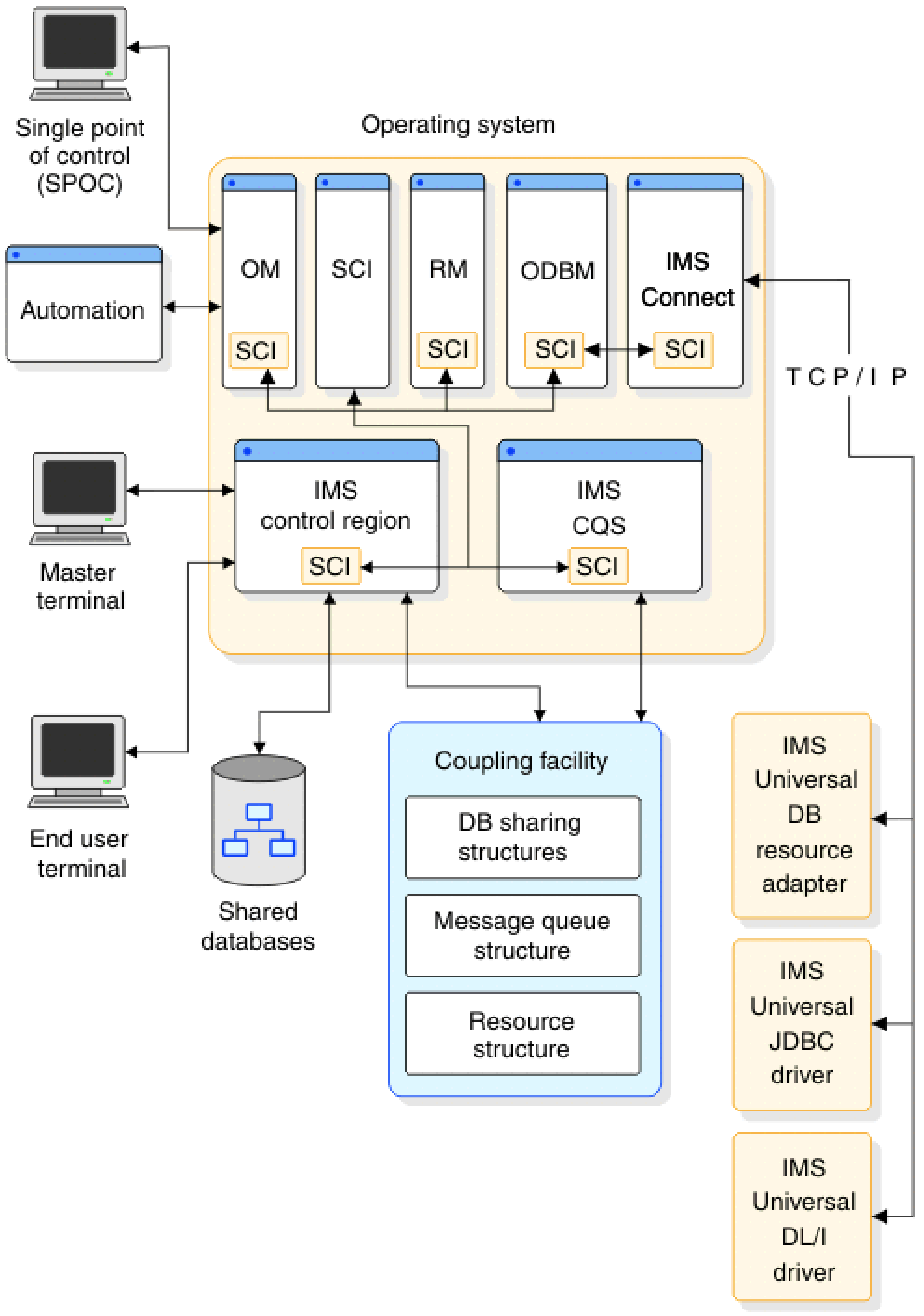


Figure 13. Sample IMSplex configuration with CSL

To obtain the fastest communication, define an OM and RM on each z/OS image; an IMSplex component can communicate more quickly with an OM or RM on the same z/OS image as that component rather than an OM or RM on a different z/OS image. However, this configuration also increases the number of address spaces on each z/OS image, which can create a more complex operating environment.

Note: If only one RM and only one OM are defined across an IMSplex, there is no backup to perform that manager's work in case of failure.

Recommendation: Define more than one RM, OM, and SCI across an IMSplex.

IMSplex minimum CSL configuration

The following figure illustrates the minimum configuration possible for the CSL in an IMSplex. Each OS image has an IMS control region and an SCI; in addition, the first OS image also has an OM, but no RM. RMENV=NO was specified on the startup parameter.

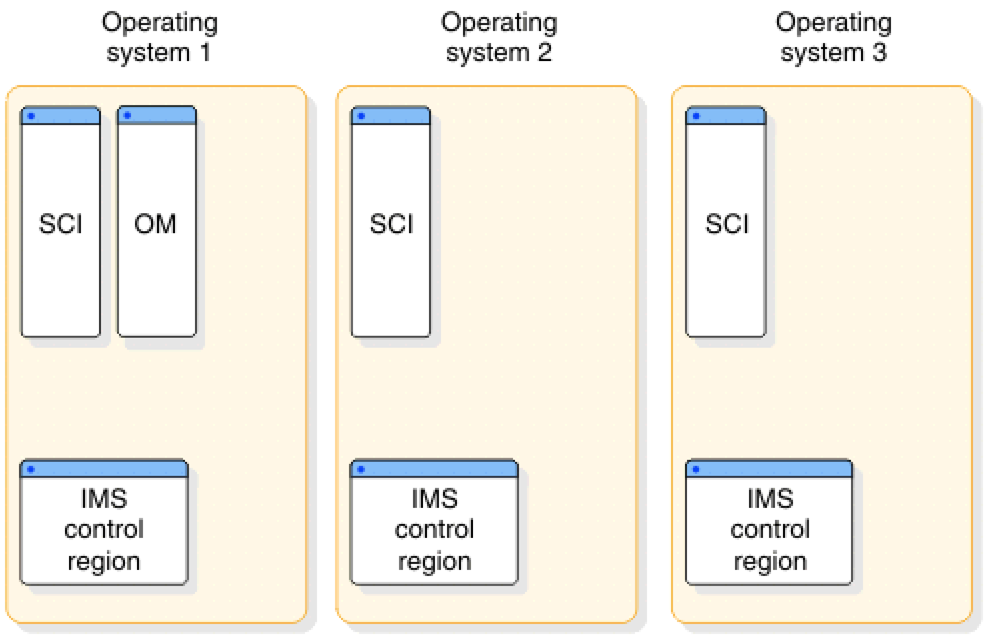


Figure 14. IMSplex minimum CSL configuration

In the figure, each z/OS image has a separate SCI since each has a distinct IMS control region. OM can reside on one z/OS image and still be used by other images in the IMSplex.

IMSplex mixed version CSL configuration

The following figure illustrates a more complex configuration having multiple versions of IMS within the IMSplex.

- Operating System 1 has IMS Version 12 control regions and is running OM and RM
- Operating System 2 has IMS Version 13 control regions, an ODBM, and an OM
- Operating System 3 has an IMS Version 11 control region

All three OS images share the coupling facility. Within the coupling facility is a resource structure. Therefore, an additional RM could be defined in another OS image.

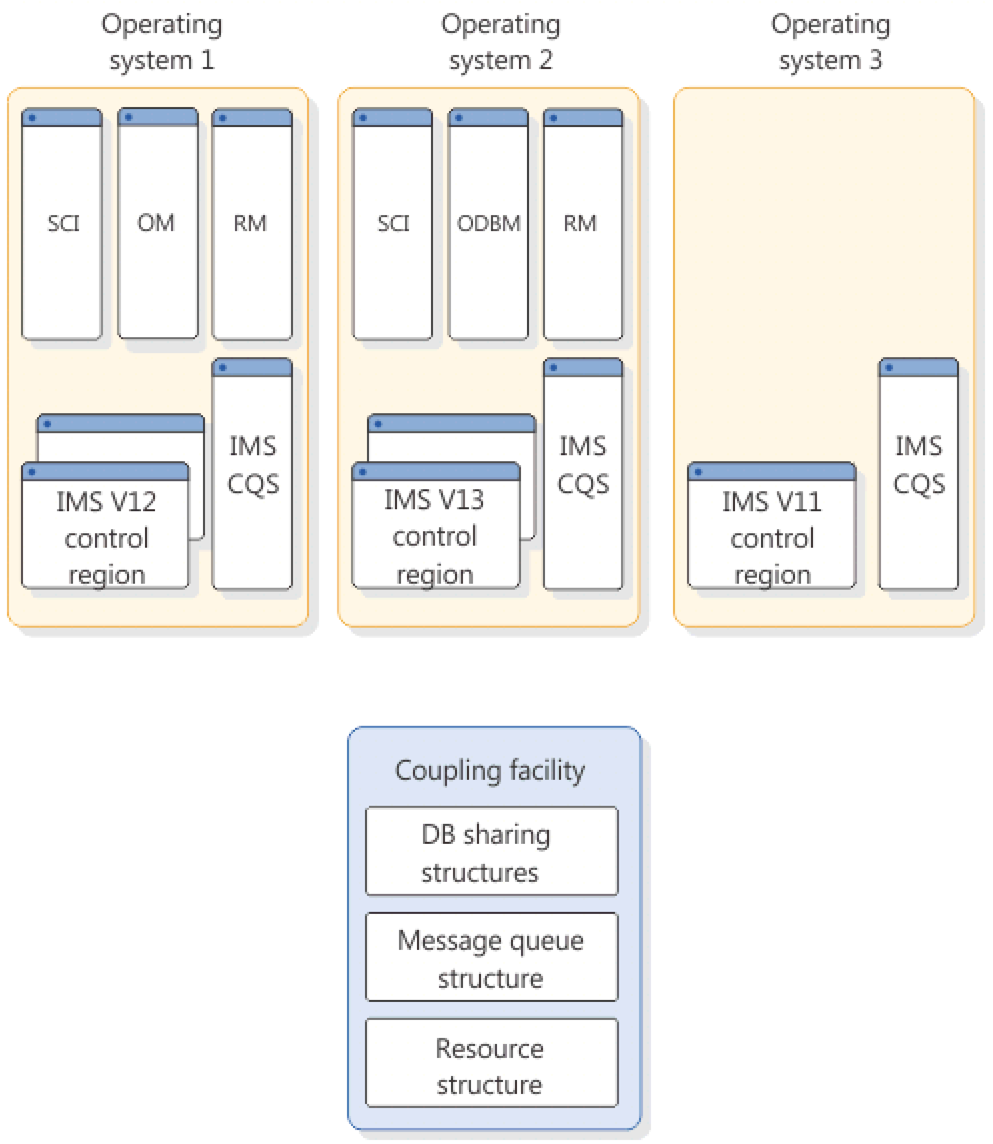


Figure 15. IMSplex mixed version CSL configuration

IMSplex DBCTL CSL configuration

The following figure illustrates the configuration of the Common Service Layer in a DBCTL environment. Here, three OS images each have an IMS DBCTL control region. All have an SCI and an OM. Two have an ODBM. Only one RM is allowed in the IMSplex, here in OS1, because no resource structure is defined.

Recommendation: Define a resource structure for DBCTL.

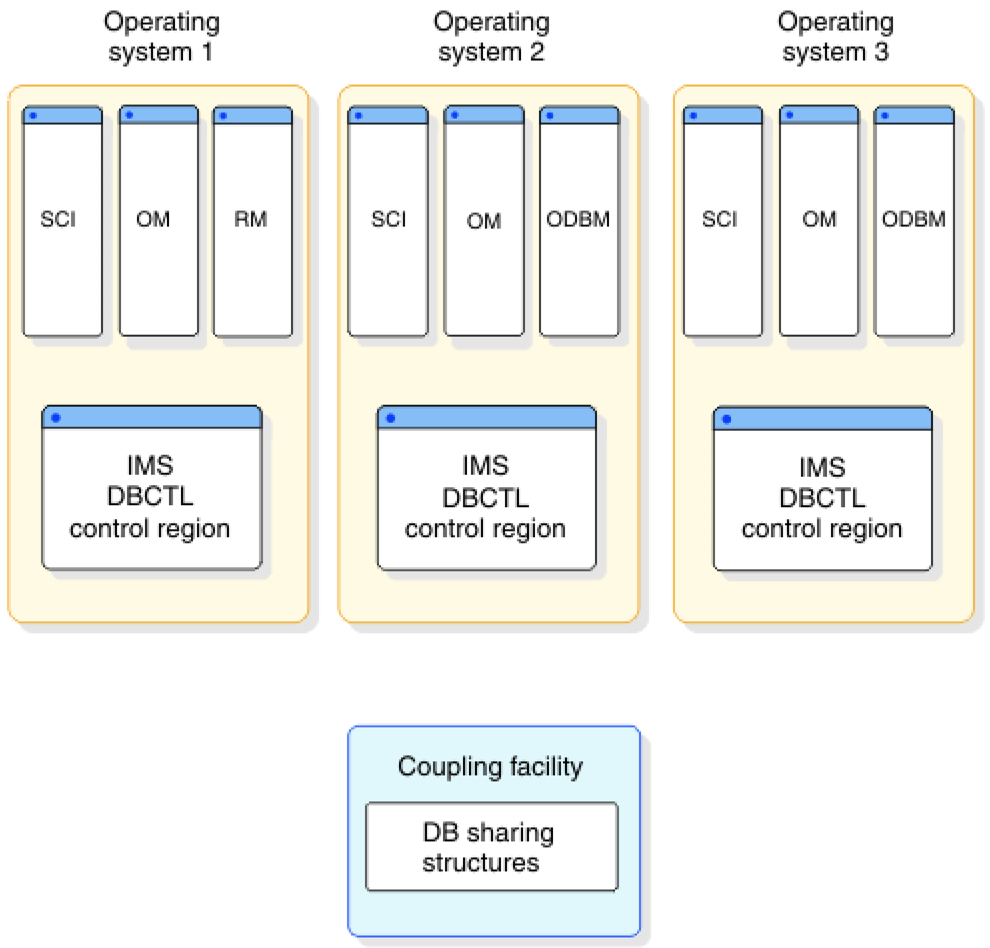


Figure 16. IMSplex DBCTL CSL configuration

In the figure, an OM is defined in each OS image to enhance overall system performance. If you do not use global online change, neither RM services nor an RM address space is required. In such a configuration, the CSL includes SCI and OM.

IMSplex minimum ODBM configuration

The following figure illustrates a minimum configuration for ODBM in an IMSplex that include three IMS systems. Each OS image has an IMS control region, an SCI, and an ODBM; in addition, the first OS image also has an OM, which is required by ODBM, but no RM. RMENV=NO was specified on the startup parameter.

The ODBM configuration shown is not a data sharing environment. Each IMS system manages its own databases separately.

Also, the databases on operating system 3 cannot be accessed by ODBM clients, such as the IMS Universal drivers, because an instance of ODBM is not running on operating system 3.

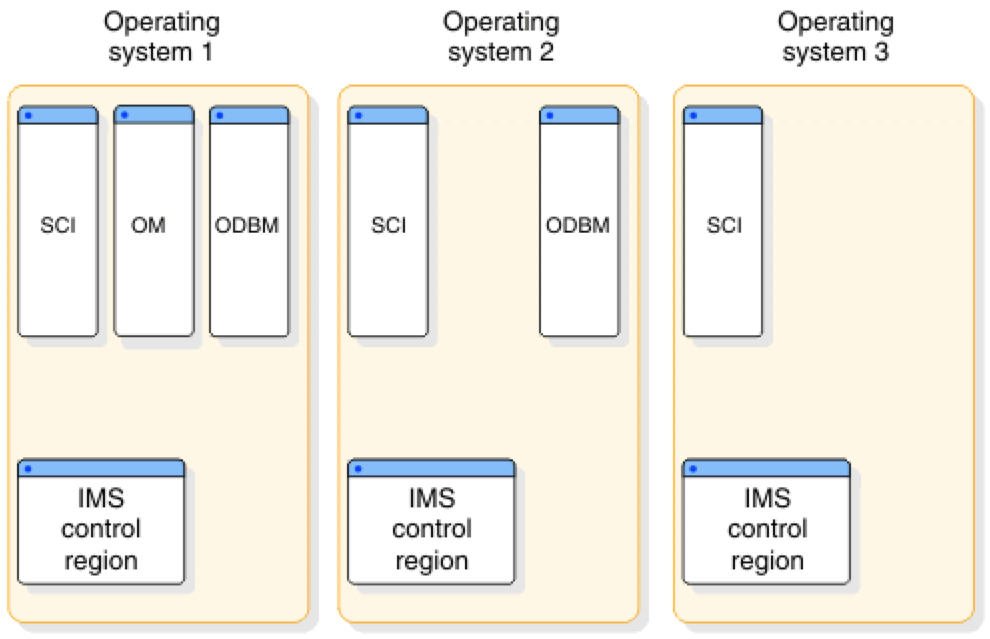


Figure 17. IMSplex minimum ODBM configuration

In the figure, each z/OS image has a separate SCI since each has a distinct IMS control region. OM can reside on one z/OS image and still be used by other images in the IMSplex. An instance of ODBM must be active on each z/OS image that contains databases to which ODBM clients require access.

Shared queues in an IMSplex without a CSL

In the following figure, three OS images are each defined with an IMS control region and IMS CQS. Each is associated with a Master Terminal Operator (MTO) console. All three OS images share a coupling facility that includes database sharing structures and a message queue structure. No CSL manager address spaces are defined.

This figure illustrate what a shared queues IMSplex environment looks like without a CSL.

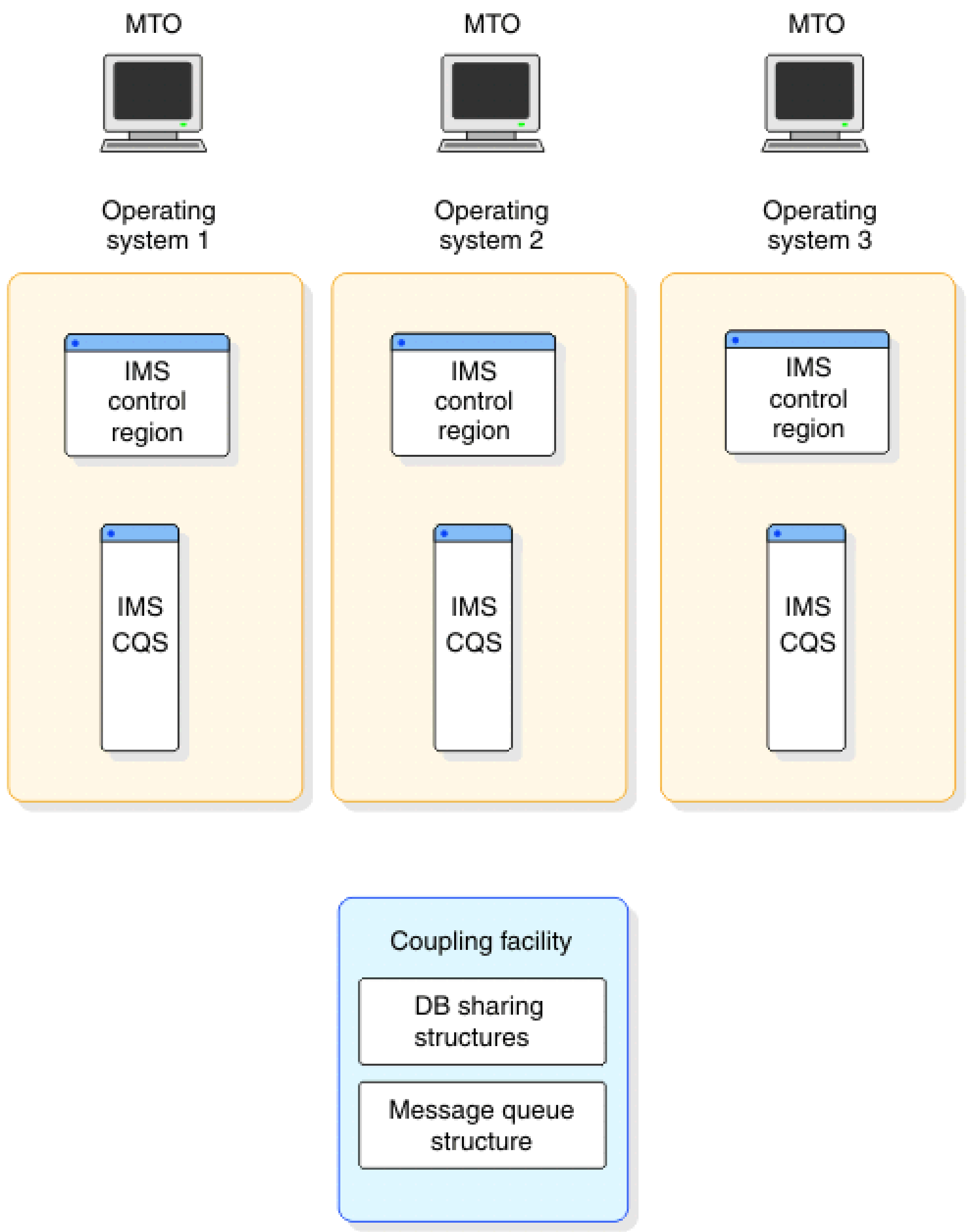


Figure 18. Shared queues in an IMSplex without a CSL

Shared queues in an IMSplex environment with a CSL

In the following figure, three OS images are defined. Each is identical, having an IMS control region, IMS CQS, and all CSL managers (SCI, OM, and RM). The three OS images share the coupling facility, which includes database sharing structures, a message queue structure, and a resource structure. A SPOC application can access any of the OS images.

This figure illustrates what a shared queues IMSplex environment looks like with a CSL.

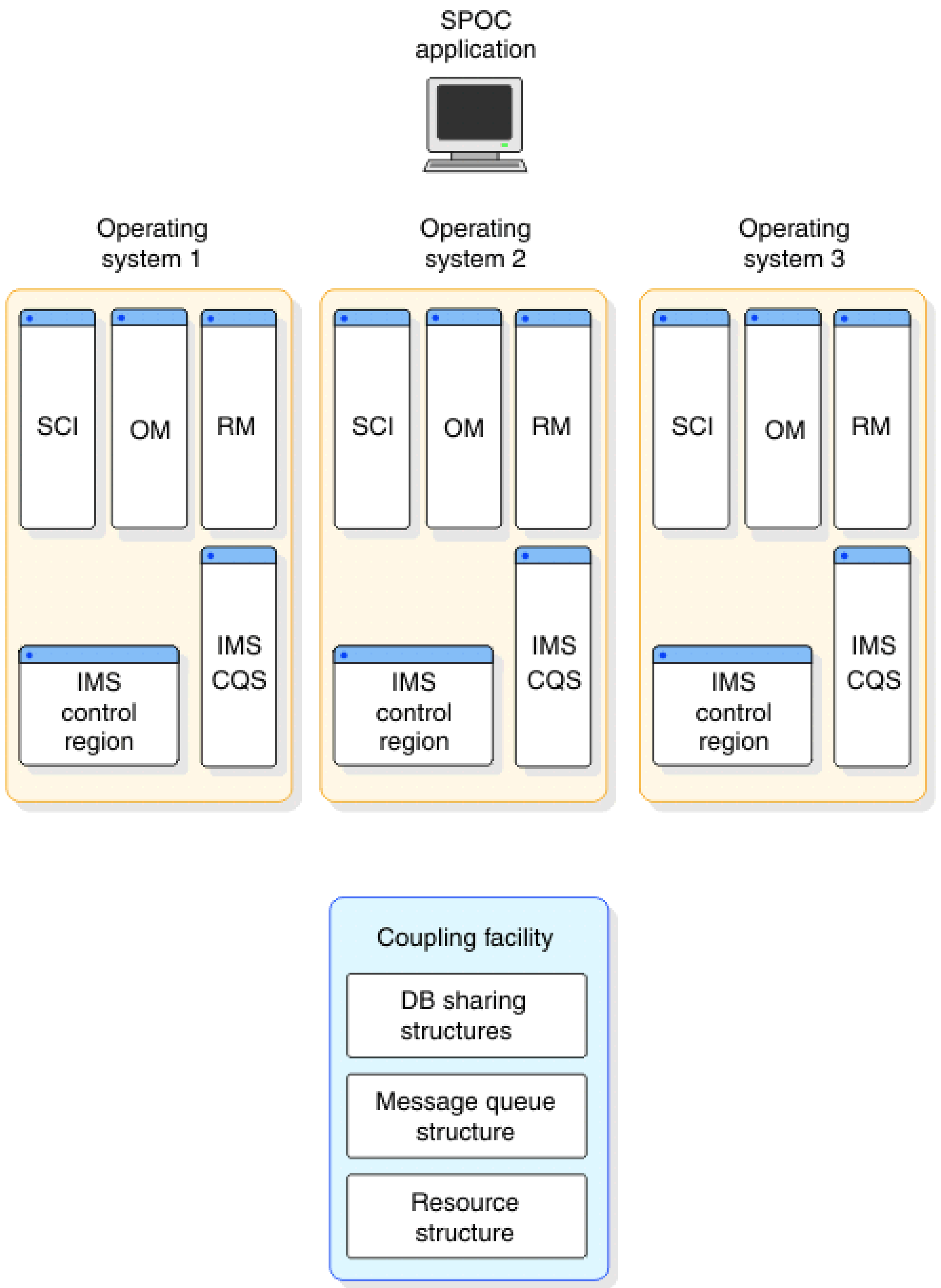


Figure 19. Shared queues in an IMSplex environment with a CSL

IMSplex configuration with an IMSRSC repository

The following figure illustrates a sample IMSplex configuration that includes an IMSRSC repository.

The following components comprise the repository: CSL, OM, RM, SCI, TSO SPOC or an automation application program, a Repository Server (RS), RS catalog repository data sets, and the IMSRSC repository data sets.

Optionally, the configuration can include a CQS address space and a z/OS cross-system coupling facility with a resource structure. CQS is required if RM is using the resource structure.

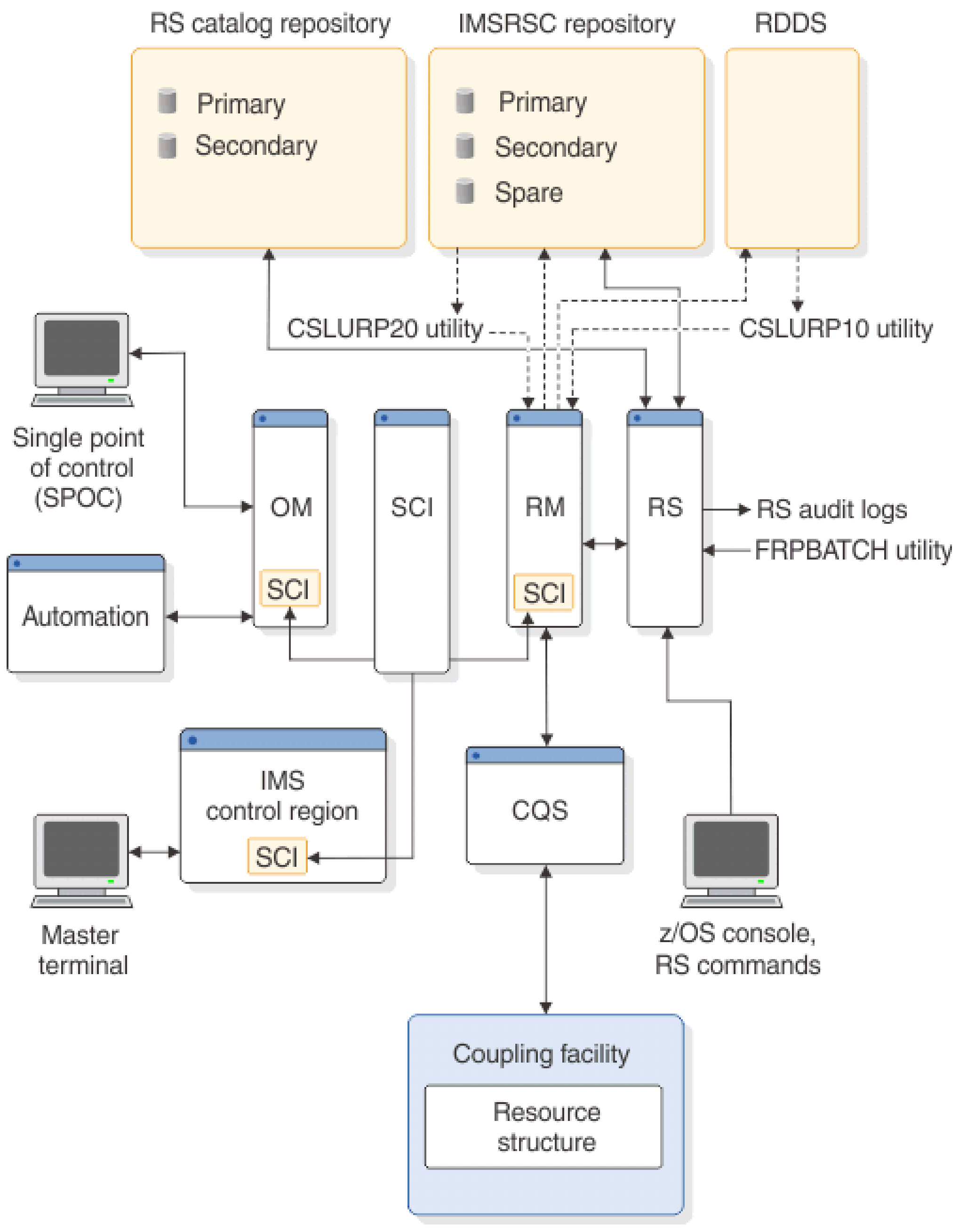


Figure 20. IMSplex configuration with an IMSRSC repository

IMSplex single system CSL configuration

In a single system IMSplex environment, each address space of the Common Service Layer is defined in a single OS image. This simple configuration is shown in the following figure. The OS image is defined with multiple IMS control regions, and one SCI, RM, OM, and ODBM.

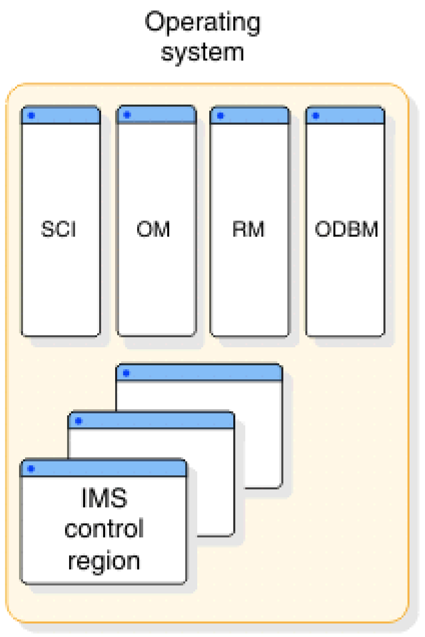


Figure 21. IMSplex single system CSL configuration

Related concepts

“Overview of the Operations Manager” on page 37
OM controls the operations of an IMSplex. OM provides an application programming interface (the OM API) through which commands can be issued and responses received. With a single point of control (SPOC) interface, you can submit commands to OM. The SPOC interfaces include the TSO SPOC and the REXX SPOC API. You can also write your own application to submit commands.

“Overview of the Resource Manager” on page 38
RM helps you manage resources that are shared by multiple IMS systems in an IMSplex. RM provides the infrastructure for managing global resource information and coordinating IMSplex-wide processes, such as global online change.

“Overview of the CSL Structured Call Interface” on page 39
SCI allows IMSplex members to communicate with one another. Communication between IMSplex members can occur within a single z/OS image or among multiple z/OS images. The individual IMSplex members do not need to know where the other members reside or what communication interface to use.

Overview of the IMSRSC repository (System Definition)

Related tasks

“Defining a simplified IMSplex for the type-2 command environment” on page 22
To define a simplified IMSplex without RM either specify RMENV=N in the DFSCGxxx member of the IMS PROCLIB data set or specify RMENV=N in the CSL section of the DFSDFxxx PROCLIB member while defining an IMSplex.

Common Queue Server overview

Common Queue Server (CQS) is a generalized server that manages data objects on a *coupling facility list structure*. CQS manages two types of structures: *queue structures* and *resource structures*. CQS receives, maintains, and distributes data objects from these structures on behalf of clients. IMS is one example of a CQS client that uses CQS to manage both its shared queues and shared resources.

This topic contains General-use Programming Interface information.

CQS uses the z/OS coupling facility as a repository for data objects. Storage in a coupling facility is divided into distinct objects called *structures*. Authorized programs use structures to implement data sharing and high-speed serialization. The coupling facility stores and arranges the data according to list structures. Queue structures contain collections of data objects that share the same name, which are known as *queues*. Resource structures contain data objects that are organized as uniquely named resources.

CQS runs on z/OS. The CQS client must also run on z/OS. CQS runs in a separate address space that can be started by the client.

Clients communicate with CQS by using CQS requests that are supported by CQS macro statements. Using these macros, CQS clients can communicate with CQS and manipulate client data on shared coupling facility structures. The following figure shows the communications and the relationship between clients, CQSs, and the coupling facility.

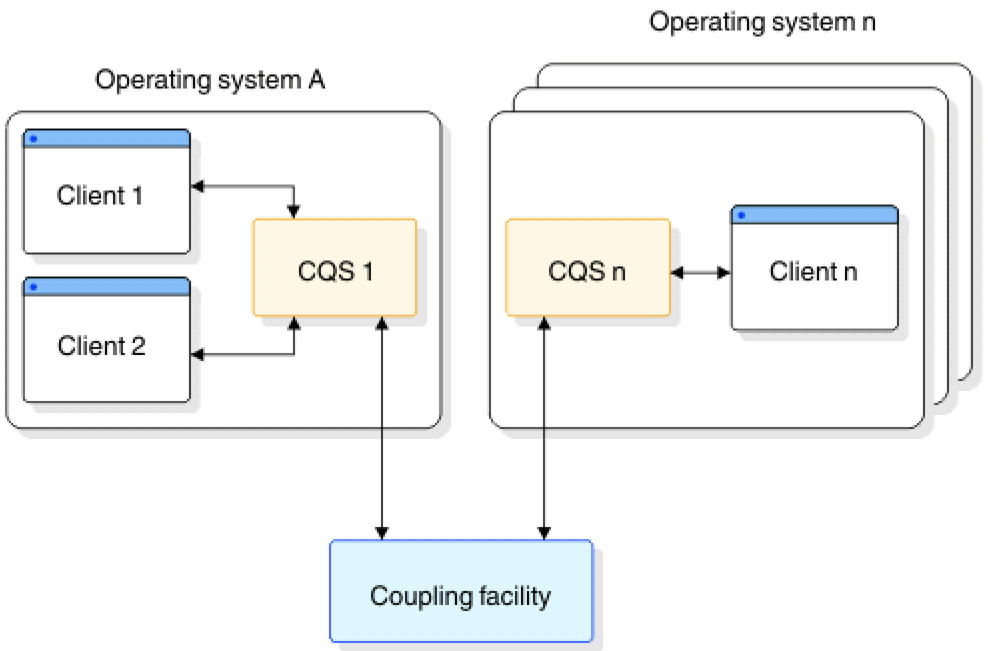


Figure 22. Client systems, CQS, and the coupling facility

Related reading:

- CQS requests are described in *IMS Version 15.2 System Programming APIs*.
- See *z/OS MVS Setting Up a Sysplex* for complete details about setting up a sysplex.

CQS benefits

CQS enables users to take advantage of the benefits of a Parallel Sysplex (sysplex) environment. These benefits include:

Automatic work load balancing
CQS places data objects on shared queues where they can be processed by any participating client system.

Any participating client system can use CQS to retrieve a data object from the shared queues.

Incremental growth
Customers can add new systems as workload increases.

Reliability
For both shared queues and resources, if one client system fails, the remaining client systems process the work.

CQS components

CQS uses the following components:

Primary structure
A z/OS coupling facility list structure that contains shared queues.

Resource structure
A z/OS coupling facility list structure that contains uniquely named resources.

Overflow structure
A z/OS coupling facility list structure that contains shared queues when the primary structure reaches an installation-specified overflow threshold. The overflow structure is optional.

z/OS log stream
A shared z/OS log stream that contains all CQS log records from all CQSs connected to a structure pair. This log stream is important for recovery of shared queues, if necessary. Each structure pair has an associated log stream.

Checkpoint data set
A local data set that contains CQS system checkpoint information.

Structure recovery data sets (SRDSs)
Shared data sets that contain structure checkpoint information for shared queues on a structure pair. Each structure pair has two associated SRDSs.

CQS functions

CQS performs the following functions:

Processes CQS requests
An architected interface that clients use to access CQS or data objects on a queue structure or a resource structure.

Notification of work on a queue
Clients register interest in the shared queues. If an empty queue becomes non-empty, CQS notifies its registered clients.

Records restart and recovery information
CQS records all the information necessary for restart and recovery in the z/OS system logger.

CQS system checkpoint
CQS system checkpoint writes log records relating to a particular CQS to the CQS log. The log records contain information necessary for CQS to restart and recover work.

Structure checkpoint
Structure rebuild
Structure rebuild is a z/OS process that allows another instance of a structure to be allocated with the same name and data reconstructed from the initial structure instance.

Overflow processing
CQS provides an overflow option to help prevent a queue full condition. When the primary list structure reaches the overflow threshold value, CQS attempts to dynamically increase the size of the primary structure, offloads selected queues to an overflow structure, or rejects requests for selected queues.

CQS requirements

The CQS has coupling facility requirements that are documented in the *IMS Version 15.2 Release Planning*.

Related reading: See *z/OS MVS Setting Up a Sysplex* for more information on:

- The planning required to migrate to a sysplex that uses a coupling facility
- Hardware configurations of a sysplex
- The software products that can use the coupling facility
- The tasks for migrating to a coupling environment
- Checklists for installing the sysplex hardware and software

Queue structures

A queue structure is a coupling facility list structure that contains a collection of data objects, some of which might have the same name. Data objects that have the same name are considered to be on the same queue.

Queue structures support structure overflow, in which an associated overflow structure can be allocated to prevent the queue structure from becoming full. A primary queue structure and its associated overflow structure are known as a structure pair.

CQS physically divides the queue structure list headers into 11 private queue types for CQS use and 11 client queue types for client use. Client queue types are defined by the client. A client can group queues associated with a type of work, such as transactions. A queue type can have a value of 1 to 255. Any queue type over 11 is mapped into one of the physical queue types.

CQS manages private queues and client queues on queue structures. CQS uses the private queue types to manipulate client data objects for CQS requests. Each client queue type can be used by a client for a different type of work. A client registers interest in only those queue types that it can process, based on the types of work you define for it.

Private queue types managed by CQS

The following table shows five of the private queue types, and the work that a client processes on them.

Queue type	Description
Cold queue	Contains data objects that are in doubt for a client or for a CQS that cold started
Control queue	Contains control list entries that CQS uses to manage list structures and control processes (such as structure checkpoint and structure recovery)
Delete queue	Intermediate queue used for CQSDEL request processing
Lock queue	Contains data objects that are locked by the CQSREAD request
Move queue	Intermediate queue used for CQSMOVE request processing

Resource structures

A *resource structure* is a coupling facility list structure, used by the Common Service Layer's Resource Manager and managed by CQS, that contains uniquely named resources.

This structure is typically used to maintain global resource information when multiple Resource Managers exist in an IMSplex. Resource structures enable CQS to perform resource management in an IMSplex.

CQS physically divides the resource structure list headers into 11 private resource types for CQS use and 11 client resource types for client use. Client resource types are defined by the client. A resource type can have a value of 1 to 255. Any resource type over 11 is mapped into one of the physical resource types.

Clients can use the resource structure to share resource information, control block information, and other types of information. The resource name is unique within the structure. Resources can be updated, queried, or deleted. A primary coupling facility list structure is used to contain the resources.

CQS structure functions

CQS provides functions for monitoring structure status and capacity, and enabling structure recovery. Some of these functions are built-in functions and do not require intervention. Other functions are optional, and can be set up or initiated as your installation needs them.

Structure overflow

CQS provides a structure overflow function that automatically warns you when a queue structure is approaching full and takes action to prevent a full structure. When the usage of a structure reaches the overflow threshold, CQS attempts to make the structure larger by initiating a structure alter. If the alter fails, CQS either allocates an overflow structure and moves selected queues to the overflow structure (if you define an overflow structure), or prevents new data objects from being put on the selected queues.

Important: Overflow processing is not supported for resource structures.

The CQS client requests, CQSPUT and CQSDEL, can provide information about current structure utilization through the FEEDBACK= and FEEDBACKLEN= parameters. IMS uses this capability to provide structure utilization information about the IMS shared message queue structure to the Queue Space Notification exit routine for shared queues (DFSQSSP0). The structure utilization information contains the total number of allocated and in-use entries and elements in the primary and overflow structures. This information helps you prevent the message queue structure from becoming full. Note that DFSQSSP0 provides only information about the IMS shared message queue structure, not the IMS shared EMH structure.

Structure rebuild

Structure rebuild is a z/OS process that allows another instance of a structure to be allocated with the same name and contain data reconstructed from the initial structure instance. z/OS supports system-managed rebuild, in which case z/OS rebuilds the structure. z/OS also supports user-managed rebuild; the user rebuilds the structure. Structure rebuild can be initiated manually by using an operator command, or automatically by CQS or z/OS.

CQS allows system-managed rebuild for queue structures and resource structures. CQS provides user-managed rebuild to support a structure copy function and a structure recovery function.

Structure copy copies the contents of a structure to another structure for planned reconfiguration. Structure copy is supported for resource structures and queue structures.

Structure recovery recovers a structure from the structure checkpoint data set and the CQS log after a structure failure. Structure recovery is supported for queue structures.

Structure duplexing

CQS can use the *duplexing* capabilities of z/OS. Duplexing occurs when the operating system creates a duplex (backup) copy of a structure, then maintains the two structures during normal mainline operation. If a structure fails, or a connection to a structure is lost, the operating system switches to the unaffected structure instance.

Related concepts

“Preventing CQS structure full” on page 169
You should manage structure usage to avoid a structure full condition. If a resource structure or queue structure becomes full, CQS issues message CQS0205E.

“Rebuilding structures in CQS” on page 172

Structure rebuild is a z/OS process that allows another instance of a structure to be allocated with the same name and data reconstructed from the initial structure instance. z/OS supports system-managed rebuild, CQS-managed rebuild, and structure duplexing.

CQS recovery functions

CQS provides functions for recovering work-in-progress, queues, and resources in case of system shutdown or failure. Some of these recovery functions are built-in and do not require intervention.

Other functions are optional and can be set up or initiated as you need them.

System checkpoint

To enable CQS restart in the event of failure, CQS periodically takes a "snapshot" of all control blocks and tables, and writes that information to the z/OS log. That process is called *system checkpoint*. System checkpoint can be initiated by CQS, the client, or manually with an IMS command.

CQS logging and the z/OS system logger

CQS uses the z/OS system logger to record information necessary for CQS to recover queue structures and restart. CQS writes log records for each recoverable coupling facility list structure pair that it uses to a separate log stream. The log stream is shared among all CQS address spaces that share the structure. The system logger provides a merged log for all CQS address spaces that are sharing queues on a coupling facility list structure.

Important: Changes to resource structures and nonrecoverable (RECOVERABLE=NO) queue structures are not logged.

Structure checkpoint

To enable queue structure recovery in case of failure, CQS periodically takes a ”snapshot” of the queues on all queue structures. That process is called *structure checkpoint*. Structure checkpoint can be initiated by CQS, the client, or manually with an IMS command.

Important: Structure checkpoint is not supported for resource structures.

CQS client requests

CQS client systems communicate with CQS using a general use interface consisting of CQS requests.

Related concepts

“Recording information necessary for starting CQS” on page 165
CQS uses the z/OS system logger to record all information necessary for CQS to recover queue structures and restart. CQS writes log records for each coupling facility list structure pair that it uses to a separate log stream. The log stream is shared among all CQS address spaces that share the queue structure.

“Using CQS structure checkpoint” on page 168
Structure checkpoint takes a snapshot of the shared queues on a queue structure and writes the data to the structure recovery data set (SRDS) so that CQS can recover the queues after a structure failure. Structure checkpoint processing copies all recoverable data objects from a structure pair to a SRDS.

CQS client requests (System Programming APIs)

Type-2 command environment

The type-2 command environment requires a Common Service Layer (CSL) with Operations Manager (OM) and the Structured Call Interface (SCI). A Resource Manager (RM) is not required. Any IMSplex (including a single-IMS IMSplex) with a CSL constitutes a type-2 command environment.

The type-2 command environment can include a single, standalone IMS system, multiple IMS systems that do not share resources, or a full-feature IMSplex that includes RM. The type-2 command environment can also be used with DB/DC, DCCTL, and DBCTL environments.

Some type-2 commands (such as the type-2 commands associated with the Open Database function) have additional requirements.

With a type-2 command environment, you can also use OM command security. You might also need to consider SCI security in a type-2 command environment.

Type-2 commands must be issued from an automated operator application program, such as the TSO single point of control (SPOC) application that is shipped with IMS. IMS offers two interfaces that such application programs can use to communicate with IMS: the Operations Manager (OM) application programming interface (API) and the REXX SPOC API.

Related concepts

"IMSplex overview" on page 16
An IMSplex is made up of IMS and z/OS components that work together.

Controlling IMS with the TSO SPOC application (Operations and Automation)

REXX SPOC API environment with the CSL OM (System Programming APIs)

Related tasks

"Defining a simplified IMSplex for the type-2 command environment" on page 22
To define a simplified IMSplex without RM either specify RMENV=N in the DFSCGxxx member of the IMS PROCLIB data set or specify RMENV=N in the CSL section of the DFSDFxxx PROCLIB member while defining an IMSplex.

Related reference

IMS type-2 command format (Commands)

Extended Terminal Option

The Extended Terminal Option (ETO) is a function of IMS TM that can be included at system definition.

With ETO:

- You can add VTAM terminals to IMS without redefining the system.
- You can dynamically add user LTERMs and remote LTERMs (for MSC links) to IMS.
- You cannot define the master terminal using ETO.
- You cannot use the XRF surveillance link with ETO.

Related reading:

- For more information on specifying ETO at system definition, see *IMS Version 15.2 System Definition*.
- For information on ETO system security features, see Chapter 19, "IMS security," on page 285.

APPC

IMS supports advanced program-to-program communication (APPC) conversations in two scenarios: APPC/IMS and the explicit CPI-C driven interface.

The two scenarios differ in the subsystem that manages the updating and synchronization of protected resources that the application program accesses.

In the APPC/IMS scenario, when SYNCLVL = NONE or SYNCLVL = CONFIRM, IMS is the synchronization-point manager. When SYNCLVL = SYNCPT, z/OS Resource Recovery Services (RRS) is the synchronization-point manager.

In the CPI-C driven scenario:

- The sync point manager is the RRS function of z/OS.
- The resource manager is IMS.
- The program that accesses and updates the protected resources is the APPC/MVS application program.

APPC/IMS defines the formats and protocols for program-to-program communication. APPC/IMS enables applications to be distributed throughout the network and to communicate with each other regardless of the underlying hardware architectures and software environments. APPC/IMS provides a facility for implementing logical unit type 6.2 (LU 6.2) support.

APPC/MVS is used for all interaction with remote LU 6.2 devices or subsystems. IMS accesses the session through APPC/MVS services. Using APPC/IMS, IMS and LU 6.2 devices access each other without requiring coding changes to existing application programs. With slight modifications to IMS application programs, Common Programming Interface (CPI) communications-driven application programs can communicate with IMS application programs (and can execute as IMS application programs).

A restriction exists, however, on LU 6.2 synchronous conversations with implicit transactions. If a transaction creates more than one daughter transaction, which, in turn, might create other transactions, and one of the daughter transactions provides the response, then the result is unpredictable. In some cases, depending on the execution sequence of the transactions, the LU receives a DFS2082 message and the response is sent to the default TP name DFSASYNC. In other cases, the LU receives the response and no DFS2082 message is issued.

APPC/IMS flood control

APPC/IMS includes an optional flood control function that queues APPC requests in 64-bit storage if the number of active APPC conversations exceeds a flood threshold that is defined on the APPCMAXC= parameter in the DFSDCxxx PROCLIB member. The default threshold is 5,000 active APPC conversations.

IMS stops all APPC input from z/OS, including input for both APPC/IMS and CPI-C, if either the number of APPC requests that are queued in 64-bit storage reaches the maximum number that is defined in the second value position of the APPCMAXC= parameter or if queueing in 64-bit storage is disabled and the maximum number of active APPC conversations is reached.

Related reference

DFSDCxxx member of the IMS PROCLIB data set (System Definition)

Related information

3303 (Messages and Codes)

Security for dependent region processing

Although security checking can be carried out by terminal, transaction, command, and other types of authorization, you can also implement security by limiting the resources that application programs that are scheduled in dependent regions can access using resource access security (RAS).

RAS uses RACF®, RACF security classes, and user IDs to define resources and the dependent regions that can use those resources. To implement RAS security, you must define in the RACF security classes resource profiles for the transactions, PSBs, and LTERMs that you want to protect. You must also specify in the resource profiles the user IDs of each dependent region that you want to allow to use each resource.

RACF is an external security product to IMS, accessed by IMS using the Security Access Facility (SAF). RACF is licensed with the IBM z/OS Security Server. Where this information directs you to use RACF, you can use a different, equivalent security product if you choose.

When an application program executing in a dependent region attempts to access a resource, RACF checks the resource's security class profile to see if the user ID of the dependent region in which the application resides is authorized for that resource. If the resource profile lists that user ID, RACF allows access; if not, RACF denies access.

You can also use exit routines, such as the Resource Access Security user exit (RASE) with RAS, which allows you to customize the security checking for dependent region processing.

MPP scheduling in DB/DC and DCCTL environments

After an MPP region has been initialized, it can execute an application program within its virtual storage. By using the scheduling algorithm, the control program selects a message for processing. Using the transaction code, the appropriate application program is loaded into dependent region storage from the IMS.PGMLIB data set.

The application program is identified by the PSB name that was declared in system definition to be associated with that transaction code. The convention used by IMS TM for MPPs is that the application program name is the same as the PSB name. The first message segment is then made available from the message queue, and control is passed to the application program.

The scheduling algorithm also controls the amount of processing performed by the application program. You can specify a limit to the number of messages processed in the scheduling of one program. When this number is reached, IMS TM does the following:

- If any equal or higher priority transactions are queued, IMS TM terminates the application program. The region becomes available for another program to be scheduled into its storage. IMS uses the scheduling algorithm to choose the program to schedule.
- If no equal or higher priority transactions are queued and messages are still queued for the current application program, the region goes through quick reschedule and returns the next message to the application program.
- If equal priority transactions are enqueued, IMS allows the transaction to quick reschedule if the PROCLIM value has not been reached. If the PROCLIM value has been reached, IMS disallows the quick reschedule and processes the other transactions.

  Quick rescheduling is also affected by the IMS scheduling algorithm, which takes into account the following factors and more:

  - MAXRGN value
  - PARLIM value
  - Current number of regions in which the transaction is scheduled
- If no more messages exist for the scheduled transaction, IMS TM determines if other work for the region is ready to be processed.
- If no additional work is ready to be processed, IMS TM determines if the region can become pseudo wait-for-input (pseudo WFI). This determination causes one of the following actions:
  - If the region is eligible for pseudo WFI, the region remains scheduled for the transaction and waits until another message is entered for the region. If the next message is for the scheduled transaction, the message is passed to the application program. If the next message is for a different transaction, IMS TM terminates the application program and schedules a new application program to process the new message.
  - If the region is not eligible to become a pseudo WFI, IMS TM tells the application program that no more messages exist, and the application program terminates.

The master terminal operator directly schedules batch message programs by the entry of the JCL to start the batch message region. The program and PSB to be used are explicitly given in the EXEC statement.

Scheduling application programs against unavailable data

IMS Transaction Manager schedules an application program even when some of the full-function databases that the application program can access are not available. When dealing with unavailable data, the application program can be sensitive or insensitive to data unavailability.

Application programs that are sensitive to unavailable data issue an INIT call when IMS schedules the applications. The INIT call tells IMS to return a status code in the PCB when the data that the application program requires is not available. The program can then take the appropriate action.

Application programs that are insensitive to unavailable data do not issue an INIT call. If an application program that is insensitive to unavailable data attempts to access data that is not available, IMS terminates the application program with a user abend code 3303 and backs out any updates that the application program has made. If an application program generates ten 3303 abends, IMS prevents further rescheduling of the application program by stopping its PSB.

If an application program terminates as a result of unavailable data, IMS places the input message that the application program was processing on the suspend queue. A separate suspend queue exists for each transaction type. Serial transactions are not placed on the suspend queue, but rather are returned to the normal queue as the next message to be processed and are USTOPPED.

If IMS determines that most messages are failing and being placed on the suspend queue, IMS stops processing that transaction type. When the transaction is started, or when a database used in processing the transaction is started, the messages on the appropriate suspend queues are transferred to the normal queue, and another attempt is made to process the message.

Related reading:

- For additional information about scheduling application programs against unavailable data, see *IMS Version 15.2 Application Programming*.
- For more information on the suspend queue, see "Scheduling transactions using the suspend queue" in *IMS Version 15.2 System Definition*.

Fast Path

Use Fast Path to improve performance for simple transactions. When data communication requirements are for a high transaction volume with rapid database updates and inquiries, the Fast Path facilities offer several advantages over full-function DL/I processing.

Examples of application programs with these requirements are the teller transactions in banking and point-of-sale transactions (inventory update) in retail marketing. Fast Path input and output messages use the expedited message handler (EMH), bypass message queuing, and priority scheduling. Most terminals have Fast Path execution potential. However, terminals that cannot run in response mode do not have Fast Path potential.

For a DB/DC environment, Fast Path requires the Database Manager and Transaction Manager and becomes an integral part of the IMS online system. The control program manages concurrent processing of Fast Path and DL/I programs.

The DCCTL environment supports Fast Path processing and transactions, but not Fast Path databases.

Related reading:

- For more information on the design, definition, initialization, monitoring, or tuning of databases used with Fast Path, see *IMS Version 15.2 Database Administration*.
- For information on Fast Path application programming, see *IMS Version 15.2 Application Programming*.

Fast Path databases

In addition to DL/I databases, two other database types are available with Fast Path: the main storage database (MSDB) and the data entry database (DEDB). These databases are designed for the kinds of application programs that require high availability. The two types offer a choice of either rapid response within high activity or partitioned access within a large volume of data.

Related reading: For a detailed description of the design advantages and implementation of these Fast Path databases, see *IMS Version 15.2 Database Administration*.

Dependent region use for Fast Path

The majority of Fast Path processing programs are similar in function to message processing programs (MPPs). Message-driven programs correspond to MPPs, and execute in a Fast Path dependent region (IFP region). These programs execute in wait-for-input mode, so that the program execution is equivalent to a dependent region operation. Parallel scheduling is supported, so that another copy of the program can execute in another dependent region.

Because of the ability to perform much of the data entry database maintenance online, such as reorganization and recovery-related functions, your IMS online system should allow for the scheduling of a Fast Path utility region.

Fast Path application programs and utilities can be active concurrently with message processing programs or BMPs. An IMS online system that is using Multiple Systems Coupling (MSC) can also be processing Fast Path transactions. However, message input received through MSC links cannot be directed to a Fast Path application program, nor can they be passed to the Fast Path input exit routine. This restriction does not apply to message input received using Intersystem Communication (ISC) connections.

Fast Path transactions

A Fast Path application program is driven by transactions that bypass IMS input message queue handling. A transaction can be declared to be Fast Path exclusive. After initial edit, the input message is passed to an exit routine. This routine helps determine the dependent region in which the transaction is executed. The message is added to a Fast Path message-handling area in the program's storage, and then the transaction is made available to the message-driven program without I/O to the message queues.

A transaction can also be declared as having Fast Path potential. After entry, the transaction is also passed to the user exit routine, which decides whether the transaction should pass directly to the message-holding area in the control program's storage, or whether it should be routed to IMS for normal message queue handling. The queue bypass again leads to the transaction being presented to a message-driven program.

The control of Fast Path messages within the storage of a control program is called expedited message handling. One of the checks ensures that messages meet the requirement that single-segment input and output messages are used. The Fast Path Input Edit/Routing exit routine is DBFHAGU0. IMS can use EMH buffers for complete input editing of both Fast Path and full-function transactions.

Fast Path in a DBCTL environment

In a DBCTL environment, Fast Path provides improved performance and data availability to programs that can use data entry databases (DEDBs). A DBCTL environment supports only the Fast Path facilities that are related to DEDBs. You cannot run main storage database (MSDB) facilities.

BMPs or CCTL threads can schedule a PSB to access DEDBs. Parallel scheduling is supported; another copy of the PSB can execute in another BMP or CCTL thread. Fast Path application programs and utilities can be active concurrently with BMPs.

Because much of the DEDB maintenance (such as reorganization and recovery-related functions) can be performed online, your IMS DBCTL environment should allow for the scheduling of a Fast Path utility region.

IMS function levels overview

An IMS function level identifies a PTF that contains one or more IMS functions. The function level is a full word hexadecimal number, which is represented as decimal in output for the **QUERY IMSFUNC** command and in messages.

The following types of IMS functions are identified by using IMS function levels:

- A PTF that contains one small, usable piece of an overall function
- A new, significant function that is delivered enabled
- Previously optional functions that are changed to be mandatory
- Large, complex fixes

You can see the current IMS function level at the following locations:

- The IMS checkpoint ID table, which includes the function level in the BCPFLVL field
- The IMS functions enabled bitmap

You can also use the **QUERY IMSFUNC** command to see the current IMS function level and the function level at which a function was added.

The IMS functions enabled bitmap, which is mapped by the not-OCO DFSFNCE macro, contains the current IMS function level and a bit for every IMS function that is enabled. You can access the bitmap by using the SCD field SCDFNCE. IMS components, including but not limited to IMS database, IMS catalog, MSC, Fast Path, OTMA, and IMS scheduling, might also store an extra function enablement bit in a component control block. When two function enablement bits are defined, one in the IMS functions enabled bitmap and one in the component control block, IMS cold start sets both bits based on the PROCLIB member function enablement parameter and UPDATE IMSFUNC sets (or resets) both bits based on the SET parameter specified.

If an IMS function is not enabled by default, you can enable the function by using one of the following methods:

- Statically by specifying a parameter in the DFSDF*xxx* member of the IMS PROCLIB data set and cold starting IMS.
- If SCI and OM are enabled, dynamically by using the **UPDATE IMSFUNC** command after the PTF containing the function is applied to an IMS system for local functions or to all IMS systems for global functions.

IMS function names and levels

The following table shows function information that you can use to run the **QUERY IMSFUNC** or **UPDATE IMSFUNC** commands.

Function name	Function level	DFSFNCE function-enabled bit
FUNCTIONLEVEL	00000001	None
DEDBGT2K	00000002	FNCE_DEDBGT2K

Related tasks

"Enabling and disabling IMS functions" on page 421
If an IMS function is not enabled by default, you can enable the function dynamically while IMS is running by issuing the **UPDATE IMSFUNC** command. Or, you can enable the function statically by defining the function as enabled in the DFSDF*xxx* member of the IMS PROCLIB data set, and then cold starting IMS.

Related reference

QUERY IMSFUNC command (Commands)
UPDATE IMSFUNC command (Commands)

Automated operator application programs

An application program that can issue a subset of IMS operator commands using the DL/I CMD or ICMD calls is called an automated operator (AO) application program.

AO application programs can use the CMD call in only the DB/DC and DCCTL environments.

AO application programs can use the ICMD call in all environments.

When the CMD or ICMD call is issued, the operator command is executed and the first segment of the command response is put in the AO application program's I/O area. Subsequent parts of the response are obtained by GCMD (if CMD is issued) or RCMD (if ICMD is issued) calls.

To maintain security, you need to decide which AO application programs can issue operator commands and which commands they can issue. An AO application program can issue a single command or a series of commands.

Related reading:

- For information on securing the CMD and ICMD calls, see Chapter 19, “IMS security,” on page 285.
- For more information on automated operator (AO) application programs, see *IMS Version 15.2 Operations and Automation*.

When using an AO application program it is also possible to automate the recovery of databases and areas by using the IMS AO facility in conjunction with a database recovery service. One way that you can control recovery is by using the ICMD call to issue **/RECOVER** commands in a type 2 AO application program.

Related reading: For more information on the **/RECOVER** commands, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

System logging and processing continuity

To protect the integrity of the data, the online IMS uses both external security checking and various internal techniques to record the transactions entered into the system and the database update activity.

The principal tool for recording online system activity is IMS system logging. Data stored on various logging data sets contains information used for restart, recovery, statistics, and audit purposes.

IMS log data is recorded in four kinds of data sets:

- Online log data set (OLDS)
- Write ahead data set (WADS)
- Restart data set (RDS)
- System log data set (SLDS)

The online system uses a minimum of three OLDSs, one WADS, and a single RDS, all residing only on DASD. When one or more online log data sets are filled, you can archive them to system log data sets using the IMS Log Archive utility. DASD or tape media can be used for SLDSs. Batch systems use system log data sets and are able to log to either tape or DASD.

The online system uses the OLDSs in wrap-around fashion. If dual logging of the OLDSs is an installation requirement, a pair of data sets (primary and secondary) must be assigned. The DD names for the OLDSs begin with the character string DFSOLP for the primary data set and DFSOLS for the secondary data set. A unique suffix (00 through 99), called an "OLDS identifier," completes the 8-character DD name. Either single logging or dual logging is performed, as determined by DD statements during system initialization or by instructions included in the LOGGER section of the DFSDF*xxx* PROCLIB member.

A WADS is a small data set containing a copy of log records that are in OLDS buffers but have not yet been written to the OLDSs. When logging to DASD (required for online processing), fixed-length blocks make direct retrieval easier. A WADS allows large fixed-length blocks (in variable blocked format) to be written to the OLDSs without the requirement to rewrite blocks. When log data has been written to the OLDSs, the WADS is reused.

If a system failure occurs, the log data in the WADS is used to close the OLDSs. The close process occurs as part of an emergency restart or as an option of the Log Recovery utility.

IMS allows up to ten WADS, only one or two of which are active while the rest are spares. When a write error is detected, a spare WADS replaces the WADS that encountered the error. Dual WADS logging is also supported if it is required to have backup in the event of a read error while closing the OLDSs from the WADS.

The online IMS system controls the log data sets that are used for startup. It makes use of entries in the checkpoint identification table written on the restart data set, and of log data set information recorded in the DBRC RECON data set. If you are using automatic restart, the **/START IMS** command issued from the system console causes the appropriate kind of restart. Normally, this restart results in the use of OLDSs records and in a normal restart that completes without the use of prior system logs. If restart processing abnormally terminates before the initial checkpoint, the appropriate restart for automatic restart is the same type (**/NRESTART** or **/ERESTART**) as the aborted restart.

Checkpointing

Checkpointing is the primary technique that IMS uses to record information that can be used to restart an interrupted operation. Using the status information recorded during a checkpoint, IMS restores the contents of the message queues and database changes.

Checkpoints are an integral part of system shutdown and startup. Also, the amount of reprocessing, back from the point of system interruption and forward to a continuation point, is reduced when checkpointing is reasonably frequent. Some processing overhead is associated with checkpoint information, but this is an acceptable trade-off for the efficient restart of the system.

In an XRF complex, SNAPQ checkpoint records taken on the active IMS system are used to build control blocks on the alternate IMS system during the synchronization phase.

IMS internal checkpoints are scheduled to occur automatically at predetermined intervals. The interval is specified in terms of an increment to the number of system log records created. As the online IMS events are logged with individual log record types, a count is maintained. When the increment exceeds the specified value, checkpoint processing is invoked. IMS checkpoints can also be invoked explicitly by the master terminal operator and by application programs that have been authorized to issue the **/CHECKPOINT** command.

Fast DB Recovery regions monitor checkpoint records on IMS systems, and uses them during database resource recovery.

Locking mechanisms and database integrity in DB/DC and DCCTL

IMS offers a choice of locking; you can use program isolation (PI) locking or the services of the Internal Resource Lock Manager (IRLM). The IRLM component is used as an integral part of data sharing.

With program isolation, all activity (modifying the database and creating messages) of an application program that is active in the DB/DC environment is isolated from any other application programs that are active in the system. The isolation persists until that application program confirms, by reaching a synchronization point, that the data it has modified or created is valid.

Note: The PI lock manager supports a maximum of 63 waiting application programs in an IMS system. If an application program that must wait for a lock exceeds the maximum that the PI lock manager supports, the application program terminates with an abend 2478, is backed out by IMS, and its locks are released. If the requestor is a message-driven program (MPP, JMP, IFP, or BMP) the message is returned to the queue and is reprocessed.

The locking mechanisms are also used to:

- Remove the effects of an abnormally terminated application program
- Perform the processing required for a ROLL, ROLB, or ROLS call
- Resolve deadlock situations

For all the above processing, the removal of database updates and held output messages is done from the previous synchronization point up to the current status. A *synchronization point* is defined as the point at which an application program can be restarted. The first such point for an application program is its initial scheduling. The most common synchronization point is when a GU to the message queue occurs. By issuing a call for the next message, a program in single message mode is indicating the start of a cycle of processing and the completion of any previous work. At this time, any output messages that are queued to a temporary destination are sent to their final destination, and database updates are committed.

An application program can also issue a CHKP call, which forces a synchronization point. For application programs executing in multiple message mode, or BMPs that are not transaction driven, the synchronization point is the time of either the initial scheduling or the last CHKP call.

Another aspect of program isolation is the control of database updates at the segment occurrence level. During the scheduling process, IMS analyzes the intent of an application program toward the database it uses. If a conflict exists with the database usage of a currently scheduled transaction and a candidate for scheduling because an application program needs exclusive use of the database, the scheduling process must select another transaction code and try again. If exclusive intent is not a factor (this is usually the case), application programs are scheduled concurrently. IMS controls the interleaved ownership of database segments with a locking mechanism. As application programs execute, they enqueue on the database records and release those resources, either after update or when the application program reaches a synchronization point.

Possible deadlock situations are resolved in a manner transparent to application programs and terminal operators. When IMS detects a deadlock situation, one of the application programs involved in the deadlock is abnormally terminated with a special abnormal termination code. The abnormal termination causes the activity of the terminated program to be dynamically backed out to a previous synchronization point. Its held resources are released. This allows other application programs to complete their processing. The special code causes the transaction that was being processed to be saved. The application program is rescheduled.

In DBCTL, if a deadlock situation forces an abnormal termination of a CCTL thread, that thread is not saved or retried by DBCTL. The CCTL, upon receiving certain deadlock termination codes, retries its transaction.

If a BMP is selected to be dynamically backed out, it cannot be rescheduled and terminates at its latest synchronization point. If the BMP did not access the message queues for input or issue CHKP calls, the BMP terminates during scheduling and all the BMP database update activity is nullified.

If an ODBA thread is active when IMS DB shuts down or terminates abnormally, the ODBA application thread is terminated. The ODBA application program is not terminated, but is no longer able to make calls on the thread.

Exit points and routines

You can modify how IMS processes a unit of work by providing IMS with subroutines (called *exit routines*) that perform special processing. IMS calls exit routines at various points in its logic flow (called *exit points*) and allows you to control how IMS performs its work.

One example of modifying how IMS processes a unit of work might be to interrogate an in-flight message (in an MSC environment) to determine if it originated from a test or production system. If it came from a test system, you might want to route that message to an application that collects data about the test system. If the message originated from a production system, you probably would not want to change its destination.

To perform the interrogation of the message, you would code a TM and MSC Message Routing and Control User Exit routine. If you name this routine DFSMSCE0, place it in the IMS.SDFSRESL library, and bind it to that library (or a concatenated library), IMS calls your exit routine when IMS receives the message. When your exit routine is done with its processing, the exit routine returns control to IMS and then IMS resumes processing the message to either the original destination or the test application.

Certain exit routines are required and others are optional. Some IBM-supplied exit routines can be used as is and some require modification before using.

Related reading: For complete information about exit points and exit routines, see *IMS Version 15.2 Exit Routines*.

Data Capture exit routines

If your installation contains both IMS DB and Db2 for z/OS databases, duplicating data in IMS DL/I and Db2 for z/OS relational databases might be required.

For example, your Db2 for z/OS application programs, written in Structured Query Language (SQL), might require data from the IMS DB database. You might be converting your site to Db2 for z/OS on a gradual basis, or you might want to take advantage of the relational technology of Db2 for z/OS for some of your IMS data.

To duplicate data between the two types of databases, you must ensure that each update to data segments occurs in both databases in a timely manner. The process of duplicating updates from an IMS DB database to a Db2 for z/OS database is known as data propagation.

The two ways to propagate data from IMS DB to Db2 for z/OS are:

- IMS DataPropagator, an IBM licensed program that provides support for data propagation and for exit routines.
- Data Capture exit routine. This is an exit routine that you write to establish a routine for data propagation. It can be written in assembler language, C language, COBOL, or PL/I, and it is called by an application program that requires data propagation.

Note: The Data Capture exit routine is not available to CICS. DBCTL can use the exit routine, but only for BMPs.

Related reading:

- For information about system requirements for data propagation, see *IMS Version 15.2 System Definition*.
- For information on the database considerations associated with the Data Capture exit routine, *IMS Version 15.2 Database Administration*.
- For information on writing a Data Capture exit routine, see *IMS Version 15.2 Exit Routines*.
- For more information about IMS DataPropagator, see:
  - *IMS DataPropagator for z/OS: An Introduction*
  - *IMS DataPropagator for z/OS Concepts*

HALDB Partition Selection exit routines

Any installation-defined High Availability Large Database (HALDB) Partition Selection exit routines must be assembled and bound into a load library.

The library you choose needs to be part of the IMS SDFSRESL concatenation. Also, all HALDB Partition Selection exit routines must be linked as re-entrant.

You can specify the name of the HALDB Partition Selection exit routine during DBD generation, with the HALDB Partition Definition utility, or on the DBRC INIT.DB command.

If a HALDB Partition Selection exit routine is defined, you must also consider the Partition String Value parameter defined by the HALDB Partitioned Definition utility or the KEYSTRNG parameter on the DBRC **`INIT.PART`** command. This parameter provides the user exit routine with optional data that can be used for HALDB partition selection. The parameter is not required and may be omitted.

Related reference

HALDB Partition Selection exit routine (DFSPSE00) (Exit Routines)

The z/OS Automatic Restart Manager (ARM)

You can use the z/OS Automatic Restart Manager (ARM) to restart a subsystem (or job) after a z/OS hardware or software failure.

In addition, in the event of a z/OS hardware or software failure that requires you to move a subsystem from one z/OS system to another, ARM will move all the subsystems defined in the same restart group as a group to a remaining z/OS system.

IMS supports ARM in these environments:

- TM-DB
- DCCTL
- DBCTL
- XRF
- FDBR

DL/I, DBB, and IMS utilities are not supported. The IMS control region is the only region restarted by ARM.

Attention: The DL/I SAS and DBRC regions are started internally by the IMS control region. IMS dependent regions are not automatically restarted, because they are normally restarted after the IMS control region has restarted.

The element name that IMS uses on the registration call to ARM is the IMSID. The element type is SYSIMS. Duplicate element names are not allowed by ARM. When ARM is used, the IMSIDs of online systems and FDBR systems must be unique.

ARM provides a default ARM level of 1 for SYSIMS.

The IMSID must be unique across the sysplex. ARM tries to move IMS to a surviving z/OS if a failure occurs on the z/OS or the CPC on which the IMS is executing. If the IMSID is not unique, ARM might move the IMS from the failing CPC to one that already has an IMS with the same IMSID.

If IMS is canceled by z/OS, IMS is only automatically restarted by ARM if the ARMRESTART option is specified on the CANCEL or FORCE command.

IMS maintains the following user abend table and de-registers from ARM any time one of these abends occurs:

1. U0020: USER 20 - MODIFY
2. U0028: USER 28 - /CHE ABDUMP
3. U0604: USER 604 - /SWITCH
4. U0758: USER 758 - QUEUES FULL
5. U0759: USER 759 - QUEUE I/O ERROR
6. U2476: USER 2476 - CICS TAKEOVER

The first three of these abends are the result of operator intervention. The last three abends require some external changes before IMS can be restarted.

Chapter 3. Documenting the IMS system

When planning to support the administration of an online IMS system, you must consider several responsibilities that involve documentation.

Reviewing this documentation for application requirements is a necessary task when you are designing the IMS online system or responding to required changes in that design.

Extracting requirements for the IMS system

Analysis of the scope and impact of an application in the online environment occurs during the design phase, as well as when application changes are proposed.

You must assess the detail of the application requirements by reviewing the following sources:

- Program specifications and logic
- Implementation plan
- Summary of business requirements
- Design change requests

When you examine application documents, you must extract several kinds of information:

- Requirements for IMS function
- Database requirements
- Predictions of the application workload
- Network definition requirements
- Security considerations
- Operating requirements
- Audit and history recommendations
- Performance factors
- Terminal requirements
- FPBUF requirements

If your system must use intelligent remote stations, you must:

- Select features of IMS that can be used to support distributed application processing (for example, ISC, MSC, LU 6.2).
- Identify the terminal support provided by IMS (including ETO).
- Identify the Fast Path requirements of your system.
- Evaluate the off-loading of application requirements to intelligent remote stations and the use of special components or screen formatting.
- Assist in the design of programs that reside in intelligent remote stations and communicate with IMS; identify the Systems Network Architecture (SNA) protocol to be used by IMS and the intelligent remote station.

Participating in design reviews

As the administrator of the online IMS system, you are concerned with adequate detail in the specifications. You need to plan for, and subsequently specify, the online system. As the development of an application package progresses, reviews of the design should be held.

The following table summarizes the kind of information that you must gather and how that information relates to system administration tasks.

In addition to application design specifications, the application development team might maintain a controlling document for schedules and responsibilities. You must contribute to this plan with your own requirements and milestones, such as completion dates and testing dates for the operating procedures.

Table 3. Administration's use of design reviews

Design stage	Information needed	Administrative tasks
Design review 1	• Scope of project • Hardware and software requirements • End-user and development contacts	• Analyze IMS function requirements • Contribute to documentation plans • Check standards compliance • Assess network impact • Assess DP operations impact • Make workload predictions
Design review 2	• Use of MFS and screen usage characteristics • Elements of end-user control • Network planning • Pointers for operator control • Transaction workload	• Establish MFS library control, identify format names • Coordinate dictionary use • Check naming standards • Track network requirements • Plan security strategy • Begin RTO and MTO procedures • Predict processing workload
Design review 3	• Message definition • Conversational attributes • Databases and programs • System resource requirements • Need for message edit • Recovery considerations • Security and audit	• Calculate message queues • Calculate SPA data • Specify system and JCL • Finalize system requirements and hardware plan • Specify message edit coding • Begin recovery procedures • Develop security design
Database application design review	• Database maintenance • Database sharing • Validation and acceptance plans • Performance predictions • Monitor plans	• Document online databases and image copy requirements • Choose system integrity options • Establish system availability • Establish performance criteria and plan monitoring
Logic review	• Monitor pointers • Program preload • Virtual storage needs • Database processing intent conflicts	• Develop a monitoring strategy • Plan dependent regions • Develop scheduling algorithm • Estimate buffer pool and system data set resources

Related reading: For more information on the purpose and scope of design reviews, see *IMS Version 15.2 Database Administration*.

Establishing naming conventions

A critical part of the application specification and the control of the IMS online system design is maintaining naming conventions for your resources.

When you define a large system that has many resources, the ability to recognize the characteristics of the resource by its name has many advantages:

- The system definition input is easier to check, and the identification of changes is easier.
- The MTO control is more effective and efficient and less prone to error.
- The modification of the application design can more easily recognize already defined resources rather than creating ambiguous or unnecessary additional resources.

You should establish naming conventions, in cooperation with database administrators, for at least the following resources:

- Databases, their DD names and data set names
- Image copy and change accumulation data set names
- Segment and field names
- PSB and program names
- Transaction codes
- MFS format names
- LTERM and node names
- ETO terminal and user names
- LU 6.2 descriptor names
- Online log data set names
- System log data set names
- IMS Monitor output data set names
- Link names and IMS system IDs (for Multiple Systems Coupling)
- Fast DB Recovery region names
- High Availability Large Databases (HALDBs), and HALDB partitions, DD names, and data set names.

The following table shows some examples of naming conventions that can be applied to resources controlled by IMS for online applications.

Table 4. Examples of naming conventions

Resource	Naming Convention	Description
Transaction	Taaatsss	**T** Transaction **aaa** Application identifier **t** U for update, or R for inquiry transactions **sss** Transaction sequence

Table 4. Examples of naming conventions (continued)

Resource	Naming Convention	Description
LTERM name	cnnxiiii	**c** L for local, S for switched, N for non-switched **nn** A 2-character code for terminal type **x** A 1-character attribute indicating the screen size, printer, or component **iiii** A 4-character identifier
MFS (MSG name) (MID and MOD)	aaaiiii	**aaa** Application identifier **iiii** A 4-character identifier
MFS (FMT name) (DIF and DOF)	aaaiii	**aaa** Application identifier **iii** A 3-character format identifier
Module name	Maaaiiii	**M** Module name **aaa** Application identifier **iiii** A 4-character identifier
Job name	Jaaannnn	**J** Job name **aaa** Application identifier **nnnn** A 4-character job identifier

For more information on naming conventions, see the:

- *IMS Version 15.2 Database Administration* for recommendations for naming conventions for databases, PSBs, programs, and HALDBs.
- *IMS Version 15.2 System Definition* for a list of restricted names.
- *IMS Version 15.2 Communications and Connections* for specific naming conventions for ETO.

Using a data dictionary

If you use the IBM DB/DC Data Dictionary licensed program (program number 5740-XXF), you can use several of its features to assist in system and network documentation. The product supports interactive processing, enabling you to use a terminal to create the documentation at your convenience.

The product provides standard categories for databases, segments, fields, PSBs, PCBs, transactions, programs, and the system. You can use these categories to build up detailed descriptions of the online

IMS system resources. Procedures and execution JCL can also be recorded (as User Data). Three of the stage 1 macros—APPLCTN, DATABASE, and TRANSACT—can be produced as card output.

You can use the Extensibility Feature to define terminal subjects and specify a selection of attributes to record for each terminal. With the Description and User Data segments, other free-form text can be associated with the terminal subjects.

For more information about the IBM DB/DC Data Dictionary, see *IBM DB/DC Data Dictionary Administration and Customization Guide*, SH20-9174.

Documenting the system characteristics

You must create an independent body of information that documents the design and operation of the production system. This material is derived from the application requirements and includes the intended use of IMS facilities.

The following list describes the main documentation activities.

- Documenting the IMS system definition

  The detail of your design for an IMS online system is reflected in the system definition macro specifications. The input to the first stage of system definition processing can be used as a major documentation tool. Including comments on the choice of parameters should help you control definition changes.

  The IMSCTRL macro statement offers an ETO option that creates a report of current ETO descriptors in the network. You can use the ETO descriptor report to monitor your current network definition.

  Related reading: For more information about commenting on parameters and information about the IMSCTRL macro, see *IMS Version 15.2 System Definition*.

- Documenting the IMS network in DB/DC and DCCTL environments

  Within the broader scope of system documentation, you need to document the details of the IMS network. Start this documentation as soon as initial plans for an application system become available. The advantage to starting early is that you can become familiar with the physical network and the way it will logically be used by IMS connections. When your online IMS system uses terminals that are part of a network defined to VTAM, you might be able to use some of the documentation developed by the system programming staff.

  Documenting in detail allows you to:

  – Familiarize yourself with the features and operation of the terminal devices
  – Find out if the application is going to use the terminal in an unusual way and, if so, research the potential problems
  – Prepare for IMS system definition stage 1 input
  – Prepare ETO descriptors
  – Prepare LU 6.2 descriptors
  – Plan for the installation and network generations
  – Understand the operational aspects of subsets of the total configuration

- Documenting terminal profiles in DB/DC and DCCTL environments

  One way to document the required network, after the major design is stable, is to record the intended use and characteristics of each terminal. You construct a profile that contains:

  – The terminal type, its required features, and the type of connection it is to use
  – What options were chosen for the device and the reason for the choice
  – The characteristics of how the terminal is to be used by the application
  – The proposed LTERM names that can be associated with the terminal or the user
  – If a VTAM-supported device, the node name and transmission characteristics

  - The extent of the proposed usage and whether the device is to be shared with non-IMS users
  - The diagnostic procedures that are appropriate for the device
  - User profiles and user profile security

  For dynamic terminals, you should maintain a record of the characteristics of dynamic terminals and users to make future changes easier.
- Documenting transaction profile names for APPC/IMS

  Definitions of transaction program names (TPNs) are contained in the APPC/MVS resource, TP_Profile. Within TP_Profile, TP profile data sets provide attribute information for each TPN. You can define TPNs that have different characteristics for each LU name with which they are associated.

  Related reading: For more information on TP names, see *IMS Version 15.2 Communications and Connections*.
- Documenting the configuration of the production system

  You should create a configuration map showing how individual terminals and clusters of terminals are to be connected in the network. This gives you some insight as to how to specify the network control to the MTO. Having the configuration map is also an aid when you interact with various technical specialists. The map should show:

  - Processors or host computers
  - Channels and lines (including the type of line)
  - Communication controllers
  - Control units and the terminal attachments
  - VTAM node names
  - For XRF systems, USERVAR or MNPS ACB name
  - Alternative connections or configurations

  To solve problems that might arise, you must be able to identify the terminal or control unit that has the problem. You can assist both end users and service personnel by placing identifying labels on the devices. In addition to the IMS address or node name and the LTERMs appropriate for that terminal, include the hardware address, circuit identification, and other data that might be necessary.

Chapter 4. Introduction to IMS operations and recovery

For any large system, operations and recovery go together: *operations* refers to the day-to-day activities involved in keeping the system running smoothly, and *recovery* refers to the activities required to bring a failed system back online.

Information about operations would be incomplete if it did not describe what you need to do for recovery, when the system (or some part of it) runs into problems.

Another important task related to operations is planning for both day-to-day operations and for recovery. Part of the task of planning is developing procedures that specify the roles of the IMS operators and system programmers. These procedures clarify the responsibilities of each of these roles.

This topic introduces both operations and recovery for IMS and describes some of the tools provided by IMS and z/OS for both tasks.

Understanding the operations task

The task of planning for operations has two major parts: selecting functions and tools and developing operating and end-user procedures.

First, you must decide how to use the tools IMS provides for operating your system. This includes choosing, for example, whether to use dual logging when setting up your log, how often to make backup copies of your database, and whether to use DBRC to control recovery of databases.

Second, you must develop procedures for operating and using IMS. Operating procedures must tell operators how to:

- Start and restart IMS
- Control IMS
- Make online changes to modify IMS
- Shut down IMS
- Run various IMS utilities
- Recover from IMS and other failures

These procedures are primarily for the master terminal operator (MTO), who operates IMS from the master terminal. They are secondarily for the people who assist in keeping IMS running smoothly, such as the recovery specialist who works with many of the more complex IMS recovery problems.

End-user procedures tell end users how to:

- Operate their terminal
- Establish a connection to IMS
- Communicate with IMS (using their specific applications)
- Terminate a connection to IMS
- Respond to any error conditions they encounter

This information is not an exhaustive treatment of operations. More detailed information on various aspects of operating your system is provided in other IMS topics.

Related reading:

- *IMS Version 15.2 Operations and Automation* gives guidance level information and details about the commands that are used to operate IMS.
- *IMS Version 15.2 Database Utilities* and *IMS Version 15.2 System Utilities* give guidance level information and detail about the various utilities that are available with IMS.

Automated operations

Automated operations are tools and techniques that help you improve your installation's productivity. As your system grows more complex and your message traffic increases, automating certain tasks can increase your system's efficiency.

You can use automated operations to:

- Minimize errors
- Increase availability
- Expedite problem diagnosis and prevention

IMS provides the following tools to help you automate your operations:

- Time-Controlled Operations (TCO)
- Automated Operator Interface (AOI)
- REXX SPOC API

IMS operations can also be automated using Tivoli NetView for z/OS. Because Tivoli NetView for z/OS functions independently of IMS, it can gather information and issue commands that are not available to IMS.

There are many advantages for automating tasks. Many of the jobs that operators perform are simple, repetitive tasks, such as monitoring the system and issuing recovery commands. You can often automate these jobs and thereby free the operator for more complex activities, such as implementing operational procedures.

In the IMS environment, TCO and AOI can improve your operator productivity by:

- Improving operator productivity and accuracy. Automating operator tasks simplifies procedures, reduces operator input, and minimizes operator errors. For example, TCO can reduce operator input by automatically monitoring system status, starting message regions and telecommunication lines, and notifying users of system status.
- Expediting problem determination. An operator's primary job is to bypass or fix problems quickly. If adequate information about a problem is not available, it might not be possible for the operator to fix it quickly. Automated operations are especially suited for problem determination. For example, TCO can automatically gather necessary information and do diagnostic analysis so that a problem can be quickly identified and corrected.

Identifying operations that can be automated and implementing them requires a sound understanding of your installation's operations. You must collect and analyze various resources to identify which tasks are good candidates for automation. Resources you should analyze include:

- System logs
- Problem management reports
- Record of calls to the help desk
- Operators' notes

Repetitive and predictable tasks are good candidates for automation.

The process of automating operations should be an iterative one. After you have developed and used automated procedures, you should evaluate them and, in the process, consider whether any new tasks can be automated.

Related reading:

- For more information about automation tools provided with IMS, see *IMS Version 15.2 Operations and Automation*.
- For more information about Tivoli NetView for z/OS, see *Tivoli NetView for z/OS Installation: Getting Started* or the *Tivoli NetView for z/OS User's Guide*.

Understanding the recovery task

Recovery is the task of restoring a failed system, application program, or database back to normal operations.

Recovery is the largest and most complex part of operations. For this reason, any discussion of operations is primarily a discussion of recovery.

Recovering a system can involve:

- Databases
- User requests to process data
- Programs that do the processing (application programs)
- Output sent to users

A recoverable system ensures:

- Data is not lost
- Incomplete changes to a database are not saved

Data can be lost in two ways. It can be physically lost—the disk or tape on which it resides can be damaged or misplaced. Or it can be "logically" lost—the data becomes incorrect or loses its relationship to other data. A system that ensures *data integrity* ensures that data cannot be lost or saved incompletely.

Example: a system without recovery

To show how things can go wrong, consider a hypothetical system that has no recovery mechanisms. A user requests a transaction that handles a customer order.

Part of the application program's work is to:

- Decrease the stock-on-hand number in the database
- Add the cost to the customer's outstanding bill
- Add the amount to a daily-total-sales accumulator

The following list provides different failure scenarios:

- Effects of system failure

  What if the system fails between the time the stock-on-hand number is decreased and the charges are allocated? If the user thinks the transaction is complete, the customer bill record and the total sales record in the database will be incorrect. If the user thinks the transaction failed and reruns it, the stock-on-hand number will be decreased a second time, and that record will also be wrong. In either case, data integrity has been lost.
- Effects of program abend

  What if the program begins changing records in the database and then terminates abnormally (abends)? The result is the same as with the effects of system failure: a half-changed database. A half-changed database is one whose integrity has been lost.
- Effects of an I/O error

  Suppose the program tries to read the stock-on-hand record and encounters a device error. Because of some I/O problem, the record cannot be obtained. Therefore, the program cannot run, and the customer order cannot be filled. In addition, any other work that depends on this record can no longer be completed.

  Also, what if the program has already changed some records expecting that it could read and update this other record? Again, the overall integrity of the data has been lost, because it is only partially updated.
- Effects of queue loss

In large systems, work requests are often saved (queued) and processed later, when the workload permits. Output that the programs send back to the requesters (or others) is also often queued and sent later, when the workload permits.

If the system fails between the time a request is entered (input) and the time it is taken off the queue and executed, the request might be lost. The same is true for queued output: If the system fails between the time the application program supplies its output and the time that output can be taken off the queue and actually sent, those results might be lost.

Requirements for a recoverable system

In order for a system to be recoverable, it must protect the database from half-complete (and, therefore, incorrect) changes. It must be able to deal with the physical loss—unavailability or disappearance, in part or in whole—of the database. Also, it must protect the input and output queues from being lost.

Steps in the process of recovery

The process of recovery includes two major tasks: planning what you intend to do for recovery before a problem occurs and reacting to a problem after it has occurred to fix the problem.

The task of planning for IMS recovery can be further refined:

- Set up logging
- Establish checkpoints
- Make backup copies
- Create procedures and assign responsibilities

The task of reacting can also be further refined:

- Notice error
- Shut down IMS (only if necessary) — can be a full or partial shutdown
- Diagnose and fix (or possibly bypass) error
- Use recovery utilities and services (if necessary) — can be run online or offline
- Restart IMS (or the now fixed part of IMS)

Of course, real recovery is rarely as straightforward as these two lists suggest.

The mechanisms of recovery in IMS

IMS is designed to be recoverable and has a number of mechanisms to help you in these areas. Some function automatically; others require your involvement.

Successful recovery depends on two things:

- Having a safe point to return to, a known point of integrity. This is a point at which you know data is correct and make a record of it.
- Keeping track of what processing has been done since that safe point.

If these two types of information are gathered, recovery is almost always possible.

Synchronization points

Strictly speaking, all you need to be able to restart a system after a failure is the log: as long as you start out correctly and log everything you do thereafter, you can always recover. Restart in this case would require the system to read every log record ever written for the system.

The more time that has passed since the first time you started the system, the more log records will have been written, and the more time-consuming the restart will be.

Definition: A synchronization point, or *sync point*, is a known point of data or system integrity. It is a point in time at which all of the changes made to the database are complete. If a failure makes restart necessary, you can begin reprocessing from the most recent sync point.

The value of having a sync point is that when IMS restarts, it can ignore all logs created before the sync point was taken. If your sync points are infrequent, they become less valuable because the amount of log data IMS has to read during a restart increases. Thus, you should establish periodic sync points. Then, if a problem arises, you do not need to examine as much of the log, because you established the last sync point relatively recently.

IMS uses two types of sync points: those taken by IMS itself (called *system checkpoints*), and those taken by individual application programs running under IMS (called application program sync points or application program *commit points*).

System checkpoints

IMS automatically takes periodic system checkpoints, so if it is necessary to restart IMS, it can begin at the last checkpoint (or an earlier one if the work done before the checkpoint was not complete).

The interval between system checkpoints is measured by how many log records IMS writes, so the checkpoints are a measure of activity rather than elapsed time. You select the interval between system checkpoints when you install IMS.

As with most choices involving preparation for recovery, the impact of taking frequent checkpoints must be weighed against the advantages of faster recovery. Because IMS does not suspend work to take a checkpoint, decreased transaction speed is not a significant impact factor.

Recommendation: Do not suppress system checkpoints to improve system performance because emergency restart must always go back at least two system checkpoints (or back to a prior restart).

The IMS MTO can also request an additional system checkpoint at any time by issuing a **/CHECKPOINT** command.

Commands, such as **/DBR**, **/DBD**, **/STA**, and **/STO**, will be rejected and message DFS140I is issued if a checkpoint is in progress when those commands are entered.

IMS takes a system checkpoint when any of the following occur:

- IMS starts
- A regular interval has elapsed
- The MTO enters a command (**/DBRECOVERY**, **UPDATE DB STOP(ACCESS) OPTION(FEOV)**, **/CHECKPOINT**, or **/SWITCH OLDS CHECKPOINT**)
- IMS shuts down

If IMS is enabled with resource definition data sets or the IMSRSC repository and AUTOEXPORT=AUTO, RDDS, or REPO is specified in the DFSDFxxx PROCLIB member, IMS will automatically export to the RDDS or the repository any changes in resource definitions for MODBLKS resources since the last automatic export or the **EXPORT DEFN** command at the end of each normal checkpoint processing.

Application sync points

IMS system checkpoints help you recover the IMS subsystem, but you also need to be able to restart application programs if they fail. IMS allows application programs, both batch and online, to take sync points.

An application program sync point has two purposes:

- It marks an intermediate completion point—a place at which the work finished so far is judged to be correct. Any future recovery can take place from this point.
- It frees locks held on database records the program has updated and enqueues any output messages the program created. IMS can send enqueued messages to their destinations.

Application program sync points are sometimes called *commit points*. By taking a sync point, the program is committing itself to the accuracy and completeness of what it has done, and releasing the data for use by other applications.

Logging

Logs are lists of activities, lists of work that have been done, and lists of changes that have been made. By knowing the state of the system when things were all right, and knowing what the system has done since then, you can recover it in the event of a failure.

Logs make recovery and restart possible. As IMS operates, it constantly writes its event information in log records. The log records contain information about:

- When IMS starts up and shuts down
- When programs start and terminate
- Changes made to the database
- Transaction requests received and responses sent
- Application program checkpoints
- System checkpoints

The following figure illustrates how logging works in an IMS online environment; logging in a batch environment is simpler.

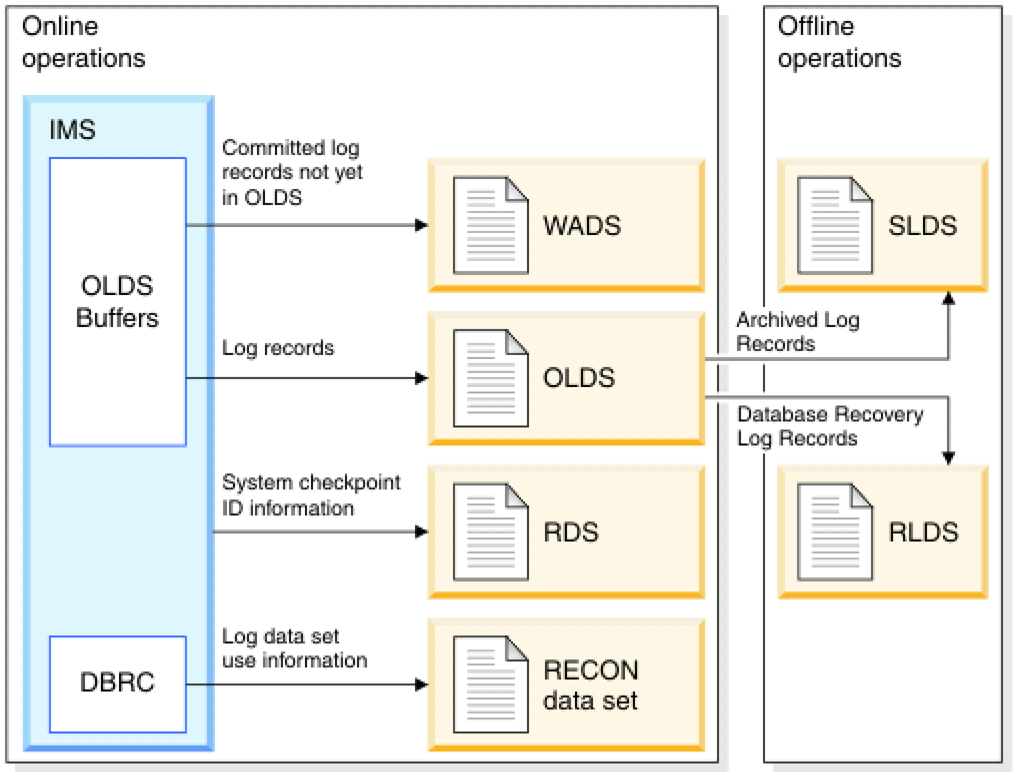

Online operations
IMS
OLDS Buffers
DBRC
Committed log records not yet in OLDS
WADS
Log records
OLDS
System checkpoint ID information
RDS
Log data set use information
RECON data set
Offline operations
SLDS
Archived Log Records
Database Recovery Log Records
RLDS

Figure 23. Overview of the logging process

IMS externalizes log records by writing them to an Online Log Data Set (OLDS). To enhance performance and to optimize space on OLDSs, incomplete or partially filled buffers are written to a Write Ahead Data Set (WADS) when necessary for recoverability. The WADSs are high-speed DASD data sets with a high write rate. Only complete log buffers are written to OLDSs. When the log data is on an OLDS, the equivalent WADS records are ignored.

IMS uses a set of OLDSs in a cyclical way, which allows IMS to continue logging when an individual OLDS is filled. Also, if an I/O error occurs while writing to an OLDS, IMS can continue logging by isolating the defective OLDS and switching to another one. Once an OLDS has been used, it is available for archiving to a System Log Data Set (SLDS) on DASD or tape by the IMS Log Archive utility. The utility can be executed automatically through an IMS startup parameter (ARC=).

When IMS is close to filling the last available OLDS, it warns you so you can ensure that archiving completes for used OLDSs or add new OLDSs to the system. You can also manually archive the OLDSs to

SLDSs using the IMS Log Archive utility. An SLDS can reside on DASD or tape. After an OLDS is archived, it can be reused for new log data. You use SLDSs as input to the database recovery process.

When you archive an OLDS, you can request that IMS write a subset of the log records from the OLDS to another log data set called the recovery log data set (RLDS). An RLDS contains only those log records required for database recovery.

While IMS is running and logging its activities, it takes periodic system checkpoints, and writes the checkpoint notification to another data set called the restart data set (RDS). IMS uses the RDS when it restarts to determine the correct checkpoint from which to restart.

In a batch environment, IMS writes log records directly to an SLDS, and does not use either the OLDSs or WADSs. Just as in the online environment, batch SLDSs can reside on DASD, tape, or mass storage. If the SLDS is on DASD, you can use an IMS utility to copy log records from DASD either to tape or to other DASD.

The IMS log data sets

These topics describe the OLDS, WADS, SLDS, RLDS, restart data set (RDS), RECON data set, the z/OS log data set used for CQS, the CQS system checkpoint data set, and the CQS structure checkpoint data set.

Online log data set

IMS uses the OLDS only in the online environment. The OLDS contains all the log records required for restart, recovery, and both batch and dynamic backout. The OLDS holds the log records until IMS archives them to the SLDS.

Define all of the OLDSs in the IMS procedure library (IMS.PROCLIB) using the OLDSDEF statement. The OLDS must be preallocated on a direct-access device. You can also dynamically allocate additional OLDSs while IMS is running by using the **/START OLDS** command.

IMS uses the Basic Sequential Access Method (BSAM) to write log records to the OLDS and to read the OLDS when IMS performs dynamic backout. Although referred to as a data set, the OLDS is actually made up of multiple data sets that wrap around, one to the other. You must allocate at least three, but no more than 100, data sets for the OLDS.

zHyperWrite can be used for writing to the OLDS to decrease replication latency. To enable or disable zHyperWrite for the OLDS, you have the following options:

- Use the OLDS= keyword in the **ZHYPERWRITE=** parameter in the LOGGER section of the DFSDFxxx PROCLIB member. You must restart IMS to make the change effective. Changes introduced by using this keyword will persist across a restart. For more information, see LOGGER section of the DFSDFxxx member (System Definition).
- Issue the **UPDATE IMS SET(LCLPARM(ZHYPERWRITE(OLDS()))** command. You don't need to restart IMS to make the changes effective. However, changes introduced by issuing this command will not persist across a restart. For more information, see UPDATE IMS command (Commands).

Before enabling zHyperWrite, you must ensure that all the OLDS that will be used by the IMS system are defined as extended format data sets.

Related reading: For more information about allocating data sets for the OLDS, see "Allocating data sets" in *IMS Version 15.2 System Definition*.

You can specify that the OLDS use *dual logging*, which is the duplication of information on two logs. When you use dual logging, an I/O error on either the primary or secondary data set causes IMS to close the nonerror OLDS and mark the error OLDS in the Recovery Control (RECON) data set as having an I/O error and a close error. IMS then continues logging with the next available pair. For dual logging, the minimum number of data sets is three pairs, and the maximum number is 100 pairs.

IMS uses as many OLDSs as you allocate. IMS issues a message each time it changes the current OLDS. This message identifies the OLDS being closed and the next OLDS to be used.

When any of the following events occur:

- IMS fills an OLDS

- An I/O error occurs
- You issue one of the following commands:
  - **/SWITCH OLDS**
  - **/DBDUMP DB**
  - **/DBRECOVERY DB**
  - **UPDATE DB STOP(UPDATES) OPTION(FEOV)**
  - **UPDATE DB STOP(ACCESS) OPTION(FEOV)**

IMS does the following:

- Opens the next OLDS
- Notifies DBRC and the MTO that it is using a new OLDS
- Closes the current OLDS (both primary and secondary if you are using dual logging)

When IMS is using the last available OLDS, it alerts the MTO that no additional OLDS space is available. If archiving has not finished by the time all of the OLDSs are full, IMS waits until OLDS space becomes available. IMS will not log to an OLDS containing active data that is not yet archived. You must run the Log Archive utility to free the OLDS space. After IMS uses the last allocated OLDS, it reuses the first OLDS, if it has been archived.

You can use the **/STOP** command to stop and dynamically deallocate an OLDS. When stopped, that OLDS is no longer involved in the wraparound process.

Recommendation: Stop an OLDS when an error occurs that requires that OLDS to be recovered.

Restriction: You cannot stop the current OLDS. You cannot stop any OLDS when two or fewer OLDSs are currently available.

Similarly, you can use the **/START** command to start and dynamically allocate an OLDS. IMS retains the status of an OLDS (in-use, stopped, and so on) from one restart to the next.

Jobs to archive OLDSs might not complete in the order in which the OLDSs were created. For example, one OLDS might not yet be archived, but a subsequent OLDS might already be archived. When this occurs, IMS issues message DFS3259I and uses the next available OLDS.

The DBRC RECON data set contains information about the OLDSs for each IMS subsystem. Information in the RECON data set indicates whether an OLDS is available for use or contains active log data that must be archived.

Write-ahead data set
IMS uses the write-ahead data set (WADS) only in the online environment. The WADS contains a copy of committed log records that are in OLDS buffers, but that have not yet been written to the OLDS.

In order to maximize log efficiency, IMS uses a log write-ahead function to write partially filled blocks to the WADS (rather than the OLDS). IMS continually reuses WADS space after writing the appropriate log data to the OLDS.

The log write-ahead function ensures that all log records are on the log before IMS writes changes to a database. IMS updates a database in any of the following situations:

- When IMS needs to reuse the database buffer (if this is before commit)
- During commit
- During VSAM background write

If IMS fails, use the log data in the WADS to complete the content of the OLDS and then close the OLDS as part of an IMS emergency restart or as an option of the Log Recovery utility. If you close the OLDS during emergency restart, you must include the WADS in use at the time of the failure.

You must preallocate and format the WADS on a DASD device that supports extended count key data (ECKD) architecture. Format a WADS using the FORMAT WADS | ALL keywords on either the **/NRESTART** or **/ERESTART** commands. All WADSs must be on the same device type and should have the same space allocation. You can also dynamically allocate additional WADSs using the **/START WADS** command.

The WADS must be defined as a VSAM linear data set with a control interval (CI) size of 4 KB (4096-bytes), secondary space allocation of 0, and the SHAREOPTIONS(3 3) parameter.

The hardware features of High Performance FICON® for z Systems™ (zHPF) and zHyperWrite can be used for writing to the WADS. zHPF increases throughput if it is enabled on the LPAR. zHyperWrite is optionally used to decrease replication latency. You can use one of the following methods to enable or disable zHyperWrite for the WADS:

- Use the WADS= keyword in the **ZHYPERWRITE=** parameter in the LOGGER section of the DFSDFxxx PROCLIB member to enable or disable zHyperWrite. You must restart IMS to make the change effective. Changes introduced by using this keyword will persist across a restart. For more information, see LOGGER section of the DFSDFxxx member (System Definition).
- Issue the **UPDATE IMS SET(LCLPARM(ZHYPERWRITE(WADS()))** command. You don't need to restart IMS to make the changes effective. However, changes introduced by issuing this command will not persist across a restart. For more information, see UPDATE IMS command (Commands).

You can change any of the following specifications for the WADS during an IMS restart:

- Number of WADS
- Sequence of WADS
- WADS names
- Use of single or dual WADS

Recommendation: To eliminate potential resource contention, place the WADS on a low-use device that is different from the device you use for the OLDS.

If you place the WADS on the same device as one of your OLDS and use full-track blocking for the OLDS (in which a block is equal to a full track), the device should be able to handle infrequent OLDS seeks. Contention can still occur.

If the WADS and OLDS are on the same device, the Log Archive utility (DFSUARC0) or dynamic backout can cause severe contention between an OLDS being archived and an active WADS.

Related reference

LOGGER section of the DFSDFxxx member (System Definition)

System log data set

IMS uses the SLDS in both the online and batch environments. In the online environment, an SLDS contains archived OLDS data. In the batch environment, an SLDS contains current log data.

Each execution of the Log Archive utility creates an SLDS. One SLDS can contain data from one or more OLDSs. You use an SLDS as input to the database recovery utilities (Database Recovery, Database Change Accumulation, and Batch Backout). You can also use an SLDS during an emergency restart of IMS. SLDSs can be stored on DASD, tape, or other mass storage.

DBRC maintains information about SLDSs in the RECON data set:

- For batch subsystems, DBRC maintains SLDS information in the PRILOG and SECLOG records.
- For online subsystems, DBRC maintains SLDS information in the PRILOG and SECLOG records only if you do not specify an RLDS when you run the Log Archive utility. Otherwise, DBRC maintains SLDS information in PRISLD and SECSLD records.

The Log Archive utility tells DBRC which OLDS it is archiving and which SLDS it is creating. The IMS online system can reuse OLDSs that have been archived.

Generally, you want to copy all the log records from the OLDS to the SLDS, but you can specify specific records. If you want to omit some types of log records from the SLDS in order to save space, include the NOLOG keyword when you run the Log Archive utility. The SLDS must always contain those records that might be needed for database recovery, batch backout, or IMS restart. The records that you can omit are:

X'10'
: Security violation records

X'45'
: Statistics records written during checkpoint

X'5F'
: Call trace record

X'67'
: Communications (SNAP) trace records

X'69'
: Unauthorized ID record (for 3275 display terminal)

IMS dynamically allocates an SLDS during IMS restart whenever log data required for restart read processing is not available from an OLDS. The OLDS might be unavailable because it has been archived, and because one of the following is true:

- The OLDS has been reused.
- The PRIOLDS and SECOLDS records have been deleted from the RECON data set.

To allow IMS to dynamically allocate SLDSs, you must specify the SLDS device type Dynamic Allocation macro (DFSMDA). DBRC provides the data set name and volume information required for dynamic allocation.

Recovery log data set
When you run the Log Archive utility to create an SLDS, you can also request creation of an recovery log data set (RLDS). The RLDS can be stored on DASD, tape, or other mass storage device.

The RLDS contains only the log records needed for database recovery:

X'24'
: Database error records

X'3730'
: Sync point records

X'4001'
: Checkpoint records

X'4084'
: Checkpoint records that contain Fast Path DMCB/DMAC control blocks.

X'4098'
: Checkpoint records for the end of Fast Path checkpoints.

X'42'
: Checkpoint ID records

X'5612'
: End of phase 2 commit records

X'5701'
: Database begin update records

X'59'
: Fast Path database change records

X'505x'
: Database change records

IMS maintains RLDS information in the RECON data set in the PRILOG and SECLOG records. Whenever possible, DBRC uses the RLDS in place of SLDSs when creating JCL for the Database Recovery and Database Change Accumulation utilities. Using the RLDS rather than the SLDS is more efficient because the RLDS contains less information than the SLDS.

Restart data set
IMS writes system checkpoint information to the restart data set (RDS). During each checkpoint, IMS creates or updates a checkpoint ID table; IMS uses this table during IMS restart to determine from which checkpoint to restart the system.

If, for any reason, the RDS is not available at restart, IMS can obtain the required checkpoint information from the log. However, using only the log could considerably lengthen the restart process.

Generally, you do not need to know the content of the RDS. IMS finds the information it needs in the RDS and uses it automatically during a restart.

RECON data sets
DBRC automatically records information in the RECON data sets. Because both RECON data sets contain identical information, this information refers to them as a single data set.

IMS uses the RECON data set in many situations:

- During warm start and normal and emergency restarts. The RECON data set shows which data set—OLDS or SLDS—contains the most recent log data for each DBDS that you registered with DBRC.
- During logging, the RECON shows its latest status for the OLDS and whether the OLDS has been archived.
- For a recovery utility, DBRC selects the correct data sets.

z/OS log data set
The IMS Common Queue Server (CQS) records information about the data in the IMS shared queues in the z/OS log data set. The z/OS system logger serves the same purpose for CQS as the OLDS serves for IMS: it records all necessary information so CQS can recover structures in the coupling facility and restart after failure.

CQS writes log records for each pair of coupling facility list structures to a separate log stream. This log stream is shared among all the CQS subsystems that share the structure pair. z/OS merges the log streams to make recovery possible.

Related reading: For more information on the z/OS system logger, see "Using System Logger Services" in *z/OS MVS Programming: Assembler Services Guide*.

CQS system checkpoint data set
Each CQS subsystem maintains a system checkpoint data set for each structure pair in the coupling facility.

Whenever the CQS subsystem takes a system checkpoint, it writes some control information to this data set. It also writes log records to the system logger log stream.

The system checkpoint data sets are not shared among CQS subsystems.

Related concepts
“Using CQS system checkpoint” on page 167
CQS system checkpoint, the checkpoint data sets that are used for recovery, applies to a CQS if it manages at least one queue structure. If a CQS manages only a resource structure, system checkpoint does not apply.

CQS structure recovery data set
Whenever a CQS subsystem takes a structure checkpoint, it writes a snapshot of the message queues to a structure recovery data set. It also writes some log records to the system logger log stream. The structure recovery data set is used to recover the message queues.

CQS subsystems in a sysplex share the structure recovery data sets; there is one pair (two data sets) for each structure. CQS alternates between the two for each checkpoint.

Related concepts
“CQS structure recovery” on page 174

The structure recovery function recovers the data objects on a structure from the SRDS and the z/OS logs after a structure failure.

Archiving log records

For online systems, you can automatically or manually initiate archiving log records from the OLDS to the SLDS.

Automatic archiving
If you have a large system with a lot of activity, you can minimize your intervention in the archiving process by using automatic archiving. It will eliminate the need to continually monitor logging to determine when to archive.

By default, IMS archives each OLDS when it is full. However, you can control how often archiving occurs by specifying how many OLDSs must be full before IMS archives them. Use the ARC= execution parameter or the AUTOARCH keyword of the **/START** command to control automatic archiving.

Recommendation: You must archive an OLDS before IMS can reuse it; be sure to archive frequently enough to avoid running out of OLDS space. If you run out of OLDS space, IMS waits until OLDS space becomes available.

Related reading: For more information about the ARC= execution parameter, see *IMS Version 15.2 System Definition*. For more information about the AUTOARCH keyword of the **/START** command, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Manual archiving
If you archive infrequently or at irregular intervals, you can initiate archiving yourself. Use the Log Archive utility (DFSUARC0) to archive logs manually.

You can use the DBRC **GENJCL.ARCHIVE** command to produce JCL for the Log Archive utility. You can issue this command using the DBRC Command request, the Database Recovery Control utility (DSPURX00), or an IMS online command (**/RMGENJCL**).

The SLDS can be on DASD or tape. Archiving is also useful for batch systems to free disk space if your SLDSs are on disk. Use the Log Archive utility to copy an SLDS from DASD to tape. Because DASD and tape typically have different block sizes, the utility reblocks the log records while it copies them.

IMS notifies DBRC whenever an OLDS is either filled or closed or both. DBRC updates the RECON data set to indicate that the OLDS is available to be archived.

Using the Log Archive utility, you can archive multiple OLDSs to a single SLDS as long as the OLDSs being archived were created consecutively by IMS. The JCL supplied to the utility defines which and how many OLDSs are to be archived. The GENJCL facility of DBRC allows you to specify:

- Which OLDSs should be included in the created JCL
- That all OLDSs not yet archived should be included

If all the specified OLDSs are archived successfully, DBRC updates the RECON data set to indicate that the OLDSs are now available for reuse by the online system. If the Log Archive utility job fails, re-run it.

If you do not specify automatic archiving, you must create the JCL to run the utility. If you specified automatic archiving, IMS calls the DBRC GENJCL function to generate JCL for the utility when the specified number of OLDSs has been filled or closed.

If the DBRC JCLOUT DD statement for the GENJCL output is directed to the internal reader, the archive jobs are automatically started. The following figure shows an overview of the Log Archive utility.

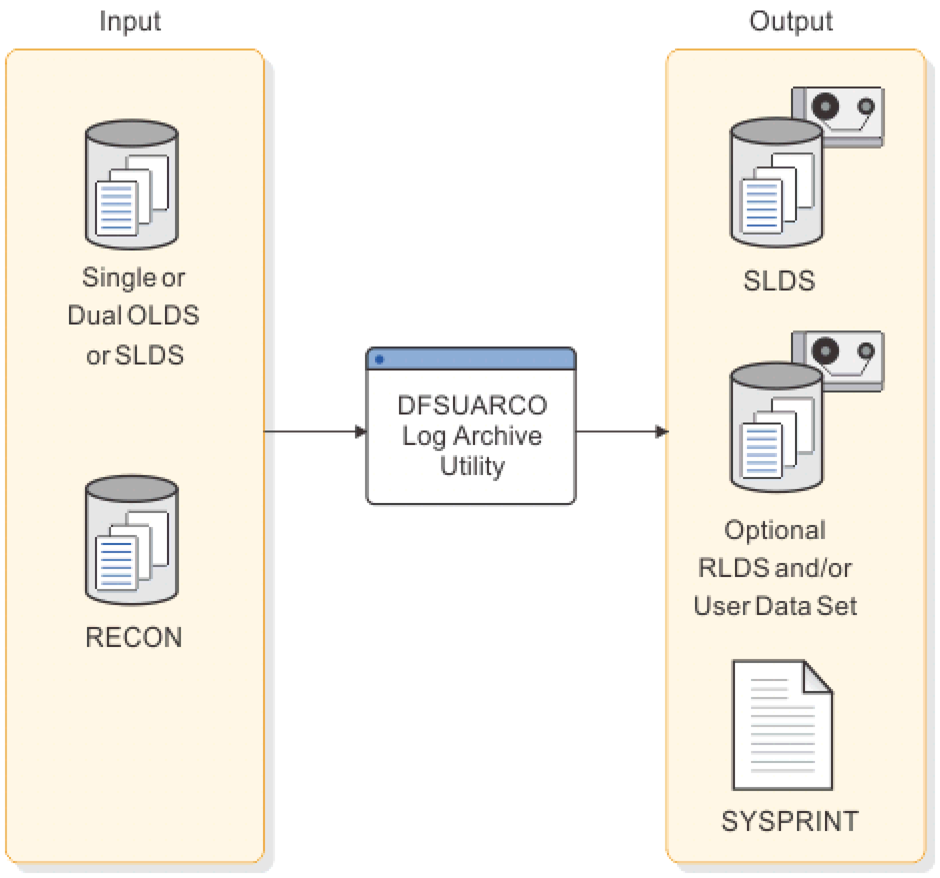

Input
Single or Dual OLDS or SLDS
RECON
DFSUARCO Log Archive Utility
Output
SLDS
Optional RLDS and/or User Data Set
SYSPRINT

Figure 24. Overview of the Log Archive utility

Copying an SLDS or an RLDS
You can use the Log Archive utility to copy an SLDS or RLDS to a new data set; however, you cannot use the **GENJCL.ARCHIVE** command to generate JCL to copy these data sets.

You can also use the Log Archive utility to create an RLDS or user data set. Use the Log Archive COPY control statement to do any of the following:

- Specify the user data sets to which you want log records copied.
- Determine which log records should be copied to a user data set.
- Specify that all log records required for database recovery should be copied to an RLDS.

Customizing archiving
You can write user exit routines to process log records and copy certain log records to user data sets. For example, you can copy all records required for restarting Batch Message Processing programs (BMPs) to a user data set.

To customize archiving, specify the entry points for the exit routine using Log Archive utility control statements. IMS gives control to the exit routines when:

- The Log Archive utility is initialized.
- IMS reads the OLDS.
- The Log Archive utility terminates.

Related concepts
Guidelines for writing IMS exit routines (Exit Routines)
Related reference
Log Archive utility (DFSUARC0) (System Utilities)

Tracing the log

You can trace logging activity to diagnose performance problems or problems with IMS. As with all traces, the trade-off when using log tracing is between increased diagnostic capabilities and the overhead of running the trace.

The overhead of tracing the log can be greater than running other types of trace, especially if you request an external trace, that is, if the trace itself is logged. But you can write the external trace to an external trace data set, and add no extra burden to the OLDS.

Specify log tracing in one of the following ways:

- Use the DLOG parameter on the OPTIONS control statement when you initialize IMS.
- Use the **/TRACE** command and omit the DLOG keyword. You can turn log tracing on and off and control whether it is to be logged to the OLDS or to an external trace data set.

Reducing Fast Path logging

Because IMS holds updates for DEDBs in storage before writing entire VSAM control intervals (CIs), you can reduce the logging for Fast Path data. You reduce the log volume by logging only the changed data for each log record during replace (REPL) calls. You can reduce the logging only if the length of the segment remains unchanged.

Use the LGNR parameter of the IMS or DBC procedures to determine the maximum number of Fast Path DEDB buffer alterations that are to be held before IMS logs the entire VSAM control interval (CI). Use the Fast Path Log Analysis utility (DBFULTA0) to evaluate the value you should use for the LGNR parameter.

Related concepts

Fast Path EXEC parameters in DBCTL (System Definition)

Fast Path EXEC parameters in DCCTL or DB/DC (System Definition)

Related reference

Fast Path Log Analysis utility (DBFULTA0) (System Utilities)

Using DBRC to track batch job logs

An IMS online subsystem always uses DBRC for tracking logs, but a batch job need not use DBRC. If you use DBRC for batch jobs, DBRC tracks which batch jobs create which SLDS.

Recommendation: Use DBRC for batch jobs to eliminate the need to manually keep track of batch SLDSs.

You do not need to create a log for read only (PROCOPT=G) batch jobs, but you do need to create a log for update jobs. For update jobs using DBRC, you cannot use DD NULLFILE or DD DUMMY in the JCL for the log data set.

Specify the use of DBRC during IMS system definition by using the DBRC keyword in the IMSCTRL macro. While IMS is running, you can use the DBRC= execution parameter in the DBBBATCH and DLIBATCH procedures to override the value specified during system definition. If you specify the FORCE keyword during system definition, you must use DBRC, except when you run the Log Archive (batch only), Log Recovery, or Batch Backout utilities.

Consolidating logged database changes for recovery

You can use the Database Change Accumulation utility (DFSUCUM0) to extract and consolidate database change records from SLDSs or RLDSs. The Database Change Accumulation utility optimizes the database change records for recovery and saves them to a change accumulation data set. You can use the change accumulation data set as input to the Database Recovery utility (DFSURDB0).

As IMS runs, the number of SLDSs or RLDSs increases. You can use these data sets to recover a lost or damaged database, but to use them without change would be inefficient for the following reasons:

- Each SLDS or RLDS contains a record of activities of the entire IMS subsystem and of all the data sets for all the databases. Yet when you are recovering a database, you usually only recover a single data set. Thus, much of what is in the SLDS and RLDS does not apply.
- The SLDS and RLDS chronologically stores each change to any single database record. If a record changes 100 times since the last backup of the data set, the SLDS or RLDS includes all 100 changes.

Yet, during recovery, you are only interested in the value the data had at the moment the data set was lost; the other 99 changes are irrelevant.

You can use the Database Change Accumulation utility to sort through your accumulated SLDSs and RLDSs in advance and condense and streamline them. This utility:

- Picks out only those log records relating to recovery of databases
- Sorts these records by data set within a database
- Finds the most recent change in each part of an individual record

As the utility creates the change accumulation data set, IMS compresses repeated single characters, such as blanks and zeros. IMS expands the data again during recovery.

Running the Database Change Accumulation utility is not required, but using it periodically speeds database recovery. Alternatively, you can run the Database Change Accumulation utility only when the need for recovery arises (just before running the Database Recovery utility). Running these two utilities instead of just the Database Recovery utility can reduce the total time needed for recovery, depending on how much unaccumulated log information exists.

Related reference

Database Change Accumulation utility (DFSUCUM0) (Database Utilities)

Input to the Database Change Accumulation utility

In addition to using archived log data (SLDS and RLDS) as input to the Database Change Accumulation utility, you can also use a subset of the IMS log or a previous change accumulation data set. The utility writes the accumulated changes to a new change accumulation data set.

If the log data is on tape, you can specify all log volumes or a subset of log volumes as input to the Database Change Accumulation utility. When you specify a subset of log volumes, DBRC checks whether the subset is complete for each DBDS. A subset of log volumes is complete for a DBDS when all of the following conditions are true:

- The first volume in the subset is the volume with the first change to the DBDS since any of the following events occurred:
  - The last change accumulation.
  - The last image copy.
  - DBRC created the ALLOC record for this area (if the image copy was concurrent).
  - Checkpoint IDs that are found on logs prior to the run time of the image copy are included to ensure all updates are considered if the image copy was taken while updates were being made.
- The remaining volumes are in sequence.
- In a data-sharing environment, all logs containing changes for a DBDS are included.

Use the DBRC **`GENJCL.CA`** command to specify the subset of log volumes. You can request a specific number of log volumes either by volume (use the VOLNUM keyword) or by time stamp (use the CATIME keyword).

You can use a change accumulation data set as input to a later run of the Database Change Accumulation utility whether your subset of log volumes is complete or incomplete; however, you can use a change accumulation data set as input to the Database Recovery utility only if it represents a complete log subset.

Related reference

GENJCL.CA command (Commands)

Purge time for database recovery and Database Change Accumulation utility

A purge time is the timestamp DBRC uses to start collecting changes for a database or area that might need to be recovered using either an image copy or a HALDB Online Reorganization as a starting point for a recovery. Changes prior to this timestamp are purged from change accumulation data sets and are not considered for database recovery.

Image copy purge times

If updates are not occurring (no active allocation records in RECON for the database), the run time of the image copy is used as the purge time. If updates are occurring when an image copy is run, DBRC checks checkpoint timestamps on logs previous to the run time of the image copy. For full function databases (including HALDBs), only one checkpoint on a previous log is considered. For Fast Path databases, two checkpoints on previous logs are considered. In order to keep purge times moving forward in time, ensure checkpoints exist on each log for each IMS updating the database.

The run time of the image copy is selected as the purge time for the following types of image copies for both full function and Fast Path databases:

- Database Image Copy (DFSUDMP0) utility was used to create the image copy while the database was unavailable for update (BATCH).
- Database Image Copy 2 utility invoked DFSMS Fast Replication to take an image copy while the database was unavailable for update processing (SMSOFFLC).
- Database Image Copy 2 was used to create the image copy while the database was unavailable for update processing (SMSNOCIC).
- Online Database Image Copy utility was used to create the image copy (ONLINE).

For other types of fuzzy image copies, DBRC uses an earlier checkpoint ID to ensure that all the changes are included for the Database Recovery or the Database Change Accumulation utility, as long as the database is allocated when the fuzzy image copy is run. If the database is not allocated by any IMS system when the fuzzy image copy is run, the run time of the image copy is selected as the purge time for the database.

Active allocations on all IMS systems when a fuzzy image copy is taken are considered and the earliest overall purge time is selected as follows when an active allocation is found for an IMS system:

- Database Image Copy utility was used to create the image copy while the database was available for update processing (CONCUR). A concurrent image copy is a “fuzzy’ copy that occurs when updates are also being done. The data set uses logs in order to complete the image.
- Database Image Copy 2 utility invoked DFSMS Fast Replication to take an image copy concurrently with update processing (SMSONLC). The image is copy is a “fuzzy” copy so logs must be applied to recover the data set to a usable state.
- Database Image Copy 2 was used to create the image copy while the database was available for update processing (SMSCIC). The image copy is a “fuzzy” copy so logs must be applied to recover the data set to a usable state.

DBRC uses the checkpoint ID on logs to determine purge times with fuzzy image copies to ensure all updates are included as follows:

- **For full function databases (including HALDBs)**: A checkpoint ID found on a log volume previous to the start time of the image copy is selected if the checkpoint ID is earlier than the allocation. Otherwise, the allocation time is selected as the purge time.
- **For Fast Path databases**: When at least two checkpoints are found on log volumes previous to the start time of the image copy, the log volume start time is used if the start time is earlier than the allocation. Otherwise, the allocation time is selected as the purge time.

If a HALDB Online Reorganization (OLR) is found that can be used as input to recovery for a HALDB that is later than the purge time of the image copy selected, the run time of the OLR is used as the purge time.

Change accumulation groups
You can use DBRC to group DBDSs for which the Database Change Accumulation utility accumulates changes. These groups of DBDSs are called change accumulation groups.

You can define a change accumulation group by using the DBRC **INIT.CAGRP** command to write a CAGRP record in the RECON data set. Within this record, DBRC lists the DBDSs that make up the change accumulation group, identified by their database names and data set DD names, and the recovery period, identified by the RECOVPD keyword. A change accumulation group can have a maximum of 32,767 members.

Before you can use the **INIT.CAGRP** command to define the change accumulation group, you must identify each member to DBRC using an **INIT.DBDS** command. A DBDS can belong to only one change accumulation group.

You can add or delete members of a change accumulation group using the **CHANGE.CAGRP** command.

Restriction: Do not issue a **CHANGE.CAGRP** command while the Database Change Accumulation utility is running because you could damage your database integrity.

Defining change accumulation data sets for future use
You can define change accumulation data sets for future use for a given change accumulation group. Use the DBRC **INIT.CA** command to inform DBRC that these data sets exist, and specify the REUSE keyword on the **INIT.CAGRP** command when you define the group.

The GRPMAX keyword of the **INIT.CAGRP** command determines how many change accumulation data sets you can define.

Reusing change accumulation data sets
To allow DBRC to reuse old change accumulation data sets, define a change accumulation group with the **REUSE** keyword and use the DBRC **GENJCL.CA** command to generate the JCL for the Database Change Accumulation utility job.

The Database Change Accumulation utility reuses the oldest change accumulation data set when all available change accumulation data sets for a particular change accumulation group have been used and the maximum number of change accumulation data sets has been reached. Reusing a change accumulation data set means that DBRC uses its data set name, volumes, physical space, and record in the RECON data set as if they were for an empty change accumulation data set.

If you define a group with the **NOREUSE** keyword, rather than reuse the data set, DBRC deletes the RECON record for the oldest change accumulation data set. In this case, DBRC does not scratch the data set. You must scratch the data set or keep track of it, because DBRC is no longer aware of it.

Specifying your choices for the log data sets
These topics describe the choices you must make when defining the various data sets involved in logging.

Related reading: For information on defining CQS data sets, see *IMS Version 15.2 System Definition*.

Defining online log data sets
These topics describe tasks related to defining OLDSs.

Choosing dual or single OLDS logging
IMS can log information to a single data set or to two identical data sets.

Definitions: *Single logging* uses a single data set, *dual logging* uses two data sets, where both data sets are identical. Whether you use single or dual logging, IMS writes information to sets of data sets, as described in "Online log data set" on page 85.

Recommendation: Use dual logging whenever possible because the OLDS is of primary importance to system integrity. Specify dual logging in IMS.PROCLIB using the OLDSDEF statement.

With dual logging, IMS has two options when one of the OLDSs (of a pair) gets an I/O error. The first option is to discard the pair and switch to a good pair. This behavior is just like single logging mode.

The second option is to not have IMS discard the pair of OLDSs unless both pairs of OLDSs fail. Then, if all the good pairs of OLDSs fail, IMS can degrade to single logging mode and use the good OLDS from each pair. From this point, IMS behaves just like if it had been in single logging mode from the beginning. This option also simplifies operational procedures because, for example, you only need to run the Log Recovery utility to clean up the OLDSs when you use (single or dual) logging and a write error occurs.

Essentially, there are three logging states for IMS to run in:

- Single logging, which has no redundancy and has the highest maintenance when an error occurs. When only two good OLDSs remain, IMS will terminate.
- Dual logging with DEGRADE=NO, which behaves just like single logging except that data is written to two LOGS instead of one. And, like in single logging mode, if only two good pairs of OLDSs remain, IMS will terminate.
- Dual logging with DEGRADE=YES. In this case, when each pair of OLDSs has at least one write error, IMS will switch to single logging mode and start logging to whichever OLDSs in the remaining pairs are good.

In all three cases, when IMS gets to the point where it only has two data sets left to write to, it terminates. If, for some reason, IMS has no good data sets to write to, it will terminate with an ABENDU0616. For more information about log errors, see *IMS Version 15.2 Operations and Automation*.

Defining the number of OLDSs
You must define at least three OLDSs (or OLDS pairs) to start IMS. However, you can define additional OLDSs (up to 100).

You must define the OLDSs to be used during initialization in the IMS.PROCLIB data set using the OLDSDEF statement. You can then dynamically allocate additional OLDSs. Specify the OLDSs you want to dynamically allocate using the DFSMDA macro; later you can use the **/START OLDS** command to add an OLDS.

Related reading: For more information on the DFSMDA macro, see *IMS Version 15.2 System Definition*.

When deciding how many OLDSs to define, consider the frequency of archiving and the amount of data you want to keep online. IMS reuses an OLDS only after IMS archives it. The number of OLDSs you define should be consistent with the frequency of archiving; if, for example, you archive frequently, you can probably plan on defining fewer OLDSs.

Recommendation: To avoid system failure, you should define more than the minimum number of OLDSs, even when using dual logging.

Defining space for each OLDS
When defining the size of each OLDS, you also need to consider the size and location (DASD or tape) of the SLDS. If the SLDS is on tape, consider the size of an SLDS volume and how often you intend to archive the OLDS.

If the SLDS is on DASD, you should allocate enough space to contain all of the OLDSs to be archived. You might want to assign enough space to each OLDS so it fills an SLDS volume when it is archived. Or you might want to make each OLDS half the size of an SLDS volume, so you can initiate archive when two OLDSs are full.

Also, when defining the size of each OLDS, consider the amount of data you want to have online for doing such tasks as emergency restart or restarting a BMP. For example, if your system has a lot of activity and processes many transactions, consider defining larger OLDSs; then the records necessary for an emergency restart will more likely be online.

Defining a block size for the OLDS
The main factor that determines OLDS block size is the track size of the OLDS devices. The OLDS block size cannot exceed the device track size. Define a block size that maximizes the amount of log data per track (for example, half-track for 3380 or 3390 DASD). Because IMS writes only full OLDS buffers to the OLDS, a large OLDS block size results in more efficient use of DASD space.

You must choose the block size for the OLDS carefully because changing the size of the OLDS after it has been established requires that you stop online work, archive all OLDSs, and scratch and reallocate them to make sure that their block sizes remain identical. When scratching and reallocating, you must delete OLDS entries from the DBRC RECON data set. If you are changing the OLDS block size from being a multiple of 2048 (not running in z/Architecture® mode) to being a multiple of 4096 (running in z/Architecture mode) to ensure that the WADS gets reformatted according to the change in the OLDS block size, you must run the **`/NRE FORMAT WA`** or **`/NRE FORMAT ALL`** command. After you change the block size of an OLDS, you can restart online work only from an SLDS.

Recommendation: Take a checkpoint soon after the restart so later restarts can use the new OLDS, rather than the SLDS.

The block size of each OLDS must be the same. The OLDS block size must meet all of the following requirements:

- The block size must be a multiple of 2048 bytes (2 KB) if IMS is not running in z/Architecture mode. The block size must be a multiple of 4096 (4 KB), if IMS is running in z/Architecture mode.
- The block size must be at least 6144.

  Recommendation: Use an 8 KB minimum or use the length of the largest message segment.
- The block size must not exceed a maximum of 30,720 bytes. This is the largest multiple of 2048 supported by BSAM.
- With the support of extended format OLDS, the calculation of how many blocks fit on a track requires that you add an extra 32 bytes of SAM overhead to the IMS block size before making the calculation.

Your OLDS can be more than 65,535 tracks if you use the large sequential data set support. To take advantage of this support, hardware that has more than 65,535 tracks must be used.

During initialization, IMS ensures that the block size specified for the OLDS is large enough to handle the maximum length log record. If the block size specified is too small, IMS discards the OLDS data set looks at the next OLDS. If, at the end of initialization, there are not at least three pairs of usable OLDSs, IMS terminates with an 0073 abend.

Defining devices for the OLDS
You configure the OLDS so that a system failure that renders an OLDS inaccessible does not stop the entire system. If you define dual OLDS logging, define each data set in an OLDS pair on different devices and, if possible, on different control units and channels.

The following figure shows an OLDS configuration with multiple devices. In this configuration, if one device fails, you can still access the OLDSs on the other two devices. For example, if device B fails while logging to primary OLDS 1 and secondary OLDS 1, you can still use primary OLDS 1 as input to the Log Archive utility, and IMS continues logging using the next available pair of OLDSs: Primary OLDS 3 and Secondary OLDS 3. If both Primary OLDS 3 and Secondary OLDS 3 have not been archived, IMS continues logging with the next available pair, Primary OLDS 6 and Secondary OLDS 6. The primary OLDS steps from one device to the next and the sequence wraps around.

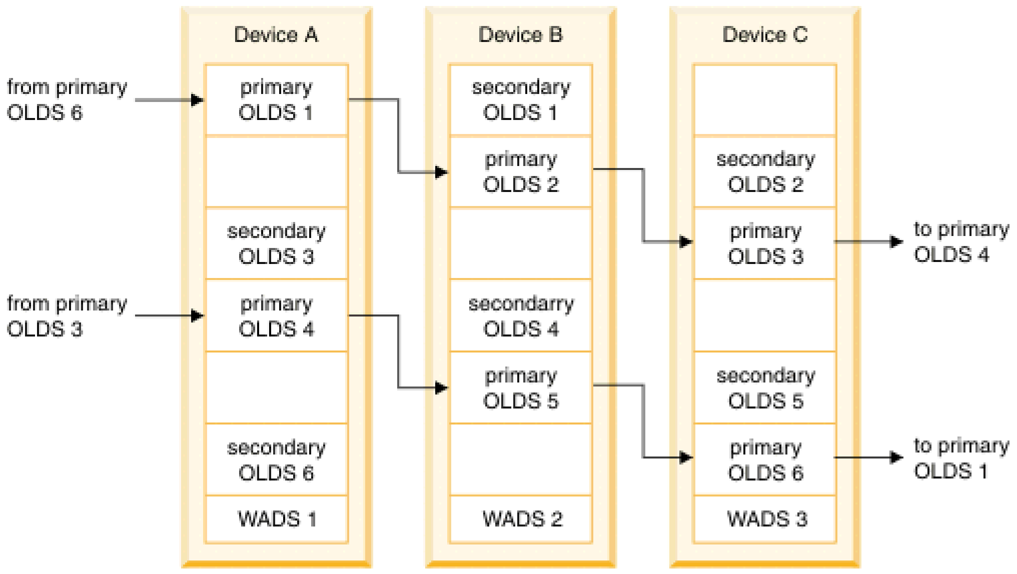


Figure 25. Sample OLDS configuration

Changing OLDS characteristics
Before you scratch and reallocate an OLDS that has been archived, you must delete the log control record in the DBRC RECON data set for this OLDS by using the **`DBRC DELETE.LOG OLDS(`***ddname***`) SSID (`***name*) command.

You must delete the log control record for an OLDS that has been scratched and reallocated because the log control record only indicates that the OLDS has been archived. If you need this OLDS for IMS restart or batch backout, DBRC indicates to IMS to use this OLDS instead of the SLDS created by the archive job.

Related concepts
"Overview of DBRC" on page 473
A feature of the IMS Database Manager that facilitates easier recovery of IMS databases, DBRC maintains information that is required for database recoveries, generates recovery control statements, verifies recovery input, maintains a separate change log for database data sets, and supports sharing of IMS databases and areas by multiple IMS subsystems.

Defining OLDS buffers
You can define from 2 to 9999 OLDS buffers (the default is 5). The number of buffers can be changed during IMS restart.

You might want to increase the number of OLDS buffers in the following circumstances:

- If IMS frequently waits for OLDS buffers
- If you have a high frequency of dynamic backout

Specify the number of OLDS buffers in the BUFNO parameter on the OLDSDEF statement in the LOGGER section of the DFSDF*xxx* PROCLIB member.

IMS places OLDS log buffers above the 2 GB boundary only when they are allocated as DFSMS extended-format data sets and when BUFSTOR=64 is specified with the OLDSDEF statement in the LOGGER section of the DFSDF*xxx* IMS.PROCLIB member.

Defining system log data sets
To define system log data sets (SLDSs), consider choosing dual or single SLDS logging and defining a block size for the SLDS.

- Choosing dual or single SLDS logging

  You can choose either single or dual logging for the SLDS, just as you can for the OLDS and WADS. When archiving to tape, you can force the primary and secondary volumes to contain the same data by specifying the number of log blocks per volume using the force-end-of-volume (FEOV) keyword of the

SLDS control statement. When IMS writes the specified number of blocks, IMS forces end of volume on both the primary and secondary data sets.

To use dual logging, supply both a primary and secondary DD statement for each SLDS.

- Defining a block size for the SLDS

  The block size of the SLDS can differ from the block size of the OLDSs being archived. However, the block size of primary and secondary SLDS must be the same when using the FEOV keyword.

Specifying the recovery log data set
To use dual logging for the RLDS, supply both a primary and secondary DD statement for each RLDS in the JCL for the Log Archive utility.

Specifying the restart data set
Allocate one cylinder of space for the restart data set (RDS). Although the checkpoint ID table uses one track, IMS also writes other recovery information to the RDS.

Contents of the log

IMS records activity on the OLDS. CQS records activity on a z/OS log stream. Each different activity is recorded as a separate log record.

Generally, you do not need to know the content of log records. IMS and CQS identify the correct records and use them automatically when they perform recovery.

The following items, however, should be kept in mind with respect to the logs:

- Log reduction

  As part of log reduction, IMS compresses log data. IMS compresses repeated single characters, such as blanks and zeros, in the segment data portions of log records. This compression applies to updates resulting from DL/I insert (**ISRT**), delete (**DLET**), and replace (**REPL**) calls.

  The counterpart to compression is expansion. During backout and database recovery, IMS expands the data in the database buffer. For change accumulation data sets, IMS expands the data in the change accumulation record.

- Logging and the Data Capture exit routine

  IMS does not write log records for the Data Capture exit routine to show that it has been called, nor does IMS write the exit routine name in any log records. IMS does not differentiate between the application program and the exit routine. If you use the Data Capture exit routine extensively, your IMS system accounting and performance monitoring information is likely to show more system use for the application programs than they actually use.

Related reading: If you need to examine log records to solve a complex recovery problem, see *IMS Version 15.2 Diagnosis*.

Backup

A *backup copy* of a data set serves the same purpose for a recoverable system as a checkpoint: it defines a place from which you can restart processing.

IMS provides utilities to allow you to make several types of backup copies.

Database backup copies

When IMS takes a regular system checkpoint, it records internal control information for DL/I and for Fast Path, but it does not record any of the contents of the database. If the database is lost, examining the last system checkpoint does not allow you to recover it. The system log can tell you what changes have occurred, but without the original database itself, recovery can be impossible.

Recommendation: Make a backup copy of all databases after you initially load them. You should also make new backup copies at regular intervals. More recent backup copies require fewer log change records to be processed during recovery, and thus the time needed for recovery is reduced.

IMS provides several utilities for making backup copies of a database:

- Image copy utilities

  The IMS database image copy utilities allow you to take a "snapshot" of a database before and after changes have been to the database. These snapshots are called *image copies*.

  Definition: An image copy is an as-is image of a database. The image copy utilities do not alter the physical format of the database as they copy it. Image copies are backup copies of your data that help speed up the processes of database recovery and backout.

  The image copy utilities are Database Image Copy utility (DFSUDMP0), Database Image Copy 2 utility (DFSUDMT0), and Online Database Image Copy utility (DFSUICP0).

- The HISAM Reorganization Unload utility (DFSURUL0)

  This utility allows you to process an entire HISAM database in one pass (the image copy utilities process each database data set individually) while it is unloaded and reorganized. This utility runs while the database is offline.

  Recommendation: Make an image copy of the database before unloading it. You can use the image copy for recovery purposes if a failure occurs in the unloading process. You can also create an equivalent of an image copy data set without using the HISAM Reorganization Reload utility.

  After you unload the database, you use the HISAM Reorganization Reload utility (DFSURRL0) before bringing the database back online. If you do not reload the database, an application program can use the database, but will use the old organization of the data set. Your backup copy will not match the log records produced for the database, and you will not be able to use the backup copy to recover the database without damaging your data integrity.

You can run these utilities with or without DBRC. You can also use various z/OS utilities to make your backup copies, but these utilities do not interact with DBRC, which could cause integrity problems depending on how your IMS system is defined.

Related concepts

“Concurrent image copy” on page 508
IMS provides the capability to take an image copy of a database without taking that database offline. This means the database can be updated while the image copy is being taken and some, all, or none of the updates might appear in the image copy.

Related reference

HISAM Reorganization Reload utility (DFSURRL0) (Database Utilities)

Message queue backup copies

Messages (the transaction requests entered by end users, and the responses going back to them) are stored on queues before being processed. If you are not sharing the IMS message queues, messages are stored in the message queue data set, which resides partly on disk and partly in virtual storage.

In a shared-queues environment, IMS messages are kept on a coupling facility. When IMS takes a regular system checkpoint, it does not record the contents of the message queues, just as it does not record the contents of the databases.

When you shut down IMS normally (rather than abnormally), any changed messages in virtual storage (not on a coupling facility) are automatically written to the disk portion of the message queue data set. Therefore, if you periodically shut down IMS, there is little or no need to backup the message queues manually.

When you run IMS for extended periods without shutting it down, you might want to back up the message queues periodically. In a non-shared-queues environment, use the **`/CHECKPOINT SNAPQ`** command to back up the message queues; this command does not shut down IMS.

In a shared-queues environment, you must periodically back up the shared queues. Use the **`/CQCHKPT`** command to copy IMS messages to the Common Queue Server's structure recovery data set.

Backing up the message queues reduces the time required for recovery if problems arise with the message queue data sets.

System data set backup copies

Database data sets and message queue data sets are not the only data sets for which backup copies should be made. You should also make backup copies of the IMS system data sets. These include the ACB library (IMS.ACBLIBx) and the MFS library (IMS.FORMATx).

IMS does not provide any special utilities or commands to make backup copies of system data sets. You can, however, make periodic backup copies using z/OS utilities, such as IEBCOPY. You can make these copies at the same time you make periodic backup copies of z/OS system libraries. You may also find it convenient to back up the system data sets and databases at the same time.

When you make online changes, you should make backup copies of the active data sets after you switch the inactive and active data sets.

You should also periodically back up RECON data sets using the **BACKUP.RECON** command.

Recovery utilities and services

IMS has a number of utilities and services to help you recover and maintain a recoverable system.

These utilities and services are:

- Database-related utilities: Use these utilities to make image copies of your databases, to accumulate and sort records of database changes, to speed up recovery, and to recover databases.
- System-related utilities: Use these utilities to help you manage the system by managing and recovering the log data sets or generating analysis reports.
- Transaction-Manager-related utilities: Use these utilities to help you manage and control your transactions.

IBM also offers a number of IMS productivity tools that can enhance your IMS environment. These tools can help you automate and speed up your IMS utility operations. They can also assist you in analyzing, managing, recovering, and repairing your IMS databases. You can learn more about these tools on the web at IMS Tools.

Database Recovery Control and recovery

The IMS Database Recovery Control (DBRC) facility makes it easier for you to recover IMS databases by extending the capabilities of IMS utilities for database recovery. DBRC can help recover both DL/I databases and Fast Path data entry databases (DEDBs).

DBRC offers two levels of control: log control and share control.

When you use log control only, DBRC controls the use and reuse of OLDSs for IMS.

When you use share control, DBRC:

- Controls the use and reuse of OLDSs for IMS
- Records recovery-related information in the Recovery Control (RECON) data set
- Assists in recovering databases
- Generates job control language (JCL) for recovery-related utilities
- Registers and controls the scheduling of databases

Related concepts

“Overview of DBRC” on page 473
A feature of the IMS Database Manager that facilitates easier recovery of IMS databases, DBRC maintains information that is required for database recoveries, generates recovery control statements, verifies

recovery input, maintains a separate change log for database data sets, and supports sharing of IMS databases and areas by multiple IMS subsystems.

Automated operator interface

The IMS automated operator interface (AOI) allows application programs to issue IMS commands and can intercept IMS messages routed to the MTO.

You can use it to develop procedures tailored to your installation and to automate some parts of the recovery process. This facility is described in "Tools for Automated Operations" in *IMS Version 15.2 Operations and Automation*.

In an IMS complex with the Extended Recovery Facility (XRF), AOI runs on the active IMS subsystem, and after an XRF takeover, AOI runs on the new active subsystem.

Recovery in an IMS DBCTL environment

The IMS DBCTL environment consists of the IMS Database Manager (not the Transaction Manager) and allows one or more transaction management subsystems, such as CICS, to access DL/I databases and DEDBs.

These transaction management subsystems are known as *coordinator controllers* (CCTLs). A CCTL communicates with the DBCTL subsystem using an interface called the database resource adapter (DRA). Connections or paths between a DBCTL control region and a CCTL are created as part of CCTL initialization. These connections are called *threads*.

Because the CCTL is a separate subsystem, failures are isolated between the IMS DBCTL subsystem and the CCTL. That is, failure of the DBCTL subsystem does not generally cause the CCTL to terminate. Likewise, failure of a CCTL does not generally cause failure of the DBCTL subsystem. Generally, the only time a CCTL failure can affect the DBCTL subsystem is if the CCTL had one or more threads executing in DL/I at the time of its failure).

An IMS DBCTL subsystem cannot restart from log data not created by a DBCTL subsystem. Likewise, non-DBCTL IMS subsystems cannot restart using log data created by a DBCTL subsystem. In other words, restarts of an IMS DBCTL subsystem must use log data produced by that system. You can, however, perform database recovery using a combination of log data sets created by other subsystems: IMS DB/DC, batch, or DBCTL.

Because DBCTL requires the DBRC log control facility, you must use DBRC with DBCTL. Log records produced by an IMS DBCTL system are compatible with an IMS DB/DC system (except for subsystem restart).

Recovery in an XRF complex

Use the Extended Recovery Facility (XRF) as an alternate IMS subsystem that monitors the log data of the active IMS subsystem and takes over the processing load of the active subsystem if it experiences a failure.

Everything about non-XRF recovery mechanisms is also true in an XRF complex, but with XRF, IMS can do more to reduce the time that data is unavailable to users after an abnormal event occurs.

You can specify conditions that trigger an automatic takeover, such as:

- A failure of the IMS control region
- A total failure of z/OS
- A single central processor complex (CPC) failure
- An internal resource lock manager (IRLM) failure that requires an IMS STATUS exit routine
- A Virtual Telecommunications Access Method (VTAM) failure that requires an IMS TPEND exit routine

You can also manually initiate an XRF takeover in a non-failure situation to introduce planned changes to your system with minimal disruption to users.

Backward recovery

When IMS or an application program fails, you need to remove incorrect or unwanted changes from the database. Backward recovery or *backout* allows you to remove these incorrect updates.

The three types of backout are:

- Dynamic backout
- Backout during emergency restart
- Batch backout

IMS performs the first two types of backout automatically, and you initiate the last one manually.

IMS automatically (dynamically) backs out database changes in an online environment when any of the following occurs:

- An application program terminates abnormally
- An application program issues a rollback call
- An application program tries to access an unavailable database
- A deadlock occurs

In a batch environment, you can specify that IMS should dynamically back out database changes if the batch job abends, or issues a rollback call.

During restart processing after a system failure, IMS determines if any application programs were executing at the time of failure and if they made changes that need to be backed out. Before IMS restarts, it scans log records for these changes, and then backs out the changes from the affected databases. If IMS runs out of memory while scanning the log, you could get a message telling you to back out the changes manually.

You can use the IMS Batch Backout utility (DFSBBO00) to remove database changes made during execution of a DL/I or DBB region or an online program. You can use the utility to back out changes since the last checkpoint. You can select a specific checkpoint if the batch region does not use data sharing, and if it is not a BMP. For BMPs, do not specify a checkpoint because the utility always backs out BMPs to the last checkpoint on the log.

If dynamic backout or backout during emergency restart fails, IMS stops the databases for which backout did not complete, and retries the backouts when you restart the databases.

Forward recovery

Forward recovery involves reconstructing information from the IMS logs and re-applying it to a database.

With forward recovery, you reconstruct information or work that has been lost, and add it to the database. With backout, you remove incorrect or unwanted changes from information. For an IMS subsystem, you perform the task of forward recovery (IMS does supply utilities to help). Backout of transactions (and their associated database updates) is handled automatically by IMS. Backout usually does not require your involvement, but an application program can control backout processing for its own transactions. Backout of batch programs can be done either automatically or using an IMS utility.

Shutdown of the IMS system

Certain problems are severe enough that they cause the IMS subsystem to fail. In these cases, IMS shuts itself down. In other cases, however, IMS keeps running, so you must shut it down manually before you can perform recovery. You can shutdown either part of the IMS subsystem (partial shutdown), or you can shut down all of it (full shutdown).

Recommendation: Because a primary goal is to keep IMS function available to users, you should only use a full shutdown when a partial shutdown does not recover the system.

When only a part of IMS is malfunctioning, you might be able to shut down only that part and leave the rest of IMS functioning productively. For example, a faulty database can be taken offline (made unavailable to application programs), while IMS and the other databases continue processing. Or a terminal that is malfunctioning can be detached, while all other functions continue unaffected.

To shut down only part of IMS, use a command that stops the component that is malfunctioning.

You can also shut down IMS in a controlled manner. A controlled shutdown is desirable because it saves current information in the system, and allows an easy and accurate restart of the system at a later time.

If DRD is enabled and automatic export is not enabled, any resources that were defined dynamically should be manually exported before shutting down IMS.

To shut down IMS, use the **`/CHECKPOINT`** command with one of the following keywords: FREEZE, DUMPQ, or PURGE.

Related reference

/CHECKPOINT command (Commands)

Recovery during restart

IMS can recover from many types of failures during restart. In some cases, the recovery is automatic, for example, backout of transactions and the associated database updates. In other cases, you request a type of recovery on the restart command, for example, you can tell IMS to rebuild the message queues from a previous backup copy.

Generally, the way you restart IMS corresponds to the way you shut it down, or the way it fails.

- If you only shut down part of IMS (for example, take a database offline or detach a line), you need to restart only that part.
- If you shut down all of IMS or if IMS fails, you need to restart the whole IMS subsystem.

After recovering a failed component, you can restart it. For example, if you take a database offline because of an I/O error and then recover it, you can again make it available to applications.

To restart only part of IMS, use a command that starts the component that is stopped.

Before you can restart the entire IMS subsystem, you must first perform all necessary recovery. You can then restart IMS in one of three ways:

- *Normal restart* initializes an IMS subsystem that has not failed. A normal restart can be either a *cold start* which restarts IMS without reference to any previous execution of IMS, or a *warm start* which restarts an IMS subsystem that was terminated using a **`/CHECKPOINT`** command. The normal restart command is **`/NRESTART`**.
- *Emergency restart* initializes an IMS subsystem that has failed. During an emergency restart, IMS resets transactions and active regions, and restores databases and message queues to the most recent sync point. You must manually restart batch and batch message processing (BMP) regions. The emergency restart command is **`/ERESTART`**.
- *Automatic restart* reduces MTO intervention and can make restart faster because IMS automatically chooses the appropriate restart command. The operator does not enter a restart command; instead you specify automatic restart by coding AUTO=Y in the JCL for the IMS control region.

Complications in recovery

If you could always recover a system simply by knowing where you started and keeping track of what you have done since then, it would be easy. Unfortunately, complications often arise, not only in the data and applications, but in the mechanisms of recovery themselves.

The IMS logs (on DASD or tape) are just as subject to I/O problems as any other data set. You can run out of space, or develop physical problems with the drive. The internal IMS modules involved in recovery are totally dependent on the operating system and hardware; if the system crashes for any reason (for example, power failure, abend, hardware malfunction), the recovery processing performed by these modules might be incomplete.

So, recovering from a problem can be a two-part process: first you must recover the IMS recovery tools themselves, then you can recover the data and applications.

What IMS cannot recover

IMS can automatically recover from some errors, and IMS provides mechanisms to allow you to recover from many more errors, but there are some errors IMS cannot recover.

Generally, these unrecoverable errors are of two types:

- Application logic or input errors

  When you detect an error caused by an application program that is running, backing out the faulty data is fairly simple. But if that program is given the wrong input or is constructed with logic errors, you might not be able to detect such problems because the program will run to completion. After the application program has finished running (or has committed its data), the faulty data can be used by other programs, and the problem can spread throughout your system.

  IMS has no automatic or guaranteed way to back out committed data. You might be able to back out the data manually (for example, by editing individual records in the affected databases), but such a process is likely to be difficult. And the more time that passes after the original errors are introduced, the more programs are likely to have used the bad data, and the less likely your chances of success. Moreover, output that has been produced or actions taken as a result of the faulty data will already be outside the boundaries of the databases.

  Recommendation: To minimize application logic or input errors, you should extensively test new applications (both individually and in an integrated system test) before bringing them online. Extensive validation of user input by application programs also helps minimize input errors.

- Operational errors, including misuse (accidental or malicious) of the IMS recovery facilities

  Successful recovery requires both proper operation of IMS on a daily basis and proper use of the recovery facilities. This proper use includes such things as:

  - Using the correct data sets
  - Avoiding improper job cancellations during regular operations
  - Regularly maintaining the logs
  - Using the correct log volumes in the correct sequence
  - Using the appropriate utilities at the appropriate time

  You should have no difficulties in operating IMS properly, and other topics in this information explain how to set up the necessary procedures and guidelines. However, if you operate IMS, its recovery facilities, or both, improperly, IMS might not be able to correct problems introduced by such improper use.

Overhead of recovery

Although the value of recovery is obvious, it is not without cost. Logging, for example, takes time and space: some amount of processing time is required when a log record is created, and each log record uses space on a data set. Each checkpoint written also takes time and space, as does each backup copy made.

Having a basic recovery scheme is a practical necessity, not really a matter of choice. However, you do have a choice about how elaborate to make your recovery scheme.

Generally, the more extensive recovery preparations you build into your daily operations, the faster you can bring your system back into production after an error occurs. The trade-off is between speed of recovery and the daily overhead of recovery maintenance. You also need to consider the cost of having your online system down.

In judging this trade-off, consider how frequently errors occur that require recovery. Errors will certainly occur, but rare errors merit less recovery preparation than those that occur more frequently. You decision about the extent to which you prepare for recovery mechanisms should be based on knowing your installation's needs and priorities, and weighing them against each other.

IMS Application Menu

The IMS Application Menu provides a common interface to IBM-supplied IMS applications that run on TSO using ISPF.

The following applications run on TSO using ISPF:

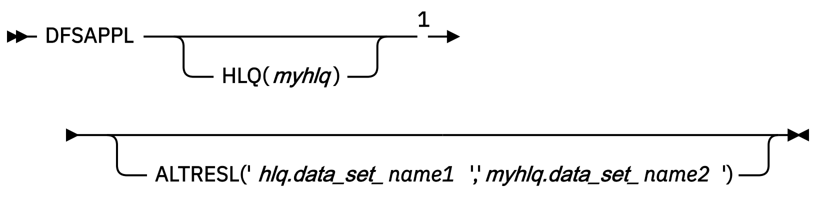
- Single Point of Control (SPOC)
- Manage Resources
- HALDB Partition Definition utility
- IMS Syntax Checker
- Installation Verification Program (IVP)
- IVP Export utility
- IPCS with IMS Dump Formatter (IPCS)
- Abend Search and Notification (ASN)

Tip: To provide access to the IMS Application Menu, include the IMS.SDFSEXEC data set in the SYSPROC DD concatenation.

Attention: Ensure that IPCS is started before the IMS Application Menu is started. Otherwise, message DFSIX103 is displayed.

Use the **DFSAPPL** command to start the IMS Application Menu. You can either use a TSO command or an EXEC command:

- **`TSO %DFSAPPL HLQ`**(*myhlq*)
- **`EXEC 'IMS.SDFSEXEC(DFSAPPL)' 'HLQ`**(*myhlq*)'


DFSAPPL
1
HLQ(*myhlq*)
ALTRESL(' *hlq.data_set_ name1* ',' *myhlq.data_set_ name2* ')

Notes:

[1] The **HLQ** parameter is required the first time you use the command. Thereafter, **HLQ** is an optional parameter.

Where:

DFSAPPL
: Command to start the IMS Application menu

HLQ
: Keyword that enables you to specify the high-level qualifier of the IMS distribution data sets

  The HLQ parameter is required the first time you use the command. If you do not specify it, the command uses the most recently specified high-level qualifier. This parameter is optional.

myhlq
: High-level qualifier of the IMS distribution data sets

ALTRESL
: Keyword that enables you to specify a list of data set names that contain load modules

  If you specify the ALTRESL parameter, you should include SDFSRESL in the list of data set names. If you do not specify the ALTRESL parameter, *myhlq*.SDFSRESL is used as the ISPLLIB data set.

myhlq.data_set_ name1
: Fully-qualified name of a data set that contains load modules

Note: Some applications require an ISPTABL data set. If the ISPTABL data set is allocated, it will continue to be used. If the ISPTABL data set is not in use, a new one is allocated using your TSO prefix or userid as the high-level qualifier.

The IMS Application menu is shown in the following figure.

```
Help
--------------------------------------------------------------------
IMS Application Menu
Command ===> 2____________________________________________________

Select an application and press Enter.

                 1  Single Point of Control (SPOC)
                 2  Manage resources
                 3  Reserved for future use
                 4  HALDB Partition Definition Utility (PDU)
                 5  Syntax checker for IMS Parameters (SC)
                 6  Installation Verification Program (IVP)
                 7  IVP Export Utility (IVPEX)
                 8  IPCS with IMS Dump Formatter (IPCS)
                 9 Abend Search and Notification (ASN)
```

Figure 26. IMS Application Menu

Using the IMS Application menu, you can start any of the TSO or ISPF applications by selecting the application and pressing the Enter key.

You can also link to the IMS Application menu from your local ISPF option menu. The following panel is an example:

```
)BODY
             Local Option Menu
Option ===>_ZCMD

  I    IMS Application Menu
  .
  .
)PROC
&ZSEL = TRANS(TRUNC(&ZCMD,'.')
  I,'CMD(%DFSAPPL HLQ(myhlq)) NEWAPPL(DFS) NOCHECK'
  .
  .
)END
```

Related reading: For more information about any of the ISPF panels within the IMS Application Menu, see the ISPF online help that comes with the applications.

Related concepts

Controlling IMS with the TSO SPOC application (Operations and Automation)

IMS Syntax Checker (System Definition)

IMS installation verification program (IVP) overview (Installation)

Using IPCS and the IMS offline dump formatter (Diagnosis)

Related reference

HALDB Partition Definition utility (%DFSHALDB) (Database Utilities)

Related information

IMS abend search and notification (Messages and Codes)

Part 2. IMS system administration considerations and tasks

These topics describe information and required steps for day-to-day operation of the IMS system.

Chapter 5. z/OS interface considerations

These topics describe information and required steps that you must consider while installing IMS and IRLM on z/OS.

Important: After the z/OS and VTAM interface steps are completed, you must IPL z/OS and specify either CLPA or MLPA=xx, or both. Also, note that IMS can run in 31-bit or 64-bit processing modes.

z/OS interface considerations for IMS

There are many requirements that you must consider and required steps to ensure a complete and correct installation of IMS on z/OS. These topics describe these requirements and required actions.

Preventing installation problems

Be sure to take the following actions to prevent problems during the installation of IMS on z/OS.

- Use z/OS macro libraries for your IMS stage 2 definition. IMS runs only under z/OS .
- Include the libraries from which IMS is loaded and executed in the appropriate authorization table, so that the control region executes as an APF-authorized program. In z/OS, IMS runs as an authorized program.

  Related reading: For information about APF authorization, see IEAAPFxx (authorized program facility list) in *z/OS MVS Initialization and Tuning Reference*.
- Use JOBLIB or STEPLIB DD statements instead of having the IMS.SDFSRESL in LNKLSTxx (those data sets concatenated to SYS1.LINKLIB). If IMS.SDFSRESL is in LNKLSTxx, it is possible for a different IMS release level (whose own IMS.SDFSRESL is not properly APF authorized) to load the modules from LNKLSTxx. The incompatible module release level can cause unpredictable results.
- Update the program properties table. The IMS control region operates as a job step task or as a system task. All control region execution is in supervisor state. See "Updating the IBM-supplied z/OS Program Properties Table" on page 112 for more information.

Related reading: For additional information on maintaining system integrity when running under z/OS, refer to *z/OS MVS Programming: Authorized Assembler Services Reference*.

Setting up JCL

The following list contains the requirements for setting up your z/OS JCL.

- The JOB or STEP libraries must be APF authorized for the control region. For the dependent region, PGMLIB does not need to be authorized and can be concatenated with SDFSRESL as STEPLIB.
- The EXEC statement must specify PGM=DFSMVRC0 for the control region.
- The following libraries must be APF authorized:
  - IMS.SDFSRESL
  - IMS.MODBLKSA and IMS.MODBLKSB
  - IMS.SDXRRESL
  - IMS.SDFSJLIB
  - The library into which your DB2 modules are loaded (DFSESL DD or a JOBLIB or STEPLIB)

Related reading: For more information on z/OS JCL, refer to the information on the system definition process in *IMS Version 15.2 System Definition*.

Configuring JES for dependent regions

The EOM broadcast is always made when an IMS dependent region running as a started task ends, but the EOM broadcast may not be made, and therefore not processed, when a batch job ends.

For IMS dependent regions run as batch jobs, an EOM broadcast is not made until both the job and the initiator used by that job ends. Therefore, if a dependent region is run as a batch job and it encounters a storage problem as described, the EOM clean up service will not be invoked until the initiator ends.

In a JES2 environment, initiators that are used for dependent regions can be set to stop automatically when the job ends by implementing an IBM-defined exit.

Use JES2 exit 32 for Subsystem Interface (SSI) Job Selection to specify that the initiator is to end (drain) or to end and restart when the job ends.

1. Set bit 4 in the response byte to end and then restart the initiator when the job ends.
2. Set bit 5 to end the initiator when the job ends (for more details see z/OS V1R12.0 JES2 Installation Exits SA22-7534-12).

In a JES3 environment, initiators that are used for dependent regions can be set to stop automatically when the job ends by specifying either DYNAMIC or DEMAND for the UNALOPT option of the EXRESC parameter in the initiator's GROUP definition (see z/OS JES3 Initialization and Tuning Guide SA22-7549-11 for more details).

Keeping some required nonstandard z/OS macros in their original libraries

The assembly of certain IMS modules requires z/OS macros not contained on the standard z/OS System Macro libraries. Because these requirements are subject to change due to IMS and z/OS maintenance, keep these macros in their original libraries, and use the JCL generated by IMS for SYS1.MODGEN (or SYS1.AMODGEN).

Updating the IBM-supplied z/OS Program Properties Table

The following modules are predefined in the default Program Properties Table (PPT) that is shipped with z/OS: BPEINI00, CQSINIT0, DFSMVRC0, and DXRRLM00.

If you do not modify the default z/OS PPT, these IMS modules are already included in the PPT, and you do not need to take any action. If you have removed the default entries for these modules, you must reinstate the entries, which will be described in the succeeding sections.

Related reading: For information on updating the PPT, see *z/OS MVS Initialization and Tuning Reference*.

Related tasks

Adding entries to the z/OS Program Properties Table (System Definition)

Adding the IMS entry to the z/OS Program Properties Table

An IMS online environment (DB/DC, DBCTL, DCCTL) requires this PPT entry. If you are only using IMS BATCH, this entry is not needed.

A sample of the required entry is shown below and may be found in the IMS.INSTALIB data set. Refer to "IVP jobs and tasks" in *IMS Version 15.2 Installation* for the correct entry titled "Update SCHEDxx -- PPT Entries".

To make this entry, edit the SCHEDxx member of the SYS1.PARMLIB data set. Add the following entry to the SCHEDxx member:

```
                              /* IMS ONLINE CONTROL REGION            */
    PPT PGMNAME(DFSMVRC0)     /* PROGRAM NAME = DFSMVRC0              */
             CANCEL           /* PROGRAM CAN BE CANCELLED             */
             KEY(7)           /* PROTECT KEY ASSIGNED IS 7            */
             NOSWAP           /* PROGRAM IS NOT-SWAPPABLE             */
             NOPRIV           /* PROGRAM IS NOT PRIVILEGED            */
             SYST             /* PROGRAM IS A SYSTEM TASK             */
             DSI              /* DOES REQUIRE DATA SET INTEGRITY      */
             PASS             /* PASSWORD PROTECTION ACTIVE           */
             AFF(NONE)        /* NO CPU AFFINITY                      */
```

The PPT Entry for program DFSMVRC0 must specify NOSWAP as shown.

Adding the CQS entry to the z/OS Program Properties Table

If you are using CQS, either the CQSINIT0 or the BPEINI00 z/OS PPT entry is required.

A sample of the CQSINIT0 entry is shown below and can be found in the IMS.INSTALIB data set. A sample of the BPEINI00 entry is shown under the heading "Adding the BPE entry to the z/OS Program Properties Table" on page 113. Refer to "IVP jobs and tasks" in *IMS Version 15.2 Installation* for the correct entry titled "Update SCHEDxx -- PPT Entries".

To make this entry, edit the SCHEDxx member of the SYS1.PARMLIB data set. Add the following entry to the SCHEDxx member:

```
                              /* CQS - COMMON QUEUE SERVER            */
    PPT PGMNAME(CQSINIT0)     /* PROGRAM NAME = CQSINIT0              */
             CANCEL           /* PROGRAM CAN BE CANCELLED             */
             KEY(7)           /* PROTECT KEY ASSIGNED IS 7            */
             NOSWAP           /* PROGRAM IS NOT-SWAPPABLE             */
             NOPRIV           /* PROGRAM IS NOT PRIVILEGED            */
             SYST             /* PROGRAM IS A SYSTEM TASK             */
             DSI              /* DOES REQUIRE DATA SET INTEGRITY      */
             PASS             /* PASSWORD PROTECTION ACTIVE           */
             AFF(NONE)        /* NO CPU AFFINITY                      */
             NOPREF           /* NO PREFERRED STORAGE FRAMES          */
```

The PPT Entry for program CQSINIT0 must specify NOSWAP as shown.

Adding the BPE entry to the z/OS Program Properties Table

The BPE program properties table entry, BPEINI00, is used by various IMS address spaces.

These address spaces include:

- Common Queue Server (CQS)
- Database Recovery Control (DBRC) - optional
- IMS Connect
- Open Database Manager (ODBM)
- Operations Manager (OM)
- Resource Manager (RM)
- Structured Call Interface (SCI)

If you are using any of these address spaces, you must have the BPEINI00 entry in your PPT.

To make this entry, edit the SCHEDxx member of the SYS1.PARMLIB data set. Add the following entry to the SCHEDxx member:

```
                              /* BPE - BASE PRIMITIVE ENVIRONMENT     */
    PPT PGMNAME(BPEINI00)     /* PROGRAM NAME = BPEINI00              */
             CANCEL           /* PROGRAM CAN BE CANCELLED             */
             KEY(7)           /* PROTECT KEY ASSIGNED IS 7            */
             NOSWAP           /* PROGRAM IS NOT-SWAPPABLE             */
             NOPRIV           /* PROGRAM IS NOT PRIVILEGED            */
             DSI              /* REQUIRES DATA SET INTEGRITY          */
             PASS             /* CANNOT BYPASS PASSWORD PROTECTION    */
             SYST             /* PROGRAM IS A SYSTEM TASK             */
             AFF(NONE)        /* NO CPU AFFINITY                      */
```

To make the SCHEDxx changes effective, take one of the following actions:

- Restart the z/OS system.
- Issue the z/OS **SET SCH=** command.

Adding the IMS Connect entry to the z/OS Program Properties Table

IMS Connect uses the predefined z/OS Program Properties Table (PPT) entry BPEINI00, but HWSHWS00 is also supported.

The example shown below uses HWSHWS00 instead of BPEINI00. HWSHWS00 is not included in the default PPT, so you must add HWSHWS00 to the PPT if you want to use it.

Recommendation: Use BPEINI00 on your IMS Connect startup JCL and then you do not have to put HWSHWS00 in the PPT.

Example

For TCP/IP communications, add the following entry in the z/OS PPT:

```
PPT PGMNAME(HWSHWS00)      /* PROGRAM NAME = HWSHWS00           */
         CANCEL            /* PROGRAM CAN BE CANCELED           */
         KEY(7)            /* PROTECT KEY ASSIGNED IS 7         */
         SWAP              /* PROGRAM IS SWAPPABLE              */
         NOPRIV            /* PROGRAM IS NOT PRIVILEGED         */
         DSI               /* REQUIRES DATA SET INTEGRITY       */
         PASS              /* CANNOT BYPASS PASSWORD PROTECTION */
         SYST              /* PROGRAM IS A SYSTEM TASK          */
         AFF(NONE)         /* NO CPU AFFINITY                   */
```

Adding the IRLM entry to the z/OS Program Properties Table

If you are using IRLM, the z/OS PPT entry is required.

A sample of the required entry is shown below and can be found in the IMS.INSTALIB data set. Refer to "IVP jobs and tasks" in *IMS Version 15.2 Installation* for the correct entry titled "Update SCHEDxx -- PPT Entries".

To make this entry, edit the SCHEDxx member of the SYS1.PARMLIB data set. Add the following entry to the SCHEDxx member:

```
                           /* IRLM - RESOURCE LOCK MANAGER         */
   PPT PGMNAME(DXRRLM00)      /* PROGRAM NAME = DXRRLM00              */
            CANCEL            /* PROGRAM CAN BE CANCELLED             */
            KEY(7)            /* PROTECT KEY ASSIGNED IS 7            */
            NOSWAP            /* PROGRAM IS NOT-SWAPPABLE             */
            NOPRIV            /* PROGRAM IS NOT PRIVILEGED            */
            SYST              /* PROGRAM IS A SYSTEM TASK             */
            DSI               /* DOES REQUIRE DATA SET INTEGRITY      */
            PASS              /* PASSWORD PROTECTION ACTIVE           */
            AFF(NONE)         /* NO CPU AFFINITY                      */
```

The PPT Entry for program DXRRLM00 must specify NOSWAP as shown.

Installing required IMS links to z/OS

You must install these modules and procedures on your z/OS system.

Stage 2 of IMS system definition might make the following modifications:

- Binds the CTC Channel-end Appendage routine (if the MSC with the CTC option is defined) into IMS.SDFSRESL
- Copies cataloged procedures into IMS.PROCLIB

In addition, you must run JCL to bind the following modules into IMS.SDFSRESL:

- Type 2 SVC routine. If you specify `TYPE=GEN,SVC2=`*xxx* in the DFSIDEF macro, where *xxx* is the type 2 SVC number, sample JCL is generated that you can use to bind the module.
- DBRC Type 4 SVC routine. If you specify `TYPE=GEN,SVC4=`*yyy* in the DFSIDEF macro, where *yyy* is the type 4 SVC number, sample JCL is generated that you can use to bind the module.

The following table provides is an overview of the actions needed in order for your IMS system to run under z/OS.

Table 5. Steps required to run under z/OS depending on the IMS environment

Action	DB batch system	DBCTL system	DB/DC system	DCCTL system
Bind the Type 2 SVC with the z/OS nucleus	Yes	Yes	Yes	Yes
Load the Type 2 SVC from SYS1.NUCLEUS using one of the following methods: • The Nucleus Module Loader facilities • A SYS1.IPLPARM member, NUCLSTxx • A SYS1.PARMLIB member, NUCLSTxx	Yes	Yes	Yes	Yes
Bind the DBRC Type 4 SVC module into LPALIB (or, optionally, into an MLPA library)	Yes	Yes	Yes	Yes

The following table shows the modules that are required by the z/OS interface. The table shows the module name in its distribution library (IMS.ADFSLOAD) and the load module name in its target library (IMS.SDFSRESL) after the module is bound.

Table 6. z/OS interface modules

IMS.ADFSLOAD	IMS.SDFSRESL	Description
DFSVC200 [2]	IGCiii [2]	Type 2 SVC Vector routine [3]
DSP00MVS[1]	IGC00nnn[1]	DBRC Type 4 SVC routine [3]

Notes:

iii
Specifies the Type 2 SVC number

nnn
Indicates the signed decimal Type 4 SVC number, for example, SVC 255 is 25E

1
These modules must be bound with the RENT and REFR attributes.

2
This module must be bound with the RENT, REFR, and SCTR Binder options. The modules are placed in SYS1.NUCLEUS.

3
You must run JCL to bind these modules.

IMS SVC modules

IMS uses a Type 2 supervisor call (SVC), in the range of 200-255, for batch, utility, DBCTL, DCCTL, and DB/DC IMS control program functions. IMS uses a Type 4 supervisor call (SVC), in the range of 200-255, for DBRC functions.

If you are installing different versions of IMS in the same z/OS system, note that the Type 2 SVC and Type 4 SVC are downward compatible. The SVC module for IMS 15 can be used by IMS Version 13 and IMS 14. However, the SVC module for IMS Version 13 cannot be used by IMS 14 or IMS 15, and the SVC module for IMS 14 cannot be used by IMS 15.

Note: The IMS V15 SVC module is the same in all levels of IMS 15. You do not need to reinstall the IMS 15 SVC if you migrate from previous IMS 15 release levels, such as IMS 15.1, to IMS 15.2.

Important: If you specify the **SVC2=** or **SVC4=** parameters in the DFSIDEF0 module, ensure that the DFSIDEF0 module resides in an APF-authorized library that is included in the IMS Control Region JOBLIB or STEPLIB concatenation, or in the z/OS LINKLIST concatenation.

Type 2 SVC

Perform the following steps to use the type 2 SVC:

1. Define a type 2 SVC number by specifying the SVC2= keyword in one of the following locations. If you do not use the SVC2= keyword to define a type 2 SVC number, IMS uses a default value of 254.

   Important: If you use the Open Database Access (ODBA) interface or the database resource adapter (DRA) and you do not want to use the default value of 254 for the type 2 SVC number, you can define the number only by using the DFSIDEF macro of the DFSIDEF0 module.

   - In the DFSPBxxx member of the IMS PROCLIB data set. The value that you specify in the DFSPBxxx member overrides the value that is specified on the DFSIDEF macro.
   - In JCL. The value that you specify in JCL overrides the SVC2= keyword value that is specified in the DFSPBxxx member and the value that is specified on the DFSIDEF macro.
   - On the DFSIDEF macro of the DFSIDEF0 module by specifying the following code:

     ```
     DFSIDEF TYPE=PARM,SVC2=
     ```

     The value that is specified on the DFSIDEF macro is overidden by the value that is specified in the DFSPBxxx member or in JCL.
2. If you used the SVC2= parameter on the DFSIDEF macro to define a type 2 SVC number, assemble and link the DFSIDEF0 module into IMS.SDFSRESL.
3. Run JCL to re-link the type 2 SVC routine, DFSVC200, as IGC*xxx*, where *xxx* is the type 2 SVC number. To generate sample JCL for re-linking the routine, specify TYPE=GEN,SVC2=*xxx* on the DFSIDEF macro.

   Important: If you specify TYPE=GEN on the DFSIDEF macro, you cannot use another form of the TYPE= statement on the macro.
4. Bind the type 2 SVC routine into the z/OS nucleus.

Type 4 SVC

Perform the following steps to use the type 4 SVC:

1. To define a type 4 SVC number, specify the following code on the DFSIDEF macro of the DFSIDEF0 module. If you do not use the SVC4= keyword to specify a type 4 SVC number, IMS uses a default value of 255.

   ```
   DFSIDEF TYPE=PARM,SVC4=
   ```
2. Assemble and link the DFSIDEF0 module into IMS.SDFSRESL.
3. Run JCL to re-link the type 4 SVC routine, DSP00MVS, as IGC00*yyy*, where *yyy* is the EBCDIC representation of the zoned-decimal type 4 value. You can use the sample JCL, which is generated when you specify TYPE=GEN,SVC4=*yyy* on the DFSIDEF macro, to re-link the routine.

   Important: If you specify TYPE=GEN on the DFSIDEF macro, you cannot use another form of the TYPE= statement on the macro.
4. Bind the type 4 SVC routine in an LPA or MLPA library.

Defining the IMS and DBRC SVCs to z/OS

When you define the IMS and DBRC SVCs to z/OS, follow this format:

Example:

```
SVCPARM 254,REPLACE,TYPE(2)
SVCPARM 255,REPLACE,TYPE(4)
```

Related reading: Refer to *z/OS MVS Initialization and Tuning Reference* for information on defining SVCs to z/OS.

Related tasks

"Installing the type 2 SVC module" on page 117
The IMS Type 2 SVC must be incorporated into the z/OS nucleus.

Installing the type 2 SVC module

The IMS Type 2 SVC must be incorporated into the z/OS nucleus.

Attention: The SYS1.NUCLEUS must not have secondary extents. z/OS cannot recognize secondary extents.

To incorporate the IMS Type 2 SVC into the z/OS nucleus, perform one of the following tasks:

- Bind the Type 2 SVC with the z/OS nucleus.

  You can bind the Type 2 SVC with the z/OS nucleus using one of the two following steps.

  a) Invoking the Binder utility through a batch job

  b) Creating and then performing a RECEIVE and APPLY for an SMP/E USERMOD
- Load the Type 2 SVC from SYS1.NUCLEUS using the Nucleus Module Loader facilities.

  a) Create a Nucleus Module List (NML) that contains the list of IMS SVCs that you want loaded into the z/OS nucleus. IMS uses the IEANS001 NML.

  b) Assemble and bind the Type 2 SVC into SYS1.NUCLEUS.

  This method is included as an example in the IVP materials.
- Load the Type 2 SVC from SYS1.NUCLEUS using a SYS1.IPLPARM member, NUCLSTxx.

  a) Bind the IMS SVCs from IMS.SDFSRESL into SYS1.NUCLEUS.

  Attention: Determine, from the z/OS systems programmer, the appropriate NUCLSTxx member to use. Note that the LOADxx member and its associated NUCLSTxx member must both reside in SYS1.IPLPARM. If the 2 members are not in this library, IMS will enter a **Disabled Wait** state and the IPL process stops.

  b) Define an INCLUDE statement for the IMS SVC in the NUCLSTxx member of SYS1.IPLPARM.
- Load the Type 2 SVC from SYS1.NUCLEUS using a SYS1.PARMLIB member, NUCLSTxx.

  a) Bind the IMS SVCs from IMS.SDFSRESL into SYS1.NUCLEUS.

  Attention: Determine, from the z/OS systems programmer, the appropriate NUCLSTxx member to use. Note that the LOADxx member and its associated NUCLSTxx member must both reside in SYS1.PARMLIB. If the 2 members are not in this library, IMS will enter a **Disabled Wait** state and the IPL process stops.

  b) Define an INCLUDE statement for the IMS SVC in the NUCLSTxx member of SYS1.PARMLIB.

Enabling memory-based data set ENQ management

The MEMDSENQMGMT function of z/OS enables jobs and subsystems to use memory-based data set ENQ management for dynamically allocated data sets, which is faster than scheduler work area-based (SWA-based) data set ENQ management. When you enable this function on your z/OS system, data set ENQs for dynamically allocated data sets are managed in memory.

To enable the MEMDSENQMGMT function, use one of the following methods:

- Update the ALLOCxx PARMLIB member to set the SYSTEM MEMDSENQMGMT value to ENABLE:

```
SYSTEM MEMDSENQMGMT(ENABLE)
```

This method is recommended because the setting remains in effect across IPLs.

- Issue the system command **SETALLOC SYSTEM,MEMDSENQMGMT=ENABLE**, and restart IMS to make the change effective.

IMS abend formatting module

IMS dynamically installs the IMS abend formatting module (DFSAFMX0). No user setup is required.

Although module DFSAFMD0 is not required, this module is still included to support users who include DFSAFMD0 in LPA directly from the IMS library.

Uninstalling the abend formatting module (DFSAFMD0)

If DFSAFMD0 is installed from a prior version of IMS, you can remove it from the host z/OS system if no IMS Version 10 or earlier code runs on the z/OS system.

To uninstall the module DFSAFMD0:

1. Remove the name DFSAFMD0 from the IEAVADFM CSECT of module IGC0805A in SYS1.LPALIB. Removing this name prevents the operating system from installing module DFSAFMD0 as an abend formatting exit at the next IPL.
2. Remove module DFSAFMD0 from SYS1.LPALIB or the MLPA library where module DFSAFMD0 was bound.
3. Restart with CLPA to enable these changes.

If you previously used the AMASPZAP utility to zap module DFSAFMD0 into the IEAVADFM CSECT (as is done in the IMS IVP), you must use the AMASPZAP utility to remove the name DFSAFMD0 from the IEAVADFM CSECT. The IEAVADFM CSECT is a table of 8-byte entries, followed by a final 4-byte entry that contains zeros to indicate the end of the exit name list. Each 8-byte entry contains the name of a dump formatting exit routine. If DFSAFMD0 is not the last entry in the table, then in addition to removing the DFSAFMD0 entry, you must move any subsequent entries to ensure that no all-zero entries exist before the end of the table.

The following example shows how to remove module DFSAFMD0 from the IEAVADFM CSECT.

1. Use the AMASPZAP utility to dump the current contents of the IEAVADFM CSECT:

   ```
   //DMPVADFM JOB ...
   //STEP001  EXEC PGM=AMASPZAP
   //SYSLIB   DD   DSN=SYS1.LPALIB,DISP=SHR
   //SYSPRINT DD   SYSOUT=A
   //SYSIN    DD   *
    DUMP IGC0805A IEAVADFM
   /*
   ```

2. Examine the contents of the IEAVADFM CSECT from the AMASPZAP dump job output. Locate the entry that contains module DFSAFMD0 (in hexadecimal: X'C4C6E2C1C6D4C4F0'):

```
**CCHHR- 0022000421   RECORD LENGTH- 000BA0         MEMBER NAME  IGC0805A  CSECT NAME  IEAVADFM
 000000   C4C6E2C1   C6D4C4F0   D4E8C4D4   D7E7F0F0     00000000   00000000   00000000   00000000
 000020   00000000   07FE0000   00000008   00000000
```

3. Use the AMASPZAP utility to replace the entry that contains module DFSAFMD0 with zeros. In the example output above, module DFSAFMD0 is the first entry in the IEAVADFM CSECT, and one other entry follows. To remove module DFSAFMD0, entry 2 must be moved to become entry 1, and entry 2 must be zapped to be all zeros, as shown:

   ```
   /ZAPVADFM JOB ...
   /STEP001  EXEC PGM=AMASPZAP
   /SYSLIB   DD   DSN=SYS1.LPALIB,DISP=SHR
   /SYSPRINT DD   SYSOUT=A
   /SYSIN    DD   *
   NAME IGC0805A IEAVADFM
   VER  0000 C4C6E2C1C6D4C4F0
   ```

```
VER  0008 D4E8C4D4D7E7F0F0
REP  0000 D4E8C4D4D7E7F0F0
REP  0008 0000000000000000
```

Adding the offline dump formatting routine to the print dump exit control table

Add the offline dump formatting module name to the Print Dump Exit Control Table in SYS1.PARMLIB member BLSCECT.

The entry must contain:

Module name DFSOFMD0
Exit flag 0
User verb IMSDUMP

An IMS Dump Formatter is also available from the component analysis section of the IPCS dialogs (IPCS ISPF selection 2.6).

If SDFSRESL is not in LNKLSTxx, IPCS users must have SDFSRESL available in the JOBLIB or STEPLIB concatenation in order to be able to load DFSOFMD0.

Related reading:

- For a description of the exit control table, see *z/OS MVS Initialization and Tuning Reference*.
- For more information about controlling IMS dumping options and about the Offline Dump Formatter, see *IMS Version 15.2 Diagnosis*.

Binding the DBRC type 4 SVC

Bind the DBRC Type 4 SVC into an LPALIB or an MLPA library. It is named IGC00nnn, where nnn is the signed decimal SVC number.

Authorizing IMS system data sets in the Authorized Program Facility

If you use JOBLIB/STEPLIB with region types of CTL (DB/DC region type), DBC (DBCTL region type), or DCC (DCCTL region type), all concatenations of the JOBLIB/STEPLIB must be APF authorized.

The following IMS system data sets must be APF authorized:

- IMS.SDXRRESL
- IMS.SDFSRESL
- IMS.SDFSJLIB
- IMS.MODBLKSA, IMS.MODBLKSB
- DFSESL DD, or the JOBLIB or STEPLIB into which your DB2 modules and tables are loaded

In addition to these data sets, in a DB/DC or DCCTL environment, SYS1.CSSLIB must be APF authorized. This is true regardless of whether you use APPC/z/OS. Even though SYS1.CSSLIB is in LNKLSTxx and LNKLSTxx is authorized, you must also have SYS1.CSSLIB in IEAAPFxx, because IMS accesses SYS1.CSSLIB without using the LNKLSTxx concatenation. SYS1.CSSLIB must be explicitly APF-authorized.

Recommendation: Do not have the IMS.SDFSRESL in LNKLSTxx when running multiple levels of IMS or when migrating to a new version or release level.

IMS conforms to z/OS rules for data set authorization. If you authorize an IMS job step, authorize all libraries used in that job step. To run an IMS batch region as non-authorized, concatenate a non-authorized library to IMS.SDFSRESL. To make this concatenation, the batch job must contain a DFSRESLB DD statement pointing to IMS.SDFSRESL.

Authorizing libraries to z/OS can also be done through the PROGxx member of the SYS1.PARMLIB.

Related reading: Refer to information on IEAAPFxx in *z/OS MVS Initialization and Tuning Reference*.

Updating the APPC/MVS administration dialog

To use the APPC/MVS Administration Dialog utility with IMS TP Profiles, you must first add "IMS" as a transaction scheduler. To do this, you must add one line to the non-display panel ICQASE00 where the variable QASTSPE is defined.

The format of the line is as follows:

```
   IMS,DFSTPPE0'
```

You must also change the single quotation mark (') on the current last line of the assignment to a plus sign (+).

In addition, IMS.SDFSEXEC must be added to the TSO SYSPROC concatenation, and IMS.SDFSPLIB must be added to the TSO ISPPLIB concatenation.

For more information on modifying this panel, see "Customizing the Dialog" in *z/OS MVS Planning: APPC/MVS Management*.

RRS Archive Log Stream considerations

When using the z/OS Resource Recovery Services feature, be aware that the optional RRS Archive Log Stream has been found to generate large amounts of additional logging to the MVS logger.

If used, RRS Archive Logging should be monitored closely for system logstream performance impact. Neither IMS or RRS use the RRS Archive Log Stream data for diagnostic purposes, so it is not necessary to use this function.

System Management Facility DDCONS parameter considerations

The setting of the System Management Facility (SMF) DDCONS parameter should be reviewed to avoid possible processing delays, as described in APAR OY31613.

The option is DDCONS. The two possible suboptions are YES and NO.

YES
(Default) Requests that the consolidation be done

NO
Requests that the consolidation not be done

z/OS interface considerations for IRLM

There are many requirements that you must consider and required steps to ensure a complete and correct installation of IRLM on z/OS.

The following steps describe these requirements and required actions.

- Add the IRLM CTRACE module to the z/OS link list

  The IRLM CTRACE start/stop routine load module, DXRRL183, must reside in the z/OS Link List (LL). This module also contains the automatic restart manager (ARM) support for IRLM.
- Authorize IRLM in the Authorized Program Facility (APF)

  The IMS.SDXRRESL system data set must be APF authorized.
- Create IRLM subsystem names

  Unless you have deleted them, z/OS preconditioning has already defined IRLM and JRLM as subsystems names. You can use these names, or you can define your own. Create a z/OS subsystem name entry for each IRLM to be executed on the z/OS system. When two IRLMs reside in the same z/OS system, each must have a unique z/OS subsystem name.

  Each message that the IRLM issues includes the IRLM z/OS subsystem name (IRLMNM on the start procedure) concatenated with the ID (IRLMID on the start procedure). A naming convention that allows easy identification of which IRLM issued a specific message is recommended. The following IRLM command displays all of the IRLM names and IDs associated with this IRLM or sharing group.

```
F irlmproc,STATUS,ALLI
```

- Add an IRLM entry to the z/OS Program Property Table (PPT)

  Unless you have deleted it, z/OS preconditioning has already defined a PPT entry for DXRRLM00.
- Update the print dump exit control table

  The entry must contain:

  Module name DXRRLM50
  Exit flag 0
  User verb IRLM

  Ensure that one of these is true:

  – The print dump formatting module DXRRLM50 is in SYS1.LINKLIB.
  – The job that prints the dump contains a JOBLIB or STEPLIB statement specifying the library containing the modules.

  Related reading:

  – For more information about the Offline Dump Formatter, see *IMS Version 15.2 Diagnosis*.
  – For information about adding an IRLM entry to the PPT, see "Adding the IRLM entry to the z/OS Program Properties Table" on page 114.
  – Refer to z/OS MVS Initialization and Tuning Reference for information about:
    - Defining a subsystem to z/OS.
    - IEAAPFxx.
    - Responding to the messages and setting up PARMLIB members to contain trace options and parameters.

Chapter 6. VTAM interface considerations

If your IMS system requires VTAM, the VTAM mode table must contain entries for all VTAM terminals defined to IMS.

You can use the table entry name at logon as any of the following:

- LOGMODE parameter on the **`VTAM VARY`** command
- MODE parameter on the **`/OPNDST`** command
- Parameter on the other terminal's **`INIT SELF`** command
- MODETBL parameter of the TERMINAL macro

The MODETBL parameter overrides any other entry supplied with the ACF/VTAM LOGON or SCIP exit CINIT. The MODETBL name for all parallel sessions with a given terminal is the same. Do not specify MODETBL for cross-domain resources.

The mode table entry creates the session parameters and thus controls the session established between IMS and the terminal.

Related reading: For a list of the BIND parameters for VTAM logical units, refer to *IMS Version 15.2 Communications and Connections*.

Define all of the following terminals:

- 3600, 3614, and SLU P as LUTYPE=0
- SLU 1 as LUTYPE=1
- SLU 2 as LUTYPE=2
- LU 6 as LUTYPE=6

A 3770P or 3790 terminal defined as SLUTYPE1 must be defined as unattended in its mode table entry. You can define a SLU 1 terminal as an exception or definite response for the secondary terminal. For terminals defined as SLUTYPEP, no options are allowed in the first 7 bytes of the **`BIND`** command.

When you specify PARSESS=NO in the VTAM APPL macro for IMS, VTAM parallel session support is not included. In this case, IMS counts as '1' within the MAXAPPL keyword of the VTAM START parameter.

When you specify PARSESS=YES in the VTAM APPL macro for IMS, VTAM parallel session support is included in the system. IMS counts as '2' within the MAXAPPL keyword of the VTAM START parameters.

Related reading: For information on IMS support for parallel sessions, see *IMS Version 15.2 Communications and Connections*.

Important: After the z/OS and VTAM interface steps are completed, you must start z/OS and specify either CLPA or MLPA=xx, or both.

Setting the Network Control Program (NCP) delay

Set the value of the DELAY parameter on the HOST macro to 0 or as low as possible considering the other work in your system.

Chapter 7. CSL administration

The tasks associated with administering a CSL are typically performed by the system administrator or system operator.

Using the z/OS Automatic Restart Manager with the CSL

A CSL address space (ODBM, OM, RM, SCI), if requested, can register with the z/OS Automatic Restart Manager (ARM). The ARM is a z/OS recovery function that can improve the availability of started tasks. When a task fails or the system on which it is running fails, the ARM can restart the task without operator intervention.

IBM provides policy defaults for automatic restart management. You can use these defaults, or you can define your own ARM policy to specify how CSL address spaces should be restarted. The ARM policy specifies what should be done if the system fails or if a CSL address space fails.

To enable the ARM, you can specify ARMRST=Y in one of two ways:

- In the CSL address space initialization PROCLIB member data set
  - CSLDIxxx for ODBM
  - CSLOIxxx for OM
  - CSLRIxxx for RM
  - CSLSIxxx for SCI
- As an execution parameter

When ARM is enabled, the CSL address spaces register to ARM with an ARM element name, which are defined in the following table.

Table 7. ARM element names

CSL address space	ARM element name
ODBM	"CSL" + odbmname + "OD"
OM	"CSL" + omname + "OM"
RM	"CSL" + rmname + "RM"
SCI	"CSL" + sciname + "SCI"

Note: The name of the CSL address space is the name defined either as an execution parameter, or in the initialization PROCLIB member data set of that CSL address space. For example, if OMNAME=OM1A in the CSLOIxxx PROCLIB member data set, the ARM element name is CSLOM1AOM.

Use the appropriate ARM element name in your ARM policy for each CSL address space. For more information, see *z/OS MVS Setting Up a Sysplex*.

An abend table exists in the module for each CSL address space:

- CSLDARM0 for ODBM
- CSLOARM0 for OM
- CSLRARM0 for RM
- CSLSARM0 for SCI

The table lists the abends for which the ARM does not restart the CSL address space after the abend occurs. You can modify this table.

Administering global online change

This topic covers enabling and disabling global online change for an IMSplex. This topic also covers maintaining an IMSplex with mixed online change scope.

Enabling global online change

Enabling global online change does not require that you shut down all the IMS systems in the IMSplex at the same time. You can enable global online change one IMS at a time. After all of the IMS systems have switched to supporting global online change, the IMSplex only supports global online change.

Global online change requires a CSL. The following steps assume that the CSL environment is already established. To enable global online change, you must perform the following steps once:

1. Define the OLCSTAT data set characteristics.
2. Initialize OLCSTAT.

After you have defined and initialized the OLCSTAT data set, you can perform the following on one IMS at a time:

1. Remove MODSTAT DD statements from IMS control region JCL.
2. Remove MODSTAT2 DD statements from IMS control region JCL (if XRF).
3. Define DFSCGxxx PROCLIB member data set parameters related to global online change.
   a. Specify OLC=GLOBAL to enable global online change.
   b. Specify OLCSTAT= with the name of the online change data set. All IMS systems in the IMSplex must specify the same OLCSTAT data set.
4. Shut down IMS.
5. Cold start IMS.

When migrating batch IMS systems, you can enable Coordinated (global) Online Change by modifying the DBBATCH JCL:

- Add the OLCSTAT DD statement.
- Remove the MODSTAT DD statement.

Disabling global online change

When disabling global online change you do not have to shut down all the IMS systems in the IMSplex at the same time. Also, after you disable global online change, the IMSplex only supports local online change.

When you are ready to disable global online change, perform the following steps on one IMS at a time:

1. Shut down IMS.
2. Modify the DFSCGxxx PROCLIB member data set parameters for online change to enable local online change by changing OLC=GLOBAL to OLC=LOCAL.
3. Run INITMOD job to initialize the MODSTAT data set.
4. Include MODSTAT DD statements to IMS control region JCL.
5. Include MODSTAT2 DD statements to IMS control region JCL (if XRF).
6. Cold start the IMS.

Typically, an IMS that was down during global online change has to cold start when it comes back up. Cold start might be required to prevent restart from processing log records against the current online change libraries that do not match, potentially causing restart to fail or causing a severe error later on. However, if the IMS was down for only the last global online change and the restart does not conflict with the last global online change, then the IMS is permitted to warm start.

If an IMS was down during only the last online change and its restart type conflicts with the last online change type, or IMS was down for two or more global online changes, IMS must cold start.

Maintaining an IMSplex with mixed online change scope

If the IMSplex contains a mix of IMS systems that support global online change and those that do not, you have to manually coordinate online change with the global online change.

The IMS systems that are defined with OLC=GLOBAL can participate in the global online change.

For example, you can have a mixed IMSplex when enabling global online change across an IMSplex one IMS at a time. If you choose to perform online change during this migration, you have to manually coordinate online change on the IMS that specifies OLC=LOCAL with the global online change.

Related reading: For more information about the global online change process, see "Overview of the global online change function" on page 441.

Administering automatic RECON loss notification

RECON loss notification is automatic after the IMSplex name for the RECON is specified.

Automatic RECON loss notification is activated as soon as one DBRC instance processes a RECON error. That error is immediately communicated to other DBRC instances through SCI. The DBRC instances that are notified of the RECON error issue a message (DSP1141I) indicating that they have been notified.

All DBRC instances that share a set of RECONs must also use the same IMSplex to ensure notification of RECON errors to all DBRC instances. The first use of RECON loss notification for a given set of RECONs sets the IMSplex name in the RECONs. Any subsequent attempts at RECON access using a different IMSplex or no IMSplex are rejected, and message DSP1136A is issued.

If you have ARLN active or parallel RECON access enabled, and you want to change the IMSplex associated with a set of RECONs, use the parameter IMSPLEX() | NOPLEX, which is on the **`CHANGE.RECON`** command. If any DBRC instances are active when the **`CHANGE.RECON IMSPLEX() | NOPLEX`** command is executed, the command is rejected, and message DSP1137I is issued. The following list briefly summarizes the procedures to alter the IMSplex associated with a set of RECONs:

1. Wait for all DBRC activity on the current IMSplex to cease.
2. Submit a DBRC utility job to change the IMSplex name.
3. Alter the IMSplex name in the user exit routine, DSPSCIX0.
4. Ensure that the new IMSplex SCI is ready.
5. Resume DBRC activity on the new IMSplex.

Note: The IMSplex name in the RECON should only be changed when your SCI registration exit routine is changed to return the new IMSplex name.

To stop or reset automatic RECON loss notification, you must clear the IMSplex name. The following briefly summarizes the procedures to stop automatic RECON loss notification associated with a set of RECONs:

1. Wait for all DBRC activity on the current IMSplex to end.
2. Submit a DBRC utility job to change the IMSplex name.
   a. Issue the command **`CHANGE.RECON IMSPLEX() | NOPLEX`**.

      Requirement: If parallel RECON access is enabled and you intend to disable ARLN, you must disable parallel RECON access by issuing the **`CHANGE.RECON ACCESS(SERIAL)`** command prior to issuing the **`CHANGE.RECON NOPLEX`** command.

Related reading:

- For detailed information about automatic RECON loss notification, see "RECON loss notification" on page 555.
- For more information about parallel RECON access, see "Accessing the RECON data sets in parallel mode" on page 532.

Monitoring an IMSplex

Monitoring the individual IMS systems in an IMSplex with CSL is not different from monitoring a standalone IMS system in any other environment. Monitoring the IMSplex as a whole, however, is different than monitoring an individual IMS system.

You can check system status by issuing the **`QRY IMSPLEX SHOW(ALL)`** command, which displays what IMS systems are active in the IMSplex and their statuses.

There are several other QUERY commands that are useful for monitoring certain aspects of the IMSplex. QUERY commands operate on an IMSplex-wide basis unless they are routed to specific IMS systems. For example:

- QRY TRAN QNT(GT,*nn*) where *nn* is a number you specify.

  The response displays the transactions that have a high queue count.
- QRY DB STATUS(ALLOCF,STOSCHD)

  The response displays the databases that have an allocation failure or that have been stopped. There are various other statuses that can be specified.

Diagnosing IMSplex problems

Each of the Common Service Layer address spaces, OM, RM, and SCI, produces SDUMPs for internal errors. You can find the CSL dumps in the SYS1.DUMP data sets.

To successfully diagnose IMSplex problems, you might need to collect one or more of the following types of system information:

- The SYSLOG from every logical partition (LPAR) where a CSL member resides.
- Output from the type-2 **QUERY** command to display the members of the IMSplex and their status.
- A z/OS SVC dump of the CSL address spaces.

You can write OM log records to a z/OS Logger data stream by specifying the AUDITLOG= parameter of the IMSPLEX keyword on the CSLOIxxx PROCLIB member data set. The log records provide an audit trail of command input, associated command output, and unsolicited message output.

You can view the messages in the audit trail using the TSO SPOC. Use the action bar to select **SPOC** > **Audit Trail** and fill in the data stream name and the time of interest.

For more information on diagnosing IMSplex problems, see *IMS Version 15.2 Diagnosis*.

Chapter 8. CSL ODBM administration

The CSL Open Database Manager (ODBM) supports access to databases that are managed by IMS DB in DBCTL and DB/TM IMS systems in an IMSplex.

ODBM manages database connection and access requests from application programs that use the following resource adapters and APIs:

- IMS Universal Database resource adapter
- IMS Universal JDBC driver
- IMS Universal DL/I driver
- Open Database Access interface (ODBA)
- ODBM CSLDMI interface

Except for ODBA application programs and application programs written directly to the CSLDMI interface, all database connection and access requests are routed to ODBM from the resource adapters and APIs through IMS Connect. ODBM routes the database connection and access requests received from IMS Connect to the IMS systems that are managing the requested database.

The IMS Universal drivers that communicate with ODBM through IMS Connect use the open standard Distributed Relational Database Architecture (DRDA) as the low-level communication protocol. ODBM translates the incoming database requests from the Distributed Data Management (DDM) architecture that is a part of DRDA into the DL/I calls expected by IMS. When ODBM returns the IMS output to the client, ODBM translates the response back into the DDM protocol.

ODBA application programs and application programs that use the ODBM CSLDMI interface do not use DRDA but instead use the DL/I calls supported by the ODBA interface.

Existing ODBA applications can also use ODBM to protect IMS from abends caused by the unexpected termination of the ODBA applications during DL/I processing. ODBM support for ODBA application programs requires no changes to the application program itself and only minor changes to the DFSPRP macro that defines the connections to IMS DB.

From the ODBM perspective, application programs that interface directly with ODBM through the CSLDMI interface, such as IMS Connect, are ODBM clients. Users can develop their own ODBM clients by using the ODBM CSLDMI interface. ODBM client application programs can access databases that are managed by IMS DB on any LPAR in an IMSplex

Related concepts
"Overview of the CSL Open Database Manager" on page 35
Open Database Manager (ODBM) provides distributed and local access to IMS databases that are managed by IMS DB systems configured for either the DBCTL or the DB/TM environments in an IMSplex.

ODBM client registration in a CSL

Before application programs can access IMS databases through ODBM, they, or their application server, must register with ODBM as a client. An ODBM client registers with ODBM by issuing the CSLDMREG request of the ODBM CSLDMI interface.

After the ODBM client has registered with ODBM, the application programs running under them can issue ODBA calls to IMS through ODBM.

Prior to registering with ODBM, the ODBM client must first register with SCI. When the ODBM client is done issuing ODBA calls to IMS, it can deregister by issuing a CSLDMDRG request.

For more information about the ODBM CSLDMI interface, including the CSLDMREG and CSLDMDRG requests, see *IMS Version 15.2 System Programming APIs*.

ODBM and RRS

The CSL Open Database Manager (ODBM) can run with or without z/OS Resource Recovery Services. By default, ODBM runs with RRS.

When ODBM runs with RRS, RRS is the sync point manager and coordinates all the resource managers, including those other than IMS that are associated with the unit of recovery (UOR). ODBM uses the two-phase sync point protocol when running with RRS.

When ODBM runs without RRS, ODBM functions similarly to a coordinator controller subsystem (CCTL) that connects to IMS DB through the database resource adapter (DRA) interface. However, unlike a CICS CCTL, ODBM does not function as a syncpoint manager. Any syncpoint management must be performed by the ODBM client.

You must run ODBM with RRS (RRS=Y) if you use WebSphere Application Server or WebSphere Liberty Profile Server and access IMS databases with type-2 IMS Universal Database resource adapter connectivity. To learn more, see IMS Universal drivers: WebSphere Application Server Liberty type-2 connections (Communications and Connections).

For more information about the DRA interface, see "The database resource adapter (DRA)" in Database resource adapter (DRA) (System Programming APIs).

RRS support for ODBM is controlled by the RRS keyword in the ODBM initialization PROCLIB member CSLDIxxx. Learn more at CSLDIxxx member of the IMS PROCLIB data set (System Definition).

ODBM and ODBA

The Open Database Manager (ODBM) supports and uses the Open Database Access (ODBA) interface.

ODBM uses the ODBA interface to communicate with IMS. Many of the parameters in the ODBM CSLDCxxx configuration PROCLIB member are the same as those used by ODBA in the DFSxxxx0 startup table that is built from the DFSPRP macro parameters.

Because ODBM uses ODBA, ODBA application servers, such as Db2 for z/OS or WebSphere Application Server for z/OS can be configured to connect to IMS through ODBM instead of through ODBA. Connecting through ODBM prevents a U0113 abend from occurring if a DB2 stored procedure or WebSphere Application Server application program terminates unexpectedly during DL/I processing.

Configuring an ODBA application server to use ODBM does not require any changes to existing application programs that run under the ODBA application server.

Configuring ODBA application servers to use ODBM

You can configure an application server that uses the Open Database Access (ODBA) interface, such as Db2 for z/OS or WebSphere Application Server for z/OS, to connect to IMS through the Open Database Manager (ODBM) component of the IMS Common Service Layer (CSL).

Prerequisites:

To configure an application server that uses ODBA to connect to IMS through ODBM, you must configure the ODBA environment, as described in Accessing IMS databases through the ODBA interface (Communications and Connections).

ODBM also requires the CSL Structured Call Interface (SCI) and the CSL Operations Manager (OM).

Restriction: The ODBA application server must reside on the same z/OS LPAR as both ODBM and IMS.

To configure an ODBA application server to use ODBM:

1. On the DFSPRP macro that is used to build the database resource adapter (DRA) startup table DFSxxxx0 (where xxxx is the alias name of the IMS system that is specified on the APSB call), in addition to the ODBA connection attributes, specify the IMSPLEX parameter and, optionally, the ODBMNAME parameter.

When the ODBMNAME parameter is not specified, ODBA selects the ODBM instance that the CSL Structured Call Interface (SCI) uses to route database connection requests.

2. Optional: Configure the ODBA application server to obtain and manage a private context under which the ODBA calls are processed. If the ODBA application server does not obtain a private context, ODBA obtains and manages the private context on behalf of the ODBA application.

The following sample JCL sets up a DRA startup table with the name DFSSYS10. The use of ODBM is indicated by the specification of IMSPLEX=PLEX1 and ODBMNAME=ODBM1. The values of MINTHRD and MAXTHRD are the same to avoid repeatedly removing and re-creating the threads.

```
//DFSSYS10 JOB A,JIM,CLASS=Q,MSGLEVEL=(1,1),MSGCLASS=A,
// USER=username,PASSWORD=password,
// REGION=3072K
/*ROUTE PRINT THISCPU/CECTOOL
//*
//*--------------------------------------------------------------------------
//* Build ODBA PRP member *
//*--------------------------------------------------------------------------
//* SYS1, CRESLIB
//*********************************************************************
//* ******
//* ASSEMBLE AND LINK DFSSYS10 USING IMS MACROS
//* Test TIMER = 100 Error. We will see RC=08. ******
//*********************************************************************
//ASMLNK10 EXEC ASMLK11A,
// PARM.LINK='SIZE=(880K,64K),NCAL,LET,,XREF,LIST',
// LNKOUT='IMSTESTL.TNUC0'
//ASSEM.SYSIN DD * SUFFIX SYS1
DFSPRP DSECT=NO,FPBUF=10,FPBOF=5,CNBA=150,DBCTLID=IMS1,
MINTHRD=10,MAXTHRD=10,FUNCLV=2,IDRETRY=100,
IMSPLEX=PLEX1,ODBMNAME=ODBM1,SOD=A,TIMEOUT=2,
DSNAME=IMS.RESLIB,DDNAME=DFSIVD1
/*
//LINK.SYSIN DD *
NAME DFSSYS10(R)
/*
//*
```

Related concepts

CSL definition and tailoring (System Definition)

Related reference

DRA startup table (System Programming APIs)

ODBM message routing

When you define the IMS systems in an IMSplex to ODBM, you also assign one or more alias names to each IMS system. The alias names are what the ODBM clients and ODBM use to route database access requests to the IMS systems.

If an alias name is shared by multiple IMS systems in an IMSplex, ODBM uses a round-robin algorithm to distribute incoming requests that specify the shared alias name among the sharing IMS systems.

Similarly, when incoming requests leave the alias name blank, ODBM also uses a round-robin algorithm to distribute incoming requests among all IMS systems available to ODBM in the IMSplex.

ODBM and security

The CSL Open Database Manager (ODBM) does not perform user authentication or authorization itself.

If you are using IMS Connect with ODBM, you can authenticate the user IDs on request messages before they reach ODBM by using IMS Connect security. In addition to being able to call RACF directly, IMS Connect provides the IMS DB Security user exit routine (HWSAUTH0) to facilitate the customization of the security checking for communications with IMS DB.

If you use RACF to authenticate user IDs in IMS Connect, you can also enable RACF statistics to be captured when IMS Connect issues the **RACF RACROUTE REQUEST=VERIFY** call to authenticate ODBM client connections to IMS DB.

You can have IMS check the authority of a user to allocate a PSB or access IMS resources by using APSB and resource access control (RAS) security.

APSB security is enabled by specifying the ODBASE parameter. RAS security is specified by the ISIS parameter. Both the ODBASE and ISIS parameters can be specified on the IMS or DBC startup procedure or the DFSPBxxx PROCLIB member.

If a user-written ODBM client passes a security object for a security product such as RACF, ODBM invokes RACROUTE REQUEST=VERIFY to create an accessor environment element (ACEE) for the APSB thread. IMS can then use the ACEE during APSB or RAS authorization for allocating PSBs or access to other resources.

Related concepts

"IMS security" on page 285
This topic provides information to help you establish resource security for an IMS online system and identifies the resources that can be protected and the facilities available to protect them, provides design considerations, and describes the steps that you need to take to activate security.

Establishing and defining security (Communications and Connections)

Related tasks

Enabling RACF security statistics for IMS Connect (Communications and Connections)

Related reference

UPDATE IMSCON TYPE(CONFIG) command (Commands)
ODACCESS statement (System Definition)

ODBM accounting

The ODBM accounting feature performs logging of accounting information, such as CPU usage, for transactions processed in the ODBM address space, that you can use for charge back.

ODBM leverages the z/OS System Management Facility (SMF) to perform the logging of the ODBM accounting information and its retrieval.

Configuring the logstream for CSL ODBM accounting

The logging of the ODBM address space is activated when the optional parameter LOGOPT=ACCOUNTING is specified in the ODBM startup member, CSLDIxxx.

1. Specify LOGOPT=ACCOUNTING in the CSLDIxxx member of the IMS PROCLIB data set.
2. Start ODBM with the modified CSLDIxxx member
3. Process the z/OS System Management Facility (SMF) type-29, subtype-1 logs to account for ODBM workload and charge back data.

   The SMF parmlib member, SMFPRMxx in the SYS1.PARMLIB, defines the method and the target data set for the SMF logging.

   The two methods of logging are logstream and data set. Only one of these methods are used at a time. Here is an example of a logstream definition:

   ```
   DEFAULTLSNAME(IFASMF.ALLSYS.DEFAULT) LSNAME(IFASMF.ALLSYS.DATA,TYPE(29))
   ```

   Here is an example of a data set definition:

   ```
   DSNAME(SYSA.MAN1,SYSA.MAN2,SYSA.MAN3)
   ```

   The SMF logs can be retrieved by issuing the SMF utility IFASMFDL (for logstream) or IFASMFDP (for data set). The output of both of these utilities is a raw dump of the SMF logs.

The logged information includes all activity from a single unit of work (UOW) for the duration of the scheduled PSB.

Related information

Setting up and Managing SMF

SMF log records for ODBM accounting

The SMF record that is reserved for ODBM accounting is type 29 (x'1D') with subtype of 1. The SMF log record consists of three main sections: the SMF header, the BPE header, and the ODBM accounting information - all contained in a single record instance.

The BPESMF29 macro contains the DSECTs for both the SMF type 29 and the BPE headers. Field smf29bhs in the SMF header contains the offset from the start of the SMF 29 record to the BPE header.

The CSLDSMF macro contains the DSECT for the ODBM accounting information (record subtype 1). Field smf29sts in the SMF header contains the offset from the start of the SMF 29 record to the ODBM accounting information section.

These tables show the breakdown of each section.

Table 8. SMF header (type 29) – BPESMF29 DSECT

Name	Length	Type	Description
smf29len	2	binary	Record length
smf29seg	2	binary	Segment descriptor
smf29flg	1	binary	System indicator flag: • x'80' – Reserved • x'40' – Subtype used • x'20' – Reserved • x'10' – MVS/SP version 4 and above • x'08' – MVS/SP version 3 • x'04' – MVS/SP Version 2 • x'02' – VS2 • x'01' – Reserved
smf229rty	1	binary	Record type (29 decimal or x'1D')
smf29tme	4	binary	Time since midnight in hundredths of a second
smf29dte	4	packed	Date (packed decimal) when the record was moved into the SMF buffer
smf29sid	4	EBCDIC	System identification
smf29ssi	4	EBCDIC	Subsystem identification
smf29sty	2	binary	Record subtype
smf29trn	2	binary	Number of triplets in this record
*	2		Reserved
smf29bhs	4	binary	Offset to BPE header from start of record
smf29bhl	2	binary	Length of BPE header
smf29bhn	2	binary	Number of BPE header
smf29sts	4	binary	Offset to subtype specific section from start of record
smf29stl	2	binary	Length of subtype specific section
smf29stn	2	binary	Number of subtype specific section

Table 9. BPE header – BPESMF29_BPEHDR DSECT (located at offset smf29bhs from the start of the record)

Name	Length	Type	Description
smf29bh_fieldFlags	4	binary	Field flags
smf29bh_asType	4	EBCDIC	Address space type that wrote this record
smf29bh_jobName	8	EBCDIC	Job or started task name
smf29bh_asName	8	EBCDIC	1. IMSID (or DBCTL RSENAME) for batch/ IMS control region-associated address spaces (CTL, DLI, DBRC, DEP) 2. BPE address space “system name” for BPE-managed address spaces
smf29bh_crType	1	binary	Associated control region type: • x'00' – Not a control-region address space • x'01' – TM/DB IMS • x'02' – DBCTL IMS • x'03' – DCCTL IMS • x'04' – FDBR region
smf29bh_flag1	1	binary	Flag byte: • x'80' – IMS or BPE address space registered as VUE • x'40' – Record written by IMS dependent region or DBCTL/ODBA thread • x'20' – Record written by IMS batch region • x'10' – IMS using IRLM • x'08' – IMS using DBRC • x'04' – IMS using shared queues • x'02' – IMS using CSL • x'01' – Reserved
*	2		Reserved
smf29bh_asVersion	3	EBCDIC	Address space version number (binary)
smf29bh_bpeVersion	3	EBCDIC	BPE version number or 0 (binary)
smf29bh_asid	2	binary	ASID of the address space that wrote this record
*	4		Reserved
smf29bh_startStck	8	EBCDIC	1. STCK value of IMS batch or control region start for batch and control region-associated address spaces (CTL, DLI, DBRC, DEP) 2. STCK value at address space start for BPE-based address spaces
smf29bh_stck	8	EBCDIC	Current STCK when this record was built

Table 10. ODBM accounting information subtype 1 – CSLDSMF DSECT (located at offset smf29sts from the start of the record)

Name	Length	Type	Description
smf29sty1_pver	4	binary	Version number
smf29sty1_func	4	binary	Function number
smf29sty1_aliasName	4	EBCDIC	Alias name (if available)
smf29sty1_plexName	8	EBCDIC	IMSplex name
smf29sty1_psbName	8	EBCDIC	PSB name (if available)
smf29sty1_apsbToken	16	EBCDIC	APSB token **Note:** This token correlates to the IMS user-defined token in the x'08' record.
smf29sty1_apsbSTCK	8	EBCDIC	APSB STCK
smf29sty1_clientidlen	4	binary	Client ID length
smf29sty1_clientidoff	4	binary	Offset to client ID name from start of ODBM subtype record (if available)
smf29sty1_useridlen	4	binary	User ID length
smf29sty1_useridoff	4	binary	Offset to user ID name from start of ODBM subtype record (if available)
smf29sty1_racflen	4	binary	RACF ID length
smf29sty1_racfoff	4	binary	Offset to RACF ID name from start of ODBM subtype record (if available)
smf29sty1_UOWtime	8	EBCDIC	Total UOW time (in STCK units)
smf29sty1_zIIPtime	8	EBCDIC	Total zIIP time (in STCK units)
smf29sty1_numDLI	4	binary	Total number of DLI calls
smf29sty1_numSQL	4	binary	Total number of executable SQL commands

Chapter 9. CSL OM administration

This topic describes the administrative tasks associated with OM.

CSL OM command routing

In an IMSplex environment, commands issued to OM can behave differently than commands issued to a single IMS system. Commands that are issued to OM are, by default, routed to all IMS systems in the IMSplex that are active and registered to process that particular command.

If you want to route a command to one or more specific command processing clients in the IMSplex, use the ROUTE parameter on the CSLOMCMD request or the CSLOMI API. When an OM command has a ROUTE parameter, IMS chooses the highest level IMS in the route list as the command master. For example, in an IMSplex configuration that includes an IMS 15.2 system with an IMS 15.2 CQS, and another IMS 15.1 system with an IMS 15.1, if an INIT OLC command (which is a ROUTE=ANY command) is issued, the IMS 15.2 system is selected as the command master.

You can also specify routing information with the TSO SPOC. For information about how to specify the ROUTE parameter, see *IMS Version 15.2 System Programming APIs*. Refer also to the online help provided with the TSO SPOC.

In an IMSplex, a command originates from a SPOC or automated operator program (AOP) and is sent to OM. It is then routed to IMS systems; each IMS system returns its command output. OM then consolidates the individual responses and sends consolidated output, encapsulated in XML tags, back to the client originating the request. The following figure shows this routing. OS1 has an SCI, a SPOC TSO/ISPF application, and an automation program. OS2 has an OM with an SCI, and IMS control region with an SCI. OS3 has two IMS control regions, each with an SCI. The command is routed from OS1's SCI to OM in OS2. The SCI that is associated with that OM routes the command to other IMS systems through those system's SCI.

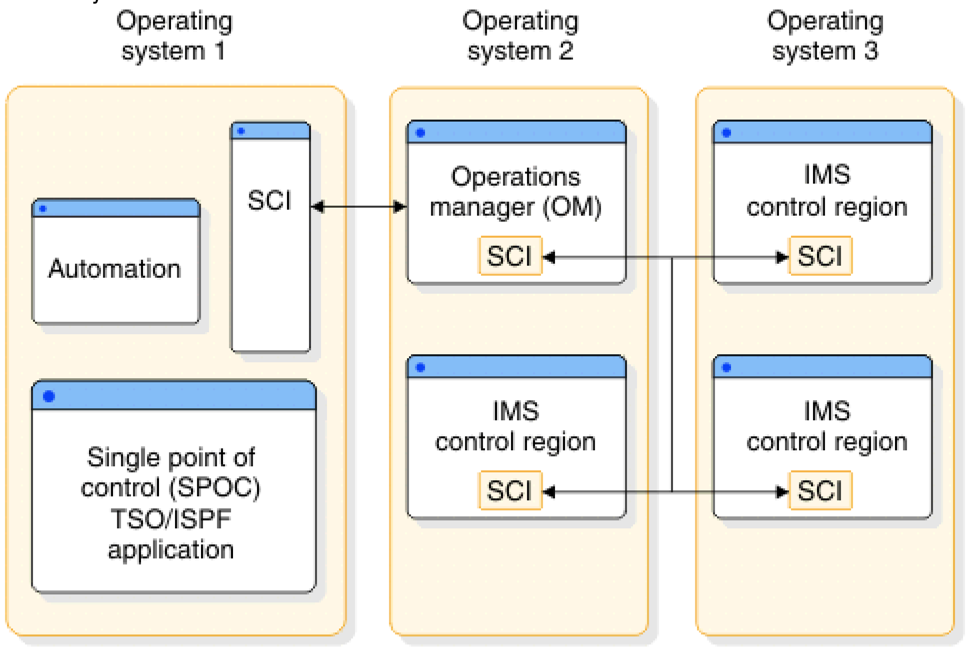


Figure 27. Command routing in an IMSplex with CSL

When commands are issued to an OM, command responses are encapsulated in XML tags.

For information about IMS commands and their responses, see *IMS Version 15.2 Commands, Volume 1: IMS Commands A-M* and *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Related concepts

Interpreting CSL OM XML output (System Programming APIs)

CSL OM automated operator program clients (System Programming APIs)

Related tasks

Registering an OM command processing client (System Programming APIs)

Related reference

Writing a CSL OM client (System Programming APIs)

Issuing commands through the OM API

You can enter type-1 commands from an IMS terminal, such as the MTO terminal or an end-user terminal. You can enter many type-1 commands and all type-2 commands through the SPOC and OM API. When you enter a command through the OM API, OM can route the command to all IMS systems that can process the command. OM consolidates the command responses from the IMS systems into one buffer as they are received.

If the command has local scope only and is entered from the IMS Master Terminal, IMS end user terminal, or the IMS automated operator program (AOP), then the command is processed only by the local IMS where the command was entered. However, if a resource structure is part of the IMSplex, some commands, such as **`/STOP USER`** and **`/STOP NODE`**, have a global status in RM. If you enter those commands from a SPOC, they might be routed to all of the IMS systems in the IMSplex and have a global effect. The TSO SPOC allows you to route commands to individual IMS systems.

CSL provides the type-2 command format. The benefits of this format are listed below:

- Elimination of resource name and command keyword conflicts
- Better command syntax checking using the BPE parser
- Simplified set of command verbs
- Parallel command processing
- Filters and wild cards for resource name selection
- Filtering and displaying of only selected output fields for **QUERY** command output
- Only one IMS with access to global information

Commands entered using this command format can only be entered through a SPOC. As a result, the command format cannot be entered through the system console, master terminal, end user terminal, DL/I call, APPC client, or through the OTMA interface. The format is based on simplified BPE parse rules. These commands work with the single-system image that an IMSplex provides of IMS systems sharing databases and queues in the IMSplex.

The type-2 command format is as follows. The keywords and parameters are optional:

```
ActionVerb ResourceType Keyword(parameter) Keyword(parameter) . . .
```

CSL OM audit trail

An OM audit trail reflects the input and output of commands that are processed through the OM API. You can log: a point of command origin, what a particular command processed, and the result of that processing.

OM can write log records to a z/OS System Logger log stream, creating an audit trail of command input, associated command output, and unsolicited message output. Before using the OM audit trail function, the log stream needs to be created. Definitions for z/OS System Logger and coupling facility structures are required. To write OM log records to a z/OS log stream, specify the AUDITLOG= parameter of the IMSPLEX() keyword on the CSLOIxxx PROCLIB member.

In the following scenarios, no data is lost in the OM audit trail when the log stream becomes available because OM logs its work:

- If the connection to the log stream fails at startup, the initialization continues. Connection is set up after the log stream becomes available.
- If writing to the log stream fails for OM, OM waits for the log stream to become available and performs the writing process again.

If an individual command or command response plus its associated control data is longer than 32,760 bytes, it is split across multiple log records. These log records include control data that log formatting utilities can use to reassemble the log records into a single command or command response.

Related concepts

Other sample applications verified by the IVP (Installation)

OM audit trail (Operations and Automation)

Related tasks

"OM audit log record formats" on page 140
Each Operations Manager (OM) audit log record contains a log record prefix, followed by data that is unique to the record.

Configuring the logstream for the CSL OM audit trail

The IMS IVP defines a logstream with parameters set to apply generally to most users. Depending on the configuration and requirements of your installation, you might consider changing these parameters when you define an OM audit trail logstream.

Parameters set by the IVP to consider changing include:

LS_SIZE(200)
LS_SIZE controls the size of the offload data sets. Your installation might benefit from larger data sets.

LOWOFFLOAD(20)
LOWOFFLOAD affects the frequency of offloads. A lower LOWOFFLOAD percentage, such as 0, generally means that offloads occur less frequently and, unless the logstream is browsed in realtime, should be the most efficient.

HIGHOFFLOAD(50)
HIGHOFFLOAD affects the frequency of offloads. A higher HIGHOFFLOAD value, such as 70, generally means that offloads occur less frequently, which improves efficiency.

The IMS IVP also does not define the system logger to duplex to staging data sets. If your installation requires a higher level of data redundancy than is provided with MVS dataspace duplexing then consider enabling external duplexing by adding STG_DUPLEX(YES) to the definition below, along with the DUPLEXMODE parameter, which defines the conditions in which the duplexing occurs. However, the higher level of data redundancy must be weighed against the performance impact of the duplexing option.

The IMS IVP defines a logstream with the following parameters:

```
DEFINE STRUCTURE NAME(IMSOM2Q01)
         LOGSNUM(1)
         AVGBUFSIZE(4000)
         MAXBUFSIZE(32760)

DEFINE LOGSTREAM NAME(SYSLOG.OM2Q01.LOG)
         STRUCTNAME(IMSOM2Q01)
         HLQ(IXGLOGR)
         LS_STORCLAS(LOGGER1)
         LS_DATACLAS(LOGGER1)
         LS_MGMTCLAS(LOGGER)
         LS_SIZE(200)
         LOWOFFLOAD(20)
         HIGHOFFLOAD(50)
```

Recommendations:

- Set the logstream maximum buffer size (MAXBUFSIZE) to 32,760 bytes. This setting ensures that all command and command response data is written to the audit trail. A lower setting will result in the truncation of some OM Audit Trail log records.

- Dedicate a log stream to the OM audit trail.

Related concepts

IBM Redbooks: System Logger

z/OS: Using system logger services

Related tasks

z/OS: Planning for system logger applications

Related reference

CSLOIxxx member of the IMS PROCLIB data set (System Definition)

Related information

OM audit log record formats

Each Operations Manager (OM) audit log record contains a log record prefix, followed by data that is unique to the record.

Use macro CSLZLGPF to map the log record prefix. To view the OM audit log record formats, assemble mapping macro CSLOLGRC.

Individual log record DSECTs can be obtained by assembling the ILOGREG macro while including a format statement for the desired log record.

The following table shows the OM audit log records. For each OM audit log record, the table shows:

- The log record type and subtype
- The macro that maps the record
- The event that causes the record to be written

Table 11. OM audit log records

Type	Subtype	Mapping macro	Event that causes the log record to be written
X'06'	X'01'	CSLOLGCM	The command input was logged before calling the OM Input user exit routine.
X'06'	X'02'	CSLOLGCM	The command input was logged after calling the OM Input user exit routine and after that user exit routine modified the command input.
X'08'	X'01'	CSLOLGCR	The command response output was logged before calling the OM Output user exit routine.
X'08'	X'02'	CSLOLGCR	The command response output was logged after calling the OM Output user exit routine and after that user exit routine modified the command response output.
X'09'	X'01'	CSLOLGOU	The unsolicited output message was logged before calling the OM Output user exit routine.
X'09'	X'02'	CSLOLGOU	The unsolicited output message was logged after calling the OM Output user routine, and after that user exit routine modified the unsolicited output message.

Printing OM log records

To print the OM log records from the z/OS system log, use the IMS File Select and Formatting Print utility (DFSERA10) with exit routine CSLOERA3.

The example in the following figure shows the JCL that is required to print the log records from a z/OS system log.

JCL to print OM log records from a z/OS system log

```
//CSLERA1   JOB   MSGLEVEL=1,MSGCLASS=A,CLASS=K
//STEP1     EXEC  PGM=DFSERA10
//STEPLIB   DD    DISP=SHR,DSN=IMS.SDFSRESL
//SYSPRINT  DD    SYSOUT=A
//SYSUT1    DD    DSN=SYSLOG.OM.AUDIT.TRAIL.LOG,
//                SUBSYS=(LOGR,IXGSEXIT),
//                DCB=(BLKSIZE=32760)
//SYSIN     DD
CONTROL    CNTL   H=EOF
OPTION     PRINT  EXITR=CSLULALE
END
//
```

Command input and command response log records longer than 32 760 bytes are divided into two or more segments. Unsolicited output has a maximum length of 320 bytes. Each segment is sent to the MVS logger on a separate IXGWRITE call from OM. Therefore, long log records appear as multiple records on the log stream. Additional control data is added to the command response and command input log records to enable the CSLOERA3 and CSLULALE formatting utilities to reassemble log records that have been separated into multiple segments. A bit is added to the log record prefix to indicate single segment log records, including log records other than command input or command response. Log records are printed in the same order that they were sent to the log stream, based on the timestamp provided in the log record. If MAXBUFSIZE is set lower than 32 760 bytes, log records will not be broken into multiple segments. Instead, any log record that exceeds the length in MAXBUFSIZE will be truncated.

Recommendation: Set the MAXBUFSIZE to 32760 bytes. Log records requiring more than 65,535 segments will continue to be truncated. Currently, this limitation affects only a log record longer than 2GB.

The DD statements for printing OM log records are:

STEPLIB
DSN= points to IMS.SDFSRESL, which contains the IMS File Select and Formatting Print utility, DFSERA10.

SYSUT1
DSN= points to the OM log stream name that is specified on the AUDITLOG= parameter in the CSLOIxxx PROCLIB member.

The control statements for printing the OM log records are:

H=
Specifies the number of log records to print. To print all of the log records, specify H=EOF.

EXITR=CSLULALE
Identifies the OM log record exit routine that is called to format each log record. Exit routine CSLULALE formats the OM audit trail in a format similar to a syslog. Exit routine CSLOERA3 can also be used. Exit routine CSLOERA3 prints the records in a dump format, including the record type and time-stamp information for each record, and provides the contents of the record up to a maximum of 32 760 bytes.

To limit the log data to a specific time range, use the FROM and TO parameters on the SUBSYS statement, as shown in the figure below. This DD card prints log records from 11:00 to 12:00 on day 42 of the year 2007.

JCL to limit log data to a specific time range

```
//SYSUT1    DD    DSN=SYSLOG.OM.AUDIT.TRAIL.LOG,
//                SUBSYS=(LOGR,IXGSEXIT,
//                'FROM=(2007/042,11:00:00),TO=(2007/042,12:00:00)'),
//                DCB=(BLKSIZE=32760)
```

Dates and times are specified in Greenwich Mean Time (GMT). The seconds field in the time value is optional. To use local dates and times, add the LOCAL keyword, as shown in the following table.

JCL to limit log data to a specific time range, using local dates and times

```
//SYSUT1    DD    DSN=SYSLOG.OM.AUDIT.TRAIL.LOG,
//                SUBSYS=(LOGR,IXGSEXIT,
//                'FROM=(2007/042,11:00:00),TO=(2007/042,12:00:00),LOCAL'),
//                DCB=(BLKSIZE=32760)
```

Related reference

File Select and Formatting Print utility (DFSERA10) (System Utilities)

CSL OM command security

OM command security is optionally performed during command processing.

Command security allows:

- The user to control which user IDs can enter IMS commands through OM
- The user ID to be associated with an application program address space
- The user ID to be the end user logged onto TSO SPOC

The CMDSEC= parameter is available on the OM startup procedure (CSLOM), the OM initialization PROCLIB member data set (CSLOIxxx), and the DFSCGxxx PROCLIB member data set. When it is issued as part of the OM startup procedure, it applies to all IMS commands, type-1 and type-2. When it is issued using the DFSCGxxx PROCLIB member data set, it applies only to type-1 commands entered through OM. The differences in OM and IMS security are described in the following table.

Table 12. Comparing OM and IMS security

Security method	N	A	E	R
OM Execution Parameter (CSLOM and CSLOIxxx)	No authorization checking is performed. This is the default.	Calls both RACF and the CSL OM Security user exit routine for command authorization.	Calls the CSL OM Security user exit routine for command authorization.	Calls RACF for command authorization. Commands are part of the OPERCMDS resource class.
DFSCGxxx PROCLIB member data set	No authorization checking is performed. This is the default. OM might perform command authorization.	Calls both RACF and the IMS Command Authorization Exit routine (DFSCCMD0) for command authorization.	Calls the IMS Command Authorization Exit routine (DFSCCMD0) for command authorization.	Calls RACF for command authorization. Commands are part of the CIMS resource class.

Recommendation: Use OM command security rather than IMS command security.

RACF access authorities (READ or UPDATE) and resource names for all commands supported through the OM API are described in "IMS commands, RACF access authorities and resource names table" on page 144. The RACF authorities indicate the access authority with which the command was registered.

Commands are registered to OM with the CSLOMBLD request. The access authority on the RACF PERMIT command must match the access authority with which the command was registered. For more information on registering commands with CSLOMBLD, see *IMS Version 15.2 System Programming APIs*.

RACF authorization for commands entered through OM

All OM security checking uses the RACF OPERCMDS class. The resource name is in the form of: IMS.*imsplexname*.*commandverb*.*commandkeyword*

IMS
The high-level qualifier for all RACF resources that protect IMS commands entered through OM.

imsplexname
The name of the IMSplex for which the command is authorized. The IMSplex name must include the characters CSL at the beginning. For example, CSLplex1.

commandverb
The name of the command verb.

commandkeyword
The primary command keyword or resource type for the command.

Example: authorizing users to commands in an IMSplex

The following example shows a RACF definition that authorizes various users to different commands.

```
/*   allow UPD TRAN command on any IMSplex for user ID Jim */
     RDEFINE OPERCMDS IMS.*.UPD.TRAN UACC(NONE)
     PERMIT IMS.*.UPD.TRAN CLASS(OPERCMDS) ID(JIM) ACCESS(UPDATE)

/*   allow INIT OLC command on IMSplex CSLPLEXA for user ID Sandy */
     RDEFINE OPERCMDS IMS.CSLPLEXA.INIT.OLC UACC(NONE)
     PERMIT IMS.CSLPLEXA.INIT.OLC CLASS(OPERCMDS) ID(SANDY) ACCESS(UPDATE)

/*   allow any QRY command on any IMSplex for user ID Tom */
     RDEFINE OPERCMDS IMS.*.QRY.* UACC(NONE)
     PERMIT IMS.*.QRY.* CLASS(OPERCMDS) ID(TOM) ACCESS(READ)

/*   allow any commands on any IMSplex for user ID Betty */
     RDEFINE OPERCMDS IMS.* UACC(NONE)
     PERMIT IMS.* CLASS(OPERCMDS) ID(BETTY) ACCESS(UPDATE)

     SETROPTS CLASSACT(OPERCMDS)
     SETROPTS RACLIST(OPERCMDS) REFRESH
```

Example: using RACF groups and the OPERCMDS class for type-2 command security

The following example shows a RACF definition that authorizes all user IDs that are part of a RACF group to use any IMS operator commands. This example shows two important elements of RACF administration:

- RACF groups
- Wildcard characters

```
/* define IMS operator group */
    ADDGROUP IMSOPER
    CONNECT (PEDRO,ROSA,PETER,MATT) GROUP(IMSOPER)

/* allow any commands on any IMSplex for group IMSOPER */
    RDEFINE OPERCMDS IMS.* UACC(NONE)
    PERMIT IMS.* CLASS(OPERCMDS) ID(IMSOPER) ACCESS(UPDATE)

    SETROPTS CLASSACT(OPERCMDS)
    SETROPTS RACLIST(OPERCMDS) REFRESH
```

Using a group definition simplifies the administration of RACF profiles. The group can be permitted to numerous RACF profiles, such as data sets and OPERCMDS profiles. If a user changes jobs, you need to remove the user ID only from the group definition instead of deleting its access to each RACF resource profile.

The benefit of using wildcard characters in the RACF resource name is that you can keep the number of RACF resource definitions to a minimum. Also, any matching operator commands that are defined after the OPERCMDS profiles are created are still protected from unspecified users.

Example: authorizing users who are associated with the RACF APPL class

The following example shows a RACF definition that adds a profile for authorizing users who are associated with the RACF APPL class.

```
/* add a profile */
    RDEFINE APPL JOBNAME UACC(NONE)
    PERMIT JOBNAME CLASS(APPL) ID(USERID OR GROUPNAME) ACCESS(UPDATE)
```

IMS commands, RACF access authorities and resource names table

The following table applies to the OM command security for IMS type-1 and type-2 command security. The table lists by IMS command verb and keyword the resource name and authorization that are used for RACF security checking.

For example, if you want to issue the **ACTIVATE NODE** command on an IMSplex named PLX01, your user ID must have UPDATE access to profile IMS.CSLPLX01.ACT.NODE in the OPERCMDS class. If you have only READ authority, the **ACTIVATE NODE** command will fail with an "insufficient authority" message.

In general, you can issue a **DISPLAY** or **QUERY** command with READ authority. To change the state of a resource, you need UPDATE authority.

Tip: You can reduce the number of RACF resources that are required by using wildcards in the resource name. For example, use IMS.CSLPLX01.DIS.* to grant authority to use all **DISPLAY** commands.

Table 13. Resource names and RACF authority for IMS commands

Command verb	Command keyword	Authority	Resource name
ACT	LINK	UPDATE	IMS.plxname.ACT.LINK
ACT	NODE	UPDATE	IMS.plxname.ACT.NODE
ALL	LU	UPDATE	IMS.plxname.ALL.LU
ASS	CLASS	UPDATE	IMS.plxname.ASS.CLASS
ASS	CPRI	UPDATE	IMS.plxname.ASS.CPRI
ASS	INPUT	UPDATE	IMS.plxname.ASS.INPUT
ASS	LCT	UPDATE	IMS.plxname.ASS.LCT
ASS	LPRI	UPDATE	IMS.plxname.ASS.LPRI
ASS	LTERM	UPDATE	IMS.plxname.ASS.LTERM
ASS	NPRI	UPDATE	IMS.plxname.ASS.NPRI
ASS	OUTPUT	UPDATE	IMS.plxname.ASS.OUTPUT
ASS	PARLIM	UPDATE	IMS.plxname.ASS.PARLIM
ASS	PLCT	UPDATE	IMS.plxname.ASS.PLCT
ASS	SEGNO	UPDATE	IMS.plxname.ASS.SEGNO
ASS	SEGSZ	UPDATE	IMS.plxname.ASS.SEGSZ
ASS	TRAN	UPDATE	IMS.plxname.ASS.TRAN
ASS	USER	UPDATE	IMS.plxname.ASS.USER
BRO	ACT	READ	IMS.plxname.BRO.ACT
BRO	LINE	READ	IMS.plxname.BRO.LINE
BRO	LTERM	READ	IMS.plxname.BRO.LTERM

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
BRO	MASTER	READ	IMS.plxname.BRO.MASTER
BRO	NODE	READ	IMS.plxname.BRO.NODE
BRO	PTERM	READ	IMS.plxname.BRO.PTERM
BRO	USER	READ	IMS.plxname.BRO.USER
CHA	APPC	UPDATE	IMS.plxname.CHA.APPC
CHA	CCTL	UPDATE	IMS.plxname.CHA.CCTL
CHA	CPLOG	UPDATE	IMS.plxname.CHA.CPLOG
CHA	DESC	UPDATE	IMS.plxname.CHA.DESC
CHA	DIR	UPDATE	IMS.plxname.CHA.DIR
CHA	FDR	UPDATE	IMS.plxname.CHA.FDR
CHA	LINK	UPDATE	IMS.plxname.CHA.LINK
CHA	NODE	UPDATE	IMS.plxname.CHA.NODE
CHA	PSWD	UPDATE	IMS.plxname.CHA.PSWD
CHA	SUBSYS	UPDATE	IMS.plxname.CHA.SUBSYS
CHA	SURV	UPDATE	IMS.plxname.CHA.SURV
CHA	TRAN	UPDATE	IMS.plxname.CHA.TRAN
CHA	UOR	UPDATE	IMS.plxname.CHA.UOR
CHA	USER	UPDATE	IMS.plxname.CHA.USER
CHE		UPDATE	IMS.plxname.CHE
CHE	DUMPQ	UPDATE	IMS.plxname.CHE.DUMPQ
CHE	FREEZE	UPDATE	IMS.plxname.CHE.FREEZE
CHE	PURGE	UPDATE	IMS.plxname.CHE.PURGE
CHE	STATISTICS	UPDATE	IMS.plxname.CHE.STATISTICS
CLS	NODE	UPDATE	IMS.plxname.CLS.NODE
CQC	SHRQ	UPDATE	IMS.plxname.CQC.SHRQ
CQC	SYSTEM	UPDATE	IMS.plxname.CQC.SYSTEM
CQQ	STATISTICS	READ	IMS.plxname.CQQ.STATISTICS
CQS	SHUTDOWN	UPDATE	IMS.plxname.CQS.SHUTDOWN
CRE	DB	UPDATE	IMS.plxname.CRE.DB
CRE	DBDESC	UPDATE	IMS.plxname.CRE.DBDESC
CRE	IMSCON	UPDATE	IMS.plxname.CRE.IMSCON
CRE	LTERM	UPDATE	IMS.plxname.CRE.LTERM
CRE	MSLINK	UPDATE	IMS.plxname.CRE.MSLINK
CRE	MSNAME	UPDATE	IMS.plxname.CRE.MSNAME

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
CRE	MSPLINK	UPDATE	IMS.plxname.CRE.MSPLINK
CRE	PGM	UPDATE	IMS.plxname.CRE.PGM
CRE	PGMDESC	UPDATE	IMS.plxname.CRE.PGMDESC
CRE	RTC	UPDATE	IMS.plxname.CRE.RTC
CRE	RTCDESC	UPDATE	IMS.plxname.CRE.RTCDESC
CRE	TRAN	UPDATE	IMS.plxname.CRE.TRAN
CRE	TRANDESC	UPDATE	IMS.plxname.CRE.TRANDESC
DBD	DB	UPDATE	IMS.plxname.DBD.DB
DBR	AREA	UPDATE	IMS.plxname.DBR.AREA
DBR	DB	UPDATE	IMS.plxname.DBR.DB
DBR	DATAGRP	UPDATE	IMS.plxname.DBR.DATAGRP
DEL	DB	UPDATE	IMS.plxname.DEL.DB
DEL	DBDESC	UPDATE	IMS.plxname.DEL.DBDESC
DEL	DEFN	UPDATE	IMS.plxname.DEL.DEFN
DEL	IMSCON	UPDATE	IMS.plxname.DEL.IMSCON
DEL	LE	UPDATE	IMS.plxname.DEL.LE
DEL	LTERM	UPDATE	IMS.plxname.DEL.LTERM
DEL	MSLINK	UPDATE	IMS.plxname.DEL.MSLINK
DEL	MSNAME	UPDATE	IMS.plxname.DEL.MSNAME
DEL	MSPLINK	UPDATE	IMS.plxname.DEL.MSPLINK
DEL	PGM	UPDATE	IMS.plxname.DEL.PGM
DEL	PGMDESC	UPDATE	IMS.plxname.DEL.PGMDESC
DEL	PSWD	UPDATE	IMS.plxname.DEL.PSWD
DEL	RTC	UPDATE	IMS.plxname.DEL.RTC
DEL	RTCDESC	UPDATE	IMS.plxname.DEL.RTCDESC
DEL	TERMINAL	UPDATE	IMS.plxname.DEL.TERMINAL
DEL	TRAN	UPDATE	IMS.plxname.DEL.TRAN
DEL	TRANDESC	UPDATE	IMS.plxname.DEL.TRANDESC
DEQ	AOITKN	UPDATE	IMS.plxname.DEQ.AOITKN
DEQ	LINE	UPDATE	IMS.plxname.DEQ.LINE
DEQ	LTERM	UPDATE	IMS.plxname.DEQ.LTERM
DEQ	LU	UPDATE	IMS.plxname.DEQ.LU
DEQ	MSNAME	UPDATE	IMS.plxname.DEQ.MSNAME
DEQ	NODE	UPDATE	IMS.plxname.DEQ.NODE

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
DEQ	SUSPEND	UPDATE	IMS.plxname.DEQ.SUSPEND
DEQ	TMEM	UPDATE	IMS.plxname.DEQ.TMEM
DEQ	TRAN	UPDATE	IMS.plxname.DEQ.TRAN
DEQ	USER	UPDATE	IMS.plxname.DEQ.USER
DIAG	ADDRESS	UPDATE	IMS.plxname.DIAG.ADDRESS
DIAG	BLOCK	UPDATE	IMS.plxname.DIAG.BLOCK
DIAG	LTERM	UPDATE	IMS.plxname.DIAG.LTERM
DIAG	NODE	UPDATE	IMS.plxname.DIAG.NODE
DIAG	SNAP	UPDATE	IMS.plxname.DIAG.SNAP
DIAG	TRAN	UPDATE	IMS.plxname.DIAG.TRAN
DIAG	USER	UPDATE	IMS.plxname.DIAG.USER
DIS	ACT	READ	IMS.plxname.DIS.ACT
DIS	AFFIN	READ	IMS.plxname.DIS.AFFIN
DIS	AOITKN	READ	IMS.plxname.DIS.AOITKN
DIS	APPC	READ	IMS.plxname.DIS.APPC
DIS	AREA	READ	IMS.plxname.DIS.AREA
DIS	ASMT	READ	IMS.plxname.DIS.ASMT
DIS	CCTL	READ	IMS.plxname.DIS.CCTL
DIS	CONV	READ	IMS.plxname.DIS.CONV
DIS	CPLOG	READ	IMS.plxname.DIS.CPLOG
DIS	CQS	READ	IMS.plxname.DIS.CQS
DIS	DB	READ	IMS.plxname.DIS.DB
DIS	DBD	READ	IMS.plxname.DIS.DBD
DIS	DESC	READ	IMS.plxname.DIS.DESC
DIS	FDR	READ	IMS.plxname.DIS.FDR
DIS	FPV	READ	IMS.plxname.DIS.FPV
DIS	HSB	READ	IMS.plxname.DIS.HSB
DIS	HSSP	READ	IMS.plxname.DIS.HSSP
DIS	LINE	READ	IMS.plxname.DIS.LINE
DIS	LINK	READ	IMS.plxname.DIS.LINK
DIS	LTERM	READ	IMS.plxname.DIS.LTERM
DIS	LU	READ	IMS.plxname.DIS.LU
DIS	MASTER	READ	IMS.plxname.DIS.MASTER
DIS	MODIFY	READ	IMS.plxname.DIS.MODIFY

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
DIS	MSNAME	READ	IMS.plxname.DIS.MSNAME
DIS	NODE	READ	IMS.plxname.DIS.NODE
DIS	OASN	READ	IMS.plxname.DIS.OASN
DIS	OLDS	READ	IMS.plxname.DIS.OLDS
DIS	OTMA	READ	IMS.plxname.DIS.OTMA
DIS	OVERFLOWQ	READ	IMS.plxname.DIS.OVERFLOWQ
DIS	POOL	READ	IMS.plxname.DIS.POOL
DIS	PGM	READ	IMS.plxname.DIS.PGM
DIS	PSB	READ	IMS.plxname.DIS.PSB
DIS	PTERM	READ	IMS.plxname.DIS.PTERM
DIS	Q	READ	IMS.plxname.DIS.Q
DIS	QCNT	READ	IMS.plxname.DIS.QCNT
DIS	RECOVERY	READ	IMS.plxname.DIS.RECOVERY
DIS	RTC	READ	IMS.plxname.DIS.RTC
DIS	SHUTDOWN	READ	IMS.plxname.DIS.SHUTDOWN
DIS	STATUS	READ	IMS.plxname.DIS.STATUS
DIS	STRUC	READ	IMS.plxname.DIS.STRUC
DIS	SUBSYS	READ	IMS.plxname.DIS.SUBSYS
DIS	SYSID	READ	IMS.plxname.DIS.SYSID
DIS	TIMEOVER	READ	IMS.plxname.DIS.TIMEOVER
DIS	TMEM	READ	IMS.plxname.DIS.TMEM
DIS	TRACE	READ	IMS.plxname.DIS.TRACE
DIS	TRACKING	READ	IMS.plxname.DIS.TRACKING
DIS	TRAN	READ	IMS.plxname.DIS.TRAN
DIS	UOR	READ	IMS.plxname.DIS.UOR
DIS	USER	READ	IMS.plxname.DIS.USER
END	LINE	UPDATE	IMS.plxname.END.LINE
END	NODE	UPDATE	IMS.plxname.END.NODE
END	USER	UPDATE	IMS.plxname.END.USER
ERE		UPDATE	IMS.plxname.ERE
ERE	BACKUP	UPDATE	IMS.plxname.ERE.BACKUP
ERE	COLDBASE	UPDATE	IMS.plxname.ERE.COLDBASE
ERE	COLDCOMM	UPDATE	IMS.plxname.ERE.COLDCOMM
ERE	COLDSYS	UPDATE	IMS.plxname.ERE.COLDSYS

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
EXC	LINE	UPDATE	IMS.plxname.EXC.LINE
EXC	NODE	UPDATE	IMS.plxname.EXC.NODE
EXC	USER	UPDATE	IMS.plxname.EXC.USER
EXI	CONV	UPDATE	IMS.plxname.EXI.CONV
EXP	DEFN	UPDATE	IMS.plxname.EXP.DEFN
IDL	LINE	UPDATE	IMS.plxname.IDL.LINE
IDL	LINK	UPDATE	IMS.plxname.IDL.LINK
IDL	NODE	UPDATE	IMS.plxname.IDL.NODE
IMP	DEFN	UPDATE	IMS.plxname.IMP.DEFN
INIT	OLC	UPDATE	IMS.plxname.INIT.OLC
INIT	OLREORG	UPDATE	IMS.plxname.INIT.OLREORG
LOC	DB	UPDATE	IMS.plxname.LOC.DB
LOC	PGM	UPDATE	IMS.plxname.LOC.PGM
LOC	TRAN	UPDATE	IMS.plxname.LOC.TRAN
LOG		UPDATE	IMS.plxname.LOG
MOD	ABORT	UPDATE	IMS.plxname.MOD.ABORT
MOD	COMMIT	UPDATE	IMS.plxname.MOD.COMMIT
MOD	PREPARE	UPDATE	IMS.plxname.MOD.PREPARE
MON	LINE	UPDATE	IMS.plxname.MON.LINE
MSA	LINK	UPDATE	IMS.plxname.MSA.LINK
MSA	MSNAME	UPDATE	IMS.plxname.MSA.MSNAME
MSA	SYSID	UPDATE	IMS.plxname.MSA.SYSID
MSA	TRAN	UPDATE	IMS.plxname.MSA.TRAN
MSV	MSNAME	UPDATE	IMS.plxname.MSV.MSNAME
MSV	SYSID	UPDATE	IMS.plxname.MSV.SYSID
NRE		UPDATE	IMS.plxname.NRE
NRE	CHKPT	UPDATE	IMS.plxname.NRE.CHKPT
OPN	NODE	UPDATE	IMS.plxname.OPN.NODE
PST	LINE	UPDATE	IMS.plxname.PST.LINE
PST	LINK	UPDATE	IMS.plxname.PST.LINK
PST	LTERM	UPDATE	IMS.plxname.PST.LTERM
PST	MSPLINK	UPDATE	IMS.plxname.PST.MSPLINK
PST	REGION	UPDATE	IMS.plxname.PST.REGION
PST	TRAN	UPDATE	IMS.plxname.PST.TRAN

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
PUR	APPC	UPDATE	IMS.plxname.PUR.APPC
PUR	FPPROG	UPDATE	IMS.plxname.PUR.FPPROG
PUR	FPRGN	UPDATE	IMS.plxname.PUR.FPRGN
PUR	LINE	UPDATE	IMS.plxname.PUR.LINE
PUR	LTERM	UPDATE	IMS.plxname.PUR.LTERM
PUR	MSNAME	UPDATE	IMS.plxname.PUR.MSNAME
PUR	TRAN	UPDATE	IMS.plxname.PUR.TRAN
QRY	AREA	READ	IMS.plxname.QRY.AREA
QRY	DB	READ	IMS.plxname.QRY.DB
QRY	DBDESC	READ	IMS.plxname.QRY.DBDESC
QRY	IMS	READ	IMS.plxname.QRY.IMS
QRY	IMSFUNC	QUERY	IMS.plxname.QRY.IMSFUNC
QRY	IMSCON	READ	IMS.plxname.QRY.IMSCON
QRY	IMSPLEX	READ	IMS.plxname.QRY.IMSPLEX
QRY	LE	READ	IMS.plxname.QRY.LE
QRY	LTERM	READ	IMS.plxname.QRY.LTERM
QRY	MEMBER	READ	IMS.plxname.QRY.MEMBER
QRY	MSLINK	READ	IMS.plxname.QRY.MSLINK
QRY	MSNAME	READ	IMS.plxname.QRY.MSNAME
QRY	MSLINK	READ	IMS.plxname.QRY.MSPLINK
QRY	NODE	READ	IMS.plxname.QRY.NODE
QRY	OLC	READ	IMS.plxname.QRY.OLC
QRY	OLREORG	READ	IMS.plxname.QRY.OLREORG
QRY	PGM	READ	IMS.plxname.QRY.PGM
QRY	PGMDESC	READ	IMS.plxname.QRY.PGMDESC
QRY	POOL	READ	IMS.plxname.QRY.POOL
QRY	RM	READ	IMS.plxname.QRY.RM
QRY	RTC	READ	IMS.plxname.QRY.RTC
QRY	RTCDESC	READ	IMS.plxname.QRY.RTCDESC
QRY	STRUCTURE	READ	IMS.plxname.QRY.STRUCTURE
QRY	TRACE	READ	IMS.plxname.QRY.TRACE
QRY	TRAN	READ	IMS.plxname.QRY.TRAN
QRY	TRANDESC	READ	IMS.plxname.QRY.TRANDESC
QRY	USER	READ	IMS.plxname.QRY.USER

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
QRY	USEREXIT	READ	IMS.plxname.QRY.USEREXIT
QRY	USERID	READ	IMS.plxname.QRY.USERID
QUE	LTERM	UPDATE	IMS.plxname.QUE.LTERM
QUE	TRAN	UPDATE	IMS.plxname.QUE.TRAN
QUI	NODE	UPDATE	IMS.plxname.QUI.NODE
RDI	MASTER	READ	IMS.plxname.RDI.MASTER
REC	ADD	UPDATE	IMS.plxname.REC.ADD
REC	REMOVE	UPDATE	IMS.plxname.REC.REMOVE
REC	START	UPDATE	IMS.plxname.REC.START
REC	STOP	UPDATE	IMS.plxname.REC.STOP
REC	TERMINATE	UPDATE	IMS.plxname.REC.TERMINATE
REFRESH	USEREXIT	UPDATE	IMS.plxname.REFR.USEREXIT
RMC		UPDATE	IMS.plxname.RMC
RMD		UPDATE	IMS.plxname.RMD
RMG		UPDATE	IMS.plxname.RMG
RMI		UPDATE	IMS.plxname.RMI
RML		READ	IMS.plxname.RML
RMN		UPDATE	IMS.plxname.RMN
RST	LINE	UPDATE	IMS.plxname.RST.LINE
RST	LINK	UPDATE	IMS.plxname.RST.LINK
RST	MSPLINK	UPDATE	IMS.plxname.RST.MSPLINK
RST	NODE	UPDATE	IMS.plxname.RST.NODE
RST	USER	UPDATE	IMS.plxname.RST.USER
RTA	DUMPQ	UPDATE	IMS.plxname.RTA.DUMPQ
RTA	FREEZE	UPDATE	IMS.plxname.RTA.FREEZE
RTA	UNPLAN	UPDATE	IMS.plxname.RTA.UNPLAN
SEC	APPC	UPDATE	IMS.plxname.SEC.APPC
SEC	OTMA	UPDATE	IMS.plxname.SEC.OTMA
SMC	MASTER	UPDATE	IMS.plxname.SMC.MASTER
SMC	TERMINAL	UPDATE	IMS.plxname.SMC.TERMINAL
STA	APPC	UPDATE	IMS.plxname.STA.APPC
STA	AREA	UPDATE	IMS.plxname.STA.AREA
STA	AUTOARCH	UPDATE	IMS.plxname.STA.AUTOARCH
STA	CLASS	UPDATE	IMS.plxname.STA.CLASS

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
STA	DB	UPDATE	IMS.plxname.STA.DB
STA	DATAGRP	UPDATE	IMS.plxname.STA.DATAGRP
STA	DC	UPDATE	IMS.plxname.STA.DC
STA	ISOLOG	UPDATE	IMS.plxname.STA.ISOLOG
STA	LINE	UPDATE	IMS.plxname.STA.LINE
STA	LTERM	UPDATE	IMS.plxname.STA.LTERM
STA	LU	UPDATE	IMS.plxname.STA.LU
STA	MADSIOT	UPDATE	IMS.plxname.STA.MADSIOT
STA	MSNAME	UPDATE	IMS.plxname.STA.MSNAME
STA	NODE	UPDATE	IMS.plxname.STA.NODE
STA	OLDS	UPDATE	IMS.plxname.STA.OLDS
STA	OTMA	UPDATE	IMS.plxname.STA.OTMA
STA	PGM	UPDATE	IMS.plxname.STA.PGM
STA	REGION	UPDATE	IMS.plxname.STA.REGION
STA	RTC	UPDATE	IMS.plxname.STA.RTC
STA	SB	UPDATE	IMS.plxname.STA.SB
STA	SERVGRP	UPDATE	IMS.plxname.STA.SERVGRP
STA	SUBSYS	UPDATE	IMS.plxname.STA.SUBSYS
STA	SURV	UPDATE	IMS.plxname.STA.SURV
STA	THREAD	UPDATE	IMS.plxname.STA.THREAD
STA	TMEM	UPDATE	IMS.plxname.STA.TMEM
STA	TRAN	UPDATE	IMS.plxname.STA.TRAN
STA	TRKARCH	UPDATE	IMS.plxname.STA.TRKARCH
STA	USER	UPDATE	IMS.plxname.STA.USER
STA	VGR	UPDATE	IMS.plxname.STA.VGR
STA	WADS	UPDATE	IMS.plxname.STA.WADS
STA	XRCTRACK	UPDATE	IMS.plxname.STA.XRCTRACK
STO	ADS	UPDATE	IMS.plxname.STO.ADS
STO	APPC	UPDATE	IMS.plxname.STO.APPC
STO	AREA	UPDATE	IMS.plxname.STO.AREA
STO	AUTOARCH	UPDATE	IMS.plxname.STO.AUTOARCH
STO	BACKUP	UPDATE	IMS.plxname.STO.BACKUP
STO	CLASS	UPDATE	IMS.plxname.STO.CLASS
STO	DB	UPDATE	IMS.plxname.STO.DB

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
STO	DATAGRP	UPDATE	IMS.plxname.STO.DATAGRP
STO	DC	UPDATE	IMS.plxname.STO.DC
STO	LINE	UPDATE	IMS.plxname.STO.LINE
STO	LTERM	UPDATE	IMS.plxname.STO.LTERM
STO	LU	UPDATE	IMS.plxname.STO.LU
STO	MADSIOT	UPDATE	IMS.plxname.STO.MADSIOT
STO	MSNAME	UPDATE	IMS.plxname.STO.MSNAME
STO	NODE	UPDATE	IMS.plxname.STO.NODE
STO	OLDS	UPDATE	IMS.plxname.STO.OLDS
STO	OTMA	UPDATE	IMS.plxname.STO.OTMA
STO	PGM	UPDATE	IMS.plxname.STO.PGM
STO	REGION	UPDATE	IMS.plxname.STO.REGION
STO	RTC	UPDATE	IMS.plxname.STO.RTC
STO	SB	UPDATE	IMS.plxname.STO.SB
STO	SERVGRP	UPDATE	IMS.plxname.STO.SERVGRP
STO	SUBSYS	UPDATE	IMS.plxname.STO.SUBSYS
STO	SURV	UPDATE	IMS.plxname.STO.SURV
STO	THREAD	UPDATE	IMS.plxname.STO.THREAD
STO	TMEM	UPDATE	IMS.plxname.STO.TMEM
STO	TRAN	UPDATE	IMS.plxname.STO.TRAN
STO	USER	UPDATE	IMS.plxname.STO.USER
STO	VGR	UPDATE	IMS.plxname.STO.VGR
STO	WADS	UPDATE	IMS.plxname.STO.WADS
STO	XRCTRACK	UPDATE	IMS.plxname.STO.XRCTRACK
SWI	OLDS	UPDATE	IMS.plxname.SWI.OLDS
SWI	SYSTEM	UPDATE	IMS.plxname.SWI.SYSTEM
SWI	WADS	UPDATE	IMS.plxname.SWI.WADS
TERM	OLC	UPDATE	IMS.plxname.TERM.OLC
TERM	OLREORG	UPDATE	IMS.plxname.TERM.OLREORG
TES	MFS	UPDATE	IMS.plxname.TES.MFS
TRA	SET	UPDATE	IMS.plxname.TRA.SET
UNL	DB	UPDATE	IMS.plxname.UNL.DB
UNL	PGM	UPDATE	IMS.plxname.UNL.PGM
UNL	SYSTEM	UPDATE	IMS.plxname.UNL.SYSTEM

Table 13. Resource names and RACF authority for IMS commands (continued)

Command verb	Command keyword	Authority	Resource name
UNL	TRAN	UPDATE	IMS.plxname.UNL.TRAN
UPD	AREA	UPDATE	IMS.plxname.UPD.AREA
UPD	DATAGRP	UPDATE	IMS.plxname.UPD.DATAGRP
UPD	DB	UPDATE	IMS.plxname.UPD.DB
UPD	DBDESC	UPDATE	IMS.plxname.UPD.DBDESC
UPD	IMS	UPDATE	IMS.plxname.UPD.IMS
UPD	IMSCON	UPDATE	IMS.plxname.UPD.IMSCON
UPD	IMSFUNC	UPDATE	IMS.plxname.UPD.IMSFUNC
UPD	LE	UPDATE	IMS.plxname.UPD.LE
UPD	MSLINK	UPDATE	IMS.plxname.UPD.MSLINK
UPD	MSNAME	UPDATE	IMS.plxname.UPD.MSNAME
UPD	MSPLINK	UPDATE	IMS.plxname.UPD.MSPLINK
UPD	OLREORG	UPDATE	IMS.plxname.UPD.OLREORG
UPD	PGM	UPDATE	IMS.plxname.UPD.PGM
UPD	PGMDESC	UPDATE	IMS.plxname.UPD.PGMDESC
UPD	POOL	UPDATE	IMS.plxname.UPD.POOL
UPD	RM	UPDATE	IMS.plxname.UPD.RM
UPD	RTC	UPDATE	IMS.plxname.UPD.RTC
UPD	RTCDESC	UPDATE	IMS.plxname.UPD.RTCDESC
UPD	TRACE	UPDATE	IMS.plxname.UPD.TRACE
UPD	TRAN	UPDATE	IMS.plxname.UPD.TRAN
UPD	TRANDESC	UPDATE	IMS.plxname.UPD.TRANDESC
VUN	AREA	UPDATE	IMS.plxname.VUN.AREA

Related concepts

Commands for IMS operations tasks (Operations and Automation)

Related reference

Equivalent IMS type-1 and type-2 commands (Commands)

CSL OM Security user exit routine

OM invokes the optional OM Security user-defined exit routine during command processing. The exit routine performs user-defined security checking when you specify with A or E for the CMDSEC parameter in the OM procedure. The OM Security user-defined exit routine is given control after the OM Input exit routine.

The CSL OM Security user exit routine is defined as TYPE=SECURITY in the EXITDEF statement in the BPE exit list PROCLIB member data set. You can specify one or more user-defined exit routines of this type. When this exit is invoked, all user-defined exit routines of this type are driven in the order specified by the EXITS=*keyword*.

Related concepts

Defining the Common Service Layer using IMS PROCLIB data set members (System Definition)

Related reference

CSL OM Security user exit (Exit Routines)

Chapter 10. CSL RM administration

The tasks associated with administering the Resource Manager (RM) are typically performed by the system administrator or system operator.

Information managed by the CSL RM

You can use the Resource Manager (RM) to create, update, query, or delete resources in the resource structure.

Examples of resources whose information is stored on a resource structure include:

- Global resource status information for databases, DEDB areas, and transactions
- Message destination resource information; for example, APPC descriptors, VTAM LTERMS, MSNAMEs, VTAM terminal nodes, users, and user IDs
- Global information for the OLCSTAT data set; for example, current online change libraries
- IMSplex-wide parameters
- Scheduled serial programs

RM maintains the following information about resources on a resource structure:

Resource Name
A client-defined resource name that is 11 bytes in length.

Name Type
A resource attribute that ensures that all resources within the same type have unique names. RM enforces this rule. Resources within a name type can have different resource types.

Resource Type
A client-defined resource attribute that allows CQS to physically group resources on a coupling facility list structure. RM supports up to 255 resource types.

RM uses resource type 253 to store its global information. Resource type 253 hashes to the same physical list headers as resource types 11, 22, 33, 44,...253. A client can define resource types that map to the same physical list headers as RM's global information.

Resource Version
The number of times that the resource has been updated. RM uses the version to serialize updates on the resource structure.

Resource Owner
A resource attribute that signifies the owner of the resource. This attribute is optional.

If the owner is identified, the client is responsible for enforcing a single owner of a resource. For example, IMS can set owners for the following resources to enforce a single active instance of the resources: nodes, lterms, users, and user IDs. A user or user ID that is signed on to one IMS in the IMSplex cannot be signed on to another IMS in the IMSplex at the same time. An lterm that is active on one IMS in the IMSplex cannot be active on another IMS in the IMSplex at the same time.

Resource Data
A resource attribute that contains any additional data about the resource.

RM registers to the Repository Server and connects to the IMSRSC repository during RM initialization or during the UPDATE RM command processing. If RM is defined with the resource structure, RM uses the resource structure to save the repository information, such as repository name, repository type, audit access values, and the RS z/OS cross-system coupling facility (XCF) group information. The resource structure ensures that all RMs in the IMSplex use the same repository.

To display information about the resource structure managed by RM, use the **QUERY STRUCTURE** command. More information on this command is in *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Related concepts

"CSL RM management of the IMSRSC repository" on page 181
The Common Service Layer (CSL) Resource Manager (RM) interacts with the Repository Server (RS) address space to manage the stored resource definitions in the IMSRSC repository.

Maintaining global information for databases, DEDB areas, and transactions

IMS uses the RM resource structure to maintain command status for databases, DEDB areas, and transactions across an IMSplex. This global status enables all IMS systems to view databases, DEDB areas, and transactions in an IMSplex as single databases, areas, and transactions.

If an IMS system restarts, it uses any global command status that is set in RM and applies that status to its resources, if necessary.

The status of global resources is maintained for most databases whether or not they are registered to DBRC, for DEDB areas, and for transactions. Database status is not maintained for shared secondary index databases and MSDBs.

If an error occurs when RM obtains database or DEDB area status, the global status is not maintained for that resource, and message DFS3308E is issued. DFS3308E lists the RM error return and reason code, followed by message DFS3500I, which indicates that global status for database or DEDB area resources is disabled. All subsequent DB or AREA commands with SCOPE(ALL) or GLOBAL prevent RM status from being updated. After the error is corrected, you can issue **`UPD IMS SET`** to enable global status again.

After an online change is committed, global status is obtained for all databases, DEDB areas, or transactions being added. Global command status for **`CREATE DB`** and **`CREATE TRAN`** commands is also obtained from RM. The databases, DEDB areas, or transactions cannot be used until global status for the resources added is obtained and processed. If an error occurs obtaining or processing the global status for the added resources, the added resources are marked in error.

To maintain global status for databases, DEDB areas, and transactions in RM, specify the appropriate PLEXPARM value during IMS initialization in either the DFSDFxxx or DFSCGxxx IMS PROCLIB member data set.

The status of global resources is set by the first IMS system that starts in an IMSplex, and the values are obtained from the PLEXPARM parameters on that system. By maintaining this information globally using RM, you can, for example, stop a database globally, and any IMS system that joins the IMSplex recognizes that the database is stopped. Similarly, an IMS system joining the IMSplex can be prevented from accessing or updating a database that is in use by an offline process.

Only the last significant global status is maintained in the RM structure for the databases, DEDB areas and transaction resources. If multiple update commands are issued for a resource, only the most recent command status is maintained and available to be queried as the global status. For example, if an **`UPD TRAN NAME(TRANA) STOP(SCHD)`** command is issued, the global status of STOSCHD is maintained for transaction TRANA. If, an **`UPDATE TRAN NAME(TRANA) STOP(Q)`** command is then issued, the global status of STOQ is maintained for transaction TRANA. To maintain both the STOSCHD and STOQ status, an **`UPDATE TRAN NAME(TRANA) STOP(Q,SHCD)`** command must be issued.

You can also change the PLEXPARMs dynamically using the **`UPD IMS SET (PLEXPARM())`** command. See *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V* for information about the **`UPD IMS`** command.

RM works with CQS to access the resource structure for the client. RM issues the requests to query and maintain the resources, and it notifies the client if there is a resource structure change that affects the client.

The IMS systems in the IMSplex still contain the resource definitions. RM does not ensure resource definition consistency across the IMSplex. You can use global online change to update the resource definitions in the IMS systems.

The following table shows how IMS obtains and updates global status for resources, based on how IMS is started:

Table 14. How IMS obtains and updates global status for resources

Type of start	How status is obtained and updated
IMS initialization	IMS reads the PLEXPARM values from RM to determine whether the RM value takes precedence, or the IMS definition value should be used. IMS updates the PLEXPARM values in RM for other IMS systems that subsequently come up. If RM global status is maintained, IMS obtains global status of databases, DEDB areas, and transactions from RM's resource structure.
Cold start or **/ERE COLDBASE** or **/ERE COLDSYS** command	Global status of databases, DEDB areas, and transactions is obtained from RM's resource structure.
Warm start and emergency restart	Local status from log records is applied first. After log record processing completes, if IMS determines that the global status that it read from RM was updated while the IMS system was down, the RM global status that was read from RM is applied to the resource.

Information stored for databases resources

The information that RM maintains for databases includes:

- Database names
- Database types
- Database access type
- HALDB master name if the database is a HALDB partition
- Database status

The commands and system actions that cause global status to be updated for databases include:

- **/DBDUMP DB** with the GLOBAL keyword
- **/DBRECOVERY DB** with the GLOBAL keyword
- **/START DB** with the GLOBAL keyword
- **/STOP DB** with the GLOBAL keyword
- **UPDATE DB** with any of the following keywords specified: START(ACCESS) SET(ACCTYP) STOP(ACCESS | SCHD | UPDATES) SET(LOCK(ON|OFF)) and SCOPE=ALL

Global database information is deleted from the RM structure using the **UPD IMS SET(PLEXPARM(GSTSDB(N)))** command.

Information stored for DEDB area resources

The information that RM maintains for DEDB areas includes:

- DEDB names
- DEDB area access
- DEDB area names
- DEDB area status

The commands and system actions that cause global status to be updated for DEDB areas include:

- **/DBRECOVERY AREA** with the GLOBAL keyword
- **/START AREA** with the GLOBAL keyword
- **/STOP AREA** with the GLOBAL keyword

- **UPDATE AREA** with any of the following keywords specified START(ACCESS) SET(ACCTYPE) STOP(ACCESS | SCHD) and SCOPE=ALL

Global DEDB area information is deleted from the RM structure using the **UPD IMS SET(PLEXPARM(GSTSAREA(N)))** command.

Information stored for transaction resources

The information that RM maintains for transactions includes the transaction name and the transaction status.

The commands and system actions that cause global status to be updated for transactions include:

- **UPD TRAN STOP(Q) SCOPE=ALL**
- **UPD TRAN STOP(SCHD) SCOPE=ALL**
- **UPD TRAN START(Q) SCOPE=ALL**
- **UPD TRAN START(SCHD) SCOPE=ALL**

To reset global status to NULL, issue the **UPD IMS SET(PLEXPARM(GSTSTRAN=N))** command.

Maintaining message destination resource information

When the RM uses a resource structure in an IMSplex, you can indicate that IMS shares IMS Transaction Manager resources. This message destination resource information includes terminals, users, user IDs, LTERMs, MSNAMEs, and APPC descriptors.

To maintain message destination resource information, specify STM=YES on the DFSDCxxx member of the IMS PROCLIB data set. See *IMS Version 15.2 System Definition* for more information about DFSDCxxx.

See Chapter 18, “Planning for Transaction Manager resources in an IMSplex,” on page 279 for more information on TM Resources.

Maintaining global information for the OLCSTAT data set

The OLCSTAT data set name is maintained in the RM resource structure. This ensures that all IMS systems in the IMSplex that can perform global online change use the same OLCSTAT data set name.

The first IMS system that starts in an IMSplex writes its OLCSTAT name to the OLCSTAT data set. IMS systems that subsequently start compare their OLCSTAT name to the OLCSTAT name in RM. If the OLCSTAT names do not match, IMS abends with 2800, code 09.

Maintaining IMSplex-wide parameters

The PLEXPARM parameters, GSTSAREA=, GSTSDB=, and GSTSTRAN=, define whether or not global status is maintained. Values for these parameters are maintained in the RM resource structure.

This ensures that all IMS systems in the IMSplex have the same values for the global PLEXPARM values. The first IMS system that starts in an IMSplex writes its global PLEXPARM values from its DFSCGxxx or DFSDFxxx PROCLIB member data set to the RM resource structure. IMS systems that subsequently start use the values defined in RM. If the PLEXPARM values in the IMS system starting up do not match the PLEXPARM values in RM, message DFS3425I is issued, indicating that the IMS values are overridden by the RM values.

How IMS maintains global information for sysplex serialized program management

Sysplex serialized program management allows users in a shared queues environment to prevent application programs that are defined as serial from being scheduled in parallel on another IMS system in an IMSplex.

Information about scheduled serial PSBs is maintained in the Resource Manager (RM) resource structure to ensure that the serial PSB is scheduled in only one IMS across an IMSplex at any point in time.

For an IMS in an IMSplex to schedule a serial PSB, that IMS must successfully create a unique instance for the PSB on the RM resource structure before it completes scheduling. If another IMS finds that the PSB instance is not unique, it discontinues the scheduling process for the PSB.

You can activate or deactivate the sysplex serial program management feature by using the GBL_SERIAL_PGM parameter in the DFSCGxxx member of the IMS PROCLIB data set. By default, sysplex serialized program management is active.

Sysplex serial program management requires the following components and structures:

- The Operations Manager (OM), Structured Call Interface (SCI), and RM components of the Common Service Layer (CSL)
- An RM resource structure defined in a coupling facility
- Shared message queues

Shared queues, RM, and RM structures must be defined in order to be active.

Resource structure duplexing requirements for CSL RM

CQS does not log resource updates or support a checkpoint of the resource structure. However, CQS supports automatic duplexing of the resource structure for backup in case of structure failure.

You must define the resource structure in the coupling facility resource management (CFRM) policy to automatically be duplexed if you want the resource structure to be recoverable.

How the CSL RM repopulates a resource structure

If the resource structure and its duplex (if applicable) fail, and CQS can allocate a new structure, CQS notifies RM to repopulate the structure. RM repopulates the structure from information in its local control blocks.

RM then issues a directive to its clients to populate the structure. Any RM or IMS resource that existed only on the resource structure is lost. When a new resource structure is allocated, the clients can choose to coordinate the repopulation of the resource structure.

Every active IMS system that receives the RM repopulate directive updates RM with the following information:

- Its own PLEXPARM values.
- Its transactions.
- Its local list of database resources, DEDB area resources, and transaction resources that have global status (if global status is maintained).
- The OLCSTAT name
- Terminals, users (or ISC subpools), LTERMs, user IDs, and APPC descriptor names (if STM=YES is specified in DFSDCxxx during IMS initialization) for VTAM resources only

If an IMSRSC repository is enabled to store resource definitions, when RM is driven to repopulate the resource structure, each RM writes its repository information to the resource structure.

If all IMS systems are down when RM requests repopulation, no IMS information like the global status for databases, DEDB areas, or transactions is available in the new RM structure.

How z/OS rebuilds a resource structure

Resource structures are defined with system-managed rebuild, so z/OS automatically rebuilds the structure when no CQS is up to build the structure. z/OS cannot rebuild the structure if the structure fails or if z/OS loses connectivity to the structure.

If any CQS is up and rebuild is initiated with the **`SETXCF START,REBUILD`** command, then CQS copies the structure. If the structure fails, no structure recovery is initiated because resource structures do not support structure checkpoint.

Chapter 11. CSL SCI administration

This topic describes the administrative tasks associated with SCI.

CSL SCI security

To access the IMSplex, clients must first register with SCI. A client can register with SCI only if the user ID of the address space in which the client is running has the authority to do so. Define the authority for a client address space in the RACF FACILITY class profile for the IMSplex.

The profile names in the RACF FACILITY class must be in the form CSL.*imsplex_name*, where *imsplex_name* is the name of the IMSplex that is being protected. The *imsplex_name* is the characters "CSL" followed by the IMSplex name as defined on the IMSPLEX parameter in the CSLSIxxx PROCLIB member data set.

SCI checks security when a client issues a CSLSCREG request to register with SCI. In response to the CSLSCREG request, SCI issues a RACROUTE REQUEST=AUTH call to RACF to see if the client has the authority to register with SCI. RACF checks the user ID of the address space that issued the CSLSCREG request. This user ID must have at least UPDATE authority to register with SCI. An IMSplex with a RACF FACILITY class that has no UACC(NONE) profile and no profiles matching a particular user ID is unprotected. CSLSCREG requests to register with SCI will be authorized for an unprotected IMSplex.

Important: The SCI address space registers with itself but does not need security authorization.

Example: An IMSplex, PLEX1, has the IMSplex ID of CSLPLEX1. To define a profile that allows only users IMSUSER1 and IMSUSER2 to register with SCI in PLEX1, issue the following RACF commands:

```
RDEFINE FACILITY CSL.CSLPLEX1 UACC(NONE)
PERMIT CSL.CSLPLEX1 CLASS(FACILITY) ID(IMSUSER1) ACCESS(UPDATE)
PERMIT CSL.CSLPLEX1 CLASS(FACILITY) ID(IMSUSER2) ACCESS(UPDATE)
SETROPTS CLASSACT(FACILITY)
```

Chapter 12. CQS administration

This topic describes the system administration tasks associated with using the Common Queue Server.

This topic contains Product-sensitive Programming Interface information.

Recording information necessary for starting CQS

CQS uses the z/OS system logger to record all information necessary for CQS to recover queue structures and restart. CQS writes log records for each coupling facility list structure pair that it uses to a separate log stream. The log stream is shared among all CQS address spaces that share the queue structure.

The z/OS system logger provides a merged log for all CQS address spaces that are sharing queues on a coupling facility list structure.

Important: Changes to resource structures are not logged.

For CQS to use a z/OS system log, you must first define the log stream and associated resources to z/OS.

CQS also provides a File Select and Formatting Print utility to print the log records.

For more information on defining the log stream, see *z/OS MVS Setting Up a Sysplex*.

For more information on the File Select and Formatting Print utility, see *IMS Version 15.2 System Utilities*.

Related concepts

"CQS recovery functions" on page 60
CQS provides functions for recovering work-in-progress, queues, and resources in case of system shutdown or failure. Some of these recovery functions are built-in and do not require intervention.

Establishing client connection to CQS during failed client takeover

When the client (such as IMS) supports XRF takeover capability, one client must take over the work for a failed client. The CQS connected to this failed client might not be active. Therefore, during the takeover process, the client taking over work from a failed client must connect to a different CQS and indicate that it is taking over work from another client.

At this point, CQS must perform a process similar to CQS restart, using the log records from the CQS connected to the failed client. Normally, CQS failed client connection restart (warm start) is automatic and you do not need to take any action.

During a failed client connection, CQS reads the log token from the structure for the CQS connected to the failed client.

- If CQS finds the log token, CQS performs warm start processing for the failed client.
- If CQS does not find the log token, CQS issues WTOR CQS0033A. At this point, you can do one of the following:
  - Cold start the client connection.
  - Reject the client connection request.
  - Specify a new log token.

Recommendation: If a CQS does not accept a log token during failed client connection restart, cold start the connection. If multiple CQSs are running, one CQS structure checkpoint might purge the log records for another CQS that previously failed and was not restarted.

Authorizing access to CQS

If RACF or another security product is installed at your installation, the security administrator can define profiles that control the ability of clients to connect to and access CQS structures.

Authorizing CQS registration

When a client issues the CQSREG request to register with CQS, CQS issues a RACROUTE REQUEST=AUTH call to determine whether the client is authorized to register with CQS. RACF checks the user ID of the client that issued the CQSREG request. This user ID must have at least UPDATE authority to register with CQS.

The RACF security administrator can define profiles in the FACILITY class to control registration with CQS. The profile names must be of the form `CQS.`*`cqs_id`*, where *cqs_id* is the ID of the CQS that is to be protected. The *cqs_id* value is the subsystem name (SSN) as defined in the CQSIPxxx PROCLIB member data set, followed by the characters `CQS`. For example, if the SSN is ABC, the *cqs_id* value is ABCCQS.

RACF commands for authorizing CQS registration

To define a profile for CQS to prevent users other than CQSUSER1 and CQSUSER2 from registering, issue the RACF commands shown in the following example.

```
RDEFINE FACILITY CQS.ABCCQS UACC(NONE)
PERMIT CQS.ABCCQS CLASS(FACILITY) ID(CQSUSER1) ACCESS(UPDATE)
PERMIT CQS.ABCCQS CLASS(FACILITY) ID(CQSUSER2) ACCESS(UPDATE)
SETROPTS CLASSACT(FACILITY)
```

Authorizing connections to CQS structures

When a client issues the CQSCONN request to connect to a CQS structure, CQS issues a **RACROUTE REQUEST=AUTH** call to determine whether the client is authorized to access the structure. RACF checks the user ID of the client that issued the CQSCONN request. This user ID must have at least UPDATE authority to connect to the structure through CQS.

The RACF security administrator should define profiles in the FACILITY class to control the connection to CQS structures. The profile names must be of the form `CQSSTR.`*`structure_name`*, where *structure_name* is the name of the primary CQS structure that is to be protected. Use the same structure name that you define in the CQSSGxxx and CQSSLxxx PROCLIB member data sets.

The `CQSSTR.`*`structure_name`* profiles only control access to the specified structures through CQS; they do not control direct access to the structures using IXL macros. You can provide control over direct structure access by defining RACF profiles of the form `IXLSTR.`*`structure_name`*. If you create such profiles, you must give the user IDs under which you run CQS access to the structures.

Related reading: For information on protecting direct access to coupling facility structures, see "Authorizing Coupling Facility Requests" in *z/OS MVS Programming: Sysplex Services Guide*.

For more information on defining structure names, see *IMS Version 15.2 System Definition*. CQS does not perform a separate check on the overflow structure name, because the primary and overflow structures are considered one unit.

Example: To define a profile for a CQS primary structure named IMSMSGQ01, and to allow only user CQSUSER to connect to it, issue the RACF commands shown in the following example.

```
RDEFINE FACILITY CQSSTR.IMSMSGQ01 UACC(NONE)
PERMIT CQSSTR.IMSMSGQ01 CLASS(FACILITY) ID(CQSUSER) ACCESS(UPDATE)
SETROPTS CLASSACT(FACILITY)
```

If you do not define a profile for a particular CQS structure, the structure is not protected, and any user ID can issue a CQSCONN request to access the structure.

Using structure alter for CQS

Structure alter is a z/OS process supported by CQS that can be used to alter the structure size or to redistribute the objects within the structure. CQS supports structure alter for primary queue structures, overflow queue structures, and resource structures. CQS allows you to dynamically change the size of a primary or overflow structure.

To enable structure alter, activate a CFRM policy and define the INITSIZE and SIZE parameters in this policy. For information on structure size, see *IMS Version 15.2 System Definition*.

To change the structure size, enter the following XES command:

```
SETXCF START,ALTER,STRNAME=strname,SIZE=size
```

The value of *size* must be within the range of values between INITSIZE and SIZE in the CFRM policy.

Automatic structure alter is a z/OS function that can automatically alter the structure size or the element to entry ratio when the structure full threshold is reached. CQS supports automatic structure alter for queue structures and resource structures. To enable automatic structure alter, activate a CFRM policy defined with INITSIZE, SIZE, ALLOWAUTOALT(YES).

Important: A structure enabled with automatic structure alter is a candidate to be contracted in size by z/OS, if the coupling facility storage becomes constrained. Be careful when enabling automatic structure alter for queue structures. If z/OS contracts the queue structure size, it might cause the queue structure to go into overflow unnecessarily. To prevent this happening, define the CFRM policy with a MINSIZE (minimum size), below which z/OS will not contract the structure.

The FULLTHRESHOLD parameter in the CFRM policy structure definition specifies the percent full threshold for the structure. This threshold is the percent full that coupling facility resources such as entries or elements must reach before auto alter processing is triggered. The default value for the FULLTHRESHOLD parameter is 80 percent. The CQSSGxxx PROCLIB member parameter OVFLWMAX specifies the percentage of elements in the structure that must be in use to trigger CQS overflow processing. The default value for OVFLWMAX is 70 percent.

Recommendation: If you are using ALLOWAUTOALT(YES), change one or both of these default values so that the OVFLWMAX value is larger than the FULLTHRESHOLD value by at least five percent. This is to give z/OS a chance to alter the entry-to-element ratio and structure size, and potentially avoid CQS overflow processing. Also, setting these parameters to different values reduces the chance of both auto alter and CQS overflow processing occurring at the same time and causing extra overhead.

For more information on automatic Structure Alter, see *z/OS MVS Setting Up a Sysplex*.

Using CQS system checkpoint

CQS system checkpoint, the checkpoint data sets that are used for recovery, applies to a CQS if it manages at least one queue structure. If a CQS manages only a resource structure, system checkpoint does not apply.

At a system checkpoint for recovering CQS information, CQS writes log records that contain restart and recovery information to the CQS log. CQS does not stop activity while the checkpoint is in progress.

CQS performs a system checkpoint in each of the following situations:

- When a client issues a CQSCHKPT FUNC=CHKPTSYS request
- When the number of log records that CQS writes reaches the value specified on the SYSCHKPT= parameter in the CQSSLxxx PROCLIB member data set
- When the client is IMS and you enter the **`/CQCHKPT SYSTEM`** command
- When a client RESYNC ends
- When structure checkpoint ends successfully
- At the end of a restart

In addition, CQS takes system checkpoints during significant events, such as a shutdown.

CQS checkpoint data set

For each structure pair, CQS maintains a checkpoint data set. CQS writes to its checkpoint data set and uses it during restart. The checkpoint data set is dynamically allocated during CQS initialization. You define the checkpoint data set DSNAME for a structure using the CHKPTDSN= parameter in PROCLIB member data set CQSSLxxx.

CQS also takes Structure Checkpoints; see Using CQS structure checkpoint (System Administration) for details.

How CQS restarts after system checkpoint

During CQS restart, CQS reads the log records from the last system checkpoint and restores the environment for committed data objects and backs out uncommitted data objects on queue structures. The frequency of system checkpoint affects this restart. CQS must read more log records when checkpoints are infrequent than when the checkpoints occur more often. Because the CQS log is shared by multiple CQSs, CQS restart time is affected by the number of log records written by the multiple CQSs, not just the CQS that is being restarted.

CQS takes an initial system checkpoint at the end of a restart.

Related concepts
"CQS system checkpoint data set" on page 89
Each CQS subsystem maintains a system checkpoint data set for each structure pair in the coupling facility.

Using CQS structure checkpoint

Structure checkpoint takes a snapshot of the shared queues on a queue structure and writes the data to the structure recovery data set (SRDS) so that CQS can recover the queues after a structure failure. Structure checkpoint processing copies all recoverable data objects from a structure pair to a SRDS.

For nonrecoverable data objects, the queue name, and UOW are copied, but not the actual data object. The client specifies whether or not a data object is recoverable when the CQSPUT FUNC=PUT request is issued to insert the data object onto the shared queues. For example, when IMS is the client, all data objects are marked as recoverable, except for Fast Path input messages.

Important: Structure checkpoint is not supported for resource structures. It supports queue structures only.

When it performs the copy operation, CQS stops all activity against the structure to ensure that the structure does not change while the checkpoint is being taken. If CQS receives a request to process work when a structure checkpoint is in progress, the request is held until after the structure checkpoint is complete.

Recommendation: Because no other work for a structure can be processed while CQS is taking a checkpoint, consider processing structure checkpoints during non-peak hours.

After all shared queues are copied to the SRDS, each CQS performs a system checkpoint to ensure its restart checkpoint has a time stamp that is more recent than the current structure checkpoint. The structure checkpoint process then deletes all log records that are not needed for structure recovery, allowing the logger to reclaim space in the CQS log and preventing the log from becoming full. After log records are deleted, CQS cannot access these log records and, therefore, cannot use these records for structure recovery or CQS restart. If only one SRDS contains valid structure checkpoint data, all log records that were written prior to that structure checkpoint are deleted. If both SRDSs contain valid structure checkpoint data, all log records that were written prior to the oldest structure checkpoint are deleted.

If a CQS was not active at the time of a structure checkpoint, it cannot initiate a system checkpoint, meaning that its restart checkpoint is older than at least one structure checkpoint. If both SRDSs contain valid structure checkpoint data, no problem exists (because the CQS restart checkpoint is still more recent than the oldest structure checkpoint, so its restart log records are not deleted). However, if this is the first or only valid structure checkpoint, or a CQS was down across two structure checkpoints, the log records needed for that CQS to restart are deleted. In this case, that CQS might need a cold start to restart.

Recommendation: Initiate a structure checkpoint after a structure cold start, or anytime the SRDSs are deleted and redefined. The structure checkpoint should start after all CQSs that share the structure are started. This provides the SRDS with an initial structure checkpoint. Also, to update the snapshot of the shared queues and to periodically delete log records, initiate a structure checkpoint at regular intervals.

When a structure recovery is required, the SRDS and the CQS log are used to recover the shared queues. CQS first repopulates the new structure from the SRDS. CQS then reads all log records from the time the structure checkpoint completed. The length of time to read the log records is dependent on how many log records are in the log. More frequent structure checkpoints reduce the number of log records that must be read during a structure recovery. Deleting the log records also helps prevent the log from becoming full. When a log stream becomes full, CQS deletes all log records older than the oldest structure checkpoint or CQS system checkpoint. CQS then takes a structure checkpoint.

CQS performs structure checkpoints in each of the following situations:

- When the z/OS log becomes full or approaches full and the **CHKNEARFULL** parameter is specified as *YES* in the CQSSGxxx PROCLIB member. For details, see CQSSGxxx member of the IMS PROCLIB data set (System Definition).
- After a successful structure recovery.
- After a successful overflow threshold process.
- When a client issues the CQSCHKPT FUNC=CHKPTSTR request.
- When the client is IMS, and you enter the **/CQCHKPT SHAREDQ** command.
- During CQS normal termination when the client requests it on the CQSDISC request. When the client is IMS, you can request a structure checkpoint at CQS termination by entering the **/CQSET SHUTDOWN SHAREDQ** command. IMS then passes this request to CQS when IMS terminates normally with a **/CHECKPOINT FREEZE|DUMPQ|PURGE** command.
- When the STRCHKPT=*xxxx* value defined for CQS in the CQSSLxxx PROCLIB member is reached. For details, see CQSSLxxx member of the IMS PROCLIB data set (System Definition).

Related concepts

“CQS recovery functions” on page 60
CQS provides functions for recovering work-in-progress, queues, and resources in case of system shutdown or failure. Some of these recovery functions are built-in and do not require intervention.

Preventing CQS structure full

You should manage structure usage to avoid a structure full condition. If a resource structure or queue structure becomes full, CQS issues message CQS0205E.

There are various ways to prevent a structure full condition:

- CQS structure overflow function for queue structures
- Monitoring shared message queue usage with the Queue Space Notification exit routine (DFSQSSP0)
- z/OS structure full monitoring capability, used with CQS, for queue structures and resource structures

Use these mechanisms to warn when a structure full condition is approaching and to take action to prevent a full structure.

Related concepts

“CQS structure functions” on page 59

CQS provides functions for monitoring structure status and capacity, and enabling structure recovery. Some of these functions are built-in functions and do not require intervention. Other functions are optional, and can be set up or initiated as your installation needs them.

CQS structure overflow function

CQS provides a structure overflow function that automatically warns you when a queue structure is approaching full and takes action to prevent a full structure. When the usage of a structure reaches the *overflow threshold*, CQS attempts to make the structure larger by initiating a structure alter. If the alter fails, CQS either allocates an overflow structure and moves data objects associated with selected queues to the overflow structure (if you define an overflow structure) or rejects data objects from being put on the selected queues. CQS stops all activity against the structure while the queues are being selected.

Definition: The *overflow threshold* is the percentage of the primary structure that must be in use before CQS goes into overflow mode. The default overflow threshold is 70%, but you can change the default by defining the OVFLWMAX parameter in the CQSSGxxx member of the PROCLIB data set.

Important: Structure overflow is not supported for resource structures.

If CQS does not succeed in altering a structure's size, the structure goes into *overflow mode*. In overflow mode, CQS selects queues using the most space on the structure as candidates for *overflow processing*. CQS stops selecting queues when enough queues have been selected to cause the primary structure usage to fall 20% below the overflow threshold. Activity against the structure is temporarily stopped while queues are being selected for overflow. CQS drives the Queue Overflow User-Supplied exit routine with the candidate queue names, which the exit then approves or rejects for overflow processing. Queues that get approved are placed into overflow mode. If an overflow structure is defined, CQS allocates the overflow structure and moves the data objects associated with the approved queues to the overflow structure. If an overflow structure is not defined, CQS rejects CQSPUT requests for the approved queues. Overflow structures can be defined in the CQSSGxxx member of the PROCLIB data set, using the OVFLWSTR parameter.

If an overflow structure is allocated, a queue remains in overflow mode until CQS detects that there are no more entries in the queue. (The queues are scanned every 15 minutes.) After the empty queue is detected, it is removed from overflow mode and returned to normal processing. Because the scan is periodic, empty queues might remain in overflow mode temporarily. When all queues have been removed, CQS exits overflow mode. If no overflow structure is allocated, CQS exits overflow mode when the primary structure usage drops 20% below the overflow threshold.

Remember: Large messages (messages that exceed the size of a queue manager long message record) are put on the shared queue structure in multiple parts. The first part of a message is placed on the transaction or LTERM ready queue. The second and subsequent parts are placed on a secondary queue called the *staging queue*. When a transaction or LTERM queue is selected for overflow processing, only the first part of any large messages on that queue are moved to the overflow structure. Any parts of a message that are on the staging queue remain on the primary structure. The staging queue is not eligible for overflow processing. You can use the LGMSGSZ parameter to change the size of the queue manager long message record.

Monitoring shared message queue usage with the Queue Space Notification exit routine (DFSQSSP0)

IMS provides the Queue Space Notification exit routine for shared queues (DFSQSSP0) with feedback on the overall utilization of the shared message queue structures on the coupling facility so that a solution can be implemented in the exit to monitor and prevent queue structures from becoming full.

When IMS runs by using shared queues, it stores transaction messages on the IMS shared messages queues coupling facility structure. The structure usage fluctuates during business operation depending on how busy the system is and how fast the transaction processing rate is. The structure might overflow if the rate of the messages arriving is faster than the rate of the messages being processed. The full condition of the shared queues might affect system availability.

You can receive information, through the Queue Space Notification exit routine, about how much of the message queue structure is used for both the primary structure and overflow structure:

- Number of structure entries in total and in use.
- Number of structure elements in total and in use.

This structure usage information is updated whenever IMS puts a message to the shared queue structure or deletes a message from the shared queue structure. The DFSQSSP0 exit parameter list, DFSPARM, provides input and output parameters for communication between IMS and the user exit DFSQSSP0.

In the Queue Space Notification exit parameter list DFSPARM, under the existing PARM DSECT, the following fields can be used to receive structure usage information:

- QSPCFBKP - Structure feedback area address
- QSPCFBKL - Structure feedback area length

For details about the feedback area, see the CQSSFBA macro, which is shipped with IMS.

Warning: The Queue Space Notification exit for shared queues (DFSQSSP0) provides structure utilization information only about the IMS shared message queue structure. It does not provide information about the IMS shared EMH structure.

Related reference
Queue Space Notification exit routine (DFSQSPC0/DFSQSSP0) (Exit Routines)
CQSDEL request (System Programming APIs)
CQSPUT request (System Programming APIs)

CQS structure full monitoring

The z/OS structure full monitoring capability can be used for queue structures and resource structures to warn you when a structure is approaching full and to prevent a full structure.

If structure full monitoring is enabled, z/OS monitors structure usage. When the number of entries or elements in use reaches the structure full threshold, z/OS issues a highlighted IXC585E message to warn the system programmer that a structure full condition is imminent. If automatic altering is enabled, z/OS automatically initiates a structure alter to increase the structure size or change the element-to-entry ratio.

Structure full monitoring is automatically enabled with a default threshold of 80%. Define a different threshold with the CFRM policy FULLTHRESHOLD parameter. Define the CFRM policy with `FULLTHRESHOLD(0)` to disable structure full monitoring. When the structure usage goes below the threshold, z/OS issues an IXC586I message.

The following command displays the structure full threshold that is in effect:

```
D XCF,STRUCTURE,STRNAME=strname
```

In the following example, the command **`D XCF,STRUCTURE,STRNAME=IMSRSRC01`** is issued and the structure full threshold is 80%.

```
STRNAME: IMSRSRC01
  STATUS: NOT ALLOCATED
    POLICY SIZE      : 4096 K
    POLICY INITSIZE  : N/A
    FULLTHRESHOLD    : 80
    REBUILD PERCENT  : N/A
    DUPLEX           : DISABLED
    PREFERENCE LIST  : LF03
    ENFORCEORDER     : NO
    EXCLUSION LIST IS EMPTY
```

The following example shows the IXC585E message, indicating the structure is full because all of the entries are in use:

```
*IXC585E STRUCTURE IMSRSRC01 IN COUPLING FACILITY LF03, 725
 PHYSICAL STRUCTURE VERSION B4704775 92D95302,
 IS AT OR ABOVE STRUCTURE FULL MONITORING THRESHOLD OF 80%.
```

```
  ENTRIES:  IN USE:     4874 TOTAL:     4874, 100% FULL
  ELEMENTS: IN USE:       19 TOTAL:     4872,   0% FULL
```

The following example shows the IXC586I message:

```
IXC586I STRUCTURE IMSRSRC01 IN COUPLING FACILITY LF03, 772
PHYSICAL STRUCTURE VERSION B4704775 92D95302,
IS NOW BELOW STRUCTURE FULL MONITORING THRESHOLD.
```

Related reading: For more details on structure full monitoring and the FULLTHRESHOLD and ALLOWAUTOALT keywords in the CFRM policy, see *z/OS MVS Setting Up a Sysplex*.

Using structure full monitoring with CQS structure overflow

You can use the structure full monitoring function (a z/OS function) with the structure overflow function (a CQS function) for queue structures.

The overflow threshold is a value defined to CQS. The structure full threshold is a value defined to z/OS. If the overflow threshold is close to the structure full threshold and automatic altering is enabled, CQS and z/OS might both try to initiate a structure alter at the same time to prevent the structure from becoming full.

If a CQS-initiated structure alter is in progress when z/OS detects the structure full threshold has been reached, z/OS stops the CQS-initiated structure alter and initiates its own structure alter. When CQS detects that its structure alter has failed, CQS goes into overflow mode, even if the z/OS-initiated structure alter reduces the structure usage below the overflow threshold.

Recommendation: Consider your structure full threshold when deciding what overflow threshold to define, so that you control when a structure goes into overflow mode. If you use structure full threshold, define it to be lower than the overflow threshold to avoid going into overflow mode unnecessarily. If the structure full threshold is lower than the overflow threshold, z/OS can attempt structure full threshold processing before the structure goes into overflow mode.

Related reading:

- For detailed information about the CQSSGxxx member of the PROCLIB data set, see *IMS Version 15.2 System Definition*.
- For detailed information about the Queue Overflow User-Supplied exit routine, see *IMS Version 15.2 Exit Routines*.
- For detailed information about the CQSPUT request, see *IMS Version 15.2 System Programming APIs*.

Rebuilding structures in CQS

Structure rebuild is a z/OS process that allows another instance of a structure to be allocated with the same name and data reconstructed from the initial structure instance. z/OS supports system-managed rebuild, CQS-managed rebuild, and structure duplexing.

CQS supports system-managed rebuild and CQS-managed rebuild for queue structures and resource structures. Note that CQS stops all activity against the structure during structure rebuild.

Related concepts

"CQS structure functions" on page 59
CQS provides functions for monitoring structure status and capacity, and enabling structure recovery. Some of these functions are built-in functions and do not require intervention. Other functions are optional, and can be set up or initiated as your installation needs them.

z/OS system-managed rebuild and CQS

System-managed rebuild is a z/OS process by which z/OS rebuilds the structure. z/OS copies the structure contents to a new structure. System-managed rebuild is supported for queue structures and resource structures. System-managed rebuild is only done if no CQS is up. If a CQS is up, the CQS performs a user-managed rebuild and does the structure copy.

Use system-managed rebuild primarily for planned reconfiguration. If the rebuild is initiated with the **`SETXCF START,REBUILD`** command and no CQS is available to perform the structure copy, z/OS performs the structure copy.

Restriction: System-managed rebuild does not address coupling facility failures, structure failures, or loss of connectivity. CQS-managed rebuild is required to handle such failures.

To enable a structure for system-managed rebuild, add the following parameter to your CFRM couple data set utility job, then run the job control language (JCL) to format the CFRM couple data set with system-managed rebuild capability:

```
ITEM NAME(SMREBLD) NUMBER(1)
```

CQS-managed rebuild

CQS-managed rebuild is a process by which CQS manages structure rebuild. CQS supports two variations of CQS-managed rebuild: structure copy and structure recovery.

Structure copy copies the contents of the structure to another structure, for a planned reconfiguration or connectivity loss. Structure copy can also be used to activate new CFRM policy attributes. Structure recovery recovers a structure from the SRDS and the z/OS log after a structure failure.

If one CQS loses connectivity to a structure and another CQS still has connectivity to that structure, CQS manages the structure rebuild and performs a structure copy. If all CQSs lose connectivity to a resource structure, structure recovery is attempted, but fails because structure recovery is not supported for resource structures or for nonrecoverable queue structures (RECOVERABLE=NO).

If a coupling facility or queue structure fails, CQS performs a structure recovery.

If a resource structure fails, it is lost and structure rebuild is not performed. CQS is not able to perform structure recovery because resource structures do not support checkpoint and logging. CQS clients can repopulate the failed resource structure. CQS attempts to allocate a new resource structure.

If a new structure is successfully allocated, CQS drives the client structure event exit with the repopulate structure event. The CQS client or clients must then repopulate the structure. If a new structure is not successfully allocated, CQS drives the structure exit event with the structure failed event. The structure is not accessible for repopulation. Correct the environmental problem that caused the structure allocate to fail so that the structure can be allocated and repopulated.

Initiating structure rebuild with z/OS and CQS

A structure rebuild can be initiated by a z/OS operator, by CQS, or by z/OS.

- A z/OS operator can initiate a structure rebuild to copy or recover queues using the following command:

  ```
  SETXCF START,REBUILD,STRNAME=strname,LOCATION=NORMAL/OTHER
  ```

- CQS initiates a structure rebuild if, during CQS initialization, it detects an empty structure and a valid SRDS (indicating a valid structure checkpoint in the SRDS). If CQS detects an empty structure and a valid SRDS, it also initiates a structure rebuild during event notification facility (ENF) 35 event processing.
- z/OS initiates a structure rebuild if the rebuild threshold for loss of connectivity is reached. The rebuild threshold for loss of connectivity is defined with the CFRM policy REBUILDPERCENT keyword. The REBUILDPERCENT default is 1. If the system programmer does not define REBUILDPERCENT, z/OS initiates a rebuild if any CQS loses connectivity to the structure.
- If structure copy aborts because of a CQS failure and no other CQS can determine if the failed CQS is the master, then the rebuild starts over as a structure recovery.

CQS structure repopulation

Structure repopulation is a process by which CQS clients repopulate a failed resource structure. CQS does not support structure recovery for resource structures because CQS does not log or checkpoint resource updates.

If a resource structure and its duplex fail, the CQS clients can repopulate the resource structure. CQS attempts to allocate a new structure. If this allocation is successful, CQS notifies its clients to repopulate. The CQS client or clients must then repopulate the structure. Any resources that were kept only on the resource structure are lost.

If CQS fails to allocate a new structure, CQS notifies the client that the structure failed. If the sysplex environment changes later and CQS is eventually able to allocate a new resource structure, CQS notifies the client to repopulate at that time. Alternately, correct the environmental problem that caused the structure allocate to fail so that the structure can be allocated and repopulated.

CQS does not coordinate resource structure repopulation between CQS clients; clients must synchronize resource structure repopulation if desired. Structure repopulation does not guarantee the restoration of all objects; some objects might be lost.

CQS structure recovery

The structure recovery function recovers the data objects on a structure from the SRDS and the z/OS logs after a structure failure.

Important: Structure recovery is not supported for resource structures.

After a structure failure, the structure might need to be recovered if it is empty or contains only CQS control information. During structure recovery, CQS allocates a structure and repopulates it from either the SRDS (containing valid client data from a previous checkpoint) and the CQS log or the CQS log by itself.

When CQS recovers the structure from a structure checkpoint, it repopulates the structure with the data objects from the structure recovery data set. CQS reads the log starting at the time of the structure checkpoint to update the structure with changes that occurred after the structure checkpoint.

If the primary structure is empty and neither SRDS contains valid structure checkpoint data, CQS determines whether it can use just the CQS log for recovery. If the first log record in the log stream is the `Beginning of Log` log record, the log stream contains all of the log records required for recovery and CQS can use the log record to complete the structure recovery.

If CQS finds that a previous structure rebuild did not complete successfully, it initiates another rebuild.

If the primary structure contains only CQS control information and the CQS that allocated the structure is not able to determine if a rebuild is necessary, CQS initiates a rebuild if either SRDS is valid or all log records are available.

If neither SRDS is valid and the log records are deleted by a previous structure checkpoint, CQS cannot rebuild the structure. In this case, if rebuild is necessary, CQS issues WTOR CQS0034A to ask you what to do. You can cold start the structure or cancel this CQS.

If no CQS has access to the structure when structure rebuild is initiated, the structure is recovered from the SRDS and the CQS log. Nonrecoverable data objects (such as IMS Fast Path input messages) are lost. Data objects are read from the SRDS and copied into a new structure. CQS then reads the log to bring the structure back to the point of currency. The log contains all the records necessary for structure recovery if no structure checkpoint was ever initiated. In this case, the structure is recovered from just the CQS log.

A client can use the CQSCONN request to specify whether work can be performed while a structure is being rebuilt. While structure recovery is in progress, CQS stops all activity against the structure. This means that CQS requests are held until the structure recovery is complete. You can allow CQS requests to continue during structure rebuild by specifying `WAITRBLD=NO` when connecting to the structure with the CQSCONN request. In this case, structure recovery stops structure activity for some time, but the structure becomes available much sooner.

Related concepts

"CQS structure recovery" on page 174

The structure recovery function recovers the data objects on a structure from the SRDS and the z/OS logs after a structure failure.

CQS structure copy

The structure copy function copies all of the data objects (both recoverable and nonrecoverable) from the structure to a new structure for a planned reconfiguration or unplanned activity such as loss of connectivity.

Structure copy can be used to change the location of the structure or any other attribute defined in the CFRM policy, such as SIZE, INITSIZE, and PREFLIST. When a structure rebuild is initiated, at least one CQS must have access to the structure for structure copy to be performed.

z/OS structure duplexing for CQS

Structure duplexing is an optional z/OS-managed process for failure recovery of queue structures and resource structures. In this process, z/OS creates a duplex copy of a structure in advance of a failure, then maintains the structures in a duplexed state during normal operation.

If a queue structure fails and duplexing is enabled, z/OS switches to the unaffected structure instance. If a queue structure fails and duplexing is not enabled, CQS rebuilds the structure based on data from the most recent checkpoint and z/OS log entries. The advantage of duplexing queue structures in the event of a failure is in avoiding the overhead of a CQS-managed structure rebuild.

If duplexing is enabled and a resource structure fails, z/OS switches to the unaffected structure instance. If duplexing is not enabled and a resource structure fails, the data objects are lost because resource structures do not support checkpoint or logging. CQS repopulates the resource structure with control information. CQS notifies its clients to repopulate the structure. It is up to the clients to repopulate the resource structure if necessary.

Recommendation: Enable structure duplexing for resource structures.

If both instances of a structure fail at the same time, structure duplexing does not work and all data objects are lost. If the failed structure is a resource structure, the CQS client must repopulate it. If the failed structure is a queue structure, CQS recovers the structure using structure rebuild.

Structure duplexing is optional. To use it, you must enable the z/OS duplexing function. Perform the following steps to enable this function:

1. Ensure that the sysplex is defined as duplexing capable.
2. Add the following parameter to your CFRM couple data set format utility: `ITEM NAME(SMDUPLEX) NUMBER(1)`
3. Migrate to an environment in which system-managed duplexing is enabled from a CFRM standpoint. A nondisruptive migration of CFRM couple data sets is required. Only z/OS systems at a level that supports system-managed duplexing are capable of using system-managed CFRM couple data sets that are duplexing-capable. Therefore, take the following steps:
   a) Incrementally migrate all systems in the sysplex that are using CFRM to the z/OS level that supports system-managed duplexing.
   b) Format system-managed duplexing-capable CFRM couple data sets and bring them into use as the primary and alternate CFRM couple data sets for the configuration.

Important: After you enable z/OS duplexing, you cannot return to downlevel CFRM couple data sets (ones that are not system-managed duplexing-capable) without disruption. Doing so requires a sysplex-wide IPL of all systems using the system-managed duplexing-capable data sets.

After an uplevel CFRM couple data set is in use in the sysplex, system-managed duplexing can be started and stopped in a nondisruptive manner. To turn this function on or off, even while the CFRM couple data set is in use, modify the CFRM policy DUPLEX parameter or use the **`SETXCF START/STOP,REBUILD,DUPLEX`** operator command.

To enable system-managed duplexing for a particular structure, the structure must be defined as duplexing-capable. Defining a structure as duplexing capable also defines it as system-managed rebuild-capable. Add the following parameter to your CFRM active policy:

```
DUPLEX (ENABLED)
```

or

```
DUPLEX(ALLOWED)
```

If DUPLEX(ENABLED) is defined in the CFRM active policy, the system programmer or z/OS internally can initiate the duplexing rebuild. z/OS triggers the start of duplexing rebuild based on a timer or upon detection of certain events (such as connect, disconnect, and policy change). When CQS initializes and connects to a structure defined with DUPLEX(ENABLED), z/OS starts a duplexing rebuild.

If DUPLEX(ALLOWED) is defined in the CFRM active policy, the duplexing rebuild must be initialized by the system programmer using the following command:

```
SETXCF START,REBUILD,DUPLEX,STRNAME=strname
```

Important: If you define overflow structures with DUPLEX(ENABLED), IMS initialization allocates the overflow structure and duplexing begins. If IMS initialization determines that the overflow structure is not needed, it deletes it and duplexing terminates. If you want to avoid this unnecessary overhead, during CQS initialization define the overflow structure with DUPLEX(ALLOWED) and initiate duplexing with a SETXCF command when the structure goes into overflow mode.

Once duplexing is established, the structure remains in that state indefinitely. Duplexing can be stopped internally by z/OS if an error occurs (such as link failure, structure failure, and CFRM policy change). The system programmer can explicitly stop duplexing using the following command, where you specify KEEP=OLD to keep the old structure and KEEP=NEW to keep the new structure.

```
SETXCF STOP,REBUILD,DUPLEX,STRNAME=strname,KEEP=OLD/NEW
```

Planned reconfiguration (such as a CFRM policy change or taking a coupling facility offline for maintenance) is supported. Structure rebuild is not permitted for a structure that has established duplexing, so the duplexing must be stopped first. Perform the following steps:

1. Stop duplexing.

   Stop duplexing and switch the structure to simplex mode by issuing the following command:

   ```
   SETXCF STOP,REBUILD,DUPLEX,STRNAME=strname,KEEP=OLD/NEW
   ```

2. Reconfigure.

   Make the change required for planned reconfiguration.

3. Initiate duplexing rebuild.

   Initiate a new duplexing rebuild by issuing the following command:

   ```
   SETXCF START,REBUILD,DUPLEX,STRNAME=strname
   ```

Chapter 13. IMSRSC repository administration

After performing system definition of the IMSRSC repository, you can use it for storing resource (and descriptor) definitions for IMS databases, transactions, programs, and routing codes.

A repository can maintain resource and descriptor definitions for up to 64 IMS systems in an IMSplex.

IMS accesses the repository through the Resource Manager (RM) address space to read or write the stored resource definitions. IMS issues the RM repository read request to read the stored resource definitions during automatic import, **`IMPORT`** command, and **`QUERY`** command processing. IMS issues the RM repository update request to write its runtime resource definitions as the stored resource definitions to the repository during **`EXPORT`** command processing.

IMS issues the RM repository delete request to delete the stored resource definitions from the repository during **`DELETE DEFN`** command processing.

You can use the RDDS to Repository utility (CSLURP10) to copy the contents of a resource definition data set (RDDS) to an IMSRSC repository.

You can use the Repository to RDDS utility (CSLURP20) to generate an RDDS from an IMSRSC repository.

You must perform system definition of the repository before IMS can be started with the repository enabled. Consider setting AUTOEXPORT=AUTO and AUTOEXP_IMSID=THIS_IMS in the DYNAMIC_RESOURCES section of the DFSDFxxx member if your IMS systems are not cloned, or AUTOEXPORT=AUTO and AUTOEXP_IMS=ALL_DEFINED if cloned.

You must perform system definition of the repository before IMS can be started with the repository enabled.

You can use the Repository Server (RS) address space batch interface (FRPBATCH) to perform administration tasks for the repository. You can also use the MODIFY (F) *reposervername* commands for the RS address space to perform some of the repository administration tasks.

The following is an example of some of the repository administration tasks that can be performed with the FRPBATCH interface:

- Add a repository to the RS catalog repository data sets
- Remove a repository from the RS catalog repository data sets
- Change the status of a repository data set pair
- Rename a repository
- Start a repository
- Stop a repository
- Update repository data sets

The following is an example of some of the repository administration tasks that can be performed with the MODIFY (F) *reposervername* commands:

- Change the status of a repository data set pair
- Start a repository
- Stop a repository
- Change audit settings for an RS
- Dynamically refresh in-storage security tables to pick up changes from a security product
- Shut down the RS

Some **`MODIFY (F)`** and **`FRPBATCH`** commands are equivalent.

Table 15. Equivalent Modify (F) and FRPBATCH commands

MODIFY (F)	FRPBATCH	Note
--	ADD	
ADMIN DISPLAY	LIST	
ADMIN START	START	
ADMIN STOP	STOP	Stops the IMSRSC repository
--	RENAME	
--	DELETE	
ADMIN DSCHANGE	DSCHANGE	
--	UPDATE	
AUDIT	--	Changes the audit level
SECURITY	--	Refreshes in-storage profiles
SHUTDOWN	--	Stops the RS. Similar to the STOP command through the z/OS STOP (P) interface.

Related concepts
Overview of the IMSRSC repository (System Definition)
Related tasks
Defining the IMSRSC repository (System Definition)
Related reference
IMPORT DEFN SOURCE(REPO | RDDS) command (Commands)
EXPORT command (Commands)
QUERY commands (Commands)
MODIFY reposervername commands (Commands)
Repository to RDDS utility (CSLURP20) (System Utilities)
RDDS to Repository utility (CSLURP10) (System Utilities)
Related information
Commands for FRPBATCH (System Programming APIs)

Updating IMSRSC repository specifications in the RS catalog repository

You can update the name, auto-open option, security class, and data sets of an IMSRSC repository.

You can change the name of a repository using the **RENAME** FRPBATCH command. You can change any other repository details using the **UPDATE** FRPBATCH command.

Note: You must stop a repository before you can update it.

Here are some implications for performing the various update operations:

- Renaming a repository:

  If you change the name of a repository, the Resource Manager (RM) must be modified to refer to the repository by the new name. To modify the RM to refer to the repository by the new name:

  1. Disable RM from using the repository by issuing the **`UPDATE RM TYPE(REPO) REPOTYPE(IMSRSC) SET(REPO(N))`** command.

2. Ensure that all RMs are disabled from using the repository by issuing the **QUERY RM TYPE(REPO) SHOW(ALL)** command.
3. Modify the CSLRIxxx member of the IMS PROCLIB data set at all RMs to have the new repository name.
4. Enable RM to use the repository by issuing the **UPDATE RM TYPE(REPO) REPO(TYPE(IMSRSC)) SET(REPO(Y))** command.

- Toggling the AUTOOPEN option for the **UPDATE** FRPBATCH command:

  Changing the AUTOOPEN option takes effect the next time the repository is started.
- Changing the security class:

  Specifying SECURITYCLASS(NULL) for the **UPDATE** FRPBATCH command results in any existing value being reset in the repository definition. You can use the **SECURITY** Repository Server (RS) command to refresh the in-storage security settings immediately. Otherwise, the new value takes effect the next time the repository is started.
- Changing the data sets a repository uses:

  You can use the REPDS*n*RID and REPDS*n*RMD parameters for the **UPDATE** FRPBATCH command to change the names of the data sets corresponding to each IMSRSC repository data set pair. This action does not change the status of that repository data set pair in the repository definition. For example, if repository data set pair 3 has a status of COPY1 and you use the **UPDATE** FRPBATCH command to change the repository index data set (RID) and repository member data set (RMD) names, the new data sets are still treated as COPY1 the next time the repository is opened. Therefore, if you decide to use the **UPDATE** FRPBATCH command to manipulate the repository data sets while the repository is offline, you must ensure that the actual state of the data sets is compatible with their state as stored in the RS catalog repository before the next time the repository is started.

  Attention: Updating repository data sets incorrectly can result in loss of data. Retain a backup of any data set you intend to change until you confirm that data recovery is successful.

Related concepts

Overview of the IMSRSC repository (System Definition)

Recovery during the IMSRSC repository data set update process (Operations and Automation)

Related tasks

"Removing an IMSRSC repository from the RS catalog repository" on page 179
To remove an IMSRSC repository from the Repository Server (RS) catalog repository, perform the procedure for falling back from using the repository, and issue the **DELETE** FRPBATCH command.

Related reference

QUERY RM command (Commands)

UPDATE RM command (Commands)

CSLRIxxx member of the IMS PROCLIB data set (System Definition)

UPDATE command for FRPBATCH (System Programming APIs)

RENAME command for FRPBATCH (System Programming APIs)

F reposervername,SECURITY (Commands)

Removing an IMSRSC repository from the RS catalog repository

To remove an IMSRSC repository from the Repository Server (RS) catalog repository, perform the procedure for falling back from using the repository, and issue the **DELETE** FRPBATCH command.

If you want to remove a repository because it has been renamed since it was added to the RS catalog repository, and you want to continue to use the IMS repository function for this IMS, do not perform step "1" on page 179.

1. Perform the procedure for falling back from using the repository.
2. Issue the **DELETE** FRPBATCH command to remove the repository from the RS catalog repository.

Related concepts

Overview of the IMSRSC repository (System Definition)

IMS repository index and member data sets (System Definition)

Related tasks

Falling back from using the IMSRSC repository (System Definition)

Related reference

DELETE command for FRPBATCH (System Programming APIs)

Viewing IMSRSC repository definitions and status

To view information about an IMSRSC repository, use one of the FRPBATCH **`LIST`** commands or the **F** *reposervername*,**`ADMIN`** command.

You can view the following information about repositories that are listed in the Repository Server (RS) catalog repository:

- Names of repositories
- Whether the repositories are started, stopped, or closed
- When repositories were last updated and who updated them

- To view the repository names managed by the RS, use the Modify command **F** *reposervername*,**`ADMIN DISPLAY`**.
- To view details for a specific repository, use the FRPBATCH command **`LIST REPOSITORY`** or the Modify command **F** *reposervername*,**`ADMIN,DISPLAY`**(*repositoryname*).
- To view details for all repositories in the RS catalog repository, use the FRPBATCH command **`LIST STATUS`** or the Modify command **F** *reposervername*,**`ADMIN`**.

Related concepts

Overview of the IMSRSC repository (System Definition)

Related reference

F reposervername,ADMIN (Commands)

LIST command for FRPBATCH (System Programming APIs)

IMS initialization with the IMSRSC repository

During IMS initialization, IMS reads the REPOSITORY section of the DFSDFxxx member of the IMS PROCLIB data set.

If IMS is not enabled with DRD (MODBLKS=OLC is specified in the Common Service Layer section of the DFSDFxxx member or in the DFSCGxxx member), or IMS is defined with RMENV=N (IMS is not using RM services), then the REPOSITORY section is ignored. A DFS4402W message is issued, indicating that the REPOSITORY section is ignored.

If TYPE=IMSRSC is specified in the CSLRIxxx member, IMS tries to connect to the Resource Manager (RM) for the IMSRSC repository. If RM is defined to manage the repository type that IMS requests, a successful return code is returned for the RM repository connect request. When IMS successfully connects to the RM for the repository, a DFS4404I message is issued.

If RM is not enabled for the repository type IMS is requesting, an error return code is returned for the RM repository connect request. IMS issues message DFS4400E and IMS initialization terminates with abend code 0400. Any other error return and reason code from RM results in a DFS4401E message with the RM return and reason code. After the DFS4401E error message, IMS initialization terminates with abend code 0400.

If AUTOIMPORT=AUTO is specified and the REPOSITORY section is defined in the DFSDFxxx member, the DFS3409I message is issued during IMS initialization instead of DFS3499I, because IMS is not yet connected to the repository. The DFS3499I message is issued during coldstart, indicating where the resource definitions are imported from.

If IMS is enabled with the repository, AUTOIMPORT=AUTO is specified in the DFSDFxxx member, and the repository contains the stored resource definitions for IMS, the DFS3399I message is issued, indicating that the automatic import is from the repository.

If the repository does not contain any stored resource definitions for the IMS, the DFS3399I message is issued, specifying that either the resource definition data set (RDDS) or the MODBLKS data set is being used.

If the repository is empty, the DFS4405W message is issued, indicating that no resources are imported from the repository.

If AUTOIMPORT=REPO or RDDS or MODBLKS is specified, the DFS3499I message is issued during IMS initialization.

When the stored resource definitions are imported from the repository during cold start or IMPORT command processing, the DEFNTYPE is set to IMPORT.

Related concepts

Overview of the IMSRSC repository (System Definition)

DFS messages (Messages and Codes)

Related reference

DFSDFxxx member of the IMS PROCLIB data set (System Definition)

DFSCGxxx member of the IMS PROCLIB data set (System Definition)

CSL RM management of the IMSRSC repository

The Common Service Layer (CSL) Resource Manager (RM) interacts with the Repository Server (RS) address space to manage the stored resource definitions in the IMSRSC repository.

RM registers to the Repository Server and connects to the IMSRSC repository during RM initialization or during the UPDATE RM command processing. If RM is defined with the resource structure, RM uses the resource structure to save the repository information, such as repository name, repository type, audit access values, and the RS z/OS cross-system coupling facility (XCF) group information. The resource structure ensures that all RMs in the IMSplex use the same repository.

You must perform system definition of the repository before RM can be started with the repository enabled. In particular, RM repository parameters are defined in the CSLRIxxx member of the IMS PROCLIB data set.

Use the UPDATE RM command to dynamically enable RM to use the repository.

Use the QUERY RM command to obtain information about the status of repositories that are being managed by the RM address space.

Related concepts

Overview of the IMSRSC repository (System Definition)

"Considerations for setting security for the IMSRSC repository in the CSL RM address space" on page 326
The Common Service Layer (CSL) Resource Manager (RM) address space is an authorized caller of the IMSRSC repository services. You can specify if the RM address space is authorized to access a repository by defining a profile with FRPREP.*repository_name*.

Related tasks

Defining the IMSRSC repository (System Definition)

Related reference

"Information managed by the CSL RM" on page 157
You can use the Resource Manager (RM) to create, update, query, or delete resources in the resource structure.

QUERY RM command (Commands)

UPDATE RM command (Commands)

CSL RM initialization with the IMSRSC repository

During Resource Manager (RM) initialization, RM reads the CSLRIxxx member of the IMS PROCLIB data set and processes the REPOSITORY section.

The CSLRIxxx member can be defined with the REPOSITORY section to define the IMSRSC repository name and repository type to be managed by the RM system. If the REPOSITORY section is defined, RM during RM initialization automatically registers to the Repository Server (RS) address space. RM connects to the repository with the name defined in the CSLRIxxx member.

Any errors during RM initialization with the RS REGISTER or CONNECT requests result in CSL2510E or CSL2511E error messages with the return or reason code from the RS. Additionally, the CSL2502A or CSL2503A messages are issued. RM does not complete its initialization and it does not issue the CSLSCRDY request to SCI indicating that it is ready to accept work. If all RMs in the IMSplex are waiting for the RS or repository, any IMS control region being initialized and using RM services waits for RM with the DFS3306A message. Any active IMS issues the DFS3306A message, indicating that RM services are not available.

The CSL2510E message during RM initialization is followed by the CSL2502A message, indicating that RM is waiting for the RS address space to be started. RM tries to register with the RS every 5 seconds until it is successful in registering with RS or it is terminated. RM must be terminated using the CANCEL command. RM cannot be terminated using the SCI SHUTDOWN request, because RM is not registered to SCI. RM cannot be terminated using the MVS STOP command, because the repository REGISTER and CONNECT is in early RM initialization and the threads to process the MVS STOP command are not yet set up. When RM is successful in registering to RS, the CSL2502A highlighted message is deleted.

The CSL2511E message during RM initialization is followed by the CSL2503A message, indicating that RM is waiting to connect to the repository with name and type specified in the CSL2503A message. RM retries to connect to the repository every 5 seconds until it is successful in connecting to the repository or it is terminated. RM must be terminated using the CANCEL command. RM cannot be terminated using the SCI SHUTDOWN request, because RM is not registered to SCI. RM cannot be terminated using the MVS STOP command, because the repository REGISTER and CONNECT is in early RM initialization and the threads to process the MVS STOP command are not yet set up. When RM is successful in connecting to the repository, the CSL2503A highlighted message is deleted.

Any errors parsing the CSLRIxxx member repository section result in the CSL25xx error messages. If this is the first RM being started in the IMSplex (when RM is with or without a resource structure), RM comes up without the repository enabled. After the error in the CSLRIxxx member is fixed, RM can be enabled with the repository using the UPDATE RM command.

If RM is being started with the resource structure, and the IMSplex is enabled with the repository, then any repository name errors result in the CSL25xx error messages followed by RM connecting to the repository with the name that is read from the resource structure.

When RM is started, if there is a mismatch in the XCFGROUP name in the CSLRIxxx member and the resource structure, RM abends with abend code 0010 and subcode X'00000634'.

When RM is started, if RM initialization modules cannot get storage, RM abends with abend code 0010 and subcode X'00000635' or X'00000636'.

The first RM to connect to the repository with the CONNECT request defines the index and key fields to be used in the repository. The subsequent RMs will use the index and key field information in the repository for their use. When RM connects successfully to the repository, the CSL2500I message is issued. Additionally, if the RM initializes the repository key and index fields and the repository is empty, the CSL2501I message is issued.

If RM is using the resource structure, the first RM stores the repository name, repository type, and audit access information from its CSLRIxxx member in the resource structure. Subsequent RMs obtain the repository name, repository type, and audit access information from the resource structure. Any mismatch in the information in the resource structure to the values in the RM's CSLRIxxx member results in the CSL2512W message, indicating the mismatch. RM connects to the repository name and type obtained from the resource structure.

If RM cannot connect to the resource structure the CSL2515E message is issued with the error return and reason code. RM continues processing without the repository services.

You can dynamically enable the repository using the UPDATE RM command after the errors are resolved.

If RM is not using the resource structure, RM always connects to the repository with the name specified in the CSLRIxxx member.

In an environment where RM is not using the resource structure, do not modify the repository name in the CSLRIxxx member between RM restarts.

Related concepts

Overview of the IMSRSC repository (System Definition)

DFS messages (Messages and Codes)

CSL SCI requests (System Programming APIs)

Related tasks

Enabling dynamic definition for IMS resource groups (System Definition)

Related reference

UPDATE RM command (Commands)

CSLRIxxx member of the IMS PROCLIB data set (System Definition)

CSL messages (Messages and Codes)

RM abend codes (Messages and Codes)

CSL RM, IMS, and Repository Server termination

When either IMS or the Repository Server (RS) terminates, the Resource Manager (RM) cancels any uncommitted units of work in progress. When RM terminates, it disconnects from any repository RM is managing and deregisters from the RS address space.

RM listens for any normal or abnormal IMS termination using the Structured Call Interface (SCI) notify exit. Any work in progress can be in an indoubt state. RM cancels any uncommitted units of work in progress for the IMS that terminates. If the error occurs after RM commits the unit of work, but before responding to IMS, the unit of work is committed.

If the RS terminates while work is in progress, the RM Register exit is driven with the RS termination event. Any work in progress can result in an indoubt state. If the work in progress is not committed, it is canceled by the RS. If the RS terminates after the COMMIT, but before responding to RM, the unit of work is committed.

You must determine the status of the work in progress to resolve the indoubts by using the **`QUERY SHOW(DEFN)`** commands. The create or update time stamps in the IMSRSC repository or import time stamps in IMS can be used to determine the status of the work in progress. The **`EXPORT`**, **`IMPORT`**, or **`DELETE DEFN`** command must be reissued if the unit of work is not committed.

RM termination disconnects from any repository RM is managing and deregisters from the RS address space.

Related concepts

Overview of the IMSRSC repository (System Definition)

Related tasks

Shutting down the CSL RM (Operations and Automation)

Related reference

IMPORT DEFN SOURCE(REPO | RDDS) command (Commands)

EXPORT command (Commands)

QUERY commands (Commands)

Chapter 14. IMS service considerations

Proper planning for implementing service to IMS is a vital part of keeping IMS available and resilient.

Types of service SMP/E SYSMODs

IMS provides maintenance packaged in SMP/E format.

IMS maintenance is packaged as one of three types of SMP/E SYSMODs:

- Program Temporary Fixes (PTFs)

  Program Temporary Fixes (PTFs) are normally considered preventative service. PTFs contain solutions for problems and are distributed in machine-readable format. The PTF is considered the final solution for a problem for the release of IMS for which it is provided.

  For modules that supersede a previous level of a module, the source changes are the cumulative delta source changes for the module. If a PTF has a prerequisite, the source changes included in the PTF are not cumulative, but reflect only the code changed for the PTF.
- Authorized Program Analysis Reports (APARs)

  Authorized Program Analysis Report (APAR) fixes are considered corrective service. APARs contain solutions for problems and are distributed in machine-readable format. The APAR is considered an interim solution, or temporary solution, for a problem. The final solution is the corresponding PTF or PTFs created at the end of the APAR process. One APAR can become one or more PTFs.
- USERMODs

  IMS provides USERMODS in the following situations:

  – As an APAR fixtest, to ensure that the problem reported by an APAR is corrected or to provide relief until the APAR or PTF is available

  – As a circumvention to a problem, to provide relief until the final APAR or PTF is available

  – As a trap (or specialized code) to obtain additional documentation or information (such as a dump) necessary to analyze and understand a problem

Related reading: For more detailed SMP/E information, see *SMP/E Reference*.

Service SYSMOD packaging

The IMS service process typically makes APARs available as soon as they are completed, which is normally a few weeks before the corresponding PTF or PTFs are completed. In situations in which a fix is urgently needed after the APAR is completed, but before the PTF is available, using the APAR might be the best short-term solution.

APARs provided by IMS define as prerequisites (PRE, IFREQ, and so on) only those SYSMODS for which the APAR has code dependencies. IMS APARs list the corresponding APARs, not PTFs, as prerequisites. Whenever IMS APARs are processed by SMP/E, regression messages might be encountered. These messages must be analyzed to ensure that no regression is actually taking place.

PTFs contain as requisites (PRE, IFREQ, and so on) all prior PTFs affecting the same elements. Processing a PTF might require the processing of many additional SYSMODS, while processing an APAR might not. In emergency situations where a problem exists and a solution must be implemented quickly, the APAR might be the best short-term solution as it might require the least amount of change. However, you must always use the PTF as the final fix.

When processing APARs, encountering regression messages from SMP/E is normal. These messages must be analyzed to ensure that no regression will occur. If needed, contact IBM Software Support for assistance.

PTFs supersede (SUP) their corresponding APARs. Therefore, removing the APAR prior to processing the PTF is not required.

Important: APAR and USERMOD fixes should not be processed using the SMP/E ACCEPT command. The corresponding PTF or PTFs should be processed as the final fix

USERMODs provided by IMS define as prerequisites (PRE, IFREQ, and so on) only those SYSMODs for which the USERMOD has code dependencies. USERMODs list the corresponding APARs, not PTFs, as prerequisites. In this way, USERMODS are like APARs. Whenever IMS USERMODS are processed by SMP/E, regression messages might be encountered. These messages must be analyzed to ensure that no regression is actually taking place. If needed, contact IBM Software Support for assistance.

USERMODs provided by IMS are not superseded (SUP) by a corresponding APAR or PTF. When the final fix is available, you must RESTORE the USERMOD from the system. The ++HOLD information provided with each USERMOD indicates that you must RESTORE the USERMOD and contains instructions on how to do so.

Obtaining IMS service

The recommended way to obtain IMS service is to order a Recommended Service Upgrade (RSU), which includes all current PTFs that have passed the Consolidated Service Test (CST).

CST is an additional service testing environment for the z/OS platform to test all the current PTFs for participating products on the z/OS platform together so that IBM can recommend PTF service for z/OS and key subsystems together in one RSU.

To order your IMS service, go to ShopzSeries at Shopz.

One of the ShopzSeries options is to create a custom order that is based on the current contents of your SMP/E Consolidated Software Inventory (CSI). If you use this option, ShopzSeries requires information on existing features, FMIDs, and PTFs in your environment. This information can be generated by using the SMP/E GIMXSID service routine. The GIMXSID service examines the specified target zones and global zone within the specified SMP/E CSI to determine which functions and service you already have. The ShopzSeries web site provides detailed instructions, sample JCLs, and video clips that guide you through the service ordering process.

You can also continue to obtain PTFs from the following channels. However, keep in mind that these PTFs may not have gone through the CST process.

IBM Software Support
: You can request specific PTFs that can be downloaded from IBMLINK, a File Transfer Protocol (FTP) site, or mailed on a cartridge.

Extended Support Offering (ESO)
: ESO tapes are available to licensed users upon request.

Custom Built Product Delivery Offering (CBPDO)
: CBPDO service tapes are created upon customer request.

ServerPac
: ServerPac orders are sent upon request. In addition to products, ServerPac orders also contain PTFs that have been incorporated in the products.

ShopzSeries
: Web-based productivity tool that makes it easier for you to order service. Service is sent to you either through the mail or through the Internet.

Maintenance recommendations

The recommendations outlined in the following topics can help you develop a maintenance strategy that works in your environment.

Assessing your readiness to install maintenance

Before you install any maintenance, or install a new IMS system, for that matter, you need to determine your readiness. This involves a careful risk assessment. This same principle applies to developing a maintenance strategy.

Numerous factors are involved in assessing your readiness to install maintenance, including the quality of the local test environment and the business cycle.

- Quality of the local test environment

  Several factors affect the quality of the test environment. Some questions to consider are these:

  - Are the software products such as RACF, DFP, or z/OS DFSMS used in your test environment in the same way they are used in production and do the levels match what is used in production?
  - Do you have a tool, such as Teleprocessing Network Simulator (TPNS), that enables you to perform stress tests to simulate peek production activity?
  - To what extent does the hardware used in your test environment match the production environment?
  - To what extent does the application software used in your test environment match the production environment?
  - Are test results closely monitored?

  These and other factors need to be considered to evaluate the quality of the test environment.
- Business cycle

  You must do everything you can during a critical business cycle, to ensure that IMS remains available. Therefore, avoid implementing maintenance to a production system during a critical business cycle.

Updating the maintenance service level of a new or migrated IMS system

After installing an IMS system, whether it is a new system or a new release for an existing IMS system, you need to bring the service level of the new system to an acceptable level of currency by installing service levels and PTFs, resolving PE PTFs, and conducting a test cycle..

Attention: SYSMODs in APPLY-only status can be regressed by an IMS system definition. See "Preventing regression of SYSMODs in APPLY-only status by an IMS system definition" on page 195 for instructions about preventing this.

Note: You can use the **`/DIAGNOSE SNAP MODULE(modulename)`** command to determine your current maintenance level for a specific module.

If you are starting with a base implementation service level of an IMS production system, the following actions are recommended:

1. Six months before you plan to implement the IMS system in production, install all available PTFs.
2. Three months before you plan to implement the IMS system in production, install all PTF fixes for HIPER APARs.
3. Resolve PEs.

   Evaluate any PE PTFs that are installed in your IMS system to determine if they impact your IMS environment. If they do, either install the fix for the PE PTF, remove the PE PTF, or contact IBM Software Support for assistance.
4. A 3-month test cycle is recommended before you implement the IMS system in production

Because IMS maintenance continues to be distributed both during and after the test cycle, incorporate ongoing procedures for maintaining the service levels of your production systems as soon as possible in your new system. For an example of the procedures you can use, see "Obtaining and installing maintenance" on page 188.

Maintaining service levels of existing IMS production systems

After your IMS system is installed, tested, and operational, maintain the service level of the IMS system by following the recommendations listed below.

- Recommendations for all products that run on the z/OS platform:
  - Upgrade RSU maintenance 4 times per year
  - Monitor HIPER, PE, SEC/INT and FIXCATs regularly and install critical fixes as appropriate weekly or monthly
- Recommendations specific to IMS production systems:
  - Install fully tested fixes for significant[1] software problems encountered
  - Install fully tested significant HIPER SYSMODs that are directly applicable to your specific IMS environment
  - Install fully tested significant SYSMODs that resolve PE PTF's that are directly applicable to your specific IMS environment
  - Review the IMS PSP bucket UPGRADE for the IMS release level and SUBSET FMIDs. Important IMS product-related information is continually added to these buckets that might require action

Obtaining and installing maintenance

The following procedure is an example that you can use as a model for implementing a maintenance procedure that is specific to your IMS installation.

This maintenance procedure is based on the recommendations that are outlined in "Updating the maintenance service level of a new or migrated IMS system" on page 187 and "Maintaining service levels of existing IMS production systems" on page 188.

For a similar example procedure with additional information, see Informational APAR II13024.

Attention: SYSMODs that are applied (APPLY) but not accepted (ACCEPT) might be regressed by the system definition process. To avoid regressing SYSMODs, see "Preventing regression of SYSMODs in APPLY-only status by an IMS system definition" on page 195.

1. Obtain current service by using your current service delivery method, such as Shopz.

   Shopz is an IBM web-based productivity tool that simplifies the ordering of System z software products, product upgrades, and system maintenance. For more information, see Shopz.
2. Obtain and RECEIVE current enhanced HOLDDATA for z/OS.

   For the most current and complete information, see Enhanced HOLDDATA for z/OS.
3. Use SMP/E to install the service.
   - Specify SOURCEID(RSU*yymm*) to include service that passed Consolidated Service Test (CST). For information, see Consolidated Service Test and the RSU.
   - Evaluate any PE PTFs that are installed in your IMS system that do not have the corresponding fix installed to determine whether the PE PTFs affect your IMS environment. If so, install the fix for the PE PTF, remove the PE PTF, or contact IBM support for assistance.
   - Resolve any system HOLDs in PTFs that are being processed.
4. Obtain information from the z Systems Security Portal. To learn more, see IBM Z and IBM® LinuxONE Security Portal Frequently Asked Questions.
5. Obtain and RECEIVE current Enhanced HOLDDATA for z/OS again to ensure that you address any HIPER APAR and PE PTF exposures that changed since you last checked.

[1] Your shop determines the significance of a fix. If your shop cannot tolerate the possible consequences of not installing a fix, the fix is significant.

6. Run the SMP/E REPORT ERRSYSMOD command to identify HIPER/PE/SEC exposure. Analyze the output and take the following actions as necessary.
   a) Obtain applicable SYSMODS
   b) Process SYSMODs that are applicable to your environment.

      Use IMS Support website or PSP Buckets for APAR descriptions.

      Courses of action for PE SYSMODs already on the system can include:

      - Removing the PTF in error if it is not already ACCEPTed.
      - Leaving the PTF in place if the reported PE symptom is not significant.
      - Instituting procedures to avoid encountering the exposure until the exposure is resolved.
      - Installing corrective the APAR/PTF fix, if one is available.
      - Requesting a FIXTEST from IBM software support for the reported problem, if the APAR is still OPEN.
      - Requesting a USERMOD code bypass for the reported problem from IBM software support.
      - Contacting IBM Software Support for assistance as needed.
7. Review the IMS PSP buckets for new service information.
8. Test the new maintenance level.
9. Implement the new maintenance level.

Repeat steps 4, 5, 6, 7, and 8 from the preceding procedure in an ongoing basis to keep your service level current. At a minimum, perform these steps immediately before you migrate an IMS system into a new environment, such as a test, development, or production environment.

Installing IMS service on a single system

IMS service can be installed in several ways, including SMP/E methods.

The SMP/E methods to install service include:

- RECEIVE/APPLY/ACCEPT (standard sequence)
- ACCEPT without APPLY (pregeneration mode)
- ACCEPT before APPLY (SYSDEF-sensitive service)

Important:

- Do not ACCEPT APARs or USERMODs.
- Before the installing service, ensure that you have read the latest HOLDDATA information.

If you have any questions about these processes, contact IBM Software Support before you begin.

Related tasks

“Installing IMS service in an IMSplex” on page 194
Installing IMS service in an IMSplex is similar to installing service on a single system.

RECEIVE/APPLY/ACCEPT (standard sequence)

This SMP/E method is the standard and recommended method for processing service.

1. Back up the IMS environment.
   a) Back up the SMP/E data sets (such as Zones, SMPMTS, and SMPPTS).
   b) Back up IMS product data sets (such as SDFSRESL and ADFSLOAD).
2. Obtain the desired service.
3. Read the documentation accompanying the package:
   - ESO tape documentation
   - CBPDO Memo to Users Extensions

- Preventative Service Planning (PSP)

4. Run the SMP/E **RECEIVE** command with both the parameters SYSMODS and HOLDDATA set.
5. Run the SMP/E **APPLY CHECK GROUPEXTEND** command.

   SMP/E Messages GIM43401 and GIM44402 can be received for modules not included in the target system during the APPLY CHECK and APPLY process. You can ignore these messages if they refer to a part that pertains to an IMS function or feature that you are not going to use. Programming exceptions (PEs) need to be resolved to ensure that service is processed to the desired level. If needed, contact IBM Software Support for assistance.

   Attention: IMS service frequently includes in-line JCLIN information. For this type of service, SMP/E does not recommend the re-APPLY of service using the REDO parameter. If REDO is used for this type of service (without NOJCLIN), SMP/E RESTORE processing might not work properly.
6. Research the APPLY CHECK reports, making changes as needed.
7. Run the SMP/E **APPLY GROUPEXTEND** command.
8. Test the corrective service.

   If an IMS system definition is done with service in APPLY only status, that service might be partially or completely regressed.

   Recommendation: For all SYSMODs in APPLY only status, issue the following SMP/E command after every IMS system definition, where *xxxx,xxxx* is a list of all SYSMODs in APPLY only status (separated by commas or spaces):

   ```
   APPLY S(xxxxx,xxxxx) REDO NOJCLIN BYPASS (....)
   ```

   Attention: Special handling might be required for SYSMODs in APPLY-only status that have holds that require SYSGEN or hold for DELETE.
9. Run the SMP/E **ACCEPT CHECK GROUPEXTEND** command.
10. Research the ACCEPT CHECK reports.
11. Run the SMP/E **ACCEPT GROUPEXTEND** command.
12. Download a new copy of Enhanced Holddata from the **RECEIVE** command.
13. Run the SMP/E **REPORT ERRSYSMODS** command.
14. Analyze the output from the SMP/E **REPORT ERRSYSMODS** command and process additional services as appropriate.

ACCEPT without APPLY (pregeneration mode)

This SMP/E method is the pregeneration method for processing service.

Important: This information is accurate as of its printing. For the most current and more detailed information, see Information APAR II13024.

This procedure requires that ACCJCLIN was set in the distribution zone when the FMIDs were ACCEPTed.

1. Back up the IMS environment.
   a) Back up the SMP/E data sets (such as Zones, SMPMTS, and SMPPTS).
   b) Back up IMS product data sets (such as SDFSRESL and ADFSLOAD).
2. Obtain the desired service.
3. Read the documentation accompanying the package:
   - ESO tape documentation
   - CBPDO Memo to Users Extensions
   - Preventative Service Planning (PSP)
4. Run the SMP/E **RECEIVE** command with both the parameters SYSMODS and HOLDDATA set.

5. Run either the SMP/E **RESTORE** or **ACCEPT** commands for all outstanding service for all products present in the IMS distribution and IMS target zones. Use the following sample SMP/E control statements to identify this outstanding service (SYSMODS that have been applied only):

```
//SMPCNTL DD *
          SET BDY(targetzonename).
          LIST APAR PTF USERMOD NOACCEPT NOSUP.
```

6. Unload the target zone DDDEFs using the SMP/E **UNLOAD** command. Use the following sample SMP/E control statements and JCL to complete this task:

```
//SMPCNTL DD *
          SET BDY(targetzonename).
          UNLOAD DDDEF.
        //SMPPUNCH DD DSN=IMS.SMPUNLD,DISP=(,CATLG),
        // UNIT=SYSDA,SPACE=(TRK,(5,1),RLSE),
        // DCB=(RECFM=FB,BLKSIZE=16000,LRECL=80)
```

7. Run the SMP/E **LIST** command on the target zone to determine the name of the OPTIONS entry. This OPTIONS entry will be used in Step "10" on page 191. Use the following sample SMP/E control statements to complete this task:

```
//SMPCNTL DD *
          SET BDY(targetzonename).
          LIST TARGETZONE.
```

8. Scratch and reallocate the following data sets:

   - SMPMTS
   - SMPSTS
   - SMPSCDS
   - SMPLTS

   Note: The SMP/E **CLEANUP** command can be used instead of scratching and reallocating the SMPPTS, SMPSTS and SMPSCDS data sets. The SMP/E **CLEANUP** command cannot be used for the SMPLTS data set. You need to scratch and reallocate this data set.

   Use the following sample SMP/E control statements to complete this task:

```
//SMPCNTL DD *
         SET BDY(targetzonename).
         CLEANUP.
```

9. Delete the SMP/E target zone.

   a) Run the SMP/E **ZONEDELETE** command for the Target zone.

   Use the following sample SMP/E control statements to complete this task:

```
//SMPCNTL DD *
          SET BDY(targetzonename).
          ZDEL TZONE(targetzonename).
```

   b) If no other SMP/E zones are in the target CSI (the VSAM cluster), run the IDCAMS **DELETE** and **DEFINE** commands on the target CSI to improve performance.

   Attention: If multiple zones are contained in the same CSI as the target zone, do not delete and redefine the cluster because you will also lose the information for those zones.

10. Re-initialize the new Target zone.

   a) Run the IDCAMS **REPRO** command to copy SYS1.MACLIB(GIMZPOOL) into the new CSI.

   Attention: If you did not delete and redefine the target CSI as described in Step "9" on page 191, do not copy GIMZPOOL into the new target zone.

   b) Rebuild the relationship between the old DLIB zone and the new Target zone. Use the following sample SMP/E control statements to complete this task:

```
//SMPCNTL DD *
        SET BDY(GLOBAL).
          UCLIN.
            ADD GZONE ZONEINDEX(
            (targetzonename,target.zone.cluster.name,TARGET)
                                ).
          ENDUCL.

         SET BDY(targetzonename).
           UCLIN.
             ADD TARGETZONE(targetzonename)
                 SREL(P115)
                 RELATED(dlibzonename)
                 OPTIONS(xxxxxx).
           ENDUCL.
```

Note: Be sure that this new target points to the correct OPTIONS entry. The correct OPTIONS entry can be determined from the output created in step "7" on page 191.

c) Run UCLIN to add the DDDEFs back to the target zone.

This step uses the data set created in step "6" on page 191 as input. Use the following SMP/E control statements and JCL to complete this task:

```
//SMPCNTL DD *
  SET BDY(targetzonename).
//        DD DSN=IMS.SMPUNLD,DISP=SHR
```

Note: Return code 4 is expected in this step because DDDEFs are being added instead of being replaced.

Attention: Before processing SMP/E in step "11" on page 192, RECEIVE the current Enhanced HOLDDATA. Doing so enables you to resolve PEs during SMP/E processing. You can get the most recent HOLDDATA at https://public.dhe.ibm.com/s390/holddata/.

11. Run the SMP/E **ACCEPT GROUPEXTEND BYPASS(APPLYCHECK)** commands for the PTFs to be processed.

    Use the following sample SMP/E control statements to complete this task:

```
//SMPCNTL DD *
        SET BDY(dlibzonename).
           ACCEPT GROUPEXTEND
               BYPASS(APPLYCHECK
                      HOLDCLASS(ERREL,UCLREL)
                      HOLDSYSTEM
                      )
                 SOURCEID (SMCREC,RSU08*,RSU090*,RSU0910,etc)
                 PTFS.
```

Attention: After SMP/E processing is complete, using the current enhanced HOLDDATA, run the SMP/E **REPORT ERRSYSMODS** command to identify missing HIPERs and PE exposures.

12. Run the SMP/E **ZONEMERGE** command specifying CONTENT to merge the distribution zone to the new target zone.

    Use the following sample SMP/E control statements to complete this task:

```
//SMPCNTL DD *
       SET BDY(targetzonename).
           ZONEMERGE(dlibzonename)
               INTO(targetzonename)
               CONTENT.
```

13. Run the SMP/E **GENERATE** command to create the JCL necessary to rebuild the target libraries.

    Tip: This step requires that ACCJCLIN was set in the distribution zone before the IMS FMIDs were ACCEPTed.

Use the following sample SMP/E control statements and additional JCL to complete this task, where the data set for DD name CNTL must have a member named **J**, which contains a sample JOB card:

```
//CNTL      DD  DSN=yourpds,DISP=SHR
       //SMPPUNCH DD  DSN=IMS.GENERATE,
       //             DISP=(,CATLG),UNIT=SYSDA,
       //             SPACE=(CYL,(25,5),RLSE),
       //             DCB=(RECFM=FB,LRECL=80,BLKSIZE=16000)
        SET BDY(targetzonename).
        GENERATE JOBCARD(CNTL,J) REPLACE.
```

14. Run the JCL that was created in step "13" on page 192.

    Note: The SMPLTS job will complete with a return code of 4 because of unresolved external references (IEW2454W). All other jobs should complete with a return code of 0.
15. Run an IMS ALL type of system definition (STAGE 1 and STAGE 2).

    Attention: Ensure that Stage 2 processing is complete before performing step "16" on page 193.
16. Run SMP/E JCLIN, pointing to the STAGE 2 JCL as input.
17. Run the SMP/E **APPLY** command for any IMS service that was not accepted. This service was identified in step "5" on page 191.
18. Run the SMP/E **APPLY** command for any service for other products that was not accepted. This service was identified in step "5" on page 191.
19. Download a new copy of Enhanced Holddata from the **RECEIVE** command.
20. Run the SMP/E **REPORT ERRSYSMODS** command.
21. Analyze the output from the SMP/E **REPORT ERRSYSMODS** command and process additional services as appropriate.
22. Test the new system.

ACCEPT before APPLY (for service that requires a system definition)

This method is a variation of pregeneration mode that can be useful when you have many products sharing the same SMP/E zones and you need to install a PTF that would normally require an ACCEPT BYPASS(APPLYCHECK) sequence (typically a PTF that affects system definition).

This method avoids disturbing other products that have outstanding service (service that has been APPLIED but not ACCEPTed).

1. Back up the IMS environment.
   a) Back up the SMP/E data sets (such as Zones, SMPMTS, and SMPPTS).
   b) Back up IMS product data sets (such as SDFSRESL and ADFSLOAD).
2. Obtain the desired service.
3. Read the documentation accompanying the package:
   - ESO tape documentation
   - Preventative Service Planning (PSP)
4. Run the SMP/E **RECEIVE** command with both the parameters SYSMODS and HOLDDATA set.
5. Run the SMP/E **ACCEPT** or **RESTORE** commands on outstanding APPLY service for all products sharing the SMP/E zones with IMS.
6. Run the SMP/E **ACCEPT CHECK GROUPEXTEND BYPASS(APPLYCHECK)** command.
7. Research the ACCEPT CHECK reports, making changes as necessary.
8. Run the SMP/E **ACCEPT GROUPEXTEND BYPASS (APPLYCHECK)** command.
9. Run an IMS ALL type of system definition STAGE 1 and STAGE 2.
10. Run SMP/E JCLIN pointing to the STAGE 2 JCL as input.
11. Run SMP/E APPLY CHECK GROUPEXTEND.

⚠ **Attention:** Do not use the REDO parameter.

12. Research the APPLY CHECK reports, making changes as necessary.
13. Run SMP/E APPLY GROUPEXTEND.

⚠ **Attention:** Do not use the REDO parameter.

14. Download a new copy of Enhanced Holddata from the **RECEIVE** command.
15. Run the SMP/E **REPORT ERRSYSMODS** command.
16. Analyze the output from the SMP/E **REPORT ERRSYSMODS** command and process additional services as appropriate.
17. Test the new system.

Installing IMS service in an IMSplex

Installing IMS service in an IMSplex is similar to installing service on a single system.

The following considerations apply when planning to install service into an IMSplex:

- Multiple Resource Managers (RMs) and Operations Managers (OMs) can run simultaneously in an IMSplex.

  If all your IMS systems on one LPAR, one OM must be running at all times to provide OM services.
- Only one Structured Call Interface (SCI) can run at any given time in an IMSplex on a single logical partition (LPAR).
- All Common Queue Server (CQS) clients connected to a CQS address space must be stopped before shutting down that CQS.

Recommendations:

- For continuous availability, do not install service to the entire IMSplex at the same time.
- If you are running multiple IMS systems on one logical partition (LPAR), treat each IMS system within the IMSplex as a separate system, bringing each down individually, implementing service, testing it, and bringing it back online. After each system has successfully been brought back online, rotate to the next system, repeating the process.

  Apply service to less complex IMS systems before applying service to more complex IMS systems.
- If you are running multiple LPARs, install service on one LPAR at a time.

  Apply service to less complex LPARs before applying service to more complex LPARs.
- If your IMSplex includes IMS Connect, ensure that all appropriate IMS systems are available before starting IMS Connect. Otherwise, IMS Connect issues a "datastore unavailable" message.

Related tasks

"Installing IMS service on a single system" on page 189

IMS service can be installed in several ways, including SMP/E methods.

Common installation and maintenance issues

You can have a more stable IMS environment by being aware of some of the common installation and maintenance issues that are presented in these topics and taking appropriate action.

Preventing regression of SYSMODs in APPLY-only status by an IMS system definition

Maintenance might be regressed if an IMS system definition is performed when maintenance is in APPLY-only status.

You can use one or the other of the following methods to prevent SYSMODs in APPLY-only status from being regressed by an IMS system definition:

1. Before the system definition, ACCEPT all PTFs in APPLY-only status.
2. Perform the system definition, allowing the SYSMODs in APPLY-only status to be regressed, and then reprocess the SYSMODs.

- Identify the SYSMODs in APPLY-only status by using the following SMP/E statements to list the SYSMODs in APPLY-only status:

```
    SET BOUNDARY (targetzone).
    LIST APAR PTF USERMOD NOACCEPT NOSUP.
```

- Method 1:

  a) ACCEPT all PTFs in APPLY-only status before system definition.

  b) Restore all APARs and USERMODs.

  c) Run stage-1 and stage-2 system definition.

  d) Reapply APARs and USERMODs, if needed.

- Method 2:

  a) Run stage-1 and stage-2 system definition.

  b) Reprocess (APPLY) the SYSMODs that you identified previously. The following example SMP/E statements can be used to reprocess SYSMODs in APPLY-only status:

  Attention: Use the NOJCLIN parameter only to process REDO. Otherwise, you might not be able to RESTORE the service.

```
     APPLY REDO NOJCLIN SELECT(
                               xxxxxxxx
                               xxxxxxxx
                               xxxxxxxx
                                      )    .
```

  In the preceding example, the *xxxxxxx* fields represent the list of each SYSMOD that was in APPLY-only status before you performed system definition.

  Attention: Some SYSMODs in APPLY-only status might require special handling for HOLDs that require either a system definition or a DELETE. Follow the instructions in the HOLD statements for those SYSMODs.

Generating JCL to build non-system definition target libraries

Some elements of IMS are not included in the IMS system definition process. These elements are identified to SMP/E and built during APPLY processing for their FMIDs.

The SMP/E GENERATE command can be used to create JCL that can be used to rebuild these components in their target libraries. SMP/E GENERATE can also be used to create JCL for other products in the IMS distribution zone, such as IRLM. SMP/E GENERATE processing is dependent on the SMP/E parameter ACCJCLIN being set in the distribution zone when the FMID is ACCEPTed.

Applying maintenance for the IVP dialog

Service affecting the IVP dialog process can require that special processing be performed.

SMP/E HOLDDATA identifies the required actions, if there are any that need to be performed.

The following actions might need to be performed, as identified in HOLDDATA:

- Table Merge

  Table Merge is necessary if rows have been added, changed, or deleted in one of the master tables. Table merge causes the changes to be propagated to the user tables in INSTALIB. Default values for variables are not updated for variables that have been changed by dialog processing.
- Variable Gathering

  You can modify the default values for new and changed variables.
- File Tailoring

  You can rerun File Tailoring to add INSTALIB members for new JOBs or TASKs or to update INSTALIB members with new or changed variable values.
- Execution

  You can run or rerun portions of the IVP processes.

Upgrading z/OS

Before you upgrade the z/OS system on which IMS is running, consider any requirements for the z/OS interface and VTAM interfaces.

Ensure that you perform the following when upgrading the z/OS system on which IMS is running:

1. Review the z/OS considerations. See Chapter 5, “z/OS interface considerations,” on page 111 for more information.
2. Review the VTAM considerations. See Chapter 6, “VTAM interface considerations,” on page 123 for more information.

Note: The IVP D series of samples contains examples of all of the z/OS and VTAM interfaces, except for the Channel-to-Channel (CTC) Channel-End Appendage. See *IMS Version 15.2 Installation* for more information.

Ensuring proper SYSLIB concatenation

The order in which your macro libraries are concatenated is critical. Otherwise, unpredictable results might occur during assembler processing. Ensure that your libraries are concatenated in the correct order.

SMP/E Apply:

SMPMTS
A target library for macros that exist only in a distribution library. This data set enables the current version of the macros to be used for assemblies during APPLY processing.

IMS.SDFSMAC
Target library for all IMS macros.

MVS Macro Libraries
Consist of the appropriate combination of SYS1.MACLIB (AMACLIB), SYS1.MODGEN (AMODGEN), SYS1.TSOMAC (ATSOMAC), and ASM.SASMMAC2. ASM.SASMMAC2 contains concept 14 macros and comes with the High Level Assembler in the HLASM Toolkit.

SMP/E Accept:

Note: The noticeable difference from the SMP/E APPLY process is the absence of SMPMTS, and SMP/E pointing to distribution libraries rather than target libraries. SMPMTS contains versions of macros that have not been accepted.

IMS System Definition Stage 1:

1. IMS.ADFSMAC

IMS System Definition Stage 2:

Interpreting binder return codes properly

Some binder return codes can be safely ignored while others cannot.

The following table lists the acceptable return codes from the various binder processes:

Table 16. Acceptable return codes from the binder

Type of SMP/E processing	Return code	How to interpret
APPLY	0	Do not ignore unresolved external references. The exception is binds into SMPLTS.
ACCEPT	4	You can safely ignore unresolved external references.
System Definition STAGE 2	0	Do not ignore unresolved external references.

Recommendation: Point to a different utility entry in SMP/E for APPLY and ACCEPT processing.

Assembling and binding a sample exit routine using SMP/E

The following example demonstrates a technique that you can use to have SMP/E assemble and bind one of the sample exit routines.

```
++ USERMOD (XYZUMOD) .
++ VER (P115)
    FMID(HMK15.200) .
++ JCLIN.
//INJCLIN  JOB . . .
//LKED     EXEC PGM=IEWL,
//         PARM='('SIZE=(880K,64K)',RENT,REFR,NCAL,LET,XREF,LIST)
//ADFSLOAD DD DSN=IMS.ADFSLOAD,DISP=SHR
//SYSPUNCH DD DSN=IMS.OBJDSET,DISP=SHR
//SYSUT1   DD UNIT=(SYSDA,SEP=(SYSLMOD,SYSLIN)),SPACE=(1024,(200,20))
//SYSPRINT DD SYSOUT=A
//SYSLMOD  DD DSN=IMS.SDFSRESL,DISP=SHR
//SYSLIN   DD  *
  INCLUDE ADFSLOAD(DFSCSI00)
  INCLUDE SYSPUNCH(DFSGMSG0)
  ENTRY DFSGMSG0
  NAME DFSGMSG0(R)
++ SRC   (DFSGMSG0) SYSLIB(SDFSSMPL) DISTLIB(ADFSSMPL) .
DFSGMSG0    TITLE 'DFSGMSG0 -- GREETING MESSAGES user exit routine routine'
...........................................................................
...........................................................................
...........................................................................
```

Migrating to a new version of IMS

When migrating to a new version of IMS, there are certain tasks that should be performed regardless of which version you are migrating from.

When migrating to a new version of IMS, ensure that you perform the following tasks:

1. Review the *Release Planning Guide* for the version that you are migrating to. In particular, review the migration and coexistence information.
2. If you are skipping a version, review the *Release Planning Guide* for those versions you are skipping. In particular, review the migration and coexistence information.

3. Review the PSP bucket for the version that you are migrating to.
4. If you are skipping a version, review the PSP bucket for those versions you are skipping.

Chapter 15. Planning for shared queues

This topic gives an overview of the concepts you should understand to use shared queues and then outlines the planning and administrative tasks associated with shared queues.

To manage message queues, your installation can use the IMS queue manager with either message queue data sets or shared queues.

Before operating in a shared-queues environment, read the information in this topic, which explains how operating in a shared-queues environment differs from operating in a non-shared-queues environment.

The role of CQS in a shared-queues environment

The Common Queue Server (CQS) is a generalized server that manages objects on a coupling facility list structure, such as a queue structure or a resource structure. CQS receives, maintains, and distributes data objects from shared queues on behalf of multiple clients.

Each client communicates with a CQS to access the shared queues. IMS is one example of a CQS client that uses CQS to manage both its shared queues and shared resources.

CQS uses the z/OS coupling facility as a repository for data objects. Storage in a coupling facility is divided into distinct objects called structures. Authorized programs use structures to implement data sharing and high-speed serialization. The coupling facility stores and arranges the data according to list structures. Queue structures contain collections of data objects that share the same name, known as queues.

The IMS role in a shared-queues environment

In a shared-queues environment, IMS acts as a front end for terminals, as a back end for processing, or as both simultaneously.

IMS as a front end

As a front end, an IMS manages terminal resources and originates communications traffic, such as commands, transactions, and message switches. The message is typically the result of data that is entered at a terminal. A front-end IMS can also play the role of back end, if database, program, and transaction resources are defined.

IMS as a back end

As a back end, an IMS manages transactions, database resources, and schedules application programs. A back-end IMS can also play the role of front end, if terminal resources are defined.

IMS as both a front end and a back end

As both a front end and a back end, IMS can process messages that are entered from one IMS terminal either on that IMS or on another IMS within the IMSplex.

Three processing environments exist when IMS acts as a back-end processor: local, remote MSC, and remote shared queues.

Definitions:

- *Local processing* occurs when the IMS dependent region that processes the message is on the same IMS as the IMS control region that received the message.
- *Remote MSC processing* occurs when the IMS dependent region that processes the message is on a different IMS than the IMS control region that received the message, and the message is sent to the remote IMS using MSC.

- *Remote shared-queues processing* occurs when the IMS dependent region that processes the message is on a different IMS than the IMS control region that received the message. In this case, however, the shared-queues facility contains the message that the remote IMS processes.

How IMS registers interest in queues

An IMS system registers and deregisters its interest in shared queues according to the type of work the IMS system can process. Consequently, the CQS for this IMS can notify the IMS when work is present on a queue.

The types of work include:

- Transactions
- LTERMs
- MSC resources (such as remote LTERMs, remote transactions, and MSNAMEs)

In general, IMS registers its interest as the resources are added (or started), and IMS deregisters its interest as the resources are deleted (or stopped).

Queue types

Various types of queues are managed in a shared-queues environment. Each queue type is used for a different type of work. An IMS registers interest in only those queue types that it can process, based on the types of work you define for it.

The queue types and the work that IMS processes on them are shown in the following table.

Table 17. Queue types maintained in a shared-queues environment

Queue type	Description
Transaction-ready queue	Contains first qbuffer of messages that are destined for transactions
Transaction-staging queue	An internal queue that the IMS Queue Manager uses for holding those parts of messages that exceed a single qbuffer
Transaction-suspend queue	Contains first qbuffer of messages destined for suspended transactions
Transaction-serial queue	Contains first qbuffer of messages that are destined for serial transactions (transactions defined to IMS on the TRANSACT macro as SERIAL=YES)
LTERM-ready queue	Contains first qbuffer of messages that are destined for LTERMs or MSNAMEs
LTERM-staging queue	An internal queue that the IMS Queue Manager uses for holding those parts of messages that exceed a single qbuffer
APPC-ready queue	Contains first qbuffer of messages that are destined for APPC devices
REMOTE-ready queue	Contains first qbuffer of messages that are destined for remote transactions or remote LTERMs
OTMA-ready queue	Contains first qbuffer of messages that are destined for OTMA devices

Registering and deregistering interest in transactions

IMS registers or deregisters interest in transaction queues when certain actions are taken.

- IMS registers interest in transaction queues when any of the following events occurs:
  - IMS is initialized.

  - Transactions are added by online change after the **/MODIFY COMMIT** command is entered or, with global online change enabled, after the **INITIATE OLC PHASE(COMMIT)** command is entered.
  - An operator enters the **/START TRAN** command.
  - An operator enters the **UPD TRAN START(SCHD)** command.
  - IMS reconnects to CQS after CQS has terminated and been restarted.
- IMS deregisters interest in transaction queues when any of the following events occurs:
  - IMS shuts down.
  - Transactions are deleted by online change after the **/MODIFY COMMIT** command or, with global online change enabled, after the **INITIATE OLC PHASE(COMMIT)** command.
  - A **/STOP TRAN** command is entered.
  - A **/PSTOP TRAN** command is entered.
  - An **UPD TRAN STOP(SCHD)** command is entered.
- For IMS Fast Path:
  - IMS registers interest in a program queue during IFP region initialization for the name of the program that the region processes.
  - IMS deregisters interest in a program queue when the IFP region is terminated.

Registering and deregistering interest in LTERMs

IMS registers and deregisters interest in LTERM queues when certain actions are taken.

- IMS registers interest in LTERM queues when any of the following events occurs:
  - A user logs on to a static terminal.
  - A user signs on to a dynamic terminal.
  - An LTERM is assigned to an active user or node.
- IMS deregisters interest in LTERM queues when any of the following events occurs:
  - A user logs off from a static terminal.
  - A user signs off from a dynamic terminal.
  - An LTERM is assigned to an inactive user or node.

Registering interest in MSC resources

When an MSC logical link is started, IMS registers interest in the MSNAME that is defined with it.

All remote transactions that are defined using the same SYSID pair are also registered.

When the logical link is stopped, IMS deregisters interest in the MSNAME and its associated remote transactions.

Message flow in a shared-queues environment

In a shared-queues environment, when an IMS retrieves a message from a shared queue, the message is locked on that shared queue until the IMS unlocks or deletes the message. If IMS does not unlock or delete a message from the shared queue (for example, because of an IMS abend), the message remains locked.

Locked messages are not available to other IMS systems for processing. If you cold start the IMS after it locks messages on a shared queue, those messages remain locked, and they are moved to the cold queue.

If RM is defined with a resource structure, IMS releases any locked messages destined for an LTERM on the failing IMS. Messages become available to a user signing on to another IMS in the IMSplex.

Definition: The *cold queue* is a type of queue on which CQS places locked messages that it finds on its private queues after a client cold starts. Messages remain on the cold queue until they are unlocked or deleted.

Units of work (UOW) tracking

In a non-shared-queues environment, a unit of work is tracked by a recovery token and by a unit of work (UOW). In a shared-queues environment, a unit of work is tracked using only a UOW.

The UOW consists of the following fields:

- Originating-system message ID: Message ID assigned by the IMS that originates the message.
  - Originating IMSID
  - Time-stamp token
- Processing-system message ID: Message ID assigned by the IMS processing the message.
  - Processing IMSID
  - Time-stamp token

Because the UOW has IDs for both the system that originates the message and the system (if any) that processes the message, all messages that are associated with an original message can be tied together by the UOW (specifically, the originating-system message ID in the UOW).

Terminal autologon

Application programs can run on any IMS back-end system; therefore, output can be generated for the same user or terminal on any of these systems.

In order to deliver output to all waiting users, an autologon terminal is often switched between IMS systems in the IMSplex.

Message and program-to-program switches

Message switches are queued globally in a shared queue environment. Program-to-program switches are eligible to be considered for local-first optimization. If the program-to-program switch message is not selected for local-first processing, then it is queued globally.

If the destination of a message switch is not defined on the local IMS, the Destination Creation exit routine (DFSINSX0) is called to identify the destination.

While ETO is active, if the exit routine indicates that the destination is an LTERM, the exit routine creates a dynamic user structure for the LTERM and then queues the message.

The DFSINSX0 exit can also indicate that the destination is a transaction. IMS will create the transaction based on the environment and options provided by the exit:

- In a shared-queues environment without DRD, IMS will create a queuing-only transaction and queue the message globally.
- In a shared-queues environment with DRD enabled, IMS will create a transaction either for queueing or for scheduling, as requested by the exit.

Dynamic control blocks

To enable a user to enter transactions on IMS systems that do not have the transaction defined, a user can supply information in the Destination Creation exit routine (DFSINSX0) to cause IMS to build a dynamic scheduler message block (SMB).

Input and output messages are enqueued to the shared queues, rather than to the control blocks.

IMS Queue Manager

The IMS Queue Manager manages the IMS message queues. The Queue Manager data sets, SHMSG and LMSG, are not used to back up messages in a shared-queues environment.

However, in order to reduce necessary changes during IMS system definition, the MSGQUEUE macro is still required; the data sets are defined, but they are never allocated or opened.

The Queue Manager uses a number of real storage queue buffers to store the messages prior to placing them in the shared queues on the coupling facility. However, if the message needs to remain in the local IMS rather than being placed on the shared queues, the message is placed on a local queue. The local queues are also managed by the Queue Manager.

Messages that are placed on the shared queues are recoverable. Because the message queue data sets are not used, a message on a local queue is only recoverable if the log record for the message was written to the IMS log after the oldest of the last two checkpoints was taken.

In an XRF environment, the local queue manager data sets (QBLKSL, SHMSGL, and LGMSGL) are used. While the XRF alternate subsystem is tracking, local messages are placed in the local message queue data sets (messages for the active subsystem are on the shared queues). When the XRF alternate subsystem takes over the active subsystem, any messages in the local message queue data sets are merged with those on the shared queues.

MTO messages

Messages for the primary master terminal operator (MTO) are kept local and are not recovered across an IMS restart. Messages for the secondary MTO are placed on the shared queues using the LOCAL=YES option.

These secondary MTO messages are recovered across an IMS restart and not are retrieved by the CQSREAD.

Serial application processing

In an IMSplex environment that does not use shared queues, if you want to define an application program as a serial program (that is, it cannot be scheduled simultaneously into message or batch message regions), you use the SCHDTYP= parameter on the APPLCTN macro. Application programs defined as serial cannot be scheduled to run simultaneously in more than one message or batch message region. Parallel application programs (SCHDTYP=PARALLEL), however, can be scheduled to run simultaneously.

In an IMSplex environment with shared queues, you can also define an application program as serial. In addition to defining the application program as serial (SCHDTYP=SERIAL), you must:

- Define an active instance of Resource Manager.
- Update the CQS global structure definition IMS PROCLIB member data set (CQSSGxxx) to define a resource structure using the RSRCSTRUCTURE parameter.
- Identify the resource structure using the RSRCSTRUCTURE parameter of the CSL RM initialization parameters PROCLIB member data.

When shared queues is enabled, and RM is active with a defined resource structure, IMS serializes the scheduling of applications defined as serial in an IMSplex.

Related reading:

- For information on defining and tailoring RM, see *IMS Version 15.2 System Definition*.
- For information on the CQSSGxxx IMS PROCLIB member data set, see *IMS Version 15.2 System Definition*.

Serial transaction processing

Messages for serial transactions are only processed by the IMS that places the messages on the queue. IMS keeps a transaction-serial queue for serial transaction messages in the coupling facility MSGQ structure.

AOI programs must define their transactions as serial (specified as SERIAL=YES on the TRANSACT macro).

Conversational transactions in a shared-queues environment

In a shared-queues environment, conversation status is local to the IMS that initiates the conversation.

When an IMS initiates a conversation, the following events occur:

1. IMS places the conversational transaction on the shared queues.
2. A locally defined SMB (or RSMB) or the Output Creation User exit routine (DFSINSX0) identifies the transaction as conversational.
3. Any IMS that registers interest in the transaction can process it. Also, any IMS can process any step of the conversation.
4. One IMS reads the message and schedules an application program to process the message.
5. The application program processes the message, saves data in the SPA, and responds to the initiating terminal.

The **/EXIT**, **/HOLD**, and **/RELEASE** commands apply only to the IMS that initiates the conversation.

Configuring a shared-queues environment

You can configure a shared-queue environment by using either a cloned configuration or a partitioned configuration.

Figure 28 on page 205 and Figure 29 on page 206 illustrate two methods of configuring a shared-queues environment.

Cloned configuration

A *cloned configuration* involves creating identical IMS systems, with almost identical system definitions. Some differences in the definitions are required, such as MSC links. You can override the MTO definitions generated during system definition in IMS PROCLIB member data set DFSDCxxx.

Related reading: For more information on overriding the MTO definitions, see *IMS Version 15.2 System Definition*.

The LTERMs, databases, and transactions for all the IMS systems are defined identically. In this configuration, any IMS is able to process a particular transaction. Cloned configurations are useful for automatic workload balancing.

Recommendation: To take full advantage of a cloned configuration, ensure that the databases are shared across the IMSplex.

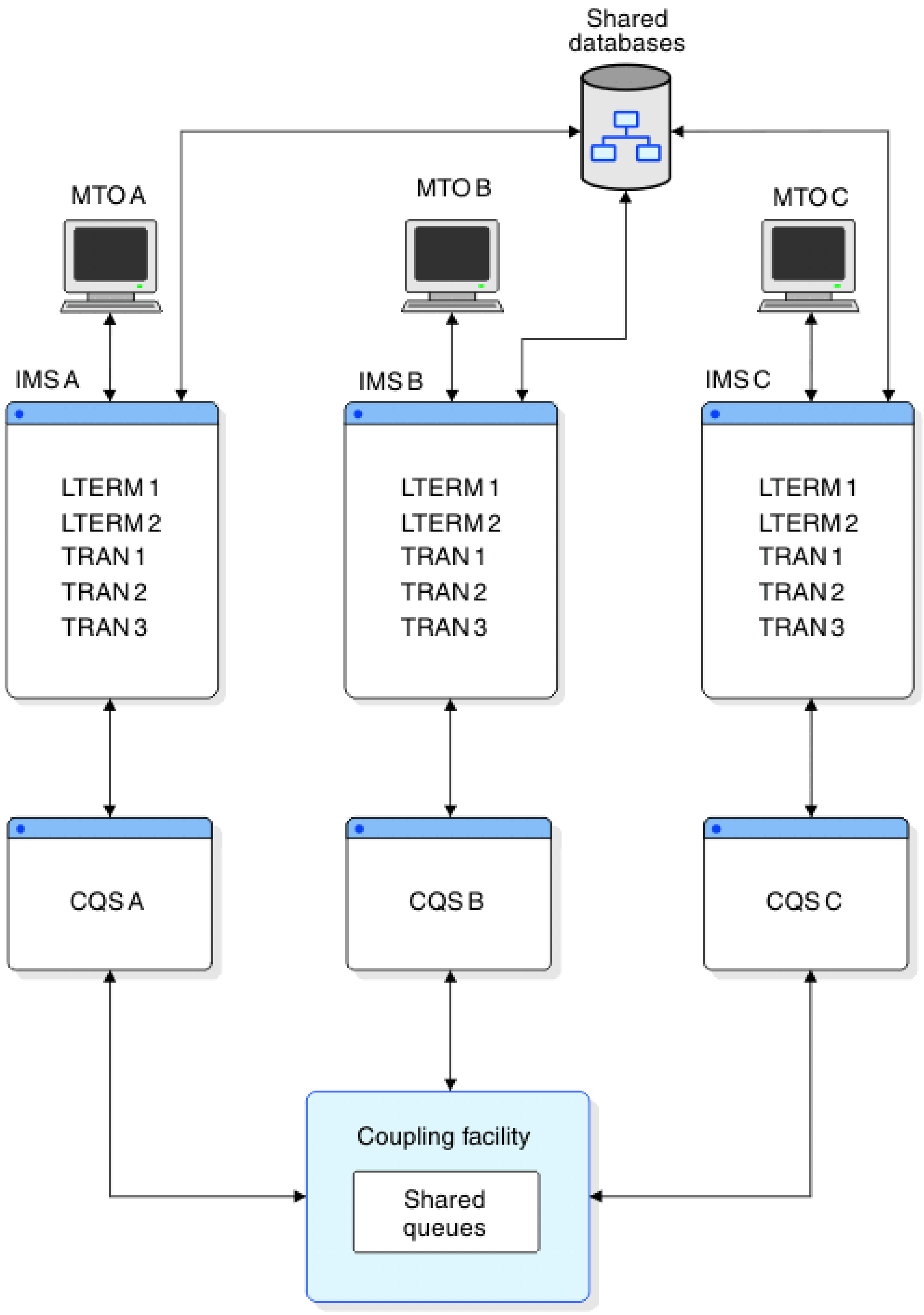


Figure 28. Cloned configuration in a shared-queues environment

Partitioned configuration

A *partitioned configuration* separates the VTAM network responsibilities from the database work. Each resource is defined to only one IMS.

You can define an ES/9000 9021 to manage the VTAM network as an IMS front end, while multiple smaller 9672 CMOS processors manage the database work.

Partitioned configurations are useful as replacements for, or additions to, MSC networks.

In the partitioned configuration in the following figure, IMS A acts as the front end, and has LTERM 1, LTERM 2, and MTO 1 defined. IMS B, IMS C, and IMS D act as back ends, each with separate transactions and databases defined. Transactions in this configuration can run only on the IMS systems on which they are defined.

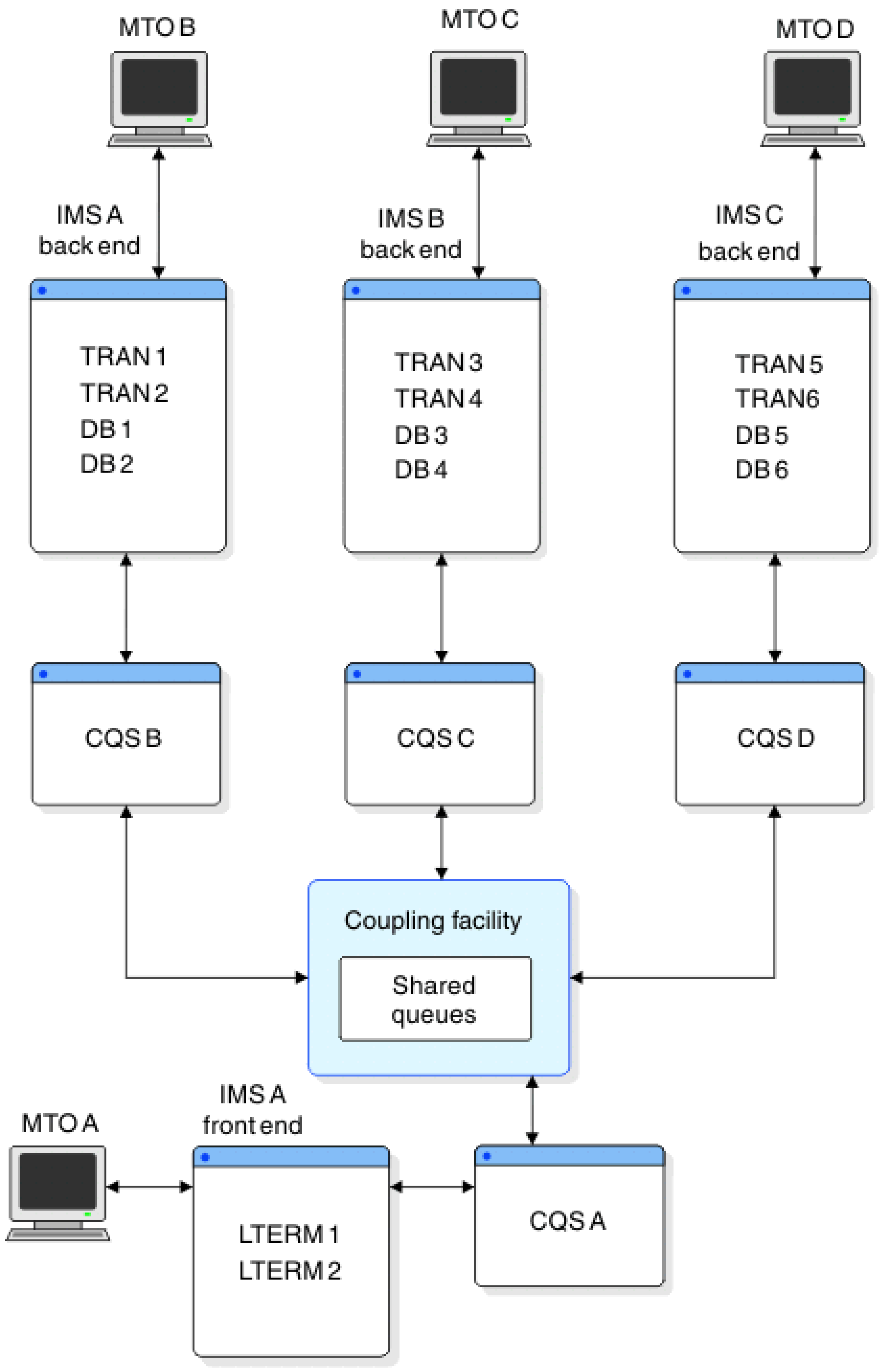


Figure 29. Partitioned configuration in a shared-queues environment

Enabling shared queues

You enable shared queues by defining them in the z/OS coupling facility resource management (CFRM) policy.

This topic discusses what you must include in the CFRM policy and the other definitions that must conform to the definitions you make in the CFRM policy.

Defining the CFRM policy

To enable shared queues, you must define a CFRM policy that contains structure names and attributes for all message queues. If you use Fast Path and share EMH queues, you must also include structure names and attributes for EMHQ structures.

Be sure that the structure names you define in the CFRM policy for message queues and any optional EMH queues are also defined in the following places:

- The shared-queues IMS.PROCLIB member data set (DFSSQxxx) parameters MSGQ= (and EMHQ= if you are sharing EMH queues)
- The CQS local structure definition PROCLIB member data set (CQSSLxxx) parameter STRNAME=
- The CQS global structure definition PROCLIB member data set (CQSSGxxx) parameters STRNAME= and OVFLWSTR=

Recommendation: If possible, place high-use structures on separate coupling facilities. For example, place your IMS data-sharing structures on a different coupling facility from your shared-queues structures. Place your VSAM and Db2 for z/OS structures on yet another coupling facility. You can place low-use structures (such as those for RACF) on any coupling facility where space is available.

Related reading: For more information on definitions that are required to use CQS, see *IMS Version 15.2 System Definition*.

The following example shows a sample CFRM policy containing definitions of shared queues and a resource structure.

```
  //CFRMPLCY JOB MSGCLASS=A,REGION=2000K,CLASS=K
  // MSGLEVEL=(1,1)
  //*******************************************************************************
  //* This JCL is used for configuration. INITSIZE is                             *
  //* used for the primary MSGQ and EMHQ structures.                              *
  //*******************************************************************************
  //* 2 CF                                                                        *
  //*******************************************************************************
  //POLICY EXEC PGM=IXCM2APU
  //STEPLIB DD DSN=SYS1.MIGLIB,DISP=SHR
  //SYSPRINT DD SYSOUT=A
  //SYSIN DD *

  DATA TYPE(CFRM)
  DEFINE POLICY
             NAME(CONFIG01)
             REPLACE(YES)

       CF NAME (CF01)
       TYPE(nnnnnn)
       MFG(aa)
       PLANT(nn)
       SEQUENCE(nnnnnnnnnnnn)
       PARTITION(n)
       CPCID(nn)

       CF NAME (CF02)
       TYPE(nnnnnn)
       MFG(aa)
       PLANT(nn)
       SEQUENCE(nnnnnnnnnnnn)
       PARTITION(n)
       CPCID(nn)
            .
            .
            .

  STRUCTURE NAME(QMSGIMS01)
           SIZE(16000)
           INITSIZE(8000)
|          MINSIZE(8000)
           PREFLIST(CF01,CF02)
           REBUILDPERCENT(1)
|          ALLOWAUTOALT(YES)
|          FULLTHRESHOLD(60)

  STRUCTURE NAME(QMSGIMS01OFLW)
           SIZE(8000)
```

```
 |          MINSIZE(8000)
            PREFLIST(CF01,CF02)
            REBUILDPERCENT(1)
 |          ALLOWAUTOALT(YES)
 |          FULLTHRESHOLD(60)

   STRUCTURE NAME(QEMHIMS01)
            SIZE(16000)
            INITSIZE(10000)
 |          MINSIZE(10000)
            PREFLIST(CF01,CF02)
            REBUILDPERCENT(1)
 |          ALLOWAUTOALT(YES)
 |          FULLTHRESHOLD(60)

   STRUCTURE NAME(QEMHIMS01OFLW)
            SIZE(8000)
 |          MINSIZE(8000)
            PREFLIST(CF01,CF02)
            REBUILDPERCENT(1)
 |          ALLOWAUTOALT(YES)
 |          FULLTHRESHOLD(60)

   STRUCTURE NAME(MVSLOGQMSG01)
            SIZE(16000)
            INITSIZE(11000)
            PREFLIST(CF01,CF02)

   STRUCTURE NAME(MVSLOGQEMH01)
            SIZE(4000)
            PREFLIST(CF01, CF02)
            REBUILDPERCENT(1)

   STRUCTURE NAME(QRSCIMS01)
             SIZE(16000)
             INITSIZE(8000)
 |           MINSIZE(8000)
             ALLOWAUTOALT(YES)
             FULLTHRESHOLD(60)
             DUPLEX(ALLOWED)
             PREFLIST(CF01,CF02)
               .
               .
               .
```

Defining message queue list structures

You must define the primary list structure, MSGQ, to IMS before using shared queues. You can optionally define the overflow structure.

In the example, `STRUCTURE NAME(QMSGIMS01)` defines a message queue structure, and `STRUCTURE NAME(QMSGIMS01OFLW)` defines an overflow structure for the QMSGIMS01 message queue.

Defining a resource structure

You must define a resource structure to use RM to share global status. In the example, `STRUCTURE NAME(QRSCIMS01)` defines a resource structure. For more information about how RM maintains global resource information, see "Information managed by the CSL RM" on page 157.

Defining Fast Path message queue list structures

If you define the MSGQ primary list structure on an IMS that has Fast Path defined, you can optionally define a primary list structure for shared EMH queues. This structure is called the EMHQ structure. You can then optionally define an EMHQ overflow structure.

If you define an EMHQ structure, Fast Path transactions can be processed by IMS systems in the shared-queues group. If you do not define an EMHQ structure, Fast Path transactions, if any, are only processed locally.

In the example, `STRUCTURE NAME(QEMHIMS01)` defines an EMHQ structure, and `STRUCTURE NAME(QEMHIMS01OFLW)` defines an overflow structure for the QEMHIMS01 EMH queue.

Defining a z/OS log stream

You must define a z/OS log stream. You define a z/OS log stream by updating definitions.

To define a z/OS log stream:

- In the GRSTNLxx PARMLIB member, specify every IMS in the Parallel Sysplex environment as being in the same global resource serialization complex. This allows global serialization of resources in the IMSplex.
- Define log stream structure names and attributes in the CFRM policy for message queues and, if sharing, EMH queues.

  Example: In the example in "Defining the CFRM policy" on page 207, `STRUCTURE NAME(MVSLOGQMSG01)` defines a log stream structure name for the message queues. Similarly, `STRUCTURE NAME(MVSLOGQEMH01)` defines a log stream structure for Fast Path EMH queues.
- Install DFSMS and change SMS DATACLAS, STORCLAS, and MGMTCLAS constructs.
- Format the inventory coupled data set and define log streams and structure names.
- Update the COUPLExx PARMLIB member.
- Define the logger inventory data set in SYS1.PARMLIB(COUPLExx).
- Define log stream and coupling facility structure names to the logger inventory coupled data set.

Related reading: For more information on defining the z/OS log stream, see *IMS Version 15.2 System Definition*.

Defining CQS parameters

The CQSIPxxx, CQSSLxxx, and CQSSGxxx members of the PROCLIB data set must be updated to enable shared queues.

To enable shared queues, define the following CQS parameters:

- CQS initialization parameters in PROCLIB member data set CQSIPxxx.
- CQS local structure definition parameters in PROCLIB member data set CQSSLxxx (for local structure definition). These definitions apply only to a single CQS.
- CQS global structure definition parameters in PROCLIB member data set CQSSGxxx (for global structure definition).

  You must define one of each of these parameters for each structure pair. All of the CQSs that share queues must have the same values specified in the CQSSGxxx PROCLIB member data set. If they differ, only the first CQS connects to the structure, and the others fail.
- Execution parameters (optional).

  If you do not specify ARMRST=, CQSGROUP=, SSN=, STRDEFG=, or STRDEFL=, CQS retrieves the values that are specified in the CQSIPxxx PROCLIB member data set.

Related reading: For more information about the CQS parameters, see *IMS Version 15.2 System Definition*.

IMS parameters for shared queues

To enable shared queues, you must define IMS control region parameters and IMS startup parameters.

Define the following IMS control region execution parameters:

LGMSGSZ=
The record length of the long message.

QBUF=
The upper expansion limit of the queue buffer pool.

QBUFHITH=
A one- to three-digit number with a value in the range of 1 to 100 that represents the high threshold percentage at which the message queue buffer is dynamically expanded. The default percentage is 80%.

QBUFLWTH=
A one- to three-digit number with a value in the range of 1 to 100 that represents the low threshold percentage at which the message queue buffer is compressed. The default percentage is 50%.

QBUFMAX=
The maximum number of message queue buffers that are allowed in the queue buffer pool.

QBUFPCTX=
A one- to three-digit number with a value in the range of 1 to 100 that represents the percentage of the originally allocated message queue buffers that are dynamically expanded when the QBUFHITH parameter value is reached. The default percentage is 20%.

QBUFSZ=
The size of the queue buffers.

SHAREDQ=
The suffix of the shared-queues PROCLIB member data set. This parameter is in the PARMLIB member DFSSQxxx.

SHMSGSZ=
The record length of the short message.

Define the following IMS startup parameters in the PROCLIB member data set DFSSQxxx:

CQS=
The PROCLIB member data set that contains the CQS procedure. CQS= is optional. The default is CQS=CQS.

CQSSSN=
The name of the CQS address space subsystem.

EMHQ=
The name of the EMH queues structure. If this statement exists, an EMHQ structure (and the other associated structures) is required in the shared-queues environment.

MSGQ=
The name of the message queues structure.

SQGROUP=
A one- to five-character identifier that represents the z/OS cross-system coupling facility IMS shared-queues group name, which is concatenated to the letters DFS. All IMS systems that share the same set of structures must specify the same SQGROUP= name. The SQGROUP= name can be the same as the CQSGROUP= name, which is specified in the CQSIPxxx PROCLIB member data set.

WAITRBLD=Y | N
Specifies whether activity on the EMHQ structure should wait until CQS completes the structure rebuild process. WAITRBLD=NO specifies that activity on the EMHQ structure should continue while CQS rebuilds the structure. The WAITRBLD parameter is optional. The default is WAITRBLD=N.

Related reading: For more information on these IMS parameters, see *IMS Version 15.2 System Definition*.

Using the Common Queue Server

IMS uses the CQS interface to manage the shared queues in a sysplex environment. From each IMS that acts as a CQS client, IMS uses this CQS interface to access the shared queues.

Restriction: When using the same LTERM name on multiple systems in a shared queues sysplex, all conversational and response mode reply messages must be processed by the originating terminal's system. This means that while a conversation or response mode operation is still in-progress on one IMS, the same LTERM name cannot be used, simultaneously or serially, by a terminal on another IMS. This Transaction Manager (TM) restriction applies to both Fast Path and non-Fast Path in a shared-queues environment.

Starting CQS

You can activate CQS in one of two ways:

- As a z/OS task, using the z/OS **START** command
- As a z/OS batch job

In addition, IMS automatically starts CQS, when appropriate.

Restarting CQS

Depending on whether a CQS structure contains data, you can warm start CQS or cold start CQS:

- If the structure is empty, you must cold start CQS.
- If the structure contains data, you can either warm start or cold start CQS.

When you have completed restarting CQS, the CQS ready message is issued (CQS0020I).

CQS warm start

During a warm start, CQS reads the checkpoint data set to find the log token representing the last system checkpoint. When CQS finds this log token, it initiates a warm start. If CQS fails to find this log token in the checkpoint data set, it reads the log token from the structure. If CQS finds the log token, it issues a WTOR in order to allow you to confirm the use of this token.

CQS cold start

When CQS cold starts after connecting to an empty structure, CQS purges all log records for the structure and performs a system checkpoint. After the system checkpoint is complete, the structure and CQS restart is complete.

Using CQS user-supplied exit routines

You can use the following exit routines to monitor and modify CQS activities:

- CQS Initialization/Termination exit routine
- CQS Client Connection exit routine
- CQS Queue Overflow exit routine
- CQS Structure Statistics exit routine
- CQS Structure Event exit routine

Changing the size of coupling facility structures

The initial size of a structure on the coupling facility is determined by the value of the INITSIZE parameter in the CFRM policy. CQS allows you to dynamically reconfigure the size of a structure.

When the first CQS connects to a structure, the value specified for INITSIZE is the size of that structure. If enough free space does not exist for this INITSIZE value, the size of the structure becomes that of the available space in the coupling facility.

Initiating system checkpoint

As a system checkpoint for recovering CQS information during restart, each CQS writes its own control blocks to the z/OS log. To retrieve this restart information, CQS also writes information to the CQS checkpoint data set. CQS does not quiesce activity while the checkpoint is in progress.

Checkpoint data sets

For each structure pair, CQS maintains a checkpoint data set. CQS writes to a checkpoint data set and then uses it during restart. Checkpoint data sets are dynamically allocated during CQS initialization.

How CQS restarts after system checkpoint

During CQS restart, CQS reads the log records from the last system checkpoint and rebuilds the work that was in progress.

Initiating structure checkpoint

Recovering structures

z/OS allows you to rebuild structures. You rebuild structures either by copying them or by recovering them.

Rebuilding structures

One or more CQSs must be running in order to copy or recover structures and the messages on those structures. When the new structure is allocated, policy changes (such as structure location) are applied.

Copying structures

If at least one CQS has access to the structure when structure rebuild is initiated, one of the CQS systems that still has access to the structure copies all of the messages (both recoverable and irrecoverable) from that structure to the new structure.

Recovering structures

If no CQS has access to the structure when structure rebuild is initiated, the structure is recovered from the SRDS and the z/OS log. Irrecoverable messages (such as Fast Path input messages) are lost.

Deleting structures

You can delete a structure when no CQS is connected to it. To delete a structure:

1. Shut down all CQSs that are connected to the structure.
2. Delete any failed persistent connections.

   Attention: If a CQS fails while it is connected to a structure, allow it to restart so that it can clean up any work that was in process at the time it failed. This command can be used to terminate the failed connections when you must delete the structure. If this command is used incorrectly, the queues or resources might be lost.

   Issue the following command:

   ```
   SETXCF FORCE,CONNECTION,STRNAME=strname,CONNAME=ALL
   ```
3. Issue the command:

   ```
   SETXCF FORCE,STRUCTURE,STRNAME=strname
   ```

   Ensure that the *strname* value in this command is the same as the *strname* value that is specified in the CQS global structure definition parameters in PROCLIB member data set CQSSGxxx and the CQS local structure definition parameters in PROCLIB member data set CQSSLxxx.

Deleting message queues

You can delete message queues on a structure by deleting the structure.

Monitoring shared queue usage

You can monitor usage of the IMS shared message queue structure by using the Queue Space Notification exit routine (DFSQSSP0). The exit is passed information about the total number of allocated and in-use entries and elements in the primary and overflow message queue structures. This information can be used to prevent the message queue structure from becoming full.

If you are writing a program that directly accesses a CQS structure, you can receive information about that structure's utilization by using the FEEDBACK= and FEEDBACKLEN= parameters on the CQSPUT and CQSDEL macros.

CQS logging and the z/OS logger

The z/OS system logger records all information necessary for CQS to recover structures and restart. CQS provides a File Select and Formatting Print utility to read the log records.

Related reading: For more information about defining the log stream, see *z/OS MVS Programming: Sysplex Services Guide*.

Related reference
CQSSGxxx member of the IMS PROCLIB data set (System Definition)
CQSSLxxx member of the IMS PROCLIB data set (System Definition)
File Select and Formatting Print utility (DFSERA10) (System Utilities)

Monitoring and requeuing structures using the Queue Control Facility

You can copy and remove messages, and query shared queues for message counts and ages, using Queue Control Facility (QCF) two ways: by submitting a QCF BMP job to run on an IMS in the shared-queues group, or by using the QCF TCO/ISPF interface.

Removing messages can be useful for clean-up or for re-inserting and processing messages later. You can also select and re-queue lost messages to the shared queues for reprocessing.

Requeuing the cold queue using QCF

When an IMS is abended and then cold started, CQS places messages in progress at the abend on the cold queues of the CQS shared-queues structure. You can analyze or delete these messages on the cold queue structure using QCF. Additionally, with QCF, you can move the messages back to one of the processing queues for re-processing.

Related reading: For more information on the QCF tool, see *IMS Queue Control Facility for z/OS User's Guide*.

Accessing CQS with IMS commands

These IMS commands apply to the local IMS only. Commands and command responses are not placed on the shared queues; therefore, they are not recoverable.

You can use the following IMS commands to interact with a CQS:

/CQCHKPT SHAREDQ
Initiates a CQS structure checkpoint.

/CQCHKPT SYSTEM
Initiates a CQS system checkpoint.

/CQQUERY STATISTICS
Displays statistics for primary and overflow structures.

/CQSET SHUTDOWN SHAREDQ
Sets status to take structure checkpoint at CQS shutdown.

/DEQUEUE
Dequeues one or more messages from one of the following:

- An outbound APPC queue
- A Fast Path program
- An LTERM
- An MSNAME

- An outbound OTMA queue
- A message queue
- A remote transaction or a remote LTERM
- A transaction

/DEQUEUE SUSPEND
Dequeues one or more suspended transactions.

/DISPLAY CQS
Displays CQS information that IMS tracks.

/DISPLAY MODIFY
Displays local work in progress.

/DISPLAY OVERFLOWQ
Displays a list of queue names that are in overflow mode.

/DISPLAY QCNT
Displays global queue information for a particular resource type.

/DISPLAY STRUCTURE
Displays structure status.

/DISPLAY TRACE TABLE QMGR
Displays queue manager trace table status.

/DISPLAY TRACE TABLE SQTT
Displays shared-queues trace table status.

/TRACE SET OFF TABLE QMGR
Stops trace of queue manager activity.

/TRACE SET ON TABLE QMGR
Starts trace of queue manager activity.

/TRACE SET OFF TABLE SQTT
Stops trace of shared-queues activity.

/TRACE SET ON TABLE SQTT
Starts trace of shared-queues activity.

QUERY TRAN
Displays information about messages on the shared queues.

QUEUE TRAN
Enqueues and dequeues messages on the shared queues.

List structures

CQS manages message queues that are shared within an IMSplex by using *coupling facility list structures*. A *primary list structure* holds the message queues. An *overflow list structure*, if you define one, holds additional queues after the primary list structure reaches a predefined threshold.

For IMS systems that do not share Fast Path transactions, define only message queue primary structures and optional message queue overflow structures. For IMS systems that do share Fast Path transactions, define both message queue primary (and optional overflow) structures and expedited message handler queue (EMHQ) primary (and optional overflow) structures.

Figure 30 on page 215 and Figure 31 on page 216 show shared queues for IMS clients A and B.

In the following figure, only message queue structures, message queue structure recovery data sets (SRDSs), message queue checkpoint data sets, and a message queue z/OS log stream are used.

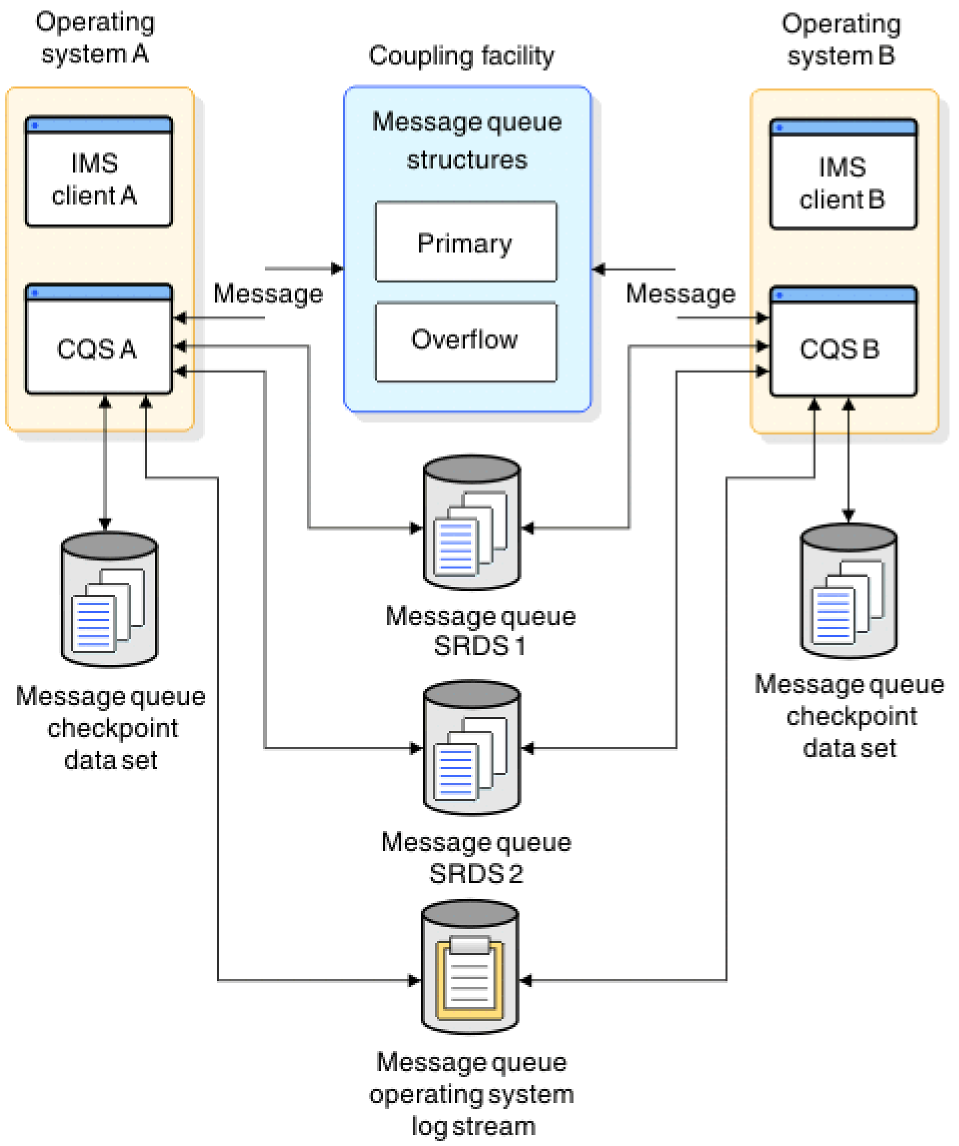


Figure 30. Shared-queues environment with its message queue (MSGQ) list structures

The configuration shown in the following figure shows EMHQ structures, along with their corresponding SRDSs, checkpoint data sets, and a z/OS log stream. A configuration such as this one would also include message queues, but these are not shown in the figure.

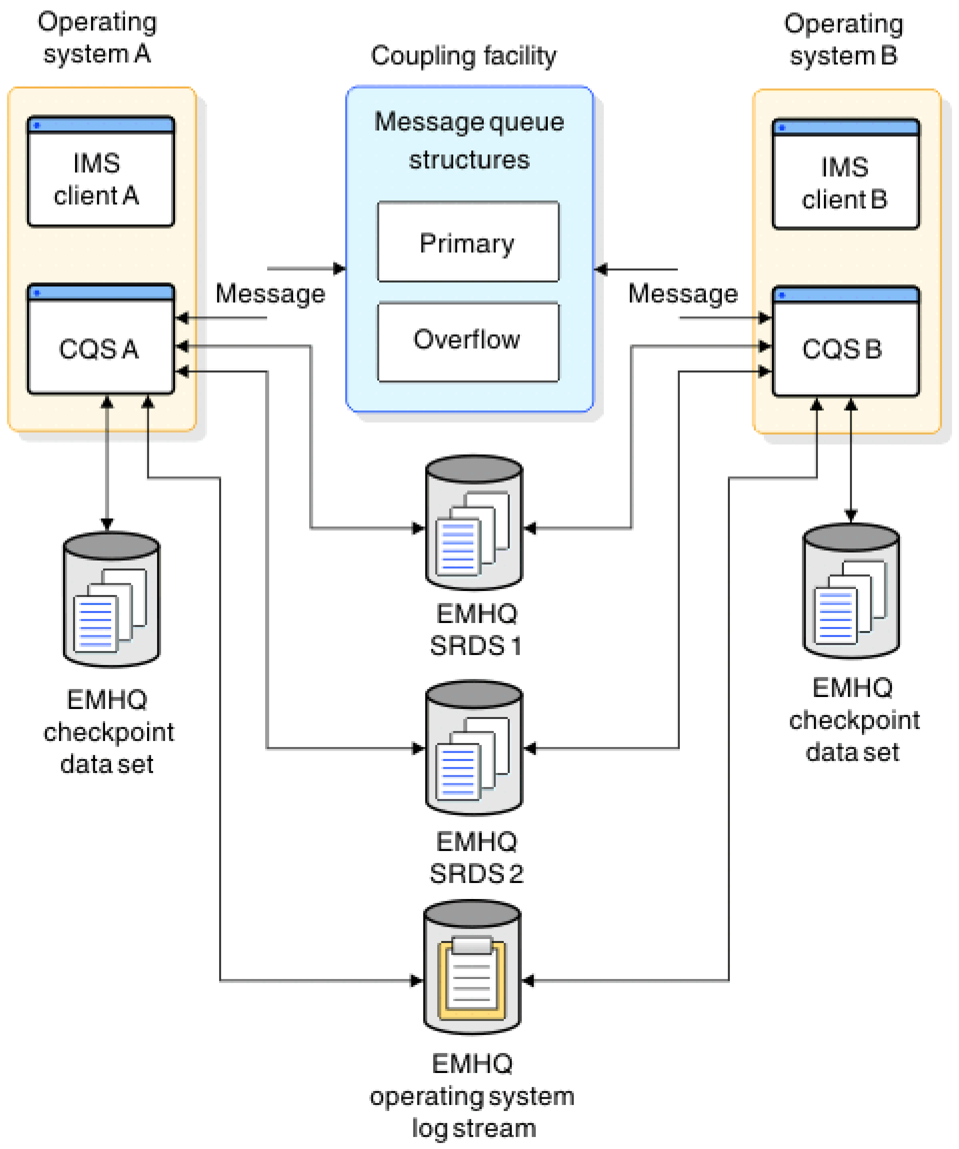


Figure 31. Shared-queues environment with its EMHQ list structures

Using associated printers

In a shared-queues environment, when a user signs on and specifies associated printer information, the IMS front end registers interest for each associated LTERM. When the application inserts a message to one of these LTERMs, the IMS front end is notified and begins autologon for the printer.

Recommendation: Do not provide autologon parameters for these users in the IMS back end; doing so causes both the front end and back end to process autologon for the associated printers.

For shared printers in a shared-queues environment, shared printer terminals are often switched between IMS systems in the IMSplex. This enables the output to be sent to any IMS in the IMSplex and to be printed.

Planning for IMS front-end switch

Using IMS Front-end Switch (FES) in a shared-queues environment requires one of two events.

To complete an IMS front-end switch, either:

- Coding the FES exit routine (DFSFEBJ0) to route the input messages to the correct LTERM (to deliver FES reply messages correctly).

- Establishing ISC sessions between IMS systems within the IMSplex (to ensure that FES input and reply messages are sent and received by any IMS).

Related reading:

- For information on the FES exit routine, see *IMS Version 15.2 Exit Routines*.
- For information on establishing ISC sessions, see *IMS Version 15.2 Communications and Connections*.

Determining eligibility for sysplex-wide processing

Some IMS messages in a shared-queues environment are eligible for sysplex-wide processing; others are not.

Definition: An *IMS message* can be one of the following statements:

- A transaction or transaction response
- A message switch
- A system message
- A message from LU 6.2 or OTMA
- A command or command response

All messages (not including commands or command responses) are placed in shared queues.

Messages eligible for sysplex-wide processing

Messages for the following types of communication are eligible to be processed on any IMS in the IMSplex:

- Transactions defined locally or using the Destination Creation exit routine (DFSINSX0)
- Fast Path transactions defined using the DBFHAGU0 exit routine with sysplex processing option LOCAL FIRST or GLOBAL ONLY
- Fast Path messages for a program that is active in the IMSplex
- The following synchronous message types:
  - APPC
  - OTMA
- LTERMs or by using the Destination Creation exit routine (DFSINSX0)

  Definition: A message for an LTERM can be:

  - A transaction response
  - A message switch
  - A system message
  - A command response

Messages not eligible for sysplex-wide processing

Messages for the following types of communication must be processed locally and are therefore not eligible for sysplex-wide processing:

- Serial transactions (transactions defined with SERIAL=YES specified on the TRANSACT macro)
- Commands and command responses

Managing APPC and OTMA messages in a sysplex environment

APPC and OTMA messages can be either asynchronous or synchronous.

Asynchronous APPC and OTMA messages

To ensure delivery of APPC and OTMA messages in a sysplex environment, enable APPC and OTMA on every back-end IMS in the shared-queues group. If a back-end IMS does not have APPC or OTMA enabled, any asynchronous APPC or OTMA output that is inserted to an alternate PCB is queued and not delivered until the operator issues a **`/STA APPC`** or **`/STA OTMA`** command.

Asynchronous APPC and OTMA messages created by a program-to-program switch can run on any IMS in the shared-queue environment.

Synchronous APPC and OTMA messages

Synchronous APPC and OTMA (send-then-commit) messages can run on any IMS in the shared-queues environment to distribute the transaction workload. Synchronous APPC or OTMA output inserted into the I/O PCB must be delivered by the front-end IMS. Therefore, APPC or OTMA route the synchronous output back to the front-end IMS, regardless of which IMS is running the transaction. Non-conversational I/O PCB reply messages (less than 61 KB) are sent to the front-end IMS using z/OS cross-system coupling facility services. Conversational I/O PCB reply messages or any synchronous output messages greater than 61 KB are sent to the front-end IMS using shared queues and a special NOTIFY message that is sent using XCF. The IMSplex supports synchronous APPC and OTMA messages when the following conditions are true:

- If AOS=Y or AOS=B, z/OS Resource Recovery Services is active.
- If AOS=Y or AOS=B, the IMS control region parameter RRS=Y is defined for all the IMS systems in the IMSplex.
- AOS=Y, AOS=F, AOS=B, AOS=S, or AOS=X is specified in the DFSDCxxx PROCLIB member data set.

If a synchronous APPC or OTMA transaction running in a back-end IMS results in asynchronous APPC or OTMA output that is inserted to an alternate PCB, APPC or OTMA must be enabled on the back-end IMS. Also, the asynchronous output is delivered to the APPC or OTMA client directly from the back-end IMS.

Synchronous APPC and OTMA transactions initiated by a non-conversational program-to-program switch to local transactions always run on the same IMS system as the transaction that initiated them. For example, if synchronous TRAN A, which can run on any front-end or back-end IMS system in the sysplex, creates synchronous TRAN B and asynchronous TRAN C, both TRAN B and TRAN C will run on the same IMS system as TRAN A.

In a shared-queues environment with APPC/OTMA SMQ enablement active, a transaction coming into IMS through APPC/OTMA can be executed by any member in the IMSplex. But when the transaction issues a program-to-program switch, the switch to transaction must be processed by the same IMS member where the switch occurred unless the control region parameters APPCASY=S or OTMAASY=S is specified.

In the APPC/OTMA shared queues environment when a transaction is processed at the back-end IMS system and performs multiple program to program switches, whether a switch to transaction can be scheduled synchronously is determined at the GU IOPCB time. However, if the control region parameter APPCASY=S or OTMAASY=S is specified, the decision as to which transaction is scheduled synchronously get made at ISRT ALTPCB time.

Related concepts

"Specifying APPC generic resource names" on page 272
Specifying an APPC generic resource name enables LU 6.2 devices to participate in the session balancing provided by a generic resource group. Specify the APPC generic resource name on either the GRNAME parameter of the LUADD statement in the APPC/MVS APPCPMxx member or with the **`SET APPC`** z/OS operator command.

Administering APPC/IMS and LU 6.2 devices (Communications and Connections)

Managing the IMS dead-letter queue

IMS maintains a dead-letter queue only for local user structures. For shared queues on the coupling facility, use the **/DIS QCNT MSGAGE** command to display potential dead-letter queues.

By examining the display output, you can determine whether a particular queue is a dead-letter queue and then take appropriate action.

Using the Expedited Message Handler

The IMS Fast Path Expedited Message Handler (EMH) can also use the shared queues in a shared-queues environment. IMS calls the Fast Path Input Edit/Routing exit routine to determine if an incoming message should be processed by Fast Path.

If Fast Path must process the message, either the local IMS system processes it, or IMS places it on the shared queues.

Fast Path can use EMHQ structures, which are defined in the same way as the MSGQ structures (in the DFSSQxxx member of PROCLIB), to share Fast Path transactions. However, an IMS system with Fast Path installed is not required to have an EMHQ structure or share Fast Path transactions. If you do not define an EMHQ structure, all Fast Path transactions are processed locally.

If an IMS system joins the z/OS cross-system coupling facility group, abends, or shuts down, it notifies all sharing IMS systems by sending a message with the program name table. Sending this message makes all IMS systems aware of which IMS systems are servicing particular programs. Likewise, when a program is started or stopped on an IMS system, it notifies all sharing IMS systems, and each IMS system updates its program name table.

If no active region is available in the IMSplex to service a program, all IMS systems that have enqueued messages for that program issue message DFS2529I to the inputting terminals. These terminals are then unlocked. Any new input for the program is rejected with message DFS2533I.

Because Fast Path input messages are not recoverable, they are not written to the SRDS data set during CQS checkpoint. You can recover EMHQ structures; however, no Fast Path input messages are retained on the SRDS data set, and only Fast Path output messages can be recovered from the EMH queues. For each terminal awaiting output from an input message that has been lost during EMHQ structure rebuild, IMS sends message DFS2766I and unlocks the terminal. In addition, IMS writes a X'5936' log record for the lost message.

The EMHQ structures can be copied, and all messages are copied from the old structure to the new one, regardless of whether they are recoverable.

For EMHQ rebuild, it is possible to specify that the rebuilt EMHQ structure be accessible while the rebuild is in progress. Use the WAITRBLD= parameter in the DFSSQxxx PROCLIB member data set.

Requeuing suspended transactions

When a transaction is suspended, it is placed on the transaction-suspend queue in the coupling facility. To make such messages available for processing, issue the **/DEQUEUE SUSPEND TRAN** command. Issuing this command moves the transaction to the transaction-ready queue or the transaction-serial queue.

The IMS that places the transaction on the transaction-suspend queue can be different from the IMS that issues the **/DEQUEUE SUSPEND TRAN** command. In this case, other IMS systems that share the message (MSGQ) structure need to issue the **/DEQUEUE SUSPEND TRAN** command.

Related reading: For more information on the transaction-suspend queue, the transaction-ready queue, or the transaction-serial queue, see "Queue types" on page 200.

Scheduling AOI transactions

To ensure that the transaction runs locally, define all AOI transactions that are scheduled by the AOI exit routine as serial. Placing the AOI transaction on the shared queue allows any IMS to process it, regardless of whether the command applies to that IMS or not.

Planning for MSC in a shared-queues environment

IMS systems in a multiple systems coupling (MSC) network can be part of a shared-queues environment, connected to IMS systems outside the shared-queues environment.

Each IMS within the IMSplex can act as any of the following:

- A front end, receiving messages from terminals or across MSC links
- A back end, processing messages received from a front end
- An intermediate subsystem, receiving messages and passing them on to other IMS systems across MSC links

Although two IMS systems can have MSC links defined, messages are placed on the shared queues from one of these IMS systems and picked up for processing by another IMS within the IMSplex—these messages are not sent over the MSC links.

Exception: If one of these IMS systems is running without shared queues enabled, the MSC links are used. This situation might occur when migrating to a shared-queues environment.

To plan for using MSC in a shared-queues environment, take each of the following actions:

- Specify all remote IMS systems on the MSNAME macro (SYSID and NAME) or the type-2 CREATE MSNAME command, and assign a unique SYSID within the local IMS. These remote MSNAMEs remain stopped and are used only to route messages from any IMS within the IMSplex to a remote IMS.
- Define MSC remote system identifiers (SYSIDs), using the MSNAME macro or the type-2 CREATE MSNAME command, for each IMS within the IMSplex.
- Within the local IMS, make all SYSIDs be unique across all IMS systems in the IMSplex and in the MSC network.
- Define remote transactions and LTERMs to the IMS that has the MSC link (the back-end IMS). Also define them to the front-end IMS. If you do not define them, IMS calls the Destination Creation exit routine (DFSINSX0) to determine whether the destination is local or remote, and to determine whether it is a transaction or an LTERM.

When an MSC link exists on the front-end IMS, you can use MSC for APPC and OTMA transactions in a shared-queues environment. The conversation is maintained with the local IMS but is not carried to the remote IMS. Therefore, the remote application program cannot issue the SETO DL/I call, or the CPI-C verbs SEND_ERROR and DEALLOCATE_ABEND. Also, if the remote IMS allocates another APPC conversation, it receives the default user ID that is associated with the MPP region, and it does not use the original APPC user ID in order to verify security.

Tuning for performance in a shared-queues environment

The structures and parameters you define for a shared-queues environment determine how the system performs.

To optimize system performance, try adjusting the following structures and parameters:

- The size of the IMS execution parameters:

```
LGMSGSZ=
QBUF=
QBUFHITH=
QBUFLWTH=
QBUFMAX=
QBUFPCTX=
QBUFSZ=
SHMSGSZ=
```

- The value of the CQS local structure definition parameter SYSCHKPT=
- The values of the CQS global structure definition parameters:

```
OVFLWMAX=
STRMIN=
```

- The size of the shared-queues structures on the coupling facility
- The size of the z/OS system log structure on the coupling facility
- The number of z/OS system log data sets that back up the z/OS system log structure
- The frequency of structure checkpoints

Recommendation for Fast Path transactions in a shared-queues environment

For shared Fast Path transactions, if you specify LOCALONLY in the Fast Path Input Edit/Routing exit routine (DBFHAGU0), do not change the size of your EMH buffer pools. The same buffer pool space is required for a shared-queues environment as for a non-shared-queues environment.

However, if you specify LOCALFIRST in DBFHAGU0 for shared Fast Path transactions and the transaction is processed locally, both a front-end EMH buffer and a back-end EMH buffer are required in the local IMS system; consequently the EMH buffer pool usage increases.

In either case, analyze EMH buffer usage and adjust the space as needed.

Related reading: For more information on tuning for performance in a shared-queues environment, see "Planning for performance in a shared-queues environment" on page 411.

Chapter 16. Data sharing in IMS environments

When more than one IMS system has concurrent access to an IMS database, those systems are sharing data, and you have a data-sharing environment. In a data-sharing environment, you manage multiple IMS systems, each running on different z/OS systems in a sysplex. Such configurations require more sophisticated operations and recovery procedures.

This information describes how to allow more than one IMS online or batch system concurrent access to data that resides in a common database. If you use IMS to control this common access to data, the data sharing support provided by Database Recovery Control (DBRC) is required.

Data sharing overview

An IMS system includes databases whose data can potentially be made available to, or shared with, all declared application programs. Access to a database is a characteristic defined in an application program's PSB. With data sharing support, application programs in separate IMS systems can concurrently access databases.

IMS systems use lock management to ensure that database changes at the segment level originating from one application program are fully committed before other application programs access that segment's data.

Data sharing among IMS systems is supported in both sysplex and nonsysplex environments.

- *Sysplex data sharing* is data sharing between IMS systems on different z/OS operating systems. IRLM uses a coupling facility to control access to databases.
- *Nonsysplex data sharing*, also referred to as *local data sharing*, is data sharing between IMS systems on a single z/OS operating system. A coupling facility can be used, but it is not required.

With data sharing, two levels of locking control are possible:

- *Database-level sharing*, in which an entire database is locked while an application program is making updates. Locking prevents concurrent database access and the scheduling of application programs that might jeopardize database integrity. For DEDB area resources, this is called *area-level sharing*.
- *Block-level sharing*, in which you can use global block locking to maintain database integrity during concurrent access of a database. The blocks are locked rather than the entire database. Multiple application programs can update a database at the same time if they are updating different blocks.

  Within a single z/OS operating system, or in a sysplex, multiple IMS online or batch systems can update a database concurrently. Within a database, resources are reserved at the block level. For OSAM databases, the block is a physical block stored on a direct access storage device. For VSAM databases and DEDBs, the block is a control interval (CI). Data integrity is preserved for the multiple IMS online or batch systems that concurrently access the shared data. Sequential dependent segments are supported. *Block-level sharing* for DEDBs is between IMS online systems only.

Data access is protected so that:

- One IMS system is authorized to update the database, and other authorized IMS systems can have read-only access.
- Multiple authorized IMS systems can concurrently schedule the database with read or read-only access.

In a data-sharing environment, an IMS system can be online or batch. For area-level sharing, participating IMS systems must be online. IMS utilities are considered as batch systems that work with database-level sharing.

Some differences exist in support for data sharing configurations. Generally, a complete database is regarded as one data resource. When the database is invoked within an IMS online system, or as a batch IMS system, it must be available for an individual application program to process. However, a database might not be available if, for example:

- The database is used exclusively by one IMS system.
- The database is flagged as needing recovery.
- Backup procedures are in process.

With data entry databases (DEDBs), however, the data resource is divided. In a DEDB, each individual area is considered a unit of data resource. In this topic on data sharing, when we use the term "database," it means a DEDB area, unless otherwise noted.

With High Availability Large Databases (HALDBs), the database can be divided into one or more partitions. In a HALDB database, each individual partition is considered a unit of data resource. When the term "database" is used in reference to data sharing, it means a HALDB partition, unless otherwise noted.

Restrictions associated with data sharing are:

- Batch IMS support excludes the use of MSDBs and DEDBs.
- Only IMS online systems that use Fast Path can share DEDBs.
- Data sharing support excludes MSDBs and GSAM databases.

DBRC and data sharing support

Concurrent access to databases by systems in one or more z/OS operating systems is controlled with a shared Database Recovery Control (DBRC) RECON data set. IMS systems automatically sign on to DBRC, ensuring that DBRC knows which IMS systems and utilities are currently participating in shared access.

Subsequently, a system's eligibility to be authorized to access a database depends on the declared degree of sharing permitted and other status indicators in the RECON data set.

To maintain data integrity, status indicators in the RECON data set control concurrent access and recovery actions for the databases. This common RECON data set is required in a data sharing IMSplex because a given database must have a DMB number that uniquely identifies it to all the sharing IMS systems. The DMB number that DBRC records in its RECON data set is related to the order in which databases are registered to DBRC. Using multiple RECON data sets can result in the same DMB number existing in each RECON data set for different databases. This condition can result in damage to databases.

Registering with DBRC

Databases that are to take part in data sharing must be registered in RECON. Each registered database has a current status that reflects whether it can take part in sharing and the *scope* of the sharing. The concept of scope combines several ideas:

- The type of access: read or update
- Whether more than one access can occur within the database simultaneously
- Whether an IMS system that needs access is in the same or a different z/OS operating system

You specify the level of sharing on an individual IMS database using the SHARELVL specification on the INIT.DB command.

Databases can be shared at the database level, at the block level, or not at all. The type of sharing used for a database should be based on the data integrity and availability needs of all the application programs using it.

When two or more IMS systems are executing concurrently and sharing data at the database level, they can be in the same or different z/OS operating systems.

Figure 32 on page 225 shows a database as a shared resource and also illustrates area-level sharing. One or more areas making up the DEDB could be a shared resource. The RECON data set is also shared as a VSAM key-sequenced data set (KSDS), with status being read and updated by a DBRC control portion of the online or batch IMS. For IMS online systems, DBRC executes in a separate region, automatically started by the control region at initialization. For a batch IMS, DBRC control is contained within the batch region, as it is for database-level sharing.

The RECON data set keeps track of the:

- Sharing level allowed for each database
- Databases or areas currently authorized for processing
- IMS systems that are involved
- Status of all of those systems
- Database status from a recovery viewpoint

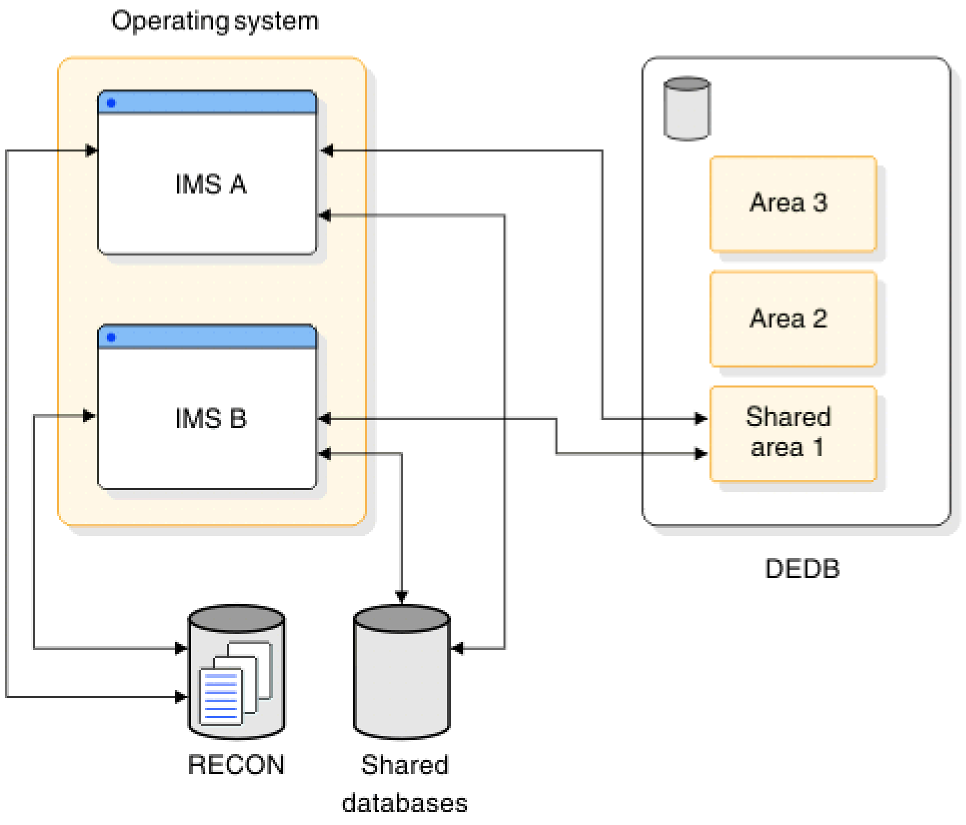


Figure 32. Database-level sharing

Naming conventions for data-sharing environments

If you decide to use data sharing and have additional databases to control with DBRC, you must review the data set naming conventions established for database data sets.

Consider indicating that a database participates in data sharing using the chosen resource names. An application program that accesses a shared database might also have the PSB name indicate a read-only property.

Review naming conventions for the following resources:

- Databases, their DD names and data set names
- Image copy and change accumulation data set names
- Online log data sets (OLDSs)
- System log data sets (SLDSs)
- PSB and application program names
- Transaction codes
- Identifying names for IMS systems and IRLMs

Database integrity without data sharing support

Without data sharing support, the responsibility for database integrity lies with administration and operations.

These are the two situations where data integrity is not protected:

- In a single z/OS operating system, with the database JCL specifying a disposition of SHR, multiple IMS online or batch systems executing simultaneously might concurrently access the database without integrity.
- If multiple IMS systems are executing in more than one z/OS operating system, and a database is on shared DASD, those IMS systems can concurrently access the database. This concurrent access cannot be directly controlled by JCL, even when a disposition of OLD is specified.

Important: It is strongly recommended that you use IRLM to maintain database integrity when data might be concurrently accessed.

How applications access data

Application programs can access data in three ways: update, read, or read-only access.

You indicate the type of access on the PROCOPT keyword as part of the group of statements that comprise the program communication block (PCB) for a particular database access. These processing options for an application program are declared in the program specification block (PSB) and express the data access and alteration intent of the application program.

If an application program is to insert, delete, replace, or perform a combination of these actions, the application program has *update access*, which is specified with the PROCOPT=A keyword. An online application program that has exclusive access is also interpreted as having update access; update access is specified with the PROCOPT=E keyword.

Application programs that need access to a database but do not update the data can do so in two ways. They can access the data with the assurance that any pending changes have been committed by the application program that initiated the change. This type of access is *read access* and is specified with the PROCOPT=G keyword. Alternatively, application programs can read uncommitted data, if the application program does not specify protection of data status. This type of access is *read-only access* and is specified with the PROCOPT=GO keyword.

For more information on PROCOPT keyword values, see *IMS Version 15.2 Application Programming*.

How IMS systems share databases

This topic describes how database access is established, data sharing at both the block and database level, and how data access is controlled by each level of sharing.

Establishing database access

To specify how an IMS system that is requesting access to a database plans to use the database, use the ACCESS keyword parameter in the DATABASE macro.

You use the ACCESS keyword in the **/START** command to show how the IMS system requesting access to a database plans to use the database. The presence of any update intent means the whole system requires update access for that database. If any database PCB requires exclusive use, the whole system requires update access for that database. If no updating application programs need update access, you must find out what kind of read access is required. If the online system is to use block-level sharing, you usually declare read access.

The access mode can be *update* (UP), *exclusive* (EX), *read* (RD), or *read-only* (RO).

If the IMS system requires exclusive use of a database, it can declare its access as *exclusive*. With exclusive access, the IMS system can insert, delete, replace, or perform a combination of these actions. Exclusive access, therefore, precludes data sharing in block-level or database-level configurations.

If the IMS system needs to insert, delete, replace, or perform a combination of these actions and must participate in data sharing, it can declare its access as *update*.

IMS systems that plan to read from a database, but not update the database, can use either read or read-only access. With *read access*, the IMS system can access the data with the assurance that any pending changes have been committed by the application program that requested the change.

With *read-only access*, the IMS system is allowed to read uncommitted data.

Important: Using read-only access is not recommended for a production environment.

How you specify database access for an IMS system depends on whether you are running a batch or online system.

If you plan to use the online version of the Database Image Copy utility, you do not need to specify ACCESS=UP. Although the utility requires a somewhat limited access while it is creating an image data set, the access and authorization are managed by DBRC.

Declaring database access for IMS batch systems

For an IMS batch system, the database access directly corresponds to the highest PROCOPT value specified in the application program's PCBs or SENSEG statements. For example, if one PCB has PROCOPT set to G, and another PCB for the same database has PROCOPT set to I, the value of I is used; that is, the access for that database is update.

Declaring and changing database access for online systems

For an online IMS, you specify access with the ACCESS keyword in the DATABASE macro during system definition. The access is declared for each individual database and reflects a desired access suitable for all the application programs that might be scheduled against that database.

Related reading: For a detailed description of the ACCESS keyword, see *IMS Version 15.2 System Definition*.

To dynamically change how an online IMS system accesses an individual database, use the **/START** command, which is described in *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Using IRLM with database-level sharing

When IMS uses the Internal Resource Lock Manager (IRLM) for locking services, IRLM provides all locking services.

If IRLM was not activated with the IRLMNM parameter in the IMSCTRL macro, and you want to activate IRLM, code IRLM=Y in the execution JCL. If the IRLMNM parameter is not specified in either the IMSCTRL macro or the execution JCL, and IRLM=Y is specified in the execution JCL, the IMSCTRL macro uses the default IRLM named ('IRLM').

When all of the IMS batch and online instances use one or more communicating IRLMs with database-level sharing, database extensions and buffer invalidations are notified to the read-only IMS instances. This is the same as full block-level data sharing, but without the locking activity.

When you use IRLM for database-level sharing, application programs that read without integrity read valid, but perhaps uncommitted, data more frequently. This does not provide any guarantees that read without integrity problems are solved (see *IMS Version 15.2 Application Programming*), but it does improve database-level sharing. Because of the order in which all changed blocks or control intervals (CIs) are written to DASD, including when they are written, an application program reading without integrity can still experience inconsistencies from one execution to another.

Furthermore, uncommitted data might lead to looping behavior that cannot be detected by IMS, which will require IMS to be restarted.

Important: Not recognizing database extensions can lead to data being overwritten. This data cannot be recovered.

IMS allows use of the coupling facility for buffer coherency for both database-level sharing and block-level data sharing.

Block-level sharing

Block-level sharing requires the Internal Resource Lock Manager (IRLM). If each z/OS operating system contains an IMS online system participating in block-level sharing, an IRLM component must exist in each z/OS operating system.

IRLM manages all requests for locks that control access to resources participating in data sharing.

Each IRLM responds to the requests that originate from its associated IMS systems. To do this, it maintains records of the participating IMS systems, as well as the status of locks that are held and waiting.

When an IMS online system uses IRLM, IRLM also provides the lock management that controls database resources that are concurrently accessed by application programs within the online system. In this way, the IRLM associated with an IMS online system provides almost all locking services for that system.

The IRLMs communicate with each other and manage the status of locks held for different database locks.

When using IRLM, you must define DBRC-registered SHARELVL=1 VSAM databases to VSAM with the appropriate SHAREOPTIONS required for block-level data sharing. This might require you to make a VSAM definitional change.

Multiple IRLMs in sysplex data sharing

Multiple IRLMs can take part in the control of data sharing at the block level when IMS systems are executing in the same z/OS operating system.

This could be the case when IMS online systems are being tested for their use of data sharing before one of the systems is installed on a second z/OS operating system.

A special case of sysplex data sharing occurs when multiple IRLMs execute on a single z/OS operating system. In this case, you declare global sharing and give each IRLM a unique z/OS subsystem name and an identifying IRLM number (ID).

Setting up IRLM procedures

For each IRLM component that can be activated, you need a procedure that is invoked when the system console operator enters the z/OS **START** command.

An example of one such procedure is DXRJPROC, described in *IMS Version 15.2 System Definition*. If you plan to use more than one IRLM procedure on a single operating system, include additional members in IMS.PROCLIB or SYS1.PROCLIB. Additional IRLMs require a unique identifier and any desired changes in parameter values. The procedure name cannot be the same as the IRLM subsystem names established for z/OS.

For each IRLM component, the procedure parameters specify:

- The component's identifying name
- The scope of its control
- Sysplex data-sharing information
- IRLM actions in failure situations
- The amount of virtual storage
- Performance factors

The PARM1= and PARM2= positional parameters for the region that executes in support of block-level sharing are shown in the following table.

Table 18. Categories and purpose of IRLM region parameters

Category	Parameter (PARM1 and PARM2)	Purpose
Component name	IRLMID	An identifier for this IRLM.
	IRLMNM	z/OS subsystem name.
Scope of control	SCOPE	Inter-z/OS communication or intra-z/OS control.
Data sharing	IRLMGRP	Name of the z/OS cross-system coupling facility group.
	MAXUSRS	Maximum number of users in the group.
	LOCKTAB	Lock structure to be used by the group.
Storage	MAXCSA	Maximum amount of z/OS common service area (CSA) to be used. This parameter does not apply to IRLM 2.2 and above.
	PC	Specifies control blocks in private storage. IRLM 2.2 always runs with PC=YES.
	PGPROT	Determines whether the IRLM load modules that are resident in common storage are placed in MVS PAGE PROTECTED STORAGE, default is YES.
Performance	DEADLOK	Deadlock detection timing.
Tracing	TRACE	Trace types DBM, XCF, and SLM turned on at IRLM initialization (EXP, INT, and XIT are always on).

Identifying IRLM

The IRLMID parameter specifies a decimal number (from 1 to 255) for IRLM identification. The IRLMID gets converted to a hexadecimal equivalent. A unique number must be assigned for each IRLM. IRLM error and status messages include the subsystem name and ID of the IRLM that issued the message.

The IRLMNM parameter specifies the 1- to 4-byte z/OS subsystem name that is to be assigned to this IRLM. Unless more than one IRLM is used at the same time on the same system, you can use "IRLM" as the subsystem name in each operating system. When IRLMs are used concurrently on a system, the subsystem names must be unique.

Specifying IRLM scope

You must specify what kind of data-sharing control is required of the IRLM component–whether sysplex or nonsysplex sharing is to be used. The SCOPE parameter accomplishes this. SCOPE=GLOBAL specifies sysplex data sharing is to be controlled. This is sharing of resources among multiple systems.

SCOPE=LOCAL limits data sharing to nonsysplex data sharing. This is sharing of resources among IMS systems all identified to a single IRLM. If a coupling facility is not available, SCOPE=LOCAL is required. SCOPE=NODISCON specifies sysplex data sharing is to be controlled with special DISCONNECT rules. SCOPE=GLOBAL or NODISCON requires that a coupling facility be defined.

IRLM deadlock management

IRLM provides deadlock management. Using the DEADLOK parameter in the DXRJPROC procedure, you can specify the local deadlock-detection interval and the number of local cycles that are to occur before IRLM initiates a global deadlock detection. You can specify the local deadlock-detection interval in seconds or milliseconds. You can also dynamically modify the DEADLOK values using the z/OS command **`MODIFY IRLMPROC`**.

In a data-sharing environment, the participating IRLMs synchronize their DEADLOK parameters to the highest values provided. You should make this parameter the same for each IRLM in the data-sharing group.

Related readings:

- For more information about the DXRJPROC procedure and the DEADLOK parameter, see *IMS Version 15.2 System Definition*.
- For more information about the command `MODIFY IRLMPROC`, see *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*.

System initialization for IRLM

To initialize the system for IRLM, you must define IRLM as a z/OS subsystem.

Defining an IRLM as a z/OS subsystem

Your installation must assign a 4-byte subsystem name. You can use "IRLM" as the name, which assists the system operator in recognizing system messages sent to the console. This name must be unique to the z/OS system. If more than one IRLM subsystem is to run in an operating system, the names must be different.

An additional identification for each IRLM is a number in the range 1 to 255. The identification is required to internally differentiate between IRLMs. For example, two IRLMs that are to share data could have IDs of 1 and 2. Status and error messages sent to the system console operator contain the IRLM subsystem name and ID. Two IRLMs executing in a single z/OS system, or in different z/OS systems that are members of the same data-sharing group, require different ID numbers.

Another consideration for the IRLM subsystem name is that it must not be the same as that used for the startup procedure. This procedure is a member of SYS1.PROCLIB and is used by the operator in the **START** command. A unique START procedure must exist for each IRLM subsystem. The procedure library members are also distinguished by the different ID numbers and other control parameters.

The execution of the IRLM requires that the subsystem be added to the z/OS program properties table (PPT).

The system initialization routine for the subsystem must be null.

Related readings:

- For more information on defining the IRLM as a subsystem and adding an entry to the subsystem name table, see *z/OS MVS Initialization and Tuning Guide*.
- For more information on adding IRLM to the PPT, see "Adding the IRLM entry to the z/OS Program Properties Table" on page 114.

Allowing for IRLM trace output printing

To provide a trace of the activity occurring in the IRLM subsystem, you use the z/OS Component Trace facility or the TRACE=YES option of DXRJPROC, the IRLM procedure.

CTRACE records

IRLM produces trace output in z/OS component trace (CTRACE) format. This allows you to use IPCS CTRACE format, merge, and locate routines to process the buffer data. Both the IRLM trace buffers in the dump and external writer data set can be formatted using IPCS. The IRLM trace formatting load module (DXRRLFTB) and buffer find routine load module (DXRRL186) must be available for IPCS.

Arranging for formatted dump output

IRLM uses the SDUMP facility of z/OS. The offline printing of dump output is done using z/OS print dump.

Related reading::

- For more information on SDUMP, see *IMS Version 15.2 Diagnosis*.
- For information on DXRJPROC, the IRLM procedure, see *IMS Version 15.2 System Definition*.

Defining an IRLM for sysplex data sharing

If you are implementing a sysplex data-sharing environment, you must define the data-sharing group to which IRLM belongs, the lock structure to be used by that group, and the maximum number of users in that group.

Defining the data-sharing group

For an IRLM to belong to a data-sharing group, you must specify the name of the data-sharing group and the name of the lock structure in the IRLM startup procedure. All IRLMs in this group can share the same data. Each IRLM in the group must:

- Have a unique IRLMID
- Specify the same data-sharing group name using the **GROUP** parameter
- Specify the same lock structure using the **LOCKTAB** parameter

Note: The **GROUP** parameter in the IRLM startup procedure is identical to the **IRLMGRP** parameter that is used in the **START** *irlmproc* command.

Although you can specify these definitions on the IRLM startup procedure, the recommended method is to define them using the CFNAMES control statement. The CFNAMES control statement defines the data-sharing group for a sysplex data-sharing environment. If you do not use the CFNAMES control statement, the lock structure name is pulled from the IRLM startup procedure.

Note: If you use the CFNAMES control statement to define the lock structure name, the **LOCKTAB** parameter is ignored.

Defining the lock structure

Each IRLM participating in a sysplex data-sharing group (specifying the same z/OS cross-system coupling facility name on the GROUP=IRLMDS parameter) must specify the same lock structure. You can specify the lock structure name by using the **LOCKTAB** parameter or the CFNAMES control statement in IMS PROCLIB DFSVSM*xx* member.

Defining the number of users in a data-sharing group

You must specify a maximum number of users for the data-sharing group. You do this by specifying a value from 2 to 32 on the MAXUSRS parameter.

Related reference

DFSVSMxx member of the IMS PROCLIB data set (System Definition)

Data sharing at the database level with update activity

This type of sharing allows one IMS batch or online system to have update access. All other IMS systems must have read-only access.

For full-function databases, uncommitted data can be read. For Fast Path, uncommitted data does not exist, but all of the updates for a transaction might not yet have been written. This can cause invalid pointers to be referenced when Get Next processing is performed.

Normal application program isolation protects the integrity of the updating system. However, you can use IRLM for lock management on an IMS online system. The following figure illustrates an online system with update access sharing data with two batch systems that are authorized for read-only access to the same data.

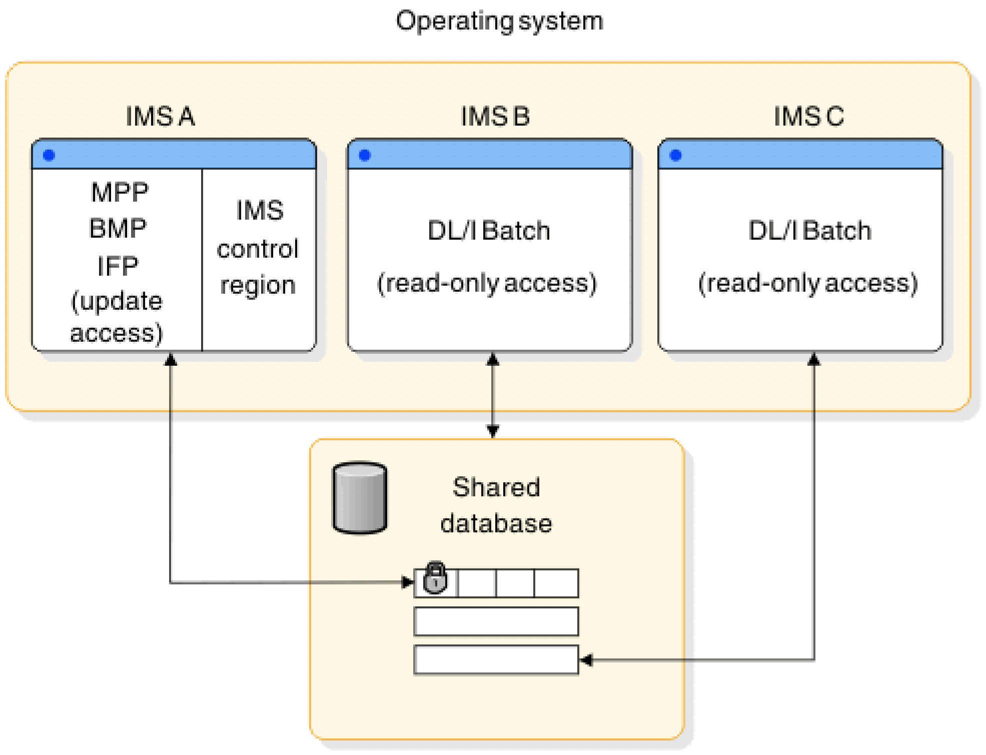


Figure 33. Example of data sharing at the database level with update access

Data sharing at the database level with multiple readers

This type of sharing allows multiple authorized IMS systems to read data concurrently with read or read-only access.

A sample configuration is shown in the following figure. Data integrity is not compromised, because no updates are allowed in this configuration.

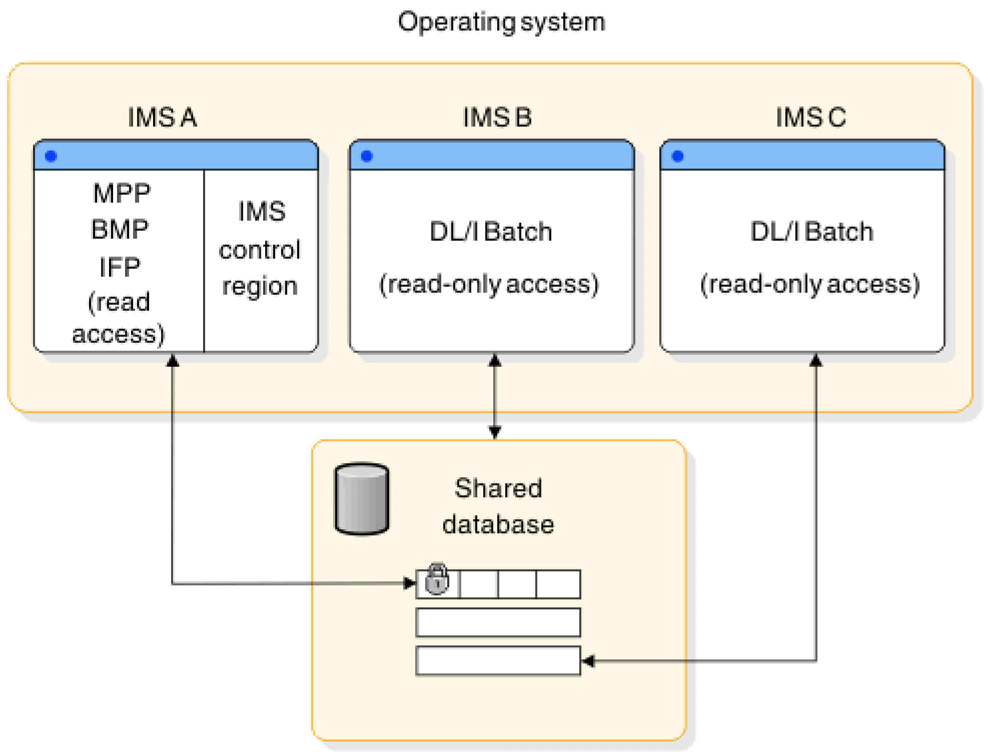


Figure 34. Example of data sharing at the database level with read access

Data sharing at the block level

Online IMS systems with full update capability can share data among themselves and can allow other batch IMS systems to read committed data or update the data.

The batch system can have the data integrity protected or use read-only access. This type of configuration is illustrated in the following figure.

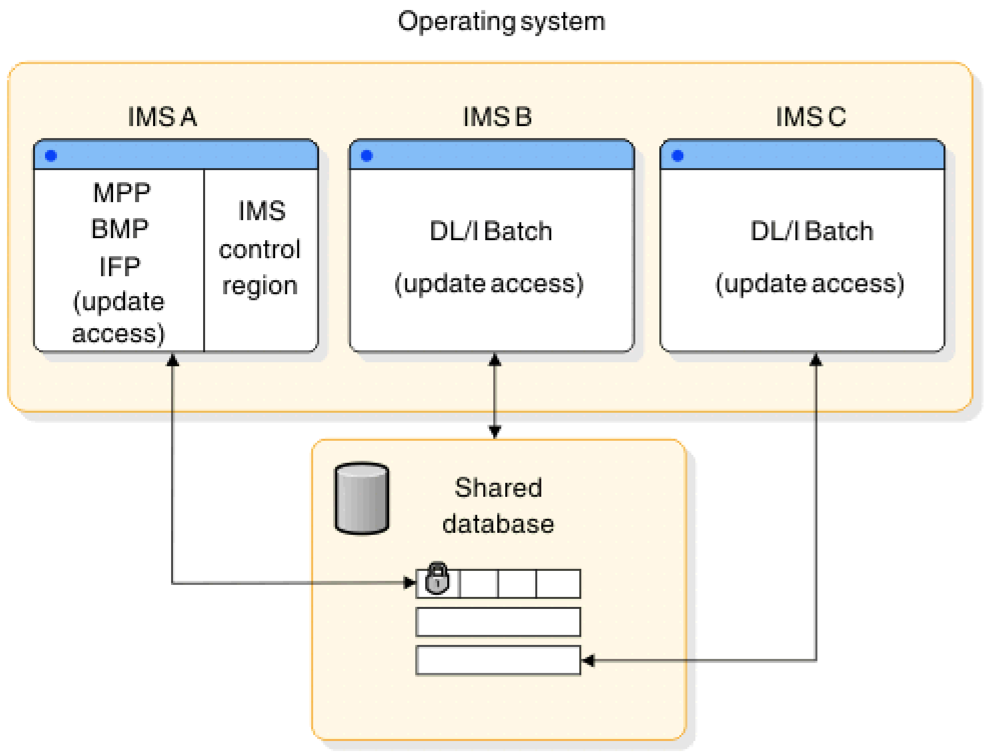


Figure 35. Example of data sharing at the block level with update access

Tailoring IMS systems that share data

Use the system definition macro and initialize the DBRC data sets and JCL to tailor an IMS system.

To tailor an IMS system to meet your data-sharing requirements:

- Use system definition macros to:
  - Include DBRC data-sharing support for IMS batch systems
  - Include IRLM (for block-level sharing)
  - Declare database access attributes
- Initializing DBRC data sets and JCL. See Chapter 44, "Initializing and maintaining the RECON data sets," on page 531.
- Tailoring the execution JCL and virtual storage usage. See "Tailoring execution JCL" on page 235.

Installation tasks

Before executing the IMS systems that use data sharing, you must complete several installation-related tasks.

The following table lists the tasks, indicates whether a task affects data sharing, and identifies other tasks that must be completed.

Table 19. Installation steps with additional data-sharing activity

Step	Installation task	Affects systems without data sharing	Affects systems with data sharing	Additional task
1	Build system libraries—IRLM	No	Yes	Add IRLM to system library.
2	Allocate and catalog IMS data sets	Yes	Yes	Prepare RECON data set. Initialize DBRC controls. Coordinate DL/I exits.
3	Prepare system definition and JCL	Yes	Yes	Macros and JCL changes. Perform system definition.
4	Tailor the z/OS operating system	No	Yes	Add IRLM subsystem.
5	Tailor IMS performance options	Yes	Yes	Define buffers.
6	Build DBDLIB	No	No	None.
7	Build PSBLIB	Yes	Yes	Review PROCOPT values.
8	Build ACBLIB.	Yes	Yes	Perform ACBGEN
9	Selections for dynamic allocation	Yes	Yes	Review options.
10	Prepare MFS libraries	No	No	None.
11	Prepare program libraries	Yes	Yes	Review call sequences.
12	Perform initial database loading	No	No	None.
13	Establish security	No	No	None.

Table 19. Installation steps with additional data-sharing activity (continued)

Step	Installation task	Affects systems without data sharing	Affects systems with data sharing	Additional task
14	Initialize IMS.MODSTAT	Yes	Yes	Coordinate ACBLIB use and DBDLIB and PSBLIB.
15	Copy staging libraries to active libraries	Yes	Yes	Coordinate ACBLIB use.

DBRC and IRLM support for batch systems in a data-sharing environment

By default, DBRC provides data-sharing support for your batch and utility regions.

To change the default, you can either:

- Assemble and bind DFSIDEF0 into the IMS execution library (If you specify RMODE, you must specify RMODE=24 and AMODE=24)
- Override the value in a DFSPBxxx IMS PROCLIB member data set
- Override the value in JCL

To define the default IRLM support specifications for batch systems in a data-sharing environment, use the IRLM= keyword on the IMSCTRL macro. You can override specifications made in the IMSCTRL macro by using the IRLM= keyword in the batch JCL.

In batch systems, you can specify IRLM=Y or IRLM=N. If you specify IRLM=N:

- IRLM does not perform locking for the batch system.
- The batch system cannot participate in block-level sharing.

When batch systems do not use IRLM, DBRC still ensures the integrity of databases. DBRC authorizes batch jobs that have update access to a database only if all other IMS systems and batch jobs that are currently authorized by DBRC to the database have read-only access.

Excluding a database from data sharing

If you specify ACCESS=EX, or leave the value for ACCESS= blank, the named database is not accessed concurrently by any other IMS system.

Register the database with DBRC, specifying a share level of zero.

Databases who data does not need to be shared with any other IMS systems are owned by the IMS online system you define. If you do not want to use DBRC to control authorization to access these databases, do not register them.

Tailoring execution JCL

The system log is required for any batch IMS that has update intent against any database and operates with active DBRC. The system log is not required if the batch system only uses read or read-only access intent.

When you are preparing a configuration that includes data sharing, you must:

- Specify system data sets.
- Specify the databases with DISP=SHR.
- Specify the placement of database data sets and device choices to suit the physical paths to the data.
- Prepare IRLM procedures.

The ACBLIB member online change process is coordinated among all IMS systems sharing the OLCSTAT data set. The system data sets that must be coordinated so that their status is common to all systems participating in data sharing are:

- RECON data sets
- Database data sets, with DISP=SHR specified
- IMS.ACBLIB, PSBLIB, and DBDLIB
- Libraries holding randomizing routines and DL/I exit routines

You also must set up IRLM procedures for your data-sharing configuration. For information on preparing the IRLM procedures for a data-sharing environment, see "Setting up IRLM procedures" on page 228.

Tailoring the z/OS operating system

Because block-level sharing requires the use of IRLM installed as an independent component under the z/OS operating system, several tailoring actions must be performed.

You must:

- Coordinate VSAM data set definitions.
- Tailor the z/OS system for IRLM.
- Define an IRLM if you are sharing data across a sysplex.

Coordinating VSAM data set definitions with share options

When your database access method is VSAM, the declaration of the data set indicates to VSAM what degree of shared access is required.

The SHAREOPTIONS parameter values (in the DFSMS DEFINE CLUSTER keyword) and the type of sharing they specify is shown in the following table.

Table 20. SHAREOPTIONS parameter specifications

SHAREOPTIONS value	Type of sharing
(1,3)	No sharing, single updater, or multiple readers
(2,3)	Single updater and multiple readers
(3,3)	Multiple updaters and multiple readers

For databases that are to participate in block-level sharing and use the VSAM access method, you must include the SHAREOPTIONS (3,3) parameter when defining the data sets. The RECON data set is also accessed as a KSDS and requires SHAREOPTIONS (3,3).

Related reading: For preparation of DFSMS Access Method Services statements that declare and catalog the share options, see *z/OS DFSMS Access Method Services for Catalogs*.

When using IRLM, you must define SHARELEVEL=1 VSAM databases to VSAM with the appropriate SHAREOPTIONS required for block-level data sharing. This might require you to make a VSAM definition change.

Tailoring the z/OS system for IRLM

Several other system initialization activities are required for the IRLM component:

- Defining the IRLM as a z/OS subsystem
- Allowing for IRLM trace output
- Arranging for formatted dump output
- Defining an IRLM for sysplex data sharing

These activities are described in "System initialization for IRLM" on page 230.

Starting and stopping data sharing

Starting and stopping data-sharing operations involves starting and stopping the IRLM and database processing.

Starting the IRLM

Start the IRLM from the system console of each participating system with the **START** *irlmproc,parms* z/OS command.

irlmproc is the unique name of the IRLM procedure, and *parms* represents one or more IRLM control parameters that override the command defaults.

Start the IRLM subsystems before starting the batch or online IMS.

If IRLM has not completed its initialization when the IMS system tries to connect to it, IMS issues a DFS039A message that says the IRLM is not active, and asks the operator to reply to the message: retry, cancel, or dump.

- If you have not started the IRLM, issue the z/OS **START** command, wait until the IRLM is active, and reply RETRY. IMS then tries to connect to the IRLM.
- If you have a problem coordinating startup, not merely starting the IRLM, reply CANCEL to cause the IMS system to terminate with abend U0039 .

  Example: Assume the following environment:

  – On processor J, an IMS system (IMSJ) uses an IRLM (JRLM).

  – On processor K, an IMS system (IMSK) uses an IRLM (KRLM).

  Next, assume the following scenario:

  1. A hardware error occurs on processor K.
  2. You try to restart IMSK on processor J to perform database backouts (but you do not change the IRLM name parameter, IRLMNM=KRLM).
  3. Message DFS039A indicates that KRLM is not active.

  Reply CANCEL in this case to immediately terminate the IMSK startup. Then, after changing IRLMNM= to JRLM, restart IMSK on processor J.
- If you reply DUMP, IMS terminates with abend U0039 and a storage dump.

The z/OS console operator must watch for messages originating from the IRLM. These messages have the three-letter prefix DXR.

Stopping the IRLM

You can terminate an IRLM normally only when all IMS systems that are using it have terminated. To stop an IRLM normally, use the **STOP** *irlmproc* z/OS command.

To stop an IRLM abnormally, use the z/OS command **MODIFY** *irlmproc***,ABEND,NODUMP**. To stop an IRLM abnormally and generate a dump, use the z/OS command **MODIFY** *irlmproc***,ABEND**.

Starting and stopping database processing

In a data-sharing environment, IMS type-1 commands can act *locally*, that is, affect the local system only, or can act *globally*, that is, affect all sharing IMS systems in the sysplex.

The **UPDATE** command can be routed to individual IMS systems or to all IMS systems in an IMSplex, using an automated operator program like the TSO SPOC.

An IMS command with the LOCAL keyword (or without the GLOBAL keyword) affects only the online IMS system on which you enter it. An IMS command with the GLOBAL keyword first affects the IMS system on which you enter it and then, if it is successful, affects other sharing IMS systems in the sysplex.

When you enter a **/START DB** or **UPDATE DB START(ACCESS)** command, either locally or globally, DBRC makes sure that each registered database does not need recovery or backout. The IRLM communicates global commands only in a block-level data-sharing environment.

The following table summarizes the commands that affect data sharing when issued for registered databases. A command entered for a database affects full function transactions; a command entered for an area affects Fast Path transactions.

Table 21. Database control commands across IMS systems

Command	Action by DBRC	Effect in local IMS system	Effect in sharing IMS systems
Local			
/RMCHANGE	Can change share level for IMS systems using the database	None	None
/START AREA | DATABASE LOCAL or **UPDATE**	Ensures that the database or area does not need recovery or backout, and records database start	Allows transactions to access the database or area, and can change the access intent if you use the ACCESS= keyword	None
Global			
/DBDUMP AREA | DATABASE GLOBAL	Ensures that the database or area does not need recovery or backout, sets a return code for any unauthorized batch IMS systems, records database close, and changes access level to read access	Prevents online updating access	Processes command as if entered locally
/DBRECOVERY AREA | DATABASE GLOBAL	Prohibits further authorization	Prevents allocation of database	Processes command as if entered locally
/STOP AREA | DATABASE GLOBAL	Records database status as stopped (restricts further authorization)	Stops transactions from accessing the database or area	Processes command as if entered locally
/STOP ADS	Records area data set as unavailable	Stops the area data set	Processes command as if entered locally
/START AREA | DATABASE GLOBAL	Ensures that the database or area does not need recovery or backout, resets the prohibit-further-authorization and read-only flags, records database start, and can change the access level	Allows transactions to access the database or area	Processes command as if entered locally

Example: You enter the **/START** command with the GLOBAL keyword on one IMS system and specify several database names, the IRLM transmits the command to other sharing IMS systems, deleting the names of any databases that are invalid for the local system before it transmits the command, and all sharing IMS systems process the command. In those online data-sharing systems, you see the DFS3334I message followed by the DFS3328I message, which tell you that the global **/START** command has started and has then completed. The messages include the database names that you include in your command.

If you omit the GLOBAL keyword (or specify the LOCAL keyword), the command applies only to the local online IMS system and does not affect access by any other IMS system.

When you enter a **/STOP** or **UPDATE** command locally, no interaction occurs between IMS and DBRC.

Related reading: For the format of these commands in a CICS environment, see *CICS Transaction Server for z/OS CICS Supplied Transactions*.

Monitoring data-sharing systems

To monitor data sharing, you obtain information on the status of the following: IRLM, IMS systems and databases, the RECON data set, and coupling facility structures.

Obtaining the status of IRLM activity

Use the **MODIFY** z/OS command to display the status of an IRLM.

To display the status of an IRLM on either your system or on another connected system, enter the z/OS command, where *irlmproc* is the name of the procedure that you used to start the IRLM, and *irlmx* is the name of the IRLM whose status you want to display:

```
MODIFY irlmproc,STATUS,irlmx
```

You can use the ALLD keyword to display the names and status of every IMS identified to an IRLM in a data-sharing group. Or, you can use the ALLI keyword to display the names and status of every IRLM in a data-sharing group.

You can also trace IRLM activity. see "Tracing IRLM activity" in *IMS Version 15.2 Operations and Automation*.

Displaying components and resources

Monitoring components and resources in a data-sharing environment requires the same kinds of procedures as in a non-sharing environment.

The following table lists keywords for the **/DISPLAY** command that you can use to obtain information about various IMS resources.

Table 22. /DISPLAY command keywords that provide information about IMS resources

Resources	/DISPLAY command keywords
Active control regions	ACTIVE REGION
Active jobs	ACTIVE
Programs, transactions, and conversations	CONVERSATION PROGRAM PSB STATUS PROGRAM STATUS TRANSACTION SYSID TRANSACTION TRANSACTION
Databases	DATABASE AREA STATUS DATABASE

Table 22. /DISPLAY command keywords that provide information about IMS resources (continued)

Resources	/DISPLAY command keywords
Terminals, lines, links, and nodes	ACTIVE DC ASSIGNMENT LINE ASSIGNMENT LINK ASSIGNMENT NODE LINE LINK LTERM MASTER MSNAME NODE PTERM STATUS LINE STATUS LINK STATUS LTERM STATUS MSNAME STATUS NODE STATUS PTERM
External subsystems and connections to external subsystems	CCTL OASN SUBSYS SUBSYS
VTAM	TIMEOVER

For example, you can issue the **/DISPLAY DB** command after you issue a **/START DB** command to determine whether the database is started.

The following table lists keywords for the **QUERY** command that you can use to obtain information about various IMS resources.

Table 23. QUERY command keywords that provide information about IMS resources

Resources	QUERY command keywords
Active control regions	MEMBER
Members in the IMSplex	IMSPLEX
Transactions	TRAN
Databases	DB AREA
Global online change status	OLC
Language environment (LE) runtime options	LE
Online reorganization (OLR) status	OLREORG
RM resource structure	STRUCTURE

Table 23. QUERY command keywords that provide information about IMS resources (continued)

Resources	QUERY command keywords
Runtime resources and descriptors whose definitions have not been exported to the IMSRSC repository since they were created or last updated	DB SHOW(EXPORTNEEDED) DBDESC SHOW(EXPORTNEEDED) PGM SHOW(EXPORTNEEDED) PGMDESC SHOW(EXPORTNEEDED) RTC SHOW(EXPORTNEEDED) RTCDESC SHOW(EXPORTNEEDED) TRAN SHOW(EXPORTNEEDED) TRANDESC SHOW(EXPORTNEEDED)

For example, you can issue the **QUERY DB** command after you issue an **UPDATE DB START(ACCESS)** command to determine whether the database is started.

Related reading: For detailed information about the **/DISPLAY**, **/START**, **QUERY**, and **UPDATE** commands, see *IMS Version 15.2 Commands, Volume 1: IMS Commands A-M* and *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Monitoring information in the RECON data set

Evaluate information from the RECON data set on a daily basis to understand the status of the total data-sharing workload and response times for all critical data-sharing transactions.

You can use the historical trace of events recorded in the RECON data set to find out the intervals when shared data was active. The **LIST** command or **QUERY** request can give you a comprehensive listing of events. For sysplex data sharing, you need a listing from all operating systems. Events such as IMS signon and time stamped database data set OPEN/CLOSE can be traced.

Monitoring structures on a coupling facility

The **DISPLAY XCF,STRUCTURE** and **DISPLAY XCF,STRUCTURE,STRNAME=**z/OS operator commands are especially useful in monitoring structure activity.

These commands let you look at structures on a coupling facility to determine resource status and, for failures, gather information for problem determination.

For detailed information on these commands, see the appropriate z/OS publication.

DISPLAY XCF,STRUCTURE command

Use this command to display the status of structures defined in your active policy.

The following example shows an example of output from this command.

```
IXC359I  11.09.26  DISPLAY XCF 376
STRNAME           ALLOCATION TIME  STATUS
CF01            02/17/96 17:03:49 ALLOCATED
CF02               --       --    NOT ALLOCATED
CF03               --       --    NOT ALLOCATED
CF04               --       --    NOT ALLOCATED
OSAMSESXI       02/17/96 17:02:54 ALLOCATED
                                  REBUILDING
                                  REBUILD PHASE: QUIESCE
VSAMSESXI       02/17/96 17:03:03 ALLOCATED
```

In the example:

STRNAME
Is the name of a structure.

ALLOCATION TIME
Is a time stamp indicating when the structure was allocated in the coupling facility.

STATUS
Is the current status of the structure. A structure can be in a number of different states, such as ALLOCATED, NOT ALLOCATED, or ALLOCATED REBUILDING.

The displayed information can help you determine whether the appropriate IRLM, OSAM, VSAM, shared MSGQ, and shared EMHQ structures have been defined and allocated. If not, check that the structure names on your CFNAMES control statement match the structure names that are displayed. If they do not match, change the names in either the coupling facility resource manager (CFRM) policy or the CFNAMES control statement.

In the example, six structures are defined (CF01-CF04, OSAMSESXI, and VSAMSESXI). Three of the structures are allocated (CF01, OSAMSESXI, and VSAMSESXI). The status of the OSAM structure is that it is currently being rebuilt after a structure failure.

DISPLAY XCF,STRUCTURE,STRNAME= command

Use this command to display detailed information about a specific structure. The three structures of importance to IMS are the IRLM, OSAM, VSAM, shared MSGQ, and shared EMHQ structures.

The following example shows an example of output from this command, specifying an OSAM structure named OSAMSESXI.

```
IXC360I  17.30.46  DISPLAY XCF 677
STRNAME: OSAMSESXI
 STATUS: ALLOCATED
  POLICY SIZE    : 2048 K
  PREFERENCE LIST: CF02     CF01
  EXCLUSION LIST IS EMPTY

 ACTIVE STRUCTURE
 ----------------
  ALLOCATION TIME: 02/17/96 17:02:54
  CFNAME         : CF02
  COUPLING FACILITY: ND02...
                    PARTITION: 0    CPCID: 00
  ACTUAL SIZE    : 2048 K
  STORAGE INCREMENT SIZE: 256 K
  VERSION        : A8DAC970 15774B04
  DISPOSITION    : DELETE
  ACCESS TIME    : 0
  MAX CONNECTIONS: 32
  # CONNECTIONS  : 5

 CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
 ---------------- -- -------- -------- -------- ---- ------
 DLI11            04 00040001 MVS1     DLI11    0031 ACTIVE
 DLI12            05 00050001 MVS1     DLI12    0033 ACTIVE
 IMS1             01 00010013 MVS1     DLIOCSA8 0036 ACTIVE
 IMS2             02 00020012 MVS2     DLIOCSB8 0035 ACTIVE
 IMS3             03 00030011 MVS3     DLIOCSC8 0038 ACTIVE
```

There are three parts to the output. The first part shows the status of the structure and information about the active policy. The following information from the display is especially useful:

STRNAME
Is the name of the structure, as specified in the command.

STATUS
Is the current status of the structure. A structure can be in a number of different states, such as ALLOCATED, NOT ALLOCATED, or ALLOCATED REBUILDING.

POLICY SIZE
Is the size of the structure as specified in the policy.

PREFERENCE LIST
Shows the coupling facility on which MVS allocates the structure. z/OS tries to allocate the structure on the first coupling facility listed (CF02 in the example), then the second one (CF01), and so on.

The second part of the output shows more detailed status information for the structure.

ALLOCATION TIME
Is the time the structure was allocated.

CFNAME
Is the name of the coupling facility on which the structure was allocated.

COUPLING FACILITY
Is the name of the coupling facility on which the structure resides.

ACTUAL SIZE
Is the actual size of the structure.

DISPOSITION
Is what the system will do with the structure in a failure situation. The IRLM structure always has a disposition of KEEP. The OSAM and VSAM structures always have a disposition of DELETE.

MAX CONNECTIONS
Is the maximum number of connections that can be made to the OSAM, VSAM, IRLM, shared MSGQ, or shared EMHQ structure. The maximum is 255 connections for any structure.

# CONNECTIONS
Is the current number of IMS systems connected to the structure.

Other information in the second part of the output is not needed for monitoring and problem determination for IMS sysplex data sharing.

The third part of the output shows which z/OS systems are connected to the structure and gives information about each use.

CONNECTION NAME
Is the name of a connection. z/OS will repeat the connection message text in a table to report all connections. If there are no connections, no connection table is displayed.

ID
Is the connection identifier.

VERSION
Is the version number of the z/OS system that is connected.

SYSNAME
Is the name of the z/OS system that is connected.

JOBNAME
Is the name of the job associated with the connection.

ASID
Is the ID of the address space associated with the connection.

STATE
Is the status of the structure, which can be one of the following:

FAILED PERSISTENT
The failed persistent status occurs if the disposition of a structure is KEEP and connection to the structure is lost.

DISCONNECTING
IMS is in the process of disconnecting from the structure.

FAILING
IMS is in the process of abnormally terminating.

ACTIVE
IMS is connected.

ACTIVE &
IMS is connected, but has physically lost connectivity to the structure.

ACTIVE OLD
The structure is being rebuilt. IMS is connected to the old structure.

ACTIVE &OLD
The structure is being rebuilt. IMS is connected to the old structure, but has physically lost connectivity to it.

ACTIVE NEW,OLD
The structure is being rebuilt. IMS is connected to both the old and the new structure.

ACTIVE NEW,&OLD
The structure is being rebuilt. IMS is connected to both the old and the new structure, but has physically lost connectivity to the old structure.

ACTIVE &NEW,OLD
The structure is being rebuilt. IMS is connected to both the old and the new structure, but has physically lost connectivity to the new structure.

ACTIVE &NEW,&OLD
The structure is being rebuilt. IMS is connected to both the old and the new structure, but has physically lost connectivity to both structures.

Using DBRC to control database allocation and access

DBRC allows you to control access to data by IMS systems that participate in data sharing. Using DBRC, you can modify, initiate, and delete the current status indicators in the RECON data set to change the access intent of online IMS systems and the share level of registered databases.

Your data sharing environment depends on the status of the databases and IMS systems indicated in the RECON data set.

You can modify the access intent indicator using a form of the **/START** or **UPDATE** command.

You can modify the share level indicator using a form of one of the DBRC online change commands, **/RMCHANGE**.

Denial of authorization

The following occurs if DBRC (in a data sharing environment) responds to authorization requests but fails to obtain authorization for an application program:

- You receive message DFS047A identifying the database.
- IMS schedules the application program's PSB without database access. IMS abnormally terminates a BMP or MPP with abend U3303 only if it tries to access the database.
- IMS abnormally terminates batch and utility regions with abend U0047.

Changing database access intent

Use one of the following commands to change the access intent that you declare during system definition, where *dbx* is the database name and *xx* is the new access intent:

- **/START DATABASE** *dbx***ACCESS=***xx*
- **UPDATE DB START(ACCESS) SET(ACCTYPE(BRWS,EXCL,READ,UPD))**

For the **/START DATABASE** *dbx***ACCESS=***xx* command, values for *xx* are:

EX
Exclusive use

UP
Update access

RD
Read access

RO
Read-only access

This command is local; it affects only the IMS system on which you enter it. The GLOBAL keyword is not valid with the ACCESS= keyword. If you need to change the access level for a shared database across the sysplex, you must enter this command on each IMS system that shares the database.

The **UPDATE DB START(ACCESS) SET(ACCTYPE(BRWS,EXCL,READ,UPD))** command can be either local or global.

In order to change the access intent for a DEDB, you must stop all PSBs that access any of the areas in the DEDB. You might also have to stop regions that have wait-for-input (WFI) transactions scheduled for the DEDB.

Online DBRC commands

The **/RMCHANGE** command is one of a set of IMS commands that control DBRC utility functions; IMS responds to them by issuing corresponding commands directly to DBRC.

The following table shows these commands and their recommended uses.

Table 24. DBRC commands and functions

Command	Recommended use
/RMCHANGE	Alter a database sharing level Prevent other subsystem authorization Set or remove backout-required status
/RMDELETE	Remove database record and associated status Delete subsystem with unauthorized databases Remove area data set (ADS) record and associated status
/RMGENJCL	Generate online recovery jobs
/RMINIT	Register a new database Start control of an area data set
/RMLIST	Obtain recovery and authorization information for a database or ADS
/RMNOTIFY	Reset a system status

If you want to change the share level of a registered database (named ORDERDB) from intraprocessor block-level data sharing (share-level 2) to interprocessor block-level data sharing (share-level 3) to allow data sharing with another IMS system in another processor, enter:

```
/RMCHANGE DBRC='DB DBD(ORDERDB) SHARELVL(3)'
```

DBRC replies with its corresponding command input (**CHANGE.DB DBD(ORDERDB) SHARELVL(3)**) and a series of DBRC messages.

Do not enter the **/RMCHANGE**, **/RMDELETE**, or any command that alters the status of database records in the RECON data set while a job accesses the database. Stop the database using a global **/DBRECOVERY DB** or **UPDATE DB STOP(ACCESS)** command before modifying any RECON record for that database. Otherwise, IMS rejects the command with an error.

Related reading: For full descriptions of these commands and their functions, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Improving database availability

Improving database availability in a data-sharing environment involves controlling online change, reorganizing databases, and making database image copies.

Controlling online change

If you modify a shared database online in a sharing environment, you must coordinate the sequence of **/MODIFY** commands across all sharing IMS systems.

1. Stop the databases you want to modify in all IMS online systems using that database.
2. Perform the online change on all sharing IMS systems
3. Restart the shared databases only when the **/MODIFY** command sequences are complete in all affected IMS systems.

If any of the data-sharing IMS systems also physically share any of the ACBLIB, FORMAT, or MODBLKS libraries, you must ensure that these IMS systems use the same designated active library, for example, IMS.ACBLIBA.

Each system keeps track of its active libraries in the IMS.MODSTAT data set. Performing an online change in only one IMS system could result in the individual IMS systems' using different control blocks, which can cause unpredictable errors in a data-sharing environment.

Also, if any library is shared, ensure that the Online Change Copy utility (DFSUOCU0) updates only inactive libraries.

Reorganizing databases

While you reorganize a database (using the database reorganization utilities), you must protect it from changes by IMS systems that are authorized to use it, unless you are using HALDB Online Reorganization (OLR).

You have three ways to protect the database:

- Issue the **/DBRECOVERY DB GLOBAL** or **UPDATE DB STOP(ACCESS)** command to close the database on all IMS systems. After the database reorganization is complete, tell the MTO of each IMS system that the database is available.
- Issue the DBRC **CHANGE.DB NOAUTH** command to specify that the database is not authorized to participate in data sharing. After the reorganization is complete, issue the **CHANGE.DB AUTH** command to resume data sharing.
- Devise an MTO procedure to coordinate application activities in sharing IMS systems. This procedure would direct the MTOs to stop applications that might be active and using the database. After the reorganization is complete, the MTOs can restart applications in the appropriate online IMS systems.

The reorganization utilities communicate with DBRC. In response, DBRC does not allow further processing for the database, even after the reorganization, until you make an image copy of the database or recovery group.

The reorganization utilities might not execute if they fail to obtain authorization from DBRC for the database. DBRC denies authorization if it has authorized another IMS system to update the database (which is why you need to issue the **/DBRECOVERY DB**, **UPDATE DB STOP(ACCESS)**, or **CHANGE.DB** commands first). If you receive message DFS044I saying that DBRC is required for the execution of the utility, and the utility abends, wait for the IMS system to terminate or, if it is an online IMS, issue a **/DBRECOVERY DB** or **UPDATE DB STOP(ACCESS)** command.

Related reading: For a sample operational scenario dealing with database recovery in a database-level sharing environment, see "Reorganizing a database" on page 249.

Making an online database image copy

When you schedule the Online Database Image Copy utility (DFSUICP0) as a BMP, DBRC manages the requirement for restrictive access. You must, however, ensure that no IMS online systems, online or batch, other than the IMS online system running the Image Copy BMP, is authorized to update the database.

For a sample operational scenario supporting image copy, see "Making an image copy of a shared database" on page 250.

Maintaining normal operations

Data sharing requires additions to your normal operating procedures. The following topics present sample operational scenarios that show how to perform routine adjustments and maintenance in a database-level data-sharing environment.

Recommendation: Use an External Time Reference (ETR) device and set it to use Universal Coordinated Time. You can set the local time for each data-sharing IMS system separately as necessary. See *z/OS MVS Initialization and Tuning Reference* for information on using an ETR.

Scheduling a batch updater

Given the following initial and target configurations, use this scenario to schedule a batch updater.

- Initial configuration
  - Database x (*dbx*) is registered at share level 1.
  - *dbx* is authorized for update access to an online IMS system (IMSA).
- Target configuration
  - *dbx* is authorized for update access to a batch IMS system (IMSB).
  - *dbx* is authorized for read-only access to IMSA.

Follow these steps to schedule a batch updater:

1. Prevent other IMS systems from obtaining access to *dbx* with the following command: `CHANGE.DB DBD (`*dbx*`) NOAUTH`
2. Suspend IMSA's activity on *dbx*, close *dbx*, and release IMSA's authorization with one of the following commands:
   - **/DBRECOVERY DB** *dbx***NOFEOV**
   - **UPDATE DB STOP(ACCESS) OPTION(FEOV)**

   You can use the **/DBRECOVERY** or **UPDATE** command for a DEDB area by specifying AREA area instead of DB.
3. Allow read-only processing from IMSA with one of the following commands:
   - **/START DATABASE** *dbx* **ACCESS=RO**
   - **UPDATE DB NAME** *dbx* **START(ACCESS) SET(ACCTYPE(READ))**
4. Allow IMSB to initialize with the following command: **CHANGE.DB DBD (***dbx***) AUTH**
5. Start IMSB with update access to *dbx*.

Reinstating an online updater

Given the following initial and target configurations, use this scenario to restore the initial configuration.

- Initial configuration
  - *dbx* is authorized for read-only access to an online IMS system (IMSA).
  - A batch IMS system has been terminated.
- Target configuration

- *dbx* is registered at share level 1.
- *dbx* is authorized for update access to IMSA.

Follow these steps to reinstate an online updater:

1. Prevent other IMS systems from obtaining access to *dbx* with the following command: **CHANGE.DB DBD (*dbx*) NOAUTH**

   When you force IMSA to release its authorization on *dbx*, another IMS system could obtain authorization to process it. The DBRC **CHANGE.DB** command with the NOAUTH keyword indicates (in the database record in the RECON data set) that the database is unavailable for processing.
2. Terminate IMSB.
3. Suspend IMSA's activity on *dbx*, close *dbx*, and release IMSA's authorization with one of the following commands:
   - **/DBRECOVERY DB *dbx* NOFEOV**
   - **UPDATE DB STOP(ACCESS) OPTION(FEOV)**
4. Allow update processing for IMSA with one of the following commands:
   - **/START DB dbx ACCESS=UP**
   - **UPDATE DB START(ACCESS)**
5. Allow access or update in IMSA with the following command: **CHANGE.DB DB *dbx* AUTH**

The initial configuration of the scenario is described in "Scheduling a batch updater" on page 247.

Transferring update capability

Given the following initial and target configurations, use this scenario to transfer update capability from one sharing IMS system to another.

- Initial configuration
  - *dbx* is authorized for update access to an online IMS system (IMSA).
  - *dbx* is authorized for read-only access to another online IMS system (IMSB).
- Target configuration
  - *dbx* is authorized for read-only access to IMSA.
  - *dbx* is authorized for update access to IMSB.

Follow these steps to transfer update compatibility:

1. Prevent other IMS systems from obtaining access to *dbx* with the following command: **CHANGE.DB DBD (*dbx*) NOAUTH**
2. Suspend activity and close *dbx* in both IMS systems with one of the following commands:
   - **/DBRECOVERY DB *dbx* NOFEOV**
   - **UPDATE DB STOP(ACCESS)**
3. Allow processing in both IMS systems. Issue one of the following commands on IMSA:
   - **/START DB *dbx* ACCESS=RO**
   - **UPDATE DB START(ACCESS)**
4. Then issue one of the following commands on IMSB:
   - **/START DB dbx ACCESS=UP**
   - **UPDATE DB START(ACCESS)**
5. Allow access or update in both IMS systems with the following command: **CHANGE.DB DBD(*dbx*) AUTH**

   Although IMSA and IMSB now have compatible database authorizations, the database is not protected from other IMS systems until IMSA and IMSB schedule an application program with a PSB that is sensitive to the database. In order to lock out any other updaters, therefore, IMSB must be the first IMS system to obtain the authorization to process the database with update access.

Reorganizing a database

Given the following initial and target configurations, use this scenario to reorganize a database.

- Initial configuration
  - *dbx* is registered for database-level sharing.
  - *dbx* is authorized for update access to an online IMS system (IMSA).
- Target configuration
  - *dbx* is successfully reorganized.

Follow these steps to reorganize a database:

1. Prevent other IMS systems from obtaining access with the following command: **CHANGE.DB DBD(***dbx***) NOAUTH**
2. Start reorganization jobs.

   Run a set of reorganization jobs, such as Scan, Unload, Reload, or Prefix Update, in a protected environment.

   You must run these reorganization utilities without interference from other IMS systems. The DBRC **CHANGE.DB NOAUTH** command prohibits authorizations to the database for online and batch IMS systems, except for all reorganization utilities.
3. Authorize the Online Database Image Copy utility (DFSUICP0) initialization with the following command: **CHANGE.DB DBD(***dbx***) AUTH**

   This command allows other batch or online IMS systems to obtain access to *dbx*.

Related reading: For a discussion of ways to reorganize a database, see "Reorganizing databases" on page 246.

Making an image copy with exclusive control

Given the following initial and target configurations, use this scenario to make an image copy of a database data set with exclusive control.

- Initial configuration
  - *dbx* is registered for database-level sharing.
  - *dbx* is authorized for update access to an online IMS system (IMSA).
  - *dbx* is authorized for read-only access to another online IMS system (IMSB).
- Target configuration
  - An image copy of database *dbx* (data set *y*).

Follow these steps to make an image copy with exclusive control:

1. Prevent other IMS systems from obtaining access with the following command: **CHANGE.DBDS DBD(***dbx***) DDN(***y***) ICON**

   Use the DBRC **CHANGE.DBDS** command with the ICON keyword to indicate (in the RECON data set) that an image copy is needed. This indication prevents any IMS system except the Image Copy utility from obtaining access to the database.
2. Suspend IMS system activity on *dbx* and close *dbx* in all sharing IMS systems with one of the following commands:
   - **/DBRECOVERY DB** *dbx* **NOFEOV**
   - **UPDATE DB STOP(ACCESS) OPTION(FEOV)**
3. Run the Online Database Image Copy utility (DFSUICP0).

When this utility completes successfully, it removes the indication in the RECON data set that an image copy is needed, so that DBRC can authorize IMS systems to access the database.

4. Resume IMS system activity on *dbx* with one of the following commands:
   - **/START DB** *dbx*
   - **UPDATE DB START(ACCESS)**

Making an image copy of a shared database

Because the Online Database Image Copy utility (DFSUICP0) always requests read access, it can share the database only with IMS systems that request a read-only access. If the other IMS systems need to update the database while you take an image copy, use the Concurrent Image Copy option.

Given the following initial and target configurations, use this scenario to make an image copy in a database-level sharing environment:

- Initial configuration
  - *dbx* is registered for database-level sharing.
  - *dbx* is authorized for update access to an online IMS system (IMSA).
  - *dbx* is authorized for read-only access to a batch IMS system (IMSB).
- Target configuration
  - The Online Database Image Copy utility (DFSUICP0) on IMSA is authorized for read access.
  - *dbx* is authorized for read access to IMSA.
  - *dbx* is authorized for read-only access to IMSB.

Follow these steps to make an image copy:

1. Prevent all IMS systems from obtaining update access with the following command: **CHANGE.DB DBD(***dbx***) READON**

   This command indicates (in the RECON data set) that the database cannot be accessed by IMS systems with update intent.
2. Suspend IMSA update activity on *dbx* and lower authorization to read access with one of the following commands:
   - **/DBDUMP DB** *dbx* **NOFEOV**
   - **UPDATE DB STOP(UPDATES) OPTION(FEOV)**
3. Run the Online Database Image Copy utility (DFSUICP0).
4. Resume update activity on IMSA with one of the following commands:

   -
     ```
     CHANGE.DB DBD(dbx) READOFF
     /START DATABASE dbx ACCESS=UP
     ```

   -
     ```
     CHANGE.DB DBD(dbx) READOFF
     UPDATE DB NAME(dbx) START(ACCESS) SET(ACCTYPE(UPD))
     ```

   The **CHANGE.DB** command with the READOFF keyword indicates (in the RECON data set) that the database can be accessed by IMS systems with update intent.

Lowering the share level of a database

Given the following initial and target configurations, use this scenario to lower the share level of a database.

- Initial configuration
  - *dbx* is authorized for update access to an online IMS system (IMSA).
  - *dbx* is authorized for read-only access to another online IMS system (IMSB).

- Target configuration
  - *dbx* is authorized for exclusive control by IMSA.

Follow these steps to lower the share level:

1. Prevent all IMS systems from obtaining access with the following command: **CHANGE.DB DBD(*dbx*) NOAUTH**
2. Suspend activity on *dbx* and close *dbx* in both IMS systems with one of the following commands:
   - **/DBRECOVERY DB dbx NOFEOV**
   - **UPDATE DB STOP(ACCESS) OPTION(FEOV)**
3. Allow processing in IMSA with the following command:

   ```
   CHANGE.DB DBD(dbx) AUTH
   /START DATABASE dbx ACCESS=EX
   /START PGM
   /START TRAN
   ```

   The database is not protected from other IMS systems until IMSA schedules an application program with a PSB that is sensitive to the database. In order to lock out all other IMS systems, therefore, IMSA must be the first IMS system to obtain the authorization to process the database with exclusive access.

   IMSA accepts the change of access intent only if the target database is not authorized.

You can also achieve the same results by modifying the sharing level of the database in the RECON data set instead of changing the access intent of IMSA. Omit the **/START DATABASE** command with the ACCESS= keyword and perform the following step before step 3:

- Request DBRC to enforce exclusive control:

  ```
  CHANGE.DB DBD(dbx) SHARELVL(0)
  ```

  In this case, DBRC grants database authorization at the requested level, and IMSA owns the database exclusively.

Recovering from failures

The following topics provide information to help you plan for an recover from failures in a data-sharing environment.

Managing system logs

Recovering a shared database requires log data from each sharing IMS system. A DBRC **GENJCL.RECOV** command or a Database Recovery utility execution using DBRC will fail if logs need to be merged for the database.

You can use the merge function of the IMS Database Change Accumulation utility (DFSUCUM0) to merge the log data from all sharing IMS systems.

You can run the Database Change Accumulation utility (DFSUCUM0) as often as your workload requires without first establishing a complete set of log volumes. Doing so allows you to accumulate change records on a timely basis, keeping the log input to a reasonable number of volumes, without taking databases offline or bringing down IMS. Doing so can, however, increase the size of the change accumulation data set.

IMS stores records from these partial subsets of log volumes on the change accumulation data set as "spill records". These records retain their individual update characteristics.

Before running the Database Recovery utility (DFSURDB0), you must run the Database Change Accumulation utility, specifying a complete set of log volumes for the DBDS that you need to recover. If you run the Database Change Accumulation utility with an incomplete set of logs, specify the change accumulation output data set as input to your next Database Change Accumulation run. If a change accumulation data set represents an incomplete set of logs for the DBDS that you need to recover, the data set is not valid for recovery of that DBDS.

The DBRC **LIST.CAGRP** command displays whether the set of log volumes for each DBDS of the change accumulation group is complete or incomplete.

You can run DFSUCUM0 with a valid set of log volumes at any time in order to minimize the size of the change accumulation data set. To establish a valid set of logs in a data-sharing complex, use the following procedure:

1. Suspending operation of all batch IMS systems
2. Using the **/DBRECOVERY DB** or **UPDATE DB STOP(ACCESS)** command to terminate database access for all but one online IMS system
3. Forcing a log switch on the remaining online IMS system

Planning recovery procedures

Recovery in a data-sharing environment is similar to standard IMS recovery.

Both standard IMS recovery and recovery in a data-sharing environment involves these primary tasks:

- Setting up the mechanisms of recovery (logging, taking checkpoints, keeping records)
- Setting up the operational procedures to be followed in the event of situations that require recovery

The logging and checkpoint mechanisms of online IMS systems in a nonsharing environment are also active in a data-sharing environment. These include:

- System log data sets and the use of the WADS and restart data sets
- Application program, system, and message-queue checkpoints
- Making image copies of databases

The primary difference between nonsharing and data-sharing environments is in their degree of reliance on DBRC. DBRC helps control the data-sharing environment; it does not merely keep records.

As in a single IMS system, a failure in a data-sharing environment might require offline (utility-type) recovery as well as restart. After a failure in a data-sharing environment, however, you must consider and perform recovery actions not only in the failing system, but also in other sharing systems. In addition, each IRLM, DBRC, and coupling facility structure in the environment introduces another potential point of failure and another component that you might need to include in a recovery and restart procedure.

Recovery for the message queue data sets, (in a nonshared-queues environment) log data sets, and system data sets are no different in a data-sharing environment:

- To recover the log, use the Log Recovery utility (DFSULTR0).
- To recover the message queue data set, request at IMS restart that IMS rebuild the queues. You need to supply the system log data sets since the last system checkpoint that recorded the queue contents (SNAPQ, DUMPQ, or PURGE).
- To recover a system data set other than the message queue data set, update the latest image copy. You must recover the RECON data set for data sharing to continue.

Recovery facilities

Dynamic backout, batch backout, and forward recovery facilities are different in data-sharing and nonsharing environments.

Dynamic backout

As in a nonsharing environment, an online IMS system in a data-sharing environment dynamically backs out an application program's uncommitted database changes and discards its uncommitted output messages if the application program fails or requests backout with a rollback call. In a data-sharing environment, however, IRLM locks and RECON status indicators ensure integrity by protecting uncommitted changes from sharing IMS systems.

After an application program failure, operation of the system and other application programs continues uninterrupted.

If the application program is a batch updater in a block-level data-sharing environment, IMS dynamically backs out changes in any of the following situations:

- When the application program issues a rollback call (ROLL, ROLB, or ROLS).
- When the batch region uses system logging allocated to DASD, you specify BKO=Y as an execution parameter, and the application program issues a ROLB or ROLS call or a DL/I-detected error occurs.
- When some types of resource shortages and deadlocks involving two or more batch update application programs occur, IMS performs dynamic backout for one or more of the batch application programs; all continue executing, provided that the involved regions use DASD logging and you specify BKO=Y.

In these situations, you are not involved in recovery actions.

For either online or batch IMS systems, if the database is shared at the database level, other application programs might read uncommitted data before the database is dynamically backed out. Backout affects only the updating application program's data in the database—not the output of any other application program that used this uncommitted data. This is a consequence of the nature of PROCOPT=GO processing, rather than of the nature of database backout in a data-sharing environment.

If a dynamic backout for a database fails, you must back out the database using batch backout. Then, you must restart that database using a **`/START DB`** or **`UPDATE DB START(ACCESS)`** command.

Batch backout

If the failing application program is a batch updater involved in block-level data sharing, IMS does not always automatically back out its uncommitted database changes.

Just as in a nonsharing environment, you must recover the database if any of the following is true:

- The batch region does not use DASD logging
- You do not specify BKO=Y as an execution parameter
- The failure is not a DL/I-detected error

In a block-level-data-sharing environment, however, you must back out database changes of all batch update application programs using the Batch Backout utility (DFSBBO00).

During block-level sharing, if an application program deadlock occurs involving only batch update jobs, IMS abnormally terminates one of these jobs with abend U0777. If IMS can dynamically backout changes, other application programs continue processing. Otherwise, IMS abnormally terminates the other application programs with abend U3303.

You must recover all of the batch jobs that did not have dynamic backout by executing the Batch Backout utility for each of them. Because the locks held by these batch jobs could affect access of other batch jobs or online IMS systems, you should begin batch backouts promptly.

Restriction: For recovery of batch IMS systems that participate in block-level data sharing:

- The Batch Backout utility requires an IRLM for its execution.
- The Batch Backout utility backs out to the last successful checkpoint.

Forward recovery

Recovery of a database in a data-sharing environment is similar to recovery in a nonsharing environment. In both environments, use the Database Recovery utility (DFSURDB0) and give it the most recent image copy of the database, along with all pertinent log data sets used since you made that image copy.

Block-level data sharing, however, can require one additional step. Because more than one IMS system might update the database, you need the log from each IMS system to reconstruct the database.

Furthermore, because these updates are most likely to be concurrent, you cannot submit those logs sequentially to the Database Recovery utility. Instead, you must first merge them by running the Database Change Accumulation utility (DFSUCUM0). Be sure to process all pertinent logs.

Even if you run change accumulation regularly, you must run the utility again before recovering the database so you merge the most recent logs with earlier logs.

However, you do not need to establish a valid set of logs for the Database Change Accumulation utility (DFSUCUM0). You can run DFSUCUM0 at any time without suspending operation of batch IMS systems, terminating database access for all but one online IMS system, or forcing a log switch on the remaining online IMS system. See "Managing system logs" on page 251.

Recovery without DBRC

If you perform any recovery-related actions when DBRC is not running, such as making database image copies, problems could arise because DBRC would be unaware of changes in status. You must, therefore, specifically inform DBRC of such changes.

DBRC offers several commands for this purpose; see "Online DBRC commands" on page 245.

Because restart might be required, you should, for example, use the Utility Control Facility (UCF) without DBRC being active. But, if you use UCF for any recovery-related work, you must specifically notify DBRC of status changes afterward.

Restart after IMS failure

Restart an IMS system in a data-sharing environment in the same way as in a nonsharing environment.

- Restart IMS normally, using the **/NRESTART** command, if you were able to shut the IMS system down normally.
- Emergency restart IMS, using the **/ERESTART** command, if the IMS system failed.

In a data-sharing environment, however, consider that if the associated IRLM also stops or fails, you must restart that IRLM before you start IMS in a block-level data-sharing environment. See "Starting the IRLM" on page 237.

Restart after DBRC failure

Because DBRC runs under the control of IMS, if IMS abnormally terminates, IMS tries to release DBRC buffers, and to close the system log.

After you correct the DBRC problem, restart IMS using the **/ERESTART** command.

Recovery involving IRLM

Failure in one part of a data-sharing configuration does not usually cause the other parts to fail.

For example:

- If one z/OS system fails because of a hardware problem or an abnormal termination of the z/OS operating system, the other systems—with their IRLMs and other IMS systems—continue to run. In this case, the surviving IRLMs are notified of the failure by z/OS and then provide the necessary retained lock protection for locks owned by the failing member.
- If an IRLM fails, its associated IMS online systems continue to run while all active application programs receive U3303 abends, and IMS suspends transaction scheduling, allowing backouts to complete. Batch IMS systems terminate to protect database integrity. The surviving IRLMs are notified of the failure by z/OS and then provide the necessary retained lock protection for locks owned by the IMS on the failing IRLM.

  If an IRLM fails in an IMS DBCTL environment, in-doubt threads receive a U3303 abend on their next DL/I calls. The in-doubt threads remain active until phase 2 of sync-point processing. After processing phase 2 commit or abort, the thread is terminated. The IMS DBCTL system tries to reconnect to IRLM after all in-doubt threads are terminated.

- If an IMS system fails, its associated IRLM and other IMS systems continue to run, with locks owned by the failing IMS protected by retained locks created by its IRLM.
- If DBRC fails, its IMS system abnormally terminates, but the associated IRLM and the other IMS systems and their IRLMs continue to run.

Transaction recovery in online IMS systems

IMS automatically backs out transactions if the IMS online system determines that transaction processing cannot proceed because of problems directly or indirectly to do with data sharing, or because the resource is unavailable, and if you specified SERIAL=NO in the TRANSACT system definition macro (or accepted it as the default).

If you specify SERIAL=YES in the TRANSACT system definition macro, IMS puts the transaction back on the front of the transaction queue to be processed first-in-first-out, and stops the transaction (USTOP state).

If the transactions do not use the DL/I INIT STATUS GROUPA call, IMS places transactions on either the suspend queue or the transaction queue when:

- A transaction requests a lock held by an IMS system identified to a failed IRLM.
- A transaction requests a lock held by a failed IMS system.
- The transaction attempts to access a database at a level higher than that for which the IMS system is authorized.
- A transaction requires a database that is stopped or locked.
- The MTO enters a **`/DBRECOVERY DB`** or **`UPDATE DB STOP(ACCESS)`** command.

IMS sends message DFS3324I, saying that the transaction is suspended, for each individual transaction that is suspended.

IMS saves the suspend queue across any restart, except a cold start, and all transactions on the suspend queue are eligible for reprocessing. When IMS reprocess the transactions, IMS releases them to the message queues and schedules them in the normal way.

Several conditions prompt IMS to attempt to reprocess these transactions:

- A restart of the online IMS system, including after an XRF takeover
- A communication from another IMS system performing restart or from a batch IMS system that has completed backout
- A reconnection of an IMS online system to an IRLM
- The MTO command, **`/DEQUEUE SUSPEND`**
- The MTO commands, **`/START DB`** or **`UPDATE DB START(ACCESS)`** to start the database
- A manual or automatic rebuild of the CF structures with **`SETXCF START,REBUILD,STRNAME=`** command.

IMS issues message DFS3336I when it attempts to reprocess suspended transactions.

Restart with a single failing IRLM

If the only IRLM in a data-sharing configuration fails: the associated IMS online systems suspend all transaction scheduling, IMS performs backout for affected transactions, IMS places the transactions on a suspend queue or transaction queue, depending on whether you specified SERIAL=NO or SERIAL=YES in the TRANSACT macro and IMS stops shared databases.

IMS issues DFS2012I messages for each database that has been stopped automatically because of the failure. IMS abnormally terminates batch IMS systems associated with the IRLM. Batch database backout is required for only those batch executions that perform database updates and do not use dynamic backout.

After you restart the failed IRLM, enter a z/OS **MODIFY** command, specifying the RECONNECT keyword, for each IMS online system. This command reestablishes the data-sharing environment, and initiates reprocessing for the suspend queue.

Related reading: For more information about transaction recovery, see "Transaction recovery in online IMS systems" on page 255)

Restart in a sysplex data-sharing environment

In a sysplex data-sharing environment, an IRLM is informed of z/OS failure, IMS failure, failure of another IRLM, or loss of connection to physical locking resources.

The IRLM affected by the failure disconnects from the data-sharing group. The remaining IRLMs in the data-sharing group are referred to as surviving IRLMs.

When an IRLM is forced to leave a data-sharing group for any reason, that IRLM issues the DXR136I message, which says that an IRLM has disconnected from the data-sharing group.

The surviving IRLMs issue the DXR137I message, which says the group status has changed, and that an IRLM has been disconnected from the data-sharing group.

No operator action is required for either message. As in a nonsysplex data-sharing environment, IRLM protects resources that are locked from being accessed by other IMS systems.

Normal restart procedures can be used to restart the IRLM and its associated IMS.

Data sharing in a sysplex environment

For the most part, nonsysplex data sharing concepts apply equally to sysplex data sharing. There are exceptions, which are discussed in these topics.

These topics assume you are familiar with z/OS sysplex concepts and terminology.

Sysplex data-sharing concepts and terminology

When multiple IMS systems share data across more than two z/OS images, it is called *sysplex data sharing*.

Each z/OS image involved in sysplex data sharing must have at least one IRLM and a corresponding Fast DB Recovery (FDBR) region. Each IRLM is connected to a coupling facility. The coupling facility, as explained in "Coupling facility" on page 258, is used to maintain data integrity across IMS systems that share data.

Buffer invalidation

When a database block or CI is updated by one IMS, IMS must make sure all copies of this block or CI in other IMS buffer pools are marked invalid.

To do this, the updating IMS issues an invalidate call. This call notifies the coupling facility that the content of a buffer has been updated and that all other copies of the buffer must be invalidated. The coupling facility then invalidates the buffer for each IMS that registered an interest in the buffer. This process is called *buffer invalidation*. Whenever IMS tries to access a buffer, it checks to be sure that the buffer is still valid. If the buffer is invalid, IMS rereads the data from DASD. Data integrity is thereby maintained.

Buffer invalidation works in all IMSplex database environments: DB/DC, DBCTL, and DB batch. In the sysplex environment, IMS supports buffer pools for VSAM, VSAM hiperspace, OSAM, and OSAM sequential buffering buffers.

Data-sharing groups

As with non-sysplex data sharing, the concept of a data-sharing group applies. The following figure shows a sample data-sharing group.

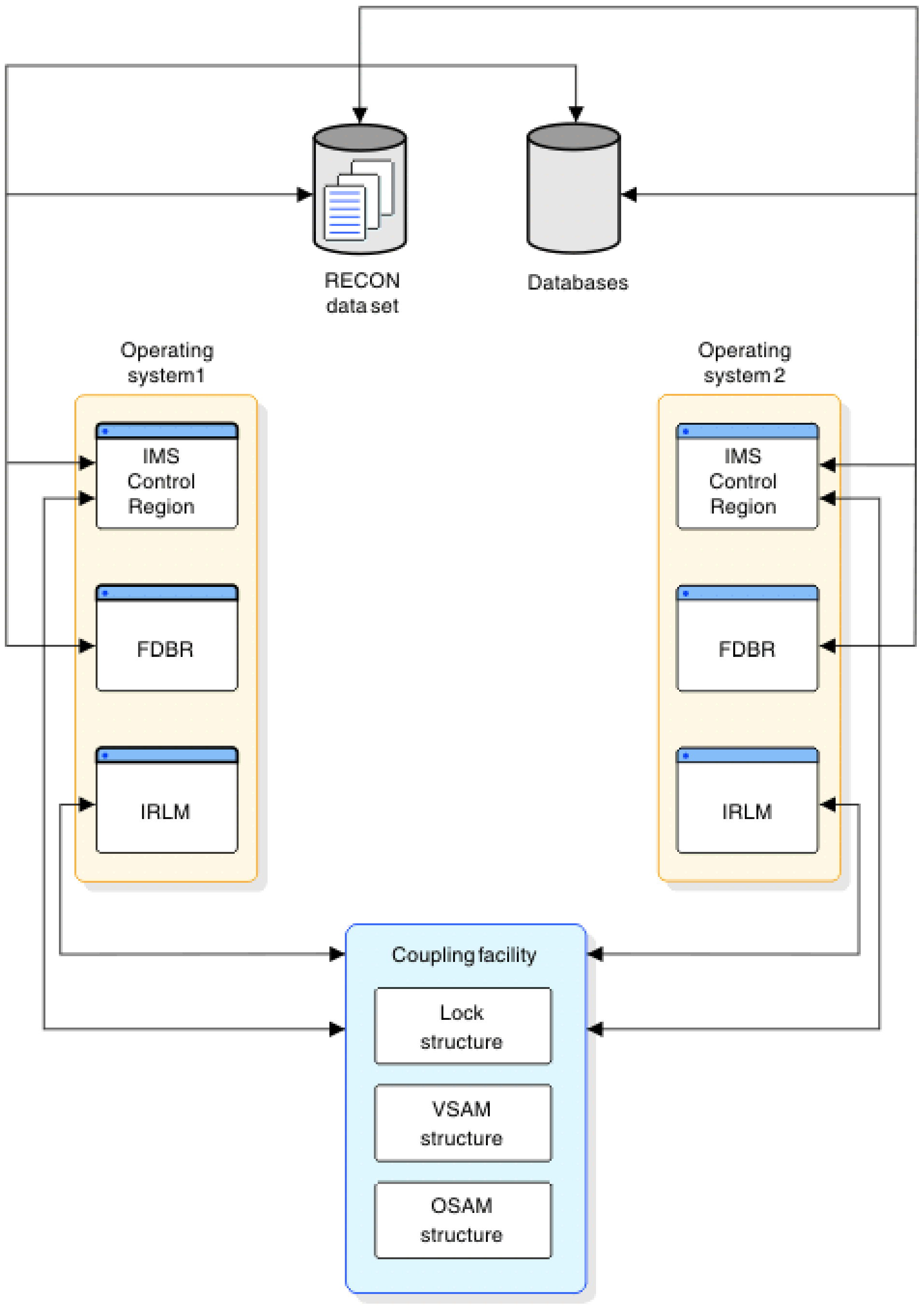


Figure 36. Sample data-sharing group

The data-sharing group has two operating systems. Each operating system contains an IMS control region, an FDBR, and an IRLM. The IMS systems in each operating system share the following resources:

- Databases
- A RECON dual copy data set
- One or more coupling facilities
- A single IRLM lock table structure (lock structure) in the coupling facility

- OSAM and VSAM buffer invalidate structures in the coupling facility (hereafter called an OSAM or VSAM structure)

Fast database recovery regions are not shared.

Communication in the data-sharing group is through the IRLMs connected to the coupling facility. Without a coupling facility, no valid sysplex data-sharing environment exists; IRLM does not grant global locks, even for non-sysplex data sharing. IRLM requires a coupling facility for sysplex data sharing. If a coupling facility is not available, IRLM allows only non-sysplex data sharing.

IMS connects to the structures in the coupling facility. Sysplex data sharing uses the OSAM and VSAM structures for buffer invalidation. This differs from non-sysplex data sharing, which does buffer invalidation using broadcasts (notifies) to each IMS. Sysplex data sharing uses the lock structure in the coupling facility to establish and control the data-sharing environment.

A data-sharing group is defined using the CFNAMES control statement in the DFSVSMxx member of the IMS PROCLIB data set. The lock, OSAM, and VSAM structures are named on the CFNAMES control statement.

Coupling facility

Figure 36 on page 257 shows the coupling facility and the three structures in it that are used for sysplex data sharing.

Although the figure shows a single coupling facility, more than one are possible. Additional coupling facilities can be defined for backup. Or, to increase throughput, structures can be split across coupling facilities; the lock structure, for example, can be put on one coupling facility and OSAM and VSAM structures on another.

The OSAM and VSAM structures are used for buffer invalidation. For each block of shared data read by any IMS system connected to the coupling facility, an entry is made in the OSAM or VSAM structure. Each entry consists of a field for the buffer ID (known to z/OS as the resource name) and 32 slots. The slots are for IMS systems to register their interest in an entry's buffer. This makes it possible for as many as 32 IMS systems to share data. Note that the sysplex data-sharing limit of 32 IMS systems is the number of IMS systems that can connect to a structure; it is not the number of IMS systems that are running. It is also not the number of total connections that can be made to the coupling facility which is 255.

The lock structure is used to establish the data-sharing environment. For a data-sharing group, the first IMS system to connect to a lock structure determines the data-sharing environment for any IMS system that subsequently connects to the same lock structure. When identifying to the IRLM, IMS passes the names of the coupling facility structures specified on the CFNAMES control statement, plus the DBRC RECON initialization time stamp (RIT) from the RECON header. The identify operation fails for any IMS system not specifying the identical structure names and RIT as the first IMS.

If a structure fails (or you initiate a manual rebuilding of the structure), IMS tries to rebuild it. If IMS is unsuccessful or if the connection to a structure is lost, IMS quiesces data-sharing transactions and stops data sharing. After the rebuilding is complete, IMS tries to reconnect to the structure. If IMS reconnects successfully, IMS continues processing data-sharing transactions. If IMS fails to reconnect, data sharing remains stopped and IMS waits to reconnect until it is again notified that coupling facility resources are available.

Note: If a batch job is connected to a structure when IMS attempts to rebuild the structure, IMS terminates the batch job and issues an abend U3303.

Related reference
z/OS: Defining a coupling facility

XRF and sysplex data sharing

XRF can be used in a sysplex data-sharing environment only if the coupling facility is available and connected when the alternate system is started.

Restriction: You cannot use Fast DB Recovery for XRF subsystems.

Sample sysplex data-sharing configurations

The following figures shows several possible configurations for sysplex data sharing.

The DBRC RECON data set and the IMS database are not shown in these figures but, as in Figure 36 on page 257, RECON and the database are shared by the IMS systems in the sysplex data-sharing group.

Three structures on one coupling facility

The following configuration shows multiple z/OS images running in the sysplex data-sharing environment using IRLM. IMS is running on each z/OS image. One coupling facility is being used. The lock, OSAM, and VSAM structures are on the same coupling facility.

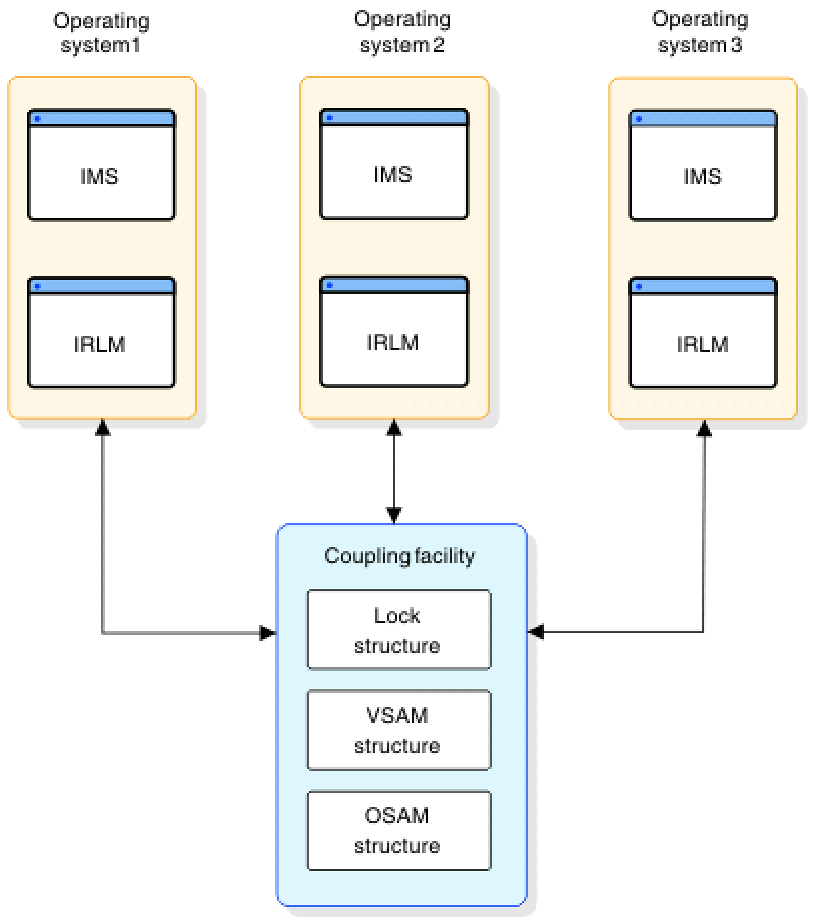


Figure 37. Sysplex data-sharing configuration with three structures on one coupling facility

Three structures on two coupling facilities

The following configuration illustrates multiple z/OS images running in the IMSplex using IRLM. IMS is running on each z/OS image. Two coupling facilities are being used. The lock structure is on coupling facility 1. The OSAM and VSAM structures are on coupling facility 2.

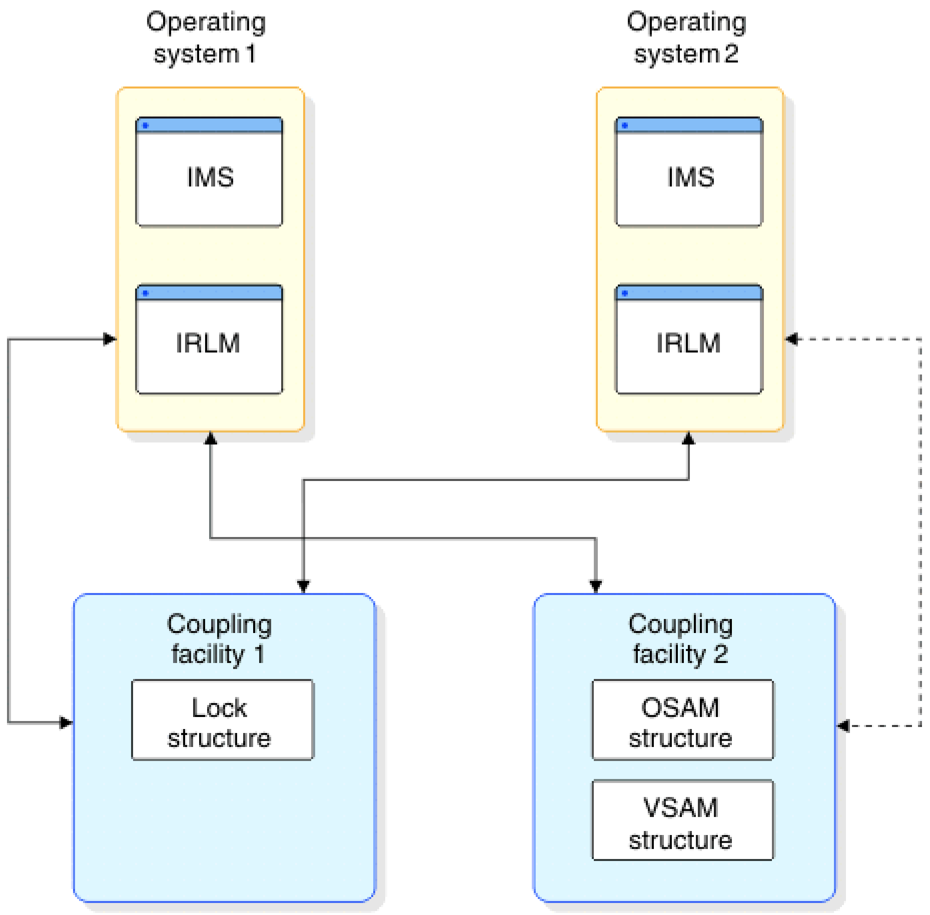


Figure 38. Sysplex data-sharing configuration with three structures on two coupling facilities

Three structures on one coupling facility with backup

The following configuration shows multiple z/OS images running in the IMSplex using IRLM. IMS is running on each z/OS image. Two coupling facilities are being used. The lock, OSAM, and VSAM structures are on coupling facility 1. Coupling facility 2 is used as a backup so that if coupling facility 1 fails, structures can be rebuilt on coupling facility 2.

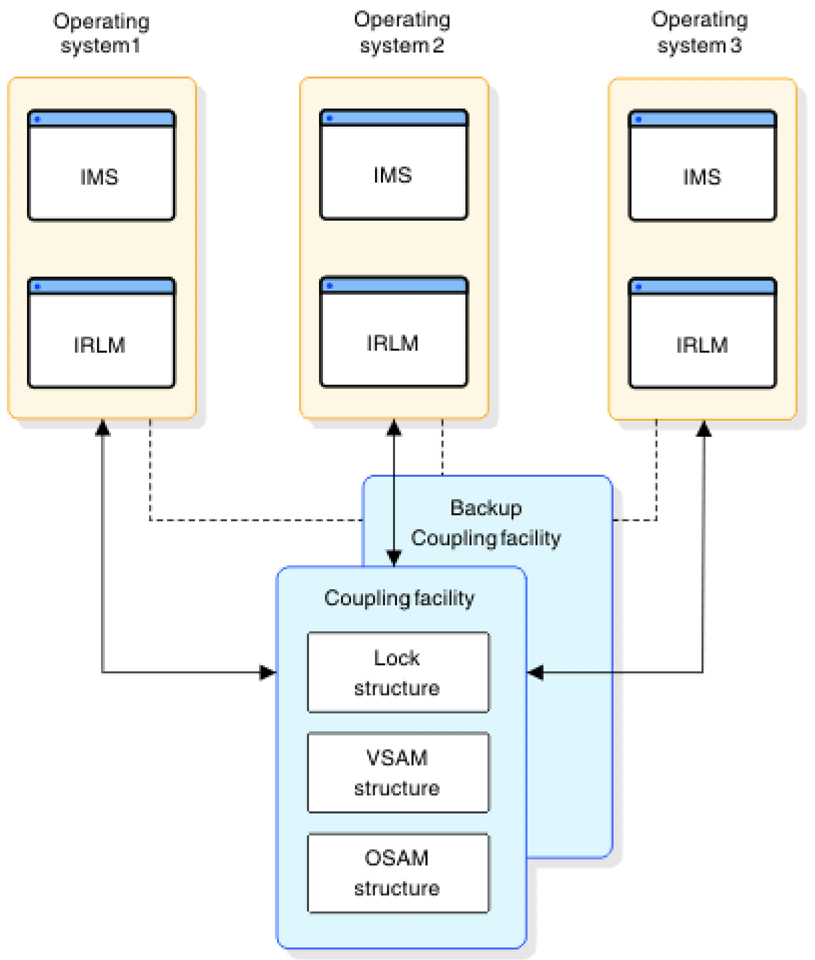


Figure 39. Sysplex data-sharing configuration with three structures on one coupling facility with backup

Three structures on one coupling facility with backup and XRF

The following configuration shows six z/OS images running in the IMSplex using IRLM. Three active (XRF) IMS systems are running, and each active IMS has an alternate system. The lock, OSAM, and VSAM structures are on the coupling facility.

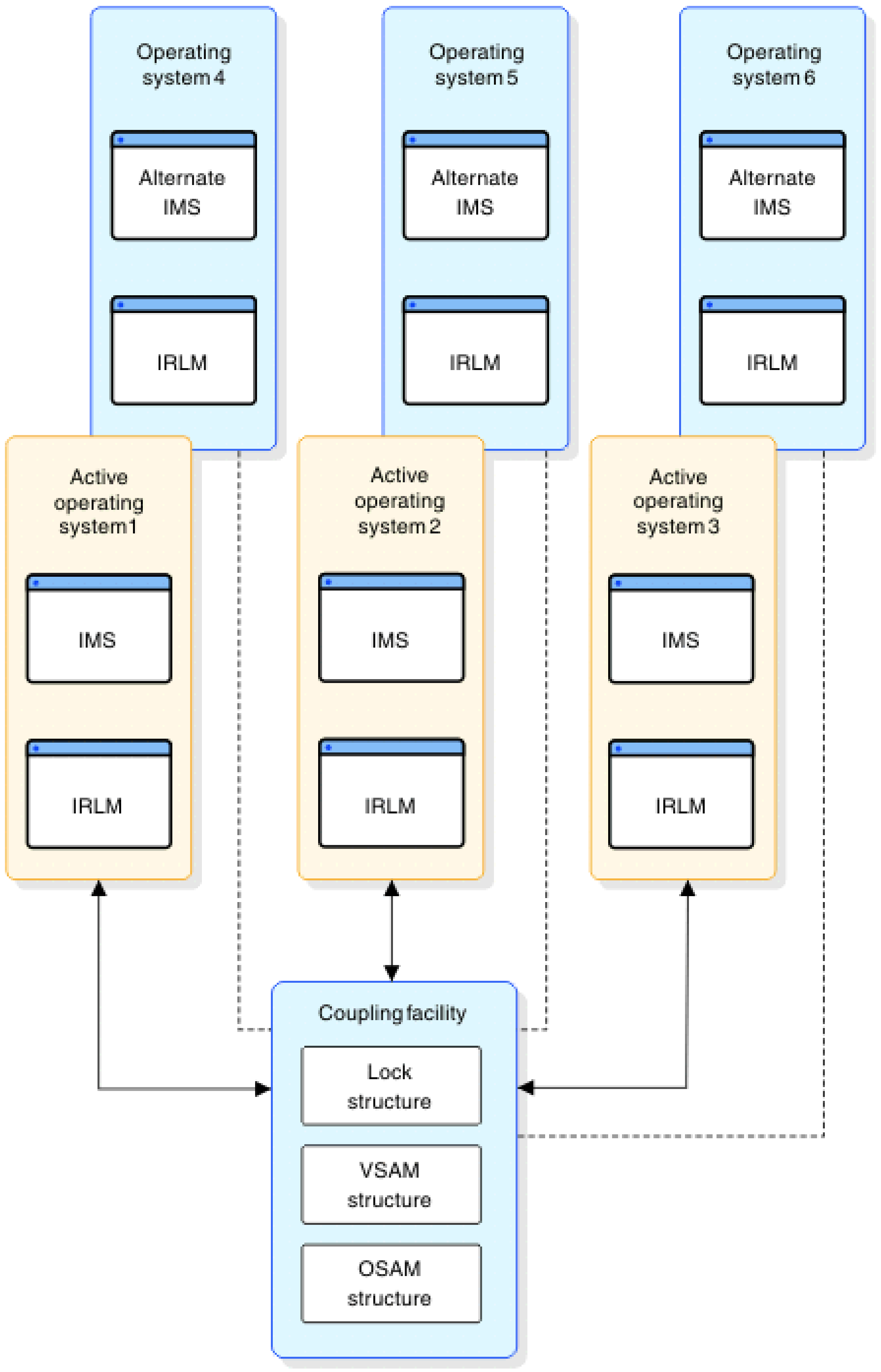


Figure 40. Sysplex data-sharing configuration with three structures on one coupling facility with backup in an XRF environment

One structure on one coupling facility

The following configuration shows multiple z/OS images running in a data-sharing IMSplex using IRLM. IMS is running on each z/OS image. One coupling facility is being used. The lock structure is on the coupling facility. No OSAM or VSAM structures are on the coupling facility. This configuration results in a data-sharing IMSplex using the notify protocol for buffer invalidation.

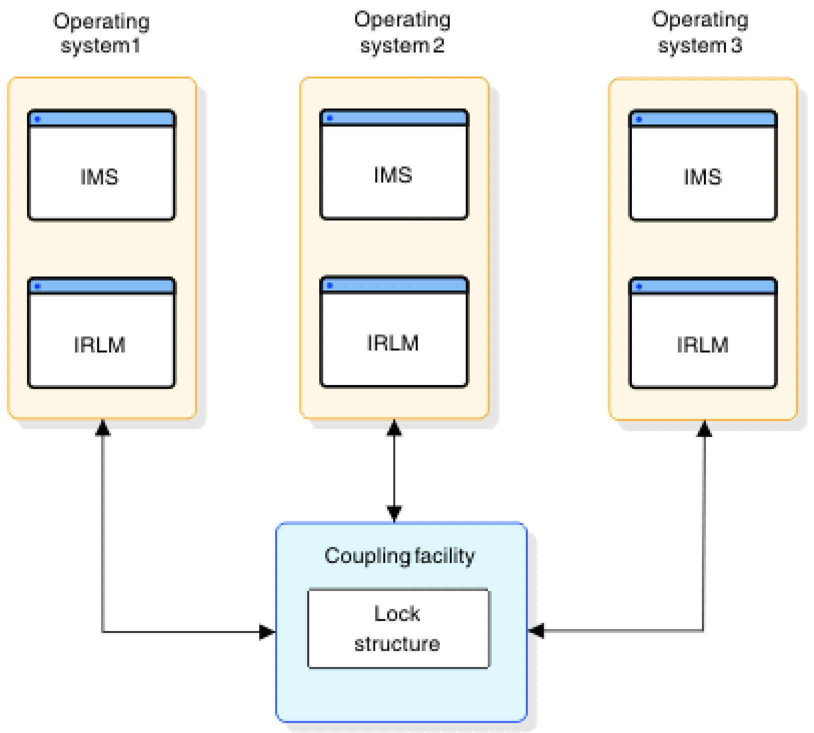


Figure 41. One structure on one coupling facility resulting in sysplex data sharing

When to use sysplex data sharing

The two primary reasons for moving to a sysplex data-sharing environment involve cost and availability. With z/OS sysplex data sharing, you can take advantage of the lower cost of the CMOS-based hardware, spreading the workload to multiple CPUs while maintaining one logical view of your databases; this can be done without making any application software changes.

Availability in this environment is improved, because if one CPU in a sysplex is lost, the workload can be moved to the remaining CPUs.

Modifying batch jobs for sysplex data sharing

Each IMS connected to an OSAM or VSAM structure uses one connection or slot to each coupling facility structure. Therefore, every batch job running in data-sharing mode uses a slot.

Any jobs in the same data-sharing group (started after the combination of active online and batch jobs) that exceed the maximum are unable to run. If you convert these batch jobs to batch-oriented BMP jobs, jobs are more likely to be able to run because BMP jobs run under the online region coupling facility connection.

For more information about converting batch jobs to batch-oriented BMP jobs, see *IMS Version 15.2 Application Programming*.

Calculating the size of coupling facility structures

The size of structures in the coupling facility is determined using a z/OS formula. You must calculate certain IMS values in this formula.

Recommendation: Use transaction rates of peak processing periods for calculating structure sizes. Doing so allows you to avoid resetting structure sizes. Overestimating sizes causes no problems, while underestimating sizes can cause abends.

Calculating the size of the lock structure

The size of a coupling facility lock structure is dependent on a variety of factors that are unique to your installation, such as transaction rates, update rates, contention rates, commit frequency, and so forth. Consequently, there is no way to calculate the optimum size of a lock structure that does not involve some estimation, monitoring, and adjustment.

The simplest method is to start with a likely size that is a power of 2, such as 256 MB or 512 MB. Then monitor your update and contention rates. Tune the lock structure size to lower contention to an acceptable level and to ensure that enough room is available for modify (update) locks. Lock structures that are too big can also adversely affect performance.

You can also use the z/OS Coupling Facility Structure Sizer Tool (CFSizer). CFSizer is a web-based application that calculates the structure size based on the input data you provide. If you use CFSizer, you still need to monitor your update and contention rates and adjust the lock structure size as necessary. To use the CFSizer tool, go to http://www.ibm.com/systems/support/z/cfsizer/.

After setting the initial value, monitor the use of the lock structure using IRLM messages, some of which suggest that you increase the size of the structure. You will begin to receive IRLM messages when 50% of the current lock structure is used.

The coupling facility lock structure contains two parts. The first part is a lock entry table used to determine if there is inter-IMS read/write interest on a particular hash class (resources that hash to a particular place in the lock table). The second part is a list of the update locks that are currently held (sometimes called a modify lock list or record list table). The division of the lock structure storage between these two components can be controlled by the user through the IRLM DXRJPROC procedure or by an IRLM **MODIFY** command. If the user does not specify how the structure is to be split, then IRLM will attempt to divide it with a 1:1 ratio between lock table entries and record list storage. The total size of the lock structure must be large enough to prevent performance problems by limiting hash contention and to prevent failures resulting from lack of record list storage to write a MODIFY entry (RLE). Proper specification for the number of lock table entries can help avoid hash contention.

IRLM reserves 10% of the record table entries for “must complete” functions (such as rollback or commit processing) so that a shortage of storage does not cause an IMS system failure. However, if storage runs short in the record table, there can be an impact on availability (transactions are terminated), response time, and throughput.

Considerations for the lock entry size

The lock entry size is the number of bytes required for lock contention control information (that is, individual entries in the lock table). The lock entry size is determined by the maximum number of IRLM members in the data-sharing group, as defined by the value specified on the **MAXUSRS=** parameter in the IRLM startup procedure. The more users in the group, the more bytes are needed to manage each lock table entry. The lock entry size can be a value that is an exact power of 2 in the range 2 - 32, and the default lock entry size is 2 bytes.

The lock entry size and the number of lock table entries of the first IRLM to join the group causing structure allocation determine the storage size needed for the lock table and the lock table entry width for the whole group. By adjusting the value of the **MAXUSRS=** parameter to the actual number of IRLMs that are connected to the data sharing group, you maximize the amount of LTE space available from the defined structure size. This can help avoid false contention.

Storage estimate for the lock structure

For installation planning purposes, the initial size of the lock structure is based on how much updating you do.

Recommendation: If you do not specify the LTE= keyword in the IRLM DXRJPROC procedure, choose a value for the INITSIZE that is a power of 2. This enables IRLM to allocate the coupling facility storage so that half will be used for lock table entries and the remainder for the record table entries. If a 1:1 split occurs and total size is not a power of 2, you may experience severe shortage of space for the record table entries, resulting in IMS or possible IRLM failures. (This will occur because the number of lock table entries requested on CONNECT must be a power of 2.) The record table is susceptible to storage shortages if the structure is too small or if the allocation of the lock table leaves too little storage for the record table.

When specifying a value for the LTE= parameter in the IRLM DXRJPROC procedure, or when issuing the **`IRLM MODIFY SET,LTE=`** command, you should monitor XES contention rates to determine the optimum value for your normal environment. If the contention rates appear too high, then increase the LTE= value to the next power of 2, keeping in mind that any increase in the size of the lock table will cause a corresponding decrease in the record table, unless the structure size is also increased. If you have little contention and want more storage available for record table entries, then decrease the LTE= value by a power of two. Anytime the number of lock table entries are decreased, it is good to monitor contention rates for a period of time.

Since the structure allocation is performed at CONNECT time, any change made to the LTE= value does not take affect unless the group is terminated, structure forced and the group restarted or a REBUILD is done. Also, the LTE= value of the first IRLM to CONNECT dictates the coupling facility attributes used by the group.

Automatic altering of the size of the lock structure

z/OS can automatically expand or contract the size of a lock structure in the coupling facility if it needs storage space. The ALLOWAUTOALT parameter in the CRFM policy specifies whether system-initiated alters (automatic alters) are allowed for the lock structure. If no competing resource is using the coupling facility, ALLOWAUTOALT(YES) might be desirable to allow the lock structure to change as the workload grows. However, with the parameter set to YES, another resource can force the lock structure to decrease, possibly causing an IRLM out-of-storage condition in the lock table during coupling facility use. In this case, the application terminates abnormally with a U3307 abend. In addition, the IMS control region might terminate with a U0113 or U1027 abend.

If you need the lock structure to be stable, use ALLOWAUTOALT(NO).

Related reference

MAXUSRS= parameter for procedures (System Definition)

Calculating the size of OSAM and VSAM structures

For sysplex data sharing, the size of the structure required by each access method depends on the number of buffers defined to the access method by each IMS in the data-sharing group. The buffers for all IMS control regions and batch jobs that are data sharing and registered to IRLM must be counted.

The following formula is for calculating the count of OSAM buffers:

```
OSAM buffer count  =  #osambfrs/IMS1 + #osambfrs/IMS2 + ...  #  osambfrs/IMSn
```

The following formula is for calculating the count of VSAM buffers:

```
VSAM buffer count  =  #vsambfrs/IMS1+ #vsambfrs/IMS2 + ... #vsambfrs/IMSn
```

Here is an example of sizing a VSAM structure with 2 IMS systems (IMS1 and IMS2):

- IMS1 has a total of 700 buffers (400 VSAM buffers plus 300 Hiperspace buffers) based on the following specifications:

  ```
  VSRBF=512,30
  VSRBF=1024,20,I
  VSRBF=1024,10,D
  VSRBF=2048,40
  VSRBF=4096,100,HS100,HSR
  VSRBF=8192,100,HS100,HSR
  VSRBF=16384,50,HS50,HSR
  VSRBF=32768,50,HS50,HSR
  ```

- IMS2 has a total of 400 buffers based on the following specifications:

```
VSRBF=512,30
VSRBF=1024,20,I
VSRBF=1024,10,D
VSRBF=2048,40
VSRBF=4096,100
VSRBF=8192,100
VSRBF=16384,50
VSRBF=32768,50
```

By adding the total number of buffers for both IMS systems (700 + 400), you get the number to be used for sizing the VSAM structure; 1100.

Recommendation: Review the size of the OSAM and VSAM Structures any time OSAM or VSAM buffer pool definitions are changed. You might encounter poor data sharing performance if these structures are not large enough.

An OSAM structure may optionally be used for caching data. If data caching is not used, the structure contains only directory entries. If data caching is used, the structure contains directory entries and data elements. In this case the size of the structure must allow for data elements. OSAM data is stored in the structure as multiples of 2 KB data elements. Using data caching requires a directory-to-element ratio to be specified. This ratio causes the structure to be subdivided with directory entries and data elements. The ratio is specified with the CFOSAM= keyword on the CFNAMES parameter statement.

For more information on the ratio is specified with the CFOSAM= keyword on the CFNAMES parameter statement, see *IMS Version 15.2 System Definition*.

Requirement: The OSAM buffer count must include any sequential-buffering buffers that are defined. The VSAM buffer count must include any hiperspace buffers that are defined.

z/OS formula

After you have calculated the OSAM buffer count, put the result in the TDEC field of the z/OS cache structure size formula. Then do the same for the VSAM buffer count. (The calculation is done separately for each access method.) TDEC is the total directory entry count, or the maximum number of concurrently executing parallel units of work. For more information on TDEC, See Defining the coupling facility. For more information on the z/OS cache structure size formula, see Determining CF cache structure size.

For information on OSAM database coupling facility caching, see *IMS Version 15.2 System Definition*.

Changing the size of OSAM and VSAM structures

You can dynamically change the size of OSAM and VSAM cache structures or reapportion objects within a structure by using Structure Alter.

You can also automatically perform Structure Alter when a structure fills to change the structure size and element-to-entry ratio. A coupling facility level of 1 is required to support Structure Alter.

For more information about Structure Alter, see "Using structure alter for CQS" on page 167.

Recovering from sysplex data-sharing failures

Multiple failures can occur in a sysplex data sharing environment.

If multiple failures occur simultaneously (for example, IRLM and OSAM structures fail on the same coupling facility), each component (IRLM, IMS, and so on) takes the appropriate recovery action.

Coupling facility connection failures

Various types of problems can cause the connection between IMS and the coupling facility to be lost. For example, the connection is lost if the physical link (a fiber optic cable) is cut, after power failures, hardware check stops, coupling facility microcode failures, and similar events.

If there are multiple links between an IMS and the coupling facility, loss of one link has no effect on IMS or the IRLM. However, if the link that is lost is the last link, it must be recovered before data sharing on that IMS can be restarted. Processing still continues for all other IMS systems in the data-sharing group.

When the last connection between IMS and the coupling facility is lost:

- If the connection was between an IMS and an OSAM or VSAM structure, IMS drops out of data sharing. Any transactions trying to access shared data terminate with a U3303 abend. However, transactions accessing nonshared databases can still be processed.
- If the connection that failed was between an IRLM and the IRLM lock structure, the IRLM disconnects from the data-sharing group. Any transactions trying to access shared data terminate with a U3303 abend.

When connection between the coupling facility and IMS is reestablished, z/OS notifies IMS and IMS resumes data sharing.

The following figure shows the connection loss between IMS, OSAM, and VSAM structures as a result of a coupling facility connection failure.

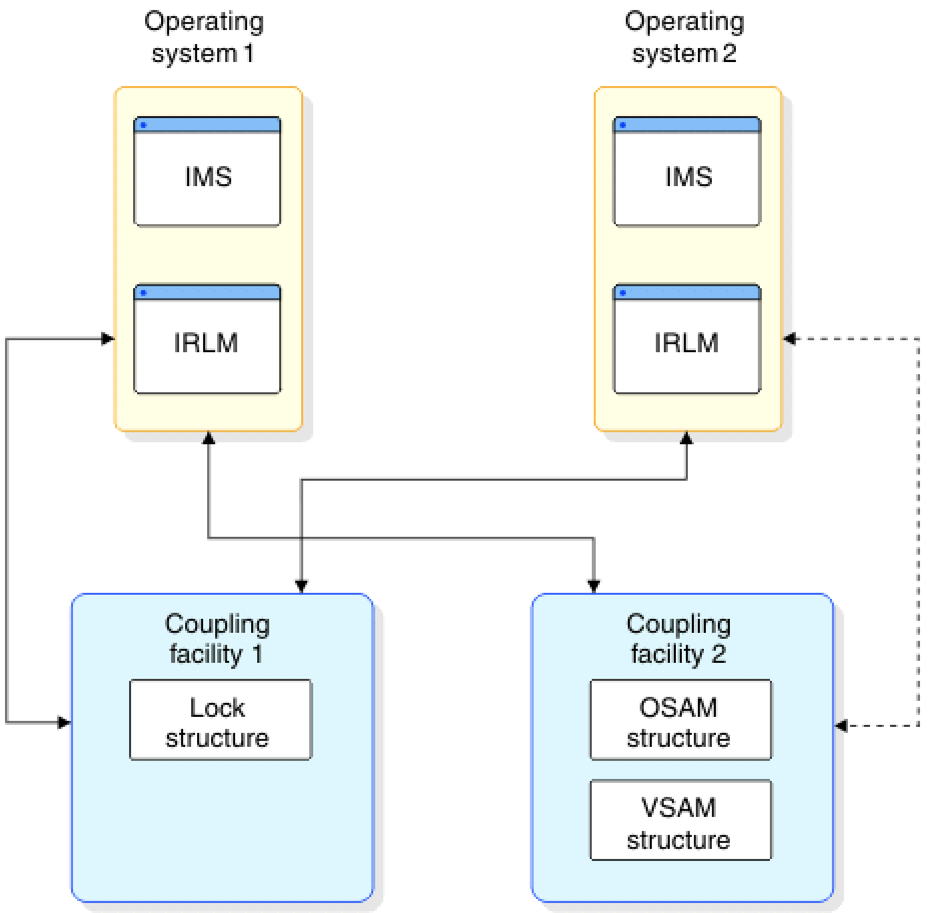


Figure 42. Connection loss between IMS and OSAM and VSAM structures

The connection that is lost is the link between the MVS2 and coupling facility CF02. When the last link fails:

- IMS2 drops out of data sharing and only nonshared databases can be accessed on the MVS2 system. IMS1 and IMS3 continue data sharing.
- Transactions trying to access shared data on MVS2 terminate with a U3303 abend.

In this circumstance, you receive the DATA SHARING STOPPED message. You can enter the z/OS **DISPLAY XCF,STRUCTURE,STRNAME=** command to determine what failed. The displayed output, as shown in the following example, shows that the MVS2 link to the coupling facility failed. MVS1 and MVS3 links are still active.

```
11:44:37.84 D XCF,STRUCTURE,STRNAME=OSAMSESXI
             ⋮
```

```
              & AMPERSAND DENOTES CONNECTOR WHO LOST CONNECTIVITY TO STRUCTURE
              CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
              ---------------- -- -------- -------- -------- ------------
              IMS1             03 0003000D MVS1     DLIOCSA8 0066 ACTIVE
             &IMS2             01 0001001D MVS2     DLIOCSB8 0066 FAILING
              IMS3             02 00020014 MVS3     DLIOCSC8 0066 ACTIVE
```

In the online environment, the MVS2 link to the OSAM structure is reestablished, MVS notifies IMS, and data sharing resumes. In the batch environment, you need to resubmit the batch job when the link is reestablished. You will receive the CF INITIALIZATION COMPLETE message, indicating IMS has resumed data sharing. If you issue a **DISPLAY XCF,STRUCTURE** command, display output shows that MVS2 is connected (ACTIVE).

Coupling facility structure failures

When a coupling facility structure fails in a IMS batch environment, all batch jobs terminate. In a DB/DC or DBCTL environment, all IMS systems in the data-sharing group temporarily suspend data sharing.

When a structure fails, what happens depends on the operating environment:

Batch environment
All IMS batch jobs that were connected to the coupling facility terminate with a U3303 abend.

DB/DC or DBCTL environment
All IMS systems in the data-sharing group temporarily suspend data sharing. If the structure that failed was an OSAM or VSAM structure, IMS tries to rebuild the structure. If the rebuild is successful, all IMS systems connect to the new structure and data sharing continues. If the structure that failed was an IRLM structure, all IRLMs try to rebuild the IRLM structure. If the rebuild is successful, all IRLMs connect to the new structure and locking for data sharing continues.

Example: Figure 42 on page 267 shows the configuration for an OSAM structure failure. When the OSAM structure on coupling facility CF02 fails, the following happens:

- All IMS systems suspend data sharing. If you display the status of the structure at this point (using the **DISPLAY XCF,STRUCTURE,STRNAME=OSAMSESXI** command), output shows that the structure is in the quiesce phase of rebuild, as shown in the following example.

```
08:50:59.29 D XCF,STRUCTURE,STRNAME=OSAMSESXI
08:51:13.32 IXC360I  08.50.59  DISPLAY XCF 143
            STRNAME: OSAMSESXI
             STATUS: REASON SPECIFIED WITH REBUILD START:
                       STRUCTURE FAILURE
                     REBUILD PHASE: QUIESCE
              POLICY SIZE    : 2048 K
              PREFERENCE LIST: CF02     CF01
              EXCLUSION LIST IS EMPTY

              ⋮

             * ASTERISK DENOTES CONNECTOR WITH OUTSTANDING REBUILD RESPONSE
             CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
             ---------------- -- -------- -------- -------- ---- -------
            *IMS1             03 00030015 MVS1     DLIOCSA8 003A ACTIVE
            *IMS2             01 00010026 MVS2     DLIOCSB8 0038 ACTIVE
            *IMS3             02 0002001D MVS3     DLIOCSC8 0038 ACTIVE
```

- When all IMS systems suspend data sharing, a new structure is built. All IMS systems are then connected to the new structure and data sharing resumes. If you display the status of the new structure after rebuilding, output is similar to that shown in the following example.

```
08:52:39.78 D XCF,STRUCTURE,STRNAME=OSAMSESXI
08:52:40.43 IXC360I  08.52.39  DISPLAY XCF 168
            STRNAME: OSAMSESXI
             STATUS: ALLOCATED
              POLICY SIZE    : 2048 K
              PREFERENCE LIST: CF02     CF01
              EXCLUSION LIST IS EMPTY

             ACTIVE STRUCTURE
             ----------------
```

```
               ALLOCATION TIME: 03/25/94 08:51:27
               CFNAME         : CF01
               COUPLING FACILITY: ND01...
                                   PARTITION: 0    CPCID: 00
               ACTUAL SIZE    : 2048 K

               ⋮

              CONNECTION NAME  ID VERSION  SYSNAME  JOBNAME  ASID STATE
              ---------------- -- -------- -------- -------- ------------
              IMS1             03 00030015 MVS1     DLIOCSA8 003A ACTIVE
              IMS2             01 00010026 MVS2     DLIOCSB8 0038 ACTIVE
              IMS3             02 0002001D MVS3     DLIOCSC8 0038 ACTIVE
```

Failure in rebuilding a coupling facility

You can initiate a rebuild of the coupling facility by issuing the z/OS **SETXCF START,REBUILD** command.

z/OS can also initiate a rebuild in response to:

- A coupling facility structure failure
- A connection failure if the active System Failure Management (SFM) policy threshold for rebuilding the coupling facility has been reached

If the rebuild fails, IMS stops data sharing. Transactions trying to access shared data terminate with a U3303 abend. If the rebuild failure was associated with an OSAM or VSAM structure, transactions accessing nonshared databases can still be processed. If the failure was associated with an IRLM structure, the IRLM disconnects from the data-sharing group.

Coupling facility connection failure during IMS restart

If the connection of the active system to the coupling facility fails during IMS restart, restart processing continues.

If the failure is on the XRF alternate subsystem and the alternate subsystem is unable to connect to the coupling facility, restart cannot proceed.

Failure when IMS tries to identify to an IRLM

A failure occurs if IMS cannot identify itself to the IRLM. When this happens, IMS issues a message indicating the reason for the failure and then abends.

You can gather information to determine what action to take using the **DISPLAY XCF,STRUCTURE** and **MODIFY irlmproc***x***,STATUS** commands.

Except for this identify failure, IRLM failure processing is the same as with nonsysplex data sharing. IMS:

- Backs out all uncommitted database changes.
- Removes authorization of all shared databases through DBRC. The system is stopped.
- Waits for the IRLM to be restarted.

z/OS and other failures requiring system recovery

If z/OS fails or other failures occur that require system recovery (for example, a power outage), IMS processing is the same as it is in the nonsysplex data-sharing environment.

For the failing system, operations consist of:

- Restarting z/OS
- Restarting the IRLM
- In the batch environment, backing out batch jobs if necessary
- In the IMS DB/DC or DBCTL environment, emergency restarting IMS (using the **/ERESTART** command)

You can restart z/OS and the IRLM on a different system. Also, batch backout and emergency restart can be done on any z/OS and IRLM in the data-sharing group.

For the non-failing systems:

- Locks are retained for the failed IMS.
- Locks are released as the failed IMS restarts and recovers its databases.
- Database buffers for the failed IMS are invalidated.
- Database error conditions are sent to the other IRLMs.

Synchronizing IMS restart after a multiple IMS system failure

Multiple IMS system failures can occur in a data-sharing environment. When this happens, you might want to synchronize the startup of all of the failed IMS systems. Consider also synchronizing the startup of all IMS systems at a disaster recovery site.

If a sysplex failure or a restart at a disaster recovery site occurs, any new authorizations to databases or DEDB areas held by a failed IMS system are not granted until the failed IMS system is restarted. If each IMS system that failed is restarted one at a time and begins normal processing, authorization to databases or DEDB areas that were authorized by failed IMS systems will also fail. For full-function databases, message DFS047 is issued with a return code of X'09'. For DEDB areas, message DFS3709 is issued with a return code of X'09'.

After all of the failed IMS systems have been restarted, the databases and DEDB areas have to be manually started. Using the OPTION keyword and SYNCPLEX parameter on the /ERE command, you can serialize the restart of failed IMS systems. Issue the command on each IMS failed system.

After IMS restart processing completes, write-to-operator-with-reply (WTOR) message DFS3067A is issued. When all of the IMS systems receive this message, the operator can reply to each WTOR. Normal processing resumes, and database and DEDB area authorizations can be granted.

The OPTION SYNCPLEX keyword and parameter are valid only an /ERE command.

Authorizing databases after an IMS failure

In a data-sharing environment, if an IMS system fails, another IMS system attempting to authorize a database or DEDB area held by the failed IMS system will fail.

For full-function databases, message DFS047 is issued with a return code of X'09'. For Fast Path DEDBs, message DFS3709 is issued with a return code of X'09'. The X'09' return code indicates that the database or area was previously authorized in one or more online IMS systems. The IRLM exit verification indicates that the IMS systems are inactive. Therefore, the identified database or DEDB area cannot be authorized.

To prevent this from occurring, specify the IRLM parameter SCOPE=NODISCON. This indicates that the IRLM is in a data-sharing environment, and that intersystem sharing is to be performed. The IRLM remains connected to the data-sharing group, even when no IMS systems are identified to it.

If an IMS system fails, the IRLM remembers the status of the failed IMS. Any new authorizations to databases that are held by the failed IMS will be allowed. The IRLM exit verification processing that occurs at restart recognizes the status of the failed IMS system and grants new authorizations.

Chapter 17. Planning for VTAM generic resource groups

This topic describes the planning and administration tasks associated with using VTAM generic resource groups.

Requirements for using VTAM generic resource groups

There are several requirements for balancing sessions using VTAM generic resource groups.

These requirements include:

- A sysplex environment.
- A z/OS coupling facility structure named ISTGENERIC defined in the active CFRM policy for the sysplex environment.
- A connection between the VTAM APPN node and the ISTGENERIC coupling facility structure.
- All IMS subsystems participating in the generic resource group reside in the sysplex environment.
- Each IMS subsystem participating in the generic resource group shares the same IMS generic resource name.
- An APPC generic resource name is defined to z/OS for LU 6.2 communications. Define this name on the GRNAME parameter of the LUADD statement in the APPC/MVS APPCPMxx member.
- All IMS systems that belong to the generic resource group must be defined with equivalent specifications.

Related reading: For information on VTAM and defining the ISTGENERIC coupling facility structure, see *z/OS Communications Server: SNA Network Implementation Guide*.

VTAM generic resource group restrictions

When creating a VTAM generic resource group, system programmers should be aware of the restrictions.

Restrictions include:

- A particular IMS cannot be a member of more than one VTAM generic resource group.
- The generic resource name specified on the GRSNAME parameter must not match the name of an XRF USERVAR.

  For more information on XRF and VTAM generic resources, see "XRF and VTAM generic resources" on page 273.

VTAM generic resource affinity

After VTAM selects a specific IMS subsystem in a generic resource group to perform the work of a specific terminal, VTAM generic resource (VGR) affinity is established, and all of the terminal's subsequent sessions connect to the same IMS subsystem until IMS or VTAM resets the terminal's affinity.

IMS assigns VGR affinity management to IMS or VTAM for each session depending on the status recovery mode of that session. IMS ignores the GRAFFIN= parameter, which is obsolete for VGR. Multiple Systems Coupling (MSC) links use local affinity only.

Note: For ISC terminals and MSC links, VTAM and IMS do not release VGR affinities until all parallel sessions have been terminated.

VGR affinity management

Either IMS or VTAM can manage VGR affinities. An important difference between the two is the ability of each to reset affinities for terminals in the event of an IMS failure. If a terminal or MSC link has an affinity with an IMS that fails, the terminal cannot continue work until its affinity is reset.

- IMS-managed – IMS tells VTAM when to delete VGR affinity when a session (or IMS) terminates. IMS deletes the affinity unless significant status for a terminal exists on a local IMS system.
- VTAM-managed – VTAM deletes VGR affinity when a session (or IMS) terminates.

When VTAM manages affinities, if IMS fails, VTAM can reset the affinities of terminals or MSC links that had sessions open with the failed IMS. With their affinities reset, the terminals can open a new session with another IMS in the generic resource group.

When IMS manages affinities, if IMS fails, IMS might not be able to reset the affinities between the terminals and the failed IMS. In this case, the terminals cannot open new sessions until the failed IMS is restarted. In the case of MSC links, the sessions cannot be moved to cloned links on another IMS when the affinity is set.

The type of terminal and the status recovery mode you set for the terminal determine whether IMS or VTAM manages VGR affinities.

IMS-managed affinity

IMS manages VGR affinity for the following:

- Static terminals with recovery mode of LOCAL, where IMS sets VGR affinity because the terminal must not be allowed to access other IMS systems while significant status exists on the local IMS system. IMS ensures that VGR affinity matches RM affinity.
- Dynamic STSN terminals with status recovery mode of LOCAL, where IMS sets VGR affinity because the terminal must not be allowed to access other IMS systems while significant status exists on the local IMS system. IMS ensures that VGR affinity matches RM affinity because the user name and status recovery mode must be provided at logon.
- MSC links use local mode and IMS managed affinity only, and do not have RM support.

Creating a VTAM generic resource group

You create a VTAM generic resource group by specifying the same generic resource name for all the IMS systems participating in the VTAM generic resource group.

You can specify the VTAM generic resource name for each IMS in two ways:

- The GRSNAME= startup parameter in the IMS and DCC procedures.
- The GRSNAME parameter of the **/START VGR** command when starting the control region of an IMS.

Restriction: The generic resource name specified on the GRSNAME parameter must not match the name of an XRF USERVAR.

Related reading:

- For more information on the GRSNAME= startup parameter, see *IMS Version 15.2 System Definition*.
- For more information on the **/START VGR** command, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Specifying APPC generic resource names

Specifying an APPC generic resource name enables LU 6.2 devices to participate in the session balancing provided by a generic resource group. Specify the APPC generic resource name on either the GRNAME parameter of the LUADD statement in the APPC/MVS APPCPMxx member or with the **SET APPC** z/OS operator command.

Restriction: The APPC generic resource name must be different from the name you specify on the GRSNAME execution parameter.

Initiating sessions with IMS in a VTAM generic resource group

Terminal sessions can be initiated either from IMS or from a terminal. All sessions initiated by IMS are generic sessions; that is, sessions in which affinity is established. MSC sessions are initiated with an **/RSTART LINK** command.

Sessions from a terminal can be initiated in the following ways:

- You can initiate a session with any IMS in a generic resource group by specifying the GRSNAME execution parameter. An affinity with the selected IMS is then established.
- You can initiate a session with a specific IMS by specifying the APPLID name of the system. No affinity is established.
- If an IMS in a generic resource group is also part of an XRF with MNPS system, you can initiate a session with that specific IMS using the MNPS name instead of the APPLID name. No affinity is established.

XRF and VTAM generic resources

Extended Recovery Facility (XRF) systems help ensure session persistence and the availability of IMS resources. There are two types of XRF systems: XRF systems that use VTAM multinode persistent sessions (MNPS) and XRF systems that use a USERVAR. XRF with MNPS systems can participate in a VTAM generic resource group, but XRF with USERVAR systems cannot.

XRF with MNPS and VTAM generic resources

For XRF with MNPS systems in a generic resource group, VTAM associates the generic resource name with the MNPS name of the XRF with MNPS systems instead of the APPLID name. Terminals that want to log on directly to an IMS that is part of an XRF with MNPS system must use the MNPS name.

Including an XRF with MNPS system in a generic resource group does not guarantee session persistence for terminals logging on using the generic resource name unless every IMS in the generic resource group is also part of an XRF with MNPS system.

If you use XRF with MNPS and are migrating off of the 3745 controller, be aware that you might encounter some performance limitations.

Recommendations: To minimize these performance limitations:

- Use the Communication Controller for Linux® on System z® (CCL) as a replacement for the 3745 controller, which allows continued use of XRF.
- Migrate from XRF to VTAM Generic Resources (VGR), which requires a parallel sysplex environment.

XRF with USERVAR and VTAM generic resources

Although XRF systems with USERVAR cannot participate in a generic resource group, they can coexist in the same sysplex as a generic resource group. In this case, generic resource names and USERVARs must be different.

XRF with USERVAR systems can participate in a shared-queues environment.

Changing the logon procedure

You can provide the option of logging on using either a generic resource name or an IMS APPLID name; therefore, plan to make one or both of these names available in the logon procedure.

Overriding the APPLID field

Each IMS identifies itself to VTAM using the application name you specify in the APPLID= keyword of the IMS COMM system definition macro and in the VTAM APPL definition statement.

In order to keep the IMS definitions as similar as possible, you can specify the same name in the APPLID= keyword of the COMM macros for all the IMS systems in a generic resource group. You must then override this name by specifying a unique APPLID name on the APPLID1 execution parameter.

Similarly, you can override the PASSWD field in the COMM macro by using the PASSWD= execution parameter.

Related reading: For more information on the APPLID1 and PASSWD execution parameters, see *IMS Version 15.2 System Definition*.

Determining when affinities are terminated

When you terminate a session by logging off, IMS terminates the affinity between your node and the IMS VTAM generic resource member. However, in some situations, affinities persist.

You can use several methods in order to terminate these affinities.

Affinity for MSC and ISC parallel sessions is maintained as long as one or more sessions are active, or at least one session has significant status and the recovery mode is local. MSC sessions terminate the VTAM affinity when a **`/PSTOP LINK`** command is issued for all parallel link sessions. MSC recovery mode is always local.

ISC sessions terminate the VTAM affinity when:

- A **`/QUIESCE NODE`** command is issued (cold termination) for all sessions and recovery mode is local or global.
- A **`/CLOSE NODE`** command is issued (warm termination) for all parallel sessions and the recovery mode is global.

The VTAM affinity is also terminated when IMS is shut down normally by a **`/CHE FREEZE|PURGE|DUMPQ`** command and all sessions are successfully terminated. The **`/CHE FREEZE|PURGE|DUMPQ LEAVEGR`** command terminates all affinity unconditionally. Whether the shutdown is initiated by IMS (**`/CHE`**) or by the session (**`/PSTOP`**, **`/CLOSE`**, or **`/QUIESCE`**), the affinity is maintained if any parallel MSC or ISC sessions between the same IMS systems remain active or have been terminated but still have a significant status.

If there are parallel sessions using both MSC and ISC between the same IMS systems, then the previously mentioned affinity rules are applied simultaneously. That is, the affinity is maintained for both MSC and ISC parallel sessions if any link or node has significant status.

In order for IMS to terminate the affinity in all cases, the VTAM ACB must be opened with a **`/START DC`** command.

MSC significant status (MSC recovery mode is always local)

- Link is in active status
- Link is in ERE status

ISC significant status

- Node is in active status
- Node is using RM local recovery mode (SRMDEF=Local for the node) and is in warm terminate status

To display the list of IMS systems and their affinities with specific nodes or MSC links, use the **/DISPLAY AFFIN** command.

Related reading: For information on the **/DISPLAY AFFIN NODE** or **/DISPLAY AFFIN LINK** command, see *IMS Version 15.2 Commands, Volume 1: IMS Commands A-M*.

For information on how to terminate affinities that persist, see "Terminating affinities that persist" on page 275.

Terminating affinities that persist

A terminal cannot participate in session balancing if it has a persisting affinity with a particular IMS from a previous session. When VTAM manages the VGR affinity, affinity is automatically reset at session termination.

When IMS manages the VGR affinity, an affinity persists after a logoff or IMS failure in each of the following situations:

- The terminal is static with LOCAL status recovery mode, and has end-user significant status (Conversation, STSN, Fast Path, or full-function response mode).
- The terminal is ETO SLUP or 3600 Finance terminal with LOCAL status recovery mode and RCVYSTSN=YES.
- The terminal is an ISC parallel-session terminal with LOCAL status recovery mode that has not quiesced.
- The terminal is an ISC parallel-session terminal, and one or more of the parallel sessions has not quiesced.
- The MSC link is in an ERE status.

You can terminate an affinity using any of the following methods:

- Enter the **/CHECKPOINT** command with the LEAVEGR keyword.
- Use the Logoff or Signoff exit routine to reset terminal status.
- Cold start the IMS TM subsystem, and specify the GRSNAME when restarting IMS.

Related reading:

- For information on the **/CHECKPOINT** command and the LEAVEGR keyword, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.
- For information on the Logoff and Signoff exit routines, see *IMS Version 15.2 Exit Routines*.

Dropping an IMS system from a generic resource group

You can use the **/STOP VGRS** command to drop an IMS out of a generic resource group. Similarly, you can use the **/START VGRS** command to bring that IMS back into the generic resource group.

Resetting terminal status

You can reset terminal status (and thus terminate affinities) using commands, exit routines or parameters.

The following list includes the ways that you can reset the terminal status:

- An authorized command
- The Logoff (DFSLGFX0) exit routine
- The Signoff (DFSSGFX0) exit routine
- The RCVYFP, RCVYRESP, RCVYSTSN, and RCVYCONV parameters
- The NONE status recovery mode

The Logoff (DFSLGFX0) exit routine resets a terminal's status during a logoff. Similarly, you can use the Signoff (DFSSGFX0) exit routine to reset a user's status during a signoff. Resetting terminal or user status enables IMS to terminate an affinity with the terminal, enabling the terminal to log on again and participate in session balancing.

Related reading:

- For more information on the Logoff and Signoff exit routines, see *IMS Version 15.2 Exit Routines*.
- For more information on the NONE status recovery mode, see "Status recovery mode for end-user significant status" in *IMS Version 15.2 Communications and Connections*.
- For more information on the RCVYFP, RCVYRESP, RCVYSTSN, and RCVYCONV parameters, see *IMS Version 15.2 System Definition*.

Logging on after IMS failure with affinity

Suppose you are operating in a shared-queues environment, and your terminal is logged onto an IMS in a generic resource group. If that IMS fails, the affinity between that IMS and your terminal might persist.

Consequently, you can have output messages waiting for you on that terminal. To obtain your messages, you can log onto another IMS within the generic resource group, provided that all of the following conditions exist:

- You log on using the APPLID name of another IMS within the generic resource group.
- Your output messages are not locked on the failed IMS, and those messages are not responses to either an IMS conversation or a Fast Path response-mode transaction.
- If you have a resource structure and Resource Manager, the status recovery mode is not LOCAL and there is not end-user significant status on the failed IMS (no RM affinity).

Controlling VTAM generic resource member selection

You can control VTAM's selection of a generic resource member for a terminal when the terminal logs on. To control the selection process, use either the VTAM Generic Resource Resolution exit routine (ISTEXCGR) or the z/OS Workload Manager.

Choosing between these facilities depends on the needs of your application program and on your installation requirements.

The criteria that VTAM uses to select a generic resource member are:

1. Existing affinity. VTAM maintains an affinity table in a list structure called ISTGENERIC on the coupling facility.
2. VTAM Generic Resource Resolution exit routine (ISTEXCGR).
3. z/OS Workload Management.
4. Current session counts.

Related reading:

- For more information on the VTAM Generic Resource Resolution exit routine (ISTEXCGR), see *z/OS Communications Server: SNA Customization*.
- For more information on the z/OS Workload Management, see *z/OS MVS Planning: Workload Management* and *z/OS MVS Programming: Workload Management Services*.

Ensuring consistency across the IMSplex

Inconsistencies in definitions, operator commands and procedures, and exit routines can create problems within a generic resource group.

- At a minimum, they can confuse terminal users, who are unaware of being in session with different IMS systems from one logon to another logon.
- At worst, they can cause processing disruptions.

Restriction: LU6.1 parallel sessions between IMS and another node must all use VGR or non-VGR. LU6.1 parallel VGR and non-VGR sessions (ISC or MSC) cannot be mixed because the first parallel session established between the LU6.1 partners establishes how the partner node is known, by either the real APPLID name (non-VGR) or the GRSNAME (VGR). This condition is remembered by VTAM, and VTAM sets a flag (INRIFLG0 = NIBNNAMS in the ISTNRIPL parameter area) on subsequent parallel session initiations to indicate this name association. IMS uses this flag to establish the session. The mismatch could cause the session initiation to fail (such as DFS3645 Logon Rejected) or the session affinity to be incorrectly managed by IMS.

You can mix VGR and non-VGR between non-parallel LU6.1 sessions, such as simultaneous use of IMS1 to IMS2 using VGR, IMS1 to CICS using non-VGR, or IMS1 to IMS3 using non-VGR.

Recommendations: To ensure consistency across a generic resource group:

- Use equivalent specifications when defining the member IMS systems.

  Example: If one IMS specifies that ETO be included (ETOFEAT=YES on the IMSCTRL macro), all other IMS systems in the generic resource group should specify that ETO be included.
- Specify execution parameters consistently across generic resource members.

  Example: IMS systems in a generic resource group should each specify that ETO be enabled or not be enabled.
- If ETO is included, specify the same information on the ETO descriptors for each IMS generic resource group member.
- Use the same naming conventions for LTERMs and transactions across a generic resource group. If the same LTERM name is assigned to different nodes on different IMS systems, unpredictable results might occur.

  Related Reading: For more information about how to ensure name uniqueness and resource type consistency, see Chapter 18, "Planning for Transaction Manager resources in an IMSplex," on page 279.
- Make all terminal definitions consistent across generic resource members.

  Example: LTERM names should be consistently defined, during IMS system definition, to the same physical terminals on all the IMS systems in the generic resource group.
- MSC links in a generic resource group should be cloned.

  Example: Generic resource groups should have the same MSPLINK, MSLINK, and MSNAME characteristics
- Ensure that master terminal operators (MTOs) on each IMS in the generic resource group have procedures for notifying the other MTOs about the commands they issue. If you are using a single point of control (SPOC), commands go to all IMS systems that can process the commands and there is no need for the notification procedures.

  Example: If you do not have Resource Manager (RM) and one MTO issues a **/STOP** command to stop a user on one IMS in a generic resource group, that user is stopped only on the IMS on which **/STOP** is entered. If you have RM and a resource structure, however, RM manages the user globally and the user is stopped on all the IMS systems in the IMSplex.

  Related reading: For more information on using a SPOC, see "Controlling IMS with the TSO SPOC application" in *IMS Version 15.2 Operations and Automation*.
- Make exit routines functionally equivalent across a generic resource group—especially the Destination Creation exit routine, the OTMA Routing exit routines, the Logon and Signon exit routines, and the Logoff and Signoff exit routines.

Bypassing affinity management during the IMS ESTAE process

If you use IMS to manage generic resource affinities, you can control whether or not IMS uses or bypasses the VTAM generic resource logic in the IMS ESTAE exit. To use the IMS ESTAE process, indicate

GRESTAE=Y for terminal sessions and GRMESTAE=Y for MSC link sessions in the DFSDCxxx IMS.PROCLIB member data set.

GRESTAE=Y or GRMESTAE=Y indicates that IMS should follow the existing ESTAE logic to delete affinity for nodes where no status remains before closing the ACF/VTAM ACB. GRESTAE=N or GRMESTAE=N indicates that IMS should close the ACF/VTAM ACB immediately to expedite IMS termination, and leave affinity for all nodes set.

IMS VTAM generic resource logic in the IMS ESTAE exit attempts to delete generic resource affinity if no terminal or MSC link status remains. The IMS ESTAE routine uses the VTAM delete affinity force option (ENDAFFNF) which does not require the sessions to be disconnected from the VTAM terminal first.

The following table summarizes how generic resource affinities are managed based on whether affinities are VTAM or IMS managed and which GRESTAE options are indicated in the DFSDCxxx IMS.PROCLIB member data set.

Table 25. Generic resource affinities management and GRESTAE options

Affinity manager	**GRESTAE/ GRMESTAE option**	**Generic resource affinities management**
IMS	Y	IMS manages generic resource affinities, including during ESTAE processing. IMS resets affinity during the ESTAE process that occurs during IMS failure.
IMS	N	IMS manages generic resource affinities, except during ESTAE processing. IMS does not reset affinity during the ESTAE process that occurs during IMS failure.
VTAM	Y or N	VTAM manages generic resource affinities.

Chapter 18. Planning for Transaction Manager resources in an IMSplex

This topic describes in detail the functions of the resource structure and Resource Manager (RM) as they relate to managing IMS Transaction Manager resources in an IMSplex.

The resource structure and RM are intended to run in a DB/DC or DCCTL environment, and with CSL operating.

Related reading:

- For more information on administering an IMSplex with CSL, see Chapter 7, "CSL administration," on page 125.
- For information on resource-structure–related parameters for the DFSDCxxx member of the PROCLIB data set, see *IMS Version 15.2 System Definition*.
- For information on user exit routines DFSINSX0, DFSSGNX0, DFSINTX0, and DFSLGNX0, see *IMS Version 15.2 Exit Routines*.
- For a list of the commands that relate to the resource structure and RM, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Resource name uniqueness

IMS ensures that a resource name is active only once in the IMSplex at any particular time. IMS systems within the IMSplex cannot activate the same resource at the same time. The IMSplex automatically enforces resource name uniqueness only when Resource Manager (RM) is active and a resource structure is defined in the coupling facility.

When a resource is active, it is owned by one IMS in the IMSplex. No other IMS can activate the same resource until it becomes inactive on the owning IMS.

Name uniqueness is enforced only for VTAM LTERMs, VTAM single-session nodes, user IDs, and users. IMS no longer supports BTAM terminals.

Name uniqueness is enforced for user IDs only if SGN is not M. User ID name uniqueness enforcement is not automatic. However, name uniqueness for VTAM LTERMs, VTAM single-session nodes, and users is automatically enforced if you have RM and a resource structure defined.

For ISC terminals, if the option for disabling resource sharing in the IMSplex is on in the DFSINTX0 user exit routine, resource name uniqueness is not enforced for static LU 6.1 sessions. Name uniqueness is not enforced for ISC TCP/IP terminals (static and dynamic).

Disabling enforcement of resource name uniqueness

You can disable the enforcement of resource name uniqueness, and TM resource sharing in general, by specifying STM=NO in the DFSDCxxx PROCLIB member data set.

Related reading: See "Transaction Manager resources" on page 280 for more detailed information about how and why IMS enforces name uniqueness for particular resources.

Resource type consistency

Resource type consistency ensures that a name is unique within a group of resources, called a *name type*. The IMSplex automatically enforces resource type consistency when RM is active and a resource structure is defined in the coupling facility.

IMS enforces resource type consistency for the name type of message destination. IMS ensures that a resource defined as a message destination is consistent across the IMSplex.

Message destinations are the following:

- LTERMs
- LTERMs defined as APPC descriptors
- MSNAMEs
- IMS-defined transactions
- CPI-C transactions

Disabling enforcement of resource type consistency

You can disable the enforcement of resource type consistency when a resource structure is present by specifying STM=NO in the DFSDCxxx PROCLIB member data set. After specifying STM=NO, resource type consistency is enforced only for IMS-defined and CPI-C transactions.

Related reading: See "Transaction Manager resources" on page 280 for more detailed information about how and why IMS enforces type consistency for particular resources.

Global callable services

Callable services are provided for user-provided exit routines in order to find resources such as nodes, LTERMs, and users. Callable services returns global resource information shared in the resource structure. If no global information is available, local information is returned by default.

If you want only local information, use the input flag LOCAL on the CSCBLK DSECT, which is defined by the DFSCCBLK macro.

Transaction Manager resources

This topic discusses the following types of Transaction Manager (TM) resources.

TM resources: APPC descriptors

IMS defines APPC descriptors to RM at initialization or during /STA LU62DESC to maintain resource type consistency for message destinations.

IMS ensures that an APPC descriptor name is not used as a transaction, MSNAME, or LTERM name. Descriptors can be defined on multiple IMS systems as name uniqueness is not enforced.

APPC descriptor creation and deletion

At initialization or during /STA LU62DESC, IMS ignores any descriptors already defined as a transaction, MSNAME, or LTERM and issues a warning message. APPC descriptors are deleted from RM when all IMS systems that have defined the descriptor have terminated, warm or cold. Systems that do not define the descriptor do not need to terminate.

TM resources: VTAM LTERMs

IMS defines VTAM LTERMs to RM for resource type consistency for message destinations, name uniqueness for LTERMs, and LTERM status recovery.

LTERM name uniqueness

IMS enforces name uniqueness for VTAM LTERM names to ensure output security within an IMSplex. When an LTERM becomes active, IMS ensures that the LTERM does not become active on any other IMS and is unavailable to any other user or terminal.

For LU 6.1 sessions, LTERM name uniqueness can be ignored by using the DFSINTX0 user exit routine. See *IMS Version 15.2 Exit Routines* for more information about this exit routine.

IMS no longer supports BTAM terminals.

LTERM creation and deletion

When an LTERM becomes active during signon or logon, IMS defines it to the RM. If the name is the same as a transaction, APPC descriptor, or MSNAME, or if the LTERM is active on another IMS, the LTERM does not become active. The signon of a dynamic user continues without an LTERM assigned (but if there is only one LTERM, the signon is rejected), and the logon for a static terminal is rejected. IMS also defines LTERMs to RM when recoverable commands affecting significant status are processed.

When an LTERM becomes inactive and there is no significant status data, IMS deletes the LTERM. An LTERM is deleted in three situations:

- During a logoff or signoff
- When a command deleting significant status is processed
- During resource cleanup after an IMS failure and no significant status exists

TM resources: MSNAMEs

MSC networks use MSNAMEs to define remote IMS systems and logical link paths between remote and local IMS systems in an MSC network. For MSNAMEs, Resource Manager (RM) enforces only resource type consistency for message destinations, but not resource name uniqueness.

When IMS initializes, IMS defines to RM each MSNAME that has a remote SYSID. Because resource name uniqueness is not enforced, the same MSNAMEs can be defined on multiple IMS systems.

Creation and deletion of the MSNAME TM resource

An MSNAME becomes a TM resource to RM at IMS initialization. RM saves the MSNAME in the resource structure. MSNAMEs are not deleted from a resource structure once they are defined and cannot be reused unless the resource structure itself is deleted.

Related reading: For additional information on MSNAMEs and how they are used in a shared resources group, see *IMS Version 15.2 Communications and Connections*.

TM resources: VTAM terminal nodes

IMS defines nodes to RM to enforce name uniqueness for single-session VTAM terminals and to recover node status.

Node name uniqueness

IMS enforces name uniqueness of VTAM terminals to ensure only one terminal can be logged on at one time and to ensure data integrity. These terminals include all VTAM terminals except LU 6.1 (ISC) parallel sessions.

IMS does not enforce name uniqueness for LU 6.1 parallel sessions, but data integrity is ensured by user name uniqueness. For LU 6.1 sessions, node name uniqueness can be ignored by using the DFSINTX0 user exit routine. See *IMS Version 15.2 Exit Routines* for more information on this exit routine.

Note: Some session manager products that assign non-unique ETO node names to terminals logging onto multiple systems in an IMSplex will not work properly with RM. IMS prevents the same non-ISC node name from being active in more than one system at a time.

Node creation and deletion

IMS defines a node to RM when the node becomes active during terminal logon. Logon is rejected if the node is already active on another IMS system, except for ISC parallel sessions. IMS also defines a node to RM when recoverable commands that process significant status are processed. These commands include the commands that set global MFSTEST, STOP, and TRACE.

When a node becomes inactive and there is no significant status data, IMS deletes the node. A node is deleted in three situations:

- During logoff
- When a command deleting significant status is processed
- During resource cleanup after an IMS failure and no significant status exists

TM resources: transactions

IMS defines a transaction dynamically with the type-2 CREATE TRAN command or statically in the system where the application will run or dynamically as a CPI-C transaction executed by an APPC conversation. A transaction can run in multiple systems concurrently as name uniqueness is not enforced.

Within the context of TM resource management, transactions are treated as a resource only to prevent another TM resource from duplicating the transaction name. RM continues to prevent the duplication of transaction names even if you disable TM resource sharing by specifying STM=NO in the DFSDCxxx PROCLIB member data set.

Transaction resource type consistency

IMS ensures that a transaction name cannot be used as an MSNAME, APPC descriptor, or an LTERM name. IMS ignores a CPI-C transaction if the name is defined as a different message destination. APPC still validates the destination and does not allow LTERMs as destinations.

Transaction creation and deletion

IMS defines a transaction to RM when IMS initializes, when online change adds a transaction, or when a transaction is created with the type-2 CREATE TRAN command. IMS ignores the transaction if the name is already defined in the IMSplex as another resource.

IMS identifies a CPI-C transaction to RM during APPC input processing and after it checks to make sure the name is not defined as another message destination already. If it is defined as another resource, IMS uses the defined destination and no error is generated.

Cold-starting an IMS does not delete a transaction.

TM resources: user names

The user name and the user ID are usually the same; however, user exits and descriptors can override the user name. The *user* is the user signed on to a dynamic terminal or parallel session subpool and has associated work and status. The *user ID* identifies a person signed on to a terminal for security authorization by a security product such as RACF.

User name uniqueness

IMS enforces name uniqueness for user names to ensure data integrity for users. Users are defined to RM during user signon or ISC session initiation. If the user is already active, IMS rejects the signon or session initiation.

For ISC sessions, user name uniqueness can be ignored by using the DFSINTX0 user exit routine. See *IMS Version 15.2 Exit Routines* for more information about this exit routine.

User creation and deletion

When a user becomes active during user signon, IMS defines it to the RM. If the user is active on another system, signon is rejected. IMS also defines users to RM when recoverable commands affecting significant status are processed.

IMS deletes the user from RM when the user becomes inactive and no significant data exists. A user is deleted in three situations:

- During user signoff
- When a command deleting significant status is processed
- During resource cleanup after an IMS failure and no significant status exits

TM resources: user IDs

The *user ID* identifies a person signed on to a terminal for security authorization by a security product such as RACF.

The user ID and the user name are usually the same, but see "TM resources: user names" on page 282 for an example of when they are not the same.

User ID name uniqueness

You can decide whether to enforce name uniqueness for user IDs. The default is to enforce user ID name uniqueness. If you want one user ID to be able to sign on to multiple terminals at a time (name uniqueness not enforced), use the IMS startup parameter SGN=M.

User ID creation and deletion

When a user signs on to a terminal and single-signon enforcement is requested, IMS defines the user ID to RM. Once a user is signed onto a VTAM terminal in the IMSplex, any signon attempt by that user is rejected. When a user ID becomes inactive, IMS deletes it from RM. A user ID is deleted in two situations:

- During user signoff
- During resource cleanup after an IMS failure

How RM and the resource structure impact IMS activities

This topic provides information on how RM and the resource structure impact some IMS activities.

Conversations

Conversation status is shared within the IMSplex, and so the conversation's IDs must be unique for individual dynamic users or static nodes. To uniquely identify a conversation, qualify the conversation with the associated input LTERM.

To display conversation status, use the **`/DISPLAY CONVERSATION`** command for local and RM conversation information.

Initialization

IMS enforces consistency of single signon, SGN=M, in the startup parameters when RM and a resource structure are active. The first IMS active in the IMSplex sets the value for the entire IMSplex. If an IMS joins an IMSplex with a different value for single signon, IMS issues a warning message and uses the IMSplex value. You can change the specification when all the IMS systems leave and then rejoin the IMSplex (warm start). You can override the value by using the **`/NRE`** or **`/ERE`** commands.

During a warm start, when RM is active, the global signon value comes from the startup parameter of the first IMS to join the IMSplex. The signon value comes from the checkpoint log records when RM is inactive during the warm start.

Shutdown

When an IMS system shuts down, all nonrecoverable status owned by the system is deleted from the resource structure. SCI notifies the other IMS systems that the IMS is leaving because of either a normal or abnormal shutdown.

When an IMS system shuts down abnormally, another IMS in the IMSplex becomes the cleanup IMS. This IMS requests a list of resources without significant status. If a resource does have significant status and the resource recovery mode is GLOBAL, the owner is cleared and the resource becomes available to other systems. If the recovery mode is LOCAL, affinity for the failed system is enforced.

Checkpoint

Because terminal and user status are maintained by RM, if active, local DC control blocks must be cleaned up and deleted during IMS checkpoint if they are inactive. Checkpoint deletes local status in local blocks with GLOBAL status recovery mode. Checkpoint maintains status in local blocks with LOCAL status recovery mode.

These recovery modes are treated the same way during a simple checkpoint after a warm or emergency restart. However, if a resource is in LOCAL status recovery mode, DFSSGNX0 and DFSLGNX0 can request that another IMS steal the resource that is owned by the failed IMS system. In that case, the restarting IMS deletes the local status during its restart.

Chapter 19. IMS security

This topic provides information to help you establish resource security for an IMS online system and identifies the resources that can be protected and the facilities available to protect them, provides design considerations, and describes the steps that you need to take to activate security.

Although there are many similarities and overlaps, the security for the DB/DC and DCCTL environments is discussed separately from the security for the DBCTL environment.

Related concepts
"ODBM and security" on page 131
The CSL Open Database Manager (ODBM) does not perform user authentication or authorization itself.

Data set encryption support for IMS

z/OS data set encryption is available on z/OS 2.3 and later and on z/OS 2.2 after APAR OA50569 and dependent APARs are installed.

You can encrypt data sets that are accessed by DFSMS access methods by using z/OS data set encryption. Define the data set as SMS-managed extended format data sets with a key label associated with it.

To create an encrypted data set, you must assign a key label to the data set when it is first allocated, that is, when the data set is created. A key label can be specified through one of the following methods:

- Create SAF rules that associate a key label with a data set name pattern by using the **DATAKEY** parameter of the DFP RACF segment.
- Specify a key label by using JCL, dynamic allocation, or TSO allocate (**DSKEYLBL** parameter).
- Specify a key label on the **IDCAMS DEFINE** command (**KEYLABEL** parameter).
- Use the **DATACLAS** parameter with a key label that is associated with it.

The order in which the methods are listed is the order of precedence. For example, if you have an SAF rule that matches the data set that is being created and you also specify a key label on the **DSKEYLBL** parameter on the JCL DD statement, the SAF key label is used. For more information on data set encryption, see APAR OA50569: z/OS Data Set Encryption.

Existing data sets must be copied into a new extended format data set defined with a key label to become encrypted. Existing data sets do not become encrypted just because their DATACLAS has a key label that is added to it, or a RACF rule associates a key label with the data sets.

Access to the key label is checked by using SAF access rules when a data set is opened. The user ID of the address space where the open operation occurs is checked against the key label CSFKEYS class. The user ID must have READ authority to the resource key label in the CSFKEYS class to be able to access the encryption key for reading from and writing to the encrypted data set.

The following table lists the data sets that support z/OS data set encryption and the IMS address spaces whose user IDs need access to the key labels associated with the data sets.

Note: In the following table, the column "IMS address spaces whose user IDs need access to the key labels" is not exhaustive. Any program or utility that opens one of the data sets in the following table needs access to the key label that is associated with the data set.

Table 26. IMS data sets that support z/OS data set encryption

Data set types	IMS address spaces whose user IDs need access to the key labels
OSAM (linear data sets only)	CTL, DLI, batch jobs, and utilities that access OSAM databases

Table 26. IMS data sets that support z/OS data set encryption (continued)

Data set types	**IMS address spaces whose user IDs need access to the key labels**
VSAM (HALDB, non-HALDB)	CTL, DLI, batch jobs, and utilities that access VSAM DBs
GSAM	IMS BMP and Batch jobs
Online log data sets (DFSOLPnn, DFSOLSnn)	CTL (including XRF alternate, FDBR regions), log archive utility, other utilities that access OLDS, and RSR transport manager
Batch log data sets	IMS batch jobs, utilities that access batch logs, and RSR transport manager
SLDS	CTL, log archive utility, Change Accumulation utility, DB recovery utilities, and other utilities that access SLDS and RSR transport manager
RLDS	Log archive utility, change accumulation utility, and DB recovery utilities
Change Accum data sets	Change accumulation utility and DB recovery utilities
Image copy data sets	Image copy utilities and DB recovery utilities
CQS SRDS	CQS
IMS Connect Recorder Trace	IMS Connect and utilities that process IMS recorder trace
BPE Trace data sets	Address spaces that use BPE, utilities that process BPE trace data (including IPCS TSO users)
Fast Path trace	Dependent region
IMS external trace data sets	CTL and utilities that process IMS external trace
z/OS log stream offload and staging data sets	z/OS logger address space
IMS repository data sets	Repository server
RRDS	CTL and utilities that access the RRDS
RECON data sets	DBRC, IMS batch jobs that use DBRC, and utilities and tools that access the RECON data sets
Monitor data sets	CTL, utilities that process monitor data output
CQS system checkpoint data sets	CQS
Write-ahead data sets (WADS)	CTL (including XRF alternate, FDBR regions), log recovery utility, and other utilities that access WADS
Fast Path DEDB area data sets (ADS)	CTL and utilities that access DEDB ADS

The following data sets cannot be encrypted either because they are accessed by using nonstandard access methods or because DFSMS does not support encryption for them:

- OSAM using sequential data sets (physical OSAM data sets)
- MSDB data sets, including dump, init, and checkpoint
- Queue manager data sets, including LGMSG, SHMSG, and QBLKS

- Restart data sets (RDS)
- All PDS/PDSE type data sets, including PSBLIB, DBDLIB, ACBLIB, MODBLKS, FMTLIB, IMSTFMTx, IMSDALIB, program libraries, PROCLIB or configuration data sets, catalog directory data sets, staging data sets, and BSDS
- Spool data sets

Encrypting batch log data sets

For new batch jobs, define the batch log data sets with a key label using one of the z/OS data set encryption methods. For batch jobs that have completed executing, you can encrypt any logs they created by copying the logs from the original log data sets to encrypted data sets using the IEBGENER utility. You cannot dynamically change a batch log to be encrypted while a batch job is executing.

Encrypting BPE trace data sets

BPE trace records do not usually contain sensitive data. However, since IMS Version 11, the IMS Connect recorder trace (TYPE=RCTR) can use the BPE trace facility, and the recorder trace might contain sensitive data (messages). Thus you may want to encrypt BPE trace data sets if they can be used for the IMS Connect recorder trace.

Use either RACF rules or a DATACLAS with a key label specified to encrypt BPE trace data sets. The BPE EXTTRACE statement in the BPE configuration PROCLIB member does not support the DSKEYLBL parameter.

If you use RACF rules, define a rule that associates the BPE trace data set names with the key label. Once RACF has been updated, newly-created BPE trace data sets will be encrypted.

If you use a DATACLAS with a key label specified, perform the following steps:

1. Update your BPE configuration PROCLIB member. Add or change the **DATACLAS** parameter of the EXTTRACE statement to the data class that has the key label you want.
2. Issue the following command:

   ```
   F jobname,UPD TRTAB NAME(*) OPTION(REREAD)
   ```

   This will re-read the BPE configuration PROCLIB member and update the data class associated with the BPE trace data sets. If BPE external trace is currently active, the current data set is closed and a new one is opened by using the updated EXTTRACE parameters. If BPE external trace is not active, then the new EXTTRACE parameters will be used the next time BPE external trace is used.

Encrypting change accum data sets

You can enable z/OS data set encryption for change accum data sets.

Update DBRC change accum skeletal JCL to specify either a key label via DSKEYLBL= or a DATACLAS with an associated key label for the change accum data sets. Alternately, use RACF to associate a key label with your change accum data sets by data set name pattern. DFSMS does not support SORT work data sets being encrypted.

Encrypting IMS Connect Recorder data sets (Non-BPE trace)

The IMS Connect Recorder data set can only be specified as a JCL-allocated DD (HWSRCORDDD in the IMS Connect startup JCL). To encrypt this data set, create a new data set with a key label, update the IMS Connect JCL to point to the new data set, and then stop and restart IMS Connect.

Encrypting CQS SRDS data sets

Use IDCAMS to define two new encrypted SRDSs the same size as or larger than the current SRDSs.

For the purpose of this example, assume the original SRDSs are named SRDS1 and SRDS2, and the new SRDSs are named SRDS1.NEW and SRDS2.NEW.

If all CQSs are down, perform the following steps to encrypt CQS SRDS data sets:

1. Use IDCAMS REPRO to copy both SRDSs from the old data sets to the new encrypted data sets.
2. If the copies were successful, perform the following steps:
   a. Rename the old SRDSs to another name (such as SRDS1 to SRDS1.OLD and SRDS2 to SRDS2.OLD)
   b. Rename the new SRDSs to be the original SRDS data set names (SRDS1.NEW to SRDS1, SRDS2.NEW to SRDS2). You must rename both the VSAM base data set name and the .DATA data set associated with it (for example, SRDS1 and SRDS1.DATA).
3. Start CQS. Take two structure checkpoints to verify CQS can read and write to the new SRDSs.

If one or more CQSs are active, perform the following steps to encrypt CQS SRDS data sets:

1. Use IDCAMS REPRO to copy one SRDS (for example, SRDS1) from the old data set to the corresponding new encrypted data set.
2. If the copies were successful, perform the following steps:
   a. Rename the SRDS1 to SRDS1.OLD.
   b. Rename the SRDS1.NEW to SRDS1. Note that you must rename both the VSAM base data set name and the .DATA data set associated with it (for example, SRDS1 and SRDS1.DATA).
   c. Take a structure checkpoint to ensure that CQS can access the new encrypted data set. If the new SRDS cannot be opened by CQS, rename the new and old SRDSs back to their original names and take two structure checkpoints.
3. If the structure checkpoint was successful, repeat step1 and 2 with SRDS2. Ensure that no CQS structure checkpoints are taken while you are copying and renaming an SRDS with active CQSs.

Encrypting IMS external trace data set

You can enable z/OS data set encryption for IMS external trace data set.

Normally, IMS external trace data sets contain internal diagnostic trace data and no sensitive customer data. However, if you use the **/DIAG** command to capture internal storage and control blocks from IMS and you specify the external trace data sets as the target output, then it is possible to capture sensitive data in the trace data sets.

If you are using dynamic allocation for the IMS external trace data sets, you can encrypt the external trace data sets while IMS is running. Ensure you are not currently writing to IMS External trace. Then, delete and redefine the external trace data sets with a key label.

If you are using JCL-allocated external trace data sets (DFSTRA01 and DFSTRA02 DD statements in the control region startup JCL), shutdown IMS in order to delete and recreate the data sets as encrypted.

Encrypting GSAM database data sets

Use different ways to encrypt existing and new GSAM database data sets.

If you want to encrypt an existing GSAM database data set, allocate a new encrypted data set and copy the original GSAM data set to the new data set. Then rename and use the new encrypted copy.

If you are creating a new GSAM database data set that will be encrypted, specify it to be encrypted when creating it.

Encrypting image copy data sets

You can enable z/OS data set encryption for image copy data sets.

Update DBRC image copy skeletal JCL to specify either a key label via DSKEYLBL= or a DATACLAS with an associated key label for the image copy data sets. Alternately, use RACF to associate a key label with your image copy data sets by data set name pattern.

Encrypting online log data sets (OLDS)

You can encrypt OLDS with IMS shutdown and restart. If the OLDS are specified in DD statements in the IMS control region JCL, for example, the OLDS are not dynamically allocated, you must shut down and restart IMS across the procedure. To encrypt OLDS without IMS shutdown and restart, either encrypt one data set at a time if you are using only 1 online log data set or more than 50 OLDS, or encrypt a set of OLDS at a time if the number of OLDS is more than 1 but not greater than 50.

Important: If you are running IMS on z/OS 2.4 or earlier and you want to enable encryption for the OLDS, ensure you install z/OS APAR OA60688.

- "Encrypting OLDS with IMS shutdown and restart" on page 289
- "Encrypting one OLDS at a time without IMS shutdown and restart" on page 289
- "Encrypting a set of OLDS (2-50) without IMS shutdown and restart" on page 290

Encrypting OLDS with IMS shutdown and restart

1. Create a set of encrypted OLDS with key labels: one OLDS for each existing OLDS in the set that is used by IMS. The new OLDS must have the same attributes as the current OLDS. They must be extended format data sets.
2. Preformat each new OLDS data sets as described in Formatting newly initialized (reinitialized) volumes for an OLDS (System Definition). One way to preformat is to copy an existing full online log data set into one of the new encrypted OLDS. Then, copy the newly encrypted data set into all other new OLDS. This method ensures that all blocks in the new OLDS are initialized.
3. If IMS is active, issue **`/CHE FREEZE`** and ensure that IMS shuts down normally. Ensure that all log archive jobs complete successfully.
4. Remove the PRIOLD/SECOLD entries from DBRC by using the DSPURX00 utility with the following command issued for every online log data set:

   ```
   DELETE.LOG OLDS(DFSOLPxx) SSID(imsid)
   ```

   Specify the LASTCLOS parameter for the last OLDS that was active when you shut down IMS:

   ```
   DELETE.LOG OLDS(DFSOLPxx) SSID(imsid) LASTCLOS
   ```

   If dual OLDS are being used, this command removes both the PRIOLD and SECOLD entries from DBRC.
5. Rename the original OLDS data sets to a backup name, and rename the new encrypted OLDS to the original OLDS data set names.
6. Warm start IMS (SLDS is used during restart).
7. Delete the old OLDS data sets when you complete the migration and confirm that all is operating correctly.

Encrypting one OLDS at a time without IMS shutdown and restart

This method requires that you use dynamic allocation for the OLDS because the old OLDS are deleted or renamed in the procedure and the encryption fails if the control region has an online log data set allocated.

Perform the following steps to migrate one OLDS at a time without shutdown and restart:

1. Create one or a new set of encrypted OLDS with key labels: one for each existing data set that is used by IMS. The new OLDS must have the same attributes as the current OLDS. They must be extended format data sets.
2. Preformat each new online log data set as described in Formatting newly initialized (reinitialized) volumes for an OLDS (System Definition). One way is to copy an existing full online log data set into one of the newly encrypted OLDS. Then, copy the newly encrypted data set into all other new OLDS. This method ensures that all blocks in the new OLDS are initialized.

3. Wait until a current online log data set fills and IMS switches to the next data set, or issue the **/SWI OLDS** command to force an immediate switch. Then, perform the following steps for the OLDS that you switched from:
   a. Wait until log archiving completes for the switched-from OLDS.
   b. Issue the **/STO OLDS nn** command for the switched-from OLDS.
   c. Delete the OLDS from DBRC by removing the PRIOLD/SECOLD entries by using one the following commands (DSPURX00 utility or online command):

      /DELETE.LOG OLDS(DFSOLPxx) SSID(imsid)LASTCLOS or **/RMD DBRC='LOG OLDS(DFSOLPxx) SSID(imsid) LASTCLOS'**

      Note: If dual OLDS are being used, this command removes both the PRIOLD and SECOLD entries from DBRC.
   d. Rename the old OLDS to another name. Rename both the primary and secondary OLDS that are switched from if you are using dual OLDS logging.
   e. Rename the newly encrypted OLDS data set to the old OLDS data set name. Rename both the primary and secondary OLDS if you are using dual OLDS logging.
   f. Issue the **/STAOLDS nn** command to start using the encrypted OLDS.
4. Repeat step 3 for each OLDS until all OLDS become encrypted.
5. You can delete the old OLDS when you complete the migration and confirm that all is operating correctly.

Encrypting a set of OLDS (2-50) without IMS shutdown and restart

IMS has a maximum of 100 OLDS that can be used and are numbered 0-99. If an IMS subsystem is using OLDS 0-20, then these OLDS are considered the original set. You will define another set of OLDS 21-41 as the extra set. You do not need to have each data set in the extra set already identified in the OLDSDEF statement in the DFSVSMxx IMS.PROCLIB member. Perform the following steps to migrate a set of non-encrypted OLDS to a set of encrypted OLDS while IMS is active:

1. Create a set of extra OLDS with key labels. The new OLDS must have the same block size as the current OLDS. They must be extended format data sets.
2. Preformat each new OLDS data sets as described in Formatting newly initialized (reinitialized) volumes for an OLDS (System Definition). One way is to copy an existing full OLDS into one of the new encrypted OLDS data sets. Then, copy the newly encrypted data set into all other new OLDS. This method ensures that all blocks in the new OLDS are initialized.
3. Create DFSMDA members for each new extra OLDS. If dual OLDS are being used, ensure that MDA members also exist for each secondary online log data set. The syntax for the DFSMDA statement for OLDS is

   DFSMDA TYPE=OLDS,DSNAME=dsname,DDNAME=DFSOLxnn

   Refer to DFSMDA macro (System Definition) for more information about the DFSMDA statement.
4. Issue **/STA OLDS nn** commands for each extra online log data set that was defined.
5. Issue **/SWI OLDS** commands repeatedly until the first one of the newly encrypted extra set of OLDS is in use by IMS.
6. Issue **/STO OLDS nn** commands for each of the original OLDS.
7. Remove the PRIOLD/SECOLD entries in each original online log data set from DBRC by using one of the following commands (DSPURX00 utility or online command):

   /DELETE.LOG OLDS(DFSOLPxx) SSID(imsid) or **/RMD DBRC='LOG OLDS(DFSOLPxx) SSID(imsid)'**.

   Specify the **LASTCLOS** parameter for the last online log data set that you stopped by using one of the following commands:

/DELETE.LOG OLDS(DFSOLPxx) SSID(imsid) LASTCLOS or **/RMD DBRC='LOG OLDS(DFSOLPxx) SSID(imsid) LASTCLOS**.

If dual OLDS are being used, this command removes both the PRIOLD and SECOLD entries from DBRC.

8. Delete or rename and then redefine each original online log data set with a KEYLABEL and pre-format it as in step 2.
9. Issue **/STA OLDS nn** commands for each original online log data set.
10. All the OLDS are encrypted. If you want to remove the extra set of OLDS from being used by IMS, issue **/STO OLDS x** commands for each of the data set. Do not delete them from DBRC.

Encrypting SLDS/RLDS (IMS archived log data sets)

You can enable z/OS data set encryption for SLDS/RLDS (IMS archived log data sets).

Update DBRC log archive skeletal JCL to specify either a key label via DSKEYLBL= or a DATACLAS with an associated key label for the SLDS / RLDS data sets. Alternately, use RACF to associate a key label with your archive log data sets by data set name pattern. Newly-allocated SLDS and RLDS will be encrypted.

Encrypting OSAM sequential data sets

To use z/OS encryption for OSAM data sets, you must convert the OSAM data sets to VSAM linear data sets (LDSs). You can convert full-function OSAM database linear data sets for HALDB, non-HALDB, OLR-capable, or not OLR-capable.

You must have the following prerequisites before you start this process:

- IMS 15.2 with PH16682/UI67505.
- z/OS 2.2 with APAR OA50569 and dependent APARs installed, or z/OS 2.3 and later.
- z196 and Crypto Express 3 or later.
- All OSAM LDS-related APARs are flagged with the IMSOSAMLDS/K fixcat keyword. IBM recommends that you install any APARs with this keyword before using OSAM linear data sets.

IMS OSAM database data sets can be defined as one of two different physical formats:

- Physical sequential data sets that are accessed by the IMS custom I/O driver code
- VSAM linear data sets (OSAM LDS) that are accessed through IBM Media Manager services

The database processing for an OSAM database is the same regardless of which physical format is used. This means that an OSAM database that uses a VSAM LDS as its physical data set is still an OSAM database, not a VSAM database.

OSAM physical sequential data sets cannot be encrypted by using z/OS data set encryption. However, OSAM LDSs can be encrypted using z/OS data set encryption if you specify a key label when the data set is defined.

To use z/OS encryption, you must first convert the OSAM data sets to VSAM linear data sets. Then, you must add a key label parameter to the data set definition.

The data sets must be in EXTENDED FORMAT, and if the data set is to extend beyond the 4 GB limit, it must also be EXTENDED ADDRESSABILITY. A DATACLAS can be used to define these attributes. The data class can also have the KEYLABEL defined to avoid defining the KEYLABEL on every data set. The encryption will be done as the data is copied into data sets with these correct attributes.

This process can be used to migrate OSAM to VSAM data sets whether the data sets are encrypted or not. The encryption can be done at a different time than when the conversion to LDS is done, but it will involve a second migration from an unencrypted LDS data set to an encrypted one.

You can use following conversion methods:

- Convert online by using the Online Reorganization command; this is for OLR-capable HALDB only.
- Convert offline by using the standard reorganization utilities; this can be use for all OSAM databases.

No application changes are required. Any application that accesses an OSAM data set continues to work with the encrypted OSAM data sets that are defined as VSAM LDS. Data that is stored in encrypted OSAM data sets is processed the same way as non-VSAM OSAM data sets.

Restrictions:

VSAM linear data sets require a CI size that is a multiple of 4096 and from a minimum of 4096 to a maximum of 32786 bytes. If you use DFSUDMP0, you must choose a smaller block size as the utility adds a 15 byte prefix to each image copy output record which cannot exceed 32767.

Any OSAM physical sequential data sets that use a block size smaller than 4096 bytes must be changed to a CI size of at least 4096 bytes when converted to OSAM LDS. This can affect current buffer pool definitions and randomization parameters for HDAM databases. Check that the **rbn** parameter times the new CI size does not exceed the OSAM data set maximum of 8 GB, (or 4 GB for OLR-capable PHDAM HALDBs).

IMS OSAM data sets that are not defined as VSAM LDSs, such as OSAM databases using sequential data sets, Queue Manager (QMGR) data sets, and recovery data sets (RDS), cannot be encrypted by using z/OS data set encryption.

Encrypting full-function OSAM database linear data sets (HALDB, OLR-capable)

Use HALDB Online Reorganization (OLR) to reorganize unencrypted OSAM sequential data sets in OLR-capable HALDBs into unencrypted VSAM linear data sets. Converting to VSAM linear data sets is required if you want to use z/OS encryption.

To reorganize unencrypted OSAM data sets into unencrypted VSAM LDSs:

1. Preallocate the new OLR target database data sets as VSAM linear data sets:

   ```
   DEFINE CLUSTER NAME(my.osam.dataset) LINEAR DATACLAS(dclasname)....
   ```

   - If the output data sets are not preallocated as LDSs, OLR will create target data sets based on the attributes of the source data sets.
   - Minimum LDS CI size is 4096. Check the DBD randomization parameters for HDAM or PHDAM and ensure **rbn** times CI size does not exceed 4 GB for OLR-capable HALDBs or 8 GB for either non-HALDBs or non-OLR-capable HALDBs.
2. Run IMS online reorganization for one or more partitions.
3. Repeat until all partitions of the HALDB are reorganized.
4. Preallocate the output data sets to use a key label. See "Data set encryption support for IMS" on page 285 for more information.

Encrypting full-function OSAM database linear data sets (non-HALDB, or not OLR-capable)

You can use this procedure to convert OSAM data sets to VSAM LDSs offline for all OSAM databases.

To convert OSAM data sets to VSAM LDSs offline:

1. Preallocate the new database data sets as VSAM linear data sets.
2. Take the database offline, for example: `/DBR DB, UPDATE DB STOP(ACCESS)`
3. Run the HD reorganization unload utility (DFSURGU0).
4. Run the HD reorganization reload utility (DFSURGL0) to reload to the LDS data sets.
5. Rename the old (original) data sets to a backup name.
6. Rename the new data sets to the correct database data set names.
7. Bring the database online, for example: `/STA DB, UPDATE DB START(ACCESS)`
8. Preallocate the output data sets to use a key label. See "Data set encryption support for IMS" on page 285 for more information.

Encrypting full function VSAM database data sets (non-HALDB, or not OLR-capable)

You can enable z/OS data set encryption for full function VSAM database data sets (non-HALDB, or not OLR-capable).

Besides the following steps, you may also be able to use a separate online reorganization product to perform a database reorganization without taking the database offline. Consult your online reorganization tool documentation for details.

1. Preallocate the new database data sets with key labels.
2. Take the database offline (e.g. **`/DBR DB, UPDATE DB STOP(ACCESS)`**).
3. Run the appropriate unload utility (DFSURUL0 or DFSURGU0).
4. **Note:** Note that if prefix resolution is required, the sort that is done will create temporary work data sets that are not encrypted. DFSMS does not support sort work data sets as extended format.

   Run the appropriate reload utility (DFSURRL0 or DFSURGL0) to reload the database into the encrypted data sets.
5. Rename the old data sets to a backup name.
6. Rename the new data sets to the correct database data set names.
7. Bring the database online (e.g. **`/STA DB, UPDATE DB START(ACCESS)`**).

Encrypting full function VSAM database data sets (HALDB, OLR capable)

Use HALDB Online Reorganization to reorganize the unencrypted data set into an encrypted data set. You can also use the offline process for non-HALDB VSAM database data sets.

1. Preallocate the new OLR target database data sets with key labels.
2. Run IMS online reorganization for one or more partitions. Repeat until all partitions are reorganized.

Refer to Encrypting full function VSAM database data sets (non-HALDB, or not OLR-capable) (System Administration) for how to use the offline process to enable z/OS data set encryption.

Encrypting z/OS log stream offload and staging data sets

Use RACF rules to associate your logger data sets with a key label or add a key label to the DATACLAS used by your logger data sets. The encryption will take effect the next time a new offload or staging data set is created by the z/OS logger.

Encrypting the write-ahead data sets (WADS)

To encrypt WADS, define WADS as extended format and associate a key label with it. For XRF and FDBR systems, APAR PI84947 must be applied to both systems before enabling WADS encryption.

You can encrypt WADS while IMS remains up or across a restart. IMS can use up to 10 WADS, which are numbered 0 - 9.

When you are migrating from non-encrypted WADS to encrypted WADS, recycle IMS before this migration if the following conditions are met:

- Different data set names are used for encrypted and non-encrypted WADS.
- Dynamic allocation (DFSMDA member) is not used for the WADS data set names. That is, there are DD statements in the control region JCL for the WADS.

This is because the online control region will have the WADS allocated and you will be unable to delete the WADS while IMS remains up.

Migration procedure across a restart

1. Ensure that IMS comes down cleanly.

2. Delete all WADS.
3. Define all WADS as extended format and with key labels specified.
4. Optional: If you use different data set names for the encrypted WADS, update the MDA members or change the DD statements in the control region JCL.
5. Cold start IMS by issuing the **/NRESTART** command with FMT WA or FMT ALL specified.

Migration procedure when IMS remains up

1. Issue the **/DISPLAY OLDS** command to see which WADS are in use.
2. Issue the **/STOP WADS** command for the WADS that are not in use.
3. Delete all of the WADS that are stopped and define them as extended format and with key labels specified.
4. Optional: If you use different data set names for the encrypted WADS, update the MDA members.
5. Issue the **/START WADS** command for each WADS that was deleted and defined. WADS are formatted as part of the **/START WADS** command processing.
6. Issue the **/SWITCH WADS** command.
7. Issue the **/STOP WADS** command for the WADS that are switched from. If you use dual WADS, issue this command for each WADS that is switched from.
8. Delete the WADS that are switched from and define them as extended format and with key labels specified.
9. Optional: If you use different data set names for the encrypted WADS, update the MDA members.
10. Issue the **/START WADS** command for the WADS that were deleted and defined.

Encrypting Fast Path DEDB area data sets (ADS)

To encrypt Fast Path DEDB area data sets (ADS), define the data sets as extended format and specify key labels with the data sets.

DEDB ADS encryption requires Fast Path 64-bit buffers. This is specified by FPBP64=Y in the FASTPATH section of the DFSDFxxx member in the IMS PROCLIB data set. DEDB ADS encryption is not supported when you are using 31-bit Fast Path buffers.

Make sure all the IMS systems that share encrypted Fast Path ADS meet the following requirements:

- All the IMS systems are IMS 14 or later with APAR PI83756 installed.
- All the IMS systems are using Fast Path 64-bit buffers.
- All the z/OS systems are z/OS 2.2 with OA50569 installed, or z/OS 2.3 and later.

If an encrypted Fast Path ADS is shared by an IMS system that is IMS 14 or earlier versions, or IMS 15 without APAR PI83756, this IMS system will not be able to open or access the ADS. Message IEC161I with return code 122 will be displayed.

To migrate from non-encrypted Fast Path DEDB ADS to encrypted Fast Path DEDB ADS, perform the following steps:

1. Perform one of the following methods to create encrypted ADS:
   - Define the shadow ADS with the same attributes as the non-encrypted ADS and specify key labels with the shadow ADS. Run DEDB ALTER utility.
   - Define new ADS as SMS-managed extended format DASD and specify key labels with them. Register the new ADS to DBRC by using the following command:

     INIT.ADS DBD(*xxxxxxxx*) AREA(*yyyyyyyy*) ADSN(*AREA data set name*) UNAVAIL

     Run CREATE utility.
2. Stop the non-encrypted ADS if necessary.

To fallback from encrypted Fast Path DEDB ADS to non-encrypted ADS, create new ADS without key label and use the DEDB ALTER and DEDB Area Data Set Create utilities to copy encrypted ADS to new ones.

Remember: If you use non-IMS tools or products to process the Fast Path ADS, confirm with the tool or product provider that the tool or product supports encrypted DEDBs before enabling encryption.

Additional Fast Path buffers are used when writing to an encrypted area data set. A temporary buffer is obtained at write I/O time to encrypt the data to be written. This buffer is released after the data has been written to the ADS. Each Fast Path buffer includes the buffer itself (64-bit storage) and a 31-bit ECSA control block called a DMHR, which is used to track the buffer. Therefore, encrypting ADS may increase Fast Path 64-bit buffer and ECSA usage. Use the **FPBP64M=** parameter in the FASTPATH section of the DFSDFxxx member to increase the limit for the amount of 64-bit Fast Path buffers to avoid buffer shortages if necessary. Monitor the 64-bit buffer usage statistics in the IMS 4516 statistics log records when migrating to encrypted ADS.

You must define all data sets that are encrypted by using z/OS data set encryption, including Fast Path DEDB ADS, as SMS-managed extended format data sets. Do not define Fast Path DEDB ADS with the extended addressability attribute. Ensure that the DATACLAS parameter that you use to allocate your ADS does not include the extended addressability attribute.

Related reference

FASTPATH section of the DFSDFxxx member (System Definition)
DEDB Alter utility (DBFUDA00) (Database Utilities)
DEDB Area Data Set Create utility (DBFUMRI0) (Database Utilities)
IEC161I (return code 122)

Overview of DB/DC and DCCTL security

When you initiate security safeguards, you must balance requirements between those responsible for the security of resources and those users who legitimately need access to those resources.

Because the person who is assigned to resource security is held responsible for resources that might be compromised, that person should not allow easy access to dominate protection measures. However, users performing their assigned tasks need convenient access to the resources. The users and the security specialist should work out a balanced approach between the ease of resource access and the complexity of protecting that resource.

In an IMS system, two types of security exist. You can address one or both types:

- Securing the kind of resource to which a user has access. For example, a user might be allowed access to the Part database but not to the Customer Order database.
- Securing what the user can do to the resource after that user has access to it. For example, a user might be allowed to read a file but not to update it.

DB/DC and DCCTL resources that can be protected

Before you decide what security facilities to use in designing a secure IMS system, you should know which resources within the system need protection.

In other words, you should decide what to protect before you decide how to protect it.

The following resources can be protected in the DB/DC and DCCTL environments:

- IMS control region
- IMS online system
- IMS system data sets
- Transactions
- Commands
- Program specification blocks (PSBs)
- Online application programs

- Databases
- Dependent regions
  - BMP (batch message processing) region
  - IFP (IMS Fast Path) region
  - JBP (Java batch processing) region
  - JMP (Java message processing) region
  - MPP (message processing program) region
- Terminals
  - Logical terminals (LTERMs)
  - Physical terminals (PTERMs)
  - Master terminals

Defining security during DB/DC and DCCTL system definition

You can make IMS security choices with initialization EXEC parameters and in two system definition macros: COMM, and IMSGEN. These specifications let you choose the type of security that is active during online execution.

You can make other security choices, including which resources you want to protect, by using the Resource Access Control Facility (RACF). You can also specify security choices using the security parameters in the IMS and DCC startup procedures.

If you do not specify any security in any of the three system definition macros, IMS provides a basic level of resource security called *default security,* which:

- Prohibits the entry of certain commands from any terminal other than the master terminal. This basic security function is activated upon completion of stage 2 of IMS system definition. When you implement input-access security with RACF, IMS removes the default security restrictions. In the case of static terminals if sign-on is not required, IMS uses the control region user ID for command validation.
- Applies only to statically defined terminals. Terminals that are defined by using ETO are automatically governed by an identical level of default security. When you modify and use the Command Authorization exit routine (DFSCCMD0), IMS removes the default security for dynamically defined terminals.

Security specifications can be made in the COMM macro, IMSGEN macro, or with initialization EXEC parameters. The security-related specifications are accepted hierarchically in the following order:

1. The initialization EXEC parameters specified in the DFSPBxxx PROCLIB member
2. The initialization EXEC parameters specified in the DFSDCxxx PROCLIB member
3. COMM
4. IMSGEN

Specifications that are coded in the initialization EXEC parameters override security specifications that are coded in the COMM and IMSGEN macros.

Security facilities for DB/DC and DCCTL resources

The following table summarizes the resources you can protect, the types of security that you can use to protect the resources, and the security facilities that you can use to implement each type of security.

When choosing which security facilities to use, also consider your installation's security standards and operating procedures.

Table 27. DB/DC and DCCTL resources and the facilities to protect them

Resource	Type of security	Security facility
IMS control region and online system	Extended resource protection (using APPL resource class)	RACF
System data set	z/OS password protection	z/OS
	Data set protection (VSAM)	RACF
Database	Segment sensitivity	PSBGEN RACF
	Field sensitivity	PSBGEN RACF
	Password security (for the **/LOCK** and **/UNLOCK** commands)	RACF
PTERM	Signon verification security	RACF with exit routine
	Terminal-user security	RACF
	Password security (for the **/LOCK** and **/UNLOCK** commands)	RACF
LTERM	Password security (for the **/LOCK** and **/UNLOCK** commands)	RACF
	Resource Access Security	RACF
Terminal defined with ETO	Signon verification security	RACF and exit routine
	Input access security	RACF and exit routine
LU 6.2 inbound and IMS-managed outbound conversations	Allocate verification security	RACF and exit routine
	Input access security	RACF and exit routine
PSB	Resource Access Security	RACF
	APSB SAF security	RACF"1" on page 298
Transaction	Input access security	RACF
	Output access security	System definition
	Resource Access Security	RACF
	Password security (for the **/LOCK** and **/UNLOCK** commands)	RACF
Command	Default security	System definition
	Transaction command security for automated operator (AO) commands	RACF or Command Authorization exit routine
	Input access security	RACF
	type-2 command security	RACF
	DBRC command authorization"2" on page 298	RACF or exit routine
Type-1 Automated Operator Interface applications	Transaction command security	RACF or Command Authorization exit routine

Table 27. DB/DC and DCCTL resources and the facilities to protect them (continued)

Resource	Type of security	Security facility
Type-2 Automated Operator Interface applications	Transaction command security	RACF or Command Authorization exit routine
Online application program	Password security (for the **/LOCK** and **/UNLOCK** commands)	RACF
	Extended resource protection (using APPL keyword)	RACF
Dependent region	APSB SAF security	RACF
	Resource Access Security	RACF

Notes:

1. Using RACF to secure APSBs applies only to CPI-C driven applications and ODBA applications.
2. DBRC command authorization is an additional command security option that applies only to DBRC commands. DBRC commands are also subject to any other command security options that are active in the IMS system.

Designing security for IMS DB/DC and DCCTL

This topic explains how the various types of IMS security can be used. When you are considering each part of your security design, consider the physical actions that a user must take to obtain access to the system. You probably will use more than one type of security checking.

This topic assumes the following control points for security:

- User identifications
- The master terminal
- Automated operator programs
- The use of regions

This section also assumes that you will use the Resource Access Control Facility (RACF) security product, which is licensed with the z/OS operating system. RACF is external to IMS.

Limiting access from a terminal

To control access for a terminal control point, have the end users identify themselves each time they use a terminal and authorize the set of transactions and commands each user needs.

For a static terminal, or a dynamic terminal that has the SPQBname the same as the node name, a user will not be allowed to signon unless all conversations are held, or the user is authorized to use the transaction for the active conversation.

If there is an active conversation for a static terminal, and the user is not authorized to use its transaction, the user can enter a **/HOLD** command prior to signing on to put all of the conversations in a held state. The user will then be allowed to sign on.

If there is an active conversation for a dynamic terminal that has the SPQBname the same as the node name, only a user that is authorized to use the transaction of the active conversation will be allowed to sign on. The **/HOLD** command is not allowed prior to signing on for a dynamic terminal.

Controlling access by signon verification

Signon verification security requires the entry of a user ID as a parameter on a **/SIGN** command at all terminals or a subset of the terminals.

To implement signon verification for statically defined VTAM terminals, set the OPTIONS= keyword to SIGNON in the TYPE or TERMINAL system definition macro. The specifications in the TERMINAL macro override those set in the TYPE macro. The default for OPTIONS= in the TYPE macro is NOSIGNON.

To avoid setting the OPTIONS= keyword in multiple system definition macros when you want all of your static terminals to sign on, specify SIGNON=ALL in the DFSDCxxx PROCLIB member. This requires all terminals to sign on except MTO, LU6.1, 3284/3286, and SLU1 printer-only devices.

Signon verification is done by RACF, an exit routine, or both. If you are using RACF, RACF checks for user ID, password or password phrase, and group. If you are using an exit routine to implement signon verification, the exit routine can check for user ID and password.

To enable password verification:

- Set the password verification byte on the Initialization exit routine (DFSINTX0)
- Reply to the DFS3656 message
- Issue the **/SIGN ON VERIFY** command

The exit routines that you might use to verify user IDs and passwords include the Sign On/Off Security exit routine (DFSCSGN0). For ETO terminals, you might also use the Sign-On exit routine (DFSSGNX0). DFSCSGN0 is described in more detail in "Preparing security exit routines" on page 321.

Recommendation: If you specify mixed-case password support in your IMS system definition (the PSWDC= parameter on the DBC, DCC, or IMS procedures), make sure that any exit routines that manipulate passwords can also support mixed-case passwords.

To display or process informational messages that are sent by RACF, such as ICH70001I and ICH70002I, when sign-on is successful, you can specify **RACFMSG=Y** in the DFSDC*xxx* PROCLIB member. When **RACFMSG=Y** and sign-on is successful, IMS passes RACF sign-on messages to the Greeting Messages exit routine (DFSGMSG0), where you can provide code that processes them. The messages are returned by IMS in the WTO format.

For more information about RACF sign-on messages, see z/OS: RACF miscellaneous messages.

Related reference

/SIGN command (Commands)

Macros used in IMS environments (System Definition)

DFSDCxxx member of the IMS PROCLIB data set (System Definition)

Transaction Manager exit routines (Exit Routines)

Using RACF PassTickets

A RACF PassTicket is a one-time-only password that is generated by a requesting product or function. The PassTicket is an alternative to the RACF password and removes the need to send RACF passwords across the network in clear text.

PassTickets also make it possible to move the authentication of a mainframe application user ID from RACF to one of these alternatives:

- Another authorized function executing on the host system
- The workstation local area network (LAN) environment

PassTickets are resistant to reuse because they only give one user access to a specific application for approximately 10 minutes. For most applications, once a particular PassTicket is used, the same user cannot use it again for the same application during the same 10-minute interval.

When a**/SIGN ON** command is received by IMS that contains a PassTicket instead of a password, the signon process fails, unless the PassTicket was created using the IMSID as the application name (name of the PTKTDATA profile). Because the IMSID may not be known to other systems that might enter the signon command, flexibility is provided to allow for the use of other names as application names when creating PassTickets. The keyword APPL in the **/SIGN ON** command allows the end-user or program to specify another name (such as the IMS VTAM application name) rather than the IMSID when creating the PassTicket.

In a VTAM generic resource (VGR) environment, the remote end-user does not know which IMS will be chosen for the connection. The DFSDCxxx PROCLIB member provides a system-wide default application name (replacing IMSID) for all the IMS systems in a generic group. With the use of this replacement name for the application name, the creator of the PassTicket does not have to know which IMS system will be the recipient of the IMS terminal session.

Related reading:

- For more information about PassTickets, see *z/OS Security Server RACF Macros and Interfaces*.
- For more information on enabling the use of PassTickets, see *z/OS Security Server RACF Security Administrator's Guide*.

Creating and using a RACF PassTicket

When you call RACF, a general PassTicket generator algorithm uses special input information to create a unique PassTicket. The PassTicket is an 8-character alphanumeric string that contains the characters A through Z and 0 through 9.

The algorithm generates a PassTicket using the following four input values:

- A RACF host user ID
- The PTKTDATA class profile name (also known as the application id)
- The RACF Secured Signon application key, which is contained in the PTKTDATA profile
- The required time and date input data from the application that is providing the logon function

  Requirement: For RACF to properly evaluate PassTickets, the TOD clock must be set to GMT rather than local time.

For Java Database Connectivity (JDBC) applications that connect to IMS DB by running the SQL Batch utility, the utility generates a PassTicket if all of the following conditions are met:

- Both the `IRRRacf.jar` and `ibmjzos.jar` files are in the job's class path.
- In the **applName** URL property of the `DriverManager.getConnection` method, the 1- to 8-character application name that is defined to RACF in the PTKTDATA class for DRDA client access to IMS DB is specified.
- The following values are the same as each other:
  - The value of the **applName** URL property of the `DriverManager.getConnection` method.
  - The value of the **APPL=** parameter of the ODACCESS statement, which is in the HWSCFG*xx* member of the IMS PROCLIB data set.
- On the JOB statement of the JCL for the SQL Batch utility, the z/OS user ID that is associated with the job is specified.

Related tasks

RACF PassTicket for IMS Connect client connections to IMS DB (Communications and Connections)

Related reference

SQL Batch utility (Database Utilities)

Authorizing transactions and commands

This topic describes transactions and commands, and explains how to authorize them.

Authorizing transactions

Transaction authorization determines if a user ID is permitted to use a certain transaction. You can use an exit routine, RACF, or both to perform the transaction authorization. To gradually phase in RACF as your installation's transaction authorization method, use the Transaction Authorization exit routine (DFSCTRN0).

This routine can reject the transaction if the transaction is entered by an ETO terminal but is not protected by RACF. If you use both DFSCTRN0 and RACF, DFSCTRN0 is effective only after RACF has authorized the transaction or when the transaction is not defined to RACF. DFSCTRN0 is described under "Preparing security exit routines" on page 321.

If you specify the REVERIFY option to RACF, the user must reenter the signon password with each transaction code. REVERIFY is not supported when a takeover occurs in an XRF complex. Users might need to sign on again after a takeover if they are not on a class-1 terminal.

With program-to-program switching through the DL/I change (CHNG) call or by changing the transaction code in the SPA, you can use RACF, an exit routine, or both, to check transaction authorization. The same applies when the transaction code status is changed by the **/SET**, **/LOCK**, and **/UNLOCK** commands. In addition, as the application program associated with the transaction produces database changes, the user ID is logged with the change records on the IMS system log to identify the changes performed by a specific user.

IMS provides the RVFY= parameter in the IMS procedure for customers who want to force reverification that the operator who signed on to a terminal is the same operator who is now entering a command or transaction. This reverification is done with RACF by including the word 'REVERIFY' in the APPLDATA field of the command or transaction profile. For example:

```
RDEFINE Txxx tran-name UACC(NONE) APPLDATA('REVERIFY')
```

Each time the user enters this transaction code, the RACF password must be entered where an IMS password would be entered if the transaction were password protected.

Authorizing commands

When the Command Authorization exit routine (DFSCCMD0) is included in IMS.SDFSRESL, it provides a command authorization check. The Command Authorization exit routine can work in conjunction with RACF, or it can work independently, without RACF .

The Command Authorization exit routine is called for each IMS command. Default security or RACF is called first to perform the authorization. The return code is passed to the Command Authorization exit routine. DFSCCMD0 performs a final verification and determines the success or failure of the command authorization. DFSCCMD0 can also be used alone to perform the verification. If you want to establish a command authorization level more discrete than that provided by RACF , you can examine DFSCCMD0's input command buffer. DFSCCMD0's input command buffer contains the complete command stream.

If DFSCCMD0 exists in IMS.SDFSRESL, DFSCCMD0 is called for all device types including static, ETO, and LU 6.2.

IMS provides the RVFY= parameter in the IMS procedure for customers who want to force reverification that the operator who signed on to a terminal is the same operator who is now entering a command or transaction. This reverification is done with RACF by including the word 'REVERIFY' in the APPLDATA field of the command or transaction profile. For example:

```
RDEFINE Cxxx tran-name UACC(NONE) APPLDATA('REVERIFY')
```

Each time the user enters this transaction code, the RACF password must be entered where an IMS password would be entered if the transaction were password protected.

Password protecting the /LOCK, /UNLOCK, and /SET commands

For the **/LOCK**, **/UNLOCK**, and **/SET** commands you can implement password security to protect LTERMs, databases, programs, and transactions.

Perform the following tasks to check the password and enforce password security using RACF or an exit routine:

1. Define the resource to be protected by using the appropriate RACF security class: LIMS, PIMS, IIMS, or TIMS.
2. Define the user IDs that you want to have access to the resource in the RACF security class that defines the resource. The user IDs must be the same user IDs that are used to sign on to the terminal from which the **/LOCK**, **/UNLOCK**, or **/SET** command is issued.
3. In the DFSDCxxx PROCLIB member, specify LOCKSEC=Y.
4. In a DBCTL only environment, the **/LOCK** command cannot be secured with password protection.

Using RACF to protect physical terminals

RACF offers a terminal-user security function that ranges from no security for a particular terminal to permitting a certain predefined list of users access through a physical control point. A terminal-user profile can be created for every PTERM in the IMS.

The following table is an example of a terminal-user profile.

Table 28. Example terminal-user profile

	Physical terminal				
User	**PTERMA**	**PTERMB**	**PTERMC**	**LTERMD**	**LTERME**
USER 1	X				X
USER 2	X	X			
USER 3				X	X
USER 4					X
USER 5		X			X

Users defined through ETO can use signon verification to gain access to IMS transactions or commands. Dynamic terminal security must be defined through a security product such as RACF or through exit routines.

Signon password with RACF

If RACF is used to implement signon verification security for a terminal, a check of the password is made with entry of the **/SIGN** command, by either the Signon Verification exit routine or by RACF. The user ID entered with the **/SIGN ON** command must be accompanied by a user password and, optionally, by other signon data. If the RACF reverification option has been specified, the password is saved so it can be compared with the password reentered with the transaction code.

Signon passphrase with RACF

If RACF is used to implement signon verification security for a terminal, a check of the passphrase is made with entry of the **/SIGN** command, by either the Signon Verification exit routine or by RACF. The user ID entered with the **/SIGN PASSPHRASE** or **/SIGN PASSPHRASEQ**command must be accompanied by a user passphrase and, optionally, by other signon data. The RACF reverification option is not supported with passphrases.

RACF password protection

RACF passwords are defined and maintained by the user. After the RACF resource class is initialized with a password, the user can change its value. If signon verification is provided by an exit routine and not by RACF, the table of user IDs and passwords must be changed by another bind into the IMS nucleus. However, for ETO signon verification, the table of user IDs and passwords does not need to be changed by another bind into the IMS nucleus. If the table is loaded by exit routine DFSINTX0 or if it is part of exit routine DFSSGNX0, IMS does not need to be restarted in order to have the table refreshed.

Related reference

/SIGN command (Commands)

Propagating network security credentials

You can enable the network security credentials that are entered by a user in a distributed environment to be associated with the end-to-end processing of an IMS transaction. You can also enable an application in a distributed environment to support network security credentials that are passed in synchronous callout requests initiated by the ICAL call of the IMS DL/I interface.

The distributed network security credentials can include a network user ID and a network session ID.

Network user ID
: The distributed identity of the user. The maximum length of a network user ID is 246 bytes. For users of the IMS TM Resource Adapter, the network user ID is a Distinguish Name (DN) in the X.500 series of standards.

Network session ID
: The session identity of the distributed user. The maximum length of a network session ID is 254 bytes. For users of the IMS TM Resource Adapter, the network session ID is a domain name, realm, or registry name.

You can enable network security credentials to be passed to IMS from applications that use one of the following user message exits:

- HWSSMPL0
- HWSSMPL1
- HWSJAVA0

Restriction: Distributed network security credentials from DataPower®, IMS Connect API, and SOAP Gateway clients are not supported by IMS Connect.

If an input message to OTMA contains network security credentials, the credentials can be propagated by IMS in synchronous callout requests that are initiated by the ICAL call. You can enable applications in a distributed environment that issue a RESUME TPIPE call to support the network security credentials in the IMS callout requests.

You can use the `otma_send_receivey` and `otma_send_asyncx` APIs of the IMS OTMA Callable Interface (OTMA C/I) to pass the network user ID and the network session ID to IMS. For each API, up to 100 bytes for the network user ID and up to 100 bytes for the network session ID can be passed to IMS.

You can also use the Transaction Authorization exit routine (DFSCTRN0) to pass the addresses of the network security credentials in the OTMA message prefix.

You can use the following OTMA user exit routines, which include the address of the security-data section of the OTMA message prefix, to access the network security credentials that are in the OTMA input message if the credentials are passed to IMS:

- DFSYIOE0
- DFSYPRX0
- DFSYDRU0

Because distributed network security credentials are passed to IMS in the security-data section of the OTMA message prefix, all IMS log records that contain information about the message prefix, such as log records X'01' and X'03', include the distributed security credentials.

If a Fast Path message contains network security credentials and is processed by the Fast Path expedited message handler (EMH) on the local IMS system, the credentials are logged in the X'5901' log record.

If a Fast Path message that contains network security credentials is processed by using the EMH queue (EMHQ) in a shared-queues environment, in the front-end IMS system, the credentials are included in the X'5911' log record. In the back-end IMS system, which is the processing IMS system, the credentials are included in the X'5901' log record.

Propagating network security credentials for user-written IMS Connect client applications

You can enable network security credentials to be passed to IMS from a user-written IMS Connect client application that uses the HWSSMPL0 or the HWSSMPL1 user message exit routine. After the network security credentials are passed to IMS, the credentials are audited in IMS log records and associated with the IMS transaction.

If network security credentials are passed in the inbound message to IMS, the credentials can also be passed by IMS in a synchronous callout message if the callout request is initiated by an ICAL call.

If you require your user-written IMS Connect client application to support network security credentials when IMS issues a callout request and your application issues a RESUME TPIPE call, your must define the RESUME TPIPE call to support the credentials.

Tip: If you pass network security credentials to IMS from an application that uses the HWSSMPL0 or the HWSSMPL1 user message exit routine, consider enabling the BPE External Trace facility for the IMS Connect Recorder Trace facility. When network security credentials from user-written IMS Connect client applications are passed to IMS, the size of both input and output messages to and from IMS Connect might be larger than 670 bytes and the BPE External Trace facility would be required to capture the data of the entire message.

1. Use IRM extensions to pass network security credentials to IMS Connect from an application that uses the HWSSMPL0 or the HWSSMPL1 user message exit routine.

   Use an IRM extension with an ID of *NETUID* to pass a network user ID, which can be 1 - 246 bytes in length. Use an IRM extension with an ID of *NETSID* to pass a network session ID, which can be 1 - 256 bytes in length.

   After the network security credentials are passed to IMS Connect, the HWSSMPL0 or the HWSSMPL1 user message exit routine builds the security-data section of the OTMA message prefix to include the credentials.
2. To generate the DSECTs, HWSECDNDS and HWSECARDS, for the network security information that are included in the OTMA message prefix, specify both of the following options in the HWSOMPFX macro:

   DSECT=
   Generates an individual DSECT for each section of the OTMA header. However, the HWSECDNDS and HWSECARDS DSECTS are not generated.

   NETSEC_OPT=YES
   Generates the HWSECDNDS and HWSECARDS DSECTs if you also specify the DSECT= option.
3. Modify code that uses the HWSOMPFX macro to map the OTMA header. If network security credentials are included in the security-data section, the size of the security section might vary and cause the locations of the fields that are below the security section to also change and become inaccessible. To inspect or modify any section of the OTMA header so that the data in the header can be accessed, do the following steps:

   a) Inspect the values of the OMCTLPFL field to determine whether the state data, security data, user data, and application data sections exist in the message.

b) To access a specific section in the OTMA header, use the values of the OMCTLLEN, OMHDRLEN, OMSECLEN, and OMUSRLEN fields, if present, to skip over the message's control data, state data, security data, and user data sections, respectively.

4. To enable a user-written IMS Connect client application that uses the HWSSMPL0 or the HWSSMPL1 user message exit routine to support network security credentials in IMS callout requests, define the RESUME TPIPE call with the following field specifications in the IRM prefix. If the following field specifications are not defined, IMS removes the distributed network security credentials from the security-data section of the OTMA message prefix in the callout request.

   IRM_ARCH
   X'05' (IRM_ARCH5)

   IRM_F6
   X'80' (IRM_F6_NWSE)

Related concepts
RESUME TPIPE/receive protocol (Communications and Connections)
Related tasks
Retrieving synchronous callout requests with RESUME TPIPE (Communications and Connections)
Related reference
Format of IRM extensions (Communications and Connections)
Input message from client and passed to message exit (Communications and Connections)
Input message returned from message exit (Communications and Connections)
Output structure from client exit (Communications and Connections)
OTMA state data fields used by IMS Connect (Communications and Connections)
OTMA security data fields used by IMS Connect (Communications and Connections)
Explanation of OTMA security data fields (Communications and Connections)
Output message from message exit to client (Communications and Connections)
Format of the returned network security segments (Communications and Connections)

Propagating network security credentials for IMS TM resource adapter client applications

You can enable network security credentials to be passed to IMS from an IMS TM resource adapter client application that uses the HWSJAVA0 user message exit routine. After the network security credentials are sent from IMS TM resource adapter to IMS, the credentials are audited in IMS log records and associated with the IMS transaction.

Ensure that the following prerequisites are enabled for your IMS TM resource adapter client application:

- One of the following application servers:
  - WebSphere Application Server Version 8.0 or later
  - WebSphere Liberty Version 8.5.5.9 or later
- Container-managed security.
- An external user account registry, such as an LDAP server, that contains authorized users.

You can also enable IMS TM resource adapter to support network security credentials when IMS applications that run in IMS dependent regions make synchronous or asynchronous callout requests to external Java EE applications.

The network session ID that is extracted by the IMS TM resource adapter is, by default, the IP address and port of the authentication server that is used. The maximum length for a network session ID is 254 bytes.

The network user ID that is extracted by the IMS TM resource adapter is one of the distinguished names that are defined in the authentication server. The maximum length for a network user ID is 246 bytes.

The following procedure describes the high-level steps that you must perform to enable network security credentials to be passed from your IMS TM resource adapter client application to IMS:

1. Install and configure the Java Authentication and Authorization Service (JAAS) login module that is provided with IMS TM resource adapter. The JAAS login module can be installed on WebSphere Application Server or WebSphere Liberty.
2. Link the JAAS login module to your Java application.

   After you link your application to the JAAS login module, users must enter their security credentials when they invoke an IMS transaction for authentication by an external user account registry. The external user account registry can be any user account registry that is supported by WebSphere Application Server or WebSphere Liberty such as an LDAP server. After the credentials are successfully authenticated, IMS TM resource adapter sends the distributed credentials to IMS Connect by using the security-data section of the OTMA message prefix.
3. If you require your IMS TM resource adapter client application to receive network security credentials that are sent from IMS in IMS callout messages, enable the application to support the credentials.

   - To enable IMS TM resource adapter client applications to support network security credentials in synchronous callout messages, set the resumeTpipeNsc property of the `IMSActivationSpec` object to true.
   - To enable IMS TM resource adapter client applications to support network security credentials in asynchronous callout messages, you must call the `setResumeTpipeNSC(int resumeTpipeNSC)` method for the `IMSInteractionSpec` object and set the value of the `setResumeTpipeNSC` property to `1`. If `1` is set for the `setResumeTpipeNSC` property, IMS TM resource adapter sets a flag byte in the OTMA message prefix that is sent to IMS to indicate that network security credentials should be included in the callout message.

Related information

Network security credentials and IMS TM resource adapter
Configuring Websphere Application Server for distributed network security credentials
Configuring Websphere Liberty for distributed network security credentials
Enabling IMS TM resource adapter to support network security credentials in callout messages
IMSActivationSpec property configuration for message-driven beans

Security considerations for the master terminal

The security of access from the master terminal is critical. Because the MTO can modify all security profiles during normal operations, you should consider protecting the terminal with a second level of control.

Signon verification security provides this capability. The primary question is how much capability to modify security should be given to this second level of control.

Default security does not and cannot prevent modifying the system's security profiles through the master terminal; however, you might want to restrict some commands from being entered from the MTO.

You can use the DFSCCMD0 exit routine to limit the commands that can be entered from the MTO.

Use the DFSCCMD0 exit routine to specify the commands that you want the MTO to be able to enter. The **`/ASSIGN`**, **`/CHANGE`**, and **`/DELETE`** commands are likely candidates to protect. For a cold start or warm start, the MTO can control the following security:

- Signon verification security
- Transaction authorization
- Transaction command security
- Command authorization

You can control these authorizations by making the specifications shown in the following table.

Table 29. Initialization EXEC parameters

Enforced security option	Initialization EXEC parameter
Signon verification	SGN
Transaction authorization	TRN
Terminal security	SGN
Command security	RCF

Security for AO application programs

Automated operator (AO) application programs can issue a subset of IMS operator commands. Because an operator command can compromise a security profile, you should prevent AO application programs from issuing sensitive commands that might modify IMS resources.

There are two types of AO application programs: type-1, which uses the CMD call to issue commands in DB/DC and DCCTL environments, and type-2, which uses the ICMD call to issue commands in DB/DC, DCCTL, and DBCTL environments. Your options for implementing security differ slightly between commands issued using CMD calls and commands issued using ICMD calls.

For both CMD and ICMD calls you must specify which security facilities will enforce AO security and then tailor the system definition macros and startup procedures to work with your security facility choices.

Related reading: For more complete information about AO applications, see *IMS Version 15.2 Operations and Automation*.

Type-1 AO command security

You can use either RACF or another external security product, the Command Authorization exit routine (DFSCCMD0), or both of the security facilities to secure commands that are issued by type-1 AO application programs using the CMD call.

Make your specifications for security facilities using the AOI1= keyword in the IMS and DCC startup procedures.

Use the AOI1= execution parameter in the startup procedure to specify the security facilities that implement security for type-1 AO application programs. Specifications made using AOI1= override the specifications made in the TRANSACT macro.

You can change the AOI1= value each time you initialize IMS because the AOI1= specification is not included in checkpoint records.

Related reading: For detailed information about the AOI1= parameters and their relationship to specifications made using the AOI= keyword in the system definition macros, see *IMS Version 15.2 System Definition*.

Defining the transactions that issue type-1 AO commands

You need to define the transactions that are allowed to issue type-1 AO commands. You can do this using the AOI= parameter in the TRANSACT macro.

The AOI= parameter also allows you to specify the user ID used for security checking when you use RACF, DFSCCMD0, or both as your security facility. You can use any of the following identifiers as a user ID:

- The user ID of the user signed on at the terminal issuing the transaction
- The transaction code
- The first three letters of the commands name

Related reading: For more information on the AOI= parameter, see *IMS Version 15.2 System Definition*.

Type-2 AO command security

You can use either RACF or another external security product, the Command Authorization exit routine (DFSCCMD0), or both of the security facilities to secure AO commands issued by type-2 AO application programs through the ICMD call.

Make your specifications for security facilities by using the AOIS= parameter in the IMS and DCC startup procedures.

Use the AOIS= execution parameter in the startup procedure to specify which security facilities to use for security checking of type-2 AO commands.

The AOIS= execution parameter allows you to specify one of the following options:

- RACF performs security checking.
- DFSCCMD0 performs security checking.
- RACF and DFSCCMD0 both perform security checking.
- Disable all ICMD calls from applications.
- Disable all security checking for ICMD calls.

You can change the AOIS= value each time you initialize IMS, because the AOIS= specification is not included in checkpoint records.

Related reading: For complete information on specifying the AOIS= keyword, see *IMS Version 15.2 System Definition*.

Defining the user ID for transactions that issue type-2 AO commands

Use the AOI= keyword on the TRANSACT macro to specify what RACF and DFSCCMD0 use as a user ID for security checking.

You can use any of the following identifiers as a user ID:

- The user ID of the user signed on at the terminal issuing the transaction
- The transaction code
- The first three letters of the commands name

In certain circumstances, the user ID that is used to issue a transaction might not be available when security checking is performed. In these cases, IMS substitutes other identifiers as the user ID. For information on when IMS substitutes identifiers and what IMS uses as a substitute user ID, see "User ID substitutions for AO application programs" on page 309.

Unlike type-1 AO commands, you cannot disable the ability of a transaction to issue type-2 AO commands. AOI=NO applies only to type-1 AO commands and does not prevent transactions from issuing type-2 commands. For type-2 AO commands, AOI=NO is the same as AOI=YES.

AO command security and system definition

Make specifications that affect AO command security in the TRANSACT macro or with initialization EXEC parameters.

The AOI= keyword in the TRANSACT macro serves the following two functions:

1. You can specify which transactions can issue type-1 AO commands
2. If you use RACF or the Command Authorization exit routine (DFSCCMD0) for security checking, you can specify what the transactions provide as a user ID when invoking type-1 and type-2 AO application programs

For type-1 AO commands, the specifications that you make in the TRANSACT macro can be overridden using the AOI1= keyword in the startup procedure.

AOI=NO is ignored by type-2 AO application programs that use ICMD calls. You cannot disable the ability of a transaction to issue commands through type-2 AO application programs.

For both type-1 and type-2 AO application programs, you can specify the following user IDs for RACF and DFSCCMD0 to use for AO command authorization:

- The user ID that is signed on to the terminal that is issuing the transaction. Specify this with AOI=YES. Because a user ID is not always available for security checking, IMS can substitute an LTERM name, PSB name, or other identifier as the user ID. For more information, see "User ID substitutions for AO application programs" on page 309.
- The transaction code. Specify this with AOI=TRAN.
- The first three letters of the command name. Specify this with AOI=CMD.

To increase your options when implementing AO command security, you can define any one of the three elements involved in security checking—the user ID, the transaction, or the command—as the user to be authorized.

For AO security, commands issued through CMD and ICMD calls are typically defined as a resource and either the signed-on user or the transaction code is defined as the user. However, you can instead choose to define the command name as the user to be authorized and the transaction code as the resource to be protected. Changing which element is defined as the user might be beneficial when, for example, you have so many different transactions that it becomes impractical to add them all to the RACF security class for a command defined as a resource.

Related reading: For detailed information about the TRANSACT macro and the AOI= keyword parameter, see *IMS Version 15.2 System Definition*.

User ID substitutions for AO application programs

In certain circumstances, IMS substitutes other identifiers, such as an LTERM name or program name, as the user ID. Unless you have anticipated this and defined substitute user IDs to RACF, RACF rejects the CMD or ICMD call.

The following examples show how IMS might use user IDs in different circumstances in the AO application program's execution environments. These examples are based on a specification of AOI=YES in the TRANSACT macro.

MPP or IFP
If a message GU call has completed, the user ID is used to determine if the user can issue commands using ICMD. The user ID is of a signed-on terminal or the LTERM name of the signed-off terminal where the transaction is issued. If GU is not issued, PSB name is used.

BMP
If a message GU call has completed, the user ID is used to determine if the user can issue commands using ICMD. The user ID is of a signed-on terminal or the LTERM name of the signed-off terminal where the transaction is issued. If GU is not issued or if the BMP is non-message driven, the value of the USER parameter specified on the JCL JOB statement is used. If the USER parameter is not specified, a user ID of 0000000 is used.

DRA THREAD
The security token that is passed in the PAPL for a schedule request is used to determine whether the user can issue commands using ICMD.

BMP regions are initiated by a **/START** command or by a JCL job. If you use a **/START** command, you should use RACF to protect the library that contains the cataloged procedures. If you use JCL, you can use RACF resource access security (RAS). You control access to the BMP region by defining a unique user ID to RACF on the JOB statement. If you are using RAS, enter the user ID in the RACF security class.

When AO application programs issue a command through the CMD or ICMD call, passwords are not used.

Related reference
IMS type-1 commands supported from an AO application (Commands)

Implementing AO command security with RACF

RACF implements security based on user IDs and resource profiles stored in RACF security classes. RACF allows a user access to a resource only if the user's user ID is associated with the resource in the appropriate RACF security class.

For both type-1 and type-2 AO application program security, you must coordinate the definitions of the user IDs and resources that you make in the RACF resource class with your specifications in the AOI= keyword parameter in the TRANSACT macro.

You might need to define substitute user IDs depending on the type of dependent region in which your application programs are executing and other circumstances. For a description of those circumstances, see "User ID substitutions for AO application programs" on page 309.

Example RACF definitions for AO application program security

When using RACF for security for both type-1 and type-2 AO application programs, you can use the following examples to help code your RACF definitions. The examples define the user IMSUSER1 and the transaction APOL13. The RACF definitions you use differ depending on the specifications that you make by using the AOI= keyword on the TRANSACT macro.

In the following example, where AOI=YES, the command **/STOP** is the protected resource and the user IMSUSER1 is authorized for the command. IMSUSER1 signs on to an IMS terminal and enters transaction APOL13. The transaction issues a CMD or ICMD call to issue the **/STOP** command (For type-2 AO security, the specification of AOI=YES is not required).

```
ADDUSER IMSUSER1 PASSWORD(IUSER1PW) DFLTGRP(SYS1)
RDEFINE CIMS (STO) UACC(NONE)
PERMIT STO CLASS(CIMS) ID(IMSUSER1) ACCESS(UPDATE)
```

In the following example, where AOI=TRAN, the command **/STOP** is the protected resource and the transaction APOL13 is authorized for the command. Any user can enter transaction APOL13. The transaction issues a CMD or ICMD call to issue the **/STOP** command. The APOL13 transaction is authorized to issue the **/STOP** command.

```
ADDUSER APOL13 NOPASSWORD DFLTGRP(SYS1)
RDEFINE CIMS (STO) UACC(NONE)
PERMIT STO CLASS(CIMS) ID(APOL13) ACCESS(UPDATE)
```

In the following example, where AOI=CMD, the transaction APOL13 is the protected resource and the user IMSUSER1 and the command **/STOP** are authorized for that transaction. Authorization is checked twice: when IMSUSER1 enters the transaction at the terminal and when the transaction issues a CMD or ICMD call to issue the **/STOP** command.

```
ADDUSER IMSUSER1 PASSWORD(IUSER1PW) DFLTGRP(SYS1)
ADDUSER STO NOPASSWORD DFLTGRP(SYS1)
RDEFINE TIMS (APOL13) UACC(NONE)
PERMIT APOL13 CLASS(TIMS) ID(IMSUSER1) ACCESS(UPDATE)
PERMIT APOL13 CLASS(TIMS) ID(STO) ACCESS(UPDATE)
```

Related reading:

- For more information about defining users for AO application program security purposes, see "AO command security and system definition" on page 308.
- For more information about programming RACF security classes, see *z/OS Security Server RACF System Programmer's Guide*.
- For more information about RACF commands, see *z/OS Security Server RACF Command Language Reference*.

Implementing AO command security with the command authorization exit routine

The Command Authorization exit routine (DFSCCMD0) is called during AO command processing to perform command authorization checking. DFSCCMD0 enables you to secure commands that are issued through either the CMD or ICMD call at the command verb, keyword, or resource name level.

DFSCCMD0 must be included in IMS.SDFSRESL.

The parameter list for DFSCCMD0 identifies:

- Who issued the command:
  - Terminal
  - LU 6.2 application
  - ICMD call, where a user ID is used for command authorization
  - ICMD call, where a PSB name is used for command authorization
- If RACF (or equivalent) was called:
  - SAF (System Authorization Facility) return code
  - RACF return code
  - RACF reason code
- The security code:
  - X'80000000' RACF was not called (AOIS=C).
  - X'00000000' User is authorized to RACF to issue command.
  - X'00000004' RACF is not available.
  - X'00000008' User is not defined to RACF.
  - X'0000000C' Command is not protected by RACF.
  - X'00000010' User is not authorized to issue command.

Related reading: For more information about the Command Authorization exit routine (DFSCCMD0), see *IMS Version 15.2 Exit Routines*.

Security for Time-Controlled Operations

There are two ways that you can implement security for Time-Controlled Operations (TCO): by restricting the users (LTERMs) that can load a TCO script, and by restricting the IMS commands that a loaded TCO script can issue.

To restrict the LTERMs that can load a TCO script, modify the TCO CNT Edit exit routine (DFSTCNT0) without changing its name. IMS always calls DFSTCNT0 when a TCO script is loaded and, by default, DFSTCNT0 does not restrict the LTERMs.

To restrict the IMS commands that a TCO script can issue, specify A or S on the RCF execution parameter and Y on the TCORACF execution parameter in your startup procedure. IMS calls RACF and, if it exists, the Command Authorization exit routine (DFSCCMD0).

Finally, you must include **`/SIGN ON`** and **`/SIGN OFF`** commands at the beginning and end of the script, respectively. The user ID that is signed on at the beginning of the script is checked for authorization to the commands that the script issues.

Or specify the TCOUSID or SIGNTCO parameter in the DFSDC*xxx* PROCLIB member to let IMS sign on the TCO terminal at restart time.

Related concepts

IMS time-controlled operations (Operations and Automation)

Related reference

RCF= parameter for procedures (System Definition)

TCORACF= parameter for procedures (System Definition)

DFSDCxxx member of the IMS PROCLIB data set (System Definition)

Command Authorization exit routine (DFSCCMD0) (Exit Routines)

Time-Controlled Operations (TCO) Communication Name Table (CNT) exit routine (DFSTCNT0) (Exit Routines)

Security for Fast Path application programs

When you design security protection for Fast Path application programs or for DL/I programs that access Fast Path databases, consider protecting the processing in a dependent region and the network of Fast Path terminals.

Protect the dependent region and terminals by:

- Protecting the processing in a dependent region by securing the PSBs of the Fast Path application programs that are scheduled in the dependent region. You can do this by using RAS security and the RACF IIMS security class.
- Protecting the network of Fast Path terminals by using signon verification and transaction authorization.

Security for JMP application programs

Application programs that run in JMP regions need to access UNIX System Services. Therefore, when running a JMP application, you must use RACF (or an equivalent product) and IMS signon verification security.

To use RACF with JMP regions, define the user IDs, which must have OMVS segments, to RACF.

Related reading: For information about defining user IDs to RACF, see *z/OS UNIX System Services Planning*.

Security and CPI-C driven application programs

You can secure any PSB specified on an APSB call from a CPI-C driven application program using the z/OS System Authorization Facility (SAF).

After APSB SAF is security-enabled, IMS calls SAF to secure the PSB specified on an APSB call against the AIMS or A*xxxxxxx* general resource class (where *xxxxxxx* is the value specified on the RCLASS initialization EXEC parameter) based on the USERID of the user associated with the CPI-C application. Therefore, you must define the PSBs that you want protected by RACF (or your installation exit) to the AIMS or A*xxxxxxx* resource class. Because the AIMS resource class can contain PSBs, all PSB names specified in the AIMS resource class should be unique. In addition, you must specify RCLASS=IMS|*xxxxxxx* on the RCLASS initialization EXEC parameter at IMS system definition time.

When a CPI-C driven application program makes an APSB call, IMS bypasses APSB SAF security if the PSB is not defined to the AIMS resource class or if the AIMS resource class is not active. If IMS bypasses APSB SAF security, IMS attempts to authorize the PSB by using RAS security. If RAS security is active, IMS assumes the CPI-C driven application program has the authority to use the PSB.

Related reading: For more information on RACF and IMS, search for IMS in the *z/OS Security Server RACF Security Administrator's Guide*.

Security for ODBA application programs

You can secure any PSB specified on an APSB call from an ODBA application program using the z/OS System Authorization Facility (SAF).

After APSB SAF is security-enabled, IMS calls SAF to secure the PSB specified on an APSB call against the AIMS or A*xxxxxxx* general resource class, based on the user associated with the ODBA application at the task level (TCB) or the address space level (ASCB/ASXB) respectively. Therefore, you must define the PSBs that you want RACF (or your installation exit) to protect to the AIMS or A*xxxxxxx* resource class.

Since the AIMS resource class can contain PSBs, all PSB names specified in the AIMS resource class should be unique.

You must also specify RCLASS=IMS|*xxxxxxx* with an initialization EXEC parameter during IMS system definition.

Related reading: For more information about RACF and IMS, search for "IMS" in *z/OS Security Server RACF Security Administrator's Guide*.

Security for ODBM allocate PSB (APSB) requests

Any PSB that is specified on an APSB request from an ODBM thread can be secured by using the z/OS System Authorization Facility (SAF), the IMS RAS user exit, or both.

Enabling security for ODBM is accomplished by using one of the following methods:

- Specify ODBMSECURE=A, E, or R.

  This method applies to all ODBM connectors to the respective IMS, irrespective of the ODBM RRS= setting. When ODBMSECURE=N, A, E, or R is specified on an IMS system, the values that are specified in the ISIS= and ODBASE= parameters are ignored for all ODBM connections to this IMS system. The resource class of AIMS or A*xxxxxxx* is used to authorize APSB resources.
- Specify ISIS=A, C, or R.

  This method applies to ODBM connections under either of the conditions:

  – ODBM runs with RRS (RRS=Y) and no SAF calls are to be made against the AIMS resource class (ODBASE=N).
  – ODBM runs without RRS (RRS=N).

  The resource class of IIMS or I*xxxxxxx* is used to authorize APSB resources.
- Specify ODBASE=Y.

  This method applies only to ODBM that runs with RRS (RRS=Y). The resource class of AIMS or A*xxxxxxx* is used to authorize APSB resources.

After the APSB SAF security is enabled, IMS calls SAF to secure the PSB specified on an APSB call by using the respective resource class, based on the user associated with the ODBM thread. IMS then defines the PSBs that are to be protected to RACF (or the installation exit). When ODBMSECURE= or ODBASE= is used, IMS defines the PSBs to AIMS or A*xxxxxxx* resource class. When ISIS= is used, IMS defines the PSBs to IIMS or I*xxxxxxx*.

During IMS system definition, RCLASS=IMS or RCLASS=*xxxxxxx* must be specified with an initialization EXEC parameter.

Related reference

DFSCGxxx member of the IMS PROCLIB data set (System Definition)
ISIS= parameter for procedures (System Definition)
ODBASE= parameter for procedures (System Definition)

Use of the RACF data space

When the RACF (or an equivalent product) data space is supported (RACF 2.1 or later), IMS loads the RACF profiles for the IMS commands and transactions into that data space instead of the IMS control region.

Use of the RACF data space invalidates the IMS online change support for RACF with the **/MODIFY** command. The IMS online change support is still valid, though, when the RACF data space is not being used.

The message DFS3432 RACF PARAMETER INVALID IF RACF DATA SPACE USED is issued if the RACF parameter is used on the **/MODIFY PREPARE** command when the RACF data space is being used. You can use the RACF command **`SETROPTS RACLIST`** *(classname)* **`REFRESH`** to refresh the RACF resource profiles in the RACF data space without requiring the IMS applications to suspend work.

Security in MSC and shared-queues environments

In Multiple Systems Coupling (MSC) and shared-queues environments that are made up of multiple IMS systems, each IMS system can receive transactions from terminals and from the other IMS systems in the same environment. The transactions from the other IMS systems require you to take additional security measures.

In an MSC network, each IMS system can be viewed as a local IMS system or a remote IMS system. A local IMS system receives transactions from a terminal. A remote IMS system receives transactions from a local IMS system through an MSC link. Each IMS system in an MSC environment can be local for one type of transaction and remote for another.

Similarly, in a shared-queues environment, each IMS system can function as a front-end IMS system, a back-end IMS system, or both. A front-end IMS system receives transactions from terminals. A back-end IMS system receives transactions from a front-end IMS system through a shared queue. Each IMS system in a shared-queues environment can function as both a front-end IMS system and a back-end IMS system.

In MSC and shared-queues environments, you must implement security separately for each IMS system. The security controls of each IMS system are independent of those of the other IMS systems. Each IMS system is unaware of the security checking that the other IMS systems perform.

Related reading: For more information about MSC and shared queues, see *IMS Version 15.2 Communications and Connections*.

Local MSC and front-end shared-queues security

Although there are no special security specifications that you need to make for local MSC and front-end shared-queues IMS systems, if the IMS system might also function as a remote MSC or back-end shared-queues IMS system, be sure to implement security appropriately. Otherwise, implement security for local and front-end IMS systems as you would for a standalone IMS system in a DB/DC or DCCTL environment.

Remote MSC and back-end shared-queues security

For remote MSC and back-end shared queues IMS systems, you can use RACF for security. CHNG and AUTH calls and IMS conversational deferred program switches (that occur in the same IMS as the inputting terminal) perform an authorization check to determine whether the user who entered the transaction is authorized to use the IMS resource.

To perform a RACF authorization from the dependent region, a security environment first must be established. If the dependent region is part of the same IMS as the inputting terminal, and the user who entered the transaction is still signed on, then the security environment that is created in the IMS control region at signon is used for the authorization call. The security environment that is created at signon is not available under the following conditions:

- The dependent region is part of the same IMS as the inputting terminal, but the user has signed off.
- The dependent region is part of another IMS, connected to the IMS with the inputting terminal by an MSC link.
- The dependent region is part of another IMS in a sysplex with IMS shared message queues support.

In these conditions, the security environment must be dynamically created in order to perform the RACF authorization check. Dynamic creation of this security environment increases the time required to process a CHNG or AUTH call. The dynamically created security environment is kept until IMS is done with the message (until the next sync point or GET UNIQUE). Use the Build Security Environment user exit (BSEX) to control when IMS performs the dynamic creation of the security environment.

For input from APPC or OTMA, if security is defined as FULL, the security environment has already been created before any CHNG or AUTH call. If APPC/OTMA security is defined as NONE, RACF is not called.

If APPC/OTMA security is defined as CHECK, and a CHNG or AUTH call is made, a dynamic security environment has to be created.

When the following IMS exit routines are called because of an application program CHNG or AUTH call, the address of the CTB is zero when the call is made from an IMS dependent region that is not part of the same IMS as the inputting terminal:

- Command Authorization exit routine (DFSCCMD0)
- Transaction Authorization exit routine (DFSCTRN0)
- Security Reverification exit routine (DFSCTSE0)

Resource Access Security (RAS) security for LTERMs in a shared-queues environment is supported only for LTERMs that are statically defined to that IMS back end.

Related reference

BSEX: Build Security Environment user exit (DFSBSEX0 and other BSEX exits) (Exit Routines)

Security for transactions that are received from an MSC link

You have several options when specifying security for transactions that are received by an IMS system from an MSC link. By default, RACF and exit routines check security for direct-routed transactions and checks security for non-direct-routed transactions received from an MSC link.

Or, if your installation does not require MSC transaction security, you can disable all security checking for transactions received from an MSC link.

If you specify the use of RACF or exit routines to check security for any transactions received on an MSC link, you also need to specify which user ID these security facilities will use.

You can make all of the security specifications that are discussed in this topic by using the positional parameters of the MSCSEC= startup parameter in the DFSDCxxx PROCLIB member. The specifications made in MSCSEC= define the default security that your IMS system performs when it receives transactions from an MSC link.

The security specifications that you make with the MSCSEC= startup parameter are local in scope and apply only to the IMS system in which you make them.

Related reading: For more information about the DFSDCxxx PROCLIB member, see *IMS Version 15.2 System Definition*.

Specifying security for MSC transaction types

The MSCSEC= startup parameter, which is found in the DFSDCxxx PROCLIB member, includes two positional parameters. The first positional parameter allows you to specify—based on the MSC transaction type—if and when a receiving IMS calls RACF and exit routines to check security for a transaction.

The are two types of transactions that can be received on an MSC link for which you might perform security checking: direct-routed transactions and non-direct-routed transactions. You specify which type of transactions that you want RACF and exit routines to check security for by using the first positional parameter of MSCSEC=. The options are:

LRDIRECT
RACF and exit routines check security on direct-routed transactions and no security checking is performed for non-direct-routed transactions. This is the default.

LRNONDR
RACF and exit routines check security on non-direct-routed transactions and no security checking is performed for direct-routed transactions.

LRALL
RACF and exit routines check both types of transactions.

LRNONE
The IMS system does not request any security checking for either type of transaction.

When you use RACF and exit routines to check security for transactions that are received from an MSC link, the receiving IMS system calls RACF and DFSCTRN0 after calling the TM and MSC Message Routing and Control user exit routine (DFSMSCE0).

Specifying user IDs for MSC security

If you use RACF, exit routines, or both to perform MSC security checking, use the second positional parameter of the MSCSEC= startup parameter in the DFSDCxxx PROCLIB member to specify the user ID on which to base security checking.

The parameter options are:

CTL
Specifies that RACF and exit routines use the user ID of the receiving IMS control region. This is the default.

MSN
Specifies that RACF and exit routines use the MSNAME as the user ID.

USR
Specifies that RACF and exit routines use the user ID of the inputting terminal.

Note: User IDs that are defined in RACF resource classes must be unique. When you define user IDs for MSC security, watch for conflicts with existing user IDs.

MSC security and the TM and MSC message routing and control exit routine

The TM and MSC Message Routing and Control exit routine (DFSMSCE0) is concerned primarily with the routing of messages for TM and MSC; however, you can also use this exit to specify which user ID will be used for security checking.

The specifications you can make with DFSMSCE0 regarding user IDs are the same as those specified by the second positional parameter of the MSCSEC= keyword in the DFSDCxxx PROCLIB member.

When IMS calls DFSMSCE0 for MSC link-receive routing, IMS passes to DFSMSCE0 the second security specification defined in the DFSDCxxx MSCSEC= parameter. If the security specification in DFSMSCE0 differs from the one received from IMS for a given transaction, DFSMSCE0 can override the specification received from IMS for that transaction. DFSMSCE0 can also reroute or cancel the transaction.

IMS calls DFSMSCE0 through one of four entry points when transactions are received from an MSC link, but only the following two entry points are used for security checking:

- LRTRAN for non-direct-routed transaction messages local to the receiving IMS system
- LRDIR for direct-routed transaction messages local to the receiving IMS system

Related reading: For more information about the TM and MSC Message Routing and Control exit routine (DFSMSCE0), see *IMS Version 15.2 Exit Routines*.

MSC security and the Transaction Authorization exit routine

The Transaction Authorization exit routine (DFSCTRN0) works with the Security Reverification exit routine (DFSCTSE0) and the Signon/off Security exit routine (DFSCSGN0) to check an individual user ID for authority to use a transaction.

DFSCTRN0 can be used with or without RACF to verify that the user ID is authorized to run a transaction. If both the RACF option and DFSCTRN0 are selected in the IMS system definition, the exit routine is activated after RACF verifies the transaction. If the transaction request is rejected by RACF, the exit routine is not called. If the RACF option is not selected in the IMS system definition, this exit routine can be used to verify the user's authorization and the password, if required, for that transaction.

Related reading: For more information about the Transaction Authorization exit routine (DFSCTRN0), see *IMS Version 15.2 Exit Routines*.

Security for APPC/IMS

APPC/MVS does not verify user authority to access transaction codes or specific IMS systems. To provide complete security verifications of a user's authority to execute transactions with LU 6.2 devices, you must use RACF.

APPC/MVS uses RACF resource class APPCTP for security. This holds a profile for every IMS transaction defined to RACF for transaction authorization verification. The Transaction Authorization exit routine (DFSCTRN0) is called for transactions, and the Command Authorization exit routine (DFSCCMD0) is called for commands.

Security checking with APPC local LUs

You can use either the base logical unit (LU) or local LU for RACF security checking of outbound asynchronous messages. Use the APPCLU startup parameter on the DFSDCxxx PROCLIB member data set to specify whether you want the base or a local LU to be used for asynchronous message responses.

In an IMS 15.2 shared-queues environment, members that could not support local logical units (LUs) can tolerate local LUs; however, the resource information about those local LUs is lost. If an IMSplex member does not support local LUs, then the base LU is used to send asynchronous messages.

It is recommended that all members of an IMSplex migrate to IMS 15.2 before using the APPCLU startup parameter. See *IMS Version 15.2 System Definition* for information on the APPCLU startup parameter.

Security for ETO terminals

ETO provides dynamic LTERM support. You can dynamically create and allocate local LTERMs to a terminal session based on user signon or the application ISRT process. An LTERM can be associated with a specific user ID instead of a physical terminal. By associating an LTERM with a user ID, you prevent the wrong user from receiving messages at a physical terminal.

Related reading: For additional information about ETO security, see "Planning a high-security environment with ETO" in *IMS Version 15.2 Communications and Connections*.

The Initialization exit routine (DFSINTX0), which is called before IMS loads ETO descriptors, can cause user-defined security information to be loaded and made available to security exit routines. DFSINTX0 can also be used to request that IMS reverify new passwords during signon.

After calling DFSINTX0, IMS searches for and uses the following security exit routines in order:

1. Logon exit routine (DFSLGNX0)

   You can include the Logon exit routine (DFSLGNX0) in IMS.SDFSRESL when ETO is implemented. DFSLGNX0 is an installation exit routine. Among the other functions of DFSLGNX0, it enables you to do the following:

   - Allow or disallow a logon attempt based on criteria that you specify.
   - Create or modify the user data that you want IMS to pass to the Sign-on exit routine (DFSSGNX0).
2. Sign-on exit routine (DFSSGNX0)

   You can include the Sign-on exit routine (DFSSGNX0) in IMS.SDFSRESL when ETO is implemented. DFSSGNX0 is an user exit routine. Among its other functions, it enables you to allow or disallow a signon attempt based on any criteria that you specify.

   You can also use the DFSSGNX0 exit routine to dynamically create node user descriptors and to create a user structure name that is different from the user ID, using a suffix and the node name.

Related reading: For more information about DFSSGNX0, DFSLGNX0, DFSINTX0, see *IMS Version 15.2 Exit Routines*.

Securing DB/DC and DCCTL dependent regions and their resources

It is important to protect dependent regions. There are two approaches to securing a dependent region: preventing the unauthorized scheduling of application programs in the dependent region and preventing the unauthorized use of resources by those programs after they have been scheduled.

The following list describes the different types of dependent regions and the resources that can be protected for each:

- For BMP regions:
  - Transaction codes
  - PSBs
  - Logical terminals or transaction codes that are specified by using the OUT= keyword of the IMSBATCH procedure
- For IFP regions: PSBs
- For JBP regions:
  - PSBs
  - Logical terminals or transaction codes that are specified by using the OUT= keyword of the DFSJBP procedure
- For JMP regions: transaction codes
- For MPP regions: transaction codes

IMS supports Resource Access Security (RAS) to protect dependent regions.

Securing dependent regions using resource access security

Resource access security (RAS) prevents an application program that is running in a dependent region from using a resource (a transaction, PSB, or LTERM) unless it is authorized to do so. RAS does not restrict the scheduling of application programs in dependent regions.

The authority of an application program to access a resource that is protected by RAS is based on the user ID of the dependent region. The user ID of the dependent region must be authorized in the RACF security class profile for the resource that the application program is attempting to use.

You can specify RAS by using the ISIS= execution parameter in the IMS, DBC, and DCC startup procedures.

Related reading:

- For additional information about the IMS, DBC, or DCC startup procedures, see *IMS Version 15.2 System Definition*.

RACF resource classes for RAS security

RAS uses the RACF resource classes to define the resources it protects and the user IDs of the dependent regions that can access those resources. The RACF resource classes that RAS uses include:

- IIMS and JIMS for PSBs and groups of PSBs
- LIMS and MIMS for LTERMs and groups of LTERMs
- TIMS and GIMS for transactions and groups of transactions

Some of these classes are predefined by RACF, but others are not. If you do not find the security class that you need among the classes RACF provides, you can use the RACF resource class macro ICHERCDE to create an installation-defined class descriptor table (CDT).

The RACF security classes contain the names of the resources that RAS protects and the user IDs that can use them. When an application program in a dependent region attempts to use a resource, RAS checks the class profile of the resource to see if the user ID of the dependent region is listed as authorized for that resource. If the user ID is specified, RAS grants access to the application program. If the user ID is not specified, RAS denies access.

Related reading:

- For more information about working with RACF security, see:
  - “Preparing a RACF security plan” on page 322
  - *z/OS Security Server RACF Security Administrator's Guide*
  - *z/OS Security Server RACF Macros and Interfaces*
- For more information about defining new classes with the ICHERCDE macro, see *z/OS Security Server RACF System Programmer's Guide*.

Authorizing resource use in a dependent region

For the IMS dependent region, you can use RACF (or an equivalent product) for security. CHNG and AUTH calls and IMS conversational deferred program switches that occur in the same IMS as the inputting terminal perform an authorization check to determine whether the user who entered the transaction is authorized to use the IMS resource.

To perform a RACF authorization from the dependent region, a security environment must be established. If the dependent region is part of the same IMS as the inputting terminal and the user who entered the transaction is still signed on, then the security environment that is created in the IMS control region at signon is used for the authorization call. The security environment that is created at signon is not available under the following conditions:

- The dependent region is part of the same IMS as the inputting terminal, but the user has signed off.
- The dependent region is part of another IMS, connected to the IMS with the inputting terminal by an MSC link.
- The dependent region is part of another IMS in a sysplex with IMS shared message queues support.

In these conditions, the security environment must be dynamically created to perform the RACF authorization check. Dynamic creation of this security environment increases the time required to process a CHNG or AUTH call. The dynamically created security environment is kept until IMS is done with the message (that is, until the next sync point or GET UNIQUE).

For input from APPC or OTMA, if security is defined as FULL, then the security environment is already created before any CHNG or AUTH call. If APPC or OTMA security is defined as CHECK and a CHNG or AUTH call is made, then a dynamic security environment has to be created.

The user ID that is used for creating the security environment for making the authorization call is based on the following environments and criteria:

MPP or IFP

If a message GU call has completed, then the security value from the input message is used to perform the authorization. The security value is the user ID of a signed-on terminal or the LTERM name of the signed-off terminal where the transaction is issued. If a GU has not been issued, then the PSB name is used.

BMP

If a message GU call has completed, then the security value from the input message is used to perform the authorization. The security value is the user ID of a signed-on terminal or the LTERM name of the signed-off terminal where the transaction is issued. If a GU has not been issued or if the BMP is non-message driven, then the value of the USER= parameter that is specified on the JCL JOB statement is used. If the USER= parameter is not specified, then a user ID of 0000000 is used.

If no accessor environment element (ACEE) exists in the IMS control region and a dynamic security environment cannot be created dynamically, then a default security environment is used. If a dependent region has PARDLI=1 specified or an IMS system is specified with LSO=Y, then the default security environment is the environment of the IMS control region that is created with the user ID that is associated with the IMS control region. Otherwise, the default security environment is that of the IMS dependent region that is created with the user ID that is associated with the IMS dependent region.

When the following IMS exit routines are called because of an application program CHNG or AUTH call, the address of the CTB is zero if the call is made from an IMS dependent region that is not part of the same IMS as the inputting terminal:

- Command Authorization exit routine (DFSCCMD0)
- Transaction Authorization exit routine (DFSCTRN0)
- Security Reverification exit routine (DFSCTSE0)

Security for OTMA

Security for OTMA is enforced by RACF or a similar security product.

You can provide OTMA security by using:

- RACF security levels
- System execution procedures, the transaction message prefixes, and the /SECURE OTMA command
- Protecting messages on asynchronous hold queues from unauthorized use

In addition to performing security checks, OTMA can pass to IMS the network security credentials, which includes a network user ID and a network session ID, that are entered from an application in a distributed environment. OTMA passes network security credentials to IMS in the security-data section of the OTMA message prefix if the credentials are entered from an application that uses one of the following user message exits:

- HWSSMPL0
- HWSSMPL1
- HWSJAVA0

You can also use the following OTMA callable interface (C/I) APIs to pass network security credentials to IMS:

- `otma_send_receivey`
- `otma_send_asyncx`

For IMS callout requests, network security credentials are included in the OTMA message prefix of the callout message if both of the following conditions are true:

- Network security credentials are in the security-data section of the OTMA message prefix of the input message to IMS.
- The application that issues the RESUME TPIPE call supports network security credentials.

Related concepts

Distributed network security credential support and OTMA (Communications and Connections)

Related tasks

OTMA security (Communications and Connections)

Retrieving synchronous callout requests with RESUME TPIPE (Communications and Connections)

Related reference

otma_send_asyncx API (Application Programming APIs)

otma_send_receivey API (Application Programming APIs)

Transaction and callout messages (Communications and Connections)

Resume output for the hold queue for tpipe (Communications and Connections)

Security data section (Communications and Connections)

Explanation of OTMA security data fields (Communications and Connections)

Security for IMS Connect

IMS Connect includes a variety of options for implementing and modifying the security checking performed on messages as they arrive in IMS Connect and as they arrive at the data store.

IMS Connect provides two options for checking security within IMS Connect: you can configure IMS Connect to call RACF directly or you can have the IMS Connect user message exit routines call a Security user exit routine.

Additional security features provided by IMS Connect include:

- Password management support
- Trusted-user classification for messages arriving at the data store
- OTMA accessor environment element (ACEE) timeout specification support
- For connections from clients connecting to IMS DB:
  - Secure Sockets Layer (SSL) support. To use SSL to secure connections from clients connecting to IMS DB, you can use IBM z/OS Communications Server Application Transparent Transport Layer Security feature (AT-TLS).
  - Support for RACF PassTickets.
- For IMS TM clients, IMS Connect provides support for RACF PassTickets.
- Support for passing to and from IMS the security credentials, including the network user ID and network session ID, that are entered by a user in a distributed environment.

Related concepts
IMS Connect security support (Communications and Connections)

Activating IMS security for DB/DC and DCCTL environments

This topic provides guidance on activating your IMS security design and on using RACF and program exit routines.

Preparing security exit routines

Each of the security exit routines need to be prepared as part of authorization.

- Signon/off Security exit routine (DFSCSGN0)

  The /SIGN ON/OFF Security exit routine must be coded by your installation as module DFSCSGN0. This exit routine should have access to a table of valid user IDs and their associated passwords and RACF PassTickets (if they are used). For addressability, the table should reside in module DFSCSGN0, the Transaction Authorization exit routine (DFSCTRN0), or in the IMS nucleus. The exit routine should note each successful signon. When the **/SIGN OFF** command is executed, the exit routine should mark that user ID available for /SIGN ON. The exit routine can place information in the data portion of the user verification string for logging. (An address in a register points to the user verification string.)
- Transaction Authorization exit routine (DFSCTRN0)

  The Transaction Authorization exit routine (DFSCTRN0) should have access to a table of valid user IDs, RACF PassTickets, passwords, and transactions associated with each valid user ID. For addressability, this table should reside in module DFSCTRN0, the /SIGN ON/OFF Security exit routine (DFSCSGN0), or in the IMS nucleus. If the table is in the nucleus, it can be shared by the Transaction Authorization exit routine and the Signon Verification exit routine.

  If you use message edit routines, security is checked after the message is edited.
- Command Authorization exit routine (DFSCCMD0)

  The Command Authorization exit routine (DFSCCMD0) should have access to a table of valid user IDs, passwords, and commands associated with each valid user ID. For addressability, this table should reside in module DFSCCMD0, the /SIGN ON/OFF Security exit routine (DFSCSGN0), or in the

IMS nucleus. If the table is in the nucleus, it can be shared by the Command Authorization exit routine, the Transaction Authorization exit routine, and the Signon Verification exit routine.

- If you are using the Resource Access Security exit routine (RASE)

  You can use the Resource Access Security exit routine (RASE) to augment or refine the security functions that are provided by RAS. The RASE user exit is called after a call to RACF to directly authorize an IMS dependent region to use IMS resources (transactions, PSBs, and LTERMs) that the dependent region attempts to access. The description of the Resource Access Security user exit in the Exit Reference tells how to define the RASE user exit to IMS.

Related tasks

Using the Transaction Authorization exit routine (DFSCTRN0) (Communications and Connections)

Related reference

RASE: Resource Access Security user exit (DFSRAS00 and other RASE exits) (Exit Routines)

Command Authorization exit routine (DFSCCMD0) (Exit Routines)

Signon/off Security exit routine (DFSCSGN0) (Exit Routines)

Preparing a RACF security plan

A security plan for RACF includes defining which resources need to be protected, specifying security options in system definition macros, and defining the resources that you want to protect to RACF.

To prepare a security plan that uses RACF:

1. Prepare a list of all of the IMS online resources to be protected, arranging them in groups to give an overview of the total resources covered.
2. Select the security facilities that protect the resource groups.
3. Design screen formats to include non-display fields for passwords in transactions and commands.
4. Specify your security options in the SECURITY, COMM, and IMSGEN system definition macros.
5. Define the RACF resource class profiles to RACF.
6. Add users, groups, and data sets to RACF.
7. Define transactions and transaction groups to RACF.
8. Define databases, segments, fields, and other resources and resource groups to RACF.
9. Define commands and command groups to RACF.
10. Define extended resource protection sources (APPL).
11. Modify JCL procedures in IMS.PROCLIB.

IMS security uses a variety of RACF resource classes. These classes define individual resources or groups of resources, and they are divided into the following categories:

APPC/IMS

The APPC resource classes used by APPC/IMS are not specific to IMS and include:

APPCTP

Identifies transaction profiles for LU 6.2 transactions to RACF.

APPCLU

Specifies the conversation security for a session.

APPCPORT

Controls access to the system from a given LU (APPC port of entry).

Related reading: For additional information on APPC/IMS and the RACF resource classes used by APPC/IMS, see the APPC Transaction Security topic in *IMS Version 15.2 Communications and Connections*.

Application

The application resource class, APPL, holds a profile of every subsystem that is defined to RACF. The application group resource class, AIMS, holds a profile for every APSB that is defined to RACF. The

IMS system is defined in this class with the IMSID name (with the IMSCTRL macro) for system access authorization checking at signon.

Command
The command resource class, CIMS, holds a profile for every command that is defined to RACF for command authorization checking. The command group resource class, DIMS, allows grouping of IMS commands that have a common access authority profile. Commands are defined for authorized user IDs.

Database
The database resource class, PIMS, holds a profile for each database that is defined to RACF for authorization checking. The database group resource class, QIMS, allows grouping of database resources that have a common access authority profile.

Field
The field resource class, FIMS, allows RACF authorization checking of fields within a database. The field resource group class, HIMS, allows grouping of common access fields for RACF authorization checking.

LTERM
The LTERM resource class, LIMS, holds a profile for every IMS LTERM that is defined to RACF for LTERM authorization checking. The LTERM group resource class, MIMS, allows grouping of IMS LTERMs that have a common access authority profile.

Other
This resource class, OIMS, and its group resource class, WIMS, are available for you to use for resources that do not fit into any other class. This calls can be used to interface with the AUTH call.

PSB
The PSB resource class, IIMS, holds a profile for every IMS PSB that is defined to RACF for PSB authorization checking. The PSB group resource class, JIMS, allows grouping of IMS PSBs that have a common access authority profile.

Segment
The segment resource class, SIMS, identifies individual segments to RACF . The segment group resource class, UIMS, allows grouping of segments with a common access authority profile for RACF authorization checking.

Transaction
The transaction resource class, TIMS, holds a profile for every IMS transaction defined to RACF for transaction authorization checking. The transaction group resource class, GIMS, allows grouping of IMS transactions that have a common access authority profile.

Use the RCLASS= initialization EXEC parameter to specify the names of the resource classes that the IMS system will use.

The following table shows resource class assignments.

Table 30. Resource class assignments

	Resource class name	
Resource class type	**RACF-defined name**	**User-defined name**
APPC/IMS	APPCTP, APPCLU, APPCPORT, and others	These are not IMS-specific resource classes
Application resource class	APPL	This is not an IMS-specific resource class
Application group name resource class	AIMS	Axxxxxxx
Command resource class	CIMS	Cxxxxxxx
Command group resource class	DIMS	Dxxxxxxx

Table 30. Resource class assignments (continued)

	Resource class name	
Resource class type	**RACF-defined name**	**User-defined name**
Database resource class	PIMS	Pxxxxxxx
Database group resource class	QIMS	Qxxxxxxx
Field resource class	FIMS	Fxxxxxxx
Field group resource class	HIMS	Hxxxxxxx
LTERM resource class	LIMS	Lxxxxxxx
LTERM group resource class	MIMS	Mxxxxxxx
Other resource class	OIMS	Oxxxxxxx
Other group resource class	WIMS	Wxxxxxxx
PSB resource class	IIMS	Ixxxxxxx
PSB group resource class	JIMS	Jxxxxxxx
Resume TPIPE class	RIMS	Rxxxxxxx
Segment resource class	SIMS	Sxxxxxxx
Segment group resource class	UIMS	Uxxxxxxx
Transaction resource class	TIMS	Txxxxxxx
Transaction group resource class	GIMS	Gxxxxxxx

The RACF resource classes are defined in the RACF resource class descriptor table (CDT). Initially, the RACF-defined resource classes (shown in column 2 of Table 30 on page 323) are predefined in the CDT. To add a resource class or to define a resource class with a user-defined name, you must use the RACF resource class macro ICHERCDE to create an installation-defined CDT.

Requirement: If you are going to allow mixed-case passwords (and you are using RACF for security), you must issue the **`SETROPTS,PASSWORD(MIXEDCASE)`** command.

Related reading:

- For more information about the ICHERCDE macro, see *z/OS Security Server RACF Macros and Interfaces*.
- For more information about updating the RACF resource class descriptor table, see *z/OS Security Server RACF System Programmer's Guide*.

Related reference

AUTH call (Application Programming APIs)

Enabling and disabling APSB SAF security

You can enable APSB SAF security using one of the following methods.

- Specify RACF=FULL in the TP scheduler section of the CPI-C application's TP profile and issue the IMS command **`/SECURE APPC PROFILE`**. The command **`/SECURE APPC PROFILE`** enables APSB SAF Security only for the CPI-C applications that have RACF=FULL specified in the TP profile. The command disables APSB SAF Security for all other CPI-C applications.
- Issue the IMS command **`/SECURE APPC FULL`** to enable APSB SAF security for all CPI-C applications.

To disable APSB SAF security, issue the IMS command **`/SECURE APPC CHECK`** or **`/SECURE APPC NONE`**.

Restricting access to the RS catalog repository and IMSRSC repository

The Repository Server (RS) uses z/OS system authorization facility (SAF) callable services to restrict access to the RS catalog repository and IMSRSC repository. You can use an SAF-enabled external security manager, such as RACF, to administer RS security.

Resources that can be restricted from or authorized for access include:

- A repository
- An RS catalog repository
- Members within a repository
- Audit levels associated with a repository

Protect resources by defining general resource profiles with the RACF **RDEFINE** command.

Grant access to users that have defined resource profiles with the RACF **PERMIT** command.

To restrict access to the RS catalog repository and user repository:

1. In your security database, do the following:
   a) Choose an existing resource class, such as FACILITY, or create a class for RS security checking.
   b) Define general resource profiles under this class as follows:
      - You can control access to user repositories by defining one or more resource profiles using the following form: `FRPREP.`*`repositoryname`*

        For example, you can define a resource profile using FRPREP.IMS_REPOS to secure a repository named IMS_REPOS, or you can define a resource profile using FRPREP.* to secure all repositories.

        In addition to a resource profile for FRPREP, you must also add a resource profile for FRPFLD with the same definitions as FRPREP to ensure that the Resource Manager (RM) address space can set the index, key, and security fields in the IMSRSC repository, as needed. For example, if you defined a resource profile in XFACILIT class using FRPREP.IMS_REPOS to secure a repository named IMS_REPOS, you should also add FRPFLD.IMS_REPOS to allow the FIELDS to be updated for the repository named IMS_REPOS. The product and type must also be specified as ***.** to ensure that the resource profile definition for XFACILIT class would be RDEFINE XFACILIT FRPFLD.IMS_REPOS.*.*
      - To control access to who can modify or access the RS catalog repository as a part of the FRPBATCH commands or other requests, you can define a resource profile for FRPREP.CATALOG and grant access only to authorized users.
      - You can control access to each resource or member by defining member level security. Member level security is enabled for access by non-authorized callers.

        Member level security is enabled based on the security fields defined for the user repository. You can define a resource profile for each member you want to secure using the following form: `FRPMEM.`*`repositoryname.product.type.membername`*

        Recommendation: For the FACILITY class, the profile name can be only 39 bytes long. If member level security is used, create a new RACF class for the RS by adding a new class to the RACF Class Descriptor Table (ICHRRCDE) and updating the RACF Router Table (ICHRFR01) with the new class.
      - To control who can modify the AUDIT levels, you can define a resource profile using the following form: `FRPAUD.`*`repositoryname.product.type`*`.TYPE`

        Considerations for specifying *product* and *type* are the same as for when setting security for the repository in the CSL RM address space.
   c) Authorize the appropriate users for these profiles by setting a RACF definition profile.

      You can authorize groups of user IDs at the same time by using the RACF **ADDGROUP** command.

   Here are examples of the RACF commands that are used to define profiles and then grant access to appropriate users:

Repository:

```
RDEFINE XFACILIT FRPREP.REPO1 UACC(NONE)
```

```
PERMIT FRPREP.REPO1 CLASS(XFACILIT) ID(VIEWER1) ACCESS(READ)
```

```
PERMIT FRPREP.REPO1 CLASS(XFACILIT) ID(ADMIN1) ACCESS(ALTER)
```

RS catalog repository:

```
RDEFINE XFACILIT FRPREP.CATALOG UACC(NONE)
```

```
PERMIT FRPREP.CATALOG CLASS(XFACILIT) ID(ADMIN1) ACCESS(ALTER)
```

Members within a repository:

```
RDEFINE XFACILIT FRPMEM.REPO1.DFS.RSC.IMSPLEX1.TRAN.PART UACC(NONE)
```

```
PERMIT FRPMEM.REPO1.DFS.RSC.IMSPLEX1.TRAN.PART CLASS(XFACILIT) ID(USER_UTIL10)
ACCESS(UPDATE)
```

```
PERMIT FRPMEM.*.*.*.*.*.* CLASS(XFACILIT) ID(USER_UTIL20) ACCESS(READ)
```

Audit levels associated with a repository:

```
RDEFINE XFACILIT FRPAUD.REPO1.DFS.RSC.TYPE UACC(NONE)
```

```
PERMIT FRPAUD.REPO1.DFS.RSC.TYPE CLASS(XFACILIT) ID(USER_ZOSMI) ACCESS(UPDATE)
```

2. In the FRPCFG member of the IMS PROCLIB data set, give the SAF_CLASS parameter the same class name as the one you have defined in the RACF definition profile.

Related concepts
Overview of the IMSRSC repository (System Definition)
Related reference
ADD command for FRPBATCH (System Programming APIs)
Repository Server commands (Commands)
FRPCFG member of the IMS PROCLIB data set (System Definition)
Related information
Commands for FRPBATCH (System Programming APIs)

Considerations for setting security for the IMSRSC repository in the CSL RM address space

The Common Service Layer (CSL) Resource Manager (RM) address space is an authorized caller of the IMSRSC repository services. You can specify if the RM address space is authorized to access a repository by defining a profile with FRPREP.*repository_name*.

If RM is authorized to access a repository, then all authorized callers to RM are able to access the repository through RM and also access all members in the repository, whether member level security is defined or not. Member level security is not used for authorized callers to RM.

If RM is authorized to access a repository, then the access to the members in the repository for non-authorized callers is based on the member level security. If member level security is not set up for the repository, then all non-authorized callers to RM will be able to access the repository through RM and also access all members in the repository. If member level security is enabled, then non-authorized callers to RM are able to access the repository through RM only if the caller is authorized to access the member.

You can define the member level security for RM using the following values for *product*, *type*, and *membername* to the Repository Server (RS):

- *product*=DFS
- *type* = RSC
- *membername=plexnamersctyperscname*

  plexname
  The 8-byte CSL IMSplex name where the repository is defined

  rsctype
  The 8-byte resource type. It can be one of the following:

  - DB
  - DBDESC
  - LTERM
  - MSLINK
  - MSNAME
  - MSPLINK
  - PGM
  - PGMDESC
  - TRAN
  - TRANDESC
  - RTC
  - RTCDESC

  rscname
  The 8-byte resource name you want to secure

As an example, to secure a transaction named PART in IMSPLEX1 in repository IMS_REPOS, define a rule for FRPMEM.IMS_REPOS.DFS.RSC.IMSPLEX1.TRAN.PART to RACF. All authorized callers to RM can access the transaction PART. All non-authorized callers to RM (such as the repository populate utility) can access the transaction PART only if the user ID on the job issued to run the utility is defined to RACF with the member level security defined with the access rule for FRPMEM.IMS_REPOS.DFS.RSC.IMSPLEX1.TRAN.PART.

Related concepts

Overview of the IMSRSC repository (System Definition)

"CSL RM management of the IMSRSC repository" on page 181
The Common Service Layer (CSL) Resource Manager (RM) interacts with the Repository Server (RS) address space to manage the stored resource definitions in the IMSRSC repository.

" Security considerations for a DBCTL environment" on page 334
This topic contains information about establishing security for an IMS DBCTL environment.

Example RACF definition profile for the Repository Server

The following is an example of how to define the Repository Server (RS) to RACF for a RACF class called XFACILIT.

```
/* Define Resource Profiles */
/* FRPREP.CATALOG */
RDEFINE XFACILIT FRPREP.CATALOG UACC(NONE)
/* FRPREP.<repository name> */
RDEFINE XFACILIT FRPREP.* UACC(NONE)
/* FRPMEM.<repository name>.<product>.<type>.<plexname>.<rsctype>.<rscname> */
RDEFINE XFACILIT FRPMEM.*.*.*.*.*.** UACC(NONE)
/* FRPAUD.<repository name>.<product>.<type> */
RDEFINE XFACILIT FRPAUD.*.*.* UACC(NONE)

/* Define Groups */
/* View Data Group */
ADDGROUP FRPVIEW
/* View Edit Group */
```

```
ADDGROUP FRPEDIT
/* Operations Group */
ADDGROUP FRPOPER
/* Administration Group */
ADDGROUP FRPADMIN
/* Grant Access to groups as follows */
/* View Data Group */
PERMIT FRPREP.* CLASS(XFACILIT) ID(FRPVIEW) ACCESS(READ)
PERMIT FRPMEM.*.*.*.*.*.** CLASS(XFACILIT) ID(FRPVIEW) ACCESS(READ)
/* Edit Data Group */
PERMIT FRPREP.* CLASS(XFACILIT) ID(FRPEDIT) ACCESS(READ)
PERMIT FRPMEM.*.*.*.*.*.** CLASS(XFACILIT) ID(FRPEDIT) ACCESS(UPDATE)

/* Operations Group */
PERMIT FRPREP.CATALOG CLASS(XFACILIT) ID(FRPOPER) ACCESS(READ)
PERMIT FRPREP.* CLASS(XFACILIT) ID(FRPOPER) ACCESS(CONTROL)

/* Administration Group */
PERMIT FRPREP.CATALOG CLASS(XFACILIT) ID(FRPADMIN) ACCESS(READ)
PERMIT FRPREP.* CLASS(XFACILIT) ID(FRPADMIN) ACCESS(ALTER)
PERMIT FRPMEM.*.*.*.*.*.** CLASS(XFACILIT) ID(FRPADMIN) ACCESS(UPDATE)
PERMIT FRPAUD.*.*.* CLASS(XFACILIT) ID(FRPADMIN) ACCESS(UPDATE)
PERMIT FRPFLD.*.*.* CLASS(XFACILIT) ID(FRPADMIN) ACCESS(UPDATE)
PERMIT FRPHST.*.*.* CLASS(XFACILIT) ID(FRPADMIN) ACCESS(UPDATE)

/* Connect Users */
CONNECT <user1> GROUP(FRPVIEW)
CONNECT <user2> GROUP(FRPEDIT)
CONNECT <user3> GROUP(FRPOPER)
CONNECT <user4> GROUP(FRPADMIN)
```

Related concepts

Overview of the IMSRSC repository (System Definition)

Controlling security during system startup for DB/DC and DCCTL

The EXEC parameters in the IMS and DCC startup procedures provide a way to control the kind of security checking that is done during the current execution. The parameters act as switches for the different types of security that are specified in the system definition macros.

They also determine what flexibility the MTO has to override the choice of security checking. You must coordinate the setting of these parameters with both overall security design and operational procedures. The EXEC parameters for security are RCLASS, SECCNT, TRN, SGN, RCF, ISIS, ASOT, ALOT, AOI1, AOIS, and TCORACF.

The default values for the IMS and DCC procedures all specify no security. You must reset them to enable security.

The security functions and the EXEC parameters that you use to specify them are shown in the following table. Some of the EXEC parameters in the table override or replace related parameters specified in the system definition macros.

You must match the level of the security tables with the suffix identifier for the nucleus. Operational restrictions for the MTO are described in "Security considerations for the master terminal" on page 306.

Related reading: For detailed information on the JCL parameters and their parameter values, see *IMS Version 15.2 System Definition*.

Table 31. EXEC parameters to control IMS security

Choice of security function	EXEC parameter	Parameter value for security choice		Notes
		Disable	Enable	
Identification of IMS to RACF as a resource class	RCLASS	do not specify RCLASS	specify a name on RCLASS and also specify RCF=	

Table 31. EXEC parameters to control IMS security (continued)

Choice of security function	EXEC parameter	Parameter value for security choice		Notes
		Disable	Enable	
Number of security violations before MTO is notified	SECCNT	0	1, 2, 3	
Transaction authorization	TRN	N	Y, F	"1" on page 330, "2" on page 330, "11" on page 330
Signon verification	SGN	N	D, E, F, M, W, X, Y, Z, G,	"1" on page 330, "2" on page 330, "3" on page 330, "8" on page 330
RACF security for transaction authorization or signon verification	RCF	N	A, B, C, R, S, T, Y	"4" on page 330, "5" on page 330, "8" on page 330
RAS dependent region security	ISIS	0, 1, 2, N	A, C, R	"6" on page 330, "9" on page 330
Autosignoff	ASOT	0 or 1440	10-1439	"7" on page 330
Autologoff	ALOT	0 or 1440	10-1439	
Security checking for CMD calls	AOI1	N	A, C, R, S	"8" on page 330
Security checking for ICMD calls	AOIS	S	A, C, N, R	"8" on page 330
RACF check of TCO-issued commands	TCORACF	N	Y	"8" on page 330, "10" on page 330

Table 31. EXEC parameters to control IMS security (continued)

Choice of security function	EXEC parameter	Parameter value for security choice		Notes
		Disable	Enable	

Notes:

1. With value N, on the **/NRESTART** command, the MTO can optionally invoke checking.
2. With value Y, the security function is active unless overridden by the MTO.
3. Value M indicates multiple signons for a single user ID. Value Z is equivalent to Y + M; value G is equivalent to F + M.
4. The RACF licensed program is used in conjunction with Command Authorization, Transaction Authorization, or Signon Verification exit routines.
5. If a null value is specified, the choice is the default to that given in system definition.
6. The ISIS keyword parameter allows you to choose the type of dependent region security you want, RAS, and which security facilities, RACF , and exit routines, the security type will use.
7. On a terminal defined with ETO, when the last autologon user's last queue is completed, the autologon user immediately signs off without waiting for the autosignoff timeout interval.
8. Because this specification is not included in a checkpoint record, you can change its value each time IMS is initialized.
9. ISIS parameter values of 0, 1, or 2 are tolerated for compatibility. Internally, these values are converted to ISIS=N.
10. If TCOUSID or SIGNTCO is specified in the DFSDC*xxx* PROCLIB member, TCORACF is set to Y.
11. Perform a cold start of the IMS system for the changes to take effect.

Related reference

DFSDCxxx member of the IMS PROCLIB data set (System Definition)
DCC procedure (System Definition)
IMS procedure (System Definition)
/NRESTART command (Commands)

Implementing security changes online in DB/DC and DCCTL environments

You can perform changes to your security definitions online when you use RACF for security enforcement.

To change RACF security definitions online, update the RACF database. Then issue the RACF command **SETROPTS RACLIST(***classname***) REFRESH**.

RACF refreshes the class that corresponds to *classname* and all classes that share the same POSIT value on their class descriptor table (CDT) entries.

Related reading: For more information on the SETROPTS command, see *z/OS Security Server RACF Command Language Reference*.

Controlling security violations in DB/DC and DCCTL environments

Security violations are handled according to the installation's security administration guidelines.

IMS records the following security violation attempts on the IMS system log:

- Input message from an unauthorized terminal
- Password omitted when one is required
- Password incorrect for authorization

- Misspelled password
- Rejected signon
- Unauthorized DL/I command (CMD) call from application program

IMS rejects invalid input messages by sending a message to the terminal entering the message and logging the violation. The IMS system log provides an audit trail for investigation of possible security problems. The IMS system log security violation is identified as a X'10' log record type. You can use the File Select and Formatting Print utility to print the log.

You might want to have tighter security so that you are immediately notified about security violations. You can arrange for the master terminal to be immediately notified about security violations by having messages sent to it whenever the violations occur. To have the master terminal notified when violations occur, specify a non-zero value for the SECCNT initialization EXEC parameter.

However, in a large network, misspelled passwords, transaction codes, and commands can cause an extremely large number of violations and violation notifications. You can reduce the number of notifications caused by operator errors, while still providing evidence of real attempts to avoid security safeguards, by specifying a notification threshold. When the number of violations from a single terminal equals the notification threshold value (as specified by the SECCNT value), the master terminal is notified.

Another method for recording security violations is available when RACF is installed. Each resource access violation creates a RACF type 80 record. You can use the RACF report writer to create reports based on these records.

Related reading:

- For more information about the SECCNT initialization EXEC parameter, see *IMS Version 15.2 System Definition*.
- For more information on the File Select and Formatting Print utility, see *IMS Version 15.2 System Utilities*.
- For more information on using the RACF report writer to format and print RACF records, see *z/OS Security Server RACF Auditor's Guide*.

Considering other access control methods

Use other access methods to augment your security software.

Physical security

You should consider physical security measures that support your system security.

These measures include:

- Controlled access to and from the computer area
- Authorization of DP operations and non-operations personnel in certain terminal areas
- Separately controlled areas for media such as tapes, disks, cards, or files
- Control of computer forms and printed output

Physical security needs are likely to be dynamic and merit periodic review and adjustment.

Using display bypass and password masking in DB/DC and DCCTL

IMS does not provide a software function to blank out or obliterate passwords from the terminal device display media after they are accepted. However, Message Format Service (MFS) facilities enable users to define fields with a non-display attribute (for 3270 display devices). IMS removes passwords from messages prior to recording them on the log.

If you plan to use passwords as part of transaction and command entry, you should design screen formats to incorporate non-display fields. This protection is especially important for the **/SIGN** command. The

DFS3649 signon required message has non-display fields built into it for entering passwords on ACF/VTAM display terminals.

Most key-driven terminals have a feature (called the bypass feature) that permits characters to be entered without displaying them. Ordinarily, a terminal with this feature is operated continuously either in display or bypass mode. If passwords are to be masked to support security requirements, this feature is a necessity.

The bypass feature can be used operationally for establishing standards of protection for not only passwords, but also command verbs, commands, transaction codes, and text.

If passwords or other sensitive data must be altered in the IMS log, IMS provides a user exit for this purpose.

Related reference

LOGEDIT: Log edit user exit (DFSFLGE0 and other LOGEDIT exits) (Exit Routines)

Protecting your resources

You can protect IMS system libraries and data sets, as well as VSAM, OSAM, and Fast Path databases, in both the online (DB/DC, DBCTL, and DCCTL) and batch environments.

IMS system libraries and system data sets

You can use RACF to protect IMS system libraries and system data sets. IMS invokes RACF to determine whether the user ID associated with the system address space (control region, DLISAS, or batch) attempting to open the resource has the necessary access authorization. Actually, when RACF authorizes access, it associates a user ID with the started procedure name (IMS or DLISAS procedure) through a started task table. If you start IMS with JCL, the RACF user ID can be on the job card along with its password.

If the user ID does not have the authorization, access is denied. The basic rule is "Whoever has the DD card must have the authority."

Related reading: For more information on this process, see *z/OS Security Server RACF Security Administrator's Guide*.

IMS procedure

If the IMS procedure is associated with a RACF user ID (with sufficient authority), the IMS control region can open a RACF-protected data set. If an association does not exist, the IMS control region is not allowed to open a RACF-protected data set that does not allow universal access for the requested authority level.

DLISAS procedure

If the DLISAS procedure is associated with a RACF user ID, it overrides the RACF user ID for the IMS procedure. If an association does not exist, the RACF user ID associated with the IMS procedure is used for RACF access checking.

Protecting databases

You can protect your VSAM and OSAM full-function databases, as well as your Fast Path DEDBs by using segment-level sensitivity, field-level sensitivity, or RACF.

Segment- and field-level sensitivity: Through centralized control over the content of database definitions, program specification blocks, and the libraries in which they reside, an effective scheme of protection attributes can be assigned to data. Note, however, that for database protection through PSBs to be totally effective, you should also protect the PSB library and the application program library (to safeguard the code that accesses the databases).

Segment-level sensitivity

If you are not using field-level sensitivity, the smallest unit of data that can be protected is the segment. The basic actions that can be authorized are:

None
No access to segment type.

Read
Segment type can only be retrieved.

One or more of the following additional actions combined with read can be authorized:

Add
New occurrences of segment type can be inserted.

Update
An existing occurrence of a segment type can be replaced.

Delete
An existing occurrence of a segment type can be deleted.

The way the PCB and the parameter values for the PROCOPT keyword are specified controls the authorization. Although access authorization is declared at the program level, enforcement of the authorization can be made to appear at the transaction code or individual hierarchic level of a database. If only one transaction code is associated with a particular program, then the access authorization is effective at the transaction level. By using SENSEG statements in the PSB and key sensitivity as a processing option for higher-level segments, masking can be effective at the individual hierarchic level.

Field-level sensitivity

Field-level sensitivity can provide another kind of database security. Database descriptions (DBDs) and PSBs can be coded to permit access to a required subset of fields within a segment. Field-level sensitivity can also be used to control the replace function at the field level to help ensure database integrity.

Related reading:

- For more information on security at the database level, see *IMS Version 15.2 Database Administration*.
- For information about generating DBDs and PSBs, see "Application Control Blocks Maintenance utility" and "Program Specification Block (PSB) generation utility" in *IMS Version 15.2 System Utilities*.

Protecting databases with RACF

You use the RACF user ID of the DLISAS or control region started procedure, depending on the environment you are executing. If you start IMS with JCL, the RACF user ID can be on the job card along with its password.

VSAM full-function database

In an online environment, if a RACF user ID is associated with the DLISAS started procedure, that ID is used for access checking. If a RACF user ID is not associated with the DLISAS started procedure, the control region RACF user ID is utilized. (In the batch environment, the user ID of the batch job is employed.) Access authority of "CONTROL" is required. The database must be defined as ICFCATALOG. Use of the older "VSAM" catalog type is restricted.

OSAM full-function database

In an online environment, the RACF user ID of DLISAS is used; in the batch environment, the user ID of the batch job is used.

Fast Path DEDBs

In an online environment, the control region RACF user ID is used. (No batch environment exists.)

Additional protection

You can also implement database security with the DATABASE, FIELD, and SEGMENT classes in RACF.

Related reading: For more information on these resource classes, see "Preparing a RACF security plan" on page 322.

Encryption: an alternative to access control

When preventing access to the data is difficult or impractical, encryption can protect data that is in files or data that is being communicated in a network. IMS offers some file encryption capability (through IMS Segment Edit/Compression exit routines, for example) but no communication encryption capability.

The following list provides additional information about encryption:

Using cryptographic support
The Programmed Cryptographic Facility, program number 5740-XY5, provides file and communications encryption under z/OS. File encryption of the physical hierarchical database keeps unauthorized individuals from looking at the data when the physical disk pack containing the database is removed from its usual area. File encryption support extends to VSAM physical databases. Communications encryption supports ACF/VTAM supported terminals.

Using the segment edit/compression exit routine (not DCCTL)
You can use this routine to provide data encryption. By including the IBM Programmed Cryptographic Facility within your exit routine, you can reduce your programming effort. The facility is executed by assembler macro calls. Segments are encrypted before being placed in the database buffer pool. The SEGM control statement in the IMS DBDGEN includes a keyword to specify the name of this exit routine.

Using the ICSF/CCA interface
You can use ICSF/CCA APIs in the IMS DB Segment Edit/Compression exit. IMS supports the Programmed Cryptographic Facility (PCF) interface transparently through the ICSF/CCA interface. Programs that are written to the PCF interface run, without modification, through the ICSF/CCA interface. If you want your PCF programs to use the ICSF/CCA APIs, however, you must modify those PCF programs.

The ICSF/CCA interface has two PCF compatibility modes.

- ICSF mode COMPAT(YES) means that programs written to the Programmed Cryptographic Facility interface run without change, as well as calls made directly to the ICSF/CCA API. There are some limitations for dynamic master key change in this mode.
- ICSF mode COMPAT(NO) means only programs coded to the CCA API run.

Security considerations for a DBCTL environment

This topic contains information about establishing security for an IMS DBCTL environment.
Related concepts
"Considerations for setting security for the IMSRSC repository in the CSL RM address space" on page 326
The Common Service Layer (CSL) Resource Manager (RM) address space is an authorized caller of the IMSRSC repository services. You can specify if the RM address space is authorized to access a repository by defining a profile with FRPREP.*repository_name*.

DBCTL resources that can be protected

Before you decide what security facilities to use in designing a secure IMS system, you should know which resources within the system need protection. In other words, you should decide what to protect before you decide how to protect it.

The following resources can be protected in a DBCTL environment:

Control region
The IMS system control region that enables online application programs to process the database through terminals.

System data set
A collection of data that is fundamental to the operation of the IMS online system. An example is the IMS.PROCLIB data set.

Dependent region
An area of storage in the IMS online system in which batch or online application programs are executed. In a DBCTL environment, the dependent region can be a BMP region, a JMP region, or a CCTL region.

PSB
Program specification block. The control block that describes a group of hierarchic databases and logical message destinations used by an online application program.

BMP application program
A program in a DBCTL environment that performs work for a user. Batch message programs are activated by the dependent region controller after the region is started by JCL.

Database
A collection of data that is fundamental to the user's activity. Using a Program Communications Block (PCB), a program has a logical view of the database, as described by the IMS physical database design.

DBCTL security choices made during system definition

You make IMS security choices with initialization EXEC parameters or in the IMSGEN macro. You can use the Resource Access Control Facility (RACF) to implement security decisions.

Use the specifications in the macro or initialization EXEC parameters to choose the type of security to be active in online execution. Use the Resource Access Control Facility (RACF) to name the resources.

Security facilities and security types for DBCTL resources

The security facility that you use to protect a particular DBCTL resource depends on the type of security that you want and the available security facilities for that type of security. Security facilities for DBCTL resources include z/OS, RACF, and PSBGEN.

The following table summarizes the types of security and facilities that are available for DBCTL resources.

Table 32. DBCTL resources, security types, and security facilities

Resources	Type of security	Security facility
System data set	OS password protection	z/OS
	Data set protection (VSAM)	RACF
Database	Segment sensitivity	PSBGEN
	Field sensitivity	PSBGEN
	Password security (for the **/LOCK** and **/UNLOCK** commands).	RACF See note [1]
PSB	resource access security	RACF
	APSB security	RACF
BMP application program	Password security (for the **/LOCK** and **/UNLOCK** commands)	RACF
	Extended resource protection (using APPL keyword)	RACF
Control region	Extended resource protection (using APPL resource class)	RACF

Table 32. DBCTL resources, security types, and security facilities (continued)

Resources	Type of security	Security facility
Dependent region	RAS security	RACF
	APSB security	RACF
	resource access security	RACF

Note:

1. RACF authority for the **`/LOCK PGM`** and **`/LOCK DB`** commands is not checked in a DBCTL-only system (SYSTEM parameter in the IMSCTRL macro during the system definition process contains DBCTL or IMS is started with the DBC procedure).

Design considerations for DBCTL security

This topic explains how the various choices of IMS security can be used. When you are deciding on each part of your security design, consider the physical actions that an end user must take to obtain access to the system. You will probably use more than one type of security checking.

This topic assumes:

- A user identification as a control point
- The master terminal as a control point
- The use of RACF protection
- The use of a region as a control point

Using password protection with command keywords

To provide verification before a command is accepted, you can require an accompanying password. The password is entered within parentheses immediately following the command verb.

Limiting access from a dependent BMP, JBP, or CCTL region

Dependent BMP, JBP, and CCTL regions are resources you should protect. You can protect them by preventing the start of an unauthorized dependent region by start of task JCL and by preventing the use of unauthorized resources in a dependent region.

Securing DBCTL dependent regions using RAS

RAS restricts the use of resources based on the user IDs of dependent regions. RAS uses RACF for security enforcement.

RAS allows an application program to access a PSB if the user ID of the dependent region in which the application program is running is authorized for that PSB. The authority of a user ID to use a PSB is defined in either the IIMS or JIMS RACF security class. RAS does not restrict the application programs that can be scheduled in a dependent region.

The IIMS and JIMS classes are predefined by RACF; however, if they have not been included with your release of RACF, you can use the RACF ICHERCDE macro to define them as new classes.

You can specify RAS by specifying ISIS=R on the initialization EXEC parameter.

Related reading:

- For more information about the ISIS= EXEC parameter, see *IMS Version 15.2 System Definition*.
- For more information about working with RACF security, see:
  - "Preparing a RACF security plan" on page 322
  - *z/OS Security Server RACF Security Administrator's Guide*

- For more information about the ICHERCDE macro, see *z/OS Security Server RACF Macros and Interfaces*.
- For more information about updating the RACF resource class descriptor table, see *z/OS Security Server RACF System Programmer's Guide*.

Security considerations for Fast Path application programs

When designing security protection for Fast Path application programs, or for DL/I programs that access Fast Path databases, consider that the processing in a dependent region can be protected by RAS security.

Security for ODBA application programs

You can secure any PSB specified on an APSB call from an ODBA application program using the z/OS System Authorization Facility (SAF) and an external security product, such as RACF.

For more information about security for ODBA application programs, see "Security for ODBA application programs" on page 312.

Security for ODBM allocate PSB (APSB) requests

Any PSB that is specified on an APSB request from an ODBM thread can be secured by using the z/OS System Authorization Facility (SAF) and/or the IMS RAS user exit.

For more information about security for ODBM allocate PSB (APSB) requests, see Security for ODBA application programs.

Activating IMS DBCTL security

This topic gives guidance on the steps you take to activate your IMS security design using RACF and program exit routines.

Depending upon the security facilities you choose to use, you must perform the appropriate tasks.

- Prepare a RACF security plan in DBCTL.

  To implement a RACF security plan:

  1. Prepare a list of all the of IMS online resources to be protected, arranging them in groups to give an overview of the total resources covered.
  2. Select the security facilities that protect the resource groups.
  3. Code the IMSGEN macro or initialization EXEC parameters.
  4. Describe the resource class profiles to RACF.
  5. Add users, groups, and data sets to RACF.
  6. Modify JCL procedures in IMS.PROCLIB.

  RACF resource classes are used by the IMS security function. The PSB class holds profiles for PSB security. RACF provides predefined resource classes or you can define your own. The names of resource classes to be used are specified with the RCLASS= parameter. Table 33 on page 337 shows resource class assignments for DBCTL.

Table 33. Resource class assignments for DBCTL

	Resource class naming convention	
Resource class	**RACF-defined name**	**User-defined name**
PSB resource class	IIMS	I*xxxxxxx*
PSB group resource class	JIMS	J*xxxxxxx*

Table 33. Resource class assignments for DBCTL (continued)

	Resource class naming convention	
Resource class	**RACF-defined name**	**User-defined name**
APSB resource class	AIMS	A*xxxxxxx*

The RACF resource classes are defined in RACF's resource class descriptor table (CDT). Initially, the AIMS, IIMS, and JIMS resource classes are predefined in the CDT. To add a resource class or to define resource classes with user-defined names, you must use the RACF resource class macro ICHERCDE to create an installation-defined CDT.

Related reading: For more information about the RCLASS EXEC parameter, see *IMS Version 15.2 System Definition*.

Controlling system startup for DBCTL security

The values generated for the DBC procedure all specify no security. You must reset them to enable security.

The ISIS execution parameter for the control region includes a way to control the kind of security checking that is done during the current execution. The parameter determines what flexibility the DBCTL operator must override the choice of security checking. You must coordinate the setting of the ISIS= keyword parameter with both overall security design and operational procedures.

The ISIS keyword parameter allows you to choose the type of dependent region security you want (RAS) and which security facilities, RACF or exit routines, will implement that security. You specify your choice using the ISIS= parameter. Your parameter options for ISIS= are as follows:

ISIS=0|N
Disables RAS security.

ISIS=1
Specifies RAS security using RACF. Specifying ISIS=1 is the same as specifying ISIS=N.

ISIS=2
Specifies RAS security using an exit routine. Specifying ISIS=1 is the same as specifying ISIS=N.

ISIS=R
Specifies RAS using RACF.

ISIS=C
Specifies RAS using an exit routine.

ISIS=A
Specifies RAS using both RACF and an exit routine.

If RACF is to be used for a CCTL, the CCTL JOB statement must include the USER specification. The CCTL itself can provide security checking (RACF or its own) independent of the DBCTL security checking that is described in this section.

Implementing DBCTL security changes online

If you use online change for system definition, you must update the security definitions for RACF.

As a result of online changes to the system definition, you might need to take the following steps to keep your security specifications current:

- Update passwords.
- Add security provisions to the online system for password security.
- Refresh the RACF database.

To change RACF security definitions online, update the RACF database. Then issue the RACF command **SETROPTS RACLIST(***classname***) REFRESH**.

Controlling DBCTL security violations

Security violations are handled according to the installation's security administration guidelines.

IMS records the following security violation attempts on the IMS system log:

- Password omitted when one is required
- Password incorrect for authorization
- Misspelled password

Any of these errors causes IMS to log the violation. The IMS system log provides an audit trail for investigation of possible security problems. The IMS system log security violation is identified as a X'10' log record type. You can use the File Select and Formatting Print utility to print the log.

Another method for recording security violations is available when RACF is installed. Each resource access violation creates a RACF type 80 record.

Related reading:

- For more information on the File Select and Formatting Print utility, see *IMS Version 15.2 System Utilities*.
- For more information on utilities to format and print RACF records, see *z/OS Security Server RACF Auditor's Guide*.

Restricting database updates

You can restrict certain users from making DLET, ISRT, and REPL calls to databases that enable the Restricted Update mode and set the RREPL attribute to ABEND or STATUS.

You can restrict certain users to make DLET, ISRT, and REPL calls to databases that have Restricted Update mode enabled (RREPL attribute is set to ABEND or STATUS).

The facility class DFS.DB.RREPL.* must be defined, and users must be permitted to make database changes with UPDATE authority.

If an unauthorized user makes an update call to a database that has Restricted Update mode enabled, it results in a 3303 abend or the RR status code depending on the RREPL value.

The RREPL value at the area and partition level will take precedence over the value on the database level.

Updates are allowed from Fast Path utilities (Create, Compare, Scan, Delete, and Reorg) and full-function database utilities (OLR and OIC) regardless of the RREPL setting on the database.

The following sample RACF command defines the facility class and grants authorization to USRT002 and USRT005:

```
RDEF FACILITY DFS.DB.RREPL.* UACC(NONE)
PERMIT DFS.DB.RREPL.* CLASS(FACILITY) ID(USRT002, USRT005) ACCESS(UPDATE)
```

Chapter 20. Controlling IMS

Controlling IMS consists of many tasks.

These tasks are discussed in the following topics.

Related reading: For more detailed information, see *IMS Version 15.2 Operations and Automation*.

Monitoring the system

Monitor the status of the system on a regular schedule to gather problem determination and performance information.

For example, to determine if you should start an extra message region, you might monitor the status of the queues during peak load.

Related reading: For more information on monitoring, see Chapter 24, "Collecting and interpreting IMS monitoring data," on page 357.

Processing IMS system log information for analysis

The system log data sets are a basic source for statistics about the processing performed by the online system. Individual log record types contain data that can be analyzed in many ways.

For example, you can select and format all activity pertaining to a specified user ID or about IMS pools.

Using IMS system log utilities

IMS provides several utilities to assist with extracting log records from the system log. These utilities also assist with reducing and merging data that is spread across several log data sets.

The sections that follow describe several of these utilities:

- **File Select and Formatting Print utility**

  You can use the File Select and Formatting Print utility (DFSERA10) if you want to examine message segments or database change activity in detail. This utility prints the contents of log records contained in the OLDS, SLDS, or the CQS log stream. Each log record is presented as one or more segments of 32 bytes. The printed output gives each segment in both character and hexadecimal formats.

- **Fast Path Log Analysis utility**

  Use the Fast Path Log Analysis utility (DBFULTA0) to prepare statistical reports for Fast Path, based on data recorded on the IMS system log.

  These reports are useful for system installation, tuning, and troubleshooting. The Fast Path Log Analysis utility is not related to the IMS Monitor or the Log Transaction Analysis utility. For more information about the IMS Monitor, see "IMS Monitor" on page 368.

- **Log Transaction Analysis utility**

  In an IMS DB/DC or DCCTL environment, you can use the Log Transaction Analysis utility (DFSILTA0) to collect information about individual transactions, based on records in the system log. Many events are tabulated in the Log Analysis report that is produced by this utility, including total response time, time on the input queue, processing time, and time on the output queue.

- **Statistical Analysis utility**

  In an IMS DB/DC or DCCTL environment, you can produce several summary reports using the IMS Statistical Analysis utility (DFSISTS0). You can use these reports to obtain actual transaction loads and response times for the system. The statistics produced are dependent on the input system log data sets.

IMS database productivity tools

IBM offers a number of IMS database productivity tools. IMS Tools is a set of database performance enhancements for your IMS environment.

These tools can help you automate and speed up your IMS utility operations. They can also assist you in analyzing, managing, recovering, and repairing your IMS databases.

Related reading: To learn more about these tools, visit the DB2 and IMS Tool web site at IMS Tools.

Recovering the system

Ensure that resource definitions have been exported to a resource definition data set (RDDS) if you use DRD and if you perform a cold start from an RDDS.

You must ensure that any resource changes made dynamically are recovered across a cold start.

While IMS is running, an IMS system programmer or operator might need to complete tasks to recover the system.

Related concepts
Overview of dynamic resource definition (System Definition)
Related tasks
Executing recovery-related functions (Operations and Automation)

Modifying and controlling system resources

You establish the initial settings of IMS resources as part of the IMS system definition process. The MTO, and other operators authorized to do so, can change various system resources using IMS commands.

You can modify and control the operating state of the following IMS system resources using type-1 commands:

- Dependent regions
- Telecommunication lines
- Terminals
- Transactions
- Databases
- ISC users (subpools)
- ETO users
- MSC resources
- Security options
- Conversations
- Subsystems

You can use many IMS commands to perform similar control functions for different types of resources.

The resources are:

- Telecommunication line, physical terminal, or node resources
- Logical terminal resources
- Logical link resources
- Logical link path resources
- Transaction resources
- Transaction class
- Program resources

- Database resources
- Subsystem resources

The *IMS Version 15.2 Operations and Automation* show the relationship between these commands and resources, and provide answers to a series of specific questions. For example, after a command is issued, can a resource:

- Receive input?
- Send output?
- Perform output message queuing?

You can also modify and control IMS resources using type-2 commands in:

- Single-IMS IMSplexes that are configured with a minimal Common Service Layer that include an Operations Manager (OM) and a Structured Call Interface (SCI), but no Resource Manager (RM). This configuration is also known as an enhanced command environment.
- Multiple-IMS IMSplexes that are configured with a full CSL (OM, SCI, and RM).

Related reading:

- For details about the system definition processes (batch and dynamic) and details about configuring a single IMS with a minimal CSL, see *IMS Version 15.2 System Definition*.
- For details about the IMS commands, see *IMS Version 15.2 Commands, Volume 1: IMS Commands A-M* and *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Controlling data sharing

Controlling data sharing involves using DBRC and monitoring the data-sharing system.

Controlling data sharing using DBRC

DBRC allows you to control access to data by IMS subsystems that participate in data sharing. Using DBRC, you can modify, initiate, and delete the current status indicators in the RECON data set to change the access intent of online IMS subsystems and the share level of registered databases.

Your data sharing environment depends on the status of the databases and subsystems as indicated in the RECON data set.

Related reading: For more information about how DBRC participates in data sharing, see Chapter 36, “Supporting data sharing,” on page 489.

You can modify the share level indicator using a form of one of the DBRC online change commands, **/ RMCHANGE**.

Monitoring the data-sharing system

To monitor data sharing, you obtain information on the status of the IRLM, IMS subsystems and databases, the RECON data sets, and, for data sharing in an IMSplex, coupling facility structures that are participating in data sharing.

To:

- Display the status of an IRLM on either your system or on another connected system, enter the following z/OS command:

  ```
  MODIFY irlmproc,STATUS,irlmx
  ```

  Related reading: For a complete description of the commands for the IRLM, see *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*.
- Monitor components and resources in a data-sharing environment, use the same types of procedures as in a non-data-sharing environment. For more information about monitoring, see Chapter 24, “Collecting and interpreting IMS monitoring data,” on page 357.

- Monitor structures on a coupling facility, use following z/OS operator commands:
  - **`DISPLAY XCF,STRUCTURE`**: Used to display the status of structures defined in your active policy.
  - **`DISPLAY XCF,STRUCTURE,STRNAME=`**: Used to display detailed information about a specific structure. The structures that are important to IMS are the IRLM, OSAM, VSAM, shared MSGQ, and shared EMHQ structures.

  These commands let you look at structures on a coupling facility to determine resource status and, for failures, gather information for problem determination.

  Related reading: For detailed information about these commands, see *z/OS MVS System Commands*.

Controlling log data set characteristics

From time to time, you need to tune and modify log data set characteristics.

For example, in the following circumstances you should tune and modify log data set characteristics:

- After monitoring
- After changing your requirements for system availability, integrity, or operator handling

For detailed information about controlling the OLDS, WADS, SLDS and RECON data sets, see *IMS Version 15.2 Operations and Automation*.

Connecting and disconnecting subsystems

Before an IMS subsystem can access external subsystems (another program executing in an z/OS address space) such as DB2, you must connect the IMS subsystem to this other subsystem. IMS can connect to another subsystem only if that subsystem is identified in the subsystem member in IMS.PROCLIB.

Specify the subsystem member name in the SSM EXEC parameter. When specified to and accessed by IMS, the subsystem member name cannot be changed without stopping IMS.

Related reading: For information on the SSM EXEC parameter, see *IMS Version 15.2 System Definition*. For more information about connecting IMS to another subsystem, see *IMS Version 15.2 Operations and Automation*.

Chapter 21. Starting or restarting IMS

Starting or restarting an IMS subsystem means initializing the IMS control region.

To restore the operating environment, the MTO must ensure the following are also initialized:

- All IRLMs
- All CQSs
- Any dependent regions (including utilities)
- All CSL address spaces (OM, RM, and SCI), if operating as an IMSplex using CSL
- Any connections to VTAM
- All communication lines and terminals

The term MTO used in this information refers to the IMS master terminal operator for the DB/DC and DCCTL environments, and to the operator for the DBCTL environment. All commands shown assume that the command-recognition character is the forward slash (/).

See *IMS Version 15.2 Operations and Automation* for additional information.

Chapter 22. Shutting down IMS

The command that is used to shut down the control region also forces termination of data communications and the dependent regions if they have not already been terminated in an orderly way, and can also tell the CQS subsystem to shut down.

A common sequence for shutting down the entire online system is:

1. For an IMS DB/DC or DCCTL environment, stop data communications. For an IMS DBCTL environment, disconnect from the CCTL.
2. Stop dependent regions.
3. If dynamic resource definition (DRD) is enabled and AUTOEXPORT is not, export any modified runtime resource and descriptor definitions that have not yet been exported.
4. Stop the control region.
5. For an IMS DB/DC or DBCTL environment, stop the IRLM.
6. For a shared-queues environment, if CQS has not shut down, shut it down.
7. For an IMSplex system using CSL, shut down the CSL manager address spaces (OM, RM, and SCI).

Related tasks
Shutting down IMS (Operations and Automation)

Chapter 23. Testing the system

As a system administrator, you must be involved in two phases of testing. One phase occurs when application programs are verified prior to a cut over to production mode. The second phase involves evaluating changes or corrections to existing application programs to ensure that application function has not regressed and to validate the new or changed function.

When you have multiple application programs, you must ensure that modifications to one application program do not impact the service provided to any end users.

The various testing phases, with the corresponding administration tasks and related development activity, are listed in the following table.

Table 34. Administration tasks related to testing phases

Test phase	Administration task	Related development activity
Unit test	Identify bottlenecks	Test development code
Function test	Plan operations procedures	Test explicit functions
Integration test	Prepare test system definition	Build the system
	Plan test database	Build test database
Component test	Verify network operation	Validate major portions of application program logic
System test	Check out operations and recovery procedures	Build fully executable system
	Coordinate use of test tools and monitoring	Validate operational procedures
	Ensure network readiness	Test coexistence
Performance testing	Plan for simulation	Validate claims and set benchmarks
Stress testing	Plan for peak loads and response criteria	Test for high volume of traffic and processing
Acceptance test	Finalize operations procedures	User liaison checkout on behalf of end users
Maintenance testing	Control libraries and online definition	Application and IMS service checkout
Design change testing	Plan response across administration tasks	Control verification of changes
Regression testing	Plan monitoring	Verify that old function is not damaged

The need for a test system

The development of a test system requires the participation of representatives from several areas.

Your primary concern at the final stage of implementation is to offer satisfactory service to the end users. Does the IMS system perform as expected? A secondary concern, when changes occur in one or more online application programs, is to preserve the integrity and service for all end users. Will changes seriously impact the production environment?

Your installation might have a separate test organization. In this case, development personnel do not test the working code but might have to demonstrate it before handing over the application programs for independent testing.

The end user might be represented by a user-liaison group that knows the business needs. The liaison group also might evaluate the adequacy and accuracy of application programs.

Your solution to the need for a testing system can take several forms:

- A separate IMS test environment that is operational during major development activity
- A separate IMS test system used for ongoing verification of maintenance and application design changes
- A production system that allows controlled changes to be tested online, possibly using regions that are initialized for test purposes at a time that does not impact production processing

Part of your administration role is to ensure that a suitable online IMS test system is created and that the test procedures and any special database requirements are well documented. This is important if the test system is also to function as the maintenance system. However, your role can be more passive, in which you only participate in testing to verify operations procedures and to verify that all system definitions and preparation for production mode are in place.

After the application package has entered the production phase, the role of the developer or tester is usually taken over by program maintenance personnel. Administration needs to assess the impact of maintenance to the existing online IMS design. This task is described in more detail in Chapter 25, “Modifying the system design,” on page 385.

Setting up a test system

An administration task is to define the procedures for applying changes to the online IMS system. Part of this task is to have an acceptance procedure that is followed by all responsible participants.

The major considerations for establishing a separate test system are:

- The installation's policy for protecting the integrity of the database
- The presence of criteria for accepting an application as properly tested
- The need for a stable production environment
- The degree of support available to solve day-to-day problems
- The complexity of the applications
- The degree to which the current programs can perform correctly, or function with interim problem bypasses.

Some of the factors involved in the decision to use a test system as an ongoing function are:

- Will there be staged implementation of new application programs or add-on function?
- Can online service interruptions caused by the need to fix application problems encountered during testing be tolerated and, if so, to what degree?
- What procedures should be followed for accepting system changes or corrections?
- How is maintenance to the IMS system itself to be handled—will the test system be used?
- Will corrections to the application programs and IMS problem fixes be incorporated on demand or batched at agreed-upon intervals?

Setting up a test database

A test database must contain enough data to adequately test the system. The data should exercise a major portion of an application program's logic to prevent regression of existing function and yet economize on the amount of data.

Usually, a test transaction stream executes against a known database status, and the results are compared to predicted results. The content of the transactions and implications of the database status needs to be analyzed for accuracy by a user-liaison, group as well as by groups responsible for the database design. As additional test cases are added, they must be validated and inspected for redundancy. The trade-off is between adding to an existing transaction or defining an additional transaction that might require extra database content. The test case stream must be adequately documented, so that you know what it does and what data it needs.

Related reading: For more information on creating test databases, see *IMS Version 15.2 Database Administration*.

Testing operational procedures

You can use the test system to ensure the accuracy and usability of your operational procedures.

The following activities are suitable during the system test phase:

- You can test preliminary versions of the MTO procedures, the run book, and the incident report forms. The MTO can perform system startup, connect a subset of the terminals and nodes, and practice the restart actions.
- You can hold an informal audit of command use and discuss any misunderstandings to help you refine the content of the operator instructions.
- During online execution, you can arrange other events that require MTO intervention to take place, such as taking image copies, invoking the IMS Monitor and traces, and responding to application program abnormal termination.
- You should also test the recovery procedures. Have the operations staff carry out a database recovery and verify the correct use of system logging control and recovery utilities. You can build a test RECON data set for DBRC and follow through the GENJCL step to recover to a given checkpoint position.

If the test system is used for checkout of changes to the application package, any significant operational changes should also be exercised, possibly with MTO observers.

Monitoring in IMS test environments

Your objectives for monitoring during a test phase are slightly different from those for production systems

Three activities you should plan for are:

- Detecting and correcting potential performance problems before entry into production mode.

  Using realistic data, you can expose patterns of processing for the application programs, especially the DL/I call occurrences. Compare the Monitor results against the expected transaction profile. Such items as excessive I/O events or large I/O wait times can reveal a performance problem at an early stage.
- Testing the monitoring part of your operations procedures.

  Become familiar with the way output is produced and the format and content of reports. Start to develop work sheets to summarize Monitor findings rather than write unstructured comments on the report output.

  Related reading: See Chapter 24, “Collecting and interpreting IMS monitoring data,” on page 357 for information about the detailed report descriptions and considerations and tools for your monitoring strategy.
- Using your monitoring tools for base profiles of new program processing.

  Discuss your findings with development personnel and with performance specialists. Try to obtain early warning of performance problems. If you are integrating a new application into an existing online IMS system, try to assess beforehand the impact of the added application workload.

During the testing stage you can also perform stand-alone or calibration runs to establish base profiles for the critical transactions. The Call Summary report from the IMS Monitor is useful for this purpose. You can also compare the results for the tested transactions to similar transactions already in the production system.

Monitoring in a DB/DC environment

In a DB/DC environment, you can obtain information about the precise sequence of calls issued by application programs using the report capabilities of IMSASAP II.

This tool requires using the online IMS Monitor to produce the monitor trace records. Its use as a monitoring tool and installation prerequisites are described in Chapter 24, "Collecting and interpreting IMS monitoring data," on page 357.

Ensuring network readiness

When you prepare an application package for production mode, you must be aware of the status of all terminals or connected devices planned for availability with the online IMS system.

- Develop a detailed implementation plan for each device and control unit.

  Some of the items in this document might be:

  - Exact type of device, model, and operational characteristics
  - The configuration for a set of components
  - Physical location and person responsible for the device installation
  - The names by which the online IMS system knows the device: LTERM, line number, unit address, and node names
  - The correct VTAM MODEENT macro's PSERVIC parameters for: LUNAME, LUTYPE, TS Profile, 32xx model, 3270 screen size, and NTO device type
  - The logon, user, and MSC descriptors for terminals and users defined with ETO
  - The appropriate exit routines
  - If an output device, the source of paper supplies and arrangements for distribution of output
  - If already in operation outside of IMS, the restrictions on the use of the device for test purposes
  - System definition requirements for the device, and the planned timing of the generation
  - If the device is programmable, what programs are necessary, locally and in the host, for IMS execution
- Use the terminal profiles built up as part of the preparation for system and network definition.

  You need this information because of the potentially large number of devices that can operate in the online IMS system. Also, you must converse in terms of the hardware with installation personnel and data communications specialists whose responsibilities usually lie beyond IMS device support.
- Contact the person responsible for the system definition regarding the status of statically defined VTAM terminals.

  The definition events are usually scheduled as part of a master plan for implementing the application package.

Network testing in DB/DC and DCCTL environments

As part of your preparation for production mode, you should arrange for online testing of each piece of the network during the system test. Try to observe the online testing for the different terminal types that are to be active in the IMS system.

Include observations of those transactions considered critical. The results of this exercise are useful when you write operations procedures for remote terminals and instructions for the master terminal operator. Emphasizing the physical actions and sequence of events helps clarify these procedures and make them more complete for the end user.

If you do not have access to an early draft of the actual procedure for the terminal operation, it is best to use a prepared script for a session. The text should include a scenario depicting a test sequence of commands as an online session:

1. Prepare to transmit the SNA terminal subsystem program, then transmit from the host (if applicable).

2. Start the ACF/VTAM or host subsystem application program, and enter the appropriate commands to start IMS and establish communication with VTAM.
3. Start the SNA terminal subsystem program at the processing unit location (if applicable) and make the intelligent terminal ready for communication with IMS.
4. Exchange messages with the host, submit an IMS transaction from the terminal, and validate the result.

Testing in a DBCTL environment

If you are testing a DBCTL environment with a CCTL, be aware that the CCTL, not IMS, controls the network, terminals, and transactions.

Testing in a DCCTL environment

You can test any DCCTL function that does not require access to a database.

For example, you can schedule transactions or perform component testing if no database calls are made. If your test system already has a GSAM or an external subsystem (for example, Db2 for z/OS) installed, you can verify connections and applications requiring those systems.

IMS testing aids

The following topics provide information about various testing aids.

Related reading: For more information on suggested procedures and utilities used during testing of application program code, see "Testing an IMS application program" in *IMS Version 15.2 Application Programming*.

Simulating online execution with Batch Terminal Simulator in DB/DC and DCCTL environments

Often an online application program is developed to the testing stage, but a suitable online IMS system is not available. At this stage you can use the Batch Terminal Simulator licensed program, program number 5655-J57.

The Batch Terminal Simulator (BTS) program can simulate the operation of message processing regions before the online IMS system is operational as a test system. This program runs as a batch IMS system using one or more applications. BTS input is transaction data, and the application programs are invoked. Database calls are executed against test databases; the DL/I calls for data communications are simulated.

With BTS, you can do the following:

- Print terminal input, output, and an optional trace of related database activity. You can request a summary of DL/I calls, by type, against each PCB.
- Simulate conversational transactions and program-to-program switches.
- Simulate the message queuing and application program scheduling functions of the online IMS system. transaction input data to the input message structure for the application program and, similarly, map the output messages to a printed layout. This is particularly useful for IMS 3270 input and output formats.
- Use the debugging and trace facilities (during the test phase) for both batch and telecommunication applications.

You do not need to modify the IMS control programs, control blocks, libraries, or the application programs.

Related reading: For more information on BTS, see *IMS Batch Terminal Simulator for z/OS User's Guide*.

Online testing of MFS formats in DB/DC and DCCTL environments

If you need to test an application program change online in either a test or a production system and the test transaction input uses an MFS-supported screen format, you can use the MFSTEST mode.

IMS allows individual MFS-supported terminals to enter MFSTEST mode. In this mode, you can use temporary message formats from an alternative MFS format library, rather than changing the message format blocks in the production format library. A format block name can be identical to the name of a message format already in the production library.

An MFS-supported terminal is placed in test mode by entering a **/TEST** command with the MFS parameter. Next, the remote terminal operator enters a **/FORMAT** command for transaction processing, causing message formats to be selected from a library of test formats (IMS.TFORMAT). If these message formats are not found in this library, they are selected from the active message format library (IMS.FORMATA/B). By entering an **/END** command, the terminal is removed from test mode.

When testing MFS formats, you can use the **/TRACE SET ON TRAP** command to trap and analyze MFS errors. This command helps you analyze errors that result from incorrect manipulation of the MFS control blocks.

System definition requirements for MFSTEST mode

To use MFSTEST formats, specify the appropriate parameter on either the IMSGEN or COMM macro.

Related reading: For more information on the IMSGEN and COMM macros, see *IMS Version 15.2 System Definition*.

Online execution requirements for MFSTEST mode

When you want to perform testing of MFS formats during execution of the online IMS production system, the IMS.TFORMAT library must be defined and must contain the appropriate test blocks.

These can be named the same as existing production formats. The IMSTFMTA/B DD statements must be present in the control region JCL, with the IMS.TFORMAT library as the first data set, followed by the corresponding IMS.FORMATA/B library concatenated to it.

The CIOP is used to hold all format blocks for terminals operating in MFSTEST mode. Storage space in CIOP is dynamically allocated according to your system demands.

Executing in MFSTEST mode does result in a performance impact, because both transmission and message formatting are being stored in the same buffer.

Using dynamic resource definition and online change for testing

Using dynamic resource definition and online change capabilities, you can plan to perform selective testing during execution of an IMS online test system. Your testing can be accomplished without generating a modified test system merely to accommodate some temporary changes.

You can perform testing that supports:

- The addition of new applications
  - New transactions, programs, and MFS formats
  - Additional databases
- Modifications to existing applications
  - Modified database definitions
  - Replaced PSBs and programs
  - Altered MFS formats
  - Necessary deletions for the above modifications

You cannot use this method to test the use of terminals or devices not already defined to the IMS system.

Another factor might be the security arrangements. Planning for change must ensure that appropriate authorization exists to use the transactions or existing terminals for the duration of the test.

You might plan for a group of application changes to make up a test package to be executed during IMS online operation in a test system, or even during a production cycle. Online change is best used to migrate pretested changes without the need for a full restart interruption. Your planning should include the following considerations:

- Online data resources must be adequately protected. Carefully assess the risk that an application program being tested could damage production data.
- Added databases must be included in the JCL for the IMS control region; a solution for z/OS systems is to use dynamic allocation.
- Although changes to database structures can be reflected in a new ACB build, this might not always be practical if the changes require a reorganization.
- You must coordinate the update of IMS.PGMLIB with the status of the online processing. A replaced program might cause problems, unless the end user is aware of the activity. The library update might have to be carried out in the interval just before the **`/MODIFY COMMIT`** command.
- You must assess the potential performance impact to tuned systems, although a monitored test execution can provide valuable information for performance planning.

The entry of the **`/MODIFY PREPARE`** command begins the cut-over operation. When activity has stabilized for the resources affected by the content of the online change, the **`/MODIFY COMMIT`** command completes the cut-over. Testing can now begin. When you complete the planned testing, repeat the **`/MODIFY PREPARE`**, **`/MODIFY COMMIT`** command sequence to reinstate the old unmodified system data sets that were inactive during the testing. It is the responsibility of those performing the test to nullify the effects of the testing and back out any inappropriate changes made to the databases.

Program testing using SYSIN/SYSOUT in DB/DC and DCCTL environments

One way of testing a message processing program is to provide an input data stream containing messages.

Specify a line group as a READER, then assign the input SYSIN to a local card reader. Messages are passed to the application program by GU calls to the message queue. No editing or logging of position occurs while messages are being processed. The accuracy of the end-of-message or segmentation is determined by the input stream.

Similarly, for output, you can assign a line group as a printer, punch, tape, or DASD device using the UNITYPE keyword for the LINEGRP macro. You can then assign the output LTERM for the appropriate device characteristics, although the actual device does not have to be allocated.

The LINEGRP macro parameters produce a DD statement for the control region, and the appropriate buffer size is defined for the output records. For printer output, a translation to the 48-character set, lowercase to uppercase, occurs; all other codes are converted to periods.

Network testing using Teleprocessing Network Simulator in DB/DC and DCCTL environments

Teleprocessing Network Simulator (TPNS), program number 5662-262, is a terminal and network simulation tool used for determining system performance and response times, evaluating teleprocessing network design, functional testing, and automating regression test procedures.

Performance and stress testing using the Queue Control Facility

Queue Control Facility (QCF) can be used in DB/DC and DCCTL environments to select messages from actual workload cases saved in archived log data sets. You can re-queue the messages at selected rates to perform stress and performance testing.

Related reading: For more information on the Queue Control Facility, see *IMS Queue Control Facility for z/OS User's Guide*.

Chapter 24. Collecting and interpreting IMS monitoring data

Monitoring is the collection and interpretation of IMS data. This data helps you understand the daily needs of the system and provides an insight into areas that might need changes.

Monitoring should be an ongoing task because:

- Monitoring helps you establish base profiles, workload statistics, and data for capacity planning and prediction.
- Monitoring gives early warning and comparative data to help you prevent performance problems.
- Monitoring validates tuning you have done in response to a performance problem and ascertains the effectiveness of that tuning.

For multi-system networks, plan to obtain both statistical and performance data for IMS online systems that are part of the network. You can use the same monitoring tools that are used for generating performance data for single IMS systems:

- The IMS Monitor can be executed concurrently in several systems. You obtain IMS Monitor reports for each individual IMS system and coordinate your processing analysis.

  Tip: The IMS Monitor user exit provides a framework and exit point for access to the IMS Monitor data in real-time.
- For DB/DC and DCCTL systems:
  - The IMS Statistical Analysis utility (DFSISTS0) produces summaries of transaction traffic for each individual system. Again, you combine the statistics for a composite picture.
  - The IMS Log Transaction Analysis utility (DFSILTA0) enables you to trace transactions across multiple systems and examine the traffic using the various active physical links.

You can use IBM z/OS Workload Interaction Correlator to gather workload data for IMS systems and other participating z/OS components and middleware to diagnose performance issues across the stack. IBM z/OS Workload Interaction Correlator generates synchronized, standardized, and summarized data every 5 seconds across the z/OS stack, including IMS, by grouping similar jobs together in a common software context.

Overall, conclusions from continuous monitoring and historical records provide a good starting place from which to answer end-user complaints and to provide initial directions for tuning projects.

Related reference
IMS Monitor user exit (IMSMON) (Exit Routines)

Establishing monitoring procedures

Several types of monitoring strategies are available.

You can:

- Summarize actual workload for the entire online execution. This can include both continuous and periodic tracking. You can track total workload or selected representative transactions.
- Take sample snapshots at peak loads and under normal conditions. It is always useful to monitor the peak periods for two reasons:
  - Bottlenecks and response time problems are more pronounced at peak volumes.
  - The current peak load is a good indicator of what the future average will be like.
- Monitor critical transactions or programs that have documented performance criteria.

- Use the z/OS Workload Manager to help manage workload distribution, balance workloads, and distribute resources.
- Use IBM z/OS Workload Interaction Correlator to gain unified workload performance data for IMS systems and other participating z/OS components and middleware across the z/OS stack.

Plan your monitoring procedures in advance. A procedure should explain the tools to be used, the analysis techniques to be used, the operational extent of those activities, and how often they are to be performed.

Regardless of which strategy you use, you need to:

- Develop performance criteria
- Develop a master plan for monitoring, data gathering, and analysis

Establishing performance objectives

Inherent in the design of your online IMS system are your performance objectives. Establishing performance objectives is a major task requiring data gathering for the IMS workload as a whole.

After defining the workload and estimating the resources required, you must reconcile the desired response with the response you consider to be attainable. Monitor the performance of the system to determine if these objectives are being met. Base performance objectives on:

- Desired, acceptable, and maximum response time
- Average and maximum resource demands (or workload) per transaction
- Predicted and actual transaction volumes

Establishing your performance objectives is an iterative process involving the following activities:

1. Defining user-oriented performance objectives and priorities.

   These derive from the way an end user perceives the service provided by the system. For IMS these objectives state expectations of response time as seen by the end user at the terminal.

   When response time objectives are established, they should not only reflect the time-in-system (the elapsed time from entry of a last input message segment to the first response segment), but also the expected amount of IMS and application program processing. You should consider whether to define your criteria in terms of the average, the 90th percentile, or even worst-case response time. Your choice depends on your installation's audit controls and the nature of the specific transactions.
2. Determining how performance against these objectives is measured and reported to users.

   This includes identification of systematic differences between the measured data and what the user sees. You should investigate the differences between internal (as seen by IMS) and external (as seen by the end user) measures of response time. To do this, you can use the following tools:

   - The Transaction Response report produced by the IMS Statistics Analysis utility, which gives the following data on internal response time by transaction type:
     - Longest and shortest response
     - 25th, 50th, 75th, and 90th percentiles of the response time distribution
   - Installation-written programs, which can analyze the output of the IMS Log Analysis utility. Other programs can also provide the same type of information, and are usually tailored to satisfy the installation's requirements.
3. Understanding and documenting the current workload.

   This requires breaking down the total work into categories and, for each category, developing a workload profile (generally estimates) that include:

   - Definition of the transaction category (for example, transaction type or group of transactions). The two characteristics of the category are:
     - The IMS workload, which is generally documented by transaction profile. In a well-designed IMS online system, most transactions perform a single function and have an identifiable workload

profile. Later in the process, transaction types with common profiles can be amalgamated for convenience.

  - The transaction volume. In situations where the workload to be documented is already operational, a summary of transaction volumes can be obtained from the IMS Statistical Analysis utility. (The Application Accounting Report gives transaction counts within program name.) In other cases, volumes are estimated.

- Relative priority of category, including periods during which priority changes.
- Resource requirements of the work:
  - Physical resources managed by the operating system (real storage, DASD I/O, terminal I/O)
  - Logical resources managed by the subsystem, such as control blocks, latches, buffers, and number of regions

To obtain a base profile of transaction resource demands, you can start IMS on a dedicated machine and execute a few transactions to accomplish initialization and buffer pool usage. Then start the IMS Monitor and measure a sample of transaction execution.

You can use the base transaction profile to examine the transaction workload to see if it can be reduced. Such design changes result in the greatest impact, occurring before system-wide contention. You can also compare the base profile to the transaction profile in the production environment.

4. Translating the resource requirements and volume information obtained into system-oriented objectives for each work category

   This includes statements about the transaction rates to be supported (including any peak periods) and the internal response time profiles to be achieved.
5. Confirming that the system-oriented objectives are reasonable.

   After initializing the system and monitoring its operation, you need to find out if the objectives are reasonable (given the hardware available), based upon the measurements of the workload. If the measurements differ greatly from the estimates used, you must revise the workload documentation and system-oriented objectives accordingly, or tune the system.
6. Establishing performance objectives.

   Establishing performance objectives is also a necessary prerequisite to using the z/OS Workload Manager. Much of the information gathered when you establish performance objectives can be used as input when planning for workload management.

Planning for workload management

z/OS provides a workload management function to help you manage workload distribution, balance workloads, and distribute resources to competing workloads.

z/OS provides this support automatically after you specify in the panel-driven application that the z/OS Workload Manager (WLM) provides how you want your workloads processed.

Workload Manager and IMS

With the z/OS WLM, you define to z/OS the performance goals for transactions and the relative importance of transactions and address spaces. Using your definitions, WLM then decides how the resources that are controlled by z/OS should be allocated.

Performance goals and business importance for transactions

A performance goal can be the average response time desired for a transaction, or it can be that a certain percentage of the transactions complete within the response time period.

The business importance is a priority level representing how critical a type of transaction is to your installation. 1 is the highest priority level.

After a transaction is scheduled by IMS, the z/OS WLM uses the performance goals that you define for each transaction to decide how much resources, such as processor cycles and storage, it should allocate to meet your goals. When contention for system resources occurs, the z/OS WLM uses the business importance assigned to the transaction to help decide which transactions get priority for the resources that are under the control of the z/OS WLM. The performance goals and business importance are defined in a service definition.

Related reading: For more information about service definitions, see “Establishing the service definition” on page 361.

Assigning relative importance to IMS address spaces

After setting performance objectives for IMS transactions, specify in WLM service classes the importance of each IMS address space relative to all other IMS address spaces. Using your specifications, WLM determines the processor dispatching priority for all address spaces in the z/OS system.

In the following figure, the WLM ISPF-driven panel for defining a WLM service class assigns relative importance of 2 to the IMS control region address space.

```
                          MODIFY A SERVICE CLASS               ROW 1 TO 00000100
COMMAND ===> ___________________________________________________________00000200
                                                                        00000300
SERVICE CLASS NAME . . . . . : CNTL                                     00000400
DESCRIPTION  . . . . . . . . . IMS CTL RGN SERVICE CLASS                00000500
WORKLOAD NAME  . . . . . . . . IMS       (NAME OR ?)                    00000600
BASE RESOURCE GROUP  . . . . . ________  (NAME OR ?)                    00000700
CPU CRITICAL . . . . . . . . . YES       (YES OR NO)                    00000800
                                                                        00000900
SPECIFY BASE GOAL INFORMATION.  ACTION CODES: I=INSERT NEW PERIOD,      00001000
E=EDIT PERIOD, D=DELETE PERIOD.                                         00001100
                                                                        00001200
        ---PERIOD---  ---------------------GOAL---------------------    00001300
ACTION  #  DURATION   IMP.  DESCRIPTION                                 00001400
  __                                                                    00001500
  __    1              2    EXECUTION VELOCITY OF 70                    00001600
                                                                        00001700
```

Figure 43. Assigning importance using the WLM ISPF panel Modify A Service Class

The following table suggests how you might assign WLM importance values to achieve a recommended relative prioritization of some of the IMS address spaces.

Table 35. Assigning WLM importance to IMS address spaces

WLM importance	IMS address space
SYSSTC	IRLM, VTAM
1	CTL, DLI, DBRC, ICON, FDBR, CQS, SCI, OM, RM
2	ODBM, MPP, IFP, JMP, BMP, JBP
3 and 4	MPP, IFP, JMP, BMP, JBP, Batch

By prioritizing IMS address spaces as shown in the table, you minimize the risk of critical functions having to wait for dependent regions to finish processing before they receive the processor resources they need. Such critical functions include locking, DB authorization, IMSplex-wide command processing, messaging, and DB open and close activities.

Related reading: For additional information on dispatching priorities, see “Dispatching priorities” on page 399.

Establishing the service definition

A *service definition* contains all of the information necessary to perform workload management processing. Workload management provides an online panel-driven application for establishing a service definition.

Much of the information you need to establish a service definition is contained in the performance objectives described in "Establishing performance objectives" on page 358. An important piece of information contained in the service definition is the classification rule.

A *classification rule* consists of a work qualifier and a service class. One set of classification rules exists for each service definition. However, you can have multiple service classes for each service definition. Using the workload management-supplied function, you establish the classification rule and define the service class by specifying one or more work qualifiers. The *work qualifier* associates incoming transactions with a particular service class, which represents a group of transactions with similar performance criteria (performance goals and business importance, for example). Work qualifiers can be the subsystem type, the IMS transaction name, the IMS transaction class, the inputting LTERM name, or the user ID. You can use any one of these values, or a combination of them, to assign the service class to a transaction. A *service class* has performance criteria defined for it. After a transaction is assigned a service class, the z/OS Workload Manager processes it according to the performance criteria defined for that service class.

In the following example, the classification rule uses the transaction name as the work qualifier. It assigns all incoming transactions with the name of DEPOSIT to the service class IMSHIGH.

- Work qualifier:
  - Subsystem name: IMS
  - Transaction name: DEPOSIT
- Service class:
  - Performance goal: response time of less than a second
  - Business importance: 1

The performance goal for the IMSHIGH service class is a response time of less than 1 second. The business importance of 1 gives the DEPOSIT transactions priority, after they are scheduled by IMS, if contention for system resources occurs.

Recommendation: Specify multiple WLM service classes, differentiating by business importance and goals. Then classify your IMS transactions in the WLM policy by using the IMS transaction class as the work qualifier for the WLM service that best fits the response time goals of each transaction.

By assigning different WLM service classifications to different IMS transactions, you ensure that the WLM manages the z/OS resources in a manner that gives all IMS transactions the best possible chance of obtaining their response time goals

If you assign a single WLM service class to all of your IMS transactions, the WLM treats all IMS address spaces, including the control region, equally. This means that WLM assigns the same dispatch priority to all of these address spaces. If some of your transactions do not make their response time goals, WLM would not know which address space to give the higher priority to (it should be the IMS Control Region) and would not change the priority for any of them.

Migrating to workload management

Workload management offers two operating modes: compatibility and goal. Compatibility mode is your present method of performance management, and goal mode is the workload management method.

To facilitate migration, you can establish your service definition while running in compatibility mode. Then, after you are comfortable with your service definition and have completed the steps for migrating to workload management, you can switch to goal mode.

Related reading: For more information on setting up service definitions and using workload management, see *z/OS MVS Planning: Workload Management*.

Interpreting z/OS WLM Change State PB service codes

The WLM Change State Performance Block (PB) service is used to show the current state of the transaction.

The PB service codes are interpreted as follows for IMS:

State
    Description

Active
    The transaction is executing an application program.

Free
    This is not reported by WLM.

Idle
    The transaction is waiting for work.

Waiting - I/O
    IMS waiting on I/O (IMS initialed I/O).

Waiting - Lock
    IMS waiting on a lock request.

Monitoring mobile workload with WLM

IMS can identify workloads transactions that are processed by a mobile application or originated from a mobile device. Adjusted billing methodology is available to mobile workload transactions.

Defining classification rules for mobile workload

To simplify mobile workload reporting, you can use the *Reporting Attribute* option in classification rules to differentiate between mobile and non-mobile workloads.

The Reporting Attribute option allows one of the following values:

NONE
    Applicable to all work. This is the default.

MOBILE
    Applicable only to mobile work.

CATEGORYA
    Reserved for a first general purpose subset of work. This is provided for future use.

CATEGORYB
    Reserved for a second general purpose subset of work. This is provided for future use.

Define the mobile works as **`MOBILE`** to identify them for separate mobile reporting.

Related concepts

“Establishing the service definition” on page 361
A *service definition* contains all of the information necessary to perform workload management processing. Workload management provides an online panel-driven application for establishing a service definition.

Related information

Measuring mobile workloads with WLM

IMS supports three mechanisms to track mobile workloads with MWP: tracking on LPAR level, subsystem level, and CPU level. Different configurations are required for each tracking mechanism.

You can configure your IMS system to use one of the following mechanisms:

- Use an individual LPAR for mobile-only workloads. All defining programs can report the General Capacity Processor (GCP) CPUs that run on the LPAR as mobile CPUs.

- Build individual subsystems of the defining programs for mobile-only workloads. The defining programs can then report the GCP CPUs that run on the subsystems as mobile CPUs.
- Use the same subsystems for mobile and non-mobile workloads, but create different CPU reports for mobile CPU.

CPU-level tracking

If the mobile workloads are tracked on the CPU level, IMS reports the TCP/IP port number along with the logical terminal (LTERM) override name or the OTMA transaction pipe (TPIPE) name to the Workload Manager (WLM) to differentiate mobile between mobile and non-mobile workloads.

The LTERM override name is set to override the predetermined value in the LTERM field of the IMS application program's I/O PCB in the Open Transaction Manager Access (OTMA). If no LTERM override name is provided, OTMA places instead the predefined TPIPE name in the I/O PCB LTERM field.

Depending on your TCP/IP port configuration, you can differentiate the workload type by one of the following values:

TCP/IP port number
Configure IMS Connect so that each TCP/IP port handles transactions of one workload type exclusively, either mobile workloads or non-mobile workloads.

LTERM override name or TPIPE name
If both mobile and non-mobile transactions arrive on the same port, use the LTERM override name if one exists, or the TPIPE name.

Use *classification rules* to specify which work is mobile work that is eligible for mobile workload pricing. Set the *Reporting Attribute* to **`MOBILE`** to explicitly define what port numbers, LTERM override names, or TPIPE names are associated with mobile transactions. Based on the values that you provided, the IWM4CLSY macro returns a service class.

For more information about setting up classification rules for mobile workload reporting, see “Defining classification rules for mobile workload” on page 362

Examining the workload records

You can examine the workloads running records in the z/OS System Management Facility (SMF) records by running the **`ERBSCAN`** and the **`ERBSHOW`** commands on the extracted SMF records.

You can also consult the X'56FA' log record to determine the workload type. The X'56FA' log record contains statistical information about the transaction workloads. Although the log record does not directly indicate whether the workloads are mobile, but you can determine the workload type from the information such as job name, program name, transaction code, transaction class.

If you are using the WLM panel to classify transactions, you do not need to consult the X'56FA' log record to determine the workload type.

Related concepts
Logical terminals (LTERMs) (Communications and Connections)
Related reference
TCPIP statement (System Definition)

Generating monthly mobile workload reports

You are responsible for collecting and retaining the source data for the mobile workload transactions that are used in monthly reporting. IMS supports two methods of data collection and reporting with the Sub-Capacity Reporting Tool (SCRT).

The mobile workload transaction reports must consist of general-purpose processor CPU seconds for each mobile transaction program summarized by hour by LPAR for all machines that are processing mobile transactions. Mobile workload reporting requires that the mobile CPU consumption reports are submitted each month.

IMS provides two methods of data collection and reporting to identify the CPU time that is used by each MWP Defining Program that is processing mobile workload transactions: with the WLM classification rules or with the original CSV files.

Note: Depending on local legislation and policy, the SMF records and mobile workload transaction data might be required to be retained for a certain period of time after the billing period for auditing purposes.

Generating reports with the WLM classification rules

You can use the WLM classification rules to identify mobile workload transactions. The parameters that are used for WLM mobile pricing support, CPUTIME, TIMEONCP, and OFFLOADONCP, enable WLM to accumulate the mobile workload transaction CPU time in the LPAR during the predefined interval. The CPU time is stored in the SMF type 70 record. The workload activity information for each service class or report class is stored in the SMF type 72 subtype 3 record.

Follow the procedure to generate reports with the WLM classification rules:

1. Track mobile workload transactions by defining new classification rules.
2. Export and load the SMF type 72, type 70, and type 89 records into SCRT each month.
3. Run SCRT Version 23 Release 13.0 or later and submit the results to IBM for each sub-capacity reporting period.

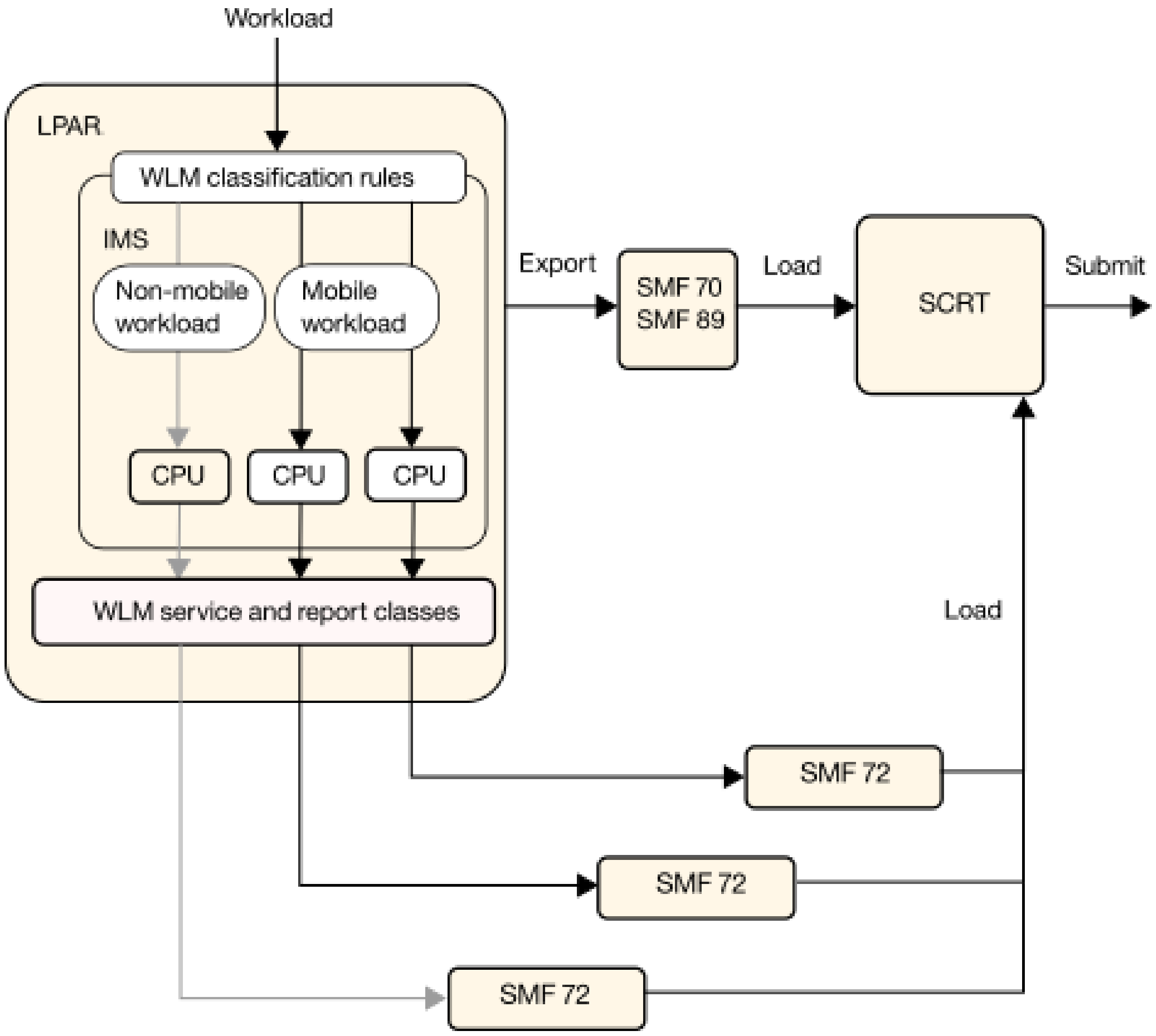


Figure 44. Generating workload reports with WLM classification rules

Generating reports with the CSV files

In cases where WLM classification rules for mobile workload transactions are not available, you are responsible for preprocessing your mobile workload transaction data into a predefined format to be loaded into SCRT for each sub-capacity reporting period. The data must consist of general-purpose processor CPU seconds for mobile workload transactions that are summarized by hour by LPAR for all machines that process mobile workload transactions.

For MWP Defining Programs that use the original CSV file method of data collection, follow the procedure to generate reports:

1. Track mobile workload transactions, including CPU seconds, per program on an hourly basis per LPAR.
2. Use the **`MWP`** command of IBM Transaction Analysis Workbench for z/OS to produce a comma-separated value (CSV) file. The CSV file must contain mobile workload transaction CPU consumption each month for each program.
3. Load the resulting data files along with the SMF type 70 and SMF type 89 records into SCRT each month.
4. Run SCRT Version 23 Release 13.0 or later and submit the results to IBM for each sub-capacity reporting period.

Deciding on monitoring activities and techniques

To develop a master plan for monitoring and analyzing performance, you should determine whether dynamic, daily, or detailed monitoring suites your needs best and develop a plan accordingly.

When you develop a master plan for monitoring and analyzing performance, you should establish:

- A master schedule of monitoring activity

  Coordinate monitoring with operations procedures to allow feedback of online events to be incorporated into instructions for daily or detailed data gathering.
- Which tools are to be used for monitoring

  The tools used for data gathering should provide for dynamic monitoring, a daily collection of statistics, and more detailed monitoring.
- The kinds of analysis to be performed

  Document what data is to be extracted from the monitoring output, identifying the source and usage of the data. Although the formatted reports provided by the monitoring tools help organize the volume of data, you should probably design worksheets to assist in data extraction and reduction.
- A list of the personnel that are to be included in any review of the findings

  The results and conclusions from analyzing monitor data should be made known to the user liaison group and to system performance specialists.
- A strategy for implementing changes to the online IMS system design resulting from tuning recommendations

  Coordinate this change implementation strategy with your installation's standards for testing and standards for frequency of production environment changes. Change management is described in Chapter 25, “Modifying the system design,” on page 385.

Plan for three broad levels of monitoring activity:

- Dynamic

  Observe the system's operation continuously to discover any serious short-term deviation from performance objectives.

  The output from the **`/DISPLAY`** or **`QUERY`** command is suitable for this level of monitoring, together with end-user feedback. One use of the Resource Measurement Facility (RMF) II is to collect information about processor, channel, and I/O device utilization.

The MTO is an important source of information about the behavior of the online IMS system. An important part of MTO feedback is the set of conditions during an IMS Monitor run. This information can help establish the validity of the monitor data.

With the status information that can be obtained using the **/DISPLAY** or **QUERY** command, you can arrange to get a processing status during online execution. The status can include the queue levels, active regions, active terminals, and the number and type of conversational transactions. Such a status can be obtained with the aid of an automated operator program invoked by the MTO. At prearranged milestones in the production cycle—such as before scheduling a message or BMP region, at shutdown of part of the network, or at peak loading—the transaction processing status and measures of system resource levels can be recorded.

To dynamically monitor paging rates, use one of the following methods:

Total System Paging Rate
The RMF Monitor II Paging report gives snapshots of paging activity by sampling interval.

Paging Rate by User
The RMF Monitor II Address Space Resource Data report gives counts of allocated frames and page faults by address space for each RMF measurement interval. The page fault count is cumulative and includes both local address space page faults (not page I/Os) and CSA/LPA.

If the monitoring indicates that IMS is experiencing temporary delays, it might be possible to stop some TSO or batch initiators.

To monitor processor resources, use the following:

Total System Utilization
The RMF CPU Activity report gives WAIT TIME PERCENTAGE for the RMF report interval, which might typically be 10 to 30 minutes.

Processor Over commitment
RMF Monitor II Real Storage/Processor/SRM report gives percentage processor utilization for each RMF sampling interval. It shows a 101% figure if any address space is ready to be dispatched but must wait for processor cycles during the interval.

- Daily

  Measure and record key system parameters daily. Record both the daily average and the peak period (usually one hour) average. Compare against major performance objectives and look for adverse trends.

  This data usually consists of counts of events and gross-level timings. In some cases, the timings are averaged for the entire IMS system, for example, elapsed times for input queuing or program execution.

  You can use the IMS system log as input to offline processing to produce statistics on a daily or regular basis. Two utilities, IMS Log Transaction Analysis and IMS Statistical Analysis, are suitable for this level of monitoring, because they impose no additional processing load on the online system.

  If the analysis of dynamic RMF Paging Activity reports shows that paging I/O has increased, this must be related to some changes in the workload during the corresponding period. More detailed analysis is usually required to decide on tuning actions.

  If analysis of the RMF CPU Activity reports show consistently high CPU utilization during a period of poor response times, examine the elapsed and processor times for transactions by IMS message region. If the elapsed time/processor ratio is significantly higher in the lower-priority regions, raise their dispatching priorities relative to other non-IMS work.

  Related reading: If all regions are equally affected, see "Minimizing path length" on page 408 for information on reducing path lengths. If appropriate, adjust priorities of competing work to move it below IMS message regions.

  To monitor utilization by subsystem for processor over commitment, use the RMF Monitor II Address Space State Data report. This report gives processor units consumed by each address space for each RMF measurement interval. IMS control and message regions can be identified by job name.

- Detailed

Periodically collect detailed statistics on system operation for performance analysis against system-oriented objectives and workload profiles.

Data at this level is much more voluminous. It typically contains sequences of events and tabulations. The timings reported are at a detailed level.

At this level of monitoring, special trace tools such as the IMS Monitor and Generalized Trace Facility (GTF) are useful. They collect a detailed sample of the online processing and distinguish between activity in dependent regions, asynchronous processing for terminals and message queues, buffer pool usage, and system data set I/O.

Additional information on using these monitoring tools is included in other topics of this section. The use of monitoring and tools to detect performance problems is explained in "Identifying and correcting performance problems" on page 411.

You can use the following methods for detailed monitoring of paging rates:

Paging Rate by User

If analysis at the daily level is insufficient, plan to run a GTF trace. If multiple page data sets are being used, private and global paging can be identified. Examine the GTF Detail Trace report to evaluate the impact on IMS of any paging by calculating the elapsed time delays due to page faults, particularly for the control region. The GTF Page Fault Summary report can be used to discover which area of IMS or system code is being affected by page faults. The report also indicates the type of page faults.

Analyze this type of data to help you decide whether to tune real storage usage.

Related reading: For other factors to consider when tuning real storage usage, see "Trade-offs between I/O controlled by IMS and paging" on page 406.

Examining NOT-IWAIT Time during Scheduling/Termination

The IMS Monitor Region Summary report can help you make an initial assessment of dispatching priority problems by examining the Scheduling or DL/I NOT-IWAIT times, that is, the elapsed time not accounted for by IWAIT time. Increases in the NOT-IWAIT times can be caused by:

- Paging delays
- Dispatching for a higher-priority task

If minimal paging is occurring, the portion of elapsed wait time that occurs during the scheduling and termination of a region is fairly consistent from system to system. This elapsed wait time is not accounted for by IWAIT time. This time is recorded in the IMS Monitor Region Summary report and tabulated under the heading NOT-IWAIT TIME. DL/I call NOT-IWAIT times can also include dynamic logging I/O delays. Any delays caused by paging or dispatching for a higher-priority task result in an increase in the NOT-IWAIT times.

If the total mean NOT-IWAIT TIME is excessive, the machine resource is probably inadequate for IMS. If no higher-priority tasks are present, the cause is probably a high paging rate for IMS scheduler code, control blocks, and PSB pool.

The SVC Mapping Summary can assist in determining where an SVC is being issued.

Related reading: For more information on reducing path lengths, see "Minimizing path length" on page 408.

Tools for detailed monitoring

Many of the monitoring tools you can use to collect detailed data are also used for general diagnostics. The principal tool provided by IMS for collecting and analyzing data is the IMS Monitor, which allows you to monitor online subsystems.

For a stand-alone IMS DB batch system driven by an SLDS, use the DB Monitor. The DB Monitor can be active during the entire execution of an IMS batch job, or you can stop and restart it from the system console.

You can also use the IMS Performance Analyzer, program isolation and lock traces, and the external trace facility.

To obtain a unified view of workload performance for IMS systems and other participating z/OS components or middleware across the z/OS stack, you can use IBM z/OS Workload Interaction Correlator to gain data in a standardized, synchronized, contextualized format.

Related concepts

“//DFSSTAT reports” on page 679
The //DFSSTAT reports show you how many DB and DC calls are issued by an application program and describe buffering activity during the application's execution. The reports are written when the application program terminates.

IMS Performance Analyzer for z/OS overview

Related tasks

Gathering performance-related data (Operations and Automation)

IMS Monitor

The IMS Monitor collects data while the online IMS subsystem is running. It gathers information for all dispatch events and places it, in the form of IMS Monitor records, in a sequential data set. Use the IMSMON DD statement in the IMS control region JCL to specify the IMS Monitor data set.

IMS adds data to this data set when you activate the IMS Monitor using the **/TRACE** command. The IMS MTO can start and stop the IMS Monitor to obtain snapshots of the system at any time. However, the IMS Monitor adds to system overhead and generates considerable amounts of data.

Control of IMS Monitor output

Plan to run the IMS Monitor for short intervals and to control its operation carefully. Shorter intervals also prevent the overall averaging of statistics, so that problems within the system can be more readily identified. The output of the IMS Monitor can be constrained by:

- Type of activity monitored
- Database or partition or area
- Dependent region
- Time interval

IMS Monitor output data sets

The IMS Monitor output can be either a tape or a DASD data set. Using DASD eliminates the need to have a tape drive allocated to the online system. If you want to use the IMS Monitor frequently, you might find that permanently allocated space for a DASD data set is convenient. One technique is to code DISP=SHR on the IMSMON DD statement so that the reports can be generated as each IMS Monitor run completes.

You must coordinate the report generation with the operator because each activation of the IMS Monitor writes over existing data. Although this overwriting does not occur for tape data sets, new volumes must be mounted. The volume is rewound, and a mount request is issued each time you start the IMS Monitor.

Recommendations:

- Do not catalog IMS Monitor data sets. The IMS Monitor can produce multiple output volumes while IMS is running if the data sets are not cataloged.

  If you want IMS to dynamically allocate the IMS Monitor data set, do not include the IMSMON DD statement in the IMS control region JCL.
- Allow IMS to dynamically allocate IMS Monitor tape data sets. A tape drive is not permanently reserved for the control region for dynamically allocated data sets.

IMS Monitor traces

After you establish monitoring requirements, you might be able to restrict the scope of the IMS Monitor activity. Restricting the scope has the advantage of reducing the impact of the IMS Monitor on system throughput. However, do not compromise the collection of useful data.

You can control what specific types of events are traced by using specific keywords on the **/TRACE** command. For example, you can monitor line activity, scheduling and termination events, activity between application programs and message queues, activity between application programs and databases, or all activity. You can also limit monitoring to:

- Particular databases, partitions, or areas
- Particular dependent regions
- A specified interval of time

IMS Monitor reports

You can obtain reports based on the IMS Monitor output by using the IMS Performance Analyzer for z/OS or the IMS Monitor Report Print utility (DFSUTR20).

The IMS Monitor Report Print utility summarizes and formats the raw data (except Fast Path data) produced by IMS and presents the information in a series of reports. You can suppress the reports pertaining to DL/I calls and tabulated frequency distributions.

The duration of the monitored events is determined by the entries for start and stop of the IMS Monitor. You cannot select a different time period for reporting, because many of the timed events are not captured continuously: only when the IMS Monitor is started and stopped. For this reason, ensure that the IMS Monitor is stopped before taking any action to stop the IMS control region.

Related concepts
IMS Monitor (Database Administration)
Monitoring the system (Operations and Automation)
IMS Performance Analyzer for z/OS overview
Related reference
IMS Monitor Report Print utility (DFSUTR20) (System Utilities)

z/OS Generalized Trace Facility (GTF)

Use the z/OS generalized trace facility (GTF) to record a wide range of system-level events. The trace activity is controlled from the z/OS system console using the **MODIFY** command.

Output is spooled to a sequential data set that is used by a generalized formatting utility. You can write exit routines that are called by the formatting utility to edit the trace records and present the data as desired.

The following GTFPARS reports are of special interest for IMS:

- Job Summary report: This report lists the following for each of the IMS message regions and for the control region: SVC counts, loading of modules, and EXCP, SIO, and IO timings.
- Detail Trace report lists the following information:
  - For IMS message regions: Detailed analysis of program management activity. This analysis gives you the ability to investigate event sequences in detail.
  - For the IMS control region: Database I/O that is handled in the control region, and network-related activity.
  - For all IMS regions running in parallel: Contention between regions (for example, contention for access to a program library), and the flow of control from the control region to the MPPs.
- Job Summary report of Master Address Space in z/OS: This report shows system paging activity.

- Detail Trace report of Master Address Space in z/OS: You can use this report for a detailed investigation of delays caused by paging.
- System Summary report: You can use this report for I/O subsystem analysis for all IMS data sets.

Related concepts

z/OS: Generalized trace facility (GTF)

z/OS Component Trace (CTRACE) Service

IRLM uses the z/OS component trace (CTRACE) service to trace IRLM activity. Because the trace output is in z/OS CTRACE format, you can use IPCS CTRACE format, merge, and locate routines to process the buffer data.

Use the z/OS **TRACE CT** command to start, stop, or modify an IRLM diagnostic trace. This command can be entered only from the z/OS master console. Entering the commands requires an appropriate level of z/OS authority. IRLM does not support all the options available on the command. You can also start the IRLM tracing by placing **TRACE=YES** in the IRLM procedure.

Related tasks

Tracing IRLM activity (Operations and Automation)

Related reference

TRACE CT command (Commands)

z/OS: MVS interactive problem control system (IPCS) CTRACE subcommand

z/OS: TRACE command

Obtaining program isolation and lock traces

In an IMS DB/DC or DBCTL environment, you can detect contention for a database segment by examining the output produced by the Program Isolation Trace Report utility (DFSPIRP0).

To get the source data for the utility, issue the **/TRACE SET ON PI OPTION ALL** command. To stop gathering source data, issue the **/TRACE SET OFF PI** command. A control statement for the utility can select a start or stop time relative to a specified date.

Tracing the program isolation function can create additional log records. These records contain the enqueue or dequeue requests issued by the program isolation function between sync points as a result of database updates, checkpoints, and message handling events.

The Program Isolation Trace Report utility reports only those events that require wait time. The report identifies the data management block (DMB) name, database control block (DCB) number, the relative byte address (RBA), the program specification block (PSB) name, and the transaction code. The utility sorts all activity by RBA number (shown as ID in the report). The report lists elapsed times for enqueues that required a wait during the trace interval, and totals the number of enqueues for each ID, DCB, and DMB. The requesting PSB or transaction is considered the holding PSB or transaction of the next enqueue waiting for the same segment. A sample report is shown in the following example. In this report, no elapsed wait time is recorded for Fast Path.

```
                     P R O G R A M  I S O L A T I O N  T R A C E  R E P O R T         PAGE   1
 DATE:  08/10/04
 TIME:  16:36  TO  16:37
 DCB *** REQUESTING *** ELAPSED   **** HOLDING *** ID TOTAL DCB TOTAL DMB      TOTAL
 DMB NAME NUM  ID  TRAN AND PSB NAMES   TIME       TIME    TRAN AND PSB NAMES ENQ'S ENQ'S ENQ'S
TABLEDBQ 1 0022D020  DE1Q     PROGDE1Q  16:36:54   0:00.061  DE2Q     PROGDE2Q
                                                                                  1
           003BE00C  DE2Q     PROGDE2Q  16:36:51   0:00.027  DE1Q     PROGDE1Q
                                                                                  1
           007D901C  DE2Q     PROGDE2Q  16:36:34   0:00.036  DE1Q     PROGDE1Q
                                                                                  1
           008EF014  DE2Q     PROGDE2Q  16:36:49   0:00.038  DE1Q     PROGDE1Q
                                                                                  1
           0090401C  DE1Q     PROGDE1Q  16:36:50   0:00.072  DE2Q     PROGDE2Q
                                                                                  1
           00A06010  DE2Q     PROGDE2Q  16:36:38   0:00.046  DE1Q     PROGDE1Q
                                                                                  1
           00A1401C  DE1Q     PROGDE1Q  16:36:50   0:00.008  DE2Q     PROGDE2Q
```

```
                                                                                1    7    7
TABLEDBR 1 002A901C  DE2R     PROGDE2R  16:36:40   0:00.034  DE1R     PROGDE1R
                                                                                1
           0045801C  DE2R     PROGDE2R  16:36:41   0:00.043  DE1R     PROGDE1R
                                                                                1
           0072F024  DE1R     PROGDE1R  16:36:30   0:00.053  DE2R     PROGDE2R
```

You can use the File Select and Formatting Print utility to select and print trace table and PI entries in the log records in the following ways:

- Specify an OPTION statement with the PRINT parameter and COND=E and EXITR=DFSERA40 keyword parameters. The output is a report containing the program isolation (PI) trace records formatted in sequential order.
- Select only the log records that contain the trace using the IMS Trace Table Record Format and Print module (DFSERA60), which is an exit routine that receives type X'67FA' log records from the File Select and Formatting Print utility and formats the records on the SYSPRINT data set.

  Specify an OPTION statement with the PRINT parameter and COND=E and EXITR=DFSERA60 keyword parameters. The output is a report containing the PI trace entries, the DL/I trace entries, and the lock trace entries formatted to show these entries in sequential order. For an explanation of the headings, see an assembly listing of the macro IDLIVSAM TRACENT.

You can use an output report from the IMS Trace Table Record Format and Print module to find out more information about:

- The level of control (LEV) column shows read only, share, exclusive control, and single update activity.
- The return code (RC) column shows return codes from DFSFXC10 or the IRLM. You can determine whether the caller had to wait for the requested resource, or if the transaction caused a deadlock situation.
- The PST post code (PC) column shows the cause of the wait. If the entry is X'60', a deadlock occurred.

You can reduce the number of records examined by specifying an additional OPTION statement to the File Select and Formatting Print utility so that only records confirming deadlock are printed.

IMS automatically resolves deadlock situations by using dynamic backout. But the detection of deadlocks is important so that you can modify your application design to prevent future deadlocks.

The advantage of the Program Isolation Trace Report is that it shows where contention for a particular segment or range of segments occurs. The report also shows which transactions are competing within a database. It also shows high wait times that might explain a delay in response time. One way to handle the segment contention might be change the database design to separate some of the fields into an additional segment type.

Related reference

Program-Isolation-Trace Report utility (DFSPIRP0) (Database Utilities)

Program Isolation Trace Record Format and Print module (DFSERA40) (System Utilities)

IMS Trace Facility

You can use the IMS Trace facility to write IMS trace tables internally or to an external trace data set.

IMS can write this external trace data set to either DASD or tape:

- DASD data sets can be allocated by JCL or can be dynamically allocated.
- Tape data sets must be dynamically allocated.

You can also write the trace tables to the OLDS, but this could adversely affect OLDS performance. The external trace data sets are independent of the OLDS, so you can write trace tables to the external trace data sets even if the OLDS is unavailable.

To display the status of traces, use the **`/DISPLAY TRACE`** command. This command can be used to determine the status of the IMS traces in effect and the status of any external trace data sets in use.

Analyzing performance in OTMA message routing

You can use the OTMA type-2 command, **QUERY OTMATI**, to analyze performance in OTMA message routing.

You can issue the **QUERY OTMATI** command in active and alternate systems in an Extended Recovery Facility (XRF) environment.

If you issue the command **QUERY OTMATI** without any parameters, IMS displays:

- The workloads on each of the IMS instances in the IMSplex with the target member (tmember) name and the transaction pipe (tpipe) name
- The total number of active messages in the queue as a subset of the total workload
- The length of time that the transaction instance block (TIB) has existed and the correlation ID of the input message

You can use this information to determine if there are problems processing OTMA input messages.

If you include the SHOW parameter with the **QUERY OTMATI** command, the total number of active messages in the queue is not displayed. Instead, each individual active message is displayed, as filtered by the other specific parameters. For example, to show how many transaction codes have an age of five seconds or more, issue the following command:

```
QUERY OTMATI MSGAG(5) SHOW(TRAN)
```

This command displays the individual TIBs, and shows how many transaction codes have an age of five seconds or more. From this information, you can determine any potential problems in the message queue that can be corrected or avoided, such as storage filling with control blocks, or you can balance the workload of the outbound transactions in the IMSplex without restarting IMS.

z/OS Workload Interaction Correlator

The IBM z/OS Workload Interaction Correlator is a priced feature that provides the collection of workload data for IMS systems and other participating z/OS components and middleware to diagnose performance issues across the z/OS stack. It generates System Management Facilities (SMF) type 98 records and graphical representations for further analysis. With this feature, you can get a summary view of activities of the IMS workload.

IBM z/OS Workload Interaction Correlator requires an IBM z14® or a higher hardware. Support is included in the PTF for APAR OA57165 for z/OS in V2R3 and V2R4.

With IMS support for IBM z/OS Workload Interaction Correlator, IMS becomes a participating application. For every normally complete transaction, Message Processing Program (MPP) regions and Java message processing (JMP) regions record both elapsed time and CPU time in a standard form and in an IBM System z Integrated Information Processor (zIIP form). z/OS System Management Facility (SMF) then logs type 98 records with subtype 1025 to record these statistics. z/OS Workload Interaction Navigator can be used to generate graphical displays of the data.

A new macro DFSSR98A contains layouts of the IMS-specific portion of the type 98 subtype 1025 SMF record. The macro can be used by z/OS Workload Interaction Navigator to generate graphical reports. DFSSR98A is classified as diagnosis, modification, and tuning information (DMTI).

To enable IMS to write IBM z/OS Workload Interaction Correlator SMF 98 subtype 1025 records, you must specify the parameter **WIC** in the z/OS SMFPRM*xx* parmlib member, and request SMF 98.1025 records to be written.

All of the following SMFPRM*xx* TYPE specification examples include the IMS 98.1025 subtype:

TYPE(98) - All SMF 98 subtypes
TYPE(98(1025)) - Only the SMF 98.1025 subtype
TYPE(98(1,1024,1025)) - SMF 98 subtypes 1, 1024, and 1025

To prevent IMS from writing IBM z/OS Workload Interaction Correlator SMF 98 subtype 1025 records while writing other SMF 98 subtypes, you must exclude SMF 98.1025 records from being written in your SMFPRM*xx* parmlib member. For example, you might specify:

```
TYPE(98(1:1024,1026:2000))
```

in SMFPRM*xx* to have SMF write all SMF 98 subtypes 1 - 2000 except for type 1025.

You can dynamically enable and disable IMS writing function of SMF 98.1025 records by using the SET SMF command to change between SMF parmlib members with different options. This can be used to change the collection state of the SMF 98.1025 record while IMS is up.

To learn more about how to enable and disable IMS to write IBM z/OS Workload Interaction Correlator, see IBM Documentation for z/OS.

IMS registers with IBM z/OS Workload Interaction Correlator in the control region initialization. If IBM z/OS Workload Interaction Correlator is not active when IMS starts, that instance of IMS will not write SMF 98 subtype 1025 records, even if IBM z/OS Workload Interaction Correlator is started later. To have IMS write SMF 98 subtype 1025 records, ensure that IBM z/OS Workload Interaction Correlator is active when IMS is started.

Related information

z/OS: About the IBM z/OS Workload Interaction Correlator

z/OS: Enabling Workload Interaction Correlator

Coordinating performance information in an MSC network

The IMS system log for each system participating in Multiple Systems Coupling (MSC) contains only the record of events that take place in that system. The logging of traffic received on links is included.

You can augment the system log documentation that records the checkpoint intervals with the system identifications of all coupled systems. This helps you interpret reports, because you know of transactions that might be present in message queues but are not processed, and you can expect additional transaction loads from remote sources. In your analysis procedures, include ways of isolating the processing triggered by transactions originating from another system.

To satisfy the need for monitoring with typical activity that includes cross-system processing, coordinate your scheduling of the IMS Monitor and other traces between master terminal operators. The span of the monitoring does not have to be exactly the same, but if it is widely different, the averaging of report summaries can make it harder to interpret the effect of the processing triggered by cross-system messages.

Related reading: For more information on interpreting MSC reports, see "IMS Monitor reports for MSC" on page 627.

Monitoring Fast Path systems in DB/DC and DCCTL environments

The major emphasis for monitoring IMS online systems that include message-driven Fast Path application programs is the balance between rapid response and high transaction rates. With Fast Path, performance data is made part of the system log information.

Also, the IMS Monitor can be used to monitor Fast Path activities. You can use the IMS Fast Path Log Analysis utility to generate statistical reports from the system log records. This utility can provide summaries of the Fast Path transaction loads, reports that highlight exceptional response time, and a breakdown of the elapsed time between key events during the time in the system.

The system administration tasks of setting up a monitoring strategy, performance profiles, and analysis procedures should be carried into the Fast Path environment.

Related reading: For more information on using either the IMS Monitor or the IMS Fast Path Log Analysis utility, see *IMS Version 15.2 System Utilities*.

Transaction flow in DB/DC and DCCTL environments

A distinct sequence of events occurs during the processing of a transaction. Message-related processing is asynchronous within IMS, that is, not associated with a dependent region's processing.

Examples of this kind of processing are message traffic, editing, formatting, and recovery-related message enqueuing, any of which can be done concurrently with application program processing for other transactions. Events from application program scheduling to termination are associated with a PST and can be regarded as synchronous.

To get an overall picture of the activity that takes place when an online IMS system is processing a mix of transactions concurrently, see the following figure. The figure shows a sequence of events. Each event is explained in notes that follow the figure.

The unit of work by which most IMS systems are measured is the transaction (or a single conversation iteration, from entering of an input message to receipt of one or more output messages in response). One way of representing the flow of units of work is to compare it to three funnels through which all transactions must pass, as illustrated in the following figure. The events that account for the principal contributions to transaction response time are numbered in the center. The items entered on the left of the diagram are message related, and those on the right are related to the application program. The arrows trace the flow for an individual transaction. (The diagram does not show the paging element or system checkpoint processing that is distributed through the elapsed times.)

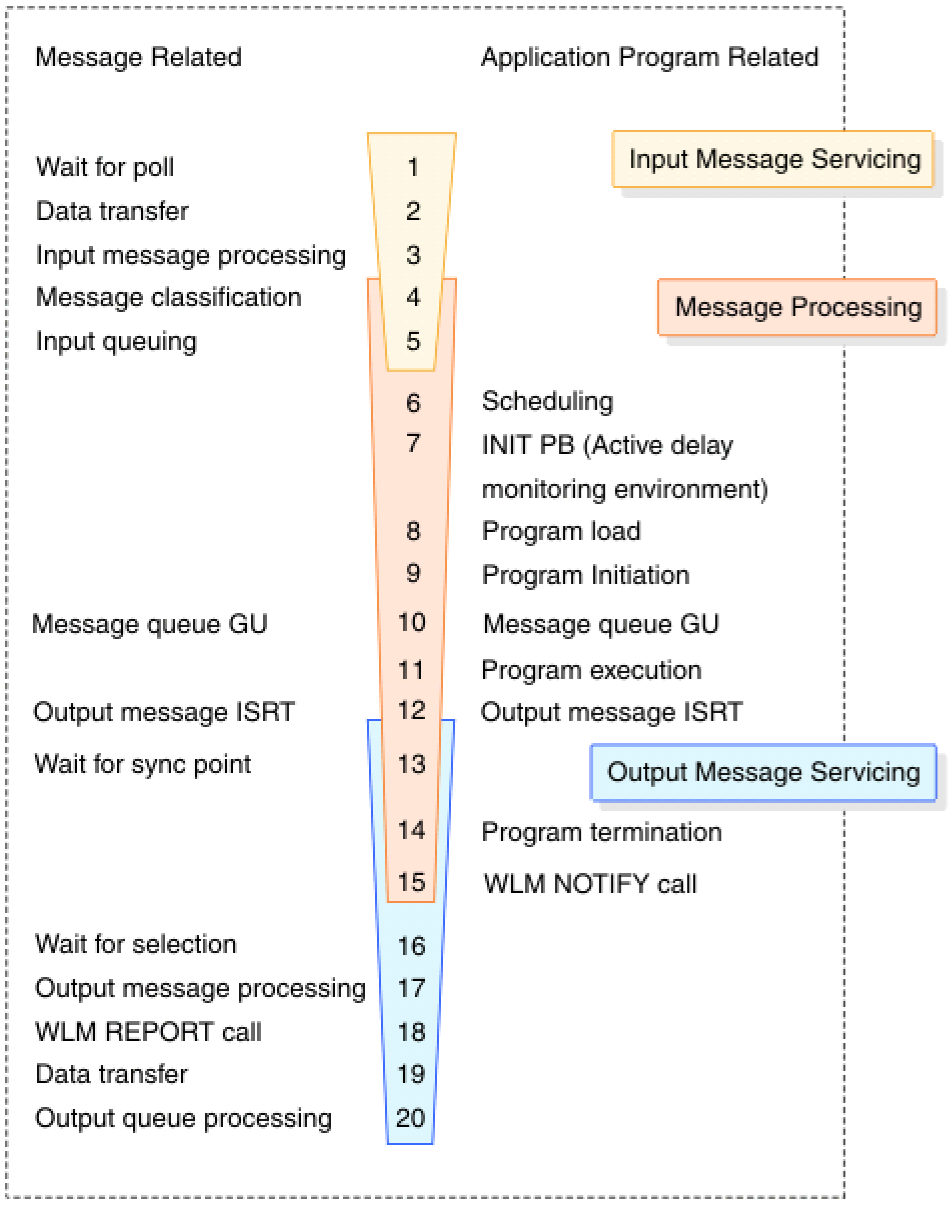


Figure 45. Processing events during transaction flow through IMS

Notes to the figure:

1. Wait for Poll.

   This is the time between pressing the Enter key and receiving a poll that results in the data being read by the channel program. With several control units on a line, a slight delay is probable, because each control unit is polled in turn until one is ready to send. There can also be delays caused by data being transmitted on the line from (or to) another terminal. Also, IMS communications can be currently processing an input message from a terminal on the line and polling has not recommenced. Some form of hardware monitor is required to measure the time waiting for poll.
2. Data Transfer.

   This time includes propagation delay and modem turnarounds for multi-block input messages. You can estimate the data transfer times if the volume of data transmitted is known.
3. Input Message Processing.

   The IMS control of the transaction begins when the input message is available in the HIOP. The time the message spends in this pool, in MFS processing, and in being moved to the message queue buffers affects response time. Individual transaction I/O to the Format library affects the message queue. A major factor in determining response time is whether the respective pools are large enough for the current volume of transactions flowing into input queuing. In particular, if the message queue pool is too small, overflow to the message queue data sets occurs.

4. Message Classification.

   This is the call to the z/OS WLM to obtain a WLM service classification for the incoming message.

5. Input Queuing.

   This is the time spent on the input queue or in the message queue buffers waiting for a message region to become available. In a busy system, this time can become a major portion of the response time. The pattern of programs scheduled into available regions and the region occupancy percentage are important and should be closely monitored.

6. Scheduling.

   Because of class scheduling, regions can be idle while transactions are still on the queue. The effects of scheduling parameters can be:

   - Termination of scheduling as a result of PSB conflict or message class priorities
   - Termination of scheduling as a result of intent conflict
   - Extension of scheduling by I/Os to IMS.ACBLIB for intent lists, PSBs, or DMBs
   - Pool space failures in either the PSB or DMB pools

7. Init PB Call (Activate Delay Monitoring Environment).

   Activate the WLM delay monitoring environment for the message when it is placed into the dependent region. The WLM PB is initialized with the Service Classification and transaction name, message arrival time, program execution start time (current time), userid, and so forth.

8. Program Load.

   See event “9” on page 376

9. Program Initialization.

   After scheduling, several kinds of processing events occur before the application program can start:

   - Contents supervision for the dependent region
   - Location of program libraries and directories to them
   - Program fetch from the program library
   - Program initialization up to the time of the first DL/I call to the message queue

   For monitoring, you can obtain the overall time for the above activities. The number of I/Os should be checked periodically.

10. Message Queue GU.

    This is the GU call to the message queue. It is chosen as a measuring point because the event is recorded on the system log and is used as a starting point for iterations of processing when more than one message is serviced at a single scheduling of the program.

11. Program Execution.

    The time for program execution, from first message call to the output message insert, is a basic statistic for each individual transaction. It is important to account for that time in terms of the work performed:

    - The number of transactions processed per schedule
    - The number and type of DL/I calls per transaction
    - The number of I/Os per transaction

    A useful breakdown of elapsed time into processor time and I/O helps determine which transactions use significant resource.

12. Output Message Insert.

This time begins the asynchronous processing for the output response. The output message requests flow into the funnel to be serviced while the application program is either beginning to process another input message or is performing closeout processing and program termination.

13. Wait for Sync Point.

    When an output message is issued by a program, it is enqueued on a temporary destination until the program reaches a synchronization point. For programs specified as MODE=MULT, a long delay in output transmission can occur when the program goes on to process several transactions at one scheduling. None of the previous output messages can be released for transmission until the program ends. If the program fails, all current transactions are backed out. Actually, when the messages are dequeued, LIFO sequence is used, from temporary to permanent destination. With MODE=SNGL, the wait for sync point (at the next GU to the message queue) is normally negligible.

14. Program Termination.

    In the case of MODE=MULT, discussed in event “13” on page 377, the synchronization point occurs at program termination. Any database updates are purged from the database buffer pools, and the waiting output messages are released.

    In the MODE=SNGL case, the synchronization point occurs at the previous message queue GU call (usually a GU with a QC status code), and no database commit processing occurs at termination, unless the application program has updated a database after the last message queue GU.

15. WLM Notify Call.

    This tells WLM that the application program has ended execution. The PB and current time are passed to WLM.

16. Wait for Selection.

    This time is similar to Wait for Poll on input, with one difference—an output message might not have to wait for the completion of a polling cycle if no polling is in progress on the line at the time the output message is enqueued. However, there might be a wait for the duration of data transmission to other terminals on the line. In a busy system, this could account for the majority of time spent on the output queue.

17. Output Message Processing.

    This activity is similar to event “3” on page 375.

18. WLM Report Call.

    This tells WLM the response is being sent. IMS passes the input message arrival time, the Service Classification, the current time (output message send time).

19. Data Transfer.

    This activity is similar to event “2” on page 375.

20. Output Queue Processing.

    Output messages that have been sent are dequeued after their receipt has been acknowledged by the terminal. In the case of paged output, the acknowledgment is a consequence of another input or a PA2 entry from the terminal.

Related concepts

“DB/DC environment” on page 5

In the DB/DC environment, data is centrally managed for applications that are being executed concurrently and made available to terminal users. Database Recovery Control (DBRC) facilities help to manage database availability, data sharing, and system logging.

The IMS Monitor in DB/DC and DCCTL environments

The principal monitoring tool provided by IMS is the IMS Monitor. It is an integral part of the control program in the DB/DC environment. The counterpart of the IMS Monitor in the batch environment is the DB Monitor.

The IMS Monitor collects data while the DB/DC environment is operating. Information in the form of monitor records is gathered for all dispatch events and placed in a sequential data set. The IMS Monitor data set is specified on the IMSMON DD statement in the control region JCL; data is added to the data set when the **/TRACE** command activates the monitor. The MTO can start and stop the monitor, guided by awareness of the system's status, to obtain several snapshots.

Related reading: For more information on interpreting IMS Monitor reports, see Part 5, “Using IMS reports,” on page 571.

Monitoring transaction-level statistics

IMS logs statistical data about transactions to the OLDS in the X'56FA' log record. You can use this log record to collect, analyze, and interpret statistical data for general business purposes.

This statistical information is only a subset of the information that is logged to the X'07' log record. Transactions within a specific scope (such as a unit of recovery) from the X'07' log record are collected. Since the statistical information about the transactions is also logged separately in the X'56FA' log record, you can access this information more easily than extracting it from the X'07' log record.

IMS logs statistical information about the transactions within a unit of recovery (work done by an application program between sync points) in log record X'56FA' for non-message-driven applications, or when either:

- The keyword MODE=SNGL is specified in the TRANSACT macro
- The CMTMODE(SNGL) parameter is specified on a **CREATE TRAN** or **UPDATE TRAN** command

IMS logs statistical information about the transactions after each message in log record X'56FA' for message-driven applications when either:

- The keyword MODE=MULT is specified in the TRANSACT macro
- The CMTMODE(MULT) parameter is specified on a **CREATE TRAN** or **UPDATE TRAN** command

You can enable or disable this logging on a program basis for non-message driven applications, and on a transaction-by-transaction basis for message-driven applications. Use the following commands to enable or disable this logging on a transaction-by-transaction basis:

- For an existing transaction, issue the **UPDATE TRAN** or **UPDATE TRANDESC** command and specify the new TRANSTAT() parameter.
- For a new transaction, issue the **CREATE TRAN** or **CREATE TRANDESC** command and specify the new TRANSTAT() parameter.

To enable or disable this logging on a program basis:

- For an existing application program, issue the **UPDATE PGM** or **UPDATE PGMDESC** command and specify the new TRANSTAT() parameter.
- For a new application program, issue the **CREATE PGM** or **CREATE PGMDESC** command and specify the new TRANSTAT() parameter.

You can enable or disable this logging globally during system definition by specifying a new parameter, TRANSTAT= Y | N, in the Diagnostics Statistics section of the new DFSDFxxx PROCLIB member. This setting applies to any transactions and application programs that are created with the system definition process.

Any transactions or application programs that are created after an IMS cold start using the online change process or the Destination Creation exit routine (DFSINSX0) inherit the TRANSTAT= parameter setting that is specified in the DIAGNOSTICS_STATISTICS section of the DFSDFxxx PROCLIB member.

To determine whether or not this logging is enabled for a given transaction or application program, issue one of the following type-2 commands:

- **`QUERY TRAN`**
- **`QUERY TRANDESC`**
- **`QUERY PGM`**
- **`QUERY PGMDESC`**

Monitoring procedures in a DBCTL environment

This topic explains how to establish monitoring procedures for your DBCTL environment. First, consider that monitoring in a DB/DC environment generally refers to the monitoring of transactions.

The transaction is entered by an end user on a terminal, is processed by the DB/DC environment, and returns a result to the user. Transaction characteristics that are measured include total response time and the number and duration of resource contentions,

A DBCTL environment has no transactions and no terminal end users. It does, however, do work on behalf of CCTL transactions that are entered by CCTL terminal users. DBCTL monitoring provides data about the DBCTL processing that occurs when a CCTL transaction accesses databases in a DBCTL environment. This access is provided by the CCTL making the database resource adapter (DRA) request.

The most typical sequence of DRA requests that represents a CCTL transaction would be:

- A SCHED request to schedule a PSB in the DBCTL environment
- A DL/I request to make database calls
- A sync point request, COMMTERM, to commit the database updates and release the PSB

The DBCTL process that encompasses this request is called a unit of recovery (UOR).

DBCTL provides UOR monitoring data, such as:

- Total time the UOR exists
- Wait time to schedule a PSB
- I/O activity during database calls

This information is similar to, and is often the same as, DB/DC monitoring data. However, in a DBCTL environment, the UOR data represents only a part of the total processing of a CCTL transaction. You must also include CCTL monitoring data to get an overall view of the CCTL transaction performance.

In this topic, the term "transaction" refers to a CCTL transaction. When it applies, UOR is specifically named.

The CCTL administrator must decide what strategy to use to monitor transaction performance. This topic describes some possible strategies and how IMS functions can help.

Several types of monitoring strategies are available. You can:

- Summarize actual workload for the whole of the online execution. This can be continuous or at an agreed-to frequency. Total workload or selected representative transactions can be tracked.
- Take sample snapshots at peak loads and under normal conditions. It is always useful to monitor the peak periods for two reasons:
  - Bottlenecks and response time problems are more pronounced at peak volumes.
  - The current peak load is a good indicator of what the future average will be like.
- Monitor critical transactions or programs that have documented performance criteria.

Plan your monitoring procedures in advance. A procedure should explain the tools to be used, the analysis techniques to be used, the operational extent of those activities, and how often they are to be performed.

You need to:

- Develop performance criteria
- Develop a master plan for monitoring, data gathering, and analysis

Establishing performance objectives in a DBCTL environment

Inherent in the design of your online DBCTL/CCTL system are your performance objectives. Establishing performance objectives is a major task requiring data gathering for the DBCTL/CCTL workload as a whole. After defining the workload and estimating the resources required, you must reconcile the desired response with the response you consider to be attainable. Monitor the performance of the system to determine if these objectives are being met.

Base performance objectives on:

- Desired, acceptable, and maximum response time
- Average and maximum resource demands (or workload) per transaction
- Predicted and actual transaction volumes

Establishing your performance objectives is an iterative process involving the following activities:

- Defining CCTL user-oriented performance objectives and priorities

  These derive from the way a CCTL end user perceives the service provided by the system. These objectives state expectations of response time as seen by the end user at the terminal.

  When response time objectives are established, they should not only reflect the time-in-system (the elapsed time from entry of a last input message segment to the first response segment), but also the expected amount of IMS, CCTL, and application program processing. You should consider whether to define your criteria in terms of the average, the 90th percentile, or even worst-case response time. Your choice depends on your installation's audit controls and the nature of the specific transactions.
- Determining how performance against these objectives is measured and reported to users

  You can combine UOR data with CCTL transaction data for a complete transaction profile, but keep the following things in mind:

  – If a CCTL allows only one UOR per transaction, this one-to-one ratio makes it easy to combine the monitoring data. It is also easier to track, because a CCTL can put a transaction-ID into a SCHED request. DBCTL puts this ID in monitor log records, and it appears in monitor reports.
  – If a CCTL allows for multiple UORs within a single transaction, the total UOR data can be obtained by adding the data from each of the UORs.

  Besides data from the IMS Monitor, UOR data is also provided to a CCTL upon the completion of a request terminates the UOR. This data can be used in either of the cases above. A CCTL can also perform real-time analysis of UOR statistics related to its transactions.

  The same UOR data is also part of the IMS system log, which records the status and end of a UOR.
- Understanding and documenting the current workload

  This requires breaking down the total work into categories and, for each category, developing a workload profile (generally estimates) that include:

  – Definition of the transaction category (for example, transaction type or group of transactions). Two characteristics of the category are:

    - The transaction profile. Most transactions perform a single function and have an identifiable workload profile. Later in the process, transaction types with common profiles can be amalgamated for convenience.
    - The transaction volume. In situations where the workload to be documented is already operational, a summary of transaction volumes can be obtained. In other cases, volumes are estimated.

  - Relative priority of category, including periods during which priority changes
  - Resource requirements of the work:
    - Physical resources managed by the operating system (real storage, DASD I/O, terminal I/O)
    - IMS logical resources managed by the subsystem, such as control blocks, latches, buffers, and number of database resource adapter (DRA) threads
    - CCTL resources required

  To obtain a base profile of transaction resource demands, you can start DBCTL/CCTL on a dedicated machine and execute a few transactions to accomplish initialization and buffer pool usage. Then start the IMS Monitor and measure a sample of transaction execution.

  You can use the base transaction profile to examine the transaction workload to see if there is any way to reduce it. Such design changes result in the greatest impact, occurring before system-wide contention. You can also compare the base profile against the transaction profile in the production environment.

- Translating the resource requirements and volume information obtained into system-oriented objectives for each work category

  This includes statements about the transaction rates to be supported (including any peak periods) and the internal response-time profiles to be achieved.

- Confirming that the system-oriented objectives are reasonable

  After initializing the system and monitoring its operation, you need to find out if the objectives are reasonable (given the hardware available), based upon the measurements of the workload. If the measurements differ greatly from the estimates used, you must revise the workload documentation and system-oriented objectives accordingly, or tune the system.

Deciding on monitoring activities and techniques in a DBCTL environment

To develop a master plan for monitoring and analyzing performance, you should determine whether dynamic, daily, or detailed monitoring suites your needs best and develop a plan accordingly.

When you develop a master plan for monitoring and analyzing performance, you should establish:

- A master schedule of monitoring activity

  Coordinate monitoring with operations procedures to allow feedback of online events to be incorporated into instructions for daily or detailed data gathering.

- Which tools are to be used for monitoring

  The tools used for data gathering should provide for dynamic monitoring, a daily collection of statistics, and more detailed monitoring.

- The kinds of analysis to be performed

  Document what data is to be extracted from the monitoring output, identifying the source and usage of the data. Although the formatted reports provided by the monitoring tools help organize the volume of data, you should probably design worksheets to assist in data extraction and reduction.

- A list of the personnel that are to be included in any review of the findings

  The results and conclusions from analyzing monitor data should be made known to the user liaison group and to system performance specialists.

- A strategy for implementing changes to the DBCTL environment design, and to the CCTL system design, resulting from tuning recommendations

  Coordinate this change implementation strategy with your installation's standards for testing and standards for frequency of production environment changes. Change management is described in Chapter 25, "Modifying the system design," on page 385.

Plan for three broad levels of monitoring activity:

- Dynamic

Observe the system's operation continuously to discover any serious short-term deviation from performance objectives.

This topic refers to IMS functions and general concepts related to dynamic monitoring. If the CCTL has similar functions (for example **DISPLAY** commands), they should also be used.

The output from the **/DISPLAY** or **QUERY** command is suitable for this level of monitoring, together with end-user feedback. One use of the Resource Measurement Facility (RMF) II is to collect information about processor, channel, and I/O device utilization.

The MTO is an important source of information about the behavior of the online IMS system. An important part of MTO feedback is the set of conditions during an IMS Monitor run. This information can help establish the validity of the monitor data.

With the status information that can be obtained using the **/DISPLAY** or **QUERY** command, you can arrange to get a processing status during online execution. The status can include the pool levels and active regions. At prearranged milestones in the production cycle—such as before scheduling a message or BMP region, at shutdown of part of the network, or at peak loading—the transaction processing status and measures of system resource levels can be recorded.

- Daily

  Measure and record key system parameters daily. Record both the daily average and the peak period (usually one hour) average. Compare against major performance objectives and look for adverse trends.

  The IMS system log can be used as input to offline processing to produce statistics on a daily or regular basis. The 07 and 08 log records contain the UOR data.

- Detailed

  Periodically collect detailed statistics on system operation for performance analysis against system-oriented objectives and workload profiles.

  Data at this level is much more voluminous. It typically contains sequences of events and tabulations. The timings reported are at a detailed level.

  At this level of monitoring, special trace tools such as the IMS Monitor and Generalized Trace Facility (GTF) are useful. They collect a detailed sample of the online processing and distinguish between activity in CCTL threads, buffer pool usage, and system data set I/O.

  Related reading:

  – For information on using the IMS Monitor, see *IMS Version 15.2 Operations and Automation*.

  – For information on how monitoring and tools are used to detect performance problems, see "Identifying and correcting performance problems" on page 411.

  System for Generalized Performance Analysis Reporting (GPAR), program number 5798-CPR, is designed as a base for reporting programs (IBM or user-written). It helps summarize sequential activity traces like the IMS system log, IMS Monitor tapes, and GTF traces. It also contains facilities to print user-tailored graphs from any performance data log or non-VSAM sequential data set. GPAR is a prerequisite for ASAP II, VTAMPARS, and GTFPARS.

Monitoring IMS and participating z/OS components

With IBM z/OS Workload Interaction Correlator, IMS data is collected as part of unified workload insights across the entire z/OS middleware stack. The unified insights enable system administrators to reduce downtime without additional subject matter experts or specialized mainframe skills, which makes monitoring IMS less challenging and costly.

The collection of activities data can be summarized based on core type, job size, and job priory. This helps diagnose performance issues when underlying IMS problems are correlated with other z/OS components .

IMS registers with IBM z/OS Workload Interaction Correlator during initialization and delivers statistical data to it through a z/OS System Management Facility (SMF) exit. For every normally complete transaction, Message Processing Program (MPP) regions and Java message processing (JMP) regions record both elapsed time and CPU time in a standard form and in an IBM System z Integrated Information Processor (zIIP) form.

IBM z/OS Workload Interaction Correlator provides high-frequency IMS dependent region statistics. At each interval, activity statistics, and exceptional jobs and activities are recorded which you can use to further analyze performance and identify anomalies.

Related concepts

z/OS Workload Interaction Correlator (System Administration)

Chapter 25. Modifying the system design

This topic summarizes the planning activities needed when changes are necessary in the IMS system design.

The need for changes can result from:

- The integration of new applications
- The staged implementation of an application package
- Modifications to an application design already in production
- Modifications to the network if ETO is not used
- Maintenance of the IMS system at the current level and incorporation of fixes for application program problems
- Tuning of system control parameters
- Redistribution of database data sets
- Storage device changes

All of these changes can affect the IMS system definition and necessitate building a new nucleus and control blocks. The last two changes might only require JCL changes and database reorganization. However, the repercussions can be widespread. With any design change, it is important to control its implementation in order to protect the existing system and the needs of all end users. Implementation of most changes to a production system is best handled by prior testing and a carefully monitored validation period.

The task of administering a change to your IMS online system is likely to involve the following activities:

- Reallocating IMS system data sets and updating execution libraries
- Performing the appropriate system definition tasks
- Reviewing security arrangements, adding changes to RACF user, group, and resource profiles
- Documenting changes in operating procedures
- Documenting changes in recovery procedures
- Being aware of the testing strategy and the timing of cutover into production mode
- Analyzing the performance impact and revising predictions
- Coordinating the use of the IMS Monitor and otherwise monitoring the system before and after implementation
- Collecting comparative performance data

Assessing application changes

Because the design and tuning of an existing production IMS online system involves many people and much machine time, you need to minimize the impact of a change and identify as many potential ramifications as possible.

The following table summarizes how a change in an application program (or your control of the system) can be assessed for impact. It shows the IMS area affected and suggests what other administrative areas of responsibility might be affected.

If you are notified of an application program design change or are tracking a staged implementation, you must extract from the documentation the necessary physical system definition changes. You also need suitable detail to enable you to assess whether your defined environment and operational control are affected.

The first column of the table shows the kind of change and the second column shows the change event you isolate from the documentation. For example, an application system adds a batch message program to generate a report after closedown of a physical part of the network. The following information can be obtained:

Programs
Name and PSB, program size, and use of overlay

Databases
Already defined, read-only access

Transactions
None

Message formats
None

Output
Name of spooled output procedure

Terminals
None

Network Control
Whether any change in network availability

Scheduling
BMP region: time, frequency, limits

Exit routines
None

Tuning
N/A

Security
No RAS control

Working with this information, you define the APPLCTN macro, perform system definition (CTLBLKS), and tailor the BMP region JCL. Then you add its BMP scheduling instructions to the operations procedure, including a spooled output print step. Alternative strategies can be to perform a MODBLKS system definition and use online change for the cutover to production mode or to use the dynamic resource definition function.

Table 36. Application change control assessment

Change type	Change event	Areas affected	Impact and actions
Programs	Coding fix	IMS.PGMLIB	Documentation, log the change
	Database access	IMS.PGMLIB IMS.PSBLIB IMS.ACBLIB	Monitoring DL/I calls and using online change
	Addition	IMS.PGMLIB IMS.PSBLIB IMS.ACBLIB APPLCTN macro	Pools and resource contention, and using online change
Databases	Access	IMS.ACBLIB	Monitoring
	Structure	IMS.ACBLIB	Database reorganization
	Additions	JCL, IMS.ACBLIB DATABASE macro	Operations and recovery procedures, and using online change

Table 36. Application change control assessment (continued)

Change type	Change event	Areas affected	Impact and actions
Transactions	Change content	TRANSACT macro	Recalculate message queues, monitoring
	Added	TRANSACT macro	Scheduling algorithm, use of online change
	Workload	Message Queues	Recalculate blocking
	Message formats	IMS.FORMAT	Buffers and monitoring, and using online change
Output	z/OS file	JCL for devices	Operation procedures
	Spool print	LINEGRP, LINE macros	Operation procedures
	LTERM	TERMINAL macro	Alternative destinations, operational changes
Terminals	VTAM	System definition macros	VTAM generation, operating procedures
	ETO	IMS.PROCLIB	ETO generation
	Other	System definition macros	JCL for online, operating procedures
	Network control	System definition macros	VTAM initialization, MTO procedures
	Scheduling	TRANSACT macro	Message class algorithms operating procedures
Exit routines	DL/I	IMS.SDFSRESL	Monitor DL/I calls
	Message edit	IMS.USERLIB	Ensure against abnormal termination
	Security	IMS.USERLIB	Ensure security needs are met
	ETO	IMS.SDFSRESL	Monitor ETO
	User descriptors	IMS.PROCLIB	
Tuning	JCL parameters	IMS.PROCLIB	Coordinate operators change
	Security	EXEC parameters	Coordinate operations
	IMS.PROCLIB parameters	DFSDCxxx	Coordinate operations
	Signon	RACF, exit routines	Validate passwords and coordinate with nucleus
	Terminal	RACF, exit routines	Validate passwords and coordinate with nucleus
	Transaction	RACF, exit routines	Validate passwords and coordinate with nucleus
	RAS	RACF and exit routines	Validate passwords and coordinate with nucleus

Related reading: For more information about using DRD commands, see "The dynamic resource definition process" in *IMS Version 15.2 System Definition*.

Introducing changed applications in an active IMS system

When introducing a changed application program into an active IMS system, the **UPDATE PGM START(REFRESH)** command is useful for some region types.

By using the **UPDATE PGM START(REFRESH)** command, you can roll out the application program change easily without having to find out all of the regions that the program is scheduled in and stopping the regions manually.

The **UPDATE PGM START(REFRESH)** command is supported for programs scheduled in the following region types:

- MPP pseudo-wait-for-input (PWFI) regions in which the program is scheduled and the program is not preloaded by the DFSMPLxx PROCLIB member
- JMP PWFI regions in which the specified program name is scheduled
- MPP, JMP, and message-driven BMP regions in which the program is scheduled and that are running a transaction defined as WFI=YES

The **UPDATE PGM START(REFRESH)** command is not supported for the following region types:

- MPP regions where the program is loaded by the DFSMPLxx PROCLIB member
- IFP regions
- JBP regions
- Non-message-driven BMP regions

To ensure that a refreshed copy of a changed application program is loaded into these region types, follow these steps:

1. Issue the **/DISPLAY PGM** command to identify associated transactions that are using the application program that you are changing.
2. Issue the **/DISPLAY TRAN** command to identify the scheduling class for each transaction identified in step 1.
3. Issue the **/DISPLAY ACT** command to identify the dependent regions that are capable of processing the scheduling classes identified in step 2.
4. Stop the application program that you are changing by issuing the **/STOP PGM** command.
5. Refresh the program.
6. Stop and restart all dependent regions identified in step 3. Stopping and restarting these regions one at a time allows the remaining active regions to process transactions that are not associated with the stopped program during this process.
7. Start the changed application program.

Resizing the inactive ACB library online

Occasionally, the size of the ACB library needs to be adjusted. The methods of bringing the change online differ depending on how the data set is allocated; with JCL or dynamically using the DFSMDA macro.

When JCL is used for allocation, IMS must be brought down and restarted to recognize the changes. To avoid having to take IMS down to perform this task, use dynamic allocation for both the active and inactive ACBLIB data sets and then perform an online change process to bring the resized ACB libraries online.

Example

Due to rapid application development cycle, an ACB library might need to be enlarged frequently. If both the active and inactive ACB libraries are dynamically allocated using the DFSMDA macro, resizing of the ACBLIBs will not require an IMS outage.

Migrating static allocation of the ACB library to dynamic allocation

Create DFSMDA members to dynamically allocate the inactive ACB library and then restart IMS to use the new DFSMDA members.

To migrate to using the DFSMDA macro for dynamic allocation of ACBLIBs, complete the following steps:

1. Create DFSMDA members for the ACBLIBA and ACBLIBB data sets.
2. Remove the IMSACBA and IMSACBB DD statements from the IMS and DL/I JCL procedures.
3. Add a DD statement that points to the data set where the new DFSMDA members reside. The DD statement can be in either the IMS STEPLIB concatenation or IMSDALIB concatenation.
4. Stop IMS and then restart it with the DFSMDA members.

Now, you can perform the following tasks without stopping and restarting IMS:

- Increase the size of the inactive ACB library data sets.
- Correct errors in the inactive ACB library.
- Add additional data sets to the inactive ACBLIB concatenation.

Planning for system definition changes

The IMS online system design reflected in the macro specifications is subject to change. This design is rarely a one-time execution that occurs at initial installation.

The allocation of real resources and terminal connections change, if not because of hardware changes, then because of application workload and design changes. The ongoing task of monitoring for performance is likely to modify elements of the online system design.

Controlling system definition processing

When planning a change to system definition, you usually have to coordinate a series of macro changes. The IMS system definition stage 1 input is suggested as a point of control.

Review changes to the macros for the accuracy of the parameter values and add adequate explanation of how the value was derived. You should have a control document that records the nature of the proposed change, the reason for it, and suitable authorization. You can then complete the record by entering the action taken, or planned, for each item.

You must also coordinate the timing of the system definition stage 2 processing. The new version of the control program, with any required JCL changes, must be tied to an operational cutover.

Re-specifying a parameter on one or more macro statements does not require a full system definition or the processing of a complete stage 2 jobstream. You can make many tuning changes by overriding parameter values with others specified in the JCL statements. You can delay making the new values a permanent part of the control program until a more extensive change occurs in the macro content. Consider that all such overrides need to be coordinated with PROCLIB contents and operator instructions.

Determining the type of system definition required

Your requirement to perform a full system definition or a partial one depends on the macro statements altered. However, the modification of a particular keyword might be the only cause for full generation.

Because the amount of processing performed during a stage 2 execution is significant, you should plan to take advantage, wherever possible, of economics in the type of generation selected.

Related reading: For a summary of types of system definition to choose, see *IMS Version 15.2 System Definition*.

Changing the network definition in DB/DC and DCCTL environments

You can frequently change terminals that are connected to an IMS online system that have attendant buffer storage tuning. If you are introducing static VTAM terminals into the system, you must regenerate the nucleus. (An online system definition is necessary if no VTAM support was previously defined.) At the same time, you can delete support for specific terminal types.

If you are introducing VTAM terminals and using ETO, it is not necessary to regenerate the nucleus. When you add an ETO terminal supported by the existing descriptors, IMS uses the appropriate ETO logon descriptor to define the terminal.

When considering deleting terminals, leaving the definitions in place might be more efficient if the terminal is physically removed and messages cannot be routed to that LTERM. You can make the deletions in a later definition change.

If you are adding MSC for the first time or adding physical links, you need an online system definition. If you are deleting an MSC MTM, TCP/IP, or VTAM connection, you need to regenerate only the nucleus. If you are changing link or system identification names, rebind the regenerated control blocks.

Related tasks

Creating logon descriptors (Communications and Connections)

Extended Terminal Option (ETO) (Communications and Connections)

Making system tuning changes

Through ongoing monitoring and awareness of end-user feedback, you should be aware of potential tuning activities.

The tools and strategies for performance-related tuning activities are described in Chapter 24, “Collecting and interpreting IMS monitoring data,” on page 357, and Chapter 26, “Tuning the system,” on page 395. These sections assume that performance administration is an iterative task made up of:

Data collection
Using monitors and traces

Data reduction
Using report processors and manual procedures

Data analysis
Finding problem indicators

Change implementation
Proposing solutions and modifying the system

This plan should include a tuning and maintenance section. You can combine items from this section with other required changes to save additional system definition processing and reduce the frequency of changes to the operating procedures. An overall design change log enables you to keep track of the proposals that are analyzed and approved, as well as those under consideration for future implementation. A control document like this also assists in detecting conflicting changes.

The documentation for a change that affects the IMS online system design should include the reasons for the change and a checklist of what parameters or components in the system require modification. Ideally, the documentation should have sections on operations, database maintenance, recovery, security, and monitoring arrangements.

Resource utilization changes

Certain changes can promote a more efficient use of virtual storage and minimize I/O for an existing system.

These changes include:

- Re-specifying EXEC parameters for the control region
- Allocating DASD storage for message queues
- Database reorganization and data set distribution
- Putting ACBs into 64-bit storage

Application and database design changes

As a result of a performance study for an application package, you might want to make changes to the online system design.

These changes can range from a simple update of application programs in IMS.PGMLIB, to redefining DBDs and ACBs in IMS.DBDLIB and IMS.ACBLIB. If application programs are changed in IMS.PGMLIB, you must stop and restart the message processing regions to store the correct directory entry in the BLDL list, or use the **`UPDATE PGM START(REFRESH)`** command if applicable. If you can use this command for your application program change, see the **`UPDATE PGM`** command description.

Related reference

UPDATE PGM command (Commands)

Communication design changes in DB/DC and DCCTL environments

If you detect a bottleneck whose symptom is the transmission rate between a communication device and IMS, the response falls into the same planning as for network definition changes.

Managing online system definition changes

A request to add an application or to modify the current set of programs, transactions, and database usage need not force a complete system definition and an IMS restart, with possible interruption for the online users. You can examine the request to see if an online change or dynamic resource definition (DRD) process can satisfy the request.

If the request does not involve changes to the IMS network or the use of static terminals as defined in the current IMS system definition, you can use the online change process to make the changes while the IMS system is executing online. Manage the preparation of the IMS system data sets as described under "IMS System Data Sets for Online Change" in *IMS Version 15.2 System Definition*. It is also important to assess the impact to the terminal users and the online system operation.

Related reading:

- For a detailed description of using DRD to make system definition changes, see *IMS Version 15.2 System Definition*.
- For an overview of the online change function, see “Overview of the local online change function” on page 436.

Deciding if system modifications can use online change

A list of the changes that can be serviced by online change is given in the following table.

The list is based on the effect on the stage 1 system definition macros, because the stage 1 input is regarded as the control point.

Table 37. System definition resource modifications allowed for online change

System definition macro	Permissible online changes
APPLCTN	Add a PSB (application) and its attributes Change attributes Delete a PSB
DATABASE	Add a database and its attributes Change attributes Delete a database
RTCODE	Add a routing code and inquiry attributes Delete a routing code
TRANSACT	Add a transaction and its attributes Change attributes Delete a transaction

The changes listed in the table require a MODBLKS generation. The corresponding security changes require that you also update the RACF security profiles to reflect any new IMS resources needing security.

Also, the system definition preprocessor can be a useful part of your preparation. When adding or changing resource names, the preprocessor can detect any invalid names or duplicate names and help ensure a successful system definition run. Some limitations to the modifications are explained in "Planning considerations for online change" on page 392. You cannot make modifications that affect the terminal network.

Planning considerations for online change

When assessing whether a set of changes can be handled with an online change, you must take into account several limitations. In general, the checking performed by the stage 1 processing does not tell you if you have made a change that cannot be implemented online.

You need to consider effects of the following:

- APPLCTN macro

  If the message class is assigned as part of the PGMTYPE parameter, that class cannot exceed the maximum number of message classes currently defined for the system.

  If the transaction is designated for Fast Path, Fast Path must be active in the system.

  Routing a transaction to another IMS system requires that the system name (SYSID parameter) and the use of MSC are not newly defined for the MODBLKS generation.

  Although you can make changes to the RESIDENT and DOPT characteristics, PSBs defined as RESIDENT operate as nonresident until after the next restart, because the action of making PSBs resident takes place at IMS system initialization time.

  Changing the scheduling attribute to a resident PSB causes that PSB to become nonresident until the next IMS restart.

  If a BMP program becomes a message processing program, the transaction characteristics defined in the TRANSACT macro that control message scheduling do not take effect until after the next restart. However, the MTO can use the **/ASSIGN** command to specify appropriate message class and processing priorities for the particular transaction. The transaction then becomes eligible for normal message scheduling.

You can use the **UPDATE PGM** command to update both status and attributes. Some changes you make to the APPLCTN macro characteristics are not implemented as part of the online change processing and only become effective at the next cold start of the IMS online system. They are as follows:

TRANSTAT

- DATABASE macro in DB/DC and DBCTL environments

  Although the RESIDENT characteristic can be added, the process of making DMBs associated with the database resident does not take effect until after the next restart of the IMS online system.

  Changes to the ACCESS parameter are not part of online change; this change can be handled with the **/START DATABASE** command. Alternately, you can use the UPDATE DB START(ACCESS) command.

  You cannot include any kind of change to MSDBs.

- RTCODE macro

  The addition of this macro statement, or changes to its specification, is only allowed if Fast Path is active in the system. Make sure that the existing Fast Path User Input Edit routine is able to handle any added routing codes.

- TRANSACT macro

  You can control several of the characteristics specified by this macro using such commands as **/ASSIGN**, **/MSASSIGN**, **/START**, and **/STOP** You can also use UPDATE TRAN to update both status and attributes. Some changes you make to the TRANSACT macro characteristics are not implemented as part of the online change processing and only become effective at the next cold start of the IMS online system. They are as follows:

  PRTY
  PROCLIM
  PARLIM
  SEGNO
  SEGSIZE
  SYSID
  TRANSTAT

  Transactions designated as Fast Path potential need Fast Path to be active in the current system.

  Routing a transaction to another IMS system requires that MSC facilities be active in the current system. You cannot introduce a system name (SYSID parameter) that was not previously defined in the current system.

  Edit exit routines specified for the transaction must already be part of the current IMS online system.

- Page fixing

  No additional page fixing is done for added control blocks until the next restart of IMS.

- EMHB size

  If you use online change to add or change a transaction-specific EMHB size, ensure that the new EMHB size is not larger than the EPSESRT size. The EPSESRT size is determined only during initialization.

  During normal transaction processing, IMS checks the size of the input message against the EMHB length and the EPSESRT length. If the input message exceeds either the EMHB length or the EPSESRT length, it is rejected with message DFS0444.

Performing capacity planning

Other topics discuss short- or medium-range planning for changes in the processing requirements of your IMS online design. However, you might want to examine the long-term adequacy of your design.

For example, if you extrapolate trends in workload, will enough computing power be available?

You can use:

- Individual transaction profiles
- An overall processing profile
- An estimate of percentage processor utilization

As a result of your performance monitoring, you can establish transaction profiles. One part of these profiles can be a trend analysis. You must determine mean and peak arrival rates: currently predicted, sample actuals, and future expectations. The overall processing summary can be taken from the IMS Monitor Run Profile report.

One way to investigate limiting capacity is to extrapolate the volume of transactions. You can establish a trend from percentage processor usage for several monitor points. A common observation from performance measurements indicates that the increase in machine cycles is nearly linear with the increase in transactions per second on systems dedicated to IMS. If you plot several points, showing increasing transaction loads against percentage utilization, you can extrapolate to find your limiting number of transactions as the percentage utilization approaches 100%.

You can also see if a predicted maximum workload corresponds to a utilization that is below a practical upper limit. The workload is estimated by weighing the current overall profile with additional transactions.

In making such predictions, you should have adequate virtual storage and an I/O configuration sufficient to handle the increased transaction load. A careful consideration of remaining available I/O and virtual storage resources is necessary, because contention for these elements has a more severe impact on performance than the level of processor utilization.

If you are managing a large system running under z/OS, you might want to reduce your requirements for CSA. You can plan to use the DL/I separate address space and, if necessary, local storage for IRLM for lock management. Using your projected workload, you might still want to estimate virtual storage with various configurations.

Chapter 26. Tuning the system

You can make changes to parameters that allow the IMS online system to execute more efficiently.

Specifically, these changes to parameters allow IMS to:

- Attain performance criteria
- Efficiently use resources

Chapter 24, "Collecting and interpreting IMS monitoring data," on page 357 explains setting up performance objectives and monitoring, which are prerequisites to tuning activities for the IMS online system. Terms defined in that section are not generally explained again. Your data for analysis probably depends on the reporting capabilities of one or more of the following tools:

- The Log Transaction Analysis utility (DFSILTA0)
- The Statistical Analysis utility (DFSISTS0)
- The IMS Monitor
- Report programs executing under GPAR: IMSASAP II
- Resource Monitoring Facility (RMF II)

This section presents a general strategy for tuning the IMS online system. After some explanation of performance management, the topics in this section approach tuning in two stages:

- Initializing z/OS and IMS parameters
- Identifying and correcting performance problems

For each activity, several major areas of concern are identified. You should examine the potential tuning activities in each area. This is an outline strategy and not a substitute for more formal performance studies. However, this information should prepare you to address most major problems.

Tuning the system involves initializing z/OS and subsystem parameters and identifying and correcting problems as they relate to tuning. These two topics can be characterized as guidance to reduce the chance of initial problems and a methodology for ongoing problem resolution. However, you should begin by understanding how they relate to the other aspects of the performance management process.

Tuning actions and system redefinition are two aspects of IMS performance management that are part of a continuous cycle of activities, as illustrated in the following figure. Interpreting system performance includes both design decisions and prediction.

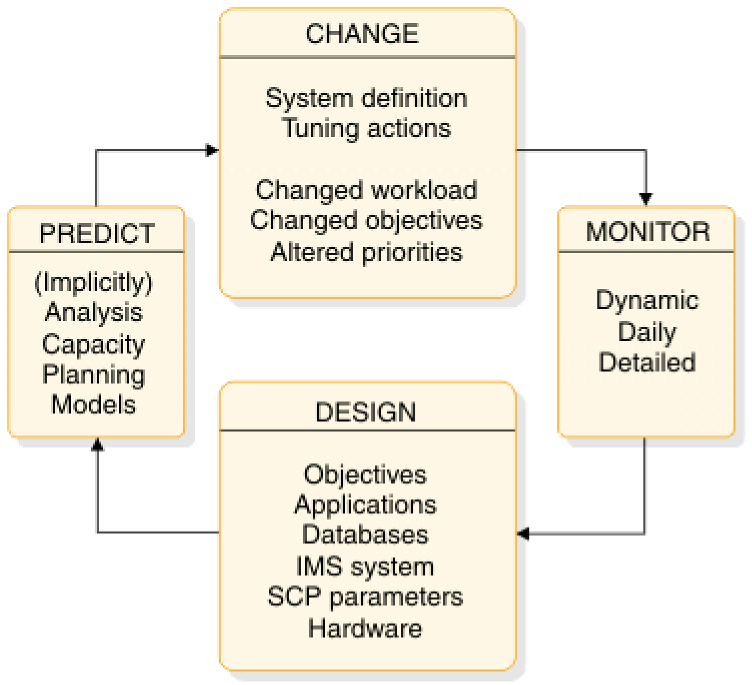


Figure 46. Activity cycle for IMS performance management

Related reading:

- For information about monitoring and tuning MSC links, see *IMS Version 15.2 Communications and Connections*.
- For information about tuning Fast Path, see *IMS Version 15.2 Database Administration*.
- For information about tuning a shared-queues environment, see "Tuning for performance in a shared-queues environment" on page 220.

Managing performance

IMS performance management is made up of a set of activities, designed to achieve a set of goals.

The goals consist of:

1. Establish performance objectives for each subsystem (IMS, TSO, Batch) and their relative priorities.
2. Initialize MVS and subsystem parameters with a view to meeting the defined performance objectives.
3. Monitor the operation of the total system to confirm that performance objectives are being met.

   The three broad levels of monitoring are:

   - Dynamic monitoring to discover if performance deviates from established objectives
   - Daily monitoring of key performance objectives in order to observe trends and early indicators of possible performance problems
   - Detailed monitoring performed periodically to review performance and decide on tuning requirements

   For more information about these levels of monitoring, see Chapter 24, "Collecting and interpreting IMS monitoring data," on page 357.
4. Identify a problem that has been monitored and correct it with minimum disruption of normal service to users.
5. Perform capacity planning or trend analysis to ensure that projected loads will not cause performance problems in the foreseeable future.

Establishing performance objectives and monitoring operations are addressed in "Monitoring the system" on page 341. Performing capacity planning or a trend analysis is described in "Performing capacity planning" on page 393 and in "Using the IMS Performance Analyzer for z/OS" in *IMS Version 15.2 Operations and Automation*.

Change management

Performance management is really the management of change. Performance is stable if few changes occur in an IMS system.

More common is a continually changing environment where the IMS system must be adjusted to the addition of new application programs and respond to changes in:

- Application design
- Transaction volumes
- Database volumes
- Workload mix
- User population
- User priorities
- Volumes of non-IMS work
- Hardware configuration

IMS tuning attempts to allocate resources for efficient transaction processing.

You can use various techniques to tune the system during operation. For example, you can:

- Start new message regions
- Change transaction PARLIM, PROCLIM, or class assignments
- Alter message region class assignments

These activities involve ensuring that sufficient IMS resources, such as regions, pool space, or control block areas, are provided. The IMS configuration needs to be matched to the installation's physical resource constraints. Typically, the use of virtual storage is exchanged for some I/O activity controlled by IMS, and contention is reduced for data stored on DASD devices. Tuning should not be crisis management but a continual compensation for changes or workload trends.

A systematic approach to tuning is included in "Identifying and correcting performance problems" on page 411.

Design variables

The performance of an IMS system depends on the interaction of the workload with many design variables.

Performance is affected by:

- IMS system definition
- IMS execution-time options and parameters
- Application design
- Database design
- Operating system configuration
- Hardware configuration

These areas are addressed in "Initializing z/OS and IMS parameters for tuning" on page 398. Design aspects also include allowing for design and tuning changes decided upon as a result of monitoring the system in operation.

Types of prediction

Two aspects of prediction exist: implicit and explicit prediction.

Implicit
When an alteration, classified as a tuning change, is made to the online IMS design, a performance improvement is implied.

Explicit
When actions are taken to make predictions, analytical or simulation models are used, and capacity planning is performed.

Initializing z/OS and IMS parameters for tuning

The IMS and z/OS parameter initialization steps taken reflect the actions that can have a performance impact.

These parameters must be set while considering other aspects. For example, if design or function requirements that contradict or override these recommendations exist, you must evaluate the alternatives.

Initializing z/OS and IMS parameters involves the activities explained in the following topics.

Assigning z/OS dispatching priorities

Choose a coordinated dispatching priority scheme for the total system. The scheme must reflect the relative worth of any subsystems that are to execute concurrently.

The IMS control region operates in protect key 7 as a system task. This reduces the number of executable instructions for some supervisory functions.

Job dispatching priority

The job dispatching priority dictates the sequence jobs are given to available machine cycles. Any job with a dispatching priority higher than the control region causes interference. Although VTAM or TCAM normally runs at high priority (as do JES and the z/OS Master Scheduler), you should review the position of other work (principally TSO and batch) in relation to the IMS control region and the dependent regions.

Dependent region dispatching priority

It is important that both the control region and the dependent regions obtain priority processing. The parallel DL/I function in IMS enables most of the DL/I call processing to be performed in the dependent regions. If the assigned dispatching priority is too low, the promptness of service to application programs declines. Usually, you set the BMP region priority lower than MPP priorities. The BMP priorities should, however, immediately follow the MPPs, because they contend for the same control region resources, such as program isolation enqueues. Normally, TSO "first period" work is placed below the IMS control region; it might be above the dependent regions. TSO "second period" work and batch are normally placed below the IMS dependent regions.

Setting a performance group number under z/OS

You can assign a unique performance group number to each IMS region. In this way, you can obtain Resource Measurement Facility (RMF) report statistics separately for each region.

If the DL/I separate address space option is used, you might decide to include both the control and DL/I separate address spaces in the same RMF performance group.

Setting priorities in a DBCTL environment

For a DBCTL environment, the CCTL region should have a higher priority than the DBCTL region, which should have a higher priority than any batch processing regions.

CICS might run at a higher priority than the DBCTL region. Adjust the priorities of the DBCTL and CICS regions depending on the workload and resources available to your system.

Dispatching priorities

Dispatching priorities are set by the z/OS Workload Manager (WLM) and are based in part on the relative importance you assign in the WLM ISPF panels to each IMS address space.

Related reading: For more information on assigning WLM importance, see "Workload Manager and IMS" on page 359.

The following figure shows an example WLM ISPF display of the relative dispatching priorities assigned to all address spaces known to WLM. The dispatching priority of each address space appears as a hexadecimal value in column DP, with X'FF' being the highest priority. Address spaces with a lower value have a lower priority.

```
 NP   JOBNAME  StepName ProcStep JobID    Owner    C Pos DP Real Paging 00000100
      ALLOCAS  ALLOCAS                               NS  FF  806   0.00 00000110
      IOSAS    IOSAS    IEFPROC                      NS  FF  256   0.00 00000120
      IXGLOGR  IXGLOGR  IEFPROC                      NS  FD  803   0.00 00000130
      SMS      SMS      IEFPROC                      NS  FE  329   0.00 00000140
      SMF      SMF      IEFPROC                      NS  FF  363   0.00 00000150
      JES2AUX  JES2AUX                               NS  F9  120   0.00 00000160
      IRL2194  IRL2194  IRLM     STC00312 ++++++++   NS  FD  283   0.00 00000170
      NETMONC4 NETMONC4 SYSTEM   STC00005 ++++++++   NS  FB  240   0.00 00000180
      JES2     JES2     IEFPROC                      NS  F9 3095   0.00 00000190
      VTAM94   VTAM94   IEFPROC  STC00002 ++++++++   NS  FD 2492   0.00 00000191
      SOFIMSP  SOFIMSP           STC00006 ++++++++   LO  FF  226   0.00 00000192
      MPC4A04  REGION            JOB00323 USRT008  A NS  F3  385   0.00 00000200
      MPC4A05  REGION            JOB00324 USRT008  A NS  F3  385   0.00 00000300
      MPC4A06  REGION            JOB00325 USRT008  A NS  F3  385   0.00 00000400
      MPC4A07  REGION            JOB00326 USRT008  A NS  F3  385   0.00 00000500
      MPC4A08  REGION            JOB00327 USRT008  A NS  F3  385   0.00 00000600
      SCIC94   SCIC94   IEFPROC  STC00313 ++++++++   NS  FD  460   0.00 00000700
      OMC94    OMC94    IEFPROC  STC00314 ++++++++   NS  FD  442   0.00 00000800
      CQSC94   CQSC94   IEFPROC  STC00315 ++++++++   NS  F7 1052   0.00 00000900
      RMC94    RMC94    IEFPROC  STC00316 ++++++++   NS  FD  507   0.00 00001000
      I91CTLC4 I91CTLC4 IEFPROC  STC00317 ++++++++   NS  F7  12T   0.00 00001100
      I91DLSC4 I91DLSC4 IEFPROC  STC00318 ++++++++   NS  F5  27T   0.00 00001200
      I91DBRC4 I91DBRC4 IEFPROC  STC00319 ++++++++   NS  FD  617   0.00 00001300
      BPXOINIT BPXOINIT BPXOINIT                     IN  FF  162   0.00 00001400
                                                                        00001500
```

Figure 47. Relative dispatching priority (column DP) displayed in the WLM ISPF panel

Choosing IMS options for performance

Performance implications are associated with many parameters. Some parameters are chosen for system definition, some for execution-time options, and some are an integral part of design decisions.

The following list discuss many of the options.

- Optimizing IMS and application module loading

  Where possible, unless availability of virtual storage is a constraint, use the pageable link pack area (PLPA) for IMS reentrant modules, region and program controllers, frequently used high-level language modules, any reentrant application code, and the overlay supervisor (if used).

  Use the execution-time parameter SRCH=1 in the IMS procedure to cause the job pack area and link pack area to be searched before STEPLIB or JOBLIB.

- Program library considerations

  Place a program library containing the production programs first in the STEPLIB concatenation for the message regions. The sequence of programs in IMS.PGMLIB should be in descending order based on frequency of use.

  The order of the system search for program libraries can account for some inefficiency in program load. STEPLIB or JOBLIB is the first program library searched. If one of these is not used, IMS.PGMLIB should be first in the LNKLSTnn member of SYS1.PARMLIB. The placement of IMS.PGMLIB is far more important than IMS.SDFSRESL, because its contents are in continual demand for application program scheduling rather than being in demand only at system initialization. The search and load times are significant because they become a performance penalty each time the program is scheduled. You should block programs to full track size in the library. The DC option of the linkage editor, used for downward compatibility, causes program block sizes to be 1024 bytes. You should bind again without this option.

- Dependent region BLDL list

  A list of entries is maintained in the dependent region containing the directory index for programs. It is maintained on the most active, most recently used basis, and reduces the I/O to the program directory. Programs with entries in this list have a lower schedule-to-first DL/I call elapsed time than infrequently used programs. You can override the default of 20 entries for the message processing region by using the DBLDL parameter in the EXEC statement of the DFSMPR procedure. The maximum number of entries for the message processing region is 9999.

  For a region that is used to test new or changed programs, set the DBLDL parameter to 0, ensuring that the most current version of the program is loaded for each execution. Specifying the DOPT parameter disables quick reschedule.

- Application control block placement

  To reduce the amount of read I/O to the ACBLIB data set, you can load the ACB members into 64-bit storage by specifying ACBIN64=ggg in the DATABASE section of the DFSDFxxx PROCLIB member. The value specified (ggg) is the amount of 64-bit storage (in gigabytes) to be allocated for PSB and DMB ACB members.

  When the ACB members in 64-bit storage function is enabled, the ACB members are retrieved from 64-bit storage at application scheduling time instead of from the ACBLIB data set.

- Application program control

  Specify MODE=SNGL on the TRANSACT macro statement to reduce message queue I/O activity, and response time.

  For the batch message program execution controlled by a time limit, select the IMSBATCH procedure parameter STIMER=2. This causes the STIMER/TTIMER macro to be issued only once per schedule, rather than once per DL/I call with STIMER=1. Similarly, use the value 2 for the positional parameter STIMER for the DFSMPR procedure when controlling message processing regions.

  For the IMSBATCH procedure, use parameter values of SPIE=0 and TEST=0 for production programs to eliminate unwanted SVCs. The equivalent positional parameters for the DFSMPR procedure are SPIE and VALCK.

  With PL/I, use the latest release of the optimizing compiler to reduce initialization and termination overhead.

- Setting checkpoint frequency

  Adjust the checkpoint frequency so that checkpoints are not taken more frequently than once every 10 to 20 minutes. You control this frequency either by using one of the following methods. If you do not specify a checkpoint frequency, IMS uses a default value of 500,000

  - By specifying the **CPLOG=** parameter on the DFSPBxxx member of the IMS PROCLIB data set
  - By specifying the **CPLOG=** parameter in JCL
  - By issuing the **/CHANGE CPLOG** command

- Message format options

  Specify FILL=NULL or FILL=PT in DOFs that have many DFLDs to minimize the transmission of blank characters.

- Setting queuing options

  If you do not want priority processing, you can use the IOS queuing option to set priority processing off. The IOS option causes IMS to process all jobs in the queue on a "first-in, first-out" basis.

- Page fixing IMS resources

If you want to ensure that the IMS virtual storage is always backed by real storage, you can specify a list of page-fix requests in the member DFSFIXxx in IMS.PROCLIB. The page fix is done during the IMS control region initialization. Long-term page fixing for the queue manager buffers can be requested using the EXVR parameter in the EXEC statement of the control region JCL.

You tune buffer pools to make them large enough to reduce or minimize I/O activity without driving up the system paging rate or constraining virtual storage. Considerations for page fixing buffer pools are given in "Page fixing the IMS buffer pools" on page 406.

- Defining IMS resources for DREF storage in DB/DC and DCCTL environments

  You can request that the IMS virtual storage never migrate to auxiliary storage. By specifying DREF requests in the DFSDRFxx member in the IMS PROCLIB member, you ensure that virtual storage does not move beyond expanded storage. If expanded storage is not available, the storage is page fixed.

  Related reading: For more information about DREF requests, see *IMS Version 15.2 System Definition*.

- MVS page fix considerations

  MVS modules that are frequently referred to by the IMS system should be page fixed at initial program load.

  Page fixing certain modules also saves a "Page fix and Wait" SVC for each occurrence of timer facilities. Place these modules close together to minimize the number of pages in the pageable link pack area (PLPA).

  Related reading: For more information on MVS modules, see *z/OS MVS Initialization and Tuning Guide*.

Avoiding contention for IMS resources (excluding buffer pools)

The interleaved processing of concurrent regions is a major area of contention for IMS resources. A set of regions processing a mix of transactions can overlap processing and use the advantage of multitasking over serially ordered execution.

The following list identifies several areas concerned with the pattern of scheduling and region efficiency:

- Using class scheduling in DB/DC and DCCTL environments

  The following points affect your decisions about using class scheduling:

  – Use classes to separate low-priority and long-running transactions from high-priority work.
  – Arrange your class structure so that any high-priority class work has the opportunity of being scheduled in more than one region, unless doing so is undesirable for some special reasons, such as program isolation conflicts. Multi-server systems generally produce lower waiting times than single-server systems.
  – To prevent message regions from being idle while transactions are waiting on the input queue, assign additional (lower-priority) classes to a message region below the primary class or classes assigned to the region. Avoid regions that only service one class, unless you are sure that work of that class will always be waiting to be processed.
  – For very high-volume transactions, dedicate a message region and classify the program as a wait-for-input program to reduce input queuing time and to eliminate scheduling and program load.

- Processing limit counts in DB/DC and DCCTL environments

  Review your transaction class and priority scheme. Set the PROCLIM parameter on the TRANSACT macro to ensure that no transaction type can monopolize a message region if other transactions are waiting and are eligible for processing in that region. Only use a PROCLIM value higher than 5 if the response times of contending transactions are unimportant. In that case, set PROCLIM=1 for the lower-priority transactions. With wait-for-input transactions, set PROCLIM to a suitably high (or maximum) value so that you avoid rescheduling the application program. Use pseudo WFI and quick reschedule to eliminate processing overhead. By going through quick reschedule, a region can process more messages per schedule than the processing limit, thereby eliminating unnecessary rescheduling and reloading of application programs.

Note: A region scheduled against a transaction whose processing limit count is 0 cannot go through quick reschedule or become a pseudo WFI.

- Parallel scheduling in DB/DC and DCCTL environments

  Do not use parallel scheduling for low-volume transactions that can be scheduled sequentially. Using message regions for parallel processing can reduce the efficiency of regions in which multiple transactions are processed for each scheduling of a program. Parallel scheduling should be reserved for those transactions where temporary peaks in the arrival rate might cause excessive queuing to develop due to "flooding" of a single message processing region. In these situations use a PARLIM level geared to the average service time and response time objective for the transaction type.

- IMS message processing regions in DB/DC and DCCTL environments

  Operating your IMS system with too few or too many message processing regions can affect performance and transaction response times. In an environment with many different transaction types, regions with a very high occupancy level must be carefully monitored to ensure that they are not causing queues to build up on the IMS message queues.

  In environments with only a few transaction types, high levels of occupancy can usually be sustained, because the application programs can often remain in the region processing multiple transactions for each schedule.

  The meaning of high occupancy depends on the transaction mix, transaction characteristics, class scheduling scheme, and response-time objectives. For some transactions, a large queuing delay might be acceptable. For others, a much smaller queuing delay must be achieved in order to satisfy the user.

  In contrast, having many low-occupancy concurrent message processing regions can unnecessarily increase the contention among regions for buffer pool space, real storage, IMS system data sets, and database records (because of program isolation enqueuing).

  You can eliminate some of the processing overhead by specifying pseudo wait-for-input (PWFI) on the MPP region startup procedure. A PWFI message processing region remains scheduled when no more messages need to be processed, instead of terminating the application program. This eliminates unnecessary termination and rescheduling.

  Adjust the number of message processing regions to the minimum possible consistent with achieving your response-time objectives. Increase the number of regions only when an increasing workload cannot be serviced within the objectives, and when changes to the message class structure, quick reschedule, PWFI, and region control parameters (including PARLIM, PROCLIM, and WFI) do not effectively contain the response-time increases. Although you can specify up to 999 dependent regions, always try to contain the response time demands with a minimum number.

  The number of required regions, excluding wait-for-input regions, can be estimated as shown in the following example.

  ```
  number of regions =
          ((arrival rate)*(time in region))/(desired region percentage occupancy)
  ```

  arrival rate is the total number of transactions received in an interval divided by that interval in seconds. *time in region* is defined as the sum of the mean times for: scheduling and termination, schedule-to-first DL/I call, and elapsed execution, taken from the IMS Monitor Region Summary report.

  Do not include regions dedicated to wait-for-input processing in this calculation, because their region occupancy is 100%.

  In addition to establishing the number of message processing regions, you control which transactions can be scheduled into those regions by:

  – The choice of class and priority structure

  – The PROCLIM and PARLIM parameters on the TRANSACT macro

- CCTL regions in DB/DC and DBCTL environments

  CCTL application program performance can be affected by contention for database resource adapter (DRA) resources. Specifically, if the maximum number of DRA threads allowed is less than the number of CCTL application programs making PSB schedule requests, some of those requests wait until a DRA thread becomes available.

- Save area sets in DB/DC and DCCTL environments

  When you consider the number of regions and their occupancy for a given transaction rate, check that the input data transmission is not delayed. One factor causing delay might be the limiting number of concurrent terminal I/Os. You can adjust this limiting number using the SAV parameter in the online EXEC statement (up to a maximum of 999 concurrent I/Os). This parameter is termed the number of "dynamic save area sets" for the terminal I/O, because one of these areas controls each active terminal I/O.

- Database contention in DB/DC and DBCTL environments

  Avoid databases with a small number of roots or root anchor points. If the databases are accessed by more than one transaction concurrently, program isolation (PI) contention is probable. The use of control records in databases or of records that keep running totals also causes PI enqueuing.

  If you cannot avoid databases with a small number of roots, use class scheduling to allocate all such contending transactions to a single region, thus forcing them to process sequentially.

IMS buffer pool tuning

You can configure the IMS buffer pools to improve overall system performance.

Message queue pool tuning

The message queue pool acts as a buffer to the message queue data sets. Optimize the message queue pool to reduce unnecessary I/O activity.

If message queue I/O occurs during transaction processing, the message queue data sets can be detected as IWAIT items on the IMS Monitor Region IWAIT report. If the LRECL sizes of the short and long message queue buffers are not in correct proportion to each other, I/O can occur, because the mix of transactions uses up one or the other section of the buffer even though the size of the pool seems large enough.

The number of buffers is specified in the MSGQUEUE macro. You can override the number at execution time with the message queue buffer (QBUF) parameter in the EXEC statement. One tuning technique is to decrease the number of buffers until the monitor data shows increasing IWAIT instances marked as message queue I/O. Then you can use the QBUF parameter value plus one page-size unit.

You might want to identify unusual demands on message queue handling. Such demands can include:

- Large multi-segment input and output messages
- Extended processing between segment inserts
- Remote print output
- Terminal operator paging
- Multi-terminal output
- Excessive numbers of program-to-program switches
- Zero-priority messages for BMPs
- SPA sizes for conversational transactions

SPA sizes can adversely affect the performance of the message queues, detracting from the efficiency of single-segment messages that are used as input to well-designed application program processing.

If the security-data section of the OTMA message prefix contains network security credentials, usage of the message queue pool is increased.

The message queue LRECL size must allow for the SPA size, because the SPA is placed on the queue as the first segment of the message. The SPA segment can be split into multiple logical records when it is written to the IMS.LGMSG data set, or a larger LRECL size can be specified. This choice depends on the relative number of transactions requiring large SPA sizes.

You can use the Log Transaction Analysis utility to monitor the I/O queue times. The IMS Statistical Analysis utility produces a Line and Terminal report showing the average size sent and received, but does not show separate long and short message statistics. Average message lengths and counts for each transaction and terminal are part of the information in the system log type X'40' checkpoint records.

Related reference

Statistical Analysis utility (DFSISTS0) (System Utilities)

Log Transaction Analysis utility (DFSILTA0) (System Utilities)

DFSFIXnn member of the IMS PROCLIB data set (System Definition)

Message format pool tuning

The message format buffer pool holds message format blocks for frequently used transactions. Optimizing it reduces the I/O burden of common transactions.

To obtain storage in the message format buffer pool, the space holding the least recently referenced format block is freed so that blocks for transactions with high arrival rates are usually available. To minimize format block I/O, plan to have sufficient space in the pool to hold the input and output format block pairs for use by your most frequently used transactions

Using fetch request elements (FREs) significantly affects efficient use of the pool. Each FRE controls one format block in the pool. If all the format blocks are 1000 bytes long and 10 FREs are specified, the maximum pool space is 10000 bytes. FRE assignments are made in the BUFPOOLS macro during system definition, but can be overridden by the FRE parameter in the control region EXEC statement. You can use the **`/DISPLAY POOL MFP`** command to compare the amount of free space to the amount of total space. If the free space is consistently high, relative to the size of the pool, make sure that the FRE assignment is not preventing full use of the allocated storage.

Another way to improve the use of the message format buffer pool is enable application programs to share formats. For example, a generalized message input format can share common DIF/MID blocks with subsequent editing performed by Input Message Field Edit and Input Message Segment Edit exit routines.

A performance-related function is available with Message Format Service (MFS). You can use the MFS Service utility to generate a directory of direct pointers to the location of all or a selected subset of the format blocks. This index is resident in the message format buffer pool and minimizes the I/O required to look up the location of a format block in the active IMS.FORMATA/B library directory. (Each index entry requires 14 bytes.) Evaluate the trade-off between allocating index storage for at least the highly referenced blocks against the reduction of I/O per block.

Related reference

MFS Service utility (DFSUTSA0) (System Utilities)

DFSFIXnn member of the IMS PROCLIB data set (System Definition)

PSB, PSBW, and DMB pool tuning

These guidelines for the program specification block (PSB) pool, PSB work (PSBW) pool, and data management block (DMB) pool can resolve pool space failures.

PSB pool and intent list considerations

The PSB pool holds all PSBs for scheduled application programs. A PSB remains in the pool until space is needed to load another PSB. Then, if all space is used, the least-referenced inactive PSB is freed. PSBs that are made resident and are not parallel scheduled are exceptions to this rule. These PSBs do not take from the pool allocation and are usually designated for highly referenced (usually preloaded) programs. You can page fix resident PSBs by including load list area DFSPSBRS in the module fix list coded for member DFSFIXxx in the IMS.PROCLIB data set.

If the DL/I address space option is used, two resident PSB pools exist, one in the z/OS common area (DFSPSBRS) and one in the DL/I address space (DFSDLIRS).

Similarly, resident-intent lists can be page fixed with the inclusion of DFSINTRS in the DFSFIXxx member. Make intent lists resident.

Storage for the PSB work pool

A portion of the PSB work pool is required by each active PSB, but the total size need not be larger than the maximum.

When you use the technique of starting large and reducing to some value greater than the maximum amount used, scheduling stops when any pool space failures occur.

DMB pool considerations in DB/DC and DBCTL environments

Make the DMB pool large enough to contain all IMS database DMBs. Each time you load a DMB into the pool, the database must be opened, and another database must be closed to free space. This activity increases the elapsed time that is required for DL/I calls. Limited page faulting is preferable to using OPEN and CLOSE. Make DMBs for active databases resident. These DMBs are good candidates to be page fixed. If all of the DMB pool is not being used, reduce the size until IWAIT for DMBs begin to occur. The IMS Monitor Region IWAIT report shows the DMB pool size is too small by the presence of DMB= entries. You can then increase the specification of the DMB parameter on the BUFPOOLS macro.

PSB pool, PSB work pool, and DMB pool space failures

Pool space failures occur when space in any pool is insufficient to sustain the current number of concurrently executing dependent regions. Each application program requires its PSB, the PSBW, the intent list, and the DMBs to be scheduled. Ensure that all pools are large enough to eliminate pool space failures. The general rule is to allow sufficient space for the (2R + 1) largest blocks that are part of the set DMB, PSB, or PSBW as appropriate, where R is the number of active dependent regions. This guidance allows for fragmentation in the pool space. Pool space requirements can be reduced by eliminating unnecessary PCBs from PSBs.

Related concepts

"IMS internal resource usage" on page 672
There are several summary reports that you can use to examine the level of internal contention for resources.

Related reference

DFSFIXnn member of the IMS PROCLIB data set (System Definition)

Database buffer pool tuning

Allocate sufficient buffers in the IMS database buffer pools to prevent I/O that results from an application program having to reread data previously brought into the pool. Careful choice of subpool sizes and matching against database block sizes and frequency of database references minimizes unnecessary I/O.

For Fast Path buffer pools, you can either explicitly specify the buffer pool size as part of the system definition process (DBBF parameter), or you can use the Fast Path 64-bit buffer manager to control the number and size of the Fast Path database buffer subpools. To enable the Fast Path buffer manager, specify FPBP64=Y in the FASTPATH section of the DFSDFxxx member of the PROCLIB data set. When the Fast Path buffer manager is used, DEDB buffer pools are placed in 64-bit storage, while buffer pools for main storage databases (MSDBs), sequential dependent (SDEP) segments, system services, and buffer headers are managed in 31-bit storage.

Multiple OSAM subpools can be defined as having the same buffer size, and specific data sets can then be directed to specific subpools. If you have many OSAM data sets with similar or equal block sizes, you might be able to obtain some performance advantage by replacing a single large pool with separate subpools. Multiple smaller subpools reduce pool scanning to locate a segment. This method can also be used to prevent low-priority transactions (or BMPs) that access many database segments from monopolizing pool space.

Tip: The time required to sequentially process OSAM databases can be considerably reduced by using Sequential Buffering (SB).

You can define multiple VSAM local shared resource pools. If you have many VSAM data sets with similar or equal control interval sizes, you might get a performance advantage by replacing a single large subpool with separate subpools of identically sized buffers. Creating separate subpools of the same size for VSAM data sets offers benefits like those offered by OSAM multiple subpool support.

Use multiple local shared resource pools to specify multiple VSAM subpools of the same size. Create multiple shared resource pools and then place a VSAM subpool in each one that is the same size as other VSAM subpools in other local shared resource pools. You can then assign a specific database data set to a specific subpool by assigning the data set to a shared resource pool. The data set is directed to a specific subpool within the assigned shared resource pool based on the control interval size of the data set.

You can also create separate subpools for VSAM KSDS index and data components within a VSAM local shared resource pool. Using these subpools can be advantageous, because index and data components do not need to share buffers or compete for buffers in the same subpool.

Multiple VSAM local shared resource pools enhance the benefits provided by Hiperspace buffering. Use Hiperspace buffering to allocate more buffers of 4 KB and multiples of 4 KB by using expanded storage in addition to virtual storage. Using multiple local shared resource pools and Hiperspace buffering allows data sets with certain reference patterns (for example, a primary index data set) to be isolated to a subpool backed by Hiperspace, which reduces the VSAM read I/O activity needed for database processing.

If you are using High Availability Large Databases (HALDBs), you can also direct HALDB partition data sets to specific OSAM or VSAM subpools.

Related concepts

OSAM sequential buffering (Database Administration)

Hiperspace buffering (Database Administration)

Fast Path buffer performance considerations (Database Administration)

Related reference

DFSVSMxx member of the IMS PROCLIB data set (System Definition)

Trade-offs between I/O controlled by IMS and paging

When you are choosing between I/O controlled by IMS and paging, you should consider several trade-offs.

General buffer pool considerations

Small tuned pools in IMS tend to minimize paging. In addition, because paging is not controlled by the IMS system, it can escalate dramatically in a storage-constrained environment.

In contrast, IMS buffer pool I/O is controlled and can be estimated and measured against those estimates. Finally, using a page fault and page I/O operation is generally more costly in processor terms than using a BSAM or IMS OSAM I/O.

Therefore, optimization of IMS in this area requires adjusting pool sizes while keeping in mind the costs in terms of paging and path length of scanning a pool.

Page fixing the IMS buffer pools

Information for initial optimization of IMS buffer pools is explained in "IMS buffer pool tuning" on page 403. Apply these guidelines to your system to achieve a minimum value for the total I/O operations (pool I/O and paging I/O). This means that you must consider a trade-off between I/O and storage across all pools, as well as within a single pool. For example, halving the maximum size of a large PSB pool might result in a very small increase in I/Os, but giving that extra storage to the message queue or format buffer pools might result in reducing even more I/O saving.

The way you tune your pools depends upon your real storage environment. In general, increasing maximum pool sizes increases paging; reducing pool sizes reduces paging of those areas.

If the IMS pools are subject to paging and IMS is sharing the processor usage, you can consider page fixing the significant buffers necessary for message and database access. This can be useful in a non-dedicated IMS system with very low-transaction rate, where IMS might be paged out unless the pools are page fixed. In heavily loaded IMS systems, response times might benefit from page fixing the data communication pools, because they hold data over the longest periods.

The principal IMS pools that can be page fixed are:

- Message queue pool (QBUF)
- DMB pool (DLDP)
- PSB pool (DLMP)
- Message format buffer pool (MFBP)
- Database subpools

If the DL/I address space option is used, two PSB pools exist: DLMP in the z/OS common area and DPSB in the DL/I address space.

Tell IMS to page fix all these areas, except the database subpools, by naming the resource in control statements that are included in the DFSFIXxx member of IMS.PROCLIB. The corresponding names used in these control statements are shown in parentheses in the above list.

To page fix the database subpools, you use control statements in the DFSVSMxx member of IMS.PROCLIB. The IOBF statement controls OSAM subpools. VSAM pool page fixing is specified with the VSAMFIX keyword of the OPTIONS control statement.

Usually, I/O for a pool impacts performance less than page faulting through the pool. If storage is a constraint, using a smaller fixed pool is preferable to using a larger pool that is subject to paging. However, page fixing is unlikely to improve performance, and might lead to higher overall paging rates, if your system is dedicated to IMS. This is because page fixing leaves a smaller available page pool for the rest of the system to use. Unless some other action is taken to reduce the storage requirements, higher rates inevitably result.

Related reference

DFSFIXnn member of the IMS PROCLIB data set (System Definition)

DFSVSMxx member of the IMS PROCLIB data set (System Definition)

IMS log buffers

When selecting the IMS online log data set (OLDS) block size and number of buffers in order to minimize system log I/O, similar concepts apply from general buffer pool considerations.

The log buffers only need to be plentiful enough to prevent IMS from waiting for a log write in order to get buffer space. The write-ahead data set (WADS) ensures that only full log buffers are written to the OLDS. The log write-ahead function does not truncate log buffers. When defining OLDS block sizes keep in mind that using a multiple of 4K will result in the log buffers being placed above the 2 gigabyte bar in real storage, but are below the 2 gigabyte bar in virtual storage.

Some buffer space is used for dynamic backout. When the backout records exist in the output buffer, they are used directly from the buffers. Otherwise, a buffer is removed from the set of output buffers and used to read the required log block from the OLDS. When backout is complete, the buffer is returned for use as an output buffer. Log buffers are long-term page fixed.

Related reading: See "General buffer pool considerations" on page 406 for more information on general buffer pool considerations.

Using Library Lookaside under z/OS

As an alternative to IMS program loading with z/OS Program Fetch, you can specify the use of the z/OS service Library Lookaside (LLA). LLA is the recommended method for managing dynamically loaded program libraries.

For IMS application programs, it saves load modules in a data space, so the modules can be retrieved more efficiently than through DASD. LLA monitors the frequency with which modules are accessed and readjusts its data space population to optimize the probability of having a module in the data space when it is requested.

Guidelines for application program preload in DB/DC and DCCTL environments

The preload option is commonly used for high-activity programs that do not use a large amount of virtual storage. You specify the names of the modules to be preloaded in member DFSMPLxx in the IMS.PROCLIB data set.

When the message processing region is initiated, you specify the suffix xx as a parameter in the EXEC statement of the DFSMPR procedure. Preloaded application programs are then branched to directly rather than through a FETCH program. Preloaded programs should have a schedule-to-first DL/I call elapsed time less than those that use the FETCH program, but page fault serialization can cause the application program elapsed time to increase.

Restriction: Do not preload application programs for batch regions executed using either the DLIBATCH or DBBBATCH procedures. There is no advantage to preloading application programs in these environments.

Guidelines for the most effective implementation of program preload are:

- All commonly used PL/I, VS Pascal, or COBOL subroutines and application program subroutines should be preloaded into each dependent region. If these subroutines are reentrant, they should be put into the pageable link pack area (PLPA) so that only one copy resides in real storage. Match subroutine usage with the region preload lists to ensure that the appropriate modules are preloaded.
- Application programs are candidates for preload when they account for a high percentage of a region transaction volume. Preload is most effective when the transaction arrival rate for the preloaded program is adequate to keep the working set of the program in real storage. If a system is constrained by real storage contention, preload only subroutines and very high-volume application programs because preloading can increase the paging rate.

  In addition to deciding to preload them, consider class scheduling of high-volume transactions. Depending on the transaction arrival rate, it can be advantageous to preload in only one or two regions and class schedule the transactions accordingly.

  The use of the preload option for system performance improvements is highly dependent on the availability of real storage and the arrival-rate of the candidate transactions.
- Application program overlay is sometimes a viable alternative to preload. If an all-purpose application program is coded to interrogate the input transaction data and execute only a portion of the application program code for the transaction subcode, program overlay I/O might be more efficient than loading or paging through the entire program. Preload is a better alternative if the transaction arrival rate is sufficient to maintain a working set in main storage.

Pseudo-wait-for-input (PWFI) and wait-for-input (WFI) programs that are specified in the DFSMPLxx PROCLIB member cannot be refreshed dynamically by use of the **`UPDATE PGM START(REFRESH)`** command.

Minimizing path length

The total number of machine instructions that are executed to process a transaction, including system services, IMS services, and the application program itself, has a direct bearing on throughput. The accumulation of executed instructions is termed the *path length*. Options chosen for performance affect path length.

The actions suggested in "Choosing IMS options for performance" on page 399 all contribute to the minimization of path length. Avoid the regular use of traces such as the DL/I Call Image Capture and other traces invoked by the **/TRACE** command. These are specified as parameters on the OPTIONS statement in the DFSVSMxx member of IMS.PROCLIB.

Do not run the IMS Monitor (DFSMNTR0), except for during 10- to 20-minute preplanned intervals.

In a real storage-constrained system, the most effective way to reduce path length is to minimize paging. Minimal pools help to minimize paging by eliminating costly scanning of directories or buffers that might need to be paged in before they can be read. If virtual storage requirements are reduced:

- A minimal PSB pool minimizes buffer searching.
- A tuned database pool minimizes buffer searching; a larger database pool costs more in path length and might not reduce I/O.
- A tuned message queue pool minimizes buffer searching; a larger pool reduces IMS message queue I/O but does so at the expense of higher processor cycles per queue pool operation.
- The same applies to the message format buffer pool as to the message queue pool.

Considerations for the communications network in DB/DC and DCCTL environments

Performance factors that affect response time can be related to NCP and network considerations or host transaction processing considerations.

- NCP and network considerations

  Review the points in the network where delays might occur and try to avoid bottlenecks. Delays can occur:

  - Within a cluster and queue for a line
  - Waiting for transmission and waiting to be polled
  - Through line errors
  - Within the NCP
  - Within VTAM
  - Waiting for IMS to receive any macro
  - Within IMS
  - Waiting for transmission of output

  Another possible problem is that VTAM cannot easily process larger output transmissions. You should check the appropriateness of the value for the OUTBUF keyword on the TERMINAL macro statements and ETO logon descriptor.

  Input can also be delayed because IMS has no more input threads left.

- Host transaction processing considerations

  Be aware of the trade-off made between the size of VTAM I/O buffers to service the communications network and the use of auxiliary storage. IMS communication traffic might share these resources with TSO and other subsystems.

  Use the VTAM authorized path option to reduce processing of SEND and RECEIVE commands by bypassing validity checks of request parameter lists (RPLs) and I/O areas specified in the RPL. You do this by specifying VAUT=1 as a parameter on the EXEC statement for the control region.

  IMS transactions should also be specified as response mode where possible. The major benefits are that the number of transmissions and line turnarounds is decreased and that line utilization is reduced.

Guidelines for IMS system data set placement

In addition to tailoring the IMS system data sets to the workload, you should avoid undue contention for their use. Correct placement of high-usage data sets helps you avoid bottlenecks.

Here are some guidelines you should consider when placing your IMS online data sets:

- Separate among lightly used DASD volumes: heavily used libraries and data sets such as IMS.PGMLIB, the online log data sets, the active IMS.FORMATA/B and IMS.ACBLIBA/B libraries, and message queues.
- If high-activity data sets are on the same device, place them close to each other. The space allocation should stipulate contiguous storage and allocation by cylinder to avoid fragmentation of stored data.
- Have the volumes available to IMS mounted as private so that operating system temporary data sets are not allocated to these volumes.
- Do not place online IMS system data sets on shared DASD volumes. Avoid contention for both the device and the control unit. Position primary and secondary OLDSs on separate control units and channels when possible.
- To optimize program fetch I/O, order PGMLIB in descending frequency of use. Use full-track blocking. This minimizes Start I/O (SIO) operations and seek times.
- If using fixed-head devices, place write-ahead data sets (WADS) and database index data sets on the fixed-head portion.
- If program library contention is a problem, create multiple copies, each for a subset of the message regions. Do not mix production and test program libraries.
- For IMS shared data sets subject to updates, place the data sets on DASD devices capable of allowing parallel I/O from the same or multiple systems.

I/O subsystem configuration

Use the RMF Channel Activity and Direct Access Device Activity reports to observe queuing delays in the I/O subsystem. Configure so that you minimize those delays by balancing the load across physical and logical channels.

Application optimization in DB/DC and DCCTL environments

To reduce the amount of read I/O to the ACBLIB data set, consider loading the ACB members into 64-bit storage by specifying ACBIN64=ggg in the DATABASE section of the DFSDFxxx PROCLIB member. When the application is scheduled, the ACB members are retrieved from 64-bit storage instead of from the ACBLIB data set.

If possible, avoid the use of overlays in message processing programs. An exception might be if the application program uses large blocks of code exclusively for part of the transaction processing.

Do not have application programs that perform non-DL/I functions (such as STIMER and TTIMER, and contents supervision). Unless carefully controlled, such application programs usually perform poorly.

If lengthy application programs must be initiated online, consider splitting the application into two parts: an interactive part, which responds to the user and sends a program-to-program message to a batch part, which can be run in a low-priority region.

Consider redesigning applications whose performance is critical. Otherwise, plan to tune the system to process the transaction as it is before embarking on a total rewrite. Consider translating applications whose performance is critical from high-level language to assembler in order to improve program-load and execution time.

Always use the rollback (ROLB) call rather than cause the program to abnormally terminate. This keeps the message region available without operator intervention and with a smaller delay in message region availability to other transactions. (Abnormally terminating an application program keeps the region unavailable to other transactions waiting to be scheduled.)

DL/I considerations

Consider the following things when designing DL/I databases.

- Use DL/I path calls wherever possible to reduce the DL/I path length and program execution time.
- Use qualified DL/I calls wherever possible to minimize the number of calls required to obtain the desired segment.
- Do not issue a GN call to the I/O PCB if the application program always retrieves single-segment input messages.
- Ensure that the application program returns to issue another GU to the I/O PCB on completion. This allows processing of another transaction of the same type if one exists on the input queue. This avoids the overhead of termination and rescheduling processing.
- When output is sent to several TP-PCBs, define multiple alternate PCBs rather than using the CHNG call.
- Select HDAM (or PHDAM, if using HALDBs) as the organization for a database wherever possible, because, when well tuned, HDAM (or PHDAM) generally results in shorter path lengths and fewer I/Os per DL/I call.

In assessing the characteristics of transaction processing and the interaction of application programs and databases, be aware of the performance design considerations presented in *IMS Version 15.2 Database Administration* and *IMS Version 15.2 Application Programming*.

Planning for performance in a shared-queues environment

In a shared-queues environment, individual transaction performance might be poorer than in a non-shared-queues environment. However, because of increased parallelism and workload balancing, overall system performance and throughput should increase. And because input transactions are processed by the originating local IMS system (if a message region is available), you can influence IMS performance by controlling the number of dependent regions for each IMS system.

Each of the following can have an effect on system performance:

- IMS execution parameters: QBUF=, QBUFMAX=, LGMSGSZ=, and SHMSGSZ=
- CQS Local Structure Definition parameter, SYSCHKPT=
- CQS Global Structure Definition parameters: OBJAVGSZ=, OVFLWMAX=, and STRMIN=
- Size of the shared queues structures on the coupling facility
- Size of the z/OS system log structure on the coupling facility
- Number of z/OS system log data sets to back up the z/OS system log structure
- Frequency of structure checkpoints

The size of all log records has increased by eight bytes, and the size of Queue Manager and EMH log records has increased by an additional 32 bytes.

The SPA pool is no longer used, decreasing the amount of data logged for each checkpoint, and increasing the amount of available storage. The SPA is now maintained with its input and output messages.

For Fast Path systems, using the Fast Path Input Edit Router (DBFHAGU0) should be considered carefully because messages defined as "Local Only" will have priority over all other messages ("Local First" and "Global Only").

Identifying and correcting performance problems

Your general strategy is to decide the origin of the problem being experienced by performing a set of tasks.

Complete the following tasks to decide on the origin of the problem:

1. Isolate the problem into one of the following categories:
   - High paging rate seems to be slowing some or all of the workload.

- High utilization of the processor by some subset of the total workload is causing dispatching problems for another (lower-priority) subset.
- I/O contention in the DASD subsystem is slowing some subset of the workload.
- Utilization of, or contention for, resources associated with the communications subsystem is slowing the transmission of input or output messages.
- Within IMS, utilization of physical resources (processor, I/O, storage) or contention for logical resources (pools, regions, control blocks, latches) has a negative impact on the performance of some or all of the transactions in one of the following areas:
  - Input or output message processing, including input queuing and Message Format Services (MFSs)
  - Program scheduling and termination
  - Program load and initialization
  - Program execution

2. Investigate further, if necessary, after you isolate the problem area, to determine:
   - The precise nature of the problem
   - The principal offenders by transaction or other category
   - The tuning action most likely to alleviate the problem
3. Take the appropriate tuning action to prevent the problem from recurring.

Examining paging rates

The operating system allocates real storage for the address spaces occupied by the control region or dependent regions. IMS might not obtain adequate machine cycles because of the excessive demand for real storage. Observe the paging rates and the frequency with which processing is delayed by a page fault. You can use the output from RMF II to examine the paging rates at peak IMS loads.

Page faulting in an IMS online system can directly increase the transaction time-in-system by the duration of the paging activity. If page faulting must occur, it is best to confine it to application programs in the dependent regions. A page fault experienced by the control region can hold up any dependent region waiting for a control region service. IMS provides some page-fixing options; these are covered in "Page fixing the IMS buffer pools" on page 406. However, if real storage is constrained, any page fixing generally serves to increase paging elsewhere and is not recommended. The primary tuning technique must be to reduce the virtual and the real storage requirement. This is particularly true in a dedicated IMS environment, where page fixing some portion of IMS only makes paging worse in the unfixed portion, possibly making overall paging rates worse.

You can monitor page rates dynamically, daily, or in detail. For more information about monitoring page rates, see "Deciding on monitoring activities and techniques" on page 365.

After the reason for the increased storage over commitment is identified, one or all of the following actions must be taken:

- Trade off IMS paging I/O against IMS controlled I/O.
- Reduce IMS paging I/O by reducing over commitment (reduce concurrent users) or by increasing IMS priority relative to other users.
- Speed up paging I/O by reducing I/O contention, splitting page data sets, or by placing page data sets on faster devices.
- In a non-dedicated IMS environment, try to reduce IMS paging, at the expense of other competing work, by fixing some portions of IMS.

Detecting processor resource problems

In an environment with many jobs or subsystems operating concurrently, IMS competes for machine cycles. If those jobs or subsystems, such as an IMS batch execution or TSO, have a higher dispatching priority than the online IMS system, their workload has a marked effect on IMS performance.

Note: Monitor your resources dynamically, daily or in detail. For more information about how to perform this monitoring, see "Deciding on monitoring activities and techniques" on page 365.

Tuning to remove I/O resource contention

A performance problem can occasionally be traced to I/O contention. Tuning activities for I/O devices, DASD storage access, channel usage, or control unit contention should not be confined to system-related devices. For IMS, an important factor is the volume of I/O activity, because reduced I/O activity reduces contention.

See "I/O subsystem configuration" on page 410 and the considerations for database I/O analysis included in "Program execution times" on page 418.

Dynamic monitoring of the I/O subsystem

Typically, the I/O subsystem is not monitored dynamically, because corrective action normally requires reconfiguration of devices or data sets. This takes some time to plan and implement. However, if RMF is being run, the RMF Channel Activity and Direct Access Activity can be used to check against your objectives for I/O times, channel utilization, and device utilization.

Immediate action might not be possible. However, it might be possible to move data sets or reconfigure devices to correct the problem before the next IMS restart.

Daily monitoring

Examination of the RMF Direct Access Device Activity and Channel Activity reports, together with knowledge of the I/O configuration and data set placement, might suggest a possible tuning action along the lines of those described in "Guidelines for IMS system data set placement" on page 410 and "Dynamic monitoring of the I/O subsystem" on page 413.

Detailed monitoring

If additional data is required, plan to run GTF and obtain Volume Map, Seek Histogram, and System/Job Summary (EXCP-SIO-IO data) reports. These reports can be used to analyze I/O activity in great detail.

Communication subsystem contention in DB/DC and DCCTL environments

Because of the great variety of transmission devices and types of lines, each suspected problem in the communication subsystem area must be separately examined.

As described in "Considerations for the communications network in DB/DC and DCCTL environments" on page 409, many possible causes of delay exist within the network hardware and software. A variety of different tools might be required to investigate any suspected problems in this area.

Dynamic monitoring

It is not possible to dynamically monitor the performance of the telecommunication subsystem.

Daily monitoring

If the response time breakdown data indicates large and variable IMS output queue times, a bottleneck might exist in the IMS output message handling or on a line. Because it is not normally possible to reconfigure the TP network without considerable preparation time, plan to use the IMS Monitor and IMS line traces to investigate those IMS lines.

Detailed monitoring

VTAMPARS can be used to analyze the GTF SMS records and to tune the VTAM buffer sizes. Sorting of DFSILTA0 and IMSPA Transit Log data by line or PTERM can isolate line-related problems for output queues to a single line or group of lines.

The IMS Monitor indicates line-related IWAITs, which require tuning of the IMS High Input/Output Pool (HIOP). Although HIOP is dynamically allocated, you might need to adjust either the buffer size definitions or the upper expansion limit. IMS Monitor output that is of interest for contention in communications are:

- The Communication Summary report gives the mean elapsed time for transactions using a VTAM node. This time can be affected by activity in the message format buffer pool.

  The report also gives mean NON-IWAIT times. These can reflect device bottlenecks or problems in obtaining an input thread as described in Save area sets in DB/DC and DCCTL environments. The count of times that threads were not available is given in a separate report under the heading "REPORTS". The line ‘Total Times ECBS waited for SAPS = nnn' gives the total. This figure is high if the monitor interval includes the time during which network initialization took place.
- The Line Functions report shows, for each active node, the data transmission activity, as described in Chapter 24, “Collecting and interpreting IMS monitoring data,” on page 357. The TRANSMISSION BLKSIZE values show the number of characters passed by IMS to VTAM in OUTBUF buffers, a possible indicator of overloading of VTAM buffers. The TURNAROUND intervals can be used to identify inactive lines that might be eligible to distribute transaction entry.
- The Communication IWAIT report gives details of IWAITs for each node.

To trace VTAM activity, you can use GTF (including the Buffer Trace) to examine the content of VTAM buffers as data enters and leaves VTAM buffers. You can also use the Storage Management trace to reflect the overall use VTAM buffers.

Ultimately, tuning of network problems probably requires one or more of the following actions:

- Reduction of volumes of data transmitted. (See “Choosing IMS options for performance” on page 399.)
- Adjustment of VTAM or NCP options and parameters.
- Reconfiguration of the network to eliminate bottlenecks on multi-dropped lines.
- Isolation and correction of hardware delays caused by device errors.

IMS message processing in DB/DC and DCCTL environments

This topic identifies possible performance problems associated with message processing and the message queues on an IMS system.

Dynamic monitoring

MFS and Queue Pool statistics can indicate problems in this area. Pool sizes cannot be dynamically increased. They must be increased at the next IMS start. Normally, once this aspect of performance is tuned, it requires only periodic monitoring.

Daily monitoring

If IMS output message processing is a problem area, IMS input message processing is probably affected by the same delays. Investigate any large or variable output queue time of IMS message queue and MFS (message format buffer pool) statistics using the IMS PA Resource Usage reports and DC Queue Manager Trace reports.

If output queue times are high in a DB/DC environment, check the IMS PA DC Queue Manager Trace report for output message lengths and run the IMS Monitor to study the Communications IWAIT report.

Detailed monitoring

For detailed monitoring, use the following:

Message Queue Processing

IMS PA can be used to investigate the IMS Queue Manager in detail, using the DC Queue Manager Trace report. DFSUTR20 and IMSASAP give statistics on frequency and duration of communications-related IWAITs.

GTFPARS Job Summary and Detail Trace reports for the IMS control region can be used to examine the performance of MFS and Message Queue I/O.

Message Queue Data Set Tuning

The three data sets, IMS.QBLKS, IMS.SHMSG, and IMS.LGMSG, contain the directory and the short and long message segments for the message queues. In a busy online system with some constraints on the terminal I/O and conventional SPA pools, these data sets are the most active and critical. Place these data sets on different devices to minimize seek time. Check to see if the I/O activity is balanced between short messages and long messages. You might have to allow for a possible difference in device access times.

You can do this by examining the IMS Monitor Communication IWAIT report. The entries are organized by line number and reveal high frequency or high I/O elapsed times. Each line in the report identifies the DD name causing the IWAIT, so that you can pick out those that pertain to IMS.QBLKS, IMS.LGMSG, and IMS.SHMSG. The Region IWAIT report similarly identifies message queue I/O and totals IWAIT times for the region.

You can define up to 10 long and 10 short message queues.

Related reading: For more information on this option, see *IMS Version 15.2 System Definition*.

MFS Usage

The GTFPARS Job Summary report for the IMS control region provides data on EXCP-SIO-IO timings to FORMATA/B. The IMS Monitor indicates in the Communication IWAIT report any IWAITs for Format Pool Space or for Directory or Block loading.

Message format library optimization

The active IMS.FORMATA/B data set is the source of all MFS format blocks used during online execution and is usually a busy data set.

You can check the frequency or high I/O elapsed times against this data set using the IMS Monitor Communication IWAIT report. Report lines identified by I/O=DIR=fn or I/O=BLK=fn, where fn is the format block name, show I/O IWAIT time for directory lookup and block fetch by increasing the size of the format buffers. This pool size is specified with the FORMAT keyword on the BUFPOOLS macro for system definition. You can override the value for the online execution using the FBP parameter in the EXEC statement.

To reduce I/O to the directory, you can also create an index of track addresses of the blocks (or only those frequently referenced) and keep this index present in the MFS pool. The index is built when you execute the MFSRVC procedure, specifying INDEX on the utility control statement. You also include the names of the format or message blocks to be in the index.

Multiple reads of the active library data set occur when the size of the MFS control blocks stored on DASD is greater than either the track size or the z/OS block size.

Input queuing and scheduling/termination in DB/DC and DCCTL environments

Another set of performance problems is brought about by failure to achieve the performance objectives. The problem might be observed by input queue buildup caused by either increased workload or application program scheduling delays.

Dynamic monitoring

This topic explains how to obtain dynamic monitoring information for IMS input queuing, IMS region occupancy, IMS scheduling, and sync point processing.

IMS input queuing
Input queue time is not available. However, using the IMS **/DISPLAY** command with the QUEUE parameter, you can monitor the sizes of the input queues. You can also use the ACTIVE parameter to track performance and detect bottlenecks in scheduling and in the region occupancy area.

IMS region occupancy
Not available in any dynamic report. The IMS command **/DISPLAY ACTIVE** is used to monitor region usage.

IMS scheduling
The PSB and DMB Pools should already be tuned, as recommended in "Initializing z/OS and IMS parameters for tuning" on page 398. If all regions are busy, start any necessary additional regions. If some regions are idle, modify classes or PARLIM if one transaction is overloading a region.

Related reading: For more information, see "Avoiding contention for IMS resources (excluding buffer pools)" on page 401.

Sync point processing
The occurrences of problems caused by waiting for synchronization points cannot be isolated dynamically.

Daily monitoring

This topic explains how to obtain daily monitoring information for transaction response times, response time breakdown, region occupancy, program scheduling, and sync point processing at program termination.

Transaction response times
Extract total (internal) response times for selected transactions that can be used as indicators of system performance.

Related reading: For more information on sources of response time data, see "Establishing performance objectives" on page 358.

Response time breakdown
For similar transactions, extract response time breakdown (input queue, switch queue, processing, output queue) from IMS PA or DFSILTA0.

Region occupancy
You can obtain indications of the relative use of the dependent regions in a DB/DC environment by using the IMS PA Resource Availability Reports. Also, you can directly use the summary of region occupancy percentages given in the IMS Monitor Region Summary report.

Program scheduling
If the response-time breakdown data indicates large and variable input queue times, check the Region Occupancy figures. If all message regions have high occupancy, then another message region might be required. Alternatively, it might be possible to reduce occupancy by reducing program load or program execution times. If some or all of the IMS message regions are not busy, analysis of IMS PA Transaction Transit reports by transaction and class probably shows that one transaction or class is more critically hit than others. In this case, you should review the designation of classes and the allocation of classes to regions. PROCLIM and PARLIM should be reviewed also.

Related reading: For information on scheduling options, see "Choosing IMS options for performance" on page 399.

Programs executing as wait-for-input never show 100% occupancy even when they are in the region 100% of the time. Zero occupancy might be cause to review operator procedures, with instructions to manage the number of message regions based upon display of the queues.

The IMS PA Transaction Transit reports Graphic Summary is useful to analyze input queuing time by time of input across the entire measurement period. This can be used to discover if high input queue times result from a transient peak in transaction volumes or from a more sustained phenomenon. DFSILTA0 can be used for the same purpose, although its output is numeric rather than graphic.

Sync point processing at program termination

If you are using IMS PA to provide response-time breakdown data, a high program switch queue time can indicate that delays are occurring because of a wait-for-synchronization point completion. This occurs if MODE=MULT is used. Check the IMS PA CPU Usage report for multiple transactions of that type being dequeued per program schedule. As a general rule, use MODE=SNGL to avoid any delays of this type.

Detailed monitoring

You should examine detailed monitoring information for scheduling and sync point processing.

Scheduling

The IMS Monitor generates reports on PSB and DMB I/O counts and elapsed times. GTFPARS Job Summary and Detail Trace reports for the IMS control region can also be used for detailed investigation of these I/O events. However, scheduling problems are more often related to region availability and message class, PROCLIM, and region class assignments. These are described in "Avoiding contention for IMS resources (excluding buffer pools)" on page 401.

Sync point processing

Delays caused by waiting for a synchronization point do not normally require detailed investigation.

Program load and initialization

This topic describes dynamic and daily monitoring and detailed monitoring for program load and initialization.

Dynamic and daily monitoring

The resource used to load or initialize a program cannot be isolated or corrected dynamically from normal daily monitoring data. Usually, this aspect of performance, once it is tuned, does not require more than periodic monitoring.

The IMS.PGMLIB data set is specified in the dependent region JCL and contains the application programs. The schedule-to-first DL/I call elapsed time shown on the IMS Monitor Region Summary and Program Summary reports includes the I/O time to bring a program into the region. Included in the I/O time is the system search through JOBLIB (searched first) or STEPLIB. If JOBLIB and STEPLIB are not used for the program library, IMS.PGMLIB should be first in the LNKLSTnn member of SYS1.PARMLIB. Program libraries should always be blocked to full track size. Placement in the dependent region STEPLIB is recommended because a library can be built containing only those members required by the region.

In environments that are not constrained by real storage availability, consider making a trade-off between program loading and preload. You can use the schedule-to-first DL/I call elapsed times from the IMS Monitor Region Summary report to assess the potential savings of preload.

z/OS Library Lookaside (LLA) is the strategic method for managing dynamically loaded program libraries.

Related reading: For more information on LLA, see "Using Library Lookaside under z/OS" on page 408.

Detailed monitoring

The GTFPARS Job Summary and Detail Trace reports for the IMS Message Regions can provide comprehensive data on the counts and sequences of SVC and I/O operations issued to locate and load the user's application program, and to execute the high-level language initialization modules.

For information about program library options and initialization considerations, see "Choosing IMS options for performance" on page 399.

Program execution times

You can obtain program execution time information with dynamic monitoring of IMS internal response times, daily monitoring of program execution, and detailed monitoring.

Dynamic monitoring of IMS internal response times

IMS internal response times are not available in any dynamic report. However, many installations have written special transactions that always perform a fixed (usually small) number of DL/I calls, such as a GHU, ISRT, REPL, DLET sequence. By use of this transaction, an IMS MTO can dynamically monitor the responsiveness of the IMS system. This is not a precise measure, but an indication of what a user is experiencing at that time.

Daily monitoring of program execution

If the response-time breakdown data available from IMS PA or DFSILTA0 indicates large execution times, the cause might be related to one of the following:

- Program Load Time

  When daily monitoring indicates increases in execution times, the problem might be in the I/O subsystem.
- Program Initialization and Application Code

  After tuning, execution times are not likely to change significantly.
- DL/I Calls and I/O

  The IMS PA Internal Resource Usage report gives database I/O statistics. If these are high, the GTFPARS Data Set Overview report can be reviewed, together with the sizes and number of database buffers. Reorganize volatile databases regularly to reduce OSAM or VSAM overflow I/O. For further investigation of this area, the IMS Monitor should be run to obtain Call Summary and Region IWAIT reports.
- Other IMS Resource Delays

  The IMS Internal Resource Usage reports in IMS PA indicate if contention within program isolation activity is high, in which case, plans should be made to run and analyze the IMS PI trace.
- Dependent Region Modifications

  Examine the reasons for high elapsed time for dependent regions or for those application programs that exhibit excessive elapsed time for each transaction. You can use the IMS Monitor Program Summary report. Assuming that the dispatching priority is adequate, you can look for high schedule-to-first DL/I call elapsed times. Setting aside factors related to program load and paging, you need to find out if any modifications have been made in the dependent region code by your installation.

  If changes have been made to support accessing z/OS data sets or the use of other z/OS services, you must evaluate the appropriateness and efficiency of these changes. One reason for these changes might be excessive overlay processing or initialization.

Detailed monitoring

GTFPARS Job Summary and Detail Trace reports for the IMS Message Regions are the best method of examining the system-related activities of the application program in detail. IMSASAP Program Trace reports provide detailed analysis of application activity related to database access in a DB/DC environment. IMS PA Database Trace reports show breakdowns of database update activity in a DB/DC environment. DFSPIRP0 should be used to investigate any long delays if PI lockout is suspected as a possible cause.

The time spent in database retrieval that incurs I/O can be a large contributor to program execution time. The frequencies of each type of DL/I call to each database segment type and the frequencies of IMS IWAITS are given in the Call Summary report. If database-related delays are suspected as the cause of high program-execution times, that report helps identify the particular call and database combination that is responsible. The I/O wait times are shown in the IMS Monitor Region IWAIT report.

IWAIT times greater than 100 milliseconds are usually caused by interference from shared DASD or over commitment of resources. Databases using VSAM often experience shorter IWAITs, because the I/O buffer control managed by VSAM is able to meet the request without physical I/O. Excessive chaining of records to overflow tracks can be a contributing factor. Sometimes the data transfer rate of the storage device itself has a limiting effect.

In addition to database design considerations, the two principal strategies for decreasing I/O IWAIT time are:

- Reduce the frequency of I/Os by adding more buffers. This entails identifying inadequate IMS buffer allocation.
- Examine the I/O resource usage to discover if the distribution of data sets is a problem. This is a more protracted analysis that uses detailed output obtained from GTF and RMF traces.

A more extensive action is to change the DL/I structure or database organization.

Related reading: For more information on changing the DL/I structure or changing database organization, see *IMS Version 15.2 Database Administration*.

Chapter 27. Enabling and disabling IMS functions

If an IMS function is not enabled by default, you can enable the function dynamically while IMS is running by issuing the **UPDATE IMSFUNC** command. Or, you can enable the function statically by defining the function as enabled in the DFSDF*xxx* member of the IMS PROCLIB data set, and then cold starting IMS.

To enable IMS functions dynamically, ensure that OM and SCI are enabled.

Changes that you make by using the **UPDATE IMSFUNC** command are logged in the x'22' map byte x'31' log record and are recoverable across an IMS restart. For example, if a function is enabled in the DFSDF*xxx* member and later disabled by using the **UPDATE IMSFUNC**, the disabled value is recovered if IMS restarts.

If you change a function enablement value by using the **UPDATE IMSFUNC** command and IMS is cold started, one of the following situations occurs:

- For local functions, the enablement value is retrieved from the DFSDF*xxx* PROCLIB member during cold start. In this case, the **UPDATE IMSFUNC** command might need to be issued following the cold start to return the enablement value to a previous state.
- For global functions, if you use RM, CQS, and a resource structure, the enablement value is retrieved from the resource structure during cold start.
- For catalog functions, the enablement value is retrieved from the catalog during cold start.

After installing a PTF that includes an IMS function that is not enabled by default, use one or more of the following steps to enable or disable the function:

- To enable IMS functions statically, define the enabling parameter for the function in the DFSDF*xxx* member, and then cold start IMS.
- To enable IMS functions dynamically, perform the following steps.

  a) Restart IMS.

     When IMS restarts, message DFS4878I is issued to display the current IMS function level.

  b) Issue the **UPDATE IMSFUNC** command with the **SET(ENABLED(Y))** option specified.

     For IMS catalog functions, the enablement value is stored in the IMS catalog after you issue the **UPDATE IMSFUNC** command.

     For all other global IMS functions, if you use RM, CQS, and a resource structure, the enablement value is stored in the resource structure.

  c) Optional: For local functions, define the function as enabled in the DFSDF*xxx* member so that the function remains enabled at the next IMS cold start. For global functions, if you do not use RM, CQS, and a resource structure, define the function as enabled in the DFSDF*xxx* member so that the function remains enabled at the next IMS cold start.

- To disable the function statically, define the function as disabled in the DFSDF*xxx* member so that the function remains disabled at the next IMS cold start.
- To disable the function dynamically, perform the following steps:

  a) Issue the **UPDATE IMSFUNC** command with the **SET(ENABLED(N))** option specified.

     For IMS catalog functions, the enablement value is stored in the IMS catalog after you issue the **UPDATE IMSFUNC** command.

     For all other global IMS functions, if you use RM, CQS, and a resource structure, the enablement value is stored in the resource structure.

  b) Optional: For local functions, define the function as disabled in the DFSDF*xxx* member so that the function remains disabled at the next IMS cold start. For global functions, if you do not use RM, CQS, and a resource structure, define the function as disabled in the DFSDF*xxx* member so that the function remains disabled at the next IMS cold start.

Related concepts

"IMS function levels overview" on page 66
An IMS function level identifies a PTF that contains one or more IMS functions. The function level is a full word hexadecimal number, which is represented as decimal in output for the **`QUERY IMSFUNC`** command and in messages.

Related reference

QUERY IMSFUNC command (Commands)

UPDATE IMSFUNC command (Commands)

Chapter 28. Making online changes

IMS provides several different options for making changes to resources in an online IMS system.

The options for making changes to resources in an online IMS system include:

- When Dynamic resource definition is enabled, the IMS type-2 commands CREATE, UPDATE, and DELETE. The IMPORT command can also be used to activate resources from definitions that are stored in either the IMSRSC repository or a resource definition data set (RDDS).
- When the IMS management of ACBs is enabled, the IMPORT command to activate DBD and PSB resources from new or modified definitions that were previously stored in the IMS catalog by using DDL or one of the IMS catalog population utilities.
- The database alter functions
- The online change function

IMPORT command for DBD and PSB changes

When the IMS management of ACBs is enabled, database definitions (DBDs) and program view definitions (PSBs) in the IMS catalog can be activated in an online IMS systems by issuing the IMPORT DEFN SOURCE(CATALOG) command. The DBDs and PSBs must be submitted to the IMS catalog by using either DDL or one of the population utilities with the MANAGEDACBS control statement specified.

Dynamic resource definition (DRD)

DRD enables you to make online changes to certain runtime resource definitions without using the online change process. To change other resources online (for example, resources in IMS.ACBLIB), you still need to use the online change process. With DRD, you can issue type-2 commands to create, update, and delete the runtime resource definitions dynamically, or use the enhanced Destination Creation exit routine (DFSINSX0), formerly called the Output Creation exit routine, to create transactions (and, if necessary, the programs that are associated with the transactions) instead of using the online change process.

You must ensure that:

- Commands are routed to and executed on the systems where you want the changes to be applied
- Any changes made dynamically with the DRD commands are recovered across a cold start
- The resource and descriptor definitions are exported to an RDDS

DRD enables you to create, update, query, and delete the following IMS resource definitions and their descriptor definitions dynamically, without using the system definition or online change processes:

- Application programs
- Databases
- Fast Path routing codes
- Transactions

Because DRD uses type-2 commands, you can only use DRD where your IMS is configured as follows:

- A single-IMS IMSplex that is configured with a minimal Common Service Layer that include an Operations Manager (OM) and a Structured Call Interface (SCI), but no Resource Manager (RM). This configuration is also known as an enhanced command environment.
- A multiple-IMS IMSplex that is configured with a full CSL (OM, SCI, and RM).

Database alter functions

The online alter function for DEDB databases is different than the online alter function for HALDB databases.

You modify online DEDB databases and areas by using the DEDB Alter utility.

You modify online HALDB databases and partitions by using the ALTER option of the INITIATE OLREORG command.

Only certain types of changes can be made to a database by using the alter function. The types of changes you can make differ depending on whether you are modifying a DEDB or HALDB database.

Online change function

There are two variations of the online change function:

Local online change
Allows you to make online changes to IMS resources for one IMS.

Global online change
Allows you to coordinate online changes to IMS resources across all IMS systems in an IMSplex.

The subtopics in this section describe the IMS online change function with DRD disabled.

Related concepts
IMS management of ACBs (System Definition)
Altering the definition of an online DEDB database with the DEDB Alter utility (Database Administration)
Overview of dynamic resource definition (System Definition)
Related tasks
Modifying online databases (Database Administration)
Altering the definition of an online HALDB database (Database Administration)
Related reference
INITIATE OLREORG command (Commands)
DEDB Alter utility (DBFUDA00) (Database Utilities)

Online changes in managed-ACB environments

When the IMS management of ACBs is enabled, you use the **`IMPORT DEFN SOURCE(CATALOG)`** command to change the active ACBs in online IMS systems. You cannot use the online change function to activate ACBs when they are managed by IMS.

The ACBs that contain the pending changes must be in the staging data set of the IMS catalog before you issue the **`IMPORT DEFN SOURCE(CATALOG)`** command.

When the **`IMPORT DEFN SOURCE(CATALOG)`** command is issued, IMS takes the following actions:

1. Adds or replaces the ACBs in the directory data set of the IMS catalog.
2. Removes the ACBs from the staging data set of the IMS catalog.
3. Removes all old copies of the resources from memory and casts out DMBs and PSBs from their respective pools.
4. If the IMS catalog is shared by multiple IMS systems, coordinates the change across the IMSplex.
5. Updates the IMS catalog HEADER segment for the imported objects.

When you issue the **`IMPORT DEFN SOURCE(CATALOG)`** command, IMS uses an internal process that is similar to the online change function to activate the resources in online IMS systems. Consequently, if you issue the **`QUERY MEMBER`** command during import processing, the status reflects the same prepare and commit phases that online change uses. If errors occur, the messages might also reflect these phases.

Depending on the type change you are making and whether you are updating PSBs, DBDs, or both, you might need to take more steps before the resources can be used. For example, you might need to reorganize the database for certain types of database changes, or you might need to allocate data sets, register the database with DBRC, or issue the IMS type-2 **`CREATE DB`** command if you are creating a new database.

Related concepts

IMS management of ACBs (System Definition)

Related reference

IMPORT DEFN SOURCE(CATALOG) command (Commands)

IMS catalog utilities (System Utilities)

Overview of resource activation when IMS manages ACBs

When IMS manages application control blocks (ACBs), you can activate new databases (DBDs) or program views (PSBs) in online IMS systems either automatically when the changes are submitted to IMS or manually after the changes are submitted and stored in the staging data set of the IMS catalog.

A database resource is considered *activated* when the ACB for the resource is imported into the IMS directory data set and the resource is loaded into memory by the online systems or will be the next time an online system schedules the resource. However, before you can use an activated database resource, you might need to take more steps. For example, for new databases, you might need to allocate database data sets, define the database in the RECON data set, initialize the database, or identify the database or program to the online system with the IMS type-2 command **`CREATE DB`** or **`CREATE PGM`** command. For a modified database, you might need to reorganize the database.

If you are using DDL CREATE statements to define a new database, IMS can activate the database automatically if `AUTOCREATE=YES` or `AUTOIMPORT(CREATE)` is specified in the `<CATALOG>` section of the DFSDF*xxx* PROCLIB member. If you are creating a DEDB database by using DDL, you can specify options for the automatic allocation of a DEDB area data set in the `<DDL>` section of the DFSDF*xxx* PROCLIB member.

If you define and generate your database resources by using the IMS generation utilities and submit your resources to IMS by using either the ACB Generation and Catalog Populate utility (DFS3UACB) or the IMS Catalog Populate utility (DFS3PU00), most resources cannot be activated automatically if any online systems are using the IMS catalog. DOPT PSBs can be activated automatically if your z/OS system supports extended sharing of PDSE data sets and you specify the `MANAGEDACBS=(UPDATE,SHARE)` when you run either of the Populate utilities.

Whether you use DDL or either of the Populate utilities, resources that are not activated immediately are stored in the staging data set of the IMS catalog, where they wait in a pending state until they are activated by issuing the **`IMPORT DEFN SOURCE(CATALOG)`** command.

Before you activate your resources, make sure that all of the prerequisite steps for activation are complete or at least planned for.

Whether IMS activates the resources automatically upon receiving DDL statements or later when you issue the **`IMPORT DEFN SOURCE(CATALOG)`** command, IMS uses the same internal *import* process to update the resources to the IMS directory data sets and to the online memory of all participating IMS systems. The internal import process that IMS uses is similar to the manual Online Change process that must be used to activate resources in online IMS systems that use ACB, DBD, and PSB libraries to manage ACBs.

This internal import process includes several prepare and commit phases that are equivalent to the prepare and commit phases of the online change function.

To determine the status of IMS import processing, especially if errors occur during activation, issue the QUERY MEMBER TYPE(IMS) SHOW(STATUS) command.

If the IMS catalog is shared by multiple IMS systems, see Activation when the IMS catalog is shared (System Administration).

Related reference

QUERY MEMBER command (Commands)

IMPORT DEFN SOURCE(CATALOG) command (Commands)

CATALOG and CATALOGxxxx sections of the DFSDFxxx member (System Definition)

DDL section of the DFSDFxxx member (System Definition)

IMS Catalog Populate utility (DFS3PU00) (System Utilities)

Activating altered resources when IMS manages ACBs

To introduce altered application programs (PSBs) and databases (DBDs) into online IMS systems when IMS manages ACBs, you submit the updated definitions to IMS, stop the affected resources, and issue the **`IMPORT DEFN SOURCE(CATALOG)`** command.

You can use either DDL or IMS generation utility macro instructions to define the altered resources.

The following procedure is for activating modifications that are stored as pending changes in the staging data set. The procedure activates changes in a single online IMS system or, when multiple IMS systems share the resources, in an IMSplex as a whole.

To activate only application program changes in a subset of IMS systems in an IMSplex, see "Activating PSBs in subsets of systems in an IMSplex" on page 429.

To activate only dynamic option (DOPT) PSBs, see Managing DOPT PSBs in IMS-managed ACB environments (System Utilities).

To activate altered PSBs, DBDs, or both in online IMS systems when IMS manages ACBs:

1. Modify the definitions by using either DDL or IMS generation utility macros.
   - For more information about modifying definitions with DDL, see Using DDL to define databases and program views (Database Administration).
   - For more information about modifying definitions with the IMS generation utilities, see Using the IMS generation utilities to design IMS databases (Database Administration).
2. Submit the modified definitions to IMS.
   - If you use DDL, submit the DDL statements to IMS by using either IMS Enterprise Suite Explorer for Development or the SQL Batch utility (Database Utilities).
   - If you use generation utility macro statements, after you run the DBD and PSB generation utilities, generate the ACBs and submit the changes to IMS in the same job step by using the ACB Generation and Catalog Populate utility (DFS3UACB) (System Utilities) with the MANAGEDACBS=STAGE control statement specified.

   If you use DDL, IMS holds the altered DBD and PSB definitions that are not activated automatically as pending changes in the staging data set of the IMS catalog.
3. If necessary, reorganize the database.
4. Stop all existing database and program resources that are affected by the changes.
5. Issue the IMPORT DEFN SOURCE(CATALOG) command (Commands).

   You can specify OPTION(DELPENDERR) on the IMPORT command to have IMS automatically clean up the staging data set if errors occur during the prepare phase of import processing. Cleaning up the staging data set after errors can help avoid more problems the next time the IMPORT command is issued.

   The value of the ACBSHR= parameter that is in effect on the command master IMS system can affect whether the other IMS systems in the IMSplex process an IMPORT DEFN SOURCE(CATALOG) command.

   - If ACBSHR=Y is specified in the command master, the IMPORT DEFN SOURCE(CATALOG) command is processed by all IMS systems in the IMSplex that specify ACBSHR=Y and any IMS systems that specify ACBSHR=N, if they are specified on the ROUTE parameter when the command is issued. For IMS systems that specify ACBSHR=N, the IMPORT command succeeds only if the IMS systems have pending ACB changes in the IMS catalog instance that they use.
   - If ACBSHR=N is specified on the command master, the command is processed by the command master and by any IMS systems that also specify ACBSHR=N and are specified on the ROUTE parameter. The command is rejected by IMS systems that specify ACBSHR=Y.

The IMS system that serves as the command master is selected by IMS from the IMS systems to which the IMPORT DEFN command was routed.

If you are activating many resources, the IMPORT DEFN SOURCE(CATALOG) command (Commands) command can take a significant amount of time to complete. To check the status of **IMPORT** command processing, issue the QUERY MEMBER command (Commands).

6. Restart the affected resources in all IMS systems.

Related concepts

"Activation when the IMS catalog is shared" on page 428
When multiple IMS systems that manage ACBs share an IMS catalog, you have limited control over the coordination of resource activation among the IMS systems in the IMSplex.

Related tasks

"Activating new resources when IMS manages ACBs" on page 427
To introduce new programs views (PSBs) and databases (DBDs) into online IMS systems when IMS manages ACBs, you submit the updated definitions to IMS and, if the resources are not activated automatically, issue the **IMPORT DEFN SOURCE(CATALOG)** command.

Activating new resources when IMS manages ACBs

To introduce new programs views (PSBs) and databases (DBDs) into online IMS systems when IMS manages ACBs, you submit the updated definitions to IMS and, if the resources are not activated automatically, issue the **IMPORT DEFN SOURCE(CATALOG)** command.

You can use either DDL or IMS generation utility macro instructions to define the altered resources.

If you use DDL, you can configure IMS to activate new DBDs and the associated PSBs automatically by specifying the AUTOIMPORT(CREATE) parameter in the CATALOG section of the DFSDF*xxx* PROCLIB member. IMS creates and activates the necessary control blocks. If the required data sets do not already exist, you must create them.

If you use DDL to define Fast Path DEDB databases and specify AUTOCREATE=YES in the CATALOG section of the DFSDF*xxx* PROCLIB member, IMS creates the data sets for the DEDB areas automatically in addition to creating and activating the DBD and PSB control blocks. You can define the attributes of these data sets in the DDL section of the DFSDFxxx PROCLIB member. When AUTOCREATE=YES is specified, you do not need to specify AUTOIMPORT(CREATE).

The following procedure activates the new resources in all IMS systems that share the same IMS catalog.

To activate new PSBs, DBDs, or both in online IMS systems when IMS manages ACBs:

1. Define the resources by using either DDL or IMS generation utility macros.
   - For more information about defining databases and program views with DDL, see Using DDL to define databases and program views (Database Administration).
   - For more information about defining DBDs and PSBs with the IMS generation utilities, see Using the IMS generation utilities to design IMS databases (Database Administration).
2. Submit the definitions to IMS.
   - If you use DDL, submit the DDL statements to IMS by using either IMS Enterprise Suite Explorer for Development or the SQL Batch utility (Database Utilities).
   - If you use generation utility macro statements, after you run the DBD and PSB generation utilities, generate the ACBs and submit the changes to IMS in the same job step by using the ACB Generation and Catalog Populate utility (DFS3UACB) (System Utilities) with the MANAGEDACBS=STAGE control statement specified.

   If you are creating a new database with DDL, the new DBD can be activated automatically depending on the automatic activation options in effect in the IMS system. Otherwise, IMS holds the ACBs for new DBDs and PSBs as pending changes in the staging data set of the IMS catalog.
3. If you are creating a new database, make sure that the following steps are complete, planned for, or are handled automatically by IMS:

- The creation of the database data sets.
- The creation of a PSB to access the new database.
- The creation of the runtime attribute control blocks (DDIRs and PDIRs) in the online IMS systems for the database and application programs.

4. Issue the IMPORT DEFN SOURCE(CATALOG) command (Commands).

   You can specify OPTION(DELPENDERR) on the IMPORT command to have IMS automatically clean up the staging data set if errors occur during the prepare phase of import processing. Cleaning up the staging data set after errors can help avoid more problems the next time the IMPORT command is issued.

   The value of the ACBSHR= parameter that is in effect on the command master IMS system can affect whether the other IMS systems in the IMSplex process an IMPORT DEFN SOURCE(CATALOG) command.

   - If ACBSHR=Y is specified in the command master, the IMPORT DEFN SOURCE(CATALOG) command is processed by all IMS systems in the IMSplex that specify ACBSHR=Y and any IMS systems that specify ACBSHR=N, if they are specified on the ROUTE parameter when the command is issued. For IMS systems that specify ACBSHR=N, the IMPORT command succeeds only if the IMS systems have pending ACB changes in the IMS catalog instance that they use.
   - If ACBSHR=N is specified on the command master, the command is processed by the command master and by any IMS systems that also specify ACBSHR=N and are specified on the ROUTE parameter. The command is rejected by IMS systems that specify ACBSHR=Y.

   The IMS system that serves as the command master is selected by IMS from the IMS systems to which the IMPORT DEFN command was routed.

   If you are activating many resources, the IMPORT DEFN SOURCE(CATALOG) command (Commands) command can take a significant amount of time to complete. To check the status of **IMPORT** command processing, issue the QUERY MEMBER command (Commands).

5. Restart the affected resources in all IMS systems.

Related concepts

“Activation when the IMS catalog is shared” on page 428
When multiple IMS systems that manage ACBs share an IMS catalog, you have limited control over the coordination of resource activation among the IMS systems in the IMSplex.

Related tasks

“Activating altered resources when IMS manages ACBs” on page 426
To introduce altered application programs (PSBs) and databases (DBDs) into online IMS systems when IMS manages ACBs, you submit the updated definitions to IMS, stop the affected resources, and issue the **IMPORT DEFN SOURCE(CATALOG)** command.

Activation when the IMS catalog is shared

When multiple IMS systems that manage ACBs share an IMS catalog, you have limited control over the coordination of resource activation among the IMS systems in the IMSplex.

IMS automatically coordinates resource changes globally among IMS systems that share IMS-managed ACBs. Internally, IMS uses a global coordination process that is similar to the process that is used by the Global Online Change process to activate changes in an ACBLIB data set.

IMS determines which IMS systems are sharing ACBs in an IMS catalog by the specification of the ACBSHR= parameter in each system. Only those systems that specify ACBSHR=Y are included in the global coordination of resource activation. IMS systems that specify ACBSHR=N are not. The ACBSHR= parameter is specified either in the `COMMON_SERVICE_LAYER` section of the DFSDF*xxx* member or in the DFSCG*xxx* member of the PROCLIB data set.

One IMS system functions as master for the global coordination of resource activation in managed ACB environments. To function as a master, ACBSHR=Y must be specified in the IMS system. The master updates the IMS catalog directory data sets and notifies the other IMS systems in the IMSplex that share the IMS catalog to refresh their resource definitions from the IMS directory. If resource activation is triggered by an **IMPORT DEFN SOURCE(CATALOG)**, the master includes all IMS systems that specify ACBSHR=Y in the global coordination of the resource activation even if the command is not explicitly routed to those systems.

If one of the sharing IMS systems is down while resources are activated globally, IMS takes note of the system that is down. If the IMS system comes up while the resources are still being activated, the restarting IMS system issues message DFS4373I and waits for the processing to complete before it continues. If the IMS system comes back up after the processing of the command is complete, the IMS system issues a message to indicate that the IMS catalog was updated while the IMS system was down. If the IMS system is restarting with a warm or emergency restart, the IMS system refreshes the ACBs from the IMS catalog.

Staging the activation of modified PSBs in a managed-ACB environment.

If you are changing PSBs only, you can stage the activation of the modified PSBs across an IMSplex by using the **IMPORT DEFN SOURCE(CATALOG)** command with the OPTION(UPDATEPSB) keyword. Staging the activation of PSBs allows you to maintain availability by activating and testing the modified PSBs on one IMS system while continuing to process work with the original PSB on other IMS systems. If DBDs are in the staging data set of the IMS catalog when the OPTION(UPDATEPSB) keyword is specified, the **IMPORT DEFN SOURCE(CATALOG)** is rejected.

Error scenarios during the activation of resources in a data sharing environment

If the master abends during the activation of resources, the other IMS systems in the data sharing group attempt to take over activation processing and clean up locally.

Only one of the other IMS systems that share the IMS catalog can take over as the new master to back out or commit the updates to the directory data set of the IMS catalog. If the new master IMS also abends or cannot complete the take over process successfully, you must resubmit the IMPORT DEFN SOURCE(CATALOG) command or the DDL statements to clean up the directory data sets.

If an IMS system abends during import processing, you can determine the status of both the import processing and the take over attempt by issuing the **QUERY MEMBER TYPE(IMS) SHOW(STATUS)** command.

Related tasks

"Activating PSBs in subsets of systems in an IMSplex" on page 429
If you are changing only PSBs in an IMSplex environment, you can activate the PSBs on a subset of the IMS systems by using the UPDATEPSB option of the **IMPORT DEFN SOURCE(CATALOG)** command. Later, you activate the PSBs on the other IMS systems in the IMSplex by using the REFRESHPSB option of the command.

Related reference

QUERY MEMBER command (Commands)
IMPORT DEFN SOURCE(CATALOG) command (Commands)
CATALOG and CATALOGxxxx sections of the DFSDFxxx member (System Definition)
COMMON_SERVICE_LAYER section of the DFSDFxxx member (System Definition)

Activating PSBs in subsets of systems in an IMSplex

If you are changing only PSBs in an IMSplex environment, you can activate the PSBs on a subset of the IMS systems by using the UPDATEPSB option of the **IMPORT DEFN SOURCE(CATALOG)** command. Later, you activate the PSBs on the other IMS systems in the IMSplex by using the REFRESHPSB option of the command.

Activating the PSBs on only a subset of systems enables programs to continue running with the original PSB on the other systems in the IMSplex while you test the PSB changes on the subset of IMS systems.

You can use the UPDATEPSB option for GSAM DBDs, so that when the IMPORT command is issued with the FOR(IMSID()) keyword, any GSAM DBDs in the staging data set are activated in the IMS directory.

Specify the FOR(IMSID()) keyword without the OPTION() keyword to perform one or all of the following operations:

- Updating staged PSBs and GSAM DBDs
- Refreshing pending PSBs

You specify the subset of IMS systems that process the UPDATEPSB or REFRESHPSB option on the FOR(IMSID()) keyword when you issue the IMPORT command. However, for initial routing of the command to specific IMS systems in the IMSplex, you must specify the routing requirements in TSO SPOC or other command interface when you first submit the command.

When the FOR(IMSID()) option is specified, you must route the command to one of the IMS systems that are specified on the FOR(IMSID()) keyword.

Use the UPDATEPSB option when 64-bit storage pools are enabled, IMS loads the old version of the non-resident PSBs into 64-bit storage and mark them as refresh pending on IMS systems that share the directory data sets but are not included on the FOR(IMSID()) keyword. For dynamic option (DOPT) PSBs and for IMS systems without 64-bit storage pools enabled, the effect of the UPDATEPSB option is no different than issuing the IMPORT DEFN SOURCE(CATALOG) command without OPTION(UPDATEPSB) specified.

The previous effects of the UPDATEPSB option also apply to the FOR(IMSID()) keyword without OPTION() keyword specified.

The IMS systems that are not specified in the FOR(IMSID()) keyword when the UPDATEPSB option is specified do not process the IMPORT command so do not reload their in-memory PSB definitions after the modified PSB definitions are imported into the IMS directory. However, after command processing completes, if an IMS system that was not specified on the command is required to reload its in-memory PSB definition from the IMS directory for any other reason, the modified PSB definition will become the active PSB definition in that IMS system.

Usually, the IMS systems that are excluded from the UPDATEPSB option continue to use the original, in-memory PSB definitions until either the IMPORT DEFN SOURCE(CATALOG) OPTION(REFRESHPSB) command is issued for the IMS system or the IMS system is restarted. In-memory PSB definitions that are prevented from reloading when the UPDATEPSB option is specified have a *refresh-pending* (REFP) status in the IMPORT command response.

To activate the PSB modifications in an IMS system and remove the PSBs refresh-pending status, issue the IMPORT DEFN SOURCE(CATALOG) OPTION(REFRESHPSB) command with the ID of the IMS system specified on the FOR(IMSID()) keyword. The REFRESHPSB option reloads all PSB definitions that have a refresh-pending status in the IMS system.

Related concepts

Activation when the IMS catalog is shared (System Administration)

Related reference

IMPORT DEFN SOURCE(CATALOG) command (Commands)

Activating modified PSBs in an initial subset of systems with UPDATEPSB

Use the OPTION(UPDATEPSB) keyword to introduce and activate modified PSBs for the first time in an IMSplex in only a subset of the online IMS systems.

Ensure that the staging data set of the IMS catalog does not contain any database definitions (DBDs) that are waiting to be activated in the IMSplex. If the staging data set contains pending DBD changes, the IMPORT DEFN command fails when the UPDATEPSB or REFRESHPSB option is specified.

To activate PSBs in a subset of IMS systems in an IMSplex:

1. Load modified PSB definitions into the staging data set of IMS catalog
2. Route workload away from the target IMS systems in which you are activating the PSBs.
3. Stop affected program resources in the target IMS systems.

4. In TSO SPOC or the OM API, explicitly route the following command to one or more of the IMS systems that are specified in the FOR(IMSID()) keyword: **`IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) FOR(IMSID())`**.

   The IMS system that serves as the command master for the UPDATEPSB option must be one of the IMS systems that are specified in the FOR(IMSID()) keyword. Routing the command to one of the specified systems ensures one of them acts as the command master.

   Upon processing the command, the modified PSBs are activated in the IMS systems that are specified on the FOR(IMSID()) keyword. In the IMS systems that are not specified on the FOR(IMSID()) keyword, the original definitions of the PSBs are still used, but the PSBs show a status of *refresh-pending* (REFP).
5. Restart the affected resources in the target IMS systems.
6. Test the new program definitions in the target IMS systems.
7. When testing is complete, resume workload on the target IMS systems with the new PSBs.

When you are ready to activate the modified PSBs in the other systems in the IMSplex, follow the procedure in "Activating modified PSBs in subsequent subsets of systems with REFRESHPSB" on page 431.

Related concepts

Activation when the IMS catalog is shared (System Administration)

Related reference

IMPORT DEFN SOURCE(CATALOG) command (Commands)

Activating modified PSBs in subsequent subsets of systems with REFRESHPSB

After modified PSBs are activated in an initial subset of IMS systems in an IMSplex by the UPDATEPSB option, you activate the modified PSBs in the remaining systems in the IMSplex by using the REFRESHPSB option of the IMPORT DEFN SOURCE(CATALOG) command.

Before you beginUse the REFRESPSB option only after the UPDATEPSB option.

Ensure that the staging data set of the IMS catalog does not contain any database definitions (DBDs) that are waiting to be activated in the IMSplex. If the staging data set contains pending DBD changes, the IMPORT DEFN command fails when the UPDATEPSB or REFRESHPSB option is specified.

The REFRESHPSB option causes all IMS systems that are specified on the FOR(IMSID()) keyword to reload all PSBs that have a refresh-pending (REFP) status in online memory. The IMS system reloads the PSB the next time the PSB is scheduled after the IMS system processes the IMPORT DEFN command with the REFRESHPSB option.

1. Route the workload away from the target IMS systems of the REFRESHPSB option.
2. Stop the affected resources on the target IMS systems.
3. In TSO SPOC or the OM API, specify one or more IMS systems to refresh on the FOR(IMS(ID)) keyword, and route this command to one of those systems: **`IMPORT DEFN SOURCE(CATALOG) OPTION(REFRESHPSB) FOR(IMSID())`**.
4. Start affected resources on the target IMS systems of the refresh.
5. Test new program definitions on the target IMS systems of the refresh.
6. When testing is complete, resume workload with the new PSBs on the target IMS systems of the refresh.

Related concepts

Activation when the IMS catalog is shared (System Administration)

Related reference

IMPORT DEFN SOURCE(CATALOG) command (Commands)

Example of activating modified PSBs in subsets of systems in an IMSplex

The following series of figures illustrate the gradual activation of a modified PSB across the systems within an IMSplex.

The IMSplex consists of five IMS systems, an IMS catalog with directory and staging data sets, and the Operations Manager (OM) of the IMS Common Service Layer (CSL). All of the IMS systems share the ACBs in the IMS catalog and directory data set, as indicated by the specification of ACBSHR=Y.

In the example, PSB1 is updated. The original version of PSB1 has a time stamp that is represented in the figures by *ts01*. The updated version of PSB1 has a later time stamp that is represented by *ts02*.

In the first figure of the series, the original version of PSB1 with time stamp *ts01* is running in three IMS systems in the IMSplex: IMS1, IMS3, and IMS5. IMS2 is down and IMS4 defines PSB1 as non-resident and does not currently have a copy in a buffer or 64-bit storage. The new version of PSB1 with time stamp *ts02* is ready to be activated after it is placed in the staging data set and the IMS catalog.

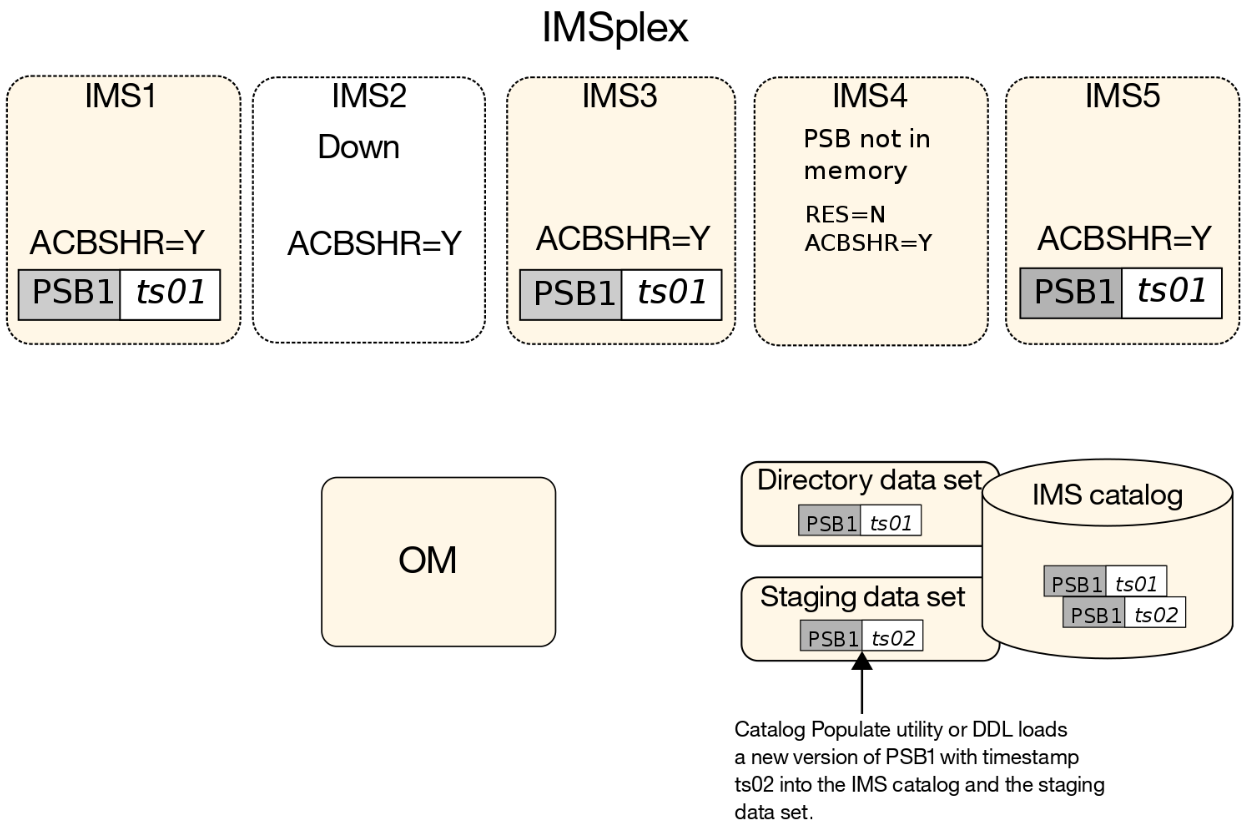


Figure 48. Initial state of PSB1 in the IMSplex before a new version is activated

In the following figure, the command **`IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) FOR(IMSID(IMS1))`** is issued to activate the new version of PSB1 with time stamp *ts02* on IMS1 only. To activate PSB1 on only IMS1, the command is both explicitly routed to IMS1 and specified in the FOR keyword. However, PSB1 is also considered activated on IMS4 because IMS4 defines PSB1 as non-resident and does not currently have a copy of PSB1 in memory. IMS4 reloads PSB1 from the IMS directory the next time the PSB is scheduled.

Because IMS3 and IMS5 are not included in the FOR keyword of the command, they do not refresh PSB1 and continue to use the original version that they have in memory. The output of the **`IMPORT DEFN`** command shows that a refresh is pending for PSB1 on IMS3 and IMS5.

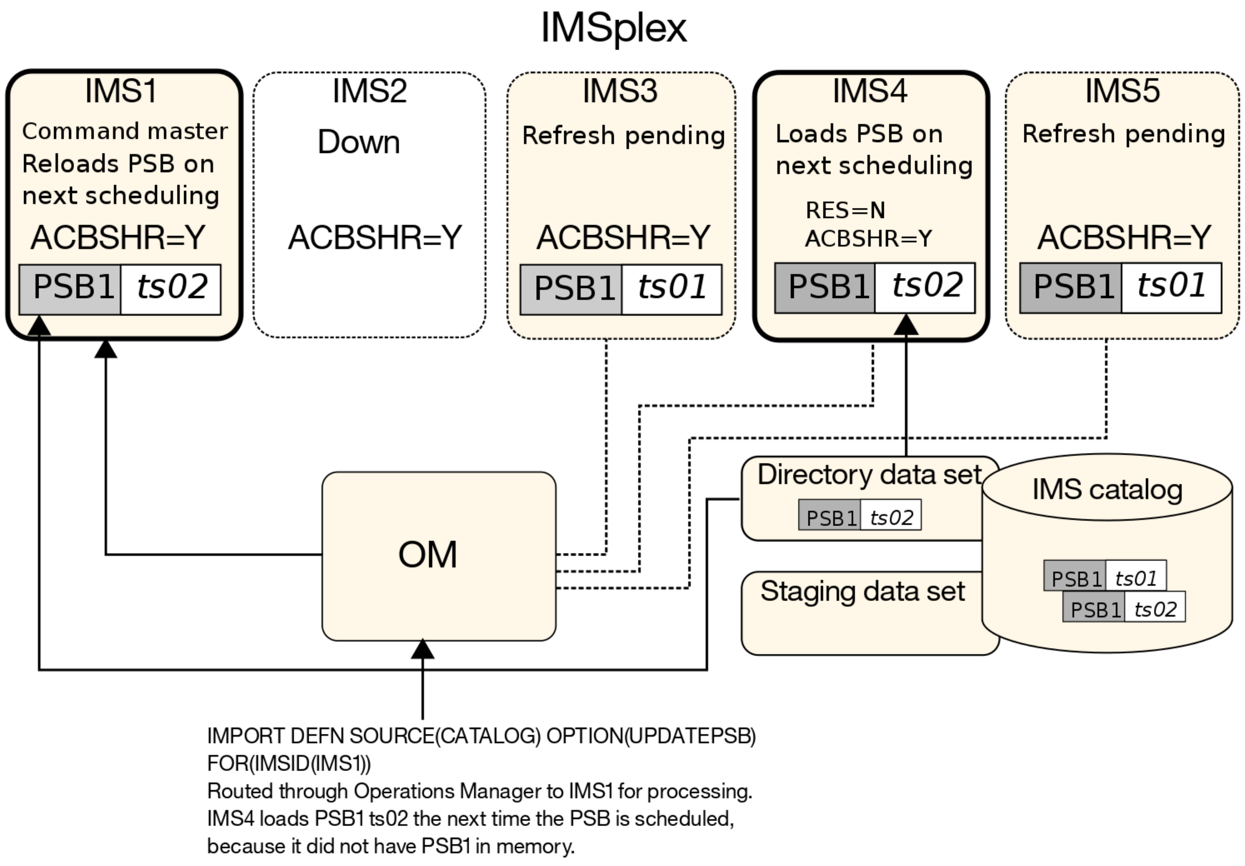

IMSplex
IMS1
Command master
Reloads PSB on next scheduling
ACBSHR=Y
PSB1 ts02
IMS2
Down
ACBSHR=Y
IMS3
Refresh pending
ACBSHR=Y
PSB1 ts01
IMS4
Loads PSB on next scheduling
RES=N
ACBSHR=Y
PSB1 ts02
IMS5
Refresh pending
ACBSHR=Y
PSB1 ts01
OM
Directory data set
PSB1 ts02
Staging data set
IMS catalog
PSB1 ts01
PSB1 ts02
IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) FOR(IMSID(IMS1))
Routed through Operations Manager to IMS1 for processing.
IMS4 loads PSB1 ts02 the next time the PSB is scheduled, because it did not have PSB1 in memory.

Figure 49. The new version of PSB1 is activated on IMS1. IMS2 loads the new version on scheduling.

In the following figure, IMS2, which was down when the **IMPORT DEFN SOURCE(CATALOG) OPTION(UPDATEPSB) FOR(IMSID(IMS1))** command, restarts. IMS2 is unaffected by the command and loads the new version of PSB1 from the IMS directory during restart.

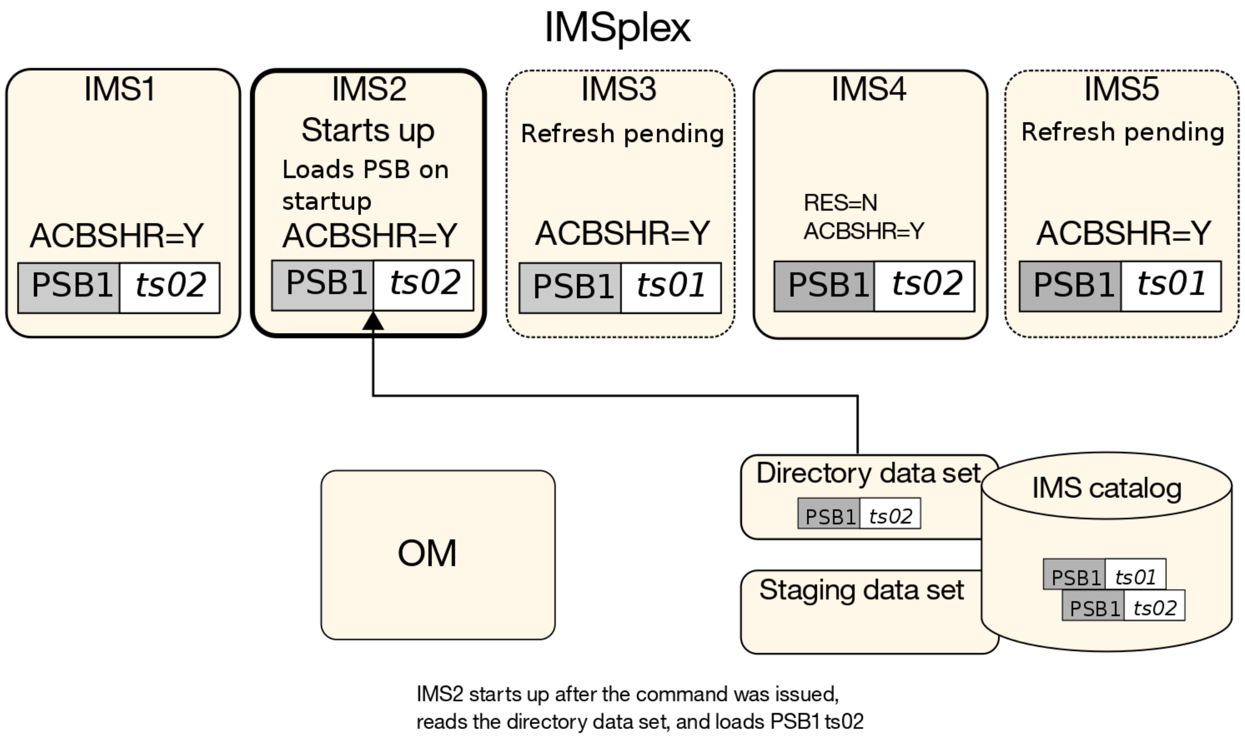


Figure 50. IMS2 loads PSB1 during restart

In the following figure, the command **`IMPORT DEFN SOURCE(CATALOG) OPTION(REFRESHPSB) FOR(IMSID(IMS3))`** is issued to activate the new version of PSB1 with time stamp *ts02* on IMS3 only. To activate PSB1 on IMS3, the command is explicitly routed to IMS1 although IMS3 is specified in the FOR keyword. When the REFRESHPSB option is specified, the IMS systems that are specified on the FOR keyword cannot serve as the command master.

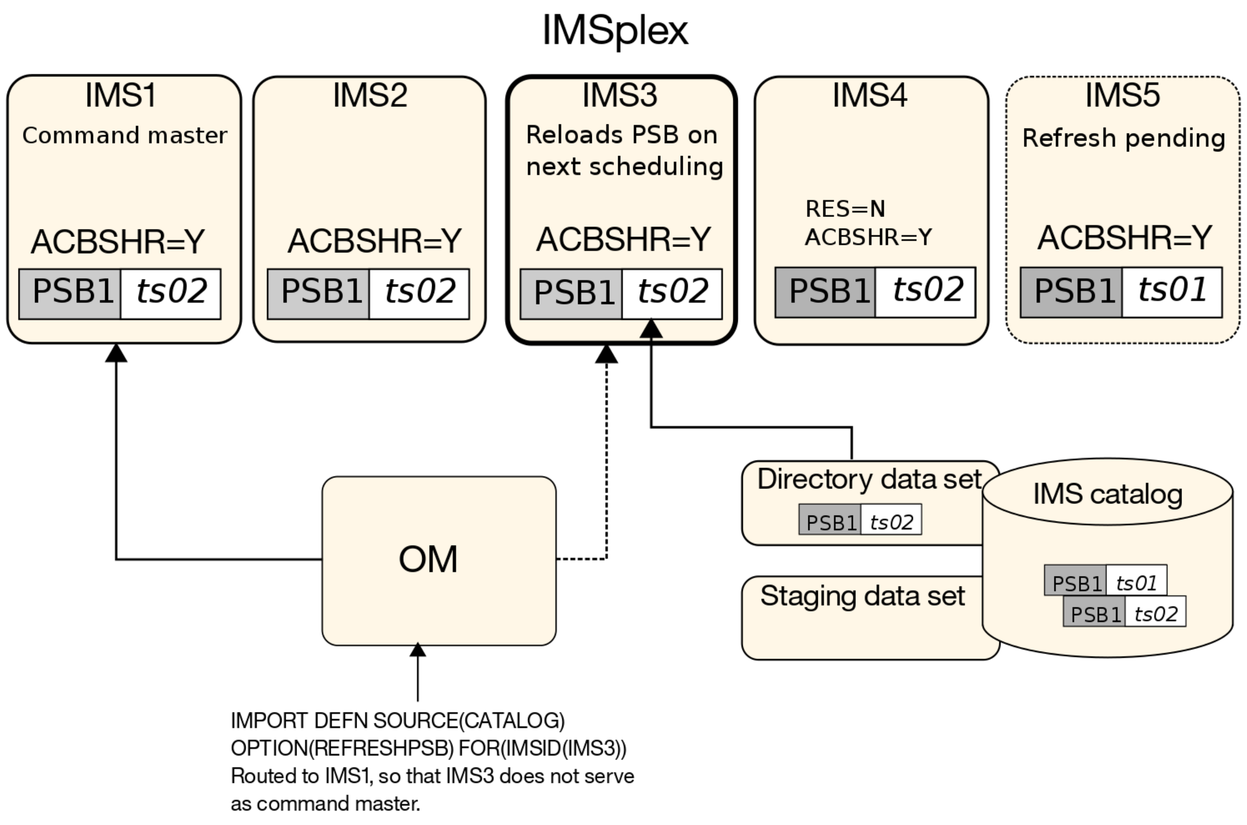


Figure 51. The REFRESHPSB option activates the new version of PSB1 on IMS3

In the following figure, IMS5 restarts. Before restarting, IMS5 had the original version of PSB1, which was in a refresh-pending state, loaded. However, during the restart, the refresh-pending state is removed and the new version of PSB1 is loaded from the IMS directory.

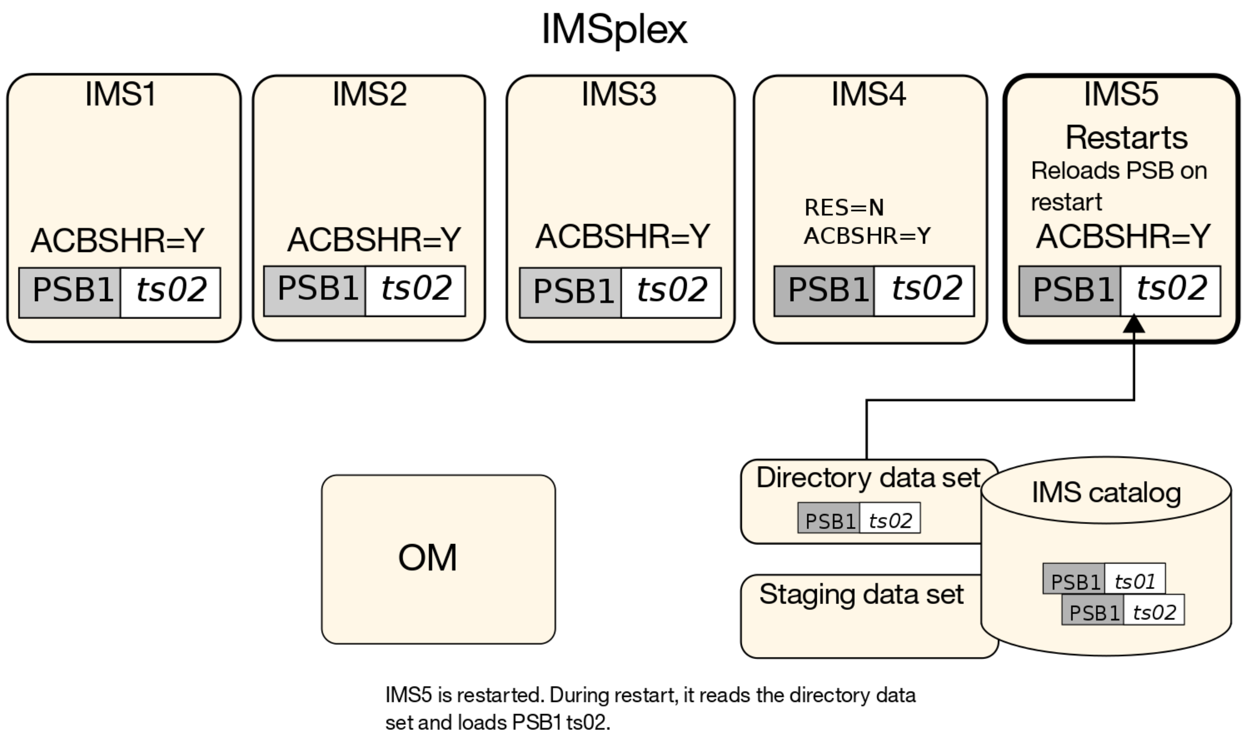


Figure 52. IMS5 restarts and loads the new version of PSB1

The online change function

There are two variations of the online change function: local online change and global online change.

Use the local online function change to make online changes to IMS resources for one IMS.

Use the global online change function to coordinate online changes to IMS resources across all IMS systems in an IMSplex.

Note: When the IMS management of ACBs is enabled, the online change function does not support changing ACBs. Instead, pending ACB changes are activated in online IMS systems by using the `IMPORT DEFN SOURCE(CATALOG)` command. For more information, see Online changes in managed-ACB environments (System Administration).

Related concepts

"ACB library member online change" on page 43
If an IMS system uses an ACB library in an IMSplex environment, you can use the ACB member online change (OLC) function to add or change individual members of the ACB library, or the entire ACB library, and bring these new or changed members online without quiescing the IMSplex or refreshing the active ACB library.

Related tasks

Activating database changes by using the online change function (Database Administration)

Overview of the local online change function

The ability to add, delete, and replace IMS databases, programs, transactions, and MFS formats online, without the necessity to bring down your IMS system, is a major step toward continuous operations. Adding, deleting, or changing IMS resources involves changes to the control blocks set up for these resources.

You can apply changes to the control block members within the IMS.FORMAT, IMS.ACBLIB, or IMS.MODBLKS data sets. You can apply changes to these data sets independently or in combination.

Restriction: The online change function is not supported for the following types of resources when dynamic definition of the resources is enabled:

- Database (DBD) and program view (PSB) resources when the IMS management of ACBs is enabled.
- MODBLKS resources when dynamic resource definition is enabled.

To use the IMS online change function, it is necessary to create three copies of each of the following libraries:

- IMS.MODBLKS–the library that contains the control blocks to support online change of databases, programs, transactions, and routing codes
- IMS.ACBLIB–the library that contains database and program descriptors
- IMS.FORMAT–the library that contains your MFS maps produced by the MFS Language and Service utilities

The libraries that are listed above are for the exclusive use of IMS offline functions and are called the staging libraries. For each library, a copy is made to produce a data set with a data set name suffixed with an A and a B, for example, IMS.FORMATA and IMS.FORMATB. These two copies of each library are used by the IMS online system.

At completion of IMS IVP processing the staging libraries and the IMS A libraries are identical. At this time, the A libraries are referred to as the active libraries. They are the libraries from which IMS draws its execution information. The B libraries are not used at this time and are referred to as the inactive libraries.

The following figure illustrates how libraries are used when you change your system online:

1. You apply changes to the staging libraries.
2. The staging libraries are subsequently copied to the inactive (B) libraries using the Online Change utility.

3. Operator commands are issued to:
   a. Query the resources affected by the change to verify that the change is the correct one.
   b. Cause the B libraries to become the active ones; the old active (A) libraries become the inactive ones.

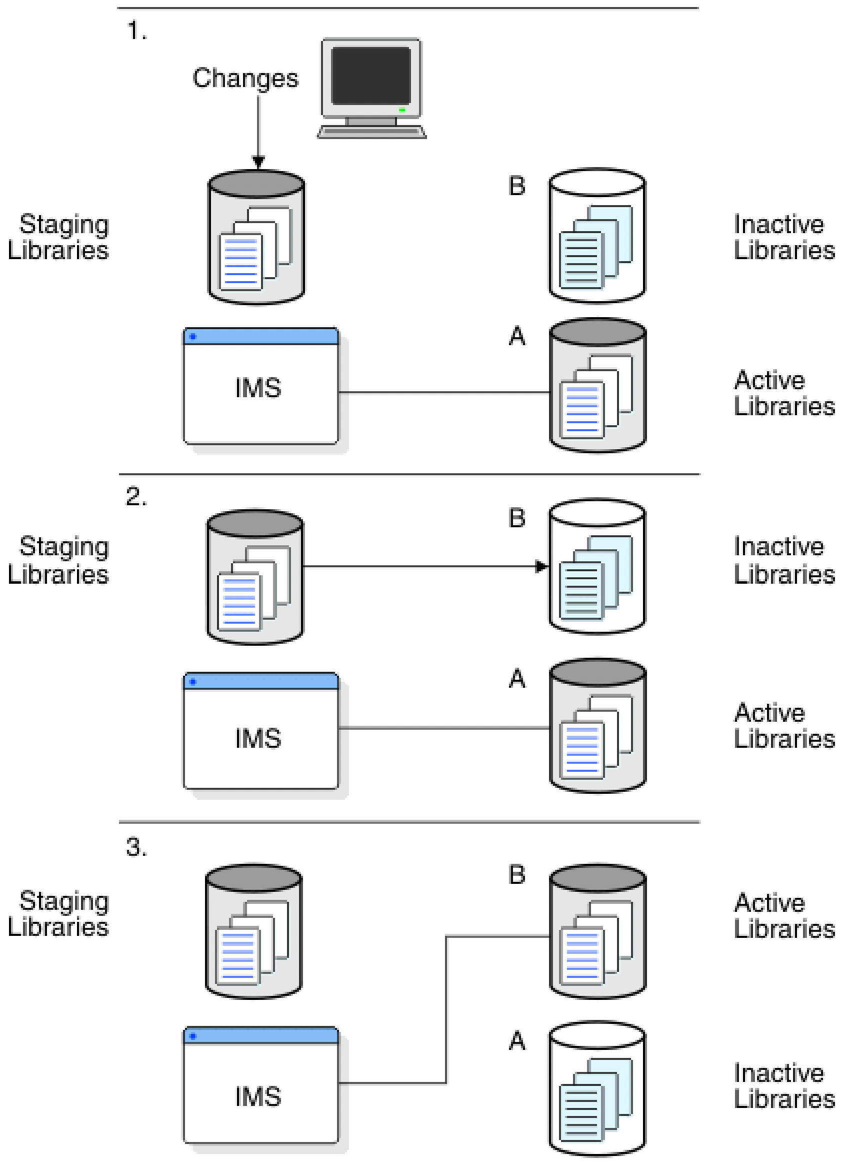


Figure 53. How libraries are used when you change your system online

The process above is repeated as necessary.

You can change individual members of the ACB library instead of performing a full library switch by using a variation of the global online change process. For details of all variations of the global online change process, see "Overview of the global online change function" on page 441.

When you choose to add, replace, or delete any of the IMS resources mentioned in this topic, you apply your changes to the offline staging libraries by running one of the following:

- A MODBLKS type of system definition–if you have added, changed, or deleted applications, programs, full-function databases, DEDBs, routing codes, or transactions.

  A MODBLKS system definition generates the control block members for resources that can be added or changed online. These control blocks are stored in the library IMS.MODBLKS, and are used by the IMS control region, and the Multiple Systems Coupling Verification utility when an online change to your IMS system is requested.

- The ACB Maintenance utility (DFSUACB0), if you have added or changed any databases or programs.
- The MFS Language and Service utilities–if you have added or changed any MFS format definitions.

After the sequence of commands (**/MODIFY** for local online change or **INITIATE OLC** for global online change) has been issued to cause the previously inactive libraries to become the active libraries, your previously active libraries now become the inactive libraries. They are not destroyed until they are overwritten by the next online change sequence. You can return to the inactive libraries if backup and recovery are necessary, or if an incorrect definition occurs during your online change run.

Additionally, IMS monitors for you which set of libraries is currently active. If local online change is enabled, this information is kept in a status data set, IMS.MODSTAT. If global online change is enabled, this information is kept in the IMSPLEX.OLCSTAT data set.

After an online change is successfully completed, it persists across all types of IMS restarts. Additionally, the new resources can be easily maintained by running an SMP/E JCLIN against the Stage 1 output stream produced by your MODBLKS system definition to record the contents of the new system definition in your SMP/E control data set. This ensures that any maintenance applied to your IMS system is applied to the currently active IMS system. Do not manage the online change data sets with a migration/recall system that might recall the data set to a volume other than the one to which it was originally allocated. If you do so, IMS might be unable to warm start or emergency start the system.

IMS environments that support the online change function

Global online change is not supported in all IMS environments that support local online change. Environments that require the MODSTAT/MODSTAT2 data sets do not support global online change. FDBR, XRF alternate and DBCTL-standby do not participate in global online process, however they keep track of the online change from the log records.

Use the following table to determine if global online change is supported in your environment.

Table 38. IMS environments that support online change

Environment	Supports local online change?	Supports global online change?
DB/DC	Yes	Yes
DBCTL	Yes	Yes
DBCTL-standby	No	No
DCCTL	Yes	Yes
FDBR	No	No
XRF active	Yes	Yes
XRF alternate	No	No

Commands applicable to online change

To perform a local online change operation, use the **/MODIFY PREPARE**, **/DISPLAY MODIFY**, and **/ MODIFY COMMIT** type-1 IMS commands.

You can also perform a local online change operation using type-2 IMS commands if you set up your IMS as a single-IMS IMSplex with a minimal Common Service Layer (CSL) consisting of a Structured Call Interface (SCI), an Operations Manager (OM), and no Resource Manager (RM). This IMS configuration is called the enhanced command environment. For information about defining your IMS with a minimal CSL (no RM), see *IMS Version 15.2 System Definition*. For information about performing a global online change operation for a single-IMS IMSplex (including performing an ACB library member online change), see "Overview of the global online change function" on page 441.

When you perform an online change, you make the inactive libraries (those containing the changes to be made) active, or copy members from the staging library to the active library. Use the following commands in the following sequence:

1. **/MODIFY PREPARE**

   Use the **/MODIFY PREPARE** command to prepare an IMS for local online change and specify which resources to add, change, or delete. When you issue this command, IMS prevents terminals from queuing messages to database programs or transactions that will be changed or deleted.

   You might have to issue additional commands after **/MODIFY PREPARE** to display and resolve work in progress that might cause the commit to fail.
2. **/DISPLAY MODIFY**

   Use the **/DISPLAY MODIFY** command to produce a list of the resources (that are to be changed or deleted) that have work pending. No changes are accepted when work is pending. Use this command to ensure no work is pending before issuing a **/MODIFY COMMIT** command.
3. **/MODIFY COMMIT**

   Use the **/MODIFY COMMIT** command to apply the changes specified in a preceding **/MODIFY PREPARE** command. When this command completes successfully, modifications persist across all IMS restarts.

If a **/MODIFY PREPARE** or a **/MODIFY COMMIT** command is unsuccessful and you do not want to retry the online change until another time, use the **/MODIFY ABORT** command to abort local online change on an IMS and reset the status to what it was before you issued the preceding **/MODIFY PREPARE** or **/MODIFY COMMIT** command.

Changing the system definition online

You do not always need to rerun a complete IMS system definition to make changes to the IMS subsystem. Examine the changes to see if an online change can be made.

If the request does not involve changes to the IMS network or to static terminals, you can probably make the change online.

The procedure to make changes to the IMS online system can be summarized as:

1. If necessary, make required system definition Stage 1 input changes.
2. If necessary, perform a MODBLKS system definition.
3. If necessary, run the MFS Language and MFS Service utilities.
4. If necessary, generate DBDs, PSBs, and ACBs.
5. As necessary, use the Online Change Copy utility to produce suitable copies of the IMS.MODBLKS, IMS.ACBLIB, and IMS.FORMAT, data sets.
6. Issue the appropriate online change command sequence to make the online change at the proper time. The changes must be copied from the inactive library for a library switch OLC.
7. Verify that the required changes took place.

If you need to undo or back out an online change, repeat the online change command sequence without changing the inactive libraries.

Changing programs and transactions

In an IMS DB/DC or DCCTL environment, the online change **PREPARE** command stops all incoming messages and transactions for a resource to be changed, but does not affect messages that have already been queued. The online change **COMMIT** command does not take effect until messages for a directly affected resource to be changed or deleted have been scheduled (dequeued), so you must either wait for the messages to be scheduled or stop the associated transactions.

If, for valid reasons, you do not want to wait for the messages to be scheduled (for the online change **COMMIT** command to take effect), use the **/STOP** or **/PSTOP** command to stop the associated transactions before issuing the online change **COMMIT** command.

Recommendation: Plan to make your changes when activity associated with the resource is low—when there are few transactions to be processed. When activity is low, you will not have to wait long for messages to be scheduled or have many transactions to stop if you do not want to wait.

You must stop all message-driven Fast Path programs and wait-for-input programs that directly refer to resources to be changed or deleted because the online change **`COMMIT`** command does not execute if any wait-for-input programs have been scheduled.

Also, any active conversations involving resources to be changed or deleted must complete before the online change **`COMMIT`** command can complete.

Implementing security changes online

While IMS is active, changes to RACF security definitions are made completely outside of IMS.

The following general procedure describes how to make changes to RACF:

1. In the RACF database, update the RACF security profiles that describe IMS resources (such as transactions and commands) by issuing RACF commands.
2. Issue the RACF command **`SETROPTS RACLIST(classname) REFRESH`** to refresh the profiles for the appropriate security class in the RACF data space.

Changing MFS formats

You can use the online change **`COMMIT`** command to change MFS formats without stopping related activity (messages can be queued and scheduled while you change an associated MFS format block).

But you should coordinate changing MFS formats and message queuing so that problems do not occur.

In the IMS DB/DC and DCCTL environments, the online change **`PREPARE`** command does not stop the queuing of related messages when you change MFS formats.

Making online changes for DEDBs

In the IMS DB/DC and DBCTL environments, you can make both database-level and area-level changes to Fast Path data entry databases (DEDBs).

A database-level change affects the structure of the DEDB, and includes such changes as adding or deleting an area, adding a segment type, or changing the randomizer routines. An area-level change involves increasing or decreasing the size of an area, including the overflow portions (IOVF and DOVF) and the control interval (CI) size.

A database-level change requires you to stop all areas of the DEDB. An area-level change requires you to stop the area using either the **`/DBRECOVERY AREA`** command.

IFP and MPP regions that access the changed DEDBs do not need to be stopped during the online change.

Related reading: For more information about adding, modifying, or deleting DEDBs online, see *IMS Version 15.2 Database Administration*.

Making online changes for HALDBs

The following restrictions apply for IMS online change with HALDB partitions.

- During an ALL or MODBLKS system definition, you cannot redefine a database that is currently defined as a HALDB partition without a performing cold start of IMS. A cold start is required even if you delete the database online. That is, if you delete a HALDB partition, you cannot reuse that database name for any database that you plan to add during an ALL or MODBLKS online change without an IMS cold start.
- You cannot convert a database that is defined during an ALL or MODBLKS system definition into a HALDB partition without performing a cold start of IMS. A cold start is required even if you delete the database online. After the IMS cold start, you must redefine the database as a HALDB partition.

These restrictions do not apply for a HALDB master database.

FDBR and local online change

FDBR tracks online changes through the x'70' log records.

To avoid problems if FDBR is shut down or restarted after an online change is performed, or to avoid problems under other circumstances after online change, take one or more IMS system checkpoints after the online change is complete and before FDBR is shut down or restarted. Doing so ensures that FDBR restarts from a checkpoint that was taken after the last online change.

For example, before you restart FDBR after an online change, issue the /CHECKPOINT command on the active system to take a simple checkpoint and verify from the DFS3804I message that the latest restart checkpoint time stamp is after the online change.

Making local online changes in an XRF complex

In the IMS DB/DC and DCCTL XRF environments, in addition to preparing the staging libraries, you need to prepare the inactive libraries.

If an XRF takeover occurs while the active IMS is processing a **`/MODIFY COMMIT`** command, the change might or might not have occurred. At the end of an online change, IMS writes a log record; the XRF alternate subsystem reads the log and makes the online changes to match the changes on the active subsystem. Use the **`/DISPLAY DB`** or **`QUERY DB`** command to check that a change occurred on the alternate subsystem.

Online change performance considerations

Deleting databases from the IMS.MODBLKS data set using Online Change incurs the performance overhead of reading the intent lists from the ACB library for all application programs in MODBLKS whose intent lists are not loaded.

This is necessary to determine which transactions need to stop queuing, to prevent work that might cause commit to fail. This performance overhead is proportional to the number of application programs defined in MODBLKS for which the intent list is not loaded.

Recommendation: If you need to delete databases from MODBLKS, consider performing an IMS cold start instead of using local or global online change.

Overview of the global online change function

Global online change is an IMS function to change resources for all IMS systems in an IMSplex. A Master IMS control region, as specified by the user or by an Operations Manager (OM), uses Resource Manager (RM) to coordinate the phases of online change with other IMS systems in an IMSplex.

Restriction: The online change function is not supported for the following types of resources when dynamic definition of the resources is enabled:

- Database (DBD) and program view (PSB) resources when the IMS management of ACBs is enabled.
- MODBLKS resources when dynamic resource definition is enabled.

For member online change, the member names specified on INIT OLC TYPE(ACBMBR) NAME(xxx) and their associated members are copied from staging to active. Staging may have members that are not copied. You can add or change individual members of IMS.ACBLIB and bring these members online without quiescing the IMSplex or requiring a full refresh of the active ACB library.

For library switch online change (local or global) the active and inactive libraries are swapped.

For more information about changing or adding members to IMS.ACBLIB online, see "Changing or adding IMS.ACBLIB members online" on page 450.

If you are running an IMSplex that does not use RM, you can specify the CSL global parameter RMENV=N in the DFSCGxxx IMS.PROCLIB member. If this parameter is specified, IMS does not require RM and will not use RM services, but type-2 command support is available. If OLC=GLOBAL, use the **`INITIATE OLC`** command to initiate local online change. The OLCSTAT data set must be defined. This data set cannot be shared between IMS systems.

With global online change, the Master IMS control region coordinates online change of the following resources among all the IMS systems in an IMSplex:

- Databases (DMB in IMS.ACBLIB)
- Database directories (DDIR in IMS.MODBLKS)
- MFS formats (IMS.FMTLIB)
- Programs (PSB-type ACBs in IMS.ACBLIB)
- Program directories (PDIR in IMS.MODBLKS)
- Transactions (SMBs in IMS.MODBLKS)

Commands applicable to global online change

To perform a global online change, issue a command to make the inactive libraries (those containing the changes to be made) active or to copy members from the staging library to the active.

The following commands make the inactive libraries active or copy the staging library to the active:

1. **`/DISPLAY MODIFY`**

   This command produces a list of the resources (that are to be changed or deleted) that have work pending. No changes are accepted when work is pending. Use this command to ensure no work is pending after issuing an **`INITIATE OLC PHASE(PREPARE)`** command and before issuing an **`INITIATE OLC PHASE(COMMIT)`** command.
2. **`INITIATE OLC PHASE(PREPARE)`**

   This command prepares all of the IMS systems in the IMSplex for global online change by specifying which resources to add, change, or delete.

   When running in an environment without RM (RMENV=N), this command performs the online change only at the command master IMS or the single IMS to which the command is routed.
3. **`INITIATE OLC PHASE(COMMIT)`**

   This command commits global online change on all IMS systems in an IMSplex.

   When running in an environment without RM (RMENV=N), this command performs a local online change on the IMS that processes it.

You might have to issue additional commands after **`INITIATE OLC PHASE(PREPARE)`** to display and resolve work in progress that might cause the commit to fail. These commands include:

TERMINATE OLC
: This command aborts global online change on all IMS systems in an IMSplex.

  When running in an environment without RM (RMENV=N), this command terminates a local online change on the IMS that processes it.

QUERY MEMBER
: This command displays status or attribute information about IMS systems in an IMSplex, including local or global online change status, if applicable.

QUERY OLC
: This command provides information about the OLCSTAT data set contents or resources being modified with TYPE(ACBMBR) OLC.

Changing the system definition online

You do not always need to rerun a complete IMS system definition to make changes to the IMS subsystem. Examine the changes to see if an online change can be made.

If the request does not involve changes to the IMS network or to static terminals, you can probably make the change online.

To make changes to the IMS online system:

1. If necessary, make required system definition Stage 1 input changes.
2. If necessary, perform a MODBLKS system definition.
3. If necessary, run the MFS Language and MFS Service utilities.
4. If necessary, perform DBDGEN, PSBGEN, and ACBGEN.
5. As necessary, use the Online Change Copy utility to produce suitable copies of the IMS.MODBLKS, IMS.ACBLIB, and IMS.FORMAT, data sets.
6. Issue the appropriate online change command sequence to make the online change. The changes must be copied from the inactive library for a library switch OLC.
7. Verify that the required changes took place.

If you need to undo or back out an online change, repeat the online change command sequence without changing the inactive libraries.

Changing programs and transactions

In an IMS DB/DC or DCCTL environment, the online change **PREPARE** command stops all incoming messages and transactions for a resource to be changed, but does not affect messages that have already been queued.

The online change **COMMIT** command does not take effect until messages for a directly affected resource to be changed or deleted have been scheduled (dequeued), so you must either wait for the messages to be scheduled or stop the associated transactions.

If you do not want to wait for the messages to be scheduled (for the online change **COMMIT** command to take effect), use the **/STOP** or **/PSTOP** command to stop the associated transactions before issuing the online change COMMIT command.

Recommendation: Plan to make your changes when activity associated with the resource is low—when there are few transactions to be processed. When activity is low, you will not have to wait long for messages to be scheduled or have many transactions to stop if you do not want to wait.

You must stop all message-driven Fast Path programs and wait-for-input programs that refer to resources to be changed or deleted because the online change **COMMIT** command does not execute if any wait-for-input programs have been scheduled.

Also, any active conversations involving resources to be changed or deleted must complete before the online change **COMMIT** command can complete.

Implementing security changes online

While IMS is active, you must make any changes to RACF security definitions completely outside of IMS.

The following general procedure describes how to make changes to RACF:

1. In the RACF database, update the RACF security profiles that describe IMS resources (such as transactions and commands) by issuing RACF commands.
2. Issue the RACF command **SETROPTS RACLIST(classname) REFRESH** to refresh the profiles for the appropriate security class in the RACF data space.

Changing MFS formats

In the IMS DB/DC and DCCTL environments, the online change **PREPARE** command does not stop the queuing of related messages when you change MFS formats.

You can use the online change **COMMIT** command to change MFS formats without stopping related activity (messages can be queued and scheduled while you change an associated MFS format block), but you should coordinate changing MFS formats and message queuing so that problems do not occur.

Making online changes for DEDBs

In the IMS DB/DC and DBCTL environments, you can make both database-level and area-level changes to Fast Path data entry databases (DEDBs). A database-level change affects the structure of the DEDB, and includes such changes as adding or deleting an area, adding a segment type, or changing the randomizer routines.

An area-level change involves increasing or decreasing the size of an area, including the overflow portions (IOVF and DOVF) and the control interval (CI) size.

A database-level change requires you to stop all areas of the DEDB. An area-level change requires you to stop the area using the **UPDATE AREA STOP(ACCESS)** command.

IFP and MPP regions that access the changed DEDBs do not need to be stopped during the online change.

Related reading: For more information about adding, modifying, or deleting DEDBs online, see *IMS Version 15.2 Database Administration*.

Making online changes for HALDBs

The following restrictions apply for IMS online change with HALDB partitions.

- During an ALL or MODBLKS system definition, you cannot redefine a database that is currently defined as a HALDB partition without performing a cold start of IMS. A cold start is required even if you delete the database online. That is, if you delete a HALDB partition, you cannot reuse that database name for any database that you plan to add during an ALL or MODBLKS online change without an IMS cold start.
- You cannot convert a database that is defined during an ALL or MODBLKS system definition into a HALDB partition without performing a cold start of IMS. A cold start is required even if you delete the database online. After the IMS cold start, you must redefine the database as a HALDB partition.

These restrictions do not apply for a HALDB master database.

Changing runtime resource definitions

You can create, update, and delete IMS resources and their descriptors dynamically by using DRD. However, you must use the ACB member online change function to change ACB members.

ACB members can be associated to a database or program resource as part of a **CREATE DB** or **CREATE PGM** command or with an online change. There are two ways that you can associate ACB members in an IMSplex:

- Create all ACBs in an ACBLIB and then use DRD to associate database and program resources to the ACB.
- Create the database and program resources using DRD commands and then perform an online change to associate the resources to the ACB.

CREATE commands for database and program resources do not require that complimentary ACB members exist in the ACBLIB. If ACB members exist for resources that are being created, the association is created between the resource and the ACB member when the **CREATE** command is issued, and an online change is not required.

Associating an ACB member using DRD commands

After running ACB generation, use the **CREATE DB** or **CREATE PGM** command to create a new database or program resource.

To associate an ACB member as part of the DRD **CREATE** command, complete the following steps:

1. Run ACB generation to create an executable ACBLIB member in a staging ACB library.
2. Issue the **INITIATE OLC** command to bring the ACB member into the active ACBLIB.
3. Issue the **CREATE DB** or **CREATE PGM** command to create a new database or program resource. The database is automatically associated to the ACBLIB through the **CREATE** command.

Associating an ACB member using the ACB member online change function

After running ACB generation, issue the **INITIATE OLC** command to associate the ACB member to the database or program resource.

To associate an ACB member with the ACB member online change function, complete the following steps:

1. Issue the **CREATE DB** command to create a new database.
2. Run ACB generation to create an executable ACBLIB member in a staging ACB library.
3. Issue the **INITIATE OLC** command with the TYPE(ACBMBR) parameter to copy the updated or new member from the staging library to the active library and associate the ACB member to the database or program resource.

Associating an ACB member using the ACB online change function

After running ACB generation, issue the **INITIATE OLC** command to associate the ACB member to the database or program resource.

To associate an ACB member with the ACB online change function, complete the following steps:

1. Issue the **CREATE DB** command to create a new database.
2. Run ACB generation to create an executable ACBLIB member in a staging ACB library.
3. Issue the **INITIATE OLC** command to perform a library switch online change and associate the ACB member to the database or program resource.

DRD command performance considerations

Deleting database resource definitions by using DRD commands incurs a performance overhead of reading the intent lists from the ACBLIB for all application program whose intent lists are not loaded.

The overhead reading the intents list is necessary to determine which transactions need to stop queuing to prevent work that might cause commit to fail. This performance overhead is proportional to the number of application programs defined for which the intent list is not loaded.

Checkpoints cannot run in parallel with any DRD command because the DRD command can change resource attributes and relationships. Checkpoints must include a clean copy of the resource definitions, which means that the resources were not changed during the checkpoint.

Related concepts

Restrictions for dynamic resource definition (System Definition)

FDBR and global online change

FDBR tracks online changes through the x'70' log records.

To avoid problems if FDBR is shut down or restarted after an online change is performed, or to avoid problems under other circumstances after online change, take one or more IMS system checkpoints after the online change is complete and before FDBR is shut down or restarted. Doing so ensures that FDBR restarts from a checkpoint that was taken after the last online change.

For example, before you restart FDBR after an online change, issue the /CHECKPOINT command on the active system to take a simple checkpoint and verify from the DFS3804I message that the latest restart checkpoint time stamp is after the online change.

Making local online changes in an XRF complex

In the IMS DB/DC and DCCTL XRF environments, in addition to preparing the staging libraries, you must prepare the inactive libraries.

If an XRF takeover occurs while the active IMS is processing a **INIT OLC PHASE(COMMIT)** command, the change might or might not have occurred. At the end of an online change, IMS writes a log record; the XRF alternate subsystem reads the log and makes the online changes to match the changes on the active subsystem. Use the **/DISPLAY DB** or **QUERY DB** command to check that a change occurred on the alternate subsystem.

Making global online changes in a sysplex that uses XRF

The XRF alternate does not participate directly in a global online change. The alternate participates indirectly once it receives the X'70' and X'22' log record written by the XRF active system.

The **`QUERY MEMBER`** command might show some online change status for the XRF alternate while it is processing the X'70' and X'22' log record.

During takeover of an active system, the XRF alternate updates the OLCSTAT data set with its IMSID. After the takeover is complete, the XRF alternate behaves like any IMS active system and can participate in a global online change. The abended XRF active system can be brought back up as the new XRF alternate, if desired.

The MODSTAT and MODSTAT2 data sets are not used. You can leave the MODSTAT and MODSTAT2 data set DD statements in your IMS control region JCL, but they are ignored.

Global online change in an XRF environment has the following limitations:

- If an XRF active system fails or is shut down (**`/SWITCH SYSTEM ACTIVE`** or **`/CHE FREEZE`**) and the XRF alternate system takes over, IMS deletes the IMSID of the failed or shut down XRF active system from the OLCSTAT data set during takeover processing, so that global online change can work even if the old active is down. After takeover, if you bring up the failed or shut down XRF active system as the new XRF alternate system, and initiate global online change, the new XRF alternate system does not process the online change command directly, but performs the online change later when it detects the online change log record.
- If the XRF active system fails and the XRF alternate system attempts to take over before it catches up with the XRF active system's global online change state, the XRF alternate system abends with U2801 RC=X'0011'. The XRF alternate is not synchronized with the OLCSTAT data set. Because other IMS systems in the IMSplex might be synchronized with the OLCSTAT data set, the XRF alternate is not permitted to takeover.

Global online change after XRF takeover and DBCTL standby emergency restart

During the XRF takeover, the IMSID of the old active is removed from the OLCSTAT and the IMSID of the new active is added to the OLCSTAT. If there was any error removing the active's IMSID, the takeover completes, but two IMSIDs will exist in the OLCSTAT (the old XRF active and the new XRF active IMSIDs).

After an emergency restart of a DBCTL standby with RMENV=N and OLC=GLOBAL, two IMSIDs exist in the OLCSTAT data set (the old DBCTL active and new DBCTL active IMSIDs).

You can run the Global Online Change utility (DFSUOLC0) with FUNC=DEL to delete the old active XRF or old DBCTL active IMS systems IMSID from the OLCSTAT data set. Otherwise, the next **`INITIATE OLC PHASE(PREPARE)`** command you issue to initiate global online change will fail if OPTION(FRCABND,FRCNRML) is not specified.

Online change performance considerations

Depending on what you are using the online change function for, different performance considerations apply.

Deleting databases from the MODBLKS data set

Deleting databases from MODBLKS using Online Change incurs the performance overhead of reading the intent lists from ACBLIB for all application programs in MODBLKS whose intent lists are not loaded.

This is necessary to determine which transactions need to stop queuing, to prevent work that might cause commit to fail. This performance overhead is proportional to the number of application programs defined in MODBLKS for which the intent list is not loaded.

Recommendation: If you need to delete databases from MODBLKS, consider performing an IMS cold start instead of using local or global online change.

Modifying ACB members

By default, when a DBD is specified on the NAME keyword of the **`INITIATE OLC PHASE(PREPARE) TYPE(ACBMBR)`** command, the ACB member online change (OLC) function automatically copies all associated PSBs and externally referenced DBDs from the staging ACB library to active ACB library. Depending on the number of PSBs and DBDs, the amount of time required for the copying process can be significant.

In many cases, you can significantly reduce the amount of time the ACB member online change function takes by specifying OPTION(NAMEONLY) on the **`INITIATE OLC PHASE(PREPARE) TYPE(ACBMBR)`** command. The NAMEONLY option limits the processing of the ACB member online change function to only the DBDs and PSBs that are specified in NAME() keyword of the command.

Related reference

INITIATE OLC command (Commands)

Making local online changes in a sysplex

IMS does not automatically coordinate local online changes across a sysplex. Use global online change to coordinate online change across a sysplex. This topic describes how to manually coordinate local online changes across a sysplex.

Recommendation: Clone your IMS subsystems and share libraries (ACBLIB, MODBLKS, FORMAT) across the sysplex. MODSTAT IDs must be the same across the sysplex.

To prepare for performing IMS online change in a sysplex environment:

1. Use an Automated operator (AO) application program. The AO application program should be able to:
   - Stop resources you want to change or delete.
   - Handle online change exception conditions for an individual IMS subsystem.
   - Issue the **`/MODIFY PREPARE`** or **`/MODIFY ABORT`** commands.

   Design the AO exit routine to look for the online change messages that IMS issues and to take the suggested action. Design an AO application program or use an automation tool to issue commands to resolve any online change problems.
2. Define the active, inactive, and staging libraries.
3. Share the active and inactive libraries.

   All IMS subsystems that you want to be able to change online should also share the staging libraries.
4. Define each IMS (with its own MODSTAT data set).

   Each IMS must have its own MODSTAT data set, with the same active and inactive ACBLIB, FMTLIB, and MODBLKS definitions.
5. Start each IMS.

To perform the online changes:

1. Perform system definitions on the staging libraries for changes.
2. Run the Online Change Copy utility.
3. Issue the **`/MODIFY PREPARE`** command for each IMS subsystem.

   You can have either the IMS operator or an AO application program issue this command. If the command fails on any IMS subsystem in the sysplex, the operator or AO application program must issue the **`/MODIFY ABORT`** command on all IMS subsystems for which the **`/MODIFY PREPARE`** succeeded.
4. **Optional:** Stop all resources that will be changed or deleted.

   This step greatly increases the chance that the online change will succeed for an individual IMS subsystem.
5. Resolve any work in progress.

6. Issue the **/DISPLAY MODIFY** command.

   Use the **/DISPLAY MODIFY** command to display the work in progress for resources to be changed or deleted, before attempting the COMMIT phase. When COMMIT is successful, the modifications persist across all IMS restarts, unless global online change occurs while an IMS is down.
7. Issue **/MODIFY ABORT|COMMIT** for each IMS subsystem.
8. Start resources that were changed or added.

Enabling the global online change function

Use the Global Online Change utility (DFSUOLC) to initialize the OLCSTAT data set and enable an IMSplex for global online change.

To enable an IMSplex for global online change, you must:

1. Run the Global Online Change utility (DFSUOLC0) to initialize the OLCSTAT data set.

   The Global Online Change utility needs to be run once before the first IMS cold starts the first time. The OLCSTAT data set is comparable to the MODSTAT data set used by local online change.

   Important:

   - Use the Global Online Change utility with caution, so you do not inadvertently destroy valid OLCSTAT data set contents. If you do destroy valid OLCSTAT data set contents, global online change and initialization of additional IMS systems fail until the OLCSTAT data set is re-initialized.

     You should establish an OLCSTAT data set recovery procedure to deal with the loss of the OLCSTAT data set. After every successful global online change, record the modify ID, the active online change library suffixes, and the list of IMS systems that are current with the online change libraries. If the OLCSTAT data set is destroyed, run the initialize function of the Global Online Change utility with the saved data to re-initialize the OLCSTAT data set.

     Related reading: For more information about the Global Online Change utility, see *IMS Version 15.2 System Utilities*.
   - If you build a multisystem sysplex, you automatically build a global resource serialization complex with it. Global online change relies on global resource serialization to serialize access to the OLCSTAT data set from multiple systems. If you use global online change in a sysplex with more than one z/OS system, you should not define the GRSRNL=EXCLUDE parameter in the MVS parmlib. Defining GRSRNL=EXCLUDE makes the OLCSTAT data set serialization local instead of global.
   - For an IMSplex running without an RM (with CSL global parameters RMENV=N and OLC=GLOBAL), you must ensure that another IMS subsystem does not attempt to use the OLCSTAT data set from the IMS subsystem that is running without RM.
2. For each IMS in the IMSplex, perform these steps:
   a) Remove the MODSTAT DD and MODSTAT2 DD statements from the IMS control region JCL.
   b) Define DFSCGxxx IMS.PROCLIB member parameters related to global online change, or in the CG section of DFSDFxxx member.

      Specify OLC=GLOBAL to enable global online change.

      Specify OLCSTAT= with the online change status data set name. All of the IMS systems in the IMSplex must specify the same OLCSTAT data set.
   c) Shut down IMS.
   d) Cold start the IMS.

Initiating a global online change

The **INITIATE OLC** (initiate online change) command is provided to support global online change, where online change is coordinated across IMS systems in the IMSplex. **INITIATE OLC** is similar to **/MODIFY**

PREPARE and **/MODIFY COMMIT**, except that it applies to an IMSplex-wide global online change. OM sends the **INITIATE OLC** command to one IMS in the IMSplex.

For syntax, environment, and usage information, see *IMS Version 15.2 Commands, Volume 1: IMS Commands A-M*.

Note: When running in an environment without RM (RMENV=N), this command performs a local online change on the IMS that processes it.

INITIATE OLC prepares for or commits a global online change of IMS resources across an IMSplex.

INITIATE OLC is not supported if local online change is enabled. The **INITIATE OLC PHASE(PREPARE)** command is rejected if the IMS to which the command is routed does not support global online change. If the **INITIATE OLC PHASE(PREPARE)** command is rejected and there is an IMS that supports global online change, route the command to an IMS that supports global online change.

The correct online change command sequence is the following:

1. **INITIATE OLC PHASE(PREPARE)**
2. **/DISPLAY MODIFY**

   Use the **/DISPLAY MODIFY** command to display the work in progress for resources to be changed or deleted, before attempting the COMMIT phase. When COMMIT is successful, the modifications persist across all IMS restarts, unless global online change occurs while an IMS is down.
3. **INITIATE OLC PHASE(COMMIT)**

If the **INITIATE OLC PHASE(PREPARE)** is specified without a FRCABND or FRCCNRML keyword and the command fails because one or more IMS are down or go down before the online change is committed, the online change must be aborted and started over. Issue the **TERMINATE OLC** command to abort the online change.

If the **INITIATE OLC PHASE(PREPARE)** is specified with the FRCNRML keyword and the command fails for any IMS, you can proceed with an **INITIATE OLC PHASE(COMMIT)** command after shutting down those IMS systems where the prepare failed. Otherwise you must abort the online change and start over.

If the **INITIATE OLC PHASE(PREPARE)** is specified with the FRCABND keyword and the command fails for any IMS, you can proceed with an **INITIATE OLC PHASE(COMMIT)** command after canceling those IMS systems where the prepare failed. Otherwise, abort the online change and start over.

If the **INITIATE OLC PHASE(COMMIT)** command fails for any IMS before the OLCSTAT data set is updated, you can either correct the errors and try the commit again or abort the online change with a **TERMINATE OLC** command.

If the **INITIATE OLC PHASE(COMMIT)** command fails for any IMS after the OLCSTAT data set has been updated, you can correct the errors and try the commit again. The online change cannot be aborted.

If an IMS abends during online change and the **INITIATE OLC PHASE(PREPARE)** command was not specified with FRCABND, then issue the **TERMINATE OLC** command to abort the online change. The **INITIATE OLC PHASE(COMMIT)** command is not permitted in this case. If an IMS abends during online change and the **INITIATE OLC PHASE(PREPARE)** command was specified with FRCABND, then the **INITIATE OLC PHASE(COMMIT)** command is permitted.

Type-1 commands and type-2 commands that come from the OM interface are rejected during the commit phase, if the command changes resources. Commands that change resources could interfere with the online change of the resources. Type-1 commands and type-2 commands that come from the OM interface are permitted during the commit phase, if the command (such as **/DISPLAY** or **QUERY**) displays resources. Type-1 commands that are entered from the system console or an IMS terminal are queued during the online change commit phase. These commands run after the online change is committed or aborted.

You can specify **INITIATE** only through the OM API.

INITIATE is invalid on an XRF alternate and FDR system.

Each IMS participating in the global online change does not issue the same synchronous online change messages to the master terminal or system console that it does for a local online change. The OM command response contains information equivalent to the online change messages that appear for the local online change, such as the DFS3499 message contents.

Each IMS participating in the global online change may issue asynchronous online change messages to the system console, such as DFS3400, DFS3445, and DFS3498. For more information about these messages, see *IMS Version 15.2 Messages and Codes, Volume 1: DFS Messages*.

The OM command timeout default of 300 seconds (5 minutes) might not be enough time for the online change phase to complete. You might need to specify a timeout value on the command based on the needs of the installation.

INITIATE OLC return and reason codes

A return and reason code is returned to OM by the **`INITIATE OLC`** command. The return and reason codes for all IMS commands that are routed from OM are defined in the DFSCMDRR macro in the IMS.SDFSMAC data set.

The **`INITIATE OLC`** command master usually performs the online change phase locally first. If the online change phase fails locally, the command master usually skips sending the online change phase to the other IMS systems, sets a completion code for each other IMS indicating that the online change phase was not attempted, and terminates command processing. However, if the **`INITIATE OLC PHASE(COMMIT)`** command fails on the local IMS because of work in progress, the command master still sends the commit phase 1 to the other IMS systems. The purpose is to report work in progress for all the IMS systems in the IMSplex, to facilitate completion of work in progress.

In a mixed IMSplex, you might have some IMS systems that support a particular version of online change and some that do not. For the command to be considered successful, at least one IMS in the IMSplex must successfully perform the online change phase. If no IMS in the IMSplex supports the same version of online change, the command reason code indicates that none of the IMS systems performed the online change phase. If you enter a version of the **`INITIATE OLC PHASE(PREPARE)`** command that does not apply to any IMS in the IMSplex, you must terminate the online change by using a **`TERMINATE OLC`** command.

Related reference

Return and reason codes for commands to OM (Messages and Codes)

Handling errors for the INITIATE OLC command

The **`INITIATE OLC`** command can result in errors that leave one or more of the IMS systems in the IMSplex in various online change states. Correct the error. Issue the **`QUERY MEMBER`** command and the **`QUERY OLC`** command to help you decide whether to terminate the online change by issuing **`TERMINATE OLC`** or to try the **`INITIATE OLC`** command again.

Before attempting a global online change, issue the **`QUERY OLC LIBRARY(OLCSTAT) SHOW(MODID)`** command to get the current modify ID. If the **`INITIATE OLC`** command fails, issue the **`QUERY OLC LIBRARY(OLCSTAT) SHOW(MODID)`** command again, to see if the modify ID is the same. If the modify ID increased by 1, the online change is considered to be successfully completed.

If the **`INITIATE OLC`** command fails, issue the **`QUERY MEMBER TYPE(IMS) SHOW(STATUS)`** command to display the online change state of all the IMS systems in the IMSplex. Evaluate the **`QUERY MEMBER TYPE(IMS) SHOW(STATUS)`** output to help you to determine what to do. For information about the **`QUERY MEMBER TYPE(IMS) SHOW(STATUS)`** output and the action to take, see "Displaying status or attribute information about IMS systems in the IMSplex" on page 452.

Changing or adding IMS.ACBLIB members online

In an IMSplex environment, you can use the ACB member online change (OLC) function to add or change individual members of the ACB library, or the entire ACB library, and bring these new or changed members online without quiescing the IMSplex or refreshing the active ACB library.

Members that are not affected by the member OLC are not quiesced.

You can make online changes to the ACB library in IMSplex environments in which:

- A single IMS system comprises the IMSplex. In this configuration, IMS must use the OLCSTAT data set with a CSL that consists of an OM and SCI. An RM is not required.
- Multiple IMS systems comprise the IMSplex. In this configuration, each IMS system must use the same shared OLCSTAT data set, or each system must use its own local OLCSTAT data set. The CSL must consist of an OM, an SCI, and an RM. Although a resource structure is recommended, it is not required.

Global online change for ACBLIB members can only be performed in an IMSplex.

You can specify whether the ACBLIB member is shared or dedicated in the IMSplex. The following example highlights how the ACBLIB member online change process can be coordinated among all IMS systems that share the OLCSTAT data set:

- OM chooses an IMS system to be the command master IMS.
- The command master IMS coordinates the member OLC process with other IMS systems sharing the OLCSTAT data set using RM.
- ACBSHR= must be specified as Y or N in the DFSCGxxx or DFSDFxxx PROCLIB member.
  - Y– Indicates that all of the IMS systems in the OLCSTAT are using the same active and inactive ACBLIB.
  - N– Indicates that each IMS in the OLCSTAT is using its own dedicated active and inactive ACBLIB.
- All IMS systems in the IMSplex that share the OLCSTAT data set must specify the value of ACBSHR=.
- The member OLC updates all the active ACBLIBs, whether or not they are shared.

The ACB member online change uses IMS type-2 commands only. The commands that are involved in performing an ACB library member online change are:

- **INITIATE OLC PHASE(PREPARE)**
- **INITIATE OLC PHASE(COMMIT)**
- **TERMINATE OLC**
- **QUERY MEMBER TYPE(IMS)**
- **QUERY OLC SHOW(RSCLIST)**

By default, when a DBD is specified on the NAME keyword of the **INITIATE OLC PHASE(PREPARE) TYPE(ACBMBR)** command, the ACB member online change (OLC) function automatically copies all associated PSBs and externally referenced DBDs from the staging ACB library to active ACB library. Depending on the number of PSBs and DBDs, the amount of time required for the copying process can be significant.

In many cases, you can significantly reduce the amount of time the ACB member online change function takes by specifying OPTION(NAMEONLY) on the **INITIATE OLC PHASE(PREPARE) TYPE(ACBMBR)** command. The NAMEONLY option limits the processing of the ACB member online change function to only the DBDs and PSBs that are specified in NAME() keyword of the command.

If you want to fall back to the previous version of the changed resource, perform a full online change process with a full library switch.

To perform an ACB member online change:

1. Define the IMS system with online change capability by removing local MODSTAT definitions and defining the OLCSTAT data set in the DFSCGxxx PROCLIB member.
2. Run the PSB generation to generate or update PSBLIB members.
3. Run the DBD generation to generate or update DBDLIB members.
4. Run the ACB generation to create an executable ACBLIB member in a staging ACB library.

   The staging ACB library can be a clone or a subset of the active ACB library.
5. Optionally, create a backup copy of the active ACBLIB member for fallback purposes; you can use the Online Change Copy utility (DFSUOCU0) to do this.
6. Issue the **INITIATE OLC** command with the TYPE(ACBMBR) parameter to copy the updated or new member from the staging library to the active library.

If your ACBs are in 64-bit storage, the ACB members that are affected by the online change process are removed from the non-resident pool and deleted from the 64-bit pool. The next time an application is scheduled that needs the new or updated ACB members, the members are read from the ACB library and copied to the non-resident pool for execution and are also copied to the 64-bit pool.

If you want to fall back to the previous version of the changed resource, perform a full online change process with a full library switch.

Related reference

INITIATE OLC command (Commands)

Displaying status or attribute information about IMS systems in the IMSplex

The **QUERY MEMBER** command displays status or attribute information about IMS systems in the IMSplex, including online change status.

You can specify **QUERY MEMBER** only through the OM API. For syntax, environment, and usage information, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Displaying information about the OLCSTAT data set

The **QUERY OLC LIBRARY** command provides information about the OLCSTAT data set contents. For syntax, environment, and usage information, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Terminating an online change

The **TERMINATE OLC** command supports aborting a global online change across an IMSplex. The **TERMINATE OLC** command can also be shortened to **TERM OLC**.

A **TERMINATE OLC** command that aborts a global online change is similar to the **/MODIFY ABORT** command, except that a global command applies to all IMS systems in an IMSplex that are participating in the global online change.

OM sends the **TERMINATE OLC** command to an IMS in the IMSplex.

You can use **TERMINATE OLC** to abort an IMSplex-wide global online change initiated by a **INITIATE OLC PHASE(PREPARE)**, before the online change is successfully committed with a **INITIATE OLC PHASE(COMMIT)**.

Use **TERMINATE OLC** to abort an online change after an **INITIATE OLC PHASE(COMMIT)** failure that occurs before the OLCSTAT data set is updated. After the commit process has updated the OLCSTAT data set, the online change is considered to be successful and cannot be aborted.

TERMINATE OLC is not supported if local online change is enabled. The **TERMINATE OLC** command is rejected if the IMS to which the command is routed does not support global online change. If this occurs and there is an IMS that supports global online change, the user must route the command to a specific IMS that supports global online change.

You can specify **TERMINATE OLC** only through the OM API.

The **TERMINATE OLC** command can be processed by DB/DC, DBCTL, and DCCTL environments.

The **TERMINATE OLC** command is invalid on the XRF alternate and FDR system.

The OM command timeout default of 300 seconds (5 minutes) might not be enough time for the online change phase to complete. You might need to specify a timeout value on the command based on the needs of the installation.

The command syntax for this command is defined in XML and is available to automation programs which communicate with OM. For more information about command syntax, environments, and usage, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

Handling errors for global online change commands

When a global online change command results in an error, the action you perform depends on the state of the IMS systems in the IMSplex after the online change error has occurred.

The following table describes actions that you can perform when an online change command results in an error.

Table 39. Handling errors for online change commands

IMSplex state after online change command error	User action to perform
One or more IMS systems in prepare complete state after a prepare failure.	Issue the **`TERMINATE OLC`** command. Correct the problem before reissuing the command.
All IMS systems in prepare complete state.	Issue the **`INIT OLC PHASE(COMMIT)`** or **`TERMINATE OLC`** command.
A mix of IMS systems in prepare complete and commit phase 1 complete state.	Correct the problem and try the commit again or issue the **`TERMINATE OLC`** command.
All IMS systems in commit phase 1 complete state before OLCSTAT data set updated.	Correct the problem and try the commit again or issue the **`TERMINATE OLC`** command.
All IMS systems in commit phase 1 complete state after OLCSTAT data set updated.	Correct the problem and try the commit again. The online change is committed and cannot be aborted.
Mix of IMS systems in commit phase 1 and commit phase 2 complete state.	Correct the problem and try the commit again.
All IMS systems in commit phase 2 complete state.	Correct the problem and try the commit again.
Mix of IMS systems in commit phase 2 complete state and not in online change state.	Correct the problem and try the commit again.

Global online change scenarios

The scenarios in this topic describe events that can occur during the processing of online change commands and explains how the IMS systems respond to the commands based on their current state.

Scenario 1: a successful global online change where all IMS systems use MODBLKS=OLC

This scenario shows the output returned to the TSO SPOC as the result of the successful global online change command sequence.

1. The **`QUERY OLC LIBRARY(OLCSTAT)`** command is issued to show the active online change libraries, ACBLIBA, FMTLIBA, and MODBLKSA. The command also shows that the current modify ID is 1 and the OLCSTAT data set name is IMSTESTL.IMS.OLCSTAT. The IMS systems that are current with the online change libraries include SYS3, IMS2, and IMS3. These IMS systems will be permitted to perform a warm start or emergency restart.
2. The **`INIT OLC PHASE(PREPARE) TYPE(MODBLKS) OPTION(FRCABND,FRCNRML)`** is issued to prepare IMS systems for a MODBLKS online change. IMS3 is the prepare master because it builds all of the output lines. Because the FRCABND and FRCNRML options are specified when the online change command is issued, any IMS systems that are down are permitted to perform the online change. IMS2, SYS3, and IMS3 are in the OLCSTAT data set, so they will participate in the online change.
3. The **`INIT OLC PHASE(COMMIT)`** is issued to commit the online change. IMS2, IMS3, and SYS3 all successfully complete the online change. The commit master, IMS3, reports the new online change libraries, MODBLKSB, ACBLIBA, and FMTLIBA, and the new modify ID of 2.
4. Another **`QRY OLC LIBRARY(OLCSTAT) SHOW(ALL)`** command is issued to show the new online change libraries and the new modify ID of 2.

The following sample output represents four responses to the four commands described in this scenario:

```
Response for: QRY OLC LIBRARY(OLCSTAT) SHOW(ALL)
MbrName  CC  Library  ACBLIB  FMTLIB  MODBLKS  Modid  DSName        LastOLC    MbrList
IMS3     0   OLCSTAT    A     A       A        01     IMSTESTL.IMS.OLCSTAT     SYS3,IMS2,IMS3

Response for: INIT OLC PHASE(PREPARE) TYPE(MODBLKS) OPTION(FRCABND,FRCNRML)
MbrName  Member     CC ACBLIB FMTLIB MODBLKS    Modid
IMS3     IMS2       0
IMS3     SYS3       0
IMS3     IMS3       0       A      A      A        1

Response for: INIT OLC PHASE(COMMIT)
MbrName  Member     CC ACBLIB FMTLIB MODBLKS    Modid
IMS3     IMS2       0
IMS3     SYS3       0
IMS3     IMS3       0      A      A       B         2

Response for: QRY OLC LIBRARY(OLCSTAT) SHOW(ALL)
MbrName  CC  Library  ACBLIB  FMTLIB  MODBLKS  Modid  DSName        LastOLC    MbrList
IMS3      0   OLCSTAT    A        A        B       02  IMSTESTL.IMS.OLCSTAT  MODBLKS  SYS3,IMS2,IMS3
```

Scenario 1a: a successful global online change where one IMS uses MODBLKS=DYN

The IMS that uses DRD does not participate in the global online change. You must use DRD on that IMS to change the resource that you are changing in the other IMS systems that participate in the global online change.

This scenario involves a global online change for an IMSplex that has four IMS systems with the following attributes:

- Three IMS systems that use the online change process for the resources in the MODBLKS data set. They have MODBLKS=OLC specified in the DYNAMIC_RESOURCES section of the DFSDFxxx IMS.PROCLIB member.
- One IMS that uses dynamic resource definition (DRD). It has MODBLKS=DYN specified in the DYNAMIC_RESOURCES section of the DFSDFxxx IMS.PROCLIB member.

The following list illustrates a successful global online change command sequence:

1. The **`QUERY OLC LIBRARY(OLCSTAT)`** command is issued to show the active online change libraries, ACBLIBA, FMTLIBA, and MODBLKSA. The command also shows that the current modify ID is 1 and the OLCSTAT data set name is IMSTESTL.IMS.OLCSTAT. The IMS systems that are current with the online change libraries include SYS3, IMS2, and IMS3. These IMS systems will be permitted to perform a warm start or emergency restart.
2. The **`INIT OLC PHASE(PREPARE) TYPE(MODBLKS) OPTION(FRCABND,FRCNRML)`** is issued to prepare IMS systems for a MODBLKS online change. IMS3 is the prepare master because it builds all of the output lines. Because the FRCABND and FRCNRML options are specified when the online change command is issued, any IMS systems that are down are permitted to perform the online change. IMS2, SYS3, and IMS3 are in the OLCSTAT data set, so they will participate in the online change.
3. The **`INIT OLC PHASE(COMMIT)`** is issued to commit the online change. IMS2, IMS3, and SYS3 all successfully complete the online change. The commit master, IMS3, reports the new online change libraries, MODBLKSB, ACBLIBA, and FMTLIBA, and the new modify ID of 2.
4. Another **`QRY OLC LIBRARY(OLCSTAT) SHOW(ALL)`** command is issued to show the new online change libraries and the new modify ID of 2.

   The following sample output represents 4 responses to the 4 commands described in this scenario:

```
Response for: QRY OLC LIBRARY(OLCSTAT) SHOW(ALL)
MbrName  CC  Library  ACBLIB  FMTLIB  MODBLKS  Modid  DSName        LastOLC    MbrList
IMS3     0   OLCSTAT    A     A       A        01     IMSTESTL.IMS.OLCSTAT     SYS3,IMS2,IMS3

Response for: INIT OLC PHASE(PREPARE) TYPE(MODBLKS) OPTION(FRCABND,FRCNRML)
MbrName  Member     CC ACBLIB FMTLIB MODBLKS    Modid
IMS3     IMS2       0
IMS3     SYS3       0
IMS3     IMS3       0       A      A      A        1
```

```
Response for: INIT OLC PHASE(COMMIT)
MbrName  Member     CC ACBLIB FMTLIB MODBLKS    Modid
IMS3     IMS2       0
IMS3     SYS3       0
IMS3     IMS3       0      A      A       B         2

Response for: QRY OLC LIBRARY(OLCSTAT) SHOW(ALL)
MbrName  CC  Library  ACBLIB  FMTLIB  MODBLKS  Modid  DSName        LastOLC    MbrList
IMS3      0   OLCSTAT    A        A        B       02  IMSTESTL.IMS.OLCSTAT  MODBLKS  SYS3,IMS2,IMS3
```

Scenario 2: online change prepare command is rejected because an IMS is down

In this online change scenario, an **INITIATE OLC PHASE(PREPARE) TYPE(ALL)** command was issued and routed to IMS A. The **INITIATE OLC PHASE(PREPARE)** command is rejected because two IMS systems are down (IMS B abends and IMS C shuts down normally), but no FRCABND or FRCNRML options were specified when the command was issued.

The following table identifies the command sequence and codes returned during this scenario:

Table 40. Scenario 2: command sequence and codes for failed online change prepare command

Action or event	IMS A completion code	IMS B completion code	IMS C completion code	Command return code	Command reason code
INIT OLC PHASE(PREPARE) TYPE(ALL)	2	B2	B2	C	3004
INIT OLC PHASE(PREPARE) TYPE(ALL) OPTION(FRCABND,FRCNRML)	0	B2	B2	4	100C
INIT OLC PHASE(COMMIT)	0	B2	B2	4	100C
INIT OLC PHASE(PREPARE) TYPE(ALL)	0			0	0
INIT OLC PHASE(COMMIT)	0			0	0

1. An **INITIATE OLC PHASE(PREPARE) TYPE(ALL)** command was issued. IMS A is the command master. IMS A detects that IMS B and IMS C are down and that no FRCABND or FRCNRML keywords were specified, so it rejects the command. IMS A returns a completion code of 2 (not done) for itself; IMS A also returns a completion code of B2 (IMS state error) for IMS B and IMS C. IMS A sets an overall return code of C and an overall reason code of 3004 (failed for all IMS systems).
2. Another **INITIATE OLC PHASE(PREPARE) TYPE(ALL)** command was issued, this time with FRCABND and FRCNRML. These options force the online change even though IMS systems are down. IMS A is again the command master. IMS A detects that IMS B and IMS C are down, but determines that this is acceptable. IMS A sets a completion code of B2 (IMS state error) for IMS B and IMS C. IMS B and IMS C do not perform the prepare phase, because they are down. IMS A performs the prepare phase locally and returns a completion code of 0 for itself. IMS A then sets an overall return code of 4 and a reason code of 100C (failed for some IMS systems for acceptable reasons).
3. An **INIT OLC PHASE(COMMIT)** command is issued; the commit succeeds. IMS A updates the OLCSTAT data set with the new information. IMS A includes a record for itself, but omits a record for IMS B and IMS C. IMS B and IMS C are no longer current with the online change libraries. IMS B and IMS C will have to cold start because the last online change included MODBLKS. If an IMS is down during a MODBLKS online change, it has to cold start.
4. Another **INITIATE OLC PHASE(PREPARE)** command is issued. This time, IMS B and IMS C do not participate because they are not current with the online change libraries. Only IMS A participates in the global online change.
5. An **INIT OLC PHASE(COMMIT)** command is issued. IMS B and IMS C do not participate because they are not current with the online change libraries. Only IMS A commits the global online change. IMS A sets a completion code of 0 for itself and an overall return code and reason code of 0.

Scenario 3: commit phase 1 fails due to work in progress

In this online change scenario, commit phase 1 fails due to work in progress. However, the **INITIATE OLC PHASE(COMMIT)** command eventually succeeds on all of the IMS systems, which results in overall return and reason codes of 0.

The following table identifies the command sequence and codes returned during this scenario:

Table 41. Scenario 3: command sequence and codes for failed commit phase 1

Action or event	IMS A completion code	IMS B completion code	IMS C completion code	Command return code	Command reason code
INIT OLC PHASE(PREPARE) TYPE(ALL)	0	0	0	0	0
INIT OLC PHASE(COMMIT)	B1	B1	4	C	3004
INIT OLC PHASE(COMMIT)	B1	4	4	C	3004
/DISPLAY MODIFY and other commands	none	none	none	none	none
INIT OLC PHASE(COMMIT)	0	0	0	0	0

1. The **INITIATE OLC PHASE(PREPARE)** command is issued and succeeds on all of the IMS systems. IMS A is the command master. Each IMS performs the prepare phase locally and sets a completion code of 0. IMS A sets an overall return code and reason codes of 0.
2. The **INITIATE OLC PHASE(COMMIT)** command is issued, and IMS A is the command master. IMS A attempts commit phase 1 locally, but the commit phase fails due to work in progress. IMS A sets a completion code of B1 (resource state) for itself. IMS A continues processing and sends the commit phase 1 to the other IMS systems that need to report any work in progress. IMS C successfully completes commit phase 1, but IMS B has work in progress. IMS A sets a completion code of 4 (incomplete) for IMS C and quits the commit process at this point due to errors.
3. Another **INITIATE OLC PHASE(COMMIT)** command is issued; IMS A is the master again. IMS A again attempts commit phase 1 locally, which fails due to work in progress. IMS A sends commit phase 1 to IMS B and IMS C. IMS B successfully completes commit phase 1. IMS C returns a completion code indicating that it is already in the correct state. IMS A quits the commit due to errors and sets the completion code to 4 (incomplete) for IMS B and IMS C. IMS A sets an overall return code of C and a reason code of 3004 (failed for all IMS systems).
4. **/DISPLAY MODIFY** and other commands are issued to resolve the work in progress that might cause the commit to fail.
5. A final **INITIATE OLC PHASE(COMMIT)** command is issued. All of the IMS systems successfully complete the commit phases 1, 2, and 3 and set completion codes of 0 for themselves. The overall return code and reason code is 0. IMS A sets an overall return code and reason code of 0.

Scenario 4: online change prepare command fails due to a timeout error

In this scenario, the online change prepare command fails due to a timeout error. Because one IMS never receives the prepare phase, the command times out while waiting for a response. Online change must be aborted after a prepare failure.

The following table identifies the command sequence and codes returned during this scenario:

Table 42. Scenario 4: command sequence and codes for failed prepare command due to timeout error

Action or event	IMS A completion code	IMS B completion code	IMS C completion code	Command return code	Command reason code
INIT OLC PHASE(PREPARE) TYPE(MODBLKS)	0	0	91	C	3000

Table 42. Scenario 4: command sequence and codes for failed prepare command due to timeout error (continued)

Action or event	IMS A completion code	IMS B completion code	IMS C completion code	Command return code	Command reason code
INIT OLC PHASE(PREPARE) TYPE(MODBLKS)				10	4110
INIT OLC PHASE(COMMIT)	2	2	B2	C	3004
TERMINATE OLC	0	0	3	4	100C
INIT OLC PHASE(PREPARE) TYPE(MODBLKS)	0	0	0	0	0
INIT OLC PHASE(COMMIT)	0	0	0	0	0

1. In this scenario, IMS A is the command master, and the **INITIATE OLC PHASE(PREPARE) TYPE(MODBLKS)** command is issued; IMS A is the command master. SCI goes down before IMS C receives the prepare phase. IMS A and IMS B successfully complete the prepare phase locally and set completion codes of 0 for themselves. The RM process step response times out, and RM returns the process step results to IMS A. IMS A sets a completion code of 91 (time out) for IMS C. IMS A rejects the prepare command because of the timeout error. After the failure, IMS A and IMS B remain in an online change prepare state, and IMS C is not in an online change state. IMS A sets an overall return code of C and reason code of 3000 (prepare worked for some of the IMS systems). Because IMS C does not do the prepare and a prepare failure occurred, the online change must be aborted.

   In this situation, you should issue **QUERY MEMBER TYPE(IMS)** to see the state of the IMS systems to determine what to do.
2. Another **INIT OLC PHASE(PREPARE) TYPE(MODBLKS)** command is issued. However, because online change must be aborted after a prepare failure, this prepare command also fails. IMS A, the command master, detects that it is already in a prepare state and rejects the command with a return code of 10 and a reason code of 4110. (The **INIT OLC** command is rejected because the command does not apply to the online change state of the command master.)
3. An **INIT OLC PHASE(COMMIT)** command is issued. The installation mistakenly tries to commit the online change after the prepare failure. IMS A is once again the command master. IMS A performs commit phase 1 locally, then sends commit phase 1 to IMS B and IMS C. IMS A and IMS B successfully perform commit phase 1 locally. IMS C rejects the commit phase 1 because it is not in an online change prepare state. IMS C returns a completion code of B2 (IMS state error). IMS A detects the error and sets the completion code to 2 (incomplete) for IMS A and IMS B. IMS A also returns a completion code of B2 (IMS state error). IMS A then terminates commit processing. The command fails with a return code of C and a reason code of 3004 (no IMS systems successfully completed command).
4. A **TERMINATE OLC** command is issued to abort the global online change. IMS A and IMS B successfully complete the abort. Because IMS C is not in an online change state, it returns a completion code of 3 (IMS already in requested online change state). IMS A, the command master, treats this as acceptable, and returns an overall return code of 4 and a reason code of 100C (successful, but not applicable to some IMS systems).
5. An **INIT OLC PHASE(PREPARE) TYPE(MODBLKS)** command is issued. The prepare succeeds.
6. An **INIT OLC PHASE(COMMIT)** command is issued. The commit succeeds.

Performing a cold start

Under certain circumstances—for example, if an IMS goes down during an online change—you might need to perform a cold start. The process you follow to perform a cold start varies depending on whether the IMS uses local or global online change.

For general information about cold start, see "Cold start" in *IMS Version 15.2 Operations and Automation*.

Local online change and cold start

Using local online change affects all types of IMS system definitions. IMS online subsystems place control blocks in the IMS.MODBLKS staging library. Before you perform a cold start, copy the MODBLKS library to either the IMS.MODBLKSA or the IMS.MODBLKSB data set.

If you copy these data sets to the inactive library, update the IMS.MODSTAT data set to change the IMS.MODBLKS data set DD name. If you copy to the active library, you cannot, in case of problems, return to the last system environment before the change.

Global online change and cold start

Attention: Typically, an IMS that was down during a global online change needs to be cold started when it comes back up. A cold start might be required to prevent restart from processing log records against the current online change libraries, potentially causing restart to fail or resulting in a severe error later.

Before the first IMS in the IMSplex cold starts the first time, you must initialize the OLCSTAT data set. To initialize the OLCSTAT, use the Global Online Change utility (DFSUOLC0).

Important: Use the Global Online Change utility with caution, so you do not inadvertently destroy valid OLCSTAT data set contents. If you do destroy valid OLCSTAT data set contents, global online change and initialization of additional IMS systems fail until the OLCSTAT data set is re-initialized.

You should establish an OLCSTAT data set recovery procedure to deal with the loss of the OLCSTAT data set. After every successful global online change, record the modify ID, the active online change library suffixes, and the list of IMS systems that are current with the online change libraries. If the OLCSTAT data set is destroyed, run the initialize function of the Global Online Change utility with the saved data to re-initialize the OLCSTAT data set.

Related reading: For more information about the Global Online Change utility, see *IMS Version 15.2 System Utilities*.

Determining when to perform a cold start

Perform a cold start when an IMS was down during the last global online change and its restart type conflicts with the last online change type or when an IMS was down for two or more global online changes.

Perform a warm start when an IMS was down during the last global online change and the restart does not conflict with the last global online change.

For more information about the types of restart commands permitted for specific online change types, see "Performing a cold start" on page 457.

Table 43. Restart commands permitted for specific online change types

Online change type	Restart commands permitted

ACBLIB	`/NRE CHECKPOINT 0`
	`/ERE COLDBASE`

ALL	`/NRE CHECKPOINT 0`

FORMAT	`/NRE CHECKPOINT`
	`/NRE`
	`/ERE`
	`/ERE COLDCOMM`
	`/ERE COLDBASE`

MODBLKS	`/NRE CHECKPOINT 0`

Cleaning up after a failed restart attempt

If an IMS is down during global online change and your restart command is rejected because the IMS requires a cold start, you might need to perform one or more cleanup steps to ensure a successful cold start.

To clean up after a failed restart attempt:

1. Issue a DBRC **`LIST.RECON`** command to check for an IMS subsystem record or any open OLDSs.
2. If a subsystem record exists, issue a DBRC **`DELETE.SUBSYS`** command to clean up the DBRC subsystem record for that IMS.
3. For are any open OLDSs, issue a DBRC **`GENJCL.CLOSE`** command.

   This command enables DBRC to read the RECON data set and determine which OLDS to close for that IMS.

Chapter 29. Printing output with IMS Spool API

This topic provides design and operational advice for the use of the IMS DL/I Spool application programming interface (API) and provides details for using the IMS Spool API to increase the access of IMS application programs to advanced printing capabilities.

IMS provides an expansion of the DL/I application programming interface that allows application programs to interface directly to JES and to create print data sets on the JES spool. These print data sets are then made available to print managers and spool servers to serve the needs of the application program.

Design and operational considerations

The following topic introduces IMS support of IMS Spool API.

Native IMS terminal support

IMS supports a set of terminals and printers that are associated with the IMS control region through the IMS systems generation process and ETO.

For most environments, a message that is destined for IMS-supported terminals is placed in the IMS message queue for intermediate storage. When the terminal can receive the message, IMS retrieves the message from the IMS message queue and sends the message to the device. If the VTAM session is lost during message transmission, or if a transmission error occurs, IMS places the message back on the IMS message queue for transmission at a later time. Because the IMS message queue is a recoverable resource, IMS can provide message integrity and recover from media failures associated with the IMS message queue.

The following figure shows a simplified diagram of the IMS environment. The figure shows terminal devices controlled by the IMS control region, the IMS message queue, and a message processing region. The DL/I pre-processing routines and the services of the IMS control region provide both terminal and database services. An application program sending output messages to a printer can identify the printer through the DL/I CHNG call and insert messages to the printer through the DL/I ISRT call. IMS places these messages in the IMS message queue until they are committed by the application program. After the calls are committed, IMS delivers the messages to the designated printer.

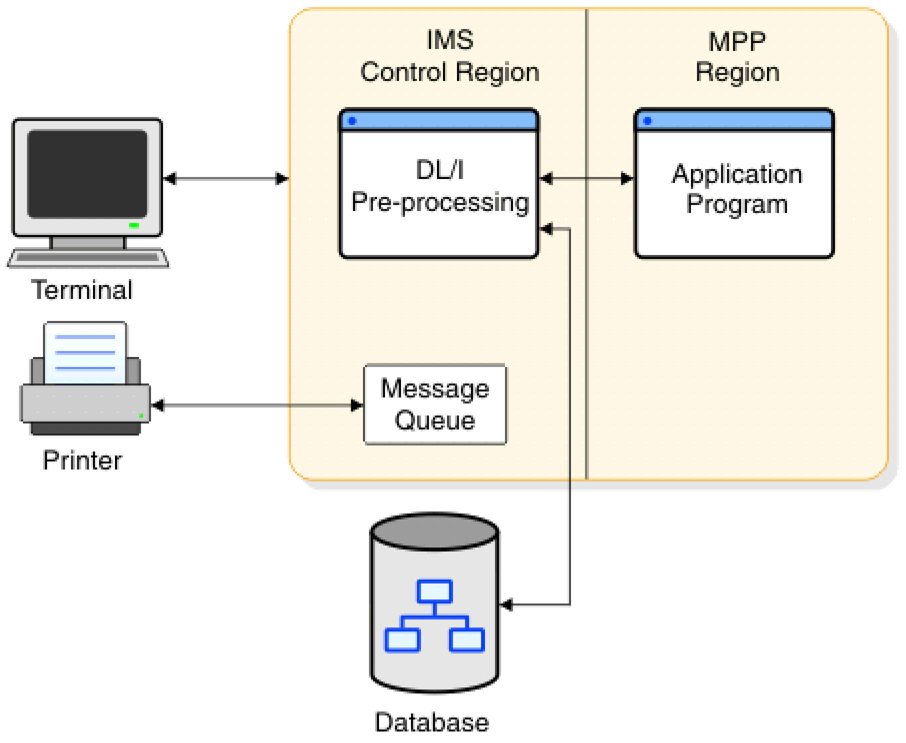


Figure 54. IMS environment without Spool API

Application requirements

IMS application programs can send messages to IMS native printers and to printers that are capable of printing high-quality text, special fonts, bars, and graphs. Application programs that create data streams for these printers and send data to these printers use the job application subsystem IMS Spool API. The IMS Spool API passes the print data sets to IBM's Advanced Function Printing (AFP) services or to an OEM print server based on data set class, remote destination, size (segment sizes of up to 32 KB are common), or other characteristics.

Applications creating JES print data sets from IMS application regions require reasonable performance, a simple interface, and flexibility in program usage. The IMS Spool API interface extensions attempt to provide for these needs. The DL/I application interface supports the creation of JES print data sets. The techniques for creating these JES print data sets are similar to the techniques used for sending messages to native IMS printers.

IMS uses several z/OS services to support IMS Spool API. These z/OS services introduce performance and availability considerations that IMS and its application programs have not had to deal with previously, ranging from additional processor requirements for data set options parsing to lack of sync point support by the JES subsystem. The IMS Spool API environment, shown in the following figure, is designed to address these issues.

Because the print data sets can be very large and because using the message queue as intermediate storage adds processing overhead without providing additional message integrity, most IMS Spool API execution is performed in the IMS dependent region. However, this asynchronous message technique also presents problems in relating application program errors that result in message delivery problems.

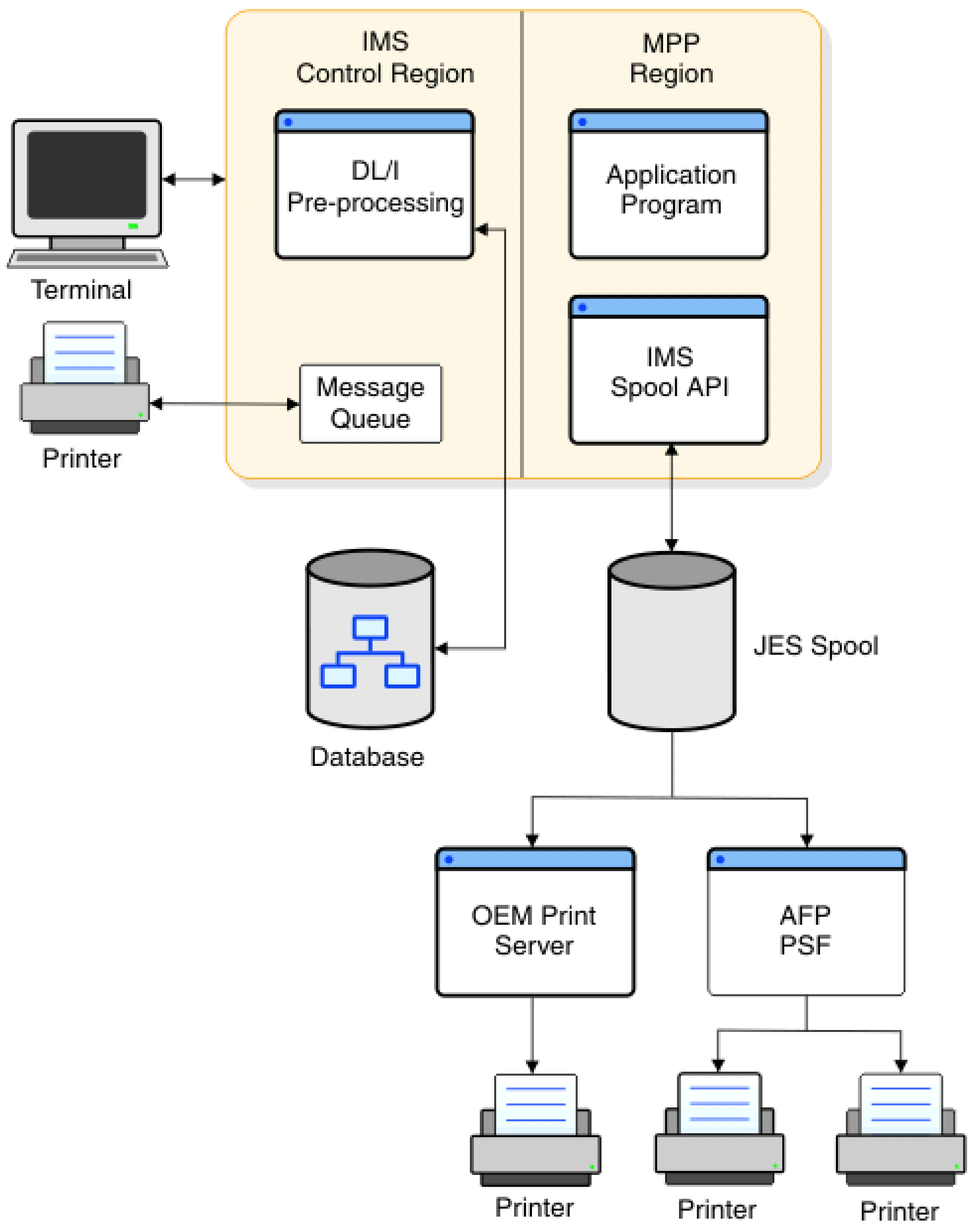


Figure 55. IMS environment with Spool API

Note: Do not confuse the support for creating JES print data sets on the IMS Spool API with the IMS terminal support for spool lines defined as UNITYPE=SPOOL in the IMS systems generation process. Terminal support for spool lines is completely unrelated to support for JES print data sets.

The IMS Spool API as a data manager

The IMS Spool API controls the input, execution, and output of jobs on the z/OS system. Because the IMS Spool API is at the JOB or time sharing (TSO) user boundary, JES creates special considerations for IMS environments where each print data set is a logical message and is not part of or associated with a job boundary.

A data manager associated with IMS provides the DL/I application programming interface, which provides temporary buffering of data with the sync point process. The sync point process commits the changes for a unit of work if the unit of work terminates normally, or backs out the changes if the unit of work terminates abnormally. The IMS Spool API does not use the sync point process, but it can back out the effect of changes made by an abending unit of work.

The IMS Spool API does not recover data as does the IMS Transaction Manager or the IMS Database Manager. Because the IMS Spool API does not log changes to the spool data set, data on the spool can be lost in a restart if the spool device finds data errors.

Print data set characteristics

You can create a JES print data set by allocating a print data set by the SYSOUT= specification on a JCL statement or dynamically by allocating a print data set using the MVS dynamic allocation support for SVC 99.

When a print data set is allocated, the characteristics of the print data set can be defined or allowed to default.

Some examples of these print data set characteristics are FORMS, COPIES, DESTINATION, and WRITER. You can specify some of these output characteristics with the DD card, or you can associate the print data set with a set of print data set characteristics defined by an OUTPUT JCL statement. JES provides a way to dynamically define these print data set descriptors using system services known as dynamic output or SVC 99 processing. When used with dynamic allocation, dynamic output processing can associate system output or SYSOUT data sets with a set of dynamically built OUTPUT descriptors.

You can build print data set descriptors for IMS Spool API in three ways, each of which is discussed in the following topics.

Related reading: For more information on z/OS services for dynamic allocation for print data sets, see *z/OS MVS Programming: Authorized Assembler Services Guide*.

The Change (CHNG) call

In most application programs, you can provide the print data set descriptors using the change (CHNG) call interface. The IMS Spool API uses this existing technology.

The CHNG call supports two optional parameters for IMS Spool API:

- The options list parameter points to a character string containing a list of options. The IAFP keyword identifies the CHNG call as a request for IMS Spool API services.
- The feedback area parameter points to an area through which IMS can return error information to the application program when mistakes are found in the options list.

If your application creates multiple data sets with the same print options, or is wait-for-input (WFI), use the SETO call to reduce the parsing impact.

The CHNG call provides the data set characteristics in one of the following ways:

- The call can provide the data set options directly by using the PRTO option.

  When the PRTO option is used with the CHNG call, IMS activates z/OS services (SJF) to verify the print options and calls z/OS services for dynamic output to create the output descriptors that are used when the print data set is allocated. This is the simplest way for the application program to provide print data set characteristics. Using the PRTO option involves the most overhead, because parsing must occur for each CHNG call with the PRTO option.

  Use the PRTO option when you cannot significantly improve application performance by using pre-built text units across multiple CHNG calls.
- The call can reference a set of pre-built text units for dynamic descriptors using the TXTU option.

  Application programs that can reuse text units can achieve better performance by using the TXTU option. If the application program can manage the text units necessary for dynamic output, use the TXTU option to avoid parsing for many of the print data sets.

  The application program builds the text unit in the necessary format within the application area and passes these text units to IMS.

  The application program can provide the print options to IMS with a SETO call and provide a work area for the construction of the text units. Using the TXTU option, IMS can perform parsing and text unit construction so that the work area passed contains the text units necessary for dynamic output after a successful SETO call. Do not relocate this work area because it contains address-sensitive information.

- The call can refer to an OUTPUT JCL statement in the dependent region's JCL by using the OUTN option.

  This is a simple way of providing print data set information if it can be used by the application program. The output JCL statements are referenced by the OUTN option in the options list on the CHNG call. When the OUTN option is used, IMS references the output JCL statement at dynamic allocation so that the IMS Spool API obtains the print data set characteristics from the output JCL statement.

The Set Options (SETO) call

You can construct a print data set descriptor with the set options (SETO) call. The SETO call is especially useful to application programs with wait-for-input (WFI) execution. The SETO call reduces the overhead necessary to perform parsing and text unit construction of the output descriptors for a data set.

If the application program needs to reuse a set of descriptors during the scheduling of the program, the application program provides the print data set characteristics to the IMS Spool API through the SETO call. The SETO call parses the output options and builds the dynamic output text units in the work area provided by the application program. Once the application program is supplied the pre-built text units, these text units are used with a CHNG call through the TXTU parameter. They provide the print characteristics for the data set without incurring the overhead of parsing and text unit build.

It is not necessary to use the SETO option to pre-build the text units if they can be pre-built using some other programming technique.

The Output DD statement

You can provide print data set descriptors by adding output JCL statements to the dependent region JCL. The application program passes the name of the output JCL statement in the dependent region JCL using a CHNG call with the OUTN parameter. IMS can use the output JCL statement when the print data set is dynamically allocated.

If the output statement does not exist in the dependent region's JCL, dynamic allocation fails when the first insert is done. The application program receives a status code AX indicating the insert (ISRT) failed.

Writing data to the IMS Spool API

You can write data to the IMS Spool API using the insert (ISRT) call or the purge (PURG) call.

You can backout print data sets using the ROLL call or the ROLB call.

Do not use the SETS call or the ROLS call to backout print data sets. These calls are not supported by the IMS Spool API.

The Insert call

The standard insert (ISRT) call writes the data to the IMS Spool API using BSAM. Because the length of the data written to the spool can be up to 32760 bytes, the BSAM write is performed directly from the application program's buffer area.

If IMS finds that the user's I/O area is above the 16 MB line, IMS moves the user's application program data to a work area below the line before performing the BSAM write.

If the application area is already below the line, the write can be done directly from the I/O area. If possible, the I/O area should be below the 16 MB line. However, you do not need to take unusual steps to place the I/O area below the line unless performance shows that such action is necessary. By writing the application program's buffer without first moving it, IMS prevents problems associated with moving large blocks of data. When BSAM is used, change the format of the I/O area so that it contains the BSAM block descriptor word (BDW).

Related reading: For more information on the format of the I/O area, see *IMS Version 15.2 Application Programming*.

The PURG call

Use the purge (PURG) call with an express alternate PCB to release a print data set to the IMS Spool API for immediate printing. When the PURG call is issued, the print data set is closed and deallocated. If the PURG call is issued against an alternate PCB that is not marked as EXPRESS=YES, no action is taken for the print data set.

You can issue the PURG call either with or without an I/O area:

- When the PURG call is issued with an I/O area, it is treated as though it were two calls. The first function handles the PURG request, and the second function inserts the data provided by the I/O area.
- When the PURG call is issued against an alternate PCB that is not marked as EXPRESS=YES, the purge function is ignored. If an I/O area is included, the data is placed in the print data set.

The PURG call differs slightly with IMS Spool API when it is used with alternate PCBs that are generated as EXPRESS=YES:

- The current print data set is closed, deallocated, and sent to JES for printing.
- Status code of A3 is returned to the application program, indicating that the alternate PCB no longer contains a valid destination.

For information on the results of the PURG call, see the following table.

Table 44. Results of PURG call

Type of PURG call	Express PCB	Non-express PCB
With I/O area	The print data set is released for printing by JES. PURG call receives status code of A3.	The PURG call functions the same as ISRT call.
Without I/O area	The print data set is released for printing by JES. PURG call receives status code of `blanks`.	No action is taken.

ROLL and ROLB calls

The ROLL and ROLB calls can be issued by an application program using the IMS Spool API functions. These calls result in backout (data set deletion) of any print data sets that have not been released to the IMS Spool API because of a PURG call to an express alternate PCB.

SETS, SETU, and ROLS calls

The IMS Spool API does not support SETS, SETU, or ROLS calls. If you issue them, no action is taken by the IMS Spool API, and no special status codes are returned to show that the SETS, SETU, or ROLS call was issued by an application program.

No restrictions exist on the use of SETS, SETU, and ROLS calls, because these calls can be used by the application program outside the processing of print data sets.

Special considerations—descriptors allowed

The output descriptors allowed by the IMS Spool API are those supported by the **TSO OUTDES** command of the z/OS system under which IMS is executing at the time that parameter validation is performed.

Related reading: For more information on output descriptors, see *z/OS TSO/E Command Reference*.

The initialization parameters you use with the IMS Spool API can have an impact on the parameters that you use to define print data set characteristics.

Controlling print data sets

Application programs have some control over the IMS Spool API files IMS creates for them. This control applies to the data streams generated by the application program as well, because IMS is not sensitive to this data.

You can embed control information within the data stream itself, or you can specify the IMS Spool API file options in an application program DL/I CHNG call. These processing options are specified on a data set basis and are specified on the TSO OUTDES statement.

The IMS Spool API routes print data based on output scheduling. Generally, scheduling is determined by output class (CLASS parameter), printer destination (DEST parameter), FORM, and PRMODE.

IMS is a transparent delivery service. IMS dynamically creates IMS Spool API files based on data set processing options, writes the messages to them, and then deallocates them so that they can be processed by an IMS Spool API data set server (such as PSF).

To ensure proper disposition of print data sets, the z/OS operator might need to become involved under certain conditions, such as:

- IMS emergency restart
- Dependent region abnormal termination
- Dynamic deallocation failure for a print data set

Under such conditions, the z/OS operator must deal appropriately with the IMS Spool API data sets that represent in-doubt messages. IMS provides informational messages to help the z/OS operator if the application program requests the proper disposition of these data sets.

Review operational procedures for unprintable data sets at your installation. Many installations have jobs that delete print data sets after a given number of days. Make sure that these jobs do not delete IMS print data sets that should be printed.

Express alternate PCBs

The IMS Spool API supports the use of express alternate PCBs. If the application program issues a purge (PURG) call against an alternate PCB that has been generated with EXPRESS=YES, the print data set is closed and deallocated. It is available for JES processing before the termination of the IMS unit of work.

Related reading: For more information on using the PURG call with an express alternate PCB, see "The PURG call" on page 466.

XRF environments

The IMS Spool API operates within XRF systems with no special support. During an XRF takeover, the IMS Spool API is affected as if a failure and normal restart occurred. If partial print data sets exist when the takeover occurs, and IMS abnormal termination processing does not execute on the primary system, the partial print data sets can be made available to IMS Spool API for printing.

Related reading: For a description of the proper procedure for these partial print data sets, see *IMS Version 15.2 Application Programming*.

Understanding allocation errors

The IMS Spool API interface dynamically allocates the print data set only after data is actually inserted to the data set. This reduces overhead and simplifies cleanup for abending transactions.

Occasionally, errors can occur during dynamic allocation that are the results of incorrect data set print options supplied with the CHNG or SETO call. The data set print options can be parsed during the processing of the CHNG and SETO calls. Destinations can only be validated during dynamic allocation.

If any of the print options are incorrect and dynamic allocation fails when the first insert is done for the data set, the ISRT call receives a status code of AX. The IMS Spool API code provides the following to the application program:

- A status code
- An error message DFS0013E
- A diagnostic log record (67D0)

The error message shows the type of service that is activated and the return and reason codes that are responsible for the error. For example, a common failure is indicated by reason code 046C: Remote work station not defined to job entry subsystem. You see this reason code if you select an invalid destination or you select integrity option 2 (non-selectable destination) when the destination of IMSTEMP has not been defined to JES. Specifying an invalid destination in the destination name parameter of the call results in a dynamic deallocation error when IMS deallocates the print data set.

Some of the services indicated by the error message include:

DYN
z/OS dynamic allocation (SVC99)

OPN
z/OS data set open

OUT
z/OS dynamic output descriptors build (SVC109)

UNA
z/OS dynamic deallocation (SVC99)

WRT
z/OS BSAM write

Related reading: If the service is for dynamic allocation or deallocation, or for dynamic output descriptor build, see *z/OS MVS Programming: Authorized Assembler Services Guide*.

Chapter 30. Accessing external systems from within IMS

IMS application programs can access external systems to use data that resides in those external system or to interact with business logic that resides outside of the IMS environment.

IMS application programs that want to use external subsystem data can use either or both of the following facilities:

- The External Subsystem Attach Facility (ESAF), which can be used by JMP, JBP, MPP, BMP, and IFP application programs, can access external subsystem data, such as Db2 for z/OS databases or IBM MQ
- The DB2 Recoverable Resource Manager Services Attach Facility (DB2RRMS), which can be used by JMP and JBP applications that access Db2 for z/OS databases

You can specify both ESAF and DB2RRMS access in one SSM PROCLIB member.

IMS application programs that want to interact with business logic that resides outside of the IMS environment can do so using the IMS callout function. The callout requests can be either synchronous or asynchronous. The target of the callout request can be:

- Any generic web service, after going through IMS Enterprise Suite SOAP Gateway.
- A message-driven bean, an Enterprise JavaBeans component, a Java EE application, or web service, after going through IMS TM Resource Adapter.
- A user-written IMS Connect TCP/IP application or a vendor-supplied solution that uses TCP/IP and the IMS Connect protocol to retrieve callout requests.

Related reading:

- For more information about installing and defining external subsystems to IMS, see:
  - "Accessing multiple external subsystems" in *IMS Version 15.2 Communications and Connections*.
  - "Specifying the SSM= EXEC parameter" in *IMS Version 15.2 System Definition*.
- For more information about the software requirements of the callout function, see *IMS Version 15.2 Release Planning*. For more information about how application programs can use the callout function, see *IMS Version 15.2 Application Programming*.

Part 3. DBRC administration

This topic provides an overview of the IMS Database Recovery Control facility (DBRC) and a discussion of the interactions between IMS and DBRC.

Chapter 31. Overview of DBRC

A feature of the IMS Database Manager that facilitates easier recovery of IMS databases, DBRC maintains information that is required for database recoveries, generates recovery control statements, verifies recovery input, maintains a separate change log for database data sets, and supports sharing of IMS databases and areas by multiple IMS subsystems.

DBRC is an integral part of IMS. IMS relies on DBRC to record and manage information about many items. DBRC keeps this information in a set of VSAM data sets that are collectively called the recovery control (RECON) data set and advises IMS (based on the information in the RECON data set) about how to proceed.

Specifically, DBRC:

- Helps you ensure IMS system and database integrity by recording and managing information associated with the logging process.
- Assists IMS in the restart process by notifying IMS which logs to use for restart.
- Assists IMS to allow or prevent access to databases in data-sharing environments by recording and managing database authorization information.
- Facilitates database and log recovery by:
  - Controlling the use and integrity of the information in the logs.
  - Recording and maintaining information about the databases and logs in the RECON data set.
  - Generating and verifying the Job Control Language (JCL) for various IMS utility programs.
- Supports Extended Recovery Facility (XRF) by identifying (in the RECON data set) if the subsystem is XRF capable.
- Supports IMSplexes by:
  - Notifying all DBRCs in the same IMSplex within the same DBRC sharing group when one of the DBRCs performs a RECON data set reconfiguration.
  - Allowing parallel access to the RECON data set.

DBRC also plays a key role in managing the log data needed to restart and recover IMS online subsystems.

Most IMS configurations require DBRC, including:

- Online configurations: DB/DC, DCCTL, or DBCTL
- Data-sharing environments, including IMSplexes
- Configurations that use Extended Recovery Facility (XRF)

Attention: DBRC is not required for IMS batch jobs and for some offline utilities. However, if batch jobs and utilities that access registered databases are allowed to run without DBRC, the recoverability and integrity of the databases could be lost. Even if your configuration does not require the use of DBRC (such as in a non-data sharing, batch environment), you can simplify your recovery process by using DBRC to supervise recovery and protect your databases.

Related reading:

- *IMS Version 15.2 Operations and Automation* provides detailed descriptions of recovery procedures with and without DBRC.
- Chapter 34, "How DBRC helps in recovery," on page 483 further describes how DBRC helps with database recovery.

Related concepts

"Database Recovery Control facility" on page 7

The Database Recovery Control facility (DBRC) helps you control log and database recovery. It also controls the data sharing environment by allowing or preventing access to databases by the IMS systems that share those databases.

"Changing OLDS characteristics" on page 98
Before you scratch and reallocate an OLDS that has been archived, you must delete the log control record in the DBRC RECON data set for this OLDS by using the **`DBRC DELETE.LOG OLDS(`***ddname***`) SSID (`***name***`)`** command.

"Database Recovery Control and recovery" on page 101
The IMS Database Recovery Control (DBRC) facility makes it easier for you to recover IMS databases by extending the capabilities of IMS utilities for database recovery. DBRC can help recover both DL/I databases and Fast Path data entry databases (DEDBs).

DBRC components

DBRC includes the RECON data sets, the Database Recovery Control utility (DSPURX00), and skeletal JCL.

RECON data sets

DBRC stores recovery-related information in the RECON data sets.

Recommendation: Define three VSAM KSDSs for the RECON data sets when you install DBRC. The first two data sets are active data sets; the third one is a spare. The second active data set is a copy of the first. For most purposes, you can think of the two active RECON data sets as if they were a single data set, the RECON data set.

Related reading: These data sets are described in detail in Chapter 44, "Initializing and maintaining the RECON data sets," on page 531.

Database Recovery Control utility (DSPURX00)

Use the Database Recovery Control utility to issue DBRC commands.

The DBRC commands allow you to perform all of the following tasks:

- List the information in the RECON data set
- Update the information in the RECON data set
- Use the information in the RECON data set to generate jobs for the IMS utilities

Skeletal JCL

DBRC uses partitioned data set (PDS) members as input models (or templates) for generating input for some of the recovery utilities. These PDS members are distributed with IMS and are called skeletal JCL.

DBRC uses the skeletal JCL, information from the RECON data set, and instructions from a **`GENJCL`** command to generate the JCL and control statements that are needed to correctly run some of the recovery utilities. Modify the skeletal JCL to reflect your installation's system configuration.

Related reading: For more information about skeletal JCL, see *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*.

IMS provides PDS members that contain skeletal JCL statements. These PDS members are called skeletal JCL execution members and can be found in the IMS.SDFSISRC target library.

The IMS Installation Verification Program (IVP), customizes the skeletal JCL execution members and places the customized members into the IMS PROCLIB library. DBRC uses the IMS PROCLIB members to generate jobs (JCL and control statements) for the IMS utilities listed in *IMS Version 15.2 Database Utilities*. There is also a skeletal JCL execution member, JOBJCL, that produces a JOB statement.

Related reference
Database Recovery Control utility (DSPURX00) (System Utilities)

DBRC tasks

There are several tasks that are automatically performed by DBRC and a few additional ones that you initiate with a command or request to DBRC.

DBRC automatically performs the following tasks:

- Records and manages information about logs for IMS
- Records recovery information in the RECON data set
- Verifies that database utilities have the correct input
- Controls the recovery of databases that are registered with DBRC
- Controls the access authorization information for the control and serialization of shared databases

You can perform the following tasks and more when you initiate them by passing commands or requests to DBRC:

- Start or stop the DBRC application programming interface (API)
- Record recovery information in the RECON data set
- Generate JCL for various IMS utilities and generate user-defined output
- List general information in the RECON data set
- Gather specific information from the RECON data set

Related reading:

- See Chapter 32, "Recording and controlling log information," on page 477 for additional information about DBRC's logging support.
- See Chapter 36, "Supporting data sharing," on page 489 for additional information about DBRC's data sharing support.
- See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for detailed information about the DBRC commands.
- See *IMS Version 15.2 System Programming APIs* for detailed information about the DBRC API requests.

DBRC commands and API requests

Use DBRC commands or DBRC application programming interface (API) requests to obtain services from DBRC.

DBRC commands

Use DBRC commands to generate a listing of the information in the RECON data set; add to, change, and delete information in the RECON data set, and generate the JCL and the control statements necessary to run the IMS utilities that are used in database recovery.

The following is a list of the DBRC batch commands, along with information about how they are used:

- Use the **BACKUP.RECON** command to create a backup copy of the RECON data set.
- Use the **CLEANUP.RECON** command to delete old or expired recovery-related information and log information from the RECON data set.
- Use the **CHANGE** commands to modify information in the RECON data set.
- Use the **DELETE** commands to delete information from the RECON data set.
- Use the **GENJCL** commands to generate jobs for the various IMS recovery utilities.
- Use the **INIT** commands to make the following changes to a RECON data set:
  - Create DB groups
  - Create DBDS groups
  - Create recovery groups

  - Register databases
  - Create image copy records that are available for use during subsequent runs of the supported image copy utilities
  - Create change accumulation data sets that are available for future use by the Change Accumulation utility
  - Create entries that defines an area data set (ADS) that belongs to a Fast Path area
  - Create CA groups
  - Register DBDSs or DEDB areas
  - Register HALDB partitions
  - Initialize a RECON data set
- Use the **LIST** commands to produce a formatted printout of all or selected parts of the RECON data set.
- Use the **NOTIFY** commands to add to the RECON data set information that is normally written there automatically.

You can issue DBRC commands by embedding them in the JCL for running the Database Recovery Control utility (DSPURX00). This utility can process these commands while running either in a batch environment or as a TSO foreground program.

You can also issue the DBRC commands (with the exception of the LIST commands) as a parameter of the COMMAND API request (see *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*).

Certain DBRC commands can be issued from an online IMS environment (**/RM***xxxxxx* commands). The online DBRC commands are:

- **/RMCHANGE**
- **/RMDELETE**
- **/RMGENJCL**
- **/RMINIT**
- **/RMLIST**
- **/RMNOTIFY**

DBRC API requests

Use the DBRC API requests to obtain services from DBRC.

Related reading: For detailed information about the DBRC application programming interface and the DBRC API requests, see *IMS Version 15.2 System Programming APIs*.

Related reference
DBRC commands (Commands)

Chapter 32. Recording and controlling log information

As IMS operates, it constantly records its activities in the IMS logs. After you have initialized DBRC, it participates in the IMS logging process by recording and controlling information about logging activities in the RECON data set.

The IMS logs contain information about:

- IMS startups and shutdowns
- Changes made to databases
- Transaction requests received and responses sent
- Application program initializations and terminations
- Application program checkpoints
- IMS system checkpoints

The IMS logs consist of the:

- Write-ahead data set (WADS)
- Online log data set (OLDS)
- System log data set (SLDS)
- Recovery log data set (RLDS)

The following figure shows the log data sets and restart data set (RDS) that IMS produces and the RECON data set that DBRC creates and maintains.

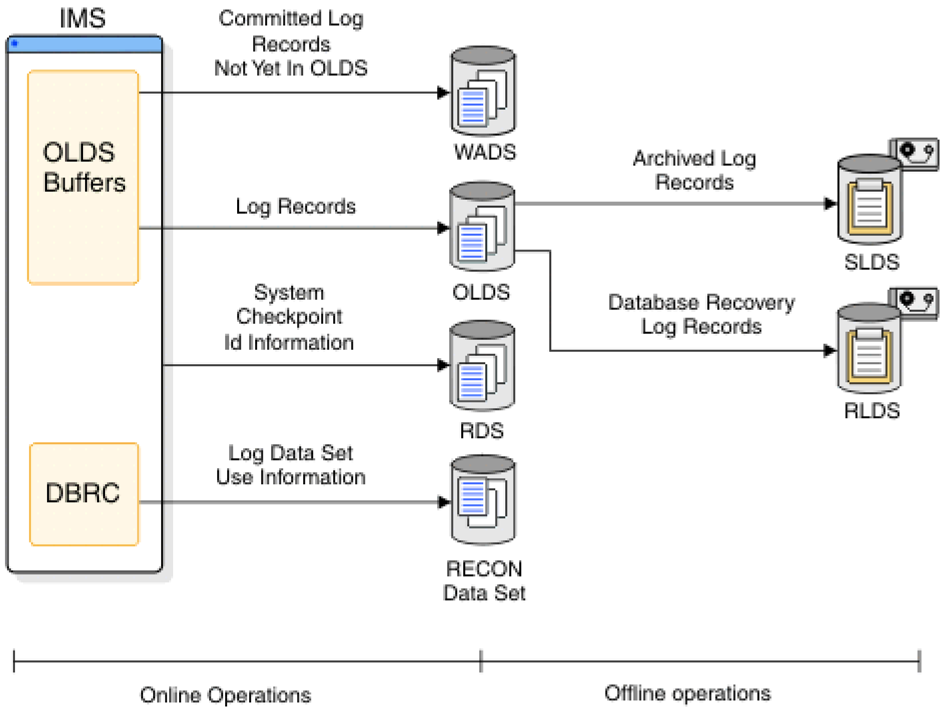

IMS
OLDS Buffers
DBRC
Committed Log Records Not Yet In OLDS
Log Records
System Checkpoint Id Information
Log Data Set Use Information
WADS
OLDS
RDS
RECON Data Set
Archived Log Records
Database Recovery Log Records
SLDS
RLDS
Online Operations
Offline operations

Figure 56. Logs produced for recovery and restart

Related reading:

- "Logging" on page 84 for detailed discussions of the IMS logging process.
- "DBRC log-related commands" on page 479

After you have initialized DBRC, it participates in the IMS logging process by recording and controlling information about logging activities. This information is recorded in the RECON data set. If you want DBRC to control the recovery of your database data sets (DBDSs), you must register them with DBRC.

DBRC automatically records many items in the RECON data set, including:

- Information about log data sets
- Information about database data sets
- Information about events, such as:
  - Database updates
  - Database authorizations
  - Creation of database image copies
  - Reorganizations of databases
  - Recoveries of databases
  - Archiving of an OLDS and creation of the corresponding SLDS and RLDS
  - Execution of the Log Recovery utility
  - Subsystem sign-on

IMS uses this information for restarting itself and for database recovery jobs (if the databases are registered with DBRC). DBRC also tracks the archiving requirements of the OLDS and, if requested, generates and submits the JCL for archiving jobs.

DBRC also provides unit-of-recovery management for all attached subsystems. DBRC provides information about these units of recovery for batch backout, dynamic backout, partial backout, and restart.

For logs produced by batch systems, you are not required to use DBRC. The advantage of using DBRC for batch jobs is that DBRC records information about all the log data sets that are produced by batch jobs and prevents batch update jobs from executing if you specify a dummy or null log data set.

Attention: If registered databases are updated without DBRC control, DBRC cannot correctly control recovery for the databases and database integrity can be jeopardized.

Changing log records in the RECON data sets

Use the **CHANGE.PRILOG** or **CHANGE.SECLOG** commands to modify the information about OLDSs, RLDSs, or SLDSs in the log record of the RECON data set.

Update the log record to indicate when errors have occurred on the data sets or that volume serial numbers have changed. You do not normally need to use these commands.

Archiving an OLDS

Run the Log Archive utility (DFSUARC0) to archive an OLDS to an SLDS so that IMS can reuse the OLDS. How frequently you should archive depends on the load on the subsystem and the number of log records written to the OLDSs.

The Log Archive utility always produces an SLDS. The SLDS contains all log records that are required for both database recovery and for online IMS restart.

You can ask the Log Archive utility to produce an RLDS in addition to an SLDS. The RLDS contains only those log records that are required for database recovery.

If you request an RLDS, information about the output RLDS data sets is recorded in the PRILOG record in the RECON data set and information about the SLDS data sets is recorded in the PRISLD record. If you do not request an RLDS, the same information about the SLDS data sets is recorded in both the PRILOG and PRISLD records.

If there is a secondary OLDS, or if you request that dual logs be produced from a single OLDS, the information about the secondary-log output is recorded in corresponding SECLOG and SECSLD records.

Important: Log data sets that are output from IMS batch jobs are technically SLDSs, but the information about them is recorded in the PRILOG and SECLOG records.

Run the Log Archive utility by issuing the **GENJCL.ARCHIVE** command. DBRC then determines which OLDSs are full, and generates the appropriate JCL.

Related reading: See member DFSDFxxx in *IMS Version 15.2 System Definition* for more information on the ARCHDEF statement and automatic archiving.

Recommendation: Whether you use automatic archiving or invoke archiving yourself, make sure the archive jobs run as quickly as possible. The online subsystem only reuses an OLDS after it has been archived. If the archive job is not run and all the OLDS become full, the online subsystem waits. One way to ensure that archive jobs run quickly is to use an initiator that runs with a fairly high priority and is not used by many other users. This ensures that the archive jobs do not remain on the internal reader queue for too long.

If DBRC has marked an OLDS in the RECON data set as having errors, the GENJCL function does not submit it for archiving. If one of a pair of OLDSs has been destroyed or is unavailable, you can choose to mark it in the RECON data set as having errors.

The following references point to where you can find more information about archiving log records.

Related reading:

- See *IMS Version 15.2 Operations and Automation* for more information about automatic, manual and custom archiving of log records.
- See *IMS Version 15.2 System Utilities* for more information about specifying entry points and running the Log Archive utility.
- Refer to *IMS Version 15.2 Exit Routines* for more information about the Log Archive and the Logger exit routines.

DBRC log-related commands

You an use commands to perform DBRC log-related functions in your operational procedures.

The following commands perform the DBRC log-related functions:

- **CHANGE.PRILOG**
- **CHANGE.RECON**
- **CHANGE.SECLOG**
- **DELETE.LOG**
- **GENJCL.ARCHIVE**
- **GENJCL.CLOSE**
- **LIST.LOG**
- **NOTIFY.PRILOG**
- **NOTIFY.SECLOG**

In addition to the **LIST.LOG** command, you can use a Log or OLDS query API request to retrieve log-related information from the RECON data set. You can also use the COMMAND API request to add to, change, and delete information in the RECON data set.

Related reading: See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for more information about DBRC commands.

Chapter 33. Recovering databases

The recovery process for IMS databases can include these three basic steps, although the details of the process can vary with the type of database to be recovered.

1. Restore the database to the most current image copy.

   ⚠ **Attention:** Skip this step if an HALDB Online Reorganization is being used because all the database changes are on log data sets.

2. Use the log data sets (or change accumulation data sets) to restore changes made to the database since the image copy was made.

   Definition: Change accumulation is the process of creating a compacted version of one or more IMS log data sets by eliminating records not related to recovery, and by merging multiple changes to a single segment into a single change. For more information about change accumulation, see "Log record change accumulation" on page 514.

3. Back out any incomplete changes.

When the recovery completes successfully, DBRC records information about the recovery in the RECON data set. If you performed a time-stamp recovery, DBRC records information about the range of omitted changes.

Requirement: If a time-stamp recovery is performed within a database allocation interval, you must immediately create an image copy to ensure a valid starting point for subsequent recovery attempts. DBRC then prevents the changes from being reapplied on any subsequent recovery.

The following figure illustrates a simple database recovery.

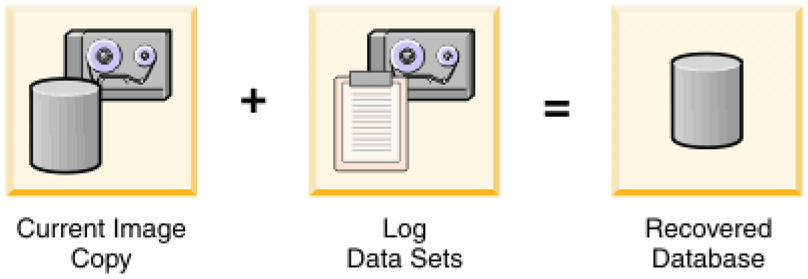


Figure 57. IMS database recovery process

Information for a database recovery can come from any or all of the following sources:

- Image copies of the database
- Database reorganization data sets
- Log data sets (SLDSs and RLDSs)
- Change accumulation data sets

You can use DBRC to track all of these information sources, greatly simplifying the task of database recovery.

Related reading: Refer to "Understanding the recovery task" on page 81 for more information about the recovery process.

If you register recoverable databases in the RECON data set, DBRC records the association of the databases to the log data sets containing database change records.

DBRC also records information about:

- Database image copies
- Reorganizations (except DEDB online reorganizations)

- Recoveries
- Change accumulations
- Backout

DBRC can generate JCL for executing a database recovery, because DBRC records this information in the RECON data set. Whether you use the **GENJCL** commands to generate JCL or provide the JCL yourself, DBRC uses information in the RECON data set to determine exactly which data sets are required for input. The Database Recovery utility runs only if DBRC verifies that the JCL is correct.

You can omit all logged changes after a certain time from the input by performing a *time-stamp recovery*. A time-stamp recovery is equivalent to backing out the omitted changes from the database.

Most time-stamp recoveries require DBRC in order to be successful. When you involve DBRC in your request for a time-stamp recovery, DBRC selects the correct logs and, at execution time, communicates to the Database Recovery utility where to stop processing the input to correctly perform your request.

The following figure shows how DBRC works with the Database Recovery utility.

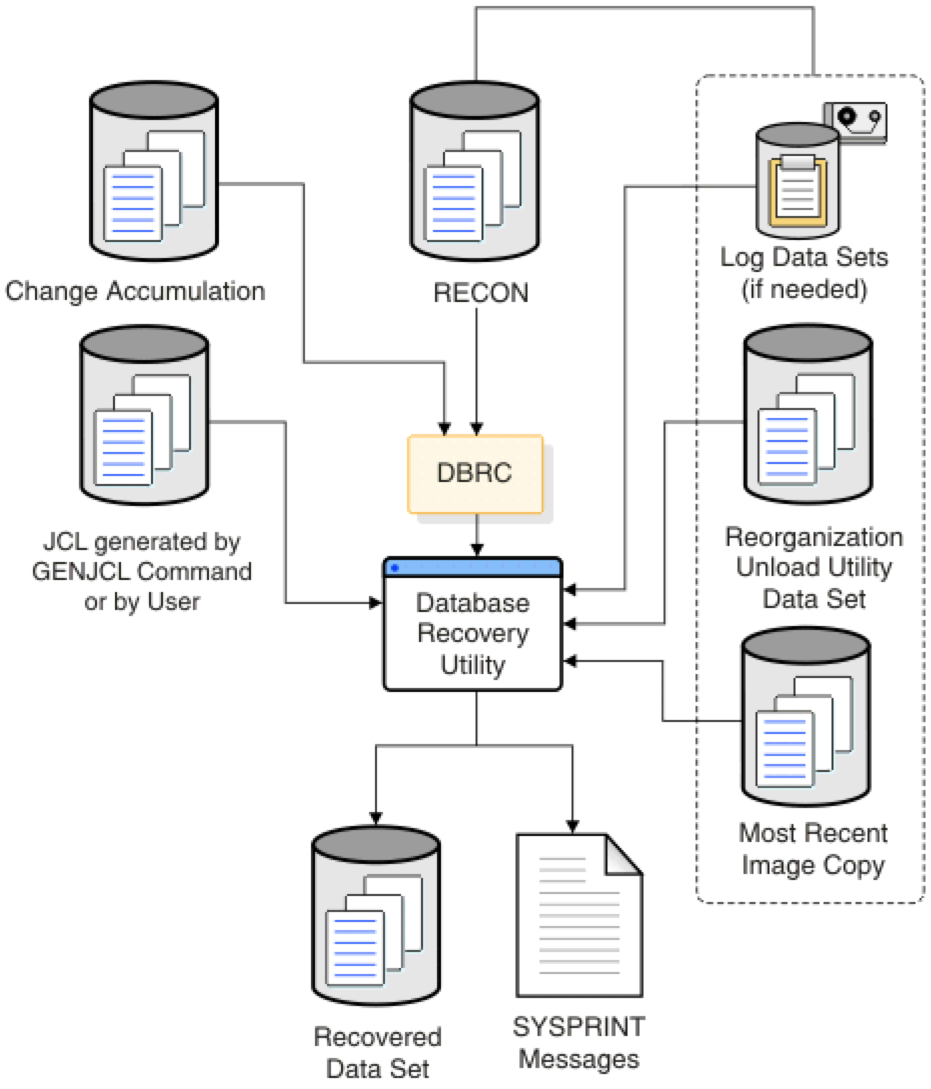

Change Accumulation
RECON
Log Data Sets (if needed)
DBRC
JCL generated by GENJCL Command or by User
Reorganization Unload Utility Data Set
Database Recovery Utility
Most Recent Image Copy
Recovered Data Set
SYSPRINT Messages

Figure 58. How DBRC works with the Database Recovery utility

Recommendation: Implement DBRC in phases, defining at first only a few recoverable databases in the RECON data set. This allows you to gain experience in the use of DBRC, and gives you an opportunity to assess, make, and test any changes needed in your backup, recovery, and operational procedures.

Chapter 34. How DBRC helps in recovery

DBRC is invoked by recovery tools and utilities to provide information about the resources required for database recovery. These resources can include information about image copies, logs, change accumulations, and database data sets.

Generating recovery JCL

You can use either the **`GENJCL.RECOV`** command or the **`/RMGENJCL`** command to generate the JCL that is necessary to recover a registered database data set.

Using information recorded in the RECON data set, DBRC:

1. Selects the image copy data set to use for loading the most recent image copy
2. Selects the change accumulation and log data sets that are to be input to apply all the changes that were logged since the image copy was created

If change accumulation input is required (because of data sharing), but it is not present or usable, DBRC informs you of that fact and the **`GENJCL.RECOV`** command fails.

The **`GENJCL.USER`** command can generate user-defined output, which can include JCL. No skeletal JCL execution members are supplied to support the **`GENJCL.USER`** command. If you want to enter **`GENJCL.USER`** commands, you must supply the members to support them.

When you issue either the **`GENJCL`** command or the **`/RMGENJCL`** command, DBRC reads skeletal JCL and replaces symbolic parameters with actual values based on the information recorded in the RECON data set to build the appropriate JCL. For example, if you request that DBRC generate JCL to recover a database, DBRC retrieves the skeletal JCL member from the library and completes the JCL with information about the latest image copy, change accumulation, and log data sets, if necessary. Your databases must be registered in order for DBRC to generate JCL to process them.

The amount of time and effort required to recover a database can be significantly reduced by using the **`GENJCL`** command to generate the JCL and control statements necessary for the recovery. Using the **`GENJCL`** command also eliminate the causes of many recovery errors. You could spend a large amount of time during database recoveries determining which input data sets should be provided in what order to the Database Recovery utility.

When change accumulation data sets or PRILOG records (in the RECON data set) are available, DBRC selects them rather than the SLDS for recovery. This results in quicker database recoveries if you run the Database Change Accumulation regularly. DBRC knows which log data sets are required and ensures that IMS processes all volumes in the correct order. DBRC also selects the most recent image copy for database recovery.

DBRC always selects the optimum input for the Database Recovery utility by using change accumulation data sets whenever possible. If you have not used the Database Change Accumulation utility, or if that utility did not process some log data sets, DBRC selects the required log data sets from the PRILOG (or SECLOG) records, which can contain RLDS, SLDS, or both RLDS and SLDS entries.

Related reading:

- See *IMS Version 15.2 System Definition* for more information about the tailoring actions for IMS PROCLIB members, the DBRC procedure, and the JCLOUT and JCLPDS DD statements.
- See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for details about customizing your own skeletal JCL and about the contents of IMS-supplied skeletal JCL.

Recommendation: For increased availability of data entry databases (DEDBs), use the DEDB Area Data Set Create utility to provide additional usable copies of an online area. It does not provide backup copies for recovery. The DEDB Area Data Set Create utility uses the RECON data set as part of its input.

Validating utility JCL

When the a recovery utility processes a registered database data set, the utility presents its input to DBRC for validation. Whether the recovery JCL was created by you or by DBRC, DBRC verifies that the input JCL to the utility is correct (according to the current information in the RECON data set). It is possible, even if you created the JCL with the **GENJCL** command, that intervening events could invalidate the input JCL before the utility is run.

DBRC is invoked by the following IMS utilities and services to validate input and record the results:

- Index/ILDS Rebuild utility (DFSPREC0)
- Database Image Copy utility (DFSUDMP0)
- Database Image Copy 2 utility (DFSUDMT0)
- Online Database Image Copy utility (DFSUICP0)
- Database Change Accumulation utility (DFSUCUM0)
- Batch Backout utility (DFSBBO00)
- Database Recovery utility (DFSURDB0)
- Log Recovery utility (DFSULTR0)
- Log Archive utility (DFSUARC0)
- HALDB online reorganization
- HD Reorganization Unload utility (DFSURGU0)
- HD Reorganization Reload utility (DFSURGL0)
- HISAM Reorganization Unload utility (DFSURUL0)
- HISAM Reorganization Reload utility (DFSURRL0)
- Database Prefix Update utility (DFSURGP0)
- DEDB Area Data Set Create utility (DBFUMRI0)
- /RECOVER commands

The following figure shows DBRC's role in running the previously mentioned utilities.

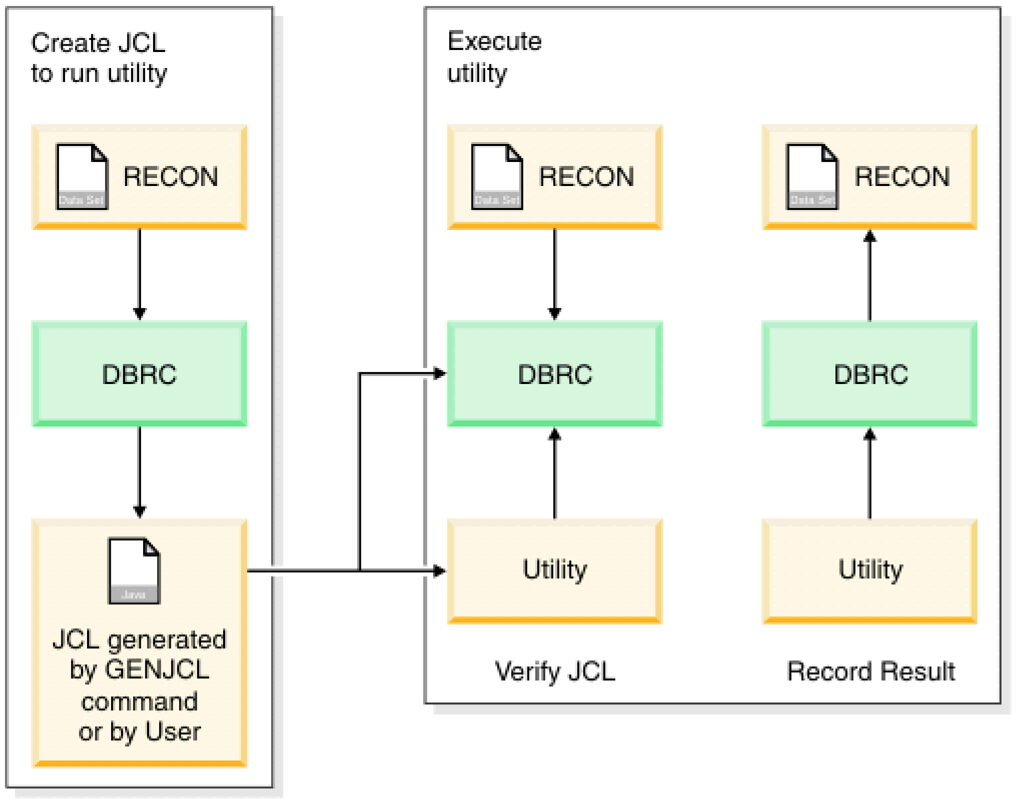


Figure 59. DBRC's role in utility execution

Related reading: For more information about:

- The IMS database utilities, see *IMS Version 15.2 Database Utilities*.
- The IMS log utilities, see *IMS Version 15.2 System Utilities*.
- The **NOTIFY.BKOUT** command, which manually creates a BACKOUT record in the RECON data set, see *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*.
- The **/RECOVER** command, see *IMS Version 15.2 Commands, Volume 2: IMS Commands N-V*.

When you run the Batch Backout utility (DFSBBO00), DBRC determines the complete set of logs that are needed for a particular backout job. In addition, DBRC manages information about the logs so that backout and restart jobs can be easily coordinated.

Exception: DBRC does not verify the JCL input for the HD and the HISAM Reorganization utilities, but does record information about their execution in the RECON data set.

Chapter 35. Recording information about opening and updating databases

After IMS opens a database, DBRC passes back the RECON data set initialization token (RIT) and any extended error queue elements (EEQEs) associated with each DBDS. The RIT allows IMS to determine whether the database has been used without DBRC or whether the database has been controlled by a different RECON data set.

Related reading: See *IMS Version 15.2 Database Administration* for information on EEQEs.

When changes to DBDSs and areas occur, DBRC records information about these changes in the RECON data set. DBRC subsequently uses this information to determine which log data sets might contain change records for a given DBDS or area.

When a DBDS that is registered in the RECON data set is first opened for updates (or allocated), IMS tells DBRC to create an ALLOC record. In the case of a DEDB area, the ALLOC record is created when the area is first opened for update. The ALLOC record identifies the DBDS or area and contains the time stamp of the first update and the open time stamp of the corresponding PRILOG.

When DBRC creates the ALLOC record, DBRC also enters the name of the DBDS or area being changed in the LOGALL record for the PRILOG that is active at the time of the change.

When you deallocate (close) a DBDS or area using a **/DBRECOVERY** command from the operator console of the online IMS subsystem, DBRC writes a deallocation time stamp in the ALLOC record. If no de-allocation time is recorded, DBRC uses the closing time of the associated log as the deallocation time. Thus the RECON data set contains a list of the names of DBDSs or areas for which change records might exist on a given log data set (LOGALL record) and a list of the time ranges where changes could exist for a specific DBDS or area (ALLOC records) and a list of the logs containing the changes.

Chapter 36. Supporting data sharing

Data sharing requires that the databases be registered with DBRC. DBRC checks that subsystems have authority to perform the requested task and that other subsystems are not currently reserving the database.

Related reading: See Chapter 16, "Data sharing in IMS environments ," on page 223 and *IMS Version 15.2 Operations and Automation* for more information on data sharing.

Levels of data sharing

DBRC supports the two levels of IMS data sharing: database level and block level.

Database level
: The entire database or DEDB area is a resource that can be accessed for update by a single IMS system at a time. For area resources this can also be called *Area-level sharing*.

Block level
: A database or DEDB area can be accessed by multiple IMS subsystems concurrently. Data integrity is preserved for the IMS subsystems that access the shared data. Within a database or area, resources are reserved at the block level.

Definition: For OSAM databases, the block is a physical data block stored on DASD. For VSAM databases and DEDBs, the block is a control interval (CI).

Recording data-sharing information in the RECON data sets

DBRC records data-sharing information in the RECON data set for all registered databases.

The following data-sharing information is recorded into the RECON data set:

- Sharing level allowed for each database
- Names of databases or areas currently authorized for processing
- Names of IMS subsystems that are involved
- Statuses of the IMS subsystems
- Database statuses from a recovery viewpoint

Assigning a sharing level with DBRC

The sharing level of a database or DEDB area determines whether a request for access is granted. DBRC allows you to establish one of four sharing levels.

The following sharing levels are defined using the **`INIT.DB`** command and modified with the **`CHANGE.DB`** command.

SHARELVL 0
: The database is not to be shared. The database can be authorized for use by one IMS system at a time. SHARELVL 0 is equivalent to specifying ACCESS=EX on the **`/START`** command.

SHARELVL 1
: Sharing is at the database level. One IMS system can be authorized for update at one time; any sharing systems can only be authorized for read-only processing. Otherwise, the data sharing is for multiple readers.

SHARELVL 2
: Sharing is at the block level but only within the scope of a single IRLM and a single z/OS. Sharing requires that IMS subsystems sharing a database use the same RECON data set. Multiple IMS systems can be authorized for update or read processing.

SHARELVL 3
Sharing is at the block level by multiple IMS subsystems on multiple Realms. Multiple IMS systems can be authorized for nonexclusive access. The IMS subsystems can be on multiple z/OS images using different IRLMs.

Important:

To ensure the integrity of databases in data sharing environments when batch jobs running without IRLM support access SHARELVL 3 databases, DBRC authorizes batch jobs that have update access only if all other IMS systems and batch jobs that are currently authorized by DBRC to the database have read-only access.

If IRLM=N is specified in the IMSCTRL macro or DBBBATCH procedure, DBRC authorizes a batch job for update access to a database only if all other IMS systems and batch jobs that are currently authorized by DBRC to the database have read-only access.

When a batch IRLM=N job has authorization for update, authorization for an online system or other batch job fails unless it is for read-only access.

Sharing data using read-only access is not recommended for production environments without the use of IRLM. IMS outages can occur due to undetectable loops. Failure to use IRLM can lead to data being overwritten. This data cannot be recovered. For more information, see "Database integrity without data sharing support" on page 226.

Chapter 37. Supporting IMSplexes

One DBRC function that is available only in an IMSplex is that when an I/O error occurs on a RECON data set in an IMSplex and a spare data set is available, the instance of DBRC that noticed the error copies the good RECON data set to the spare, activates the spare, and deallocates the original RECON data set.

At this point in the processing, the DBRC that noticed the I/O error can automatically notify the other DBRCs in the IMSplex about the reconfiguration. Then, after the original RECON data set is deallocated, it can be deleted and redefined as the spare RECON data set.

You can access the RECON data set in either serial or parallel mode. In serial mode, the RECON data set is available only to one DBRC instance at a time because that DBRC uses a z/OS RESERVE macro to lock the entire RECON data set while the update is in progress. In parallel mode, DBRC uses the Transactional VSAM function of DFSMS to provide record-level sharing for the RECON data set. All access to the RECON for a single DBRC request (from IMS or a command or API) is processed by Transactional VSAM as one or more transactions.

Related reading: For more information about accessing the RECON data set in parallel mode, see "Accessing the RECON data sets in parallel mode" on page 532.

Chapter 38. Defining DBRC to IMS

You define DBRC to IMS as part of the IMS system definition process. This topic provides a brief overview of the requirements for creating and executing a DBRC region.

Related reading:

- See Chapter 44, "Initializing and maintaining the RECON data sets," on page 531 for information on creating and allocating the RECON data set.
- See Chapter 39, "Registering databases and database data sets," on page 495 for information about registering databases.
- Refer to *IMS Version 15.2 Installation*, *IMS Version 15.2 Release Planning*, and *IMS Version 15.2 System Definition* for:
  - A complete description of the IMS installation procedures and requirements.
  - Information about the DBRC procedure and its parameters.
  - Information about the DFSIDEF0 module.
- See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for detailed information about the **`INIT.RECON`** command.

IMS online systems always use DBRC; you cannot override this.

You can choose whether IMS batch jobs can use DBRC or not, but certain functions, such as data sharing, cannot be used without DBRC.

Batch and utility regions can use the DFSIDEF0 module during initialization. In DFSIDEF0, you can set the DBRC= keyword to YES, NO, or FORCE. The DFSIDEF0 module that is shipped in the ADFSSMPL library contains a batch and utility region default of DBRC=YES. This value is coded on the DFSIDEF macro.

1. Use the DBRCNM= parameter to request that IMS create a cataloged DBRC procedure.

   You can override the DBRC procedure name specified in your system definition in the DFSPBIMS, DFSPBDBC, and DFSPBDCC members.
2. Determine whether DBRC is used in a batch procedure.

   If DBRC is going to be used in a batch procedure, specify this in the DBRC= EXEC parameter. This EXEC parameter is ignored by an online IMS.

   The DBRCNM= parameter can be used to override the DBRC procedure name for an online IMS execution.
3. Place DBRC's load modules into a load library that is in the normal load library search sequence for your IMS load modules, for example, IMS.SDFSRESL.
4. Include the DBRC procedure during system definition.

   IMS automatically starts the DBRC procedure by issuing a z/OS **START** command during control region initialization. This procedure specifies parameters for the DBRC region. DBRC runs in its own address space for an online IMS subsystem.

   In IMS 15.2, the DBRC procedures are created during SMP/E processing in the IMS.SDFSPROC data set and should be copied to SYS1.PROCLIB. The member name must match the name specified on the DBRCNM parameter in the IMSCTRL macro or the applicable EXEC procedure.

   If DBRCNM is specified in more than one place, or if DBRCNM is not explicitly specified, the following order of precedence applies:

   - DBRCNM=DBRC is the default.
   - DBRCNM=*name* in the IMSCTRL macro overrides the default.
   - DBRCNM=*name* defined in a DFSPBxxx member in IMS.PROCLIB overrides the IMSCTRL macro setting.

- DBRCNM=*name* defined in a JCL EXEC parameter overrides the IMS.PROCLIB member setting.

5. Initialize the RECON data sets.

   Use the Access Method Services (IDCAMS) **DEFINE CLUSTER** command to create the RECON data sets and then use the **INIT.RECON** command to initialize the RECON data sets as usable by DBRC.

   If you do not intend to register databases, the **INIT.RECON** command is the only command you need to issue in order to initialize the data set.

Chapter 39. Registering databases and database data sets

The RECON data set must have a DB record for each database whose recovery DBRC is to control.

If you want DBRC to control database recovery, you must register the databases in the RECON data set. DBRC then records information about database updates and about the corresponding log data sets that contain updated log records. DBRC also records the creation of image copy and change accumulation data sets, and records database recoveries and reorganizations that affect registered databases.

For non-HALDBs (High Availability Large Databases), use the **INIT.DB** and **INIT.DBDS** commands to register databases in the RECON data set and to define them as recoverable or nonrecoverable. For HALDBs, you can use the **INIT.DB** and **INIT.PART** commands, or the HALDB Partition Definition utility.

For each non-HALDB database that you have registered, issue the **INIT.DBDS** command to register all its data sets or DEDB areas. For DEDBs, use the **INIT.ADS** command to identify the data sets within each area. An area can have up to seven area data sets (ADSs).

Requirement: Utility jobs that issue either an **INIT.DB** or **INIT.PART** command (to register a HALDB) or any **INIT.DBDS** command, require a DD name for the IMS.DBDLIB data set that contains an entry for the HALDB DBD (or non-HALDB DBDS) for which you are issuing the command. A DD name for the IMS.DBDLIB is not needed for non-HALDB command.

From a DBRC perspective, a HALDB consists of a HALDB master (TYPE=HALDB) and HALDB partitions (TYPE=PART).

To update or delete information about HALDBs in the RECON data set, you can use the HALDB Partition Definition utility or the following DBRC commands; **CHANGE.DB**, **CHANGE.DBDS**, **CHANGE.PART**, **DELETE.PART**, or **DELETE.DB**.

Related reading:

- See *IMS Version 15.2 Database Administration* for an overview of HALDBs and detailed information about how to create them.
- See *IMS Version 15.2 Database Utilities* for information about the HALDB Partition Definition utility.
- See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for information about the CHANGE commands and specifics about the **INIT** commands.

Chapter 40. DBRC access to DBD information when IMS manages ACBs

To perform certain functions, DBRC needs to read the database definitions (DBDs) that are currently active in the IMS system.

When IMS manages ACBs, the DBDs that are currently active are stored in the IMS catalog that the IMS system is using. For DBRC to read active DBDs, you need to specify the IMS catalog name to DBRC, either by recording it in the CATALOG field in the RECON data set, or by specifying it in the DBRC command that requires the DBD information. The CATALOG field in the RECON data set determines the IMS catalog that DBRC uses by default, and the CATALOG field is set by using the **`INIT.RECON`** or **`CHANGE.RECON`** commands. If DBRC does not find the name of an IMS catalog when it requires DBD information, DBRC looks for a DBD library data set, which, if it exists, might not contain the DBDs that are in use by the IMS system.

If you need to override the default IMS catalog, you can specify a different catalog name in the CATALOG field of each command that requires DBD information to instruct DBRC where to read DBDs from. Before ACBs managed by IMS are enabled, DBRC reads from the DBDLIB by default, with the value for the CATALOG field set to NULL.

If you use DDL when ACBs managed by IMS are enabled, IMS does not propagate DDL-defined DBD changes to a DBDLIB and does not check to see whether the DBDs in a DBDLIB are the same as those in use by the IMS catalog. Consequently, if you use DDL to modify or define any databases, you must either ensure that the DBDLIB is kept in sync with the IMS catalog, or use only the DBDs in the IMS catalog for IMS utilities and processes.

Requirement: The catalog that you specify for the CATALOG field in the RECON data set must be an IMS catalog that is registered with DBRC. Registering an IMS catalog with DBRC validates the catalog as a HALDB, and allows you to perform specific DBRC commands.

- If you need DBRC to temporarily read DBDs from a DBDLIB when DBRC is configured to read from an IMS catalog, you can specify NOCATLG on the DBRC command that requires the DBD information in the DBDLIB.
- If you need DBRC to read DBDs from a DBDLIB by default, omit the CATALOG parameter from the RECON or specify NOCATALG to DBRC by using the **`INIT.RECON`** or **`CHANGE.RECON`** commands to change the field to NULL. If you need to use the catalog DBD records later, you can specify a keyword as part of DBRC commands that require access to DBDs.

Related tasks

Defining the IMS catalog with DBRC (System Definition)

Related reference

INIT.RECON command (Commands)

CHANGE.RECON command (Commands)

IMS Catalog Library Builder utility (DFS3LU00) (System Utilities)

Chapter 41. Planning for recovery

Planning for recovery (in non-data sharing and data sharing environments) involves two primary tasks: setting up the mechanisms of recovery (logging, taking checkpoints, keeping records) and setting up your operational procedures for situations that require recovery.

Setting up recovery mechanisms

The logging and checkpoint mechanisms of online IMS subsystems in a non-sharing environment are also active in a data sharing environment.

These include:

- System log data sets and the use of the WADS and restart data sets
- Program, system, and message-queue checkpoints
- Making image copies of databases

The primary difference between non-sharing and data sharing environments is in their degree of reliance on DBRC. DBRC helps control the data-sharing environment; it does not merely keep records.

Specifically, the following items are different in a non-data sharing environment:

- Database reconstruction (forward recovery)
- Program-level (dynamic) backout
- Database backout
- IMS restart (with embedded recovery)

Related reading: See *IMS Version 15.2 Operations and Automation* for detailed information on recovery procedures in non-data sharing and data-sharing environments.

Recovery facilities

Dynamic backout and batch backout support behave differently in non-data sharing and data-sharing environments.

- Dynamic backout

  In non-data sharing and data-sharing environments, an online IMS subsystem dynamically backs out the uncommitted database changes of an application program and discards its uncommitted output messages under either of these conditions:

  - The program fails.
  - The program requests backout with a rollback call.

  In a data-sharing environment, however, IRLM locks and status indicators in the RECON data set ensure integrity by protecting uncommitted changes from sharing subsystems. Operation of the system and other programs continues uninterrupted.

  Related reading: See *IMS Version 15.2 Database Administration* for detailed information about dynamic backout.

- Batch backout support

  DBRC improves database integrity by interfacing with IMS restart dynamic backout and the Batch Backout utility to control access to databases in need of backout. DBRC creates a BACKOUT record to track each unit-of-recovery (UOR) for each database in need of backout. DBRC verifies logs that are input to the Batch Backout utility.

  BACKOUT records are created for online subsystems. BACKOUT records are not created for DL/I batch subsystems unless dynamic backout was being used and it failed.

Prior to performing an emergency restart with the COLDSYS parameter (**/ERESTART COLDSYS**), run batch backout with either the COLDSTART or ACTIVE parameter. After the batch backout completes, DBRC creates a BACKOUT record for all in-flight and indoubt UORs as candidates for backout. The next IMS restart promotes candidate UORs to backout-needed status and DBRC sets BACKOUT NEEDED=ON in the database records in the RECON data set.

When dynamic backout fails, a BACKOUT record is created with a UOR indicating dynamic backout failure and DBRC sets BACKOUT NEEDED=ON in the database record in the RECON data set.

When the databases are backed out successfully, DBRC resets the backout flags in the database records (in the RECON data set) appropriately and updates the BACKOUT records. When backout processing for all of the UORs have been completed, DBRC deletes the BACKOUT record from the RECON data set.

For DBCTL, if you choose not to back out an unresolved indoubt UOR, use either the **DELETE.BKOUT** or **CHANGE.BKOUT** command to remove the unresolved indoubt UOR from the BACKOUT record in the RECON data set.

DBRC commands are available to update BACKOUT records if needed. The need to make manual changes to the backout record should be minimal.

Attention: Use the **DELETE.BKOUT** or **CHANGE.BKOUT** commands with extreme caution because you could compromise database integrity.

DBRC verifies the validity of the logs used as input to the Batch Backout utility. DBRC checks the input log volumes to ensure that they are in the correct sequence, that all the needed logs are provided, and that they are properly closed. When you include ACTIVE or COLDSTART statements in the Batch Backout utility SYSIN statement, DBRC performs an additional check to ensure all volumes related to restart are included. For DL/I batch logs, a check is also performed to ensure that the correct volumes are from the last batch execution.

Attention: If BYPASS LOGVER is included in the SYSIN statement of the Batch Backout utility, then DBRC does not verify the input log and the Batch Backout utility does not notify DBRC about the UORs. Existing UORs are updated when backout processing has completed successfully.

Although DBRC's backout support protects databases from damage caused by the most common errors, the following list describes some other possible sources of damage to databases:

– If an IMS subsystem abends and an **/ERE COLDSYS** command is issued before the in-flight and indoubt UORs have been identified to DBRC (by a Batch Backout run), the databases associated with those in-flights and in-doubts are not protected from erroneous updates until the first Batch Backout run using the COLDSTART or ACTIVE statements.
– Including logs from multiple runs of a batch job in the same log data set (by specifying DISP=MOD) makes log verification unreliable.
– You can modify the RECON data set with DBRC commands. Careless use of these commands could lead to errors, such as:
  - Allowing backouts that should not be performed.
  - Allowing other subsystems access to databases that need to be backed out.
– The Batch Backout BYPASS LOGVER control statement allows backouts to be performed when information in the RECON data set indicates that the input log is invalid or that the backouts are not needed. Careless use of this control statement can cause backouts to be performed that should not be performed.
– Unregistered databases are not protected from being used while backout is needed.
– Backing out a normally terminated job (that did not use IRLM) after a time-stamp recovery was performed (to recover the database to a point in time prior to this log) makes log verification unreliable. DBRC will erroneously validate that the log can be used as input because the log being backed out is the last log for this subsystem name (SSID).

Related reading: See *IMS Version 15.2 Database Administration* and *IMS Version 15.2 Database Utilities* for more information about UORs, the database backout process, and the Batch Backout utility.

- Forward recovery

  The process of recovering a database in a data-sharing environment has certain similarities to recovering a database in a non-data sharing environment. In both environments, use DBRC for database recovery and use the following as input to the recovery process:

  – The most recent image copy of the lost database

    Note: If HALDB Online Reorganization is used for the starting point of recovery, then an image copy is not needed because all database changes will be on the log data set(s).

  – The recovery log data set (RLDS) or all pertinent system log data sets used since that copy was made

  Recommendation: Use the RLDS as input to the recovery process. The RLDS is a more efficient input to the recovery process than the pertinent system log data sets because the RLDS contains only database change log records.

  If you are using the Database Recovery Control utility for forward recovery and block-level data sharing is involved, you might require the additional step of change accumulation.

  Related reading: See *IMS Version 15.2 Database Administration* for further discussion of forward recovery.

Recovery without DBRC

If you perform any recovery-related actions offline when DBRC is not running, such as making database image copies, problems could arise because DBRC would not be notified of the changes in status.

Therefore, DBRC must be specifically informed of such changes. DBRC offers you several commands for this purpose.

Because restart is required, the Utility Control Facility (UCF) might have to be used without DBRC being active. If you use UCF for any recovery-related work, DBRC must be informed of status changes afterward.

Related reading: See *IMS Version 15.2 Database Administration* for detailed information about recovery without DBRC and its related commands.

Restart after an IMS failure

Restart an IMS subsystem in a data sharing environment in the same way as in a non-data sharing environment.

- If you can shut IMS down normally (a checkpoint shutdown using the **/CHECKPOINT PURGE**, **/CHECKPOINT DUMPQ**, or **/CHECKPOINT FREEZE** commands), use the **/NRESTART** command to restart IMS normally.
- If the IMS failed, you must use the **/ERESTART** command to perform an emergency restart.

In a data-sharing environment, if the associated IRLM also stops or fails, you must first start IRLM before you start IMS in a block-level environment.

Related reading: See *IMS Version 15.2 Operations and Automation* for more information about restarting IMS after it fails.

Restart after a DBRC failure

IMS knows whether its DBRC fails because DBRC runs under the control of IMS. If a DBRC failure occurs, IMS terminates abnormally, attempts to flush its buffers, and closes the system log. After correcting the DBRC problem, you can restart IMS by issuing the **/ERESTART** command.

Recovery involving IRLM configurations

DBRC records IRLM status information in the subsystem record in the RECON data set.

Related reference

Fields in a RECON listing, by record type (Commands)

Batch backout

Prior to executing an emergency restart with the COLDSYS parameter (**/ERESTART COLDSYS**), run the Batch Backout utility with either the COLDSTART or ACTIVE parameter.

Provide DBRC with the names of the logs to protect registered databases (that need to be backed-out) from being accessed until their backouts are complete.

For DB level control, if you choose not to back out an unresolved in-doubt UOR, use the **DELETE.BKOUT** command to remove it from the BACKOUT record.

Attention: Use the **DELETE.BKOUT** command with extreme caution. It deletes all backout information for a subsystem from the RECON data set.

Chapter 42. Considerations for a DBRC system

DBRC controls database-related and log-related processes.

These processes include:

- Creating backup copies of databases
- Creating DB change accumulation data sets
- Recovering databases
- Protecting databases that need backout
- Archiving OLDSs

General considerations for using DBRC

DBRC has several considerations to be aware of during it's use.

Be aware of the following considerations when using DBRC:

- DBRC does not support main storage databases (MSDBs).
- DBRC plays no role in the processing of GSAM databases, so there is no reason to register them.
- Logging is required for batch jobs that use DBRC and have update access.
- Never update the RECON data set information (such as for DBDSs or log data sets) about a data set that is currently in use.
- Be sure to set the time-of-day (TOD) clock value in the host processors accurately, or unpredictable results might occur in the RECON data set.

Data set naming conventions

In most cases, the use of z/OS catalog facilities for image copy and change accumulation data sets is optional because DBRC always records volume serial information pertaining to these data sets in the RECON data set. If you catalog image copy and change accumulation data sets, they must have unique data set names.

Requirement: Image copies taken using the DFSMSdss fast replication copy function of the Database Image Copy 2 utility must use z/OS catalog facilities.

DBRC provides a data set naming convention to help you generate unique data set names for those image copy data sets (for HSSP image copies and concurrent image copies, as well as standard image copies) and change accumulation data sets that you define for future use.

If you use this convention all the time, uniqueness of your data set names is assured. If you use the convention only occasionally, you are sent a message at the end of your job step that indicates that you did not follow the naming convention and that duplicate data set names could exist in the RECON data set. DBRC assumes that data set names specified in quotation marks do not follow the naming convention. Therefore, DBRC does not check data set names surrounded by quotation marks.

When you add records to the RECON data set that create data sets for one of the recovery utilities to use in the future and you are using this data set naming convention, you can specify either the fully-qualified data set name or simply the abbreviation `high-level-qualifier.*.low-level-qualifier` (for example, ALPHA1.*.OMEGA). You can use these abbreviated names on any **`INIT`**, **`CHANGE`**, **`DELETE`**, or **`NOTIFY.REORG`** command of DBRC when you are specifying the name of a data set that follows the naming convention. DBRC expands the abbreviated name to its fully-qualified form before it accesses the RECON data set.

The following topics describe additional data set naming conventions.

Naming convention for image copy data sets

The naming convention for image copy data sets that do not use the DFSMSdss fast replication function is `*high-level-qualifier.dbdname.ddname.*IC*.low-level-qualifier*`.

high-level-qualifier
A character string that can be from 1 to 8 alphanumeric characters long. The first character must be alphabetic.

dbdname
The database name of the DBDS for which the image copy data set is being recorded in the RECON data set.

ddname
The data set DD name of the DBDS for which the image copy data set is being recorded in the RECON data set.

IC
Indicates that this is the image copy data set.

low-level-qualifier
A character string that can be from 1 to 8 alphanumeric characters long. It must be unique for each DBDS and the first character must be alphabetic.

The naming convention for image copy data sets that use the DFSMSdss fast replication function can be either of the two following formats:

```
high-level-qualifier.dbdname.ddname.Dyyddd.Thhmmss
```

```
high-level-qualifier.dbdname.ddname.
```

high-level-qualifier
A character string that can be from 1 to 25 alphanumeric characters long. The first character must be alphabetic. The character string may represent multiple data set qualifiers, each from 1 to 8 alphanumeric characters long, separated by a period. The first character of each qualifier must be alphabetic, and the last character of the string must not be a period.

dbdname
The DBD name of the database for which the image copy data set is being recorded in the RECON data set.

ddname
The data set DD name of the DBDS for which the image copy data set is being recorded in the RECON data set.

yyyddd
Is the date. This string must begin with character string 'D'.

hhmmss
Is the time stamp. This string must begin with character string 'T'.

Related reading: For the details of using the DFSMSdss fast replication function with the Database Image Copy 2 utility (DFSUDMT0), see the information about this utility in *IMS Version 15.2 Database Utilities*.

Naming convention for duplicate image copy data sets

The format for duplicate image copy data sets is `*high-level-qualifier.dbdname.ddname.*IC2*.low-qualifier*`.

This format is identical to the convention for image copy data sets, except that the IC2 field indicates that this is a duplicate image copy data set.

Naming convention for change accumulation data sets

The format for change accumulation data sets is: `high-level-qualifier.cagrpname.CA.low-level-qualifier`.

high-level-qualifier
A character string that can be from 1 to 8 alphanumeric characters long. The first character must be alphabetic.

cagrpname
The name of the CA group for which you are creating the change accumulation data set.

CA
Indicates that this is a change accumulation data set.

low-level-qualifier
A character string that can be from 1 to 8 alphanumeric characters long and must be unique for each CA group. The first character must be alphabetic.

Database backup copies

When IMS takes a regular system checkpoint, it records internal control information for DL/I (for Fast Path, IMS also records buffers and MSDBs), but it does not record the external contents of the database. If the database is lost, examining the last system checkpoint does not help. The log can tell you what changes have occurred, but without a backup copy of the database, recovery is impossible.

Recommendation: Make a backup copy of the database after you initially load it, and make new backup copies at regular intervals. The more recent the backup copy is, the fewer log change records need to be processed during recovery, thus reducing the time that is needed for recovery.

IMS enables you to make backup copies in these ways:

- The Database Image Copy utility (DFSUDMP0) runs offline and uses access method services to make the copy.
- The Database Image Copy 2 utility (DFSUDMT0) runs offline and invokes either DFSMSdss Concurrent Copy or DFSMSdss fast replication to make the copy.
- The Online Database Image Copy utility (DFSUICP0) runs online and uses IMS services to make the copy.
- The unloaded data that is output from the HISAM Reorganization Unload utility (DFSURUL0) can be used as a backup copy.
- HSSP processing can also create image copies of DEDB areas.

When these utilities run, they can (depending on installation parameters) call DBRC to update essential information in the RECON data set. If you use a supported image copy utility, DBRC records the image copies for registered databases. DBRC also generates the JCL for the utility if you enter the **`GENJCL.IC`** or **`GENJCL.OIC`** command.

You can also use various utilities supplied by the operating system to make your backup copies; however, these do not interact with DBRC, and so you need to take certain actions to notify DBRC of your non-standard image copies.

Related reading: See *IMS Version 15.2 Database Administration* for more information on HSSP processing of DEDB areas.

Image copy utilities (DFSUDMP0, DFSUDMT0, DFSUICP0)

IMS enables you to "take a picture" of your database before and after changes have been made to the database. The "pictures" are called *image copies*. The term refers to the fact that the copy is an as-is image; the image copy utilities do not alter the physical format of the database as they copy it. Image copies are backup copies of your data that help speed up the process of database recovery.

The Database Image Copy utility (DFSUDMP0), Database Image Copy 2 utility (DFSUDMT0), and Online Database Image Copy utility (DFSUICP0) create image copies of databases. All of the image copy utilities operate on data sets or DEDB areas, so if a database is composed of multiple data sets or areas, be sure to supply the utility with multiple specifications. You can request that one of the supported image copy utilities produce both an image copy data set and a duplicate image copy data set in one run of the utility.

Recommendation: Copy all data sets or areas belonging to a database at one time. If you perform multiple recoveries in order to reset a database to a prior state, recover all data sets belonging to the database and to all logically related databases (including those related by application processing) to the same point to avoid integrity problems.

Each of the image copy utilities provide the option to create backup copies without taking databases and areas offline. You can use this capability to provide increased database availability. Image copies taken while the database is available for concurrent update processing by IMS applications are called concurrent image copies or *fuzzy* image copies. Changes already made to the database by active applications might be missing from the copy because the changes might not have been physically written to the data set. These changes, however, have been written to the log. In this case, it is necessary to go back to some earlier point in the logs to ensure that all changes are applied. How far to go back depends on the type of database and which image copy utility was used.

When concurrent copy or fast replication image copies are not used, the database must be either taken offline or made available only for 'read' access and a consistent or 'clean' image copy is taken. See “Concurrent image copy” on page 508 for more information.

If the image copy was made while the database was not being accessed for update, only changes that were logged after the run time of the copy are required.

When using these utilities, you have the option of creating one to four output image copies. Only the Database Image Copy 2 utility allows three or four output copies and only the first two output copies are recorded in the RECON data set. When the Database Image Copy 2 utility uses the DFSMSdss fast replication function, only one output copy is created.

The advantage of making multiple copies is that if an I/O error occurs on one copy, the utility continues to completion on the other copies. Also, if one copy cannot be read, you can perform recovery using another. The trade-off in deciding whether to make multiple copies, is that performance can be degraded because of the time required to write the additional copies.

DBRC works similarly with the three image copy utilities. When the DFSMSdss fast replication function is not used, the rules for pre-definition and reuse of image copy data sets apply to all three. Each utility calls DBRC:

- To verify the input to the utility (DBRC allows it to run only if the input is valid.
- To record information in the RECON data set about the image copy data sets that it creates.

An image copy record in the RECON data set has the same format whether its corresponding image copy data set was created by the Database Image Copy utility, the Database Image Copy 2 utility, or by the Online Database Image Copy utility.

Two different commands create image copy jobs:

- **`GENJCL.IC`** for the offline utilities Database Image Copy and Database Image Copy 2
- **`GENJCL.OIC`** for the online utility Online Database Image Copy

When you run batch jobs without logging, take an image copy immediately afterwards; do not count on rerunning the batch jobs, as part of a subsequent recovery, in combination with the Database Recovery utility. The database could be damaged by the combination because the batch processing is not guaranteed to be physically repeatable.

Database image copy (DFSUDMP0)

The Database Image Copy runs offline. It supports a concurrent image copy (CIC) option that enables you to create an image copy while the database remains online. The CIC option can create image copies of OSAM data sets and VSAM Entry Sequenced Data Sets (ESDSs). The Database Image Copy does not support creating image copies of VSAM Key Sequenced Data Set (KSDSs).

Use the Database Image Copy (DFSUDMP0) utility to create data set copies of:

- HISAM databases
- HIDAM databases
- HDAM databases
- PHDAM (partitioned HDAM databases)
- PHIDAM (partitioned HIDAM) databases
- PSINDEX (partitioned secondary index) databases
- DEDB areas

To request a concurrent image copy, use the CIC keyword on the **GENJCL.IC** command. Alternatively, you can specify the CIC parameter on the EXEC statement for image copy job. DBRC must be used by the utility and you can only take a concurrent image copy of a database that has been registered with DBRC.

When you run the Database Image Copy utility to take a consistent image copy (CIC option not specified), DBRC is not required but is recommended. DBRC ensures that there is no update activity against the database or area while the utility is executing. If you run the utility without using DBRC you must make certain that no updates occur to the database or area. You can issue a **/DBDUMP** command or a**/STOP AREA** command, for example, to prevent updating of the database or area by transactions in the system previously doing updates.

Database image copy 2 (DFSUDMT0)

The Database Image Copy 2 utility (DFSUDMT0) uses either the DFSMSdss concurrent copy function or the DFSMSdss fast replication function to take an image copy.

When using the concurrent copy option, DFSUDMT0 invokes DFSMSdss to dump the input data set. The output is recorded in the RECON data set and can be used for database recovery in the same way as the output from either of the other image copy utilities. You can use this utility for HISAM, HIDAM, HDAM, PHDAM, PHIDAM, and PSINDEX databases and for DEDB areas. Database data sets that are to be copied by this utility must reside on hardware that supports the Concurrent Copy option.

An optional COMPRESS parameter invokes the "compress" option in DFSMSdss concurrent copy. The compress option enables you to reduce the storage space required to hold the image copy; however, using the compress option increases the CPU time that is required to perform the copy operation.

To lower CPU usage and improve compression, the DFSUDMT0 requests compression from DFSMDdss with the COMPRESS ZCOMPRESS(PREFERRED) options, so that if z/Enterprise Data Compression (zEDC) services are enabled in your z/OS system, zEDC is used for compression. If zEDC is not enabled, or if the request for zEDC services is rejected for any reason, the standard DFSMSdss compression routines are used.

Note: The COMPRESS parameter is not available to DFSMSdss fast replication.

When using the fast replication option, DFSUDMT0 invokes DFSMSdss to COPY the input data set. The output is recorded in the RECON data set and can be used for database recovery in the same way as the output from either of the other image copy utilities. You can use this utility for HISAM, HIDAM, HDAM, PHDAM, PHIDAM, and PSINDEX databases and for DEDB areas. Database data sets that are to be copied by this utility must reside on hardware that supports the fast replication option.

The Database Image Copy 2 utility provides greater database availability when taking consistent image copies. The database needs to unavailable for update processing for only a very short period of time while DFSMSdss establishes a concurrent copy or fast replication session. Update processing can then be resumed while the image copy data sets are actually being written. The updates are not included in the image copy.

This utility can also copy the database while it is being updated by IMS applications. The image copy created in this case is a "fuzzy" image copy. The Database Image Copy 2 utility can copy all supported data set types, including VSAM KSDSs, while the databases remain online. In order to copy a VSAM KSDS while it is online, it must be SMS managed and allocated with IDCAMS using the BWO(TYPEIMS) parameter.

Requirement: The Database Image Copy 2 utility must use DBRC and the databases and areas being copied must be registered with DBRC. The utility can create up to 4 copies using concurrent copy; only 1 copy can be made using fast replication. Only the information about the primary and secondary copies are recorded in the RECON data set.

The concurrent copy image copy output from the Database Image Copy 2 utility is in DFSMSdss dump format, which is different than the format of the output of the other image copy utilities. It is, however, directly usable as input to database recovery. Information about the image copies are recorded in the RECON data set with image copy type SMSCIC, or SMSNOCIC.

The image copy output from a fast replication invocation of the Database Image Copy 2 utility is in DFSMSdss copy format, which is different than the format of the output of the other image copy utilities. Like the DFSMSdss dump format, it is directly usable as input to database recovery. Information about the image copies are recorded in the RECON data set with image copy type SMSONLC or SMSOFFLC.

Requirement: The Database Recovery utility must use DBRC when using the SMSCIC, SMSNOCIC, SMSONLC, or SMSOFFLC image copy data sets for recovery.

Online database image copy (DFSUICP0)

Online Database Image Copy (DFSUICP0) runs as a BMP program in the online IMS and DBCTL environments.

You can use it for HISAM, HIDAM, HDAM, PHDAM, PHIDAM, and PSINDEX databases only; it does not support areas. A database being copied by this utility is available for updating by the IMS subsystem in which the utility is executing. Other IMS subsystems cannot have concurrent update access to the database.

If the database being copied is updated while the utility is running, a fuzzy image copy is produced. Recovery with this image copy requires all logs starting with the log that was in use when the Online Database Image Copy utility was started.

Concurrent image copy

IMS provides the capability to take an image copy of a database without taking that database offline. This means the database can be updated while the image copy is being taken and some, all, or none of the updates might appear in the image copy.

This image copy is called "fuzzy" because the copy represents the state of the database over a certain period of time rather than at one specific instant in time. It is also called a 'concurrent image copy' because the copy was taken while update processing is happening.

The ability to take image copies while the databases are being updated by IMS applications allows increased database availability. The offline image copy utilities, Database Image Copy and Database Image Copy 2, provide an option to take a concurrent image copy. Concurrent image copies can be created by both the concurrent copy and fast replication options of Database Image Copy 2. A database being copied by the Online Database Image Copy utility can be concurrently updated by the IMS subsystem in which the utility is running (but not by other IMS subsystems). Image copies created by HSSP processing are also 'fuzzy' copies because the areas are available for update processing while HSSP is running.

When a consistent image copy is input to database recovery, the recovery only requires logs from after the image copy job completed. A concurrent image copy, might not include updates that were made before the image copy process started or while it was executing. Therefore, when a concurrent image copy is input to recovery, logs from before the image copy process was started might have to be supplied to database recovery.

Restrictions: The following restrictions apply to creating concurrent image copies:

- Only databases that are registered with DBRC are eligible for concurrent image copying.
- The Database Image Copy utility (which can take consistent image copies without using DBRC) must use DBRC when creating a concurrent image copy.

- Nonrecoverable databases are not eligible for concurrent image copying because there is no log data to complete the image copy.

Related concepts

"Database backup copies" on page 99
When IMS takes a regular system checkpoint, it records internal control information for DL/I and for Fast Path, but it does not record any of the contents of the database. If the database is lost, examining the last system checkpoint does not allow you to recover it. The system log can tell you what changes have occurred, but without the original database itself, recovery can be impossible.

Creating image copy data sets for future use and reuse

Use the REUSE parameter to inform DBRC that you want to be able to define image copy data sets and record them in the RECON data set for future use.

You define image copy data sets with the **`INIT.IC`** command. When processing the **`GENJCL.IC`** command, DBRC selects one of the image copy data sets for use by the image copy utility.

Restriction: DFSMSdss fast replication, as invoked by the Database Image Copy 2 utility, does not support the REUSE parameter.

When the Database Image Copy utility uses an available image copy data set, DBRC updates its record in the RECON data set with the time stamp of the run of the Database Image Copy utility during which the image copy data set was used.

If you specify the NOREUSE parameter, you cannot predefine image copy data sets (This is the default). You need to supply the output data set name for the utility in either the skeletal JCL member used in processing the **`GENJCL.IC`** command or in the JCL that you produce yourself. When you specify NOREUSE, DBRC dynamically sets the unit type of the output image copy data set. DBRC sets it to the default unit type for the device as specified in the **`INIT.RECON`** and **`CHANGE.RECON`** commands. Specify NOREUSE when you want more than two DFSMSdss concurrent copies (you can have up to four).

If you do not specify REUSE, every time the image copy utility is run DBRC deletes the oldest image record that exceeds both the GENMAX and RECOVPD values. The image copy data set itself is not deleted-only its record in the RECON data set is deleted. You must either delete the data set or keep track of it yourself, because DBRC is no longer aware of its existence.

If you are using the image copy option of HSSP for a DEDB area, the area must be defined with the REUSE parameter and the data sets you predefine must be cataloged.

Controlling the number of image copies managed

Use the **`INIT.DBDS`** and the **`CHANGE.DBDS`** commands to specify how many image copies of the DBDS or area that DBRC is to maintain records of.

The GENMAX parameter specifies the maximum number of recovery generations (images) that DBRC maintains for the identified DBDS or DEDB area. Duplicate image copy data sets are not included in this number.

Use the RECOVPD parameter to maintain data for a certain period.

Related reading:

- See "Recovery period of image copy data sets and GENMAX" on page 510 for more information about the RECOVPD parameter.
- See **`INIT.DBDS`** command in *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for more information on the REUSE, NOREUSE, GENMAX, and RECOVPD.

Recovery period of image copy data sets and GENMAX

When you register a DBDS in the RECON data set, you specify the maximum number of image copy generations for DBRC to record with the GENMAX parameter of the **INIT.DBDS** command. When this number is exceeded, DBRC discards the information relating to the oldest image copy.

For example, if you take image copies daily and want to keep four days of back level image copies, specify a GENMAX of 4. However, in some emergency situations, you may take backup copies more frequently, possibly three in one day.

To prevent DBRC from discarding information relating to the earlier copies, specify the optional recovery period parameter (RECOVPD) to indicate the number of days you want information retained. If the GENMAX limit is reached, but the RECOVPD for the oldest image copy record has not expired, DBRC issues a warning message (DSP0065I), and does not discard the record. If the DSP0065I warning message appears frequently, you might need to tune the GENMAX or RECOVPD values with the **CHANGE.DBDS** command.

A *recovery period* is the minimum amount of time that information is maintained in the RECON data set. For example, if the recovery period of a DBDS or DEDB area is 14 days, DBRC maintains sufficient recovery-generation information for at least 14 days.

If both GENMAX and RECOVPD have been specified for a DBDS or DEDB area, DBRC considers both when deciding whether to reuse or delete an image copy data set.

The following table shows the results of **GENJCL.IC** processing when both GENMAX and RECOVPD have been specified for a DBDS or area defined with the REUSE parameter.

*Table 45. Results of **GENJCL.IC** processing when GENMAX and RECOVPD are specified with REUSE*

Number of Image Data Sets	Number of In-Use Image Copies	Age of Oldest Image Copy	GENJCL Result
=GENMAX	=GENMAX	<RECOVPD	Fail (DSP0063I)
>GENMAX	=GENMAX	<RECOVPD	Avail IC DS used (DSP0065I)
=GENMAX	=GENMAX	>RECOVPD	Oldest IC DS reused
>GENMAX	=GENMAX	>RECOVPD	AVAIL IC DS used

The following table shows the results of running an image copy utility when both GENMAX and RECOVPD have been specified for a DBDS or area defined with the NOREUSE parameter.

*Table 46. Results of **GENJCL.IC** processing when GENMAX and RECOVPD are specified with NOREUSE*

Number of Image Copies	Age of Oldest Image Copy	Utility EOJ Result
=GENMAX	>RECOVPD	Delete oldest image copy
=GENMAX	<RECOVPD	No delete (DSP0064I)
<GENMAX	N/A	No delete

If you issue a **CHANGE.DBDS** command and specify new GENMAX and RECOVPD values that are less than the existing values, any used image copy data sets that are beyond the recovery period are deleted until the number of remaining image copy data sets equals the specified GENMAX value.

If you issue the **DELETE.IC** command, any specified image copy data set record is deleted regardless of RECOVPD or GENMAX.

Related reference

CHANGE.DBDS command (Commands)

Recovery period of change accumulation data sets and GRPMAX

When you register a CA group in the RECON data set, you specify the maximum number of change accumulation generations for DBRC to record with the GRPMAX keyword of the **INIT.CAGRP** command. When this number is exceeded, DBRC discards the information related to the oldest change accumulation execution record.

For example, if you run change accumulations daily and want to keep 30 days of back level change accumulations, specify a GRPMAX value of 30. In some situations, however, you might run change accumulation more frequently.

To prevent DBRC from discarding information related to the earlier change accumulations, specify the optional recovery period keyword (RECOVPD) to indicate the number of days you want information retained. If the GRPMAX limit is reached, but the RECOVPD for the oldest change accumulation record has not expired, DBRC issues an informational message (DSP1232I), and does not discard the record. If the DSP1232I message appears frequently, you might need to tune the GRPMAX or RECOVPD values with the **CHANGE.CAGRP** command.

A recovery period is the minimum amount of time that information is maintained in the RECON data set. For example, if the recovery period of a CA group is 30 days, DBRC maintains sufficient change accumulation generation information for at least 30 days. DBRC uses the stop time in the change accumulation execution records to determine if the recovery period has expired.

If both GRPMAX and RECOVPD keyword values are specified for a CA group, DBRC uses both values to determine whether to reuse or delete a change accumulation execution data set.

The following table shows the results of running the Change Accumulation utility when both GRPMAX and RECOVPD values are specified for a CA group that is defined with the REUSE keyword:

Table 47. Results of running the Change Accumulation utility when GRPMAX and RECOVPD are specified with REUSE

Number of change accumulation data sets	Number of in-use change accumulation data sets	Age of the oldest change accumulation data set	GENJCL results
= GRPMAX	= GRPMAX	< RECOVPD	Fails (DSP1229A)
> GRPMAX	= GRPMAX	< RECOVPD	Available CA data set is used (DSP1232I)
= GRPMAX	= GRPMAX	> RECOVPD	Oldest CA data set is reused
> GRPMAX	= GRPMAX	> RECOVPD	Available CA data set is used

The following table shows the results of running the Change Accumulation utility when both GRPMAX and RECOVPD values are specified for a CA group that is defined with the NOREUSE keyword:

Table 48. Results of running the Change Accumulation utility when GRPMAX and RECOVPD are specified with NOREUSE

Number of change accumulation data sets	Age of the oldest change accumulation data set	Utility end-of-job results
= GRPMAX	> RECOVPD	Delete the oldest CA data set
= GRPMAX	< RECOVPD	No delete (DSP1233I)
< GRPMAX	N/A	No delete

Related reference

INIT.CAGRP command (Commands)

Reusing image copy data sets

DBRC enables you to reuse old image copy data sets. The REUSE parameter of the **INIT.DBDS** command, in addition to enabling you to define image copy data sets for future use, enables DBRC to reuse image copy data sets. To reuse the image copy data set means that the same name, volumes, and physical space are used for the new image copy.

Restriction: DFSMSdss fast replication, as invoked by the Database Image Copy 2 utility, does not support the REUSE parameter. Consequently, the information in this topic does not apply for fast replication image copies.

A run of one of the IMS image copy utilities automatically reuses the oldest image copy data set for a DBDS or area with the REUSE attribute when all of the following conditions are met:

- A number of image copy data sets equal to the current GENMAX value are recorded in the RECON data set. To see the current GENMAX value, use the **LIST.DBDS** command.
- The Database Image Copy utility or Online Database Image Copy utility used all image copy data sets for this DBDS. None of the image copy data sets is available.
- The oldest image copy is beyond the recovery period.
- A run of one of the IMS image copy utilities automatically uses an available image copy if one exists. If GENMAX has been reached and the oldest image copy is beyond the recovery period or there is no recovery period, the oldest image copy data set is made available.

When you use a **GENJCL.IC** command to generate the job for the Database Image Copy utility, the image copy data set that is to be reused is automatically selected. If the number of image copy data sets is less than the GENMAX value and all image copy data sets have been used, more image copy data sets must be defined for the DBDS or area before running the Database Image Copy utility. The number of image copy data sets should be greater than the GENMAX value if you want to use a recovery period.

The Database Image Copy 2 utility can create up to four output image copy data sets. However, if you issue a **GENJCL.IC** command for a DBDS defined as REUSE, DBRC generates JCL for only one or two output copies (because you can define image copy data sets for only one or two copies). If you use the **GENJCL.IC** command for the Database Image Copy 2 utility to process a DBDS defined as REUSE and you want more than two output copies, modify the generated JCL before you run the job.

DBRC generates JCL so that the oldest DASD data set is always used for output for the image copy. DBRC has the same capability with tape volumes. However, you need to analyze your existing tape library techniques to make sure there is no conflict.

HSSP image copy

The high speed sequential processing (HSSP) image copy (IC) process requires that a database be registered with DBRC and requires that an area be registered with the REUSE attribute for recycling the predefined IC data sets. Image copies that are created by HSSP are concurrent image copies.

HISAM copies (DFSURUL0 and DFSURRL0)

Using the HISAM Reorganization Unload utility (DFSURUL0) to make backup copies of a database allows you to process an entire HISAM database in one pass (the image copy utilities deal with single data sets or areas). The unload utility (DFSURUL0) also reorganizes the database as it copies it.

Because the unload utility (DFSURUL0) reorganizes the database, before resuming normal online operations, the data set must be reloaded using the HISAM Reorganization Reload utility (DFSURRL0). The logging, which is done between the unload and reload, reflects the old data set organization.

When using the HISAM utility to make a backup copy, reload immediately, or the actual database and the backup database are mismatched.

The reload utility (DFSURRL0) notifies DBRC. The unload utility creates a reorganized copy of each data set. Then the reload utility reloads each data set from the reorganized copy, and through DBRC, creates a REORG and an IC record in the RECON data set for each data set. The IC record points to the data set that was output from the Unload utility and input to the Reload utility. After a database has been reorganized, a DBDS can be authorized only if an image copy of that data set has been created.

Updates of the database between unload and reload operations must be prevented. Updates of the database made after an unload operation but before a reload operation are wiped-out by the reload operation. In addition, the change records that are logged reflect the old organization, so that a subsequent recovery using those log records damages the database.

You can prevent access to a shared database during reorganization by using one of the following methods:

- From an online IMS subsystem, issue a global **/DBRECOVERY** command for the database that is to be reorganized. This prevents any subsequent authorizations except for reorganization and recovery. Ensure that recovery utilities do not run during the reorganization.
- Manually update the RECON data set by specifying the NOAUTH parameter of the **CHANGE.DB** command. This prevents any subsequent authorizations except for the reorganization and recovery utilities. Ensure that recovery utilities do not run during the reorganization. After the reorganization process is complete, manually update the RECON data set by specifying the AUTH parameter of the **CHANGE.DB** command for the database that was just reorganized.

Nonstandard image copy data sets

You can create backup copies of DBDSs by using some means other than the Database Image Copy utility.

For example, you could make a copy of the volume on which a DBDS resides. DBRC does not record the existence of these nonstandard image copy data sets in the RECON data set automatically; use a **NOTIFY.UIC** command to inform the RECON data set of these data sets. If this information is not recorded in the RECON data set, DBRC might misinterpret subsequent information about changes to the DBDS. You cannot issue the **NOTIFY.UIC** command for a DBDS that is defined with the REUSE option.

Before recovering a DBDS or DEDB area with a nonstandard image copy data set, perform the following steps:

1. Close the database using **/DBR** (without NOFEOV). Load the data set from the nonstandard image copy (UIC) and record the event in the RECON data set (by issuing **NOTIFY.RECOV** with RCVTIME specified).
2. Apply the change records from the logs that were produced since the UIC (by running DBRC with USEDBDS or USEAREA for the **GENJCL.RECOV** command or DFSDUMP DD DUMMY statement in the DBRC JCL).

DBRC does not allow DBRC to process any log that contains changes outside the recovery range because an image copy is not used for Step 2. The recovery range is defined by the time-stamp recovery record RECOV TO (image copy time) and RUNTIME values.

For more information about recovering a DBDS or DEDB area with a nonstandard image copy data set, see *IMS Version 15.2 Database Administration*.

Frequency and retention of backup copies

Generally, the more frequently you copy, the less time recovery takes. The further back in time your old copies go, the further back in time you can recover. (Remember that program logic errors are sometimes not discovered for weeks.) Conversely, making each new copy requires work, and each old copy that you save uses additional resources.

Consider the following questions when making backup copies of databases:

- How frequently should I make new copies?
- How long should I keep old (back-level) copies?

There are no precise answers to these questions.

The only firm guidelines are these:

- If a database is composed of several data sets, be sure to copy all of the data sets at the same time.

- If you reorganize a database, immediately make a new backup copy of it. (This is not necessary for online DEDBs or HISAM reorganizations.)
- If you create a new database, immediately make a backup copy of it.
- If you perform a time-stamp recovery, make a backup copy for use in subsequent recoveries.

Log record change accumulation

As IMS runs, the number of stored SLDSs or RLDSs grows. You can use these stored volumes to recover a lost or damaged database, but to use them in their raw form would be inefficient.

It would be inefficient because:

- Each SLDS or RLDS contains a record of activities of the entire system and all the data sets within all the databases. Yet when you are recovering a database, you are doing so for a single data set only. Thus, much of what is on the SLDS or RLDS does not apply.
- The SLDS or RLDS chronologically notes each change to any single record. If a record were changed 100 times since the last backup copy of the data set, the SLDS or RLDS would include 100 such notations. Yet, in recovery, you are only interested in the value the data had at the moment the data set was lost. The previous 99 changes are irrelevant.

The Database Recovery utility requires that change records be input in chronological order, but if a database is shared, the change records might be distributed among different log data sets in a way that makes their input to the utility impossible. A DBRC **`GENJCL.RECOV`** command or DB Recovery utility execution fails if this log data has not been properly merged. Such a failure is accompanied by a message informing you that a change accumulation must be run.

Condensing the accumulated SLDS or RLDS (change accumulation)

The Database Change Accumulation utility offers you a way of sorting through your accumulated log data sets in advance, merging and condensing them.

This utility:

- Merges updates from different subsystems
- Picks out only those log records relating to recovery of databases
- Sorts these records by data set within a database
- Saves only the most recent state of each changed part of each data set.

The following figure illustrates how the Database Change Accumulation utility merges the relevant log data (a, b, and c) from 3 different SLDSs (A, B, and C), from one IMS system, into one change accumulation data set record.

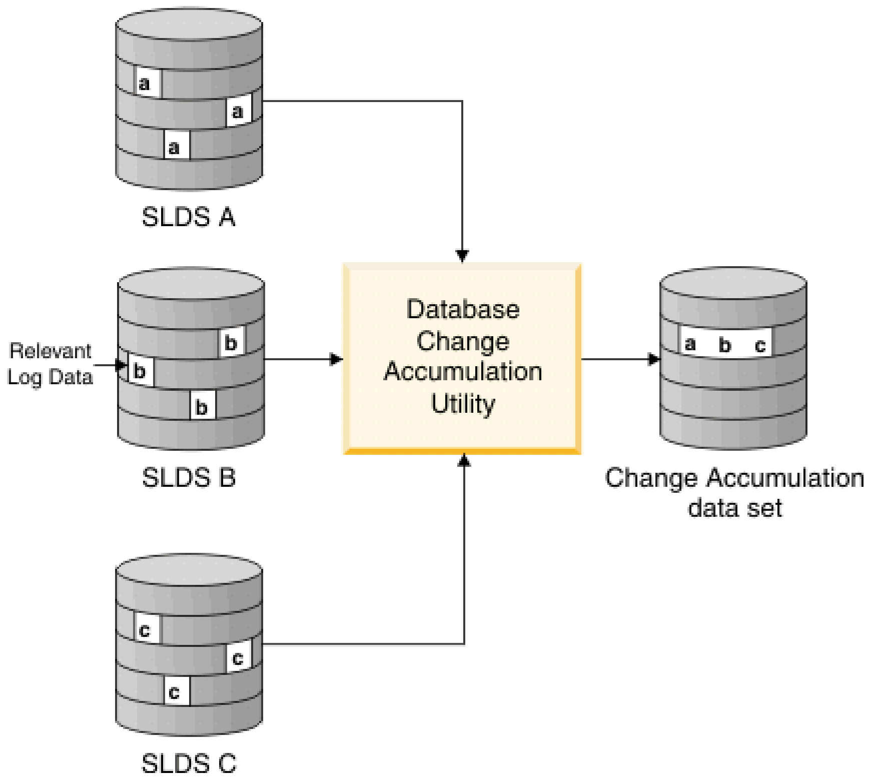

a
a
a
SLDS A
Relevant Log Data
b
b
b
SLDS B
Database Change Accumulation Utility
a b c
Change Accumulation data set
c
c
c
SLDS C

Figure 60. What change accumulation does with redo data from divergent data streams

When is change accumulation required?

Running the Database Change Accumulation utility is not required if, during the time period in question, only one system updated the database; using it periodically speeds any database recovery that becomes necessary. Alternately, you can run the Database Change Accumulation utility only when the need for recovery arises (just before running DB Recovery utility).

Change accumulation would not be required when non-concurrent data set update information exists in various logs. The database changes are received in the correct order if the logs are input serially to DB Recovery utility.

Change accumulation can be required when concurrent data set update information exists in various logs. The logs cannot be input to DB Recovery utility in a way that the change records are seen in the correct order.

Related reading: See *IMS Version 15.2 Database Utilities* for detailed instructions on running the Database Change Accumulation utility.

Input to the database change accumulation utility

In addition to stored SLDSs and RLDSs, you can use a previous change accumulation data set and other IMS log volumes as input. The utility writes the accumulated changes in a new change accumulation data set.

You can specify all log volumes or a subset of log volumes as input to the Database Change Accumulation utility. When you specify a subset of log volumes, DBRC determines whether the subset is complete for each DBDS or area. A subset of log volumes is complete for a DBDS or area when all of the following conditions are true:

- The first volume in the subset is the earliest volume, that could possibly have changes to the DBDS, that were not included in the last change accumulation or in the last image copy.
- The remaining volumes are in sequence.

- In a data sharing environment, logs from all updating subsystems containing changes and any open data streams for a DBDS are included.

The DBRC LIST.CAGRP command indicates whether the log subset for each DBDS of the change accumulation group is complete.

You can use the change accumulation data set as input to a later run of the Database Change Accumulation utility whether your subset of log volumes is complete or incomplete; however, you can use a change accumulation data set as input to DB Recovery utility only if it represents a complete log subset.

Recommendation: If DBRC is being used, perform the change accumulation process using the **GENJCL.CA** command. This command creates the correct JCL and includes all unprocessed log data sets. If you use your own JCL instead, verify the specifications for change accumulation before execution.

An image copy of the specified database data set is needed and must be identified to the RECON data set in order to create a valid starting point for change accumulation records.

All changes since the last valid image copy are collected by the utility. If a time-stamp recovery has occurred since the last image copy, the change accumulation that is created is invalid for use in future recoveries. No error messages are generated by **GENJCL.CA** or by the execution of the utility.

You can run the Change Accumulation utility with a valid log subset at any time to reduce data to a minimum.

When the most recent image copy is used as input to the Change Accumulation utility and that image copy is a concurrent image copy, changes already made to the database by active applications might be missing from the copy because the changes might not have been physically written to the data set. These changes, however, have been written to the log. In this case, it is necessary to go back to some earlier point in the logs to ensure that all changes are applied. How far to go back depends on the type of database and which image copy utility was used.

The point-in-time selected to start the Change Accumulation utility is called the *purge time*.

Related reading: See *IMS Version 15.2 Database Utilities* for more information on the Database Change Accumulation utility.

Change accumulation groups

You can use the Database Change Accumulation utility to process some or all of your recoverable, registered databases as change accumulation (CA) groups. You can, for example, organize the DBDSs into the following groups:

- Application-associated databases
- Physical database clusters
- Logical database clusters
- Volatile (critical data) databases

You can add or delete members of a CA group after you have created it. A database can be a member in only one CA group. To move a member from one CA group to another CA group, you must first delete it from its current CA group and then add it to the new CA group.

Recommendation: Create an image copy of a CA-group member before you move that member to a new CA group. The change accumulation utility processes all logs for the member that are needed for recovery, based on the last good image copy.

A Change Accumulation group can contain from one to 32767 members, which must be registered DBDSs or areas.

Use the **INIT.CAGRP** command to create your Change Accumulation group. To modify your Change Accumulation group, use the **CHANGE.CAGRP**. A DBDS or area can belong to only one Change Accumulation group.

DBRC supports the Change Accumulation utility only by Change Accumulation group. Likewise, **GENJCL.CA** generates JCL only for a Change Accumulation group. The DBRC verification exit routine verifies for a whole Change Accumulation group.

You can run the Change Accumulation utility for DBDSs that do not belong to a Change Accumulation group even with DBRC=YES in effect. However, DBRC does not verify the input to the utility nor record its output.

Defining change accumulation data sets for future use

You can use the REUSE keyword on the **INIT.CAGRP** command to inform DBRC that you want to define a "pool" of data sets to receive the output from the Change Accumulation utility. Define the data sets using the **INIT.CA** command. The number of change accumulation data sets that you can define, can equal up to the value of the **INIT.CAGRP** command GRPMAX parameter that defined the group.

If you define a Change Accumulation group with the REUSE parameter and also use a **GENJCL.CA** command to generate the job for the Database Change Accumulation utility, data set reuse can occur. When all available change accumulation data sets for this Change Accumulation group have been used and the maximum number of change accumulation data sets has been reached, the next run of the Database Change Accumulation utility for this group reuses the change accumulation data set containing the oldest change records. To reuse a change accumulation data set means that its data set name, volumes, and physical space are used as if they were for an empty change accumulation data set.

If you define the Change Accumulation group with NOREUSE:

- You must specify the output data set name for the utility either in the skeletal JCL member used for the **GENJCL.CA** command or in the JCL that you produce.
- When you run the Change Accumulation utility and the number of data sets specified by GRPMAX has been reached, DBRC deletes the record of the change accumulation data set (from the RECON data set) that contains the oldest change records. The data set itself is not scratched. The data set must be manually scratched or monitored if you want to keep it, because DBRC no longer recognizes its existence.

Related reading: See *IMS Version 15.2 Database Utilities* and *IMS Version 15.2 Commands, Volume 1: IMS Commands A-M* for details on the Change Accumulation JCL specifications.

DBDS group considerations

A DBDS group is a named collection of DBDSs or DEDB areas. DBRC can perform various operations by DBDS group so that you do not need to repeat the command for every member of the group.

You can specify DBDS groups on the following commands:

- **GENJCL.IC**
- **GENJCL.OIC**
- **GENJCL.RECOV**
- **GENJCL.USER**
- **LIST.DBDS**
- **LIST.HISTORY**

When you specify a DBDS group on a command, DBRC invokes that command for each member of the DBDS group. For example, you might have a DBDS group for a particular application, like payroll. When performing a time-stamp recovery, for example, all DBDSs of a particular application of a database must be recovered to the same point. If you specify a DBDS group on the **GENJCL.RECOV** command, you need only invoke the command once to recover all DBDSs.

You can also specify a CA group as a DBDS group. DBRC then executes the command for each member of the CA group.

You can define as many DBDS groups as you want. Up to 32767 DBDSs can be in a group. All DBDSs in a group must be registered in the RECON data set. A DBDS can belong to more than one DBDS group.

A database is an implied DBDS group for the **GENJCL** and **LIST** commands. It is unnecessary to define a DBDS group consisting of the DBDSs or areas of a single database. Specify the database name and omit the DD name to operate on the whole database.

The following commands affect the definition of a DBDS group:

- **INIT.DBDSGRP**
- **CHANGE.DBDSGRP**
- **DELETE.DBDS**
- **DELETE.DBDSGRP**
- **LIST.DBDSGRP**

DBDS groups can include ILDS (Indirect List Data Set) and index data sets.

Related reading: See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for the impact of these data sets on the GENJCL commands.

DB groups

A DB group is a named collection of databases or DEDB areas. A DB group name can be specified in the **/START**, **/STOP**, **/DBRECOVERY**, and **/RECOVER** commands, instead of issuing these commands separately for each database or area.

Specifying a DB group name with these commands greatly reduces the number of times these commands must be issued. Use the DATAGROUP keyword to specify the DB group name.

DB groups can also include HALDB master and HALDB partition names. Be aware of the effects of a command issued using a DB group that has a HALDB master name and one or more of its partitions. See *IMS Version 15.2 Database Administration* for more information.

You can define as many DB groups as you want. Up to 32767 databases or areas can be in a group. A database or area can belong to more than one DB group and need not be registered in the RECON data set.

Recommendation: Use a database group whenever possible, even though a DBDS group can be used as a DB group. Processing a DBDS group as a DB group entails increased overhead.

DB groups are a form of a DBDS group, so they are stored in the RECON data set using the DBDS group record. The following commands affect the definition of a DB group:

- **INIT.DBDSGRP**
- **CHANGE.DBDSGRP**
- **DELETE.DBDSGRP**
- **LIST.DBDSGRP**

DBRC groups

When more than one DBRC accesses the same RECON data set in an IMSplex environment, they are considered a DBRC group.

The group:

- Is identified by a DBRC group ID.
- Can access the shared RECON data set in serial or parallel mode.
- Is supported in IMS Version 10 and later.

A DBRC group ID can be specified in the **CHANGE.RECON** command. You can view the DBRCGRP in the **QUERY IMSPLEX** command in the subtype column.

Restriction: DBRCs in the same IMSplex that use different RECON data sets must use different DBRC group IDs when parallel RECON data set access is being used.

Recommendation: If serial RECON access is being used in the IMSplex, DBRCs in the same IMSplex that use different RECON data sets can use the same DBRC group ID, but doing so is not a good idea. When one RECON data set encounters an error, automatic RECON loss notification processing is performed by all DBRC instances in the IMSplex with the same DBRC group ID, thus adding processing overhead to the DBRC instances that are not using the RECON data set in error. Using different DBRC group IDs avoids this additional processing overhead.

DBRC considerations for HALDB Online Reorganization

The following topics describe the DBRC considerations with respect to HALDB online reorganization.

The following DBRC commands support HALDB online reorganization:

- **CHANGE.DB**
- **GENJCL.IC** and **GENJCL.OIC**

  You can generate the JCL for the **GENJCL.IC** and **GENJCL.OIC** commands in advance and not execute the JCL immediately. Therefore, the active set of DBDSs might change in the interim time period.

  GENJCL no longer generates DD statements for the HALDB DBDS that is being copied because dynamic allocation is recommended for the Database Image Copy utility and the Database Image Copy 2 utility. The SYSIN control statement identifies which DBDS to copy: either the A-through-J or the M-through-V set of data sets. The image copy utility then copies the corresponding active set of data sets: A-through-J or M-through-V.
- **INIT.DB**
- **NOTIFY.IC**, **NOTIFY.REORG**, and **NOTIFY.UIC**

  You cannot take an image copy if a HALDB online reorganization cursor is active (OLREORG CURSOR ACTIVE =YES) or has been terminated. The **NOTIFY.IC** and **NOTIFY.UIC** commands fail if they attempt to record an image copy while the HALDB online reorganization cursor is active.

HALDB online reorganization relies on a number of records to track information and activities related to online reorganization. This section includes descriptions of some of the records impacted by HALDB online reorganization:

- HALDB master record
- Partition DB record

  The DB record for a HALDB partition contains information for HALDB online reorganization. For a complete description of the fields in the DB record for a HALDB partition, see "Fields present in a listing of a RECON by record type" in *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*. The DB record includes:

  – A field that indicates whether the database can be reorganized online (ONLINE REORG CAPABLE=). This field is also in the HALDB master DB record. This field cannot be reset if:

    - Online reorganization is active
    - Any of the M-through-V and Y data sets are active
    - Any partition for the HALDB is authorized

    If these conditions do not exist and the field can be reset, all of the M-through-V and Y data set information in the RECON data set is deleted.

  – A field that indicates whether the HALDB online reorganization cursor is active (OLREORG CURSOR ACTIVE=).

  – The SSID of the IMS that is currently performing a HALDB online reorganization (OLRIMSID=).

  – A field that indicates which set or sets of DBDSs is or are active for the partition (ACTIVE DBDS=).
- DBDS record

The M-through-V and Y DBDS records are defined in the RECON data set with the same attributes as the corresponding A-through-J, L, and X DBDS records. The M-through-V and Y DBDS records are added to the same CAGROUP as the A-through-J and X DBDS. Similarly, the M-through-V and Y DBDS records are added to each DBDS group where the A-through-J DBDS records are found.

The exceptions to the similarities in attributes include DD names, data set names, and the DSID. The DSID of the M-through-V and Y data set is the same as the corresponding A-through-J, L, and X data set, except that its X'80' bit is set on. For the RECON data set list purposes, the X'80' bit is ignored.

The following table shows the data set ID (DSID) DCB number associated with the A-through-J, L, and X data sets and the M-through-V and Y data sets.

Table 49. DSID DCB numbers for the A-J, L, and X and the M-V and Y data sets

DSID DCB Number	**A-J, L, and X Data Sets**		**M-V and Y Data Sets**	
	PHIDAM	**PHDAM**	**PHIDAM**	**PHDAM**
1	A	A	M	M
2				
3	L	L		
4		B		N
5	X	C	Y	O
6	B	D	N	P
7	C	E	O	Q
8	D	F	P	R
9	E	G	Q	S
10	F	H	R	T
11	G	I	S	U
12	H	J	T	V
13	I		U	
14	J		V	

- REORG record

  The REORG record for a HALDB partition DBDS contains a stop time for HALDB online reorganization. To run an offline reorganization of a database, you must take it offline by issuing a **/DBR** command. Therefore, only the start time of the online reorganization is significant. With HALDB online reorganization, the database is online, so the duration of the process is significant. The stop time of the online reorganization is recorded in the REORG record (STOP=*timestamp*), with information that indicates that it is an online reorganization and indicates whether the online reorganization can be used as input to recovery.

  Related reading: For more in formation on Fields present in a REORG record, see *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*.

  A REORG record is recorded under both the A-through-J, L, and X DBDS and the M-through-V and Y DBDS. The REORG record for the DBDS that was the output data set indicates that it can be used as input for recovery. The REORG record for the DBDS that was the input data set indicates that it cannot be used for recovery after online reorganization completes.

Maintaining recovery-related records affected by online reorganization

For recovery purposes, the A-through-J and M-through-V data sets are treated in the RECON data set as one logical set of DBDSs.

The recovery information in the two DBDS headers is maintained as if they are one logical DBDS. For example, the values for GENMAX, RECOVPD, and CAGRP are the same. Any changes that are made to the active DBDS are also made to the inactive DBDS.

There is a one-to-one relationship between each partition data set; for example, A correlates to M, B correlates to N, C correlates to O, and so on. Also, the M-through-V data sets are defined in the RECON data set with the same attributes as their corresponding A-through-J data sets. You can perform online reorganization on any of these data sets (A-through-J, L and X or M-through-V and Y) at any time, depending on which set is active before the online reorganization starts. Therefore, data sets A and M are one logical data set during an online reorganization.

After an image copy is taken, DBRC removes any extraneous recovery-related records (IMAGE, ALLOC, RECOV, and REORG) from the RECON data set. The extraneous records are determined by the maximum number of recovery generations (GENMAX) settings and the optional recovery period (RECOVPD) settings for the DBDS.

For example, assume that the GENMAX value for data set A (and M) is 2, and that data set A has two image copies. When data set M becomes the active data set and an image copy is taken, the image copy cleanup process deletes the oldest image copy from data set A, including any of its extraneous recovery records. Another image copy of data set M causes the rest of data set A's image copy and recovery records to be deleted. Note that data set A becomes unrecoverable if its image copy and recovery records are deleted.

Online reorganization process

When online reorganization is started, the ID of the IMS that is responsible for the reorganization is assigned to the partition.

After the cursor-active status is recorded in the RECON data set and copying begins between the A-through-J and M-through-V data sets, these changes also occur:

- The DB record for the partition indicates that online reorganization is active (OLREORG CURSOR ACTIVE =YES).
- The status of the active DBDSs shows that both data sets are active (ACTIVE DBDS=A-J and M-V).
- REORG records that show the start time of the online reorganization are written. The stop time is zero.
- The REORG record associated with the output data set now indicates that this data set can be used as input for recovery.
- The REORG record associated with the input data set does not indicate that this data set can be used for recovery. DBRC uses this record to ensure that this data set cannot be recovered after online reorganization completes.
- The M-through-V and Y data sets are defined in the RECON data set if they do not already exist.
- The M-through-V and Y DBDSs are added to the same change accumulation group as the A-through-J, L, and X DBDSs, if a change accumulation group is assigned.
- The M-through-V and Y DBDSs are added to any DBDS group where the A-through-J, L, and X DBDSs are found.

After these changes take place, the online reorganization must complete; otherwise, you must run an offline reorganization to reset the HALDB online reorganization settings.

You cannot start online reorganization if it is already running for a partition. However, another IMS can take over the online reorganization if it is terminated; that is, when there is no IMS ID assigned.

You can instruct the IMS system assigned to the reorganization to release its ownership if the system terminates with one of the following options:

- OPTION(REL) on the **`INITIATE OLREORG`** or **`UPDATE OLREORG`** commands

- RELOLROWNER=Y in the DATABASE section of the DFSDFxxx member of the IMS.PROCLIB data set.

Another IMS system can then resume the suspended reorganization.

When the online reorganization process completes successfully, these changes occur:

- The REORG records are updated to include the HALDB OLR stop time.
- The DB record for the partition is changed to indicate that online reorganization is not active (OLREORG CURSOR ACTIVE = NO).
- The status of the active DBDSs is switched to the newly reorganized data sets.
- The online reorganization IMS ID is cleared.

Recommendation: Create an image copy of the HALDB partition DBDSs as soon as possible after a successful online reorganization.

Restriction: An image copy of a DBDS cannot be taken while the online reorganization cursor is active (OLREORG CURSOR ACTIVE =YES). DBRC causes the image copy to fail in this situation.

Chapter 43. DBRC security

The RECON data sets are critical to the integrity of IMS databases. Therefore, consider restricting access and providing access to only a subset of the DBRC commands for those users who must issue them.

In the online region, IMS provides some authorization functions for commands issued. By default, IMS restricts **/RMxxxxxx** commands (except for **/RMLIST**) to the Master Terminal Operator (MTO).

You can establish authorization control for DBRC commands through RACF (or an equivalent security product), a user exit routine, or both. For example, logon ID SMITH could have access to issue an **INIT.DB** command against DBD PAYROLL but not against DBD CUSTOMER.

The HALDB Partition Definition utility is an ISPF application that allows you to manage the definitions of IMS HALDBs and their partitions in the RECON data set, providing functionality equivalent to the following:

- **INIT.DB**
- **INIT.PART**
- **CHANGE.DB**
- **CHANGE.PART**
- **CHANGE.DBDS**
- **DELETE.DB**
- **DELETE.PART**
- **LIST.DB**

Authorization for the HALDB Partition Definition utility is controlled using the same resources as defined for these commands with one exception: the **CHANGE.PART** resource is used in place of the **CHANGE.DBDS** resource.

Related reading: See *IMS Version 15.2 Database Utilities* for more information about the HALDB Partition Definition utility.

Security for DBRC commands and API requests

You can establish authorization control for DBRC commands and DBRC API requests by using RACF (or an equivalent security product), an exit routine, or both. You can also disable DBRC security on the RECON data set when necessary.

Authorizing DBRC commands and API requests with RACF

You can set up DBRC command and API request authorization support with RACF by defining resource profiles (explicit or generic) that cover all of the DBRC commands and API requests, and permitting appropriate user access to these profiles.

The resource name corresponds directly to the command and consists of a high-level qualifier (HLQ) and up to three elements of the DBRC command. The resource name model has the following format:

- High-level qualifier (for example, SAFHLQ)
- Command verb (for example NOTIFY)
- Command modifier (for example, PRILOG or IC)
- Command qualifier (for example, dbname or OLDS)

DBRC uses a list of resource names for command authorization support. See "Resource names for command authorization" on page 525.

Different profiles can be used based on the set of RECON data sets being used, but only one HLQ name per RECON data set is allowed. For example, you might use an HLQ name of PRODRECN for the production RECON data sets and an HLQ name of TESTRECN for the test RECON data sets.

Other considerations for your definitions are as follows:

- Resource profiles that protect the DBRC commands must be defined in the FACILITY resource class.
- User identifiers in the ACEEUSRI field of the accessor environment element (ACEE) are used for all authorization checking and the user has READ access to the resource profile, at a minimum.
- DBRC command authorization is enabled or disabled by using the **CHANGE.RECON** and **INIT.RECON** command with the CMDAUTH keyword. The HLQ name is required when enabling command authorization.

Authorizing security profiles for DBRC commands and API requests

You can define security resource profiles for your system to protect specific DBRC commands and requests, and permit appropriate user access to each command or request.

The security resource name corresponds directly to the DBRC command or request and consists of a high-level qualifier (HLQ) and up to three elements of the command or request. Assume that you are using RACF for data security and decide to use IMSA as your HLQ to identify your IMS system.

To restrict access to the **DELETE.SUBSYS** command:

1. Issue the following RACF command to allow access to all DBRC commands:

   ```
   RDEFINE FACILITY IMSA.** OWNER(...) UACC(READ)
   ```

2. Issue the following two RACF commands to restrict the **DELETE.SUBSYS** command to the user IDs listed in the ID parameter:

   ```
   RDEFINE FACILITY IMSA.DELETE.SUBSYS.* OWNER(...) UACC(NONE)
   ```

   ```
   PERMIT IMSA.DELETE.SUBSYS.* CLASS(FACILITY) ID(...) ACC(READ)
   ```

   Command protection is now in effect.

3. Issue the following command to clear the cache and cause RACF to refresh and read the new values from the database:

   ```
   SETROPTS REFRESH GENERIC(FACILITY) RACLIST(FACILITY)
   ```

4. Issue the following DBRC command to specify accesses to the RECON data set. For example, to use the prefix IMSA for security checking:

   ```
   CHANGE.RECON CMDAUTH(SAF,IMSA)
   ```

Authorizing DBRC commands and API requests with the DSPDCAX0 exit routine

You can use the DBRC Command Authorization exit routine (DSPDCAX0) to verify that a user is authorized to issue a particular command or API request.

If you use this exit, you will make the appropriate selection when using the **INIT.RECON** or **CHANGE.RECON** commands. Refer to *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for more information.

The DSPDCAX0 exit routine is required if it is specified to authorize DBRC commands (that is, if either sub-parameter EXIT or BOTH is specified in the CMDAUTH keyword on the **INIT.RECON** or **CHANGE.RECON** commands). The DSPDCAX0 exit routine must be specified in an authorized library or in LINKLST. If the DSPDCAX0 exit routine is in a concatenated STEPLIB or JOBLIB, only the data set that contains the DSPDCAX0 exit routine must be authorized. If the DSPDCAX0 exit routine is in LINKLST, no authorization check is performed.

Related reading: See *IMS Version 15.2 Exit Routines* for more information on the DBRC Command Authorization exit (DSPDCAX0).

Authorizing DBRC commands and API requests with both the DSPDCAX0 exit routine and RACF

The DBRC command authorization exit routine (DSPDCAX0) can be used in conjunction with RACF to provide command authorization.

In such cases, the security product is invoked first. The return and reason codes are passed to the exit routine. The return code that the exit routine issues ultimately determines the success or failure of the command authorization; the exit routine overrides the outcome of the security product. Therefore, DBRC messages that are a result of unsuccessfully invoking the security product will be suppressed.

Again, if EXIT or BOTH is specified in the CMDAUTH keyword on the **`INIT.RECON`** or **`CHANGE.RECON`** commands, DSPDCAX0 is a required exit routine. Also, DSPDCAX0 must be found in an authorized library or in LINKLST. If DSPDCAX0 is found in a concatenated STEPLIB or JOBLIB, only the data set containing DSPDCAX0 must be authorized. If DSPDCAX0 is found in LINKLST, no authorization check is performed.

Related reading: See *IMS Version 15.2 Exit Routines* for more information about the DBRC Command Authorization exit (DSPDCAX0) routine.

Resource names for command authorization

DBRC uses a list of resource names for command authorization support.

The following table lists the resources that can be protected.

Table 50. Resource names for command authorization

Verb	Modifier	Qualifier
AUTH	not applicable	.dbname
BACKUP	.RECON	
CHANGE	.ADS	.dbname
CHANGE	.BKOUT	.ssid
CHANGE	.CA	.grpname
CHANGE	.CAGRP	.grpname
CHANGE	.DB	.dbname
CHANGE	.DB	.ALL
CHANGE	.DBDS	.dbname
CHANGE	.DBDSGRP	.grpname
CHANGE	.IC	.dbname
CHANGE	.PART	.dbname
CHANGE	.PRILOG	.OLDS
CHANGE	.PRILOG	.RLDS
CHANGE	.PRILOG	.SLDS
CHANGE	.RECON	
CHANGE	.RECON	.CATDS
CHANGE	.RECON	.NOCATDS
CHANGE	.RECON	.CHECK17
CHANGE	.RECON	.CHECK44

Table 50. Resource names for command authorization (continued)

Verb	Modifier	Qualifier
CHANGE	.RECON	.NOCHECK
CHANGE	.RECON	.CMDAUTH
CHANGE	.RECON	.DASDUNIT
CHANGE	.RECON	.DUAL
CHANGE	.RECON	.REPLACE
CHANGE	.RECON	.FORCER
CHANGE	.RECON	.NOFORCER
CHANGE	.RECON	.IMSPLEX
CHANGE	.RECON	.NOPLEX
CHANGE	.RECON	.LISTDL
CHANGE	.RECON	.NOLISTDL
CHANGE	.RECON	.LOGALERT
CHANGE	.RECON	.LOGRET
CHANGE	.RECON	.MINVERS
CHANGE	.RECON	.SIZALERT
CHANGE	.RECON	.SSID
CHANGE	.RECON	.STARTNEW
CHANGE	.RECON	.NONEW
CHANGE	.RECON	.TAPEUNIT
CHANGE	.RECON	.TIMEZONE
CHANGE	.RECON	.TIMEZIN
CHANGE	.RECON	.TIMEFMT
CHANGE	.RECON	.TRACEON
CHANGE	.RECON	.TRACEOFF
CHANGE	.RECON	.UPGRADE
CHANGE	.SUBSYS	.ssid
CHANGE	.SUBSYS	.ALL
CHANGE	.UIC	.dbname
CLEANUP	.RECON	
CLEANUP	.RECON	CAGRANGE
CLEANUP	.RECON	CAONLY
CLEANUP	.RECON	LASTCA
CLEANUP	.RECON	.RETPRD
CLEANUP	.RECON	.TIME

Table 50. Resource names for command authorization (continued)

Verb	Modifier	Qualifier
CLEANUP	.RECON	.DBRANGE
CLEANUP	.RECON	.DBONLY
CLEANUP	.RECON	.LASTIC
CLEANUP	.RECON	.LISTDL
CLEANUP	.RECON	.NOLISTDL
DELETE	.ADS	.dbname
DELETE	.ALLOC	.dbname
DELETE	.BKOUT	.ssid
DELETE	.CA	.grpname
DELETE	.CAGRP	.grpname
DELETE	.DB	.dbname
DELETE	.DBDS	.dbname
DELETE	.DBDSGRP	.grpname
DELETE	.IC	.dbname
DELETE	.LOG	.INACTIVE
DELETE	.LOG	.OLDS
DELETE	.LOG	.STARTIME
DELETE	.LOG	.TOTIME
DELETE	.PART	.dbname
DELETE	.RECOV	.dbname
DELETE	.REORG	.dbname
DELETE	.SUBSYS	.ssid
DELETE	.UIC	.dbname
GENJCL	.ARCHIVE	.ssid
GENJCL	.CA	.grpname
GENJCL	.CLOSE	.ssid
GENJCL	.IC	.dbname
GENJCL	.IC	.grpname
GENJCL	.OIC	.dbname
GENJCL	.OIC	.grpname
GENJCL	.RECOV	.dbname
GENJCL	.RECOV	.grpname
GENJCL	.USER	.mbrname
GENJCL	.USER	.dbname

Table 50. Resource names for command authorization (continued)

Verb	Modifier	Qualifier
GENJCL	.USER	.grpname
INIT	.ADS	.dbname
INIT	.CA	.grpname
INIT	.CAGRP	.grpname
INIT	.DB	.dbname
INIT	.DBDS	.dbname
INIT	.DBDSGRP	.grpname
INIT	.IC	.dbname
INIT	.PART	.dbname
INIT	.RECON	
LIST	.BKOUT	.ssid
LIST	.BKOUT	.ALL
LIST	.CAGRP	.grpname
LIST	.CAGRP	.ALL
LIST	.DB	.dbname
LIST	.DB	.ALL
LIST	.DB	.TYPEIMS
LIST	.DB	.TYPEFP
LIST	.DB	.TYPHALDB
LIST	.DBDS	.dbname
LIST	.DBDS	.grpname
LIST	.DBDSGRP	.grpname
LIST	.DBDSGRP	.ALL
LIST	.HISTORY	.dbname
LIST	.HISTORY	.grpname
LIST	.LOG	.ALL
LIST	.LOG	.ALLOLDS
LIST	.LOG	.OLDS
LIST	.LOG	.STARTIME
LIST	.RECON	
LIST	.RECON	.STATUS
LIST	.SUBSYS	.ONLINE
LIST	.SUBSYS	.ssid
LIST	.SUBSYS	.ALL

Table 50. Resource names for command authorization (continued)

Verb	Modifier	Qualifier
LIST	.SUBSYS	.BATCH
NOTIFY	.ALLOC	.dbname
NOTIFY	.BKOUT	.ssid
NOTIFY	.CA	.grpname
NOTIFY	.IC	.dbname
NOTIFY	.PRILOG	.OLDS
NOTIFY	.PRILOG	.RLDS
NOTIFY	.PRILOG	.SLDS
NOTIFY	.PRILOG	.DPROP
NOTIFY	.RECOV	.dbname
NOTIFY	.REORG	.dbname
NOTIFY	.SECLOG	.OLDS
NOTIFY	.SECLOG	.RLDS
NOTIFY	.SECLOG	.SLDS
NOTIFY	.SECLOG	.DPROP
NOTIFY	.SUBSYS	.ssid
NOTIFY	.UIC	.dbname
REPAIR	.RECON	
STDBRC	not applicable	.ssid

For more information about command authorization, see Chapter 43, “DBRC security,” on page 523.

Overriding DBRC security on the RECON data set

You can create a copy of the production RECON data set for test purposes or to send to IBM Software Support to help re-create a problem with the original RECON data set.

A copy of the RECON data set inherits the security settings of the original RECON data set. Disable DBRC security for a copy of the RECON data set when the copy needs to be accessed by someone lacking the level of authorization required by the current security settings.

To create a copy of the RECON data set that does not inherit the security settings of the original RECON data set:

1. Issue the **CHANGE.RECON** command with the *rcnqual* option.

   If the *rcnqual* value is not a substring of the RECON COPY1 data set name, the command bypasses security. However, if the *rcnqual* value is a substring of the RECON COPY data set name, the command bypasses security only if the *rcnqual* value is followed by an asterisk (*) and not at the beginning of the data set name. For example, if rcnqual=IMSTEST*, and IMSTESTL.RECON1 is copied into RCNCOPY.IMSTESTL.RECON1, security would be bypassed because the copied data set name contains the rcnqual value, but does not begin with it. If rcnqual=IMSTEST, security would not be bypassed for the copied data set.
2. Create the copies of the production RECON data set with names that do not contain the string that was specified in the *rcnqual* option.

These copies of the RECON data set can be accessed by those without command authorization.

When an attempt to use the copies is made, the system compares the string value stored in the RECON data set against the RECON COPY1 data set name. Once it verifies that the strings do not match and the string in the copy is not a substring of the RECON data set, message DSP1211W is issued which notifies the user that the current level of command authorization is not being enforced.

After message DSP1211W is issued, data sets without command authorization can complete testing or debugging against the copy of the RECON data set. Message DSP1211W also is issued after every subsequent command as a warning message.

Chapter 44. Initializing and maintaining the RECON data sets

DBRC records recovery-related information in a pair of key-sequenced data sets (KSDSs) called the RECovery CONtrol (RECON) data set. DBRC uses two RECON data sets to increase availability and recoverability. They contain identical information. The data sets are identified by the DD names RECON1 and RECON2.

If you define only two RECON data sets and an error occurs on one of them during operation, the current jobs continue using the remaining one. New jobs cannot start unless your RECON setting allows a job to start with only one active RECON data set. If you want to continue operations in dual mode, you can define a third RECON data set (RECON3). DBRC does not use this spare data set unless an error occurs on one of the two active RECON data sets. Then, DBRC copies the good RECON to the spare data set (RECON3), which then becomes active (thus maintaining RECON dual-mode operation).

The following figure shows the recommended three RECON data set operating configuration.

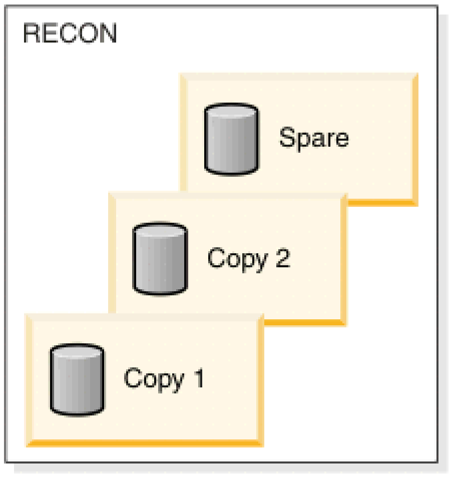


Figure 61. DBRC dual RECON set up with a spare

The RECON data sets are critical resources for both DBRC and IMS. If both RECON data sets are lost, DBRC abnormally terminates rather than compromising database integrity. IMS cannot continue processing transactions without a viable RECON data set, so IMS also abnormally terminates.

Planning considerations for the RECON data sets

Define your RECON data sets specifying VSAM SHAREOPTION(3,3). Each data set should be on a different volume and catalog. The RECON data sets should also be of different sizes.

Each RECON data set is a VSAM KSDS with a 32-byte key.

Recommendations:

- When defining the RECON data sets using Access Method Services (AMS):
  - Use the same index control interval (CI) size and data CI size for all the RECON data sets.
- Ensure that the data CI size specified exceeds the specified index CI size by at least 2048 bytes. Failure to do so can seriously degrade your DBRC performance.
- Use secondary allocation to safeguard against a RECON data set becoming full.

Related concepts

"Avoiding RECON data set space problems" on page 537

Allocate the RECON data sets with different amounts of space so that if one becomes full, the system can continue using the other RECON data set while you provide a replacement. Allocate secondary extents for the RECON data set when you define space for it.

Initializing the RECON data sets for the first time

After allocating the RECON data sets, use the **`INIT.RECON`** command to initialize the RECON data set.

This command is only valid for an empty, un-initialized RECON data set. If the initialization job fails, delete and redefine the data sets before rerunning the job.

Avoiding RECON data set contention problems

Maximum RECON data set availability depends on eliminating deadlock situations and minimizing situations in which more than one RECON data set becomes simultaneously unavailable.

For maximum availability, each RECON data set must:

- Have different space allocations. The spare data set must be at least as large as the largest RECON data set.
- Be on a different volume.
- Be in different user catalogs.

Accessing the RECON data sets in parallel mode

You can access the RECON data sets in parallel mode to avoid data set deadlock situations.

The following terms are used in the topics related to parallel mode:

DBRC group
One or more DBRC instances that share a single RECON data set in an IMSplex environment.

DBRC group ID
The identifier of a DBRC group

DBRC request
Each action asked of DBRC (command process, query, and update) by IMS, DBRC application programs, and jobs.

DBRC uses the Transactional VSAM function of DFSMS to provide parallel access to the RECON data set. Transactional VSAM uses:

- The two-phase commit and backout service provided by z/OS Resource Recovery Services to ensure that either all of the changes made by a transaction are hardened or all of the changes made by a transaction are backed out.
- A coupling facility (CF) for buffer caching, locking, and logging services.

Use the following commands to set the RECON data set access mode to either serial or parallel:

- For a new RECON data set, use a **`INIT.RECON ACCESS(SERIAL or PARALLEL)`** command.
- For an existing RECON data set, use a **`CHANGE.RECON ACCESS(SERIAL or PARALLEL)`** command.

Before parallel RECON data set access can be enabled, both of the following conditions must be true:

- All DBRC instances that are accessing the RECON data set belong to a DBRC group with a unique DBRC group ID.
- All DBRC instances that are accessing the RECON data set are registered to Structured Call Interface (SCI), similar to the situation where RECON Loss Notification is involved. DBRC registers with SCI using the group ID as the member subtype.

Planning for parallel RECON data set access

Set up the Structured Call Interface and Transactional VSAM to prepare for parallel RECON data set access.

You must perform the following tasks to prepare for parallel RECON access:

- Set up the Structured Call Interface (SCI).
  - Define a Common Service Layer (CSL) and configure the IMSplex environment such that there is a Structured Call Interface on each z/OS image from which IMS systems and batch jobs will access the RECON data set.
  - Tailor the DBRC SCI Registration exit routine, DSPSCIX0, as necessary to specify the correct IMSplex name for the RECON data set. Ensure that your exit routine is coded to work with the IMS DSPSCIX0 extended interface.
- Set up Transactional VSAM:
  - Update the IGDSMSxx member in SYS1.PARMLIB to support Transactional VSAM.

    Recommendation: Specify the DEADLOCK_DETECTION in IGDSMSxx so that detection times for both local and global detection are less than 2 seconds.
  - Set up the SMS configuration to enable Transactional VSAM processing.
  - If the RECON data sets are not SMS-managed, convert them to be SMS-managed.
  - Specify a storage class for each RECON data set to assign the data set to a coupling facility (CF) cache structure.
  - Define at least one CF cache structure for the SMS configuration.
  - Define to z/OS the CF lock structure, IGWLOCK00.
  - Define a primary system log stream for each instance of Transactional VSAM.
  - Define a secondary system log stream for each instance of Transactional VSAM.

  Related reading: For more information about configuring SMS and setting up a CF structure for Transactional VSAM, see the *z/OS DFSMS Storage Administration Reference (for DFSMSdfp, DFSMSdss, and DFSMShsm)*. For more information about SYS1.PARMLIB changes and log stream definition for Transactional VSAM, see the *z/OS TSO/E Customization*.
- If you use a RECON I/O exit routine (DSPCEXT0), ensure that the exit routine works with the IMS extended RECON I/O exit interface.
- Ensure that users that issue the **CHANGE.RECON.ACCESS** command, using either the batch command or through the DBRC API, have the following authority permissions:
  - Data set ALTER authority to maintain the LOG parameter on the **DEFINE CLUSTER** command
  - UPDATE authority to the specified data set and to the RACF class profile STGADMIN.IGWSHCDS.REPAIR
- Ensure that read-only jobs have READ authority to the RACF class profile STGADMIN.IGWSHCDS.REPAIR.
- Except for jobs that will run in online DBRC regions, any jobs that are not read-only must have UPDATE authority to the class profile STGADMIN.IGWSHCDS.REPAIR. If UPDATE authority is not allowed, these regions might not be able to recover from errors that involve shunted I/O.
- Ensure that all DBRC instances that share the RECON data set use the same IMSplex name for the Parallel RECON Access and Automatic RECON Loss Notification functions.

  If you activate the Automatic RECON Loss Notification and Parallel RECON Access functions within the same IMSplex, you must use the **CHANGE.RECON IMSPLEX()** command to ensure that all DBRCs in the IMSplex are using the same IMSplex name as is specified in the RECON data set. If you use the DBRC SCI Registration exit routine (DSPSCIX0) or use the IMSPLEX EXEC parameter before issuing the **CHANGE.RECON IMSPLEX()** command, message DSP1136A is issued and subsequent jobs fail due to an unavailable RECON data set.

Avoiding RECON data set deadlock situations for serial access mode

To eliminate deadlocks (if you are using implementing hardware reserves for the RECON data sets), the RECON data sets must be the only objects cataloged in their respective catalogs and be on the same device as their catalogs. When the RECON data sets are accessed, an enqueue on the RECON data sets can result, followed by an enqueue on the catalog. When the RECON data sets and catalog are on the same device, the possibility of conflicts with another job enqueuing the devices in reverse order is eliminated.

Give special consideration to the placement of RECON data sets that are shared among multiple processors. During a physical open, DBRC reserves RECON1, RECON2, and RECON3. DBRC determines which are available and which are Copy1, Copy2, and spare. DBRC then closes and dequeues the spare (if it exists) and any unusable RECON data sets. So, during the use of DBRC, two RECON data sets are reserved most of the time. DBRC always reserves both RECON data sets in this order: RECON1 and RECON2. If RECON1 and RECON2 are specified consistently throughout jobs, DBRC does not encounter deadlock.

However, other jobs that reserve multiple volumes can cause deadlock if any of the volumes also contain a RECON data set.

Recommendation: To eliminate contention and facilitate recoverability, the RECON data sets should be the only type of data set on their respective devices if using implementing hardware reserves for the RECON data sets.

Access to the RECON data sets must be controlled to avoid deadlock. The following z/OS macros are used to control access to the RECON data sets: RESERVE, GRS, OBTAIN, and DEQ. The z/OS macros, DFP Record Management Services, and RECON data set serialization strategies are discussed in the following list.

- RESERVE

  DBRC issues the z/OS RESERVE macro to serialize access to each RECON data set. DBRC keeps the RECON data sets reserved until it completes its processing. The more RECON records DBRC must examine or change, the longer it holds the RECON data sets. The RESERVE macro serializes access to a resource (a data set on a shared DASD volume) by obtaining control of the volume on which the resource resides to prevent jobs on other systems from using any data set on the entire volume. This reserve is done under the major name DSPURI01 and has a scope of SYSTEMS.

  Batch jobs will serialize on another resource name first, before issuing a RESERVE for the RECON data sets. The resource name for batch is DSPURI02 and has a scope of SYSTEM. Each job gets control of the resource, based on the position of the task's request and whether the request was exclusive or share control. The queue is not ordered by the priority of tasks. The effect of this serialization of batch jobs is that an IMS online region never has to wait for more than one batch job to complete before gaining access to the RECON data set.
- Global Resource Serialization (GRS)

  The GRS macro provides a method of converting a RESERVE request into an ENQ request. The ENQ, DEQ, and RESERVE macros identify a resource by its symbolic name. The symbolic name has three parts:

  – Major name (qname)

  – Minor name (rname)

  – Scope (which can be STEP, SYSTEM, or SYSTEMS)

  For example, on an ENQ or DEQ macro, a resource might have a symbolic name of APPL01,MASTER,SYSTEM. The major name (qname) is APPL01, the minor name (rname) is MASTER, and the scope is SYSTEM.

  When an application uses the ENQ, DEQ, and RESERVE macros to serialize resources, global resource serialization uses resource name lists (RNLs) and the scope on the ENQ, DEQ, or RESERVE macro to determine whether a resource is a local resource or a global resource. Global resource serialization identifies each resource by its entire symbolic name. For example, a resource that is specified as

A,B,SYSTEMS is considered a different resource from A,B,SYSTEM or A,B,STEP because the scope of each resource is different. To ensure that resources are treated as you want them to be without changes to your applications, global resource serialization provides three resource name lists (RNLs):

- SYSTEMS EXCLUSION RNL

  The SYSTEMS exclusion RNL contains a list of resources that are requested with a scope of SYSTEMS that you want global resource serialization to treat as local resources.
- RESERVE CONVERSION RNL

  The RESERVE conversion RNL contains a list of resources that are requested on RESERVE macros for which you want global resource serialization to suppress the RESERVE.
- SYSTEM INCLUSION RNL

  The SYSTEM inclusion RNL contains a list of resources that are requested with a scope of SYSTEM that you want global resource serialization to treat as global resources.

  Related reading: See the following publications for more information about GRS and the z/OS RESERVE, DEQ, and ENQ macros:

  - *z/OS MVS Programming: Assembler Services Reference Vol 1*
  - *z/OS MVS Planning: Global Resource Serialization*

- OBTAIN

  DBRC uses a VSAM DADSM (Direct Access Device Storage Management) OBTAIN request to the FORMAT-4 DSCB (the VTOC) to force an I/O that insures that DBRC really has the RECON data set reserved in a multisystem environment.

  The OBTAIN macro is not issued if the Synchronous RESERVE (SYNCHRES) option is in effect. The Synchronous RESERVE feature allows specification upon installation of whether the hardware RESERVE should be obtained for a device prior to granting a global resource serialization ENQ.

  The SYNCHRES option can be activated through either the GRSCNFxx parmlib member or the SETGRS operator command. The GRSDEF statement of GRSCNFxx contains the SYNCHRES (YES | NO) parameter. The default value for SYNCHRES is YES. During normal system operation, the operator can modify the setting of SYNCHRES by issuing the **SETGRS** command. Activate SYNCHRES by issuing **SETGRS SYNCHRES=YES** and deactivate it by issuing **SETGRS SYNCHRES=NO**.
- DEQ

  DBRC releases the RECON data sets by using the z/OS DEQ macro.
- DFP Record Management Services

  DBRC uses VSAM services to retrieve, manipulate, and store the RECON data set records. These records have a 32-byte record key.
- RECON data set serialization strategies

  Recommendations:

  - In a GRS Star configuration, a RESERVE CONVERSION RNL should be implemented for DSPURI01 if all systems accessing the RECON data sets are within the sysplex (or GRSPlex).

    Note: If the RECON data sets are accessed by systems that are outside of the sysplex, the reserve must not be converted. A SYSTEMS EXCLUSION RNL must be implemented instead.

    If you implement GRS RNL CONVERSION by adding the QNAME for the RECON data set, DSPURI01, to the conversion list, the hardware reserve is eliminated and replaced by a GRS enqueue that is communicated to all other sharing z/OS systems.

    Other data sets on the same DASD volume can be used while the RECON data set is reserved; this is the benefit of performing the RNL conversion. GRS RNL conversion uses CPU and storage and affects system performance positively. The performance (in terms of least CPU time used, least storage used, and least elapsed time) is best using this option in a GRS STAR configuration.

    To implement this method, follow these steps:

1. Add the RECON data set QNAME to the RESERVE CONVERSION RNL. For example:

   ```
   RNLDEF RNL(CON) TYPE(GENERIC) QNAME(DSPURI01)
   ```

2. Consider carefully the placement of the following VSAM QNAMEs: SYSZVVDS and SYSIGGV2.

Related reading: For more information on implementing GRS RNL CONVERSION, see *z/OS MVS Planning: Global Resource Serialization*.

– In a GRS Ring configuration, a SYSTEMS EXCLUSION RNL should be implemented for DSPURI01 so that the RECON is serialized by hardware reserve.

  If you implement GRS SYSTEMS EXCLUSION RNL, then GRS does not perform global serialization and the RESERVE macro is issued. This might work well provided that the RECON data set is located on a DASD volume that does not contain other data sets that are needed by other z/OS systems.

  To implement this method, follow these steps:

  1. Add the RECON data set QNAME to the SYSTEMS EXCLUSION RNL. For example:

     ```
     RNLDEF RNL(EXCL) TYPE(GENERIC) QNAME(DSPURI01)
     ```

  2. Carefully consider the placement of the following VSAM QNAMEs; SYSZVVDS and SYSIGGV2.

  Related reading: The *z/OS MVS Planning: Global Resource Serialization* provides more information about VSAM QNAMEs.

Allocating the RECON data sets

For both online and batch DBRC jobs, you can allocate the RECON1, RECON2, and RECON3 data sets with JCL, or you can let DBRC dynamically allocate them.

When dynamically allocating the RECON data set, omit the DD statements for RECON1, RECON2, and RECON3.

Use the TYPE=RECON macro statement of the IMS DFSMDA dynamic allocation macro to establish three dynamic allocation parameter lists in IMS.SDFSRESL. When multiple processors access the same RECON data sets, keep the IMS.SDFSRESL information pertaining to dynamic allocation parameters in synchronization on all processors.

Related reading: See *IMS Version 15.2 System Definition* for information about the DFSMDA macro.

DBRC always allocates RECON data sets with DISP=SHR.

Although JCL allocation and dynamic allocation are both valid methods for allocating RECON data sets, JCL allocation should be used only in a controlled test.

Recommendation: Use dynamic allocation for your production system and all other test or development environments.

The principal advantages of dynamically allocating the RECON data set are:

- All DBRC jobs automatically use the correct and current RECON data sets, and no JCL statements are left to become outdated.
- You can reorganize and restore RECON data sets, in case of error, without having to shut down online IMS systems.

Requirement: When accessing the RECON data sets in parallel mode, they must be managed by DFSMS.

Avoiding RECON data set space problems

Allocate the RECON data sets with different amounts of space so that if one becomes full, the system can continue using the other RECON data set while you provide a replacement. Allocate secondary extents for the RECON data set when you define space for it.

If one RECON data set becomes full during online operation, IMS deallocates it. DBRC responds by copying and reorganizing the good RECON data set to a spare RECON data set, if one is available. If no spare RECON data set is available, the system runs in single-RECON mode.

When all active subsystems have deallocated the failing RECON data set, you can delete and redefine it offline using AMS. See "Replacing a discarded RECON data set" on page 553 for information on replacing a discarded RECON data set. If you are in single mode, and a spare RECON data set is available, the next time DBRC accesses the RECON data set, it automatically enters dual-RECON mode. You do not have to enter the **CHANGE.RECON** command with the DUAL or REPLACE option.

Creating a RECON data set

Create a RECON data set using the **DEFINE CLUSTER** command.

The keywords that are recommended when defining a RECON VSAM KSDS are discussed in the following list. Information on all the keywords can be found in the *z/OS DFSMS Access Method Services for Catalogs*.

CONTROLINTERVALSIZE

The values used with this keyword affect the total amount of storage used by DBRC for VSAM and internal buffers. DBRC uses the Local Shared Resources (LSR) option of VSAM to process the RECON data sets. If the number of index and data buffers created by DBRC is allowed to default, the amount of storage used for RECON buffers is:

```
(60 X index_ci_size) + (120 X data_ci_size)
```

This amount of storage is used when the index and data CI sizes are the same for all RECON data sets. You can change the default number of index or data buffers used by DBRC in an online or batch environment with the DSPBUFFS Buffer Size Specification Facility.

DBRC uses one set of internal buffers for each RECON data set to build logical records from smaller segments. The size of these buffers depends initially upon the CI size or the VSAM maximum record size (LRECL), whichever is smaller. After the initial allocation of these buffers, the logical record buffers grow over time as the largest logical record on the RECON data set grows. Because only the initial allocation of buffers is determined by the VSAM data set definition, the storage allocated to internal buffers cannot ultimately be affected by changes to either the CI or the record size.

DBRC divides its own records into segments, each of which is always smaller than a single control interval and which is seen by VSAM as a complete physical record. VSAM record spanning is not used. Segmenting allows a logical RECON record to be as large as 16 MB independent of the VSAM RECORDSIZE parameter.

Recommendation: Initially set your minimum CI size to a minimum of eight KB. The allowable CI size is affected by the value you select for RECORDSIZE. Also, ensure that the smallest data CI size exceeds the largest index CI size by at least 2048 bytes. Failure to do so can seriously degrade your DBRC performance.

Related reading: See *IMS Version 15.2 Exit Routines* for further details about using the Buffer Size Specification Facility (DSPBUFFS).

CYLINDERS

Specifies the amount of space to allocate to the cluster.

FREESPACE

The default values of FREESPACE(0 0) must not be used. While you are entering initial information in the RECON data set, you must specify a high control-interval percentage (for example, 70%) as free space. Later, you can lower the percentage with an Access Method Services (AMS) **ALTER** command.

INDEXED
: Specifies that the cluster is defined for key-sequenced data.

KEYS
: KEYS(32 0) is required.

LOG
: When parallel RECON access is used, let DBRC manage the LOG parameter on the AMS **DEFINE CLUSTER** command. When parallel RECON access is enabled, DBRC sets the value of the LOG parameter to UNDO (meaning that the data sets are recoverable) and sets the ACCESS= field in the RECON header to PARALLEL. When parallel RECON access is disabled, DBRC sets the value of the LOG parameter to NONE and sets the ACCESS= field in the RECON header to SERIAL. You should specify the appropriate value for the LOG value whenever you delete and redefine a RECON data set. For example, if you are using parallel access, you should specify LOG(UNDO)

NAME
: Defines the cluster's entry name and is required.

NONSPANNED
: See RECORDSIZE.

RECORDSIZE
: DBRC always writes physical VSAM records that are less than the CI size in length, even though the logical RECON records can be a long as 16 MB.

  Recommendation: Set the maximum record size to be, at most, equal to the CI size minus seven bytes and ensure that NONSPANNED is specified. For example, if the CI size was defined as 8192 bytes, then RECORDSIZE (4086,8185) is appropriate.

SHAREOPTIONS
: SHAREOPTIONS(3 3) must be specified. The first value is required with single-host processors. Both values are required with multiple-host processors.

SPEED
: This is recommended because the initial load is faster.

NOWRITECHECK
: Avoid using WRITECHECK. It can degrade the RECON data set I/O performance. Using dual RECON data sets eliminates the need for WRITECHECK.

Recommendation: Do not use authorization keywords because frequent operator prompting results.

Security considerations for the RECON data sets

Jobs that update the RECON data sets must have UPDATE authority to the RECON data sets.

For jobs that require read-only access, READ authority should be used. With parallel access, the job that issues the **CHANGE.RECON ACCESS** command must have ALTER authority. When parallel access is being used, the online-IMS DBRC regions should have ALTER authority.

For more information about preserving the integrity of the RECON data sets, see Chapter 43, "DBRC security," on page 523.

Initially accessing the RECON data sets

When a job needs to read the RECON data set, the RECON data set must be opened. When three RECON data sets exist, DBRC determines which two are the active RECON data sets and which is the spare data set.

The following table shows how this determination is made. The columns in the table have the following meanings:

- In the DD statement column, RECONA, RECONB, and RECONC represent the RECON1, RECON2, and RECON3 DD statements in no particular sequence.
- The data set status column indicates the status of the data set during open time.

- The data set use column indicates how DBRC assigns the data set.

Table 51. Determining which RECON data sets are accessed

Case	DD statement	Data set status	Data set use	DBRC selection criteria
1	RECONA	Create Mode	Copy1	Not selected until **INIT.RECON** command is specified
	RECONB	Create Mode	Copy2	Not selected until **INIT.RECON** command is specified
	RECONC	Create Mode	Spare	
2	RECONA	RECON	Copy1	Current copy of RECON
	RECONB	Create Mode	Copy2	Produced from Copy1
	RECONC	Create Mode	Spare	
3	RECONA	RECON	Copy1	Current copy of RECON
	RECONB	RECON	Copy2	Current copy of RECON
	RECONC	Create Mode	Spare	
4	RECONA	RECON	Copy1	Current copy of RECON
	RECONB	RECON	Unused	Older copy than RECONA
	RECONC	Create Mode	Copy2	Produced from Copy1
5	RECONA	RECON	Copy1	Current copy of RECON
	RECONB	RECON	Unused	Older copy than RECONA
	RECONC	RECON	Unused	Older copy than RECONA
6	RECONA	RECON	Copy1	Current copy of RECON
	RECONB	RECON	Copy2	Current copy of RECON
	RECONC	RECON	Unused	Older copy than RECONA
7	RECONA	RECON	Copy1	Current copy of RECON
	RECONB	Create Mode	Copy2	Produced from Copy1
8	RECONA	Create Mode	Copy1	Not selected until **INIT.RECON** command is specified
	RECONB	Create Mode	Copy2	Not selected until **INIT.RECON** command is specified
9	RECONA	RECON	Copy1	Current copy of RECON
	RECONB	RECON	Copy2	Current copy of RECON
10	RECONA	RECON	Copy1	Current copy of RECON
	RECONB	RECON	Unused	Older copy than RECONA
11	RECONA	Create Mode	None	Discontinue processing
12	RECONA	RECON	Copy1	Current copy of RECON

Case 4 is the situation where two RECON data sets are available, but one is now out of date. DBRC does not use the out-of-date RECON data set. Instead, it copies the up-to-date RECON data set to the spare data set.

Only one RECON data set is available in cases 5, 10, and 12. If you have specified the STARTNEW parameter of the **INIT.RECON** or **CHANGE.RECON** command, processing continues with one RECON data set. Otherwise, processing ends. If another DBRC instance is active and parallel RECON access is enabled, the new DBRC must be able to access the same data sets (both in name and active count) as the existing DBRC regardless of the STARTNEW parameter. If the new DBRC does not access the same data sets, a WTOR will be issued for online DBRC regions and processing will end for non-online DBRC regions.

Records in the RECON data sets

The RECON data set contains many types of records. Some records, such as header records, exist primarily to control processing of the RECON data set. Other records exist to define the various data sets used in the recovery of DBDSs. Still others exist to record events related to the use of DBDSs.

The following figure shows the major relationships between RECON record types. These relationships are described in detail in the topics that follow the figure.

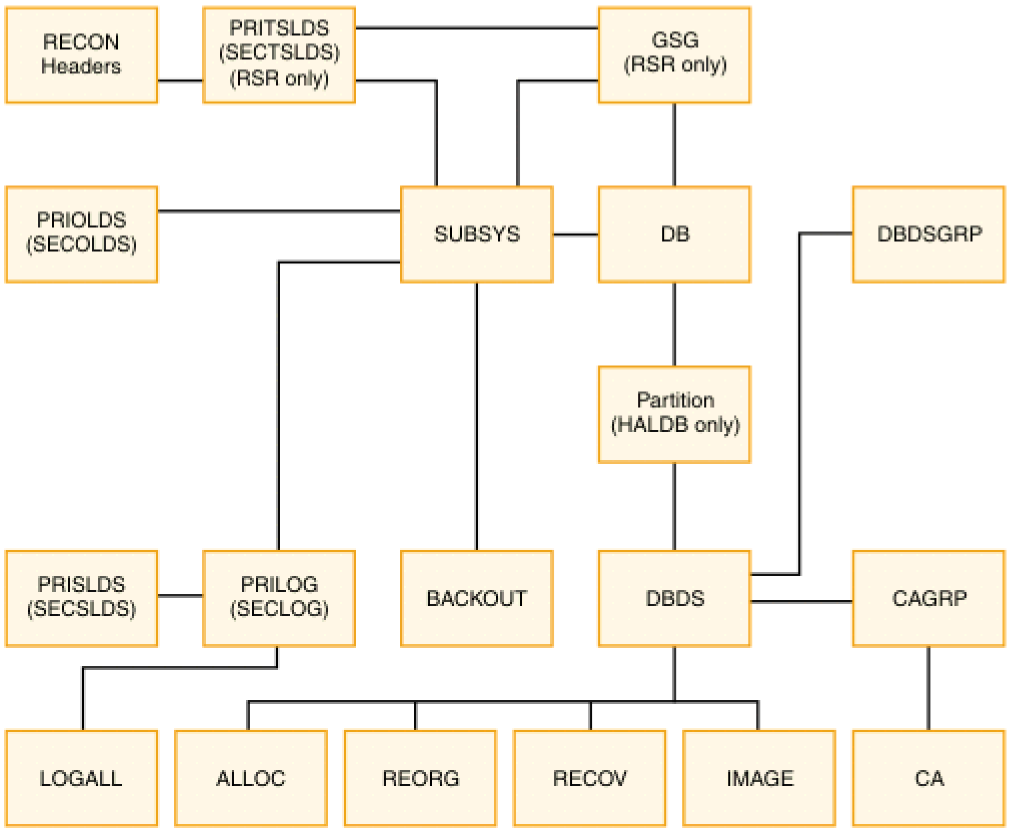


Figure 62. Major RECON record types and the relationships between them

Related reading:

- See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for details of what the records contain.
- See "Database Recovery Control service aids" in *IMS Version 15.2 Diagnosis* for information about RECON keys.

RECON header records

The RECON data set has two record types that contain control information DBRC uses in its processing.

- RECON header

In addition to internal control information that DBRC creates and uses, the RECON header record contains information provided by you in the parameters of the **INIT.RECON** and **CHANGE.RECON** commands.

- RECON header extension

  The RECON header extension record contains RECON data set configuration and state data that DBRC uses to process the RECON data sets.

Log data set records

Three types of log data sets exist. Each type can have a primary log record, secondary log record, interim-primary log record, and interim-secondary log record.

The log data set records for each log data set are:

- Recovery log data set
  - PRILOG (primary log record)
  - SECLOG (secondary log record)
  - IPRI (interim primary log record)
  - ISEC (interim secondary log record)
- System log data set
  - PRISLD (primary system log data set)
  - SECSLD (secondary system log data set)
  - IPRISL (interim primary system log data set)
  - ISECSL (interim secondary system log data set)
- Online log data set
  - PRIOLDS (primary online log data set)
  - SECOLDS (secondary online log data set)
  - IPRIOL (interim primary online log data set)
  - ISECOL (interim secondary online log data set)

Log records come in sets called *PRILOG families*. A PRILOG family consists of a PRILOG and one or more of the following: SECLOG, PRISLD, and SECSLD for a given time period and IMS subsystem. All records in this set have the same start and end times and normally have matching data set entries. The same LOGALL record applies to all members of the set.

DBRC creates the PRILOG and PRISLD records whenever an online IMS opens the first OLDS, and updates them each time an OLDS is archived. If you use dual archiving, DBRC creates SECLOG and SECSLD records when the first OLDS is archived and updates them each time an OLDS is archived.

Related reading: See “Archiving an OLDS” on page 478 for more information about DBRC and the archiving of online logs.

Log data sets output from IMS batch jobs are recorded in PRILOG / SECLOG records even though, technically, they are SLDSs. These records are created whenever the output log is opened and updated when volume switches occur.

In addition, during log recovery processing, DBRC creates an IPRISL or IPRIOL record for each interim primary-log data set and an interim secondary-log record for each interim secondary-log data set whenever the Log Recovery utility runs. An interim log record is an internal record that is used to reflect intermediate processing of the DUP function of the Log Recovery utility.

Related reading: See "Log Recovery utility (DFSULTR0)" in *IMS Version 15.2 System Utilities* for more information about interim primary and interim secondary-log data sets.

Although not part of normal DBRC operation, you can use the following commands to create log records (for example, to set up a test environment or for RECON data set repair purposes):

- NOTIFY.PRILOG
- NOTIFY.SECLOG

Database recovery records

In addition to the log data set records, the RECON data set contains records with database recovery information in them.

Related reading: See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for a listing of the records in the RECON data set and tables that describe the fields in the RECON data set records.

Backout record (BACKOUT)

A BACKOUT record contains information about units of recovery including time stamp, associated PSB name, recovery token, and database name. The names of nonrecoverable databases are not stored in the BACKOUT record.

Change accumulation group record (CAGRP)

A CAGRP record identifies a change accumulation (CA) group. This record includes up to 32,767 names of DBDSs whose change records are accumulated during one run of the Database Change Accumulation utility. Each DBDS (for which DBRC is controlling recovery) can be a member of only one CA group in order for the Database Change Accumulation utility to accumulate its changes. You specify the CAGRP name in the **INIT.CAGRP** command that you use to create a CAGRP record in the RECON data set.

The CAGRP record contains the name of a member of a partitioned data set. This member contains the skeletal JCL that is used to generate the JCL to run the Database Change Accumulation utility for this CA group. The CAGRP record also contains an indicator that specifies whether change accumulation data sets that correspond to this group can be reused. It contains an indication of the maximum number of change accumulation data sets that are maintained for the group and the recovery period.

Change accumulation run record (CA)

A CA record contains information about a change accumulation data set and can be either available or in use. For each CAGRP record, there can be up to 1024 CA records.

An available CA record is created by an **INIT.CA** command. This CA record contains the volume serial numbers and the data set name of a data set that is to be used for output from a subsequent run of the Database Change Accumulation utility. You can create available CA records only for those CA groups that are defined with the REUSE parameter.

An in-use change accumulation record is created by a run of the Database Change Accumulation utility. It can be a formerly available CA record that was used during a run of the Database Change Accumulation utility, or it can be a new record with information obtained from the JCL. The information in an in-use change accumulation record includes:

- Data set name
- Volume serial numbers
- Run time of the Change Accumulation utility
- Stop time of the last log volume that the Database Change Accumulation utility processed; or, if the CA processed a subset of logs, the start time of the first log that should be processed on the next execution

You can use a **NOTIFY.CA** command to create a CA record.

Data group record (DBGRP, DBDSGRP, RECOVGRP)

You can use a DBDSGRP record to define the following types of named groups: DBDSGRP, DBGRP, and RECOVGRP.

The named groups are:

- DBDSGRP, a group of DBDSs and DEDB areas
- DBGRP, a group of DL/I DBs and DEDB areas that can be named in the DATAGROUP parameter for the **/STA**, **/STO**, and **/DBR** commands
- RECOVGRP, a group of DL/I DBs and DEDB areas that are logically related for database recovery purposes

All groups must have unique names. For example, a DBDSGRP cannot have the same name as a DBGRP.

You use the **INIT.DBDSGRP**, **CHANGE.DBDSGRP**, **DELETE.DBDSGRP**, and **LIST.DBDSGRP** commands to manipulate all three data group types.

Recommendation: Although any type of group can be named in the DATAGROUP parameter for the **/STA**, **/STO**, and**/DBR** commands, the use of a DBDSGRP is not recommended because it is inefficient.

Related reading: See "DBDS group considerations" on page 517 for more information on DBDS groups.

Database records (DB)

DBRC treats DL/I, Fast Path, and HALDB (High Availability Large Database) database records differently.

These types of records, their treatment, and contents are explained as follows:

DL/I database records

A DB record identifies a database that is registered and whose recovery is under the control of DBRC. This record contains information about the database and related recovery information including:

- Database name
- Database type
- Share level of database
- List of subsystems using the database
- Extended Error Queue Element (EEQE) counter
- IRLM identification of the first subsystem that authorized the database (if IRLM is used)

A DBDS record identifies a DBDS whose recovery DBRC is to control. This record contains information about the DBDS (such as its data set organization) and related recovery information including:

- Name of the CA group to which the DBDS belongs
- Maximum number of image copy data sets to be maintained for this DBDS
- Indication of whether image copy data sets are to be reused
- Period of time that image copy data sets are to be maintained for this DBDS
- Name of the implied skeletal JCL default member
- Extended Error Queue Elements (EEQEs)
- Names of the members of the partitioned data set of skeletal JCL to be used in order to generate JCL for utilities that are run for this DBDS

To describe DL/I databases and DBDSs, DBRC maintains logically related records, as shown in the following figure.

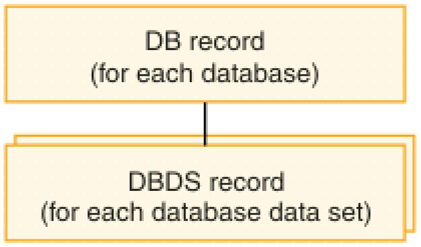


Figure 63. DBRC DL/I records

HALDB records

From a RECON data set perspective, HALDBs consist of a HALDB master (TYPE=HALDB) and one or more HALDB partitions (TYPE=PART).

The following figure shows the relationship of the RECON data set records that represent a HALDB.

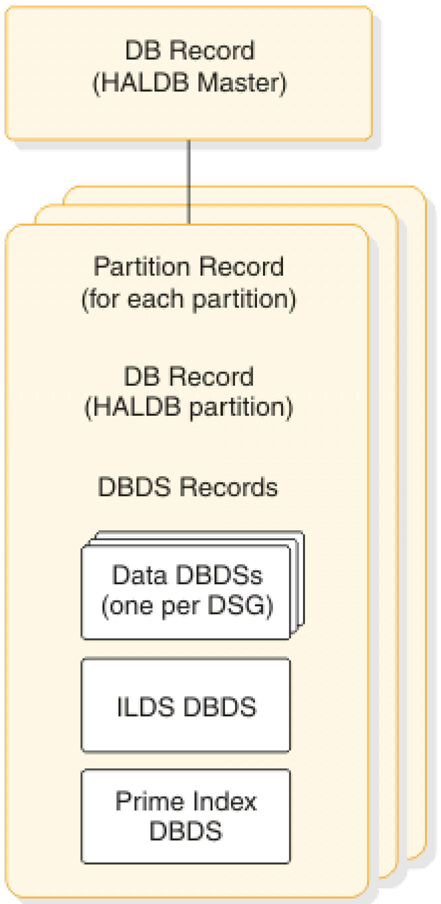


Figure 64. Record structure for a HALDB in the RECON data sets

HALDB Master (DSPDBHRC)

The RECON data set stores information pertaining to the entire HALDB by using a DB header record. The DB header record includes the following:

- HALDB master name
- Global DMB number
- Current change version number
- TYPE=HALDB
- Data sharing level
- Recovery group
- HALDB Partition Selection exit routine name
- Database and data set organization
- Maximum data set size for OSAM data sets
- Partition ID of the latest partition defined
- Recoverability of the HALDB
- Number of partitions in the HALDB
- Capability for online reorganization

- Number of data set group members

The TYPE=HALDB DB record stores information about the HALDB. Applications conduct DB activity at the HALDB master level; DBRC conducts the DB activity at the HALDB partition level. A subsystem authorizes a HALDB partition, not a HALDB master.

HALDB Partition

Each partition of the HALDB consists of the following RECON data set records:

- Partition record (DSPPTNRC)

  DSPPTNRC contains information that applies to the individual partition. The HALDB Partition Definition utility displays the partition record information. The `LIST` command does not display the partition record.

- Partition DB record (DSPDBHRC)

  DSPDBHRC accesses the HALDB at the partition level. Like the TYPE=IMS DB record, the DB record for the HALDB partition records all sharing and recovery information. The partition name sets the database name field in this record. TYPE=PART has been defined for this record. The following fields have the same settings for each partition across the entire HALDB:

  – Global DMB number
  – Data sharing level
  – Recoverability of the partitions
  – HALDB master name
  – Capability for online reorganization

- Partition DBDS records (DSPDSHRC)

  Depending on the organization of the HALDB, there can be three types of DBDSs for each HALDB partition: data, index, and ILDS data sets. Multiple data DBDSs can exist, but only one of each of the others. Only data DBDSs can be recovered. The other DBDSs are rebuilt using the HALDB Index/ILDS Rebuild utility (DFSPREC0).

  Related reading: See *IMS Version 15.2 Database Administration* for information about the data set and DDN naming conventions for DBDS records.

Fast Path database records

To describe DEDBs, areas, and Area Data Sets (ADSs), DBRC has a logical structure of records, as shown in the following figure.

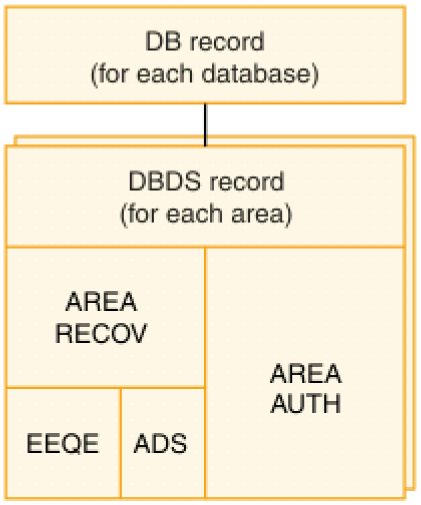


Figure 65. DBRC Fast Path database records

DBRC uses the DB and DBDS records to describe both DL/I databases and DEDBs; however, DBRC adds an ADS list to the Fast Path DBDS record giving information about each ADS. Each DEDB may contain multiple areas, and each area may contain up to seven ADSs.

The Fast Path DB record contains information similar to the information in a DL/I DB record, except that it describes a DEDB; and it does not contain a list of authorized subsystems. For Fast Path, this list is in the DBDS record, which is composed of an area authorization record and an area recovery record. The Fast Path DB record is displayed in the listing as the DBDS record.

Related reading: See the sample RECON listing in *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands*.

When an area is registered in the RECON data set, it ensures that:

- The area names in the DEDB are unique
- The ADS DD names in an area are unique
- No more than seven ADSs are defined for an area

Image copy record (IMAGE)

An IMAGE record contains information about an image copy data set and can be either *available* or *in-use*.

An available IMAGE record is created by an **`INIT.IC`** command. It describes data sets that are to be used for output from a subsequent run of an image copy utility. You can create available IMAGE records only for those DBDSs that are defined with the REUSE parameter.

An in-use IMAGE record is created by an execution of an image copy utility:

- If the DBDS or area is defined with REUSE, it is a formerly available IMAGE record or a reused in-use record.
- If the DBDS or area is defined with NOREUSE, it is a new record with the data set description that is obtained from the JCL.

In addition to the data set description, the in-use IMAGE record identifies the type of copy operation and contains the copy operation's run time and, depending on the type of copy, the copy operation's stop time.

If you request that the image copy utility create an image copy data set and a duplicate image copy data set, DBRC records information about both in the same image copy record. The first record is designated IC1, and the duplicate is designated IC2.

If you create a nonstandard image copy data set (one that the image copy utility did not create), you must use a **`NOTIFY.UIC`** command to record its existence in the RECON data set.

A nonstandard image copy, also known as a user image copy, is created automatically for each area altered by the Fast Path DEDB Alter utility. This is done by using the shadow image copy data set (IDS).

Related Command: See "Image copy utilities (DFSUDMP0, DFSUDMT0, DFSUICP0)" on page 505 for more information on the IMAGE COPY record.

Reorganization record (REORG)

DBRC creates a REORG record each time you reorganize a data set of a registered database or partition by using one of the following methods.

- DEDB Alter utility (ALTRAREA function only)
- HALDB Online Reorganization
- HALDB alter reorganization
- The HISAM Reorganization Reload utility
- The HD Reorganization Reload utility

For a HALDB Online Reorganization, the REORG record indicates that the type of REORG is online, includes a stop time, and indicates if the reorganized data set can be used for recovery. DBRC creates a REORG record in the RECON data set for each database data set that you reorganize.

The REORG records that are created by the HALDB alter function are the same as the REORG records for a HALDB Online Reorganization, except for the addition of a flag that indicates that the records were created for HALDB alter.

Similarly, a REORG record is created for each area that is altered by the Fast Path DEDB Alter utility.

Related concepts

HALDB online reorganization (Database Administration)

Related tasks

Altering the definition of an online HALDB database (Database Administration)

Related reference

HD Reorganization Reload utility (DFSURGL0) (Database Utilities)

HISAM Reorganization Reload utility (DFSURRL0) (Database Utilities)

Log allocation record (LOGALL)

A LOGALL record identifies the registered DBDSs that were changed while the corresponding log data set was open.

DBRC creates a log allocation (LOGALL) record for each PRILOG record.

Database allocation record (ALLOC)

For Fast Path, DBRC creates an ALLOC record when the area is placed in OPEN-for-update status. For DL/I, DBRC creates an ALLOC record when IMS updates a DBDS the first time during a run of IMS or the first time after you enter a **/DBRECOVERY** command. This record contains the time stamp of its creation and the time stamp of the opening of the corresponding log. This time stamp identifies the log data sets containing the database change records that are needed for recovery. If the DBDS is subsequently deallocated, DBRC adds the time stamp of the deallocation to the ALLOC record.

DBRC automatically deletes allocation records if they are older than the oldest image copy record and if DBRC no longer needs them for database recovery. As DBRC deletes an ALLOC record, it changes the associated LOGALL record. This is part of the DBRC Image Copy utility exit routine processing. This automatic deletion of ALLOC records for a DBDS does not occur under either of the following conditions:

- The ALLOC record has no deallocation time.

  A deallocation time is recorded when the database or area has had the **/DBR** command run against it. Otherwise, the ALLOC record uses the log close time as an implicit deallocation time.
- The PRILOG record associated with the ALLOC record is open (it has a STOPTIME of zero) but is not marked in error.

In cases such as these, you can use the **DELETE.ALLOC** command to delete unwanted ALLOC records from the RECON data set. DBRC automatically updates allocation time stamps during online and concurrent image copy utility exit routine processing to move the allocation time stamps forward in time.

Recovery record (RECOV)

DBRC creates a RECOV record each time you run the Database Recovery utility to recover a DBDS.

The RECOV record can indicate one of these types of recovery:

- A full recovery of a DBDS where the RECOV record contains the time stamp of the recovery.
- A time-stamp recovery where the RECOV record contains the time stamp of the run of the Database Recovery utility and the time stamp to which the DBDS was recovered.

Subsystem records (SSYS)

The RECON data set uses the subsystem (SSYS) record to describe data sharing information.

An SSYS record is created when an IMS subsystem signs on to DBRC. This SSYS record contains information about the subsystem and related recovery information including:

- Subsystem name and type (online or batch)
- IRLM identification
- Abnormal-end flag and the recovery-process start flag
- List of authorized databases
- Time stamp that correlates the subsystem entry with the appropriate log records
- A flag indicating whether this is a BPE-based subsystem

Mapping the records in the RECON data set

You can map DBRC records by assembling mapping macro members that are in the ADFSMAC and SDFSMAC libraries.

Member (DSECT) name	RECON record that the DSECT maps
DSPALLRC	ALLOC record
DSPBKORC	BACKOUT record
DSPCAGRC	Change accumulation group (CAGRP) record
DSPCHGRC	Change accumulation (CA) record
DSPDBHRC	Database (DB) record
DSPDGRC	DBDSGRP record
DSPDSHRC	DBDS record
DSPIMGRC	Image copy (IC) record
DSPLGARC	LOGALL record
DSPLOGRC	PRILOG / SECLOG / PRISLD / SECSLD record
DSPOLDRC	PRIOLD / SECOLD record
DSPPTNRC	HALDB partition record
DSPRCNRC	HEADER record
DSPRCR1	HEADEREXT record
DSPRCVRC	RECOV record
DSPRRGRC	REORG record
DSPRDTRC	RECON DMB table record
DSPSSRC	SUBSYS record

Related reference

RECON record types (Diagnosis)

Maintaining the RECON data sets

You should perform periodic maintenance on the RECON data set to maintain data integrity and an acceptable level of performance.

The following topics discuss the main tasks involved.

Backing up the RECON data sets

Back up the RECON data sets frequently. They are a critical resource. Always make backup copies of the RECON data set after performing any RECON record maintenance, such as registering databases and adding or deleting change accumulation groups.

Use the **BACKUP.RECON** command to perform backup. This command issues the necessary **RESERVE** commands (reserving the device during backup processing) to ensure backup integrity. Then it invokes the AMS **REPRO** command to copy the data set. The **BACKUP.RECON** command copies only the first copy of the RECON data set. Its parameters determine whether it makes one or two copies. In parallel access mode, no reserve commands are performed. Instead RECON activity is quiesced for all active DBRC instances.

If parallel access is enabled, the user has the option to take a backup copy of a Transactional VSAM (TVS) RECON using DFSMSdss COPY or DUMP. When DFSMSdss COPY or DUMP is used, TVS automatically quiesces the data set so that a consistent or "sharp" backup copy is produced.

Related reading: See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for additional information on the **BACKUP.RECON** command.

Deleting unnecessary RECON records

You can delete unnecessary RECON records using the following methods.

Related concepts

"Recovery period of image copy data sets and GENMAX" on page 510

When you register a DBDS in the RECON data set, you specify the maximum number of image copy generations for DBRC to record with the GENMAX parameter of the **INIT.DBDS** command. When this number is exceeded, DBRC discards the information relating to the oldest image copy.

Automatic deletion of extraneous records

Normally you do not need to perform much record maintenance for database-related records.

When the RECON data set is notified of an image copy, it may delete or reuse the oldest in-use IMAGE record, and a later IMAGE record becomes the oldest IMAGE record. RECOV and REORG records with start times earlier than the (now) oldest IMAGE record, and ALLOC records with DEALLOC times earlier than the earliest time, are now extraneous, and are deleted from the RECON data set. This is the image copy cleanup process.

When extraneous IMAGE records are deleted from the RECON data set, all active ALLOC records are updated to the time of the first log volume necessary for recovery, based on the oldest image copy for the DBDS or area. When the cleanup process deletes an extraneous ALLOC record, it changes the state of the associated LOGALL records. After all the ALLOC records associated with the LOGALL record have been deleted (this may take place over many image copies for many databases), the PRILOG record associated with the LOGALL record becomes inactive.

PRILOG compression

PRILOG record compression is the deletion of all inactive data set entries in an open PRILOG record.

A data set entry is defined as being inactive when it is older than all of the following criteria:

- Log retention period
- Oldest allocation (ALLOC) for any database updated on that log
- Earliest restart checkpoint for the online IMS

PRILOG record compression deletes inactive data set entries up to the oldest ALLOC on the log or the first gap in the data set entries. A gap occurs when an OLDS has not yet been archived.

When inactive data set entries are deleted from active PRILOGs, they are compressed to a single dummy data set entry that has the same start time as the start time of the log and the same stop time as the stop time of the last inactive data set entry deleted.

PRILOG compression can be performed using the following two methods:

Automatic PRILOG compression
PRILOG record compression is attempted automatically after an OLDS is archived.

Compression includes the deletion of all inactive data set entries in the PRILOG record. When applicable, corresponding entries in the SECLOG, PRISLD, and SECSLD records are also deleted.

Manual process for PRILOG compression
You can initiate PRILOG record compression manually by using the **`DELETE.LOG INACTIVE`** command. This command deletes inactive data set entries from active PRILOG records and deletes entire inactive PRILOG records.

If your PRILOG is not getting compressed, you will receive message DSP1150I for one of the following reasons:

- EARLIEST CHECKPOINT

  If this time stamp is within the first DSN entry in the PRILOG, then compression is not possible. Most often, the oldest checkpoint needed for restart causing the problem is the checkpoint to rebuild the message queues. If so, take a SNAPQ checkpoint. When the OLDS that records this SNAPQ checkpoint is archived, the earliest checkpoint time stamp is updated.
- EARLIEST ALLOC TIME

  The databases listed in the LOGALL record are in earliest allocation time stamp order. The earliest allocation time listed in the LOGALL record is associated with the first databases in the listing.

  Concurrent image copies (CICs) and Online image copies (OICs) update the allocation time to reflect the earliest log that is needed for recovery. The checkpoint IDs and checkpoint counts on the logs before the start of the CIC are used to determine where the new allocation time is set. If the allocation times are not being moved forward-in-time, ensure that checkpoint IDs and counts are being recorded in the DSN entries.

  Old allocations for any databases in the LOGALL record can cause compression to fail. Image copy databases on a regular basis to allow old allocations to be deleted.
- LOG RETENTION TIME

  Indicates the minimum amount of time that DBRC is to keep log records in the RECON data set.

Manual log record deletion

You can manually delete log records by using the **`DELETE.LOG INACTIVE`** or the **`CLEANUP. RECON`** command.

DBRC does not automatically delete RECON records that describe log data sets (PRILOG and SECLOG records). This design gives you control over which RECON records associated with log data sets are deleted. You must periodically delete PRILOG and SECLOG records that are no longer needed for recovery.

The current size of the PRILOG record is printed on the listing of the PRILOG record.

Use the **`DELETE.LOG INACTIVE`** command to delete inactive PRILOG records, SECLOG records, and their associated LOGALL records.

Common problems are old open (nonzero stop time) PRILOG records that were created by jobs that terminated abnormally (without closing their logs). You can find these open logs by issuing the **LIST.LOG** command with the OPEN parameter. You can use the **DELETE.LOG** command with the STARTIME parameter to remove old, unnecessary, open PRILOG family records.

Alternatively, use the **CLEANUP.RECON** command to unconditionally delete old or expired log information, CA data sets, and recovery-related information (image copy, allocation, reorganization, and recovery records) from the RECON data set. Using the specified time stamp or retention period, recovery-related data and inactive logs are deleted and PRILOG compression is forced on both open logs and closed logs that have a close time after this time.

Deleting log records does not prevent the RECON data sets from filling up because the space freed by deletions might not be reused by VSAM. However, if you delete the log records before you back up or reorganize the RECON data set, you can reclaim the space during backup or reorganization.

Attention: To avoid deleting necessary recovery and log information from the RECON data set, run the CLEANUP.RECON command on a copy of the RECON data set first and verify the results, which indicate the length of time that is needed by the job. You can then use this information to determine when to run this command on your active RECON data set.

Related reference

CLEANUP.RECON command (Commands)

DELETE.LOG command (for OLDS) (Commands)

DELETE.LOG command (for RLDS and SLDS) (Commands)

LIST.LOG command (for a PRILOG family) (Commands)

Reorganizing the RECON data sets

You need to reorganize the RECON data sets periodically. Many of the record keys in the RECON data set include the date and time. DBRC recording of IMS log and database activity can cause CI and CA splits, that can degrade performance.

In addition, deleting unnecessary records might not prevent the RECON data set from filling up because VSAM does not always reuse the space freed.

Using CHANGE.RECON to reorganize a RECON data set

You can reorganize a RECON data set online if you are using dynamic allocation for the RECON data set by using the **CHANGE.RECON** command. A spare RECON data set must be available.

In this situation, you can issue the **CHANGE.RECON** command with the REPLACE option. This causes DBRC to copy and reorganize the active RECON data set (specified on the **CHANGE** command) to the spare data set. VSAM removes all CI and CA splits, and restores the original FREESPACE attributes.

The **CHANGE** command also deallocates the old RECON data set (the one that needed reorganization). Before you can delete and redefine this data set, however, you must wait for all other subsystems that are using it to deallocate it. If you redefine the data set with the same name it originally had, it is available to the online system for use as a spare data set. You can repeat this process to reorganize the second active RECON data set. If you use JCL in order to allocate data sets, dynamic deallocation does not occur.

If you do not use dynamic allocation or if a spare RECON data set is not available, you must wait for the online subsystem and all other subsystems that access the RECON data set to deallocate it before you can reorganize.

Backing up RECON data sets before reorganization

Back up the RECON data sets before and after reorganizing, using the following procedure.

1. Copy the RECON data sets to temporary data sets with different data set names.
2. Verify that the copied data sets are uncorrupted.
3. Delete the original data sets.

4. Redefine the original data sets using the same data set names.
5. Copy the temporary data sets back to these original data sets.

Procedure for reorganizing the RECON data sets

To clean up your logs and reorganize your RECON data sets, follow this procedure. This process assumes that you are running DBRC in dual mode with RECON1 (as Copy1), RECON2 (as Copy2), and a spare.

1. Issue a **`LIST.LOG OPEN`** command to identify any open logs. Determine which of the identified logs should be open, and which should be closed or deleted. Close any logs that should be closed or deleted.
2. Issue a **`DELETE.LOG INACTIVE`** command to delete any inactive or unused logs. You are now ready for the RECON data set reorganization.
3. Issue a **`CHANGE.RECON`** with the REPLACE option for RECON1. DBRC then:
   a) Signals the start of the RECON data set reconfiguration with message DSP0380I
   b) Identifies the subsystems that are active at reconfiguration with message DSP0388I
   c) Signals that the process has completed with message DSP0381I.
4. Replace the discarded RECON data set. See "Replacing a discarded RECON data set" on page 553 for more information.

Your RECON data sets have now been reorganized and the logs are now clean.

Replacing damaged RECON data sets

If an I/O error occurs in the current RECON data set that is the only one remaining, DBRC stops the job. Any other jobs that are currently using the RECON data set continue to run if no other I/O error is encountered.

If an I/O error occurs in a RECON data set and two RECON data sets exist, DBRC attempts to locate a spare data set. If a spare is available, DBRC copies the RECON data set without the I/O error to the spare RECON data set. DBRC then establishes the spare as the Copy2 RECON data set.

Recommendation: After the spare RECON data set replaces the RECON data set that had the error, redefine the discarded RECON data set as quickly as possible. If you immediately replace the RECON with the I/O error, you are unlikely to experience a subsystem failure due to loss of all RECON data sets. See "Replacing a discarded RECON data set" on page 553 for more information.

If DBRC cannot locate a spare RECON data set and you have specified the STARTNEW parameter of the **`INIT.RECON`** command, DBRC continues processing with one RECON data set. Otherwise, DBRC completes the current job but does not start new jobs until you define a spare RECON data set.

Recovering the RECON data sets

Steps for recovering your RECON data set depend on the set up of your RECON data set and the situation.

The following paragraphs discuss various scenarios for recovering a RECON data set.

A Spare RECON data set is available

If an I/O error occurs on a RECON data set and a spare data set is available, DBRC copies the good RECON data set to the spare, and then activates the spare.

If, however, you want to analyze the RECON data set error, before deleting and redefining the discarded RECON data set, make a copy of it for later problem diagnosis.

A Spare RECON data set is not available

If a spare RECON data set is not available, all currently executing jobs continue processing using the RECON data set in single mode. If you specified the STARTNEW parameter in the **`INIT.RECON`** or **`CHANGE.RECON`** command, DBRC allows new jobs to start with only one RECON data set. This is not recommended as it jeopardizes the integrity of the system.

If one of the data sets in the set of RECON data sets becomes unusable by DBRC, you need to deallocate the RECON data set that is unusable and allocate a new spare.

Both RECON data sets are not usable:
It is unlikely that both RECON data sets would be not usable. If, however, both RECON data sets ever become unusable, follow this procedure:

1. Stop all jobs that require access to the RECON data set.
2. Optional: Use the AMS **REPRO** command to back up the RECON data sets if you can access both of them.
3. Use the AMS utility to delete and redefine your RECON data sets.
4. Use the AMS **REPRO** command to restore one of the RECON data sets.
5. Use the AMS **REPRO** command to restore the other RECON data set from the first.
6. Use the **LIST.RECON** command to list one of the RECON data sets. Evaluate the list and determine which DBDSs have been updated since you made the backup in step "2" on page 553. If you cannot determine which DBDSs have been updated, assume that all have been updated.
7. Use the **CHANGE.IC** command with the INVALID parameter to mark all image copy records that are in error for all applicable DBDSs in step "6" on page 553.
8. Make an image copy of all applicable DBDSs in step "6" on page 553.
9. Use the **BACKUP.RECON** command to make a backup copy of the RECON data sets.

The RECON data sets are now restored and resynchronized with the databases.

If you do not control an excessive number of databases, it might be easier to follow this procedure:

1. Stop all jobs that require access to the RECON data set.
2. Define new RECON data sets.
3. Initialize these RECON data sets.
4. Register the environment. Always keep a backup copy of the most recently initialized, but not-yet-used, RECON data set available.
5. Take image copies of all databases.

Finally, before you proceed with your regular operations, clean up the new RECON data set by, for example, closing any open, out-of-date OLDSs with the **NOTIFY.PRILOG** command.

Replacing a discarded RECON data set

DBRC detects that a RECON data set is discarded only when some activity occurs that causes DBRC to access the RECON data sets. You can have multiple instances of DBRC whenever you have multiple IMS subsystems, online or batch. You cannot delete and redefine a discarded RECON data set until all instances of DBRC detect that a change has occurred and they deallocate the discarded data set.

DBRC lists the subsystems that are active at reconfiguration in message DSP0388I. This message enables you to identify the subsystems that might need your help in detecting the status change of the RECON data sets.

To redefine a RECON data set after an I/O error has occurred, or in conjunction with the **CHANGE.RECON REPLACE** command, follow this procedure:

1. Allow all batch jobs using DBRC to finish.
2. Issue **LIST.RECON STATUS** in all online subsystems if you do not have Automatic Loss Notification active. Issuing the command causes the online subsystems to obtain the same Copy1 and Copy2 RECON data sets and to deallocate the discarded RECON data set. If you do have Automatic Loss Notification active, all subsystems are automatically notified to deallocate the discarded RECON data set. See "RECON loss notification" on page 555 for more information on RECON Loss Notification.
3. Use the AMS **DELETE** command to delete the discarded RECON data set.

4. Use the AMS **DEFINE** command to recreate the RECON data set as an empty VSAM KSDS data set. Use the same procedure that you used originally to create the RECON data set. See "Creating a RECON data set" on page 537.

Repairing the RECON data sets

You can fix specific problems that can occur in the RECON data sets to ensure data integrity.

Specific problems can arise in the RECON data sets that are often created outside the control of DBRC. In most cases, the problems are related to the DMB Table record and inconsistencies in the DMB number in the RECON header, database, and Fast Path area records. You can use the **REPAIR.RECON** command to rebuild or build the DMB Table record and fix inconsistent DMB numbers in the records.

The **REPAIR.RECON** command with the DMBTABLE keyword processes the entire DMB Table and all of the records that contain DMB numbers that are used to build or rebuild the DMB Table record. Use the command to make the following repairs:

- Correct inconsistent DMB numbers, which involves ensuring the following items:
  - The high-order bit in the RECON header, database, and Fast Path area records is on
  - The DMB numbers in the partition database records match the DMB numbers in the HALDB master records
  - The DMB numbers in the Fast Path area records match the DMB numbers in the DEDB records
- Update the number of registered databases in the RECON listing
- Rebuild the DMB Table record or build the DMB Table record if it does not exist

The **REPAIR.RECON** command with the DMBNUM keyword also finds and corrects inconsistent DMB numbers, but allows the scope to be limited to all database records, specific types of database records (HALDB, Fast Path, or IMS), a specific database record, or the RECON header.

- The ALL keyword causes all the database records to processed and ensures that the high-order bit is on in the in RECON header, database, and Fast Path area records, the DMB numbers in the partition database records match the DMB numbers in the HALDB master records, and the DMB numbers in the Fast Path area records match the DMB numbers in the DEDB records.
- If the RESET keyword is specified with the RECON keyword, the DMB# and LAST USED# fields are set to 32767 if the DMB Table record exists. When the next database is registered in the RECON data set and these fields are set to 32767, the search in the DMB Table record for the next available DMB number to assign starts at 1 instead of from the last DMB number added.
- The CHECK keyword determines whether any inconsistencies with the DMB numbers exist without fixing them. The UPDATE keyword both finds and correct errors with the DMB numbers.

DBRC can change the records to fix inconsistencies in the DMB numbers only if the database or area is unauthorized. When the command is processed, a message is issued for any database or area that is authorized. Unauthorize the identified databases or areas and then rerun the command.

Recommendations:

- The **REPAIR.RECON** command has the potential of impacting the availability of the RECON data set because of the number of records that might be read and changed. First, run **REPAIR.RECON** command on a backup copy of the RECON data set to assess the impact. Then, run the command when access to the actual RECON data set is minimal. For **REPAIR.RECON DMBNUM**, run the command first with the CHECK keyword on the backup copy, and, if any DMB numbers need to be fixed, then run the command with the UPDATE keyword on the actual RECON.
- Schedule the **REPAIR.RECON DMBNUM CHECKUP** to run at least once a year to ensure the integrity of your RECON data sets and run the command before upgrading the RECON. If errors are found, rerun the command with the UPDATE keyword to fix the errors.

Related reference

REPAIR.RECON command (Commands)

RECON loss notification

All DBRC instances that are allocated to the RECON data set, using the same IMSplex, are notified of errors, at the time the errors occur, through the IMS Structured Call Interface (SCI). Then, any DBRC allocated to the discarded RECON data set will deallocate.

In parallel RECON access, DBRC must ensure that no other DBRC instance can access the RECON data set during the reconfiguration process. At that point, all the other DBRCs re-access in parallel mode.

DBRC obtains the IMSplex name from the DBRC SCI Registration exit routine (DSPSCIX0), or from an EXEC parameter, IMSPLEX. Any attempt at accessing a RECON data set using a different IMSplex, or no IMSplex, is rejected and message DSP1136A is issued. Otherwise, RECON Loss Notification is automatically invoked.

If you want to change the IMSplex associated with a set of RECONs, use the IMSPLEX or NOPLEX parameter on the **`CHANGE.RECON`** command. To change an IMSplex-to-RECON association when a DBRC instance is active:

1. Wait for all DBRC activity on the current IMSplex to stop.
2. Submit a Database Recovery Control utility job to change the IMSplex name.

   If parallel access is enabled and you want to remove the IMSplex name from the RECON data set by using the **`CHANGE.RECON NOPLEX`** command, parallel access must be disabled first by using the **`CHANGE.RECON ACCESS(SERIAL)`** command.
3. Alter the IMSplex name in the DSPSCIX0 exit routine.
4. Ensure that the new IMSplex SCI is ready.

Related reading:

- See "IMSplex overview" on page 16 for introductory information about IMSplexes.
- See *IMS Version 15.2 Exit Routines* for more information on the DBRC SCI Registration exit routine (DSPSCIX0).
- See *IMS Version 15.2 System Definition* for more information about the IMSPLEX EXEC parameter.

Tracking changes made to the RECON data sets

For administrative purposes, it is a good practice to keep a log of the changes made to the RECON data set. You can provide your own exit routine which is to be called every time RECON records are changed or read.

This exit routine, also called the RECON I/O exit routine (DSPCEXT0), allows you to keep track of changes to the RECON data set in the form of a journal.

Related reading: See *IMS Version 15.2 Exit Routines* for detailed information about the RECON I/O exit routine.

Chapter 45. Hints and tips for DBRC

This topic provides task-oriented instructions for frequently-used procedures, including the following topics.

Changing the RECON data sets to output time stamps in local time of origin

If the listing of the RECON data set does not list the offset value for time stamps, the record might have been created when the offset value was different from the current offset value. Using DBRC commands to change or delete such a record will result in a "record-not-found" condition because the current offset is applied if it is not included in the time stamp parameter of the command.

To change the RECON data set to output time stamps in local time of origin, when they were created, use one of the following:

```
CHANGE.RECON TIMEFMT(O,O,P,4,PERM)
```

or

```
CHANGE.RECON TIMEFMT(O,O,P,2,PERM)
```

This will permanently change the RECON data set to output time stamps when the records were originally created with the appropriate offsets in punctuated four-digit or two-digit year formats. The time stamp listed can be used in commands such as the following:

```
CHANGE.PRILOG STARTIME('2007.063 16:11:58.123456 -08:00') -
 DSSTART('2007.063 16:11:58.123456 -08:00') ERROR RLDS
```

Locating the last SLDS stop time in the RECON data set

Use the following procedure to find the last SLDS stop time in the RECON data set using the **GENJCL.USER** command. This procedure can be used while IMS is still running and can also be used if the PRISLD is already closed.

1. Create a skeletal JCL execution member.

   Here is an example of a member named SELSLDS:

   ```
   %SELECT SLDS(%USSID,LAST)
   %ENDSEL
   SLDSDD  BEG=%SLDSTIM
           END=%SLDETIM
           VOL=%SLDVOLS
           UNT=%SLDUNIT
           DSN=%SLDSDSN
   ```

2. Issue the **GENJCL.USER** command indicating the execution member to run with.

   Below is a sample of the **GENJCL.USER** command indicating that the command is to be run with member SELSLDS:

   ```
   GENJCL.USER MEMBER(SELSLDS) NOJOB USERKEYS((%USSID,'IMS1')) -
     JCLOUT(SYSPRINT)
   ```

Below is the SLDS listing:

```
PRISLD                                       RECORD SIZE=    1104
  START = 07.064 16:58:06.000000         *    SSID=IMS1     VERSION=10.1
  STOP  = 07.064 17:00:04.656844              #DSN=6
  GSGNAME=**NULL**
```

```
   FIRST RECORD ID= 0000000000000001       PRILOG TOKEN= 0

   DSN=IMSVS.SLDSP.IMS1.D07064.T1658060.V00                 UNIT=SYSDA
   START = 07.064 16:58:06.000000           FIRST DS LSN= 0000000000000001
   STOP  = 07.064 16:58:16.108103           LAST  DS LSN= 0000000000000128
   FILE SEQ=0001    #VOLUMES=0001
   CHECKPOINT TYPES=80: SIMPLE=Y SNAPQ=N DUMPQ=N PURGE=N FREEZE=N

     VOLSER=000000 STOPTIME = 07.064 16:58:16.108103
       CKPTCT=1    CHKPT ID = 00.000 00:00:00.000000
       LOCK SEQUENCE#=  000000000000

   DSN=IMSVS.SLDSP.IMS1.D07064.T1658161.V00                 UNIT=SYSDA
   START = 07.064 16:58:16.108103           FIRST DS LSN= 0000000000000129
   STOP  = 07.064 16:59:00.886236           LAST  DS LSN= 0000000000000BE9
   FILE SEQ=0001    #VOLUMES=0001
   CHECKPOINT TYPES=00: SIMPLE=N SNAPQ=N DUMPQ=N PURGE=N FREEZE=N

     VOLSER=000000 STOPTIME = 07.064 16:59:00.886236
       CKPTCT=0    CHKPT ID = 07.064 16:58:06.603513
       LOCK SEQUENCE#=  000000000000

   DSN=IMSVS.SLDSP.IMS1.D07064.T1659008.V00                 UNIT=SYSDA
   START = 07.064 16:59:00.886236           FIRST DS LSN= 0000000000000BEA
   STOP  = 07.064 16:59:15.330532           LAST  DS LSN= 000000000000111A
   FILE SEQ=0001    #VOLUMES=0001
   CHECKPOINT TYPES=00: SIMPLE=N SNAPQ=N DUMPQ=N PURGE=N FREEZE=N

     VOLSER=000000 STOPTIME = 07.064 16:59:15.330532
       CKPTCT=0    CHKPT ID = 07.064 16:58:06.603513
       LOCK SEQUENCE#=  000000000000

   DSN=IMSVS.SLDSP.IMS1.D07064.T1659153.V00                 UNIT=SYSDA
   START = 07.064 16:59:15.330532           FIRST DS LSN= 000000000000111B
   STOP  = 07.064 16:59:28.219762           LAST  DS LSN= 00000000000016A5
   FILE SEQ=0001    #VOLUMES=0001
   CHECKPOINT TYPES=00: SIMPLE=N SNAPQ=N DUMPQ=N PURGE=N FREEZE=N

     VOLSER=000000 STOPTIME = 07.064 16:59:28.219762
       CKPTCT=0    CHKPT ID = 07.064 16:58:06.603513
       LOCK SEQUENCE#=  000000000000

 07.064 17:00:18.768831               LISTING OF RECON                  PAGE 0005
   DSN=IMSVS.SLDSP.IMS1.D07064.T1659282.V01                 UNIT=SYSDA
   START = 07.064 16:59:28.219762           FIRST DS LSN= 00000000000016A6
   STOP  = 07.064 17:00:04.440567           LAST  DS LSN= 0000000000001907
   FILE SEQ=0001    #VOLUMES=0001
   CHECKPOINT TYPES=08: SIMPLE=N SNAPQ=N DUMPQ=N PURGE=N FREEZE=Y

     VOLSER=000000 STOPTIME = 07.064 17:00:04.440567
       CKPTCT=1    CHKPT ID = 07.064 16:58:06.603513
       LOCK SEQUENCE#=  000000000000

   DSN=IMSVS.SLDSP.IMS1.D07064.T1659282.V02                 UNIT=SYSDA
   START = 07.064 17:00:04.440567           FIRST DS LSN= 0000000000001908
   STOP  = 07.064 17:00:04.656844           LAST  DS LSN= 00000000000019A0
   FILE SEQ=0001    #VOLUMES=0001
   CHECKPOINT TYPES=00: SIMPLE=N SNAPQ=N DUMPQ=N PURGE=N FREEZE=N

     VOLSER=000000 STOPTIME = 07.064 17:00:04.656844
       CKPTCT=0    CHKPT ID = 07.064 17:00:04.417809
       LOCK SEQUENCE#=  000000000000
```

Below is the output that results from issuing the previous sample **GENJCL.USER** command:

```
SLDSDD  BEG=07064170004440567-0800
        END=07064170004656844-0800
        VOL=000000
        UNT=SYSDA
        DSN=IMSVS.SLDSP.IMS1.D07064.T1659282.V02
```

Adjusting GENMAX when it is reached or it is too high

Prior to image copy utility execution, verification exit processing may determine that one of the following situations is true when image copy data sets are being reused. In such cases, the GENMAX value is either too high or too low. In either case, a command must be issued to correct the problem.

- Situation 1: GENMAX value is too low
  - The GENMAX value was reached.
  - The oldest image copy cannot be reused because it is within the recovery period.
  - No available image copies exist that can be used.
  - Message DSP0063I was displayed.
  - Increase GENMAX and define additional image copy data sets.
  - Change the recovery period to allow the oldest image copy data set to be used.

  Issue the **CHANGE.DBDS** command to increase the GENMAX and INIT.IC parameters to define the additional image copy data sets:

  1. Create a JCL job similar to one of the following jobs. These examples assume that your DBD name is THISDB, that your area name is AREA1, and that your previous GENMAX value was 2.

     ```
     //CHNGDBDS JOB

     //SYSIN DD *
          CHANGE.DBDS DBD(THISDB) AREA(AREA1) -
                      GENMAX(4)
     /*//INITIC JOB

     //SYSIN DD *
          INIT.IC DBD(THISDB) DDN(DDN1) -
                  ICDSN(IMS.*.NEWICDSN)
          INIT.IC DBD(THISDB) DDN(DDN1) -
                  ICDSN(IMS.*.NEWICDSN2)
     /*
     ```

     Issue **CHANGE.DBDS RECOVPD** to reduce the recovery period.

     ```
     //CHNGDBDS JOB
     //SYSIN DD *
           CHANGE.DBDS DBD(THISDB) AREA(AREA1) RECOVPD(10)
     /*
     ```

  2. Run the job.
  3. Check the JES log to ensure that the job ran successfully.
  4. Run a **LIST** command to see the new GENMAX value (optional).
  5. Run your image copy job.

- Situation 2: GENMAX value is too high
  - GENMAX has not been reached so the in use image copy data sets cannot be used.
  - No available image copy data sets exist that can be used.
  - Message DSP0084I was displayed.
  - No recovery period is defined.
  - Issue an **INIT.IC** command to define a new image copy data set for the identified database data set or area data set

    Or, you can perform the following task:
  - Issue a **CHANGE.DBDS** command to lower the GENMAX value.

  Attention: If GENMAX is lowered using the **CHANGE.DBDS** command, the new GENMAX value is recorded regardless of whether the oldest image copies are able to be deleted or not, because they are within the recovery period.

Related reference

CHANGE.DBDS command (Commands)

Adjusting GRPMAX when it is reached or it is too high

Before execution of the Change Accumulation utility, DBRC might determine that the change accumulation data sets are being reused and that the GRPMAX value must be changed. A command must be issued to correct this problem.

Situation 1 - The GRPMAX value is too low

- The GRPMAX value was reached.
- The oldest change accumulation data set cannot be reused because it is within the recovery period.
- No available change accumulation data sets exist that can be used.
- Message DSP1229A was displayed.

Either of the following actions resolves Situation 1:

- Increase the GRPMAX value and define additional change accumulation data sets.
- Change the recovery period so that the oldest change accumulation data set can be used.

To increase the GRPMAX value and to define additional change accumulation data sets, issue the **CHANGE.CAGRP** command and the **INIT.CA** commands as shown in the following examples. These examples assume that your CA group name is CAGRP1 and that your previous GRPMAX value was 10.

```
//CHNGCAG JOB

//SYSIN DD *
     CHANGE.CAGRP GRPNAME(CAGRP1)  -
                 GRPMAX(12)
/*
```

```
//INITIC JOB

//SYSIN DD *
     INIT.CA GRPNAME(CAGRP1) -
             CADSN(IMS.*.NEWCADSN1)
     INIT.CA GRPNAME(CAGRP1) -
             CADSN(IMS.*.NEWCADSN2)
/*
```

To change the recovery period so that the oldest change accumulation data set can be used, issue the **CHANGE.CAGRP** command with the RECOVPD keyword, as follows:

```
//CHNGCAG JOB
//SYSIN DD *
      CHANGE.CAGRP GRPNAME(CAGRP1) RECOVPD(10)
```

Situation 2 - The GRPMAX value is too high

- The GRPMAX value has not been reached so the in-use change accumulation data sets cannot be used.
- No available change accumulation data sets exist that can be used.
- Message DSP0085I was displayed.
- No recovery period is defined.

Either of the following actions resolves Situation 2:

- Issue the **INIT.CA** command to define a new change accumulation data set for the identified CA group.
- Issue the **CHANGE.CAGRP** command to lower the GRPMAX value.

Attention: If the GRPMAX value is lowered by using the **CHANGE.CAGRP** command, the GRPMAX value is recorded regardless of whether the oldest change accumulation data sets can be deleted, because they are within the recovery period.

Related reference

CHANGE.CAGRP command (Commands)

Closing an open online PRILOG record

During time stamp recoveries, when DBRC reports that there are open allocations that are not needed for the recovery, it might be easier to close the open PRILOG record rather than deleting the individual allocations. Use the **NOTIFY.PRILOG** command to close the open PRILOG record.

If an open PRILOG record is found for an online IMS subsystem, and the PRILOG record is not for the current run of IMS, it indicates that the last OLDS for this online IMS has not yet been archived. If the OLDS is no longer available and you must close the open PRILOG record, the following commands can be used to create a dummy DSN entry in the PRILOG:

```
/* This command creates dummy dataset */
NOTIFY.PRILOG DSN(DUMMY) STARTIME('2010.072 10:23:53.979935 -07:00') -
            FIRSTREC(00) RLDS SSID(IMS1) VOLSER(VOL001)

/* This command closes PRILOG record-Stoptime of 0's becomes runtime */
NOTIFY.PRILOG  RUNTIME('2010.072 11:23:53.980001 -07:00') -
             STARTIME('2010.072 10:23:53.979935 -07:00') -
             LASTREC(500) RLDS SSID(IMS1)

/* This command changes last PRILOG, DUMMY, to ERROR */
CHANGE.PRILOG STARTIME('2010.072 10:23:53.979935 -07:00')   ERROR  -
              DSSTART ('2010.072 10:24:43.286431 -07:00')
```

The following example is the PRILOG record as it appears in the RECON data set before issuing the **NOTIFY.PRILOG** command to close it:

```
PRILOG                                       RECORD SIZE=     464
  START = 2010.072 10:23:53.979935 -07:00*    SSID=IMS1     VERSION=15.2.1
  STOP  = 0000.000 00:00:00.000000 -07:00     #DSN=2
  GSGNAME=**NULL**
  FIRST RECORD ID= 0000000000000001       PRILOG TOKEN= 0
  EARLIEST CHECKPOINT = 2010.072 10:23:54.517056 -07:00

  DSN=IMSVS.RLDSP.IMS1.D07072.T1023539.V00                 UNIT=SYSDA
  START = 2010.072 10:23:53.979935 -07:00  FIRST DS LSN= 0000000000000001
  STOP  = 2010.072 10:24:32.697834 -07:00  LAST  DS LSN= 00000000000006D1
  FILE SEQ=0001    #VOLUMES=0001

    VOLSER=000000 STOPTIME = 2010.072 10:24:32.697834 -07:00
      CKPTCT=1    CHKPT ID = 2010.072 10:23:54.517056 -07:00
      LOCK SEQUENCE#=  000000000000

  DSN=IMSVS.RLDSP.IMS1.D07072.T1024326.V00                 UNIT=SYSDA
  START = 2010.072 10:24:32.697834 -07:00  FIRST DS LSN= 00000000000006D2
  STOP  = 2010.072 10:24:43.286431 -07:00  LAST  DS LSN= 0000000000000831
  FILE SEQ=0001    #VOLUMES=0001

    VOLSER=000000 STOPTIME = 2010.072 10:24:43.286431 -07:00
      CKPTCT=0    CHKPT ID = 2010.072 10:23:54.517056 -07:00
      LOCK SEQUENCE#=  000000000000

 LOGALL
  START   = 2010.072 10:23:53.979935 -07:00*
  EARLIEST ALLOC TIME = 2010.072 10:24:23.704405 -07:00
  DBDS ALLOC=1                        -DBD-    -DDN-    -ALLOC-
                                      DBVHDJ05 CJVHDG1E  1
```

The following example is the PRILOG record as it appears in the RECON data set after issuing the **NOTIFY.PRILOG** command to close it (differences are highlighted).

```
PRILOG                                       RECORD SIZE=     624
  START = 2010.072 10:23:53.979935 -07:00*    SSID=IMS1     VERSION=15.2.1
  STOP  = 2010.072 11:23:53.980001 -07:00     #DSN=3
  GSGNAME=**NULL**
  FIRST RECORD ID= 0000000000000001       PRILOG TOKEN= 0
  EARLIEST CHECKPOINT = 2010.072 10:23:54.517056 -07:00

  DSN=IMSVS.RLDSP.IMS1.D07072.T1023539.V00                 UNIT=SYSDA
  START = 2010.072 10:23:53.979935 -07:00  FIRST DS LSN= 0000000000000001
  STOP  = 2010.072 10:24:32.697834 -07:00  LAST  DS LSN= 00000000000006D1
  FILE SEQ=0001    #VOLUMES=0001

    VOLSER=000000 STOPTIME = 2010.072 10:24:32.697834 -07:00
      CKPTCT=1    CHKPT ID = 2010.072 10:23:54.517056 -07:00
      LOCK SEQUENCE#=  000000000000

  DSN=IMSVS.RLDSP.IMS1.D07072.T1024326.V00                 UNIT=SYSDA
  START = 2010.072 10:24:32.697834 -07:00  FIRST DS LSN= 00000000000006D2
  STOP  = 2010.072 10:24:43.286431 -07:00  LAST  DS LSN= 0000000000000831
  FILE SEQ=0001    #VOLUMES=0001

    VOLSER=000000 STOPTIME = 2010.072 10:24:43.286431 -07:00
      CKPTCT=0    CHKPT ID = 2010.072 10:23:54.517056 -07:00
      LOCK SEQUENCE#=  000000000000

  DSN=DUMMY                                         ERROR  UNIT=3400
  START = 2010.072 10:24:43.286431 -07:00  FIRST DS LSN= 0000000000000000
  STOP  = 2010.072 11:23:53.980001 -07:00  LAST  DS LSN= 00000000000001F4
  FILE SEQ=0001    #VOLUMES=0001

    VOLSER=VOL001 STOPTIME = 2010.072 11:23:53.980001 -07:00
      CKPTCT=0    CHKPT ID = 0000.000 00:00:00.000000 -07:00
      LOCK SEQUENCE#=  30E100000000

 LOGALL
  START   = 2010.072 10:23:53.979935 -07:00*
  EARLIEST ALLOC TIME = 2007.072 10:24:23.704405 -07:00
  DBDS ALLOC=1                        -DBD-    -DDN-    -ALLOC-
                                      DBVHDJ05 CJVHDG1E  1
```

Related reference

NOTIFY.PRILOG command (for RLDS) (Commands)

LIST.LOG command (for a PRILOG family) (Commands)

Working with subsystem records (SSYS)

If authorization for a database fails and you list the database and all the subsystem records in the RECON data set, you might wonder how the subsystem names (SSIDs) were created.

The following list discusses different aspects of the SSIDs.

- Naming conventions for SSIDs in RECON subsystem records

  The SSYS record is created when a signon call is issued to DBRC. The SSID must be unique. Normal conventions for creating the subsystem name are:

```
  IMS and DBCTL subsystems = IMSID value from IMSCTRL SYSGEN macro
                             (four characters)

  BATCH and UTILITY SUBSYSTEMS = JOBNAME

  XRF = RSENAME (Recovery Service Element) of the IMS systems
        (active and alternate)
```

  Exceptions include the following:

```
 ONLINE IMAGE COPY - XXXTZZZZ where

         XXXT is the DMB number of the database + DCB number translated
         to 0-9,A-Z
         ZZZZ is the control region IMSID
```

  Online image copy is the only BMP that creates its own subsystem record in the RECON data set.

- Batch backout

  If batch backout is processing a log created by a batch subsystem, the SSID that is used is the job name of the subsystem that is being backed out. The job name of the batch backout utility job is not used.

If batch backout is processing a log that was created by an online subsystem, the job name of the backout utility job is used as the SSID.

- Deleting a subsystem record

  When you no longer need a particular subsystem record in the RECON data set, the following commands can be used to delete the SSYS:

  ```
      CHANGE.SUBSYS SSID(XXXXX) STARTRCV
      CHANGE.SUBSYS SSID(XXXXX) ENDRECOV  <---(removes auth from DB)
      DELETE.SUBSYS SSID(XXXXX)
  ```

- Subsystem record size

  The subsystem record grows in size. If the subsystem record grows to exceed the 16M maximum record size, IMS abends.

Removing authorization inconsistency between the SSYS from DB/AREA records

To change database authorization or if authorization fails, a **LIST.DB** command shows that the DB or area is authorized to a nonexistent subsystem and backout is not required.

Use the following command to remove the authorization. You can also use the command below to remove a database/area name from the list in a subsystem record when a **LIST.SUBSYS** shows it authorized and a **LIST.DB** does not show it authorized.

```
    CHANGE.DB DBD(dbname) SSID(XXXX) UNAUTH   (or)
    CHANGE.DB DBD(dbname) AREA(areaname) SSID(XXXX) UNAUTH
```

Restarting the change accumulation process for logs

Change accumulation erroneously stops processing logs and the source of the problem is traced to an OLDS left in the ARC STARTED state. An IPL of z/OS is done on the CPU where the archive job was run and the status of the OLDS remains in this state.

GENJCL.ARCHIVE jobs do not include any OLDS in ARC STARTED status. The online IMS that created the OLDS is not running on this CPU.

If the online IMS is run on the same CPU as the archive job when IMS is restarted, it resets the status of all OLDS in ARC STARTED status to ARC NEEDED. The OLDS is included in the next archive job. To avoid this problem, run archive jobs on the same CPU as their corresponding IMS systems.

In cases where manual intervention is required, use the following command to change the status of the OLDS to archive needed (ARNEEDED):

```
  CHANGE.PRILOG OLDS(DFSOLPXX) ARNEEDED SSID(XXXX)
```

Restarting the change accumulation work when it states nothing to process

Change accumulation has issued a message stating that it found no logs to process. A review of the listing reveals that there are many logs that need to be processed.

DBRC selects all logs needed to satisfy the allocations for each DBDS since the effective image copy time. The logs are put in log volume start time order as a way for DBRC to keep track of which logs it has processed. DBRC processes all available logs and truncates the list of log volumes when it encounters a log in error, an open log, or when it detects an unarchived OLDS. A message indicating the condition that was encountered (such as an open log) is issued.

If the last change accumulation execution record shows that a SUBSET of logs was input to it last, the STOPTIME reflects the start time of the first unavailable log volume. Find this log volume time stamp in the listing of the RECON data set. If no log volume exists with this start time, there is an unarchived OLDS.

The following timeline illustrates this open OLDS condition:

```
T1--T2---T3--T4-T5--------T6--------T7---------T8----T9---


|---A1---D1--|--------------------A4-----------------// (open)
Log1 (DSN1)    (OPEN OLDS)

                |--A2--|  |--A3-|
                Log2      Log3
```

If GENJCL.CA is run at T7, it includes Log1(DSN1) and has a STOPTIME of T4 and is a SUBSET. T4 is the start time of the first unavailable log. After this JCL is executed, any attempts to run GENJCL.CA result in issuing two messages; one stating that an unarchived OLDS exists and another stating that there are no logs to process until the open OLDS is closed and archived.

Moving log data sets

DBRC records information about IMS logs in the RECON data set. This information includes the log data set name (DSN), its starting and ending times, and the volume serial (VOLSER).

The best way to avoid lost logs is to use cataloged log data sets and the DBRC CATDS option.

For non-cataloged log data sets, inform DBRC about changes by using the **CHANGE.PRILOG** or **CHANGE.SECLOG** command.

Related reading: See the **CHANGE.PRILOG** and the **CHANGE.SECLOG** commands in *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* .

Cataloging data sets

You can indicate whether image copy, change accumulation, and log data sets are cataloged or SMS managed by using the CATDS or NOCATDS keywords on the **CHANGE.RECON** or **INIT.RECON** command.

The **CHANGE.RECON** and **INIT.RECON** commands update the RECON data set header record accordingly if you specify either CATDS or NOCATDS. The only difference between the two commands in respect to cataloging is that **INIT.RECON** can be issued only once for each set of RECON data sets (3) required for an IMS region.

To specify that these data sets are cataloged specify:

```
    //CHGRECON JOB
    ⋮
    //SYSIN   DD  *
         CHANGE.RECON  CATDS
    /*
```

Or, you can specify:

```
    //INITRCON JOB
    ⋮
    //SYSIN   DD  *
         INIT.RECON  CATDS
    /*
```

The data set must have been initialized as cataloged for CATDS to be effective with the **CHANGE.RECON** command.

Specifying NOCATDS on the **CHANGE.RECON** or **INIT.RECON** command establishes that these data sets, regardless of their cataloged status, are not to be treated as cataloged.

To specify that these data sets are not to be treated as cataloged specify:

```
    //CHGRECON JOB
    ⋮
    //SYSIN    DD  *
         CHANGE.RECON  NOCATDS
    /*
```

Or, you can specify:

```
    //INITRCON JOB
    ⋮
    //SYSIN    DD  *
         INIT.RECON  NOCATDS
    /*
```

Related reading: See *IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands* for more information.

Performing multiple cold starts in a test environment

Many times in a test environment, you may want to cold start IMS. In order to cold start IMS the last OLDS must be closed.

If you have no need to close the OLDS, you can use the following commands to close the OLDS and mark it archived in the RECON data set so it can be reused:

```
NOTIFY.PRILOG STARTIME(%OLDOTIM) RUNTIME(070682226022-0800) -
      LASTREC(670) OLDS(DFSOLP02) SSID(IMS1)
CHANGE.PRILOG OLDS(DFSOLP02) ARCHIVED SSID(IMS1)
```

The entries for each OLDS (such as: DFSOLP00, DSPOLP01, and DSPOLD02) in the PRIOLD record are built when the OLDSs are used (if they do not already exist in the RECON data set). If you also need to delete the OLDS from the RECON data set, the following commands can be used:

```
DELETE.LOG OLDS(DFSOLP00) SSID(IMSA)
DELETE.LOG OLDS(DFSOLP01) SSID(IMSA)
DELETE.LOG OLDS(DFSOLP02) SSID(IMSA) LASTCLOS
```

Note that the LASTCLOS is necessary to delete the last OLDS used by IMS. The PRIOLD record is also deleted when the last DD name entry is removed.

The PRIOLD record before issuing the commands:

```
PRIOLD
  SSID=IMS1            # DD ENTRIES=3
  EARLIEST CHECKPOINT = 07.068 20:55:58.622862

  DDNAME=DFSOLP00   DSN=IMSTESTL.IMS01.OLDSP0
  START = 07.068 20:55:57.984378           FIRST DS LSN= 0000000000000001
  STOP  = 07.068 20:56:39.570062           LAST  DS LSN= 00000000000006D1
  LOCK SEQUENCE# = 000000000000
  STATUS=ARC COMPLT                                FEOV=YES   AVAIL
  PRILOG TIME=07.068 20:55:57.984378           ARCHIVE JOB NAME=JT205639
  VERSION=10.1

  DDNAME=DFSOLP01   DSN=IMSTESTL.IMS01.OLDSP1
  START = 07.068 20:56:39.570062           FIRST DS LSN= 00000000000006D2
  STOP  = 07.068 20:56:53.689006           LAST  DS LSN= 0000000000000831
  LOCK SEQUENCE# = 000000000000
  STATUS=ARC COMPLT                                FEOV=YES   AVAIL
  PRILOG TIME=07.068 20:55:57.984378           ARCHIVE JOB NAME=JT205653
  VERSION=10.1

  DDNAME=DFSOLP02   DSN=IMSTESTL.IMS01.OLDSP2
  START = 07.068 20:56:53.689006           FIRST DS LSN= 0000000000000832
  STOP  = 00.000 00:00:00.000000           LAST  DS LSN= 0000000000000000
  LOCK SEQUENCE# = 000000000000
  STATUS=ACTIVE                                    FEOV=NO    AVAIL
  PRILOG TIME=07.068 20:55:57.984378
  VERSION=10.1
```

The PRIOLD record after issuing the commands to close and mark DFSOLP02 as archived (with the differences highlighted):

```
PRIOLD
  SSID=IMS1            # DD ENTRIES=3
  EARLIEST CHECKPOINT = 07.068 20:55:58.622862

  DDNAME=DFSOLP00   DSN=IMSTESTL.IMS01.OLDSP0
  START = 07.068 20:55:57.984378           FIRST DS LSN= 0000000000000001
  STOP  = 07.068 20:56:39.570062           LAST  DS LSN= 00000000000006D1
  LOCK SEQUENCE# = 000000000000
  STATUS=ARC COMPLT                                FEOV=YES   AVAIL
  PRILOG TIME=07.068 20:55:57.984378           ARCHIVE JOB NAME=JT205639
  VERSION=10.1

  DDNAME=DFSOLP01   DSN=IMSTESTL.IMS01.OLDSP1
  START = 07.068 20:56:39.570062           FIRST DS LSN= 00000000000006D2
  STOP  = 07.068 20:56:53.689006           LAST  DS LSN= 0000000000000831
  LOCK SEQUENCE# = 000000000000
  STATUS=ARC COMPLT                                FEOV=YES   AVAIL
  PRILOG TIME=07.068 20:55:57.984378           ARCHIVE JOB NAME=JT205653
  VERSION=10.1

  DDNAME=DFSOLP02   DSN=IMSTESTL.IMS01.OLDSP2
  START = 07.068 20:56:53.689006           FIRST DS LSN= 0000000000000832
  STOP  = 07.068 22:26:02.200000           LAST  DS LSN= 000000000000029E
  LOCK SEQUENCE# = 000000000000
  STATUS=ARC COMPLT                                FEOV=NO    AVAIL
  PRILOG TIME=07.068 20:55:57.984378
  VERSION=10.1
```

Reducing potential RECON data set enqueue problems

The following list describes some scenarios that can cause the RECON data set enqueue problems and contain suggestions for avoiding these problems.

- Shared DASD environment scenarios

  Operating in a shared DASD environment, the most common cause of RECON data set enqueue problems is failing to follow the recommendation to catalog each RECON data set in its own ICF catalog on the same volume as the RECON data set.

  If you use serial access and if you catalog each RECON data set in its own ICF catalog on the same volume as the RECON data set and still have problems; examine your GRS, (or equivalent) RESERVE conversion list, to determine how you process SYSIGGV2 and DSPURI01 QNAMEs. A couple of combinations might lead to deadlocks.

- Non-shared DASD environment scenarios

  If you are operating in a non-shared DASD environment and are having problems, this is probably not caused by deadlock but rather by contention and slow performance. Here are a few things to look at in this situation:

  – Minimize the amount of time any job holds the RECON data set.

    One way to minimize that time is to tune the LSR buffer pool DBRC uses when accessing the RECON data sets. There is a CSECT, DSPBUFFS (serial access only), that contains the values used for online and batch. See "Buffer Size Specification facility (DSPBUFFS)" in *IMS Version 15.2 Exit Routines* for information about how to zap it and change the values. The defaults might be low for your usage. For parallel access, the IDGSMSxx RLS_MAX_POOL_SIZE definitions should be looked into instead.

  – Analyze the other contents of the volumes where the RECON data set resides. Consider isolating these volumes, to prevent interference from other I/O activity (and vice versa). Consider placing the RECON data sets on high performance cached DASD, or perhaps solid state DASD.

Related reading: See “Avoiding RECON data set deadlock situations for serial access mode” on page 534 for more information.

Part 4. IMS system recovery

An Extended Recovery Facility (XRF) or an IMSplex allows you to maintain continuity of online transaction processing by giving you the ability to rapidly resume end-user service when a failure occurs.

Chapter 46. Recovery in an IMSplex

Recovery in an IMSplex is performed differently for the member IMS systems, Repository Server (RS), and Common Service Layer (CSL).

The following topics provide an overview of each type of recovery:

Recovery of IMS systems in an IMSplex

The recovery procedures for a failed IMS system within an IMSplex are the same as the procedures for recovering a stand-alone IMS system. For example, the procedures for an IMS system that terminates because of a z/OS failure or a hardware power failure are the same regardless of whether the IMS system is a member of an IMSplex: you must IPL z/OS and then restart IMS using an **/ERESTART** command.

Recovery of CSL in an IMSplex

If an entire CSL fails, each CSL manager must be restarted. If OM fails and you only have one OM defined for an IMSplex, you lose your type-2 command interface capability and cannot enter commands through a SPOC. You can still enter type-1 commands through supported sources. OM can be restarted by using the z/OS Automatic Restart Manager (ARM) as defined on the OM execution parameter, specifying a started procedure, or submitting JCL.

Similarly, if RM or SCI fails, each must be restarted, either by the z/OS Automatic Restart Manager, specifying a started procedure, or submitting JCL. If no RMs are available, the IMSplex continues functioning, but all RM functions cease until an RM is restarted. If every RM, resource structure, coupling facility, or CQS fails or is inaccessible within an IMSplex, global resource management does not function until they are all restarted. If every SCI fails, the CSL cannot operate because all communication is down. SCI failure also affects automatic RECON loss notification. SCI must be restarted to bring the CSL back in operation.

Recovery of an RS in an IMSplex

The IMSRSC repository is managed by one master RS that manages the repositories defined to it. One or more subordinate RSs can be defined in the sysplex.

The subordinate RSs wait in an initialization state until they identify that the master RS has shut down, at which point, they all attempt to complete the startup process and become the master RS. One subordinate RS becomes the new master RS. The others remain as subordinate RSs.

The master and subordinate RSs must belong to the same z/OS cross-system coupling facility (XCF) group. When you start RSs in different XCF groups, they are independent of one another.

When a subordinate RS attempts to become the master RS the FRP2003I message is issued. When the subordinate RS successfully becomes the master RS, the FRP2002I and FRP2025I messages are issued.

The implementation of subordinate RSs is designed to help accept new connections as quickly as possible if a master RS terminates. However, a subordinate RS does not shadow the master RS. RS clients pass a registration exit to the RS during registration. The registration exit gets driven with the RS unavailable event when the master RS goes down and with the available event when the subordinate RS becomes the master. When the new master is available, the client must then reregister to the new RS and reconnect to any repositories in use.

Related concepts

Overview of the IMSRSC repository (System Definition)
Starting subordinate Repository Servers (Operations and Automation)
IMSRSC repository and RS catalog repository data sets (System Definition)
Opening the IMSRSC repository (Operations and Automation)

Related information

FRP2003I (Messages and Codes)

FRP2002I (Messages and Codes)

Part 5. Using IMS reports

These topics describe IMS reports and how to use them.

Chapter 47. DB Monitor reports

These topics describe the DB-Monitor reports and how to use them.

Related concepts

Monitoring VSAM buffers (Database Administration)

VSAM-Buffer-Pool report

The VSAM-Buffer-Pool report gives you processing information about a specific VSAM subpool. One report is produced for each subpool you specify.

For subpools within a VSAM local shared resource pool, reports are produced in ascending order based on subpool buffer size. If both index and data subpools exist in a given shared resource pool, index subpool reports follow data subpool reports. Reports for subpools in different shared resource pools are always produced in the order the shared resource pools are specified. You specify shared resource pools and subpools in control statements processed when IMS is initialized. In a batch system, the control statements are put in the DFSVSAMP data set. In an online system, they are put in the IMS.PROCLIB data set with the DFSVSMnn member name.

The VSAM-Buffer-Pool report has no meaning for HSAM or SHSAM databases, because neither of these databases can use VSAM as the access method.

Using the VSAM-Buffer-Pool report

The primary usefulness of the VSAM-Buffer-Pool report is to calculate how many I/O operations were required to read to or write from the subpool.

You might want to increase buffer pool size to see if you can decrease the number of I/O operations. You might also want to turn on background write. Or, if the number of I/O operations is increasing over time, you might need to reorganize your database.

To calculate the number of I/O operations required to read to or write from the buffer pool, use the following formula:

Total I/O equals the sum of:

1. The number of times a CI had to be read into the buffer from the database (NUMBER of CONTR INT READ FROM EXTERNAL STORAGE field in the report).
2. The number of times a buffer had to be written to the database (NUMBER OF VSAM WRITES INITIATED BY IMS field in the report).
3. The number of times a buffer had to be written to the database so a new CI could be read into the buffer (NUMBER OF VSAM WRITES TO MAKE SPACE IN THE POOL field in the report).

Fields in the VSAM-Buffer-Pool report

The following example is a sample of a VSAM-Buffer-Pool report.

```
             V S A M   B U F F E R   P O O L
                                                                          FIX INDEX/BLOCK/
DATA           N/Y/N
                                                                          SHARED RESOURCE POOL
ID        VPL/1
                                                                          SHARED RESOURCE POOL
TYPE          D
                                                                          SUBPOOL
ID                         1
                                                                          SUBPOOL BUFFER
SIZE              2048
                                                                          NUMBER  HIPERSPACE
BUFFERS          0
                                                                          TOTAL BUFFERS IN
SUBPOOL            4
```

```
                                                                           17:08:15          17:10:16
                                                                     START TRACE        END
TRACE        DIFFERENCE

    NUMBER OF RETRIEVE BY RBA CALLS RECEIVED BY BUF HNDLR                 0
0                  0
    NUMBER OF RETRIEVE BY KEY CALLS                                       0
0                  0
    NUMBER OF LOGICAL RECORDS INSERTED INTO ESDS                          0
0                  0
    NUMBER OF LOGICAL RECORDS INSERTED INTO KSDS                          0
0                  0
    NUMBER OF LOGICAL RECORDS ALTERED IN THIS SUBPOOL                     0
0                  0
    NUMBER OF TIMES BACKGROUND WRITE FUNCTION INVOKED                     0
0                  0
    NUMBER OF SYNCHRONIZATION CALLS RECEIVED                              18566
27923              9357
    NUMBER OF WRITE ERROR BUFFERS CURRENTLY IN THE SUBPOOL                0
0                  0
    LARGEST NUMBER OF WRITE ERRORS IN THE SUBPOOL                         0
0                  0
    NUMBER OF VSAM GET CALLS ISSUED                                       0
0                  0
    NUMBER OF VSAM SCHBFR CALLS ISSUED                                    0
0                  0
    NUMBER OF TIMES CONT INT REQUESTED ALREADY IN POOL                    229956
349375             119419
    NUMBER OF CONTR INT READ FROM EXTERNAL STORAGE                        1078
1229               151
    NUMBER OF VSAM WRITES INITIATED BY IMS                                33
64                 31
    NUMBER OF VSAM WRITES TO MAKE SPACE IN THE POOL                       0
0                  0
    NUMBER OF VSAM READS FROM HIPERSPACE BUFFERS                          0
0                  0
    NUMBER OF VSAM WRITES FROM HIPERSPACE BUFFERS                         0
0                  0
    NUMBER OF FAILED VSAM READS FROM HIPERSPACE BUFFERS                   0
0                  0
    NUMBER OF FAILED VSAM WRITES FROM HIPERSPACE BUFFERS                  0
0                  0
```

The meaning of the various fields in the report is as follows:

FIX INDEX/BLOCK/DATA
This field indicates the fix options for the index buffers/data buffer prefix/data buffers for this subpool.

SHARED RESOURCE POOL ID
This field is a four character, VSAM local-shared-resource-pool ID provided at subpool definition time. This ID is specified in the DFSVSAMP or IMS.PROCLIB data set control statements.

SHARED RESOURCE POOL TYPE
This field indicates whether this subpool contains index (I) or data (D) buffers.

SUBPOOL ID
This field describes the subpool ID. A unique subpool ID is assigned for each different database buffer size in each VSAM local shared resource pool you specify in the DFSVSAMP or IMS.PROCLIB data set control statements.

SUBPOOL BUFFER SIZE
This field describes the size, in bytes, of buffers in this subpool. You can specify buffers of different sizes. A subpool contains all buffers of the same size. All subpools grouped together make up the buffer pool. Buffer size is specified by you in the control statements for the DFSVSAMP or IMS.PROCLIB data sets. For example, suppose you specified:

```
//DFSVSAMP DD input for VSAM and OSAM buffers and options
⋮
POOLID=BB
VSRBF=4096,4,D,HSR,10
VSRBF=8192,4,D,HSO,20
POOLID=FF
VSRBF=4096,4,I
VSRBF=8192,4,D
```

```
⋮
/*
```

This gives you four subpools and the four VSAM-Buffer-Pool reports displayed in the following examples. The headings in each report display as shown in the examples.

```
SHARED RESOURCE POOL ID:                  BB
SHARED RESOURCE POOL TYPE:                  D
SUBPOOL ID:                                           1
SUBPOOL BUFFER SIZE:                      4096
HIPERSPACE BUFFERS:                           10
TOTAL BUFFERS IN SUBPOOL:                   4
```

```
SHARED RESOURCE POOL ID:                  BB
SHARED RESOURCE POOL TYPE:                  D
SUBPOOL ID:                                           2
SUBPOOL BUFFER SIZE:                      8192
HIPERSPACE BUFFERS:                           20
TOTAL BUFFERS IN SUBPOOL:                   4
```

```
SHARED RESOURCE POOL ID:                  FF
SHARED RESOURCE POOL TYPE:                  D
SUBPOOL ID:                                           1
SUBPOOL BUFFER SIZE:                      8192
HIPERSPACE BUFFERS:                            0
TOTAL BUFFERS IN SUBPOOL:                   4
```

```
SHARED RESOURCE POOL ID:                  FF
SHARED RESOURCE POOL TYPE:                  I
SUBPOOL ID:                                           2
SUBPOOL BUFFER SIZE:                      4096
HIPERSPACE BUFFERS:                            0
TOTAL BUFFERS IN SUBPOOL:                   4
```

NUMBER OF HIPERSPACE BUFFERS
This field indicates the number of hiperspace buffers specified at subpool definition time.

TOTAL BUFFERS IN SUBPOOL
This field describes how many buffers are in the specified subpool.

START TRACE, END TRACE, and DIFFERENCE
The start trace and end trace fields tell you the time the DB Monitor program was *last* started and stopped. The time is generated by the time-of-day clock. Clock times are read as follows:

```
Clock time = hh.mm.ss
```

where:

hh
Hours 0 through 23

mm
Minutes

ss
Seconds

If the DB Monitor program was on during an entire batch run, the start and end trace times are when the batch run started and stopped. If the DB Monitor program was turned on and off more than once in the same batch run, the start and end trace times are when the monitor was *last* started and stopped.

The numbers under the start and end trace fields are cumulative numbers for the entire batch run. If the monitor was only turned on and off once, the start trace number is zero. If the monitor was turned on and off more than once, the start trace numbers are the cumulative numbers when the monitor was last turned on; the stop trace numbers are the cumulative numbers when the monitor was last turned off.

The difference column describes the difference between the cumulative numbers in the start and end trace fields. If the monitor was only turned on and off once, the difference column contains the same numbers as the end trace column.

NUMBER OF RETRIEVE BY RBA CALLS

This field describes how many retrieve by relative byte address (RBA) calls were issued for this subpool. Retrieve by RBA calls are calls issued internally by DL/I. One retrieve by RBA call is issued for each direct-address pointer that must be followed in searching for a segment. For example, a GN call for a dependent segment in an HDAM or PHDAM database uses a series of RBA calls to search for the dependent segment, one call for each direct-address pointer it follows.

If you want to know the exact sequence of a search when a retrieve by RBA call is used, you can record the sequence by turning on the buffer handler trace and using a SNAP call to see the trace records. You can turn on the buffer handler trace using the DL/I= operand on the OPTIONS statement for the DFSVSAMP or DFSVSMnn data set. The SNAP call can be issued from the application program or by using the DFSDDLT0 test program.

One call from an application program can generate more than one retrieve by RBA call. The retrieve by RBA call might or might not require an I/O operation. Because the number in this field does not reflect the number of I/O operations to access a segment, do not use it to judge VSAM performance.

NUMBER OF RETRIEVE BY KEY CALLS

This field describes how many retrieve by key calls were issued for this subpool. Retrieve by key calls are calls issued internally by DL/I. The calls are issued to search a KSDS using a key as a qualification (where key is equal to or greater than X). For example, a GU call for a root segment in a HIDAM database causes DL/I to issue a retrieve by key call to access the index segment pointing to the requested root segment.

One call from an application program can generate more than one retrieve by key calls. The retrieve by key calls might or might not require an I/O operation. Because the number in this field does not reflect the number of I/O operations to access a segment, do not be use it to judge VSAM performance.

NUMBER OF LOGICAL RECORDS INSERTED INTO ESDS

This field describes how many of the logical records in your ESDS that were previously empty now contain segments. When a dependent segment is inserted into an ESDS in a HISAM or HIDAM database, the segment might not fit into a logical record that already contains other segments. In this case, the segment is put into a new ESDS logical record. When a dependent segment is inserted into a logical record in an ESDS in a HISAM database, other segments in the same logical might need to be shifted into a new ESDS logical record to make room for the segment being inserted.

Look at this field from one report to the next. It helps you determine when you are running out of logical records in the primary space you have allocated. It is best to avoid using logical records from secondary space because this space is probably not close to the primary space. The distance between the two areas of space might cause extra seek time and therefore poor performance. In general, it is best to reorganize your database before you need to use secondary space.

NUMBER OF LOGICAL RECORDS INSERTED INTO KSDS

This field describes how many of the logical records in your KSDS that were previously empty now contain segments. HISAM databases use a new logical record when a root segment is inserted. HIDAM index databases use a new logical record for the index segment created when a root segment is inserted. Secondary index databases use a new logical record when a new pointer segment is inserted.

Look at this field from one report to the next. It helps you determine when you are running out of logical records in the primary space you have allocated. It is best to avoid using logical records from secondary space because this space is probably not close to the primary space. The distance between the two areas of space might cause extra seek time and therefore poor performance. In general, it is best to reorganize your database before you need to use secondary space.

NUMBER OF LOGICAL RECORDS ALTERED IN THIS SUBPOOL

This field describes how many logical records, while in the buffer pool, were marked as altered. When a segment is inserted or replaced in a logical record, the logical record in the buffer is marked as altered until it is written back to the database.

NUMBER OF TIMES BACKGROUND WRITE FUNCTION INVOKED

If you have specified use of the background write function, this field tells how many times the function was used. The background write function, at intervals, writes buffers containing modified data back to the database. It does this so buffers are available for use when an application program needs them. Without background write, if an application program wants to read data into a buffer that already contains modified data, the application program has to wait while the contents of the buffer are written back to the database. The number of times background write is invoked is the same on each subpool report produced, during a given execution of the monitor, for a given local shared resource pool. Once invoked, the background write function writes buffers from all subpools within a local shared resource pool.

Background write is specified in the BGWRT= operand of the OPTIONS statement for the DFSVSAMP or DFSVSMnn data set.

NUMBER OF SYNCHRONIZATION CALLS RECEIVED

This field describes how many requests were made to write all altered buffers in the local shared resource pool while the monitor was on. CHKP and STAT calls are typical requestors of this.

NUMBER OF PERM WRT ERROR BUFFS NOW IN THE SUBPOOL

This field describes how many buffers are currently "frozen" in storage because a permanent I/O error occurred when writing them to the database. When a VSAM write operation results in a permanent I/O error, the affected buffers are frozen in storage until the data set is closed or, in the online system, until the system is shut down. Once the data set is closed or the system shut down, the buffers are written to the log and the number in this field returns to zero.

Ensure that the operator performed database recovery of the affected data set before you ever receive this report.

LARGEST NUMB OF PERM ERR BUFFS EVER IN THE SUBPOOL

This field describes how many buffers *were* frozen in storage when the condition described in the previous field occurred.

NUMBER OF VSAM GET CALLS ISSUED

This field describes how many times VSAM GET calls were issued. VSAM GET calls are calls issued internally by DL/I. The GET call might be satisfied by data in the buffer pool or it might require that data be read into the buffer pool. Because the number in this field does not reflect the number of I/O operations required to access a segment, do not use it to judge VSAM performance.

NUMBER OF VSAM SCHBFR CALLS ISSUED

This field describes how many times the HD space management routine issued calls to search for space in which to insert segments.

If, from one monitor report to the next, the number in this field is increasing, it means that space for storing new segments is not available in the most desirable location. Eventually, this means you must reorganize your database to improve performance. In reorganizing, pay special attention to the operands affecting database space (the bytes operand in the RMNAME= keyword in the DBD statement and the fbff and fspf operands in the FRSPC= keyword in the DATA SET statement).

NUMBER OF TIMES CONT INT REQUESTED ALREADY IN POOL

This field describes how many times a logical record was found in a CI that was already in the buffers. When this occurs, no I/O operations are required to access the desired segments.

If you are trying to improve performance, increase the number of buffers you have allocated. If you increase the number of buffers, you can monitor this field to see if the number in it increases, which indicates improved performance.

NUMBER OF CONTR INT READ FROM EXTERNAL STORAGE

This field describes how many times a logical record was not found in a CI that was already in the buffers. When this occurs, an I/O operation is required to read the CI containing the logical record into the buffer pool. Because performance is always better when fewer I/O operations are performed, you might want to increase the number of buffers you have specified to see how that affects the number in this field. Specifying more buffers keeps more CIs (and therefore logical records) in the buffer pool. There is a break-even point in this process, however, where too many buffers are specified, and it takes longer to search and maintain the buffers than it takes to read a CI into the buffer.

The number of buffers is specified in the control statements for the DFSVSAMP or DFSVSMnn data sets.

NUMBER OF VSAM WRITES INITIATED BY IMS

This field describes the number of times DL/I issued a write request to write data to the database. Write operations are issued when:

- A data set is closed. Database buffers containing data that has been altered by the data set being closed are written to the database.
- Abnormal termination occurs during application program processing. Database buffers containing data that has been altered are written to the database.
- The background write function is invoked. Selected database buffers containing data that has been altered are written to the database.
- A checkpoint call is issued. All altered database buffers are written to the database.

NUMBER OF VSAM WRITES TO MAKE SPACE IN THE POOL

This field describes how many times a CI had to be read into a buffer containing a logical record with altered data. When this happens, the buffer (because it contains altered data) has to be written back to the database before the new CI can be read into it. This means the application program has to wait while the write operation takes place.

For best performance, ensure that the number in this field is close to zero. This can generally be achieved by turning on the background write function during batch processing and adjusting the number of buffers allocated.

NUMBER OF VSAM READS FROM HIPERSPACE BUFFERS

This field describes the total number of successful VSAM reads (MOVEPAGE and NON-MOVEPAGE) from the hiperspace buffers.

NUMBER OF VSAM WRITES FROM HIPERSPACE BUFFERS

This field describes the total number of successful VSAM writes (MOVEPAGE and NON-MOVEPAGE) to the hiperspace buffers.

NUMBER OF FAILED VSAM READS FROM HIPERSPACE BUFFERS

This field describes the number of times that a VSAM read request from hiperspace failed, resulting in a read from DASD.

NUMBER OF FAILED VSAM WRITES FROM HIPERSPACE BUFFERS

This field describes the number of times that a VSAM WRITE request to hiperspace failed, resulting in a write to DASD.

VSAM-Statistics report

VSAM-Statistics reports are based on: a specific application program PCB, a data set the application program is using, and the type of DL/I call the application program issued.

The VSAM-Statistics report has no meaning for HSAM or SHSAM databases because neither of these databases can use VSAM as the access method.

Using the VSAM-Statistics report

From a VSAM-Statistics report, you can determine which calls in an application program require a great many I/O operations. After you know this, you can improve performance by tuning either the database or the application program to reduce I/O operations.

The following fields in the report tell actual I/O activity and are therefore the most important ones to monitor:

- READS
- USR WTS
- NUR WTS

If you can reduce the averages in these fields, performance can be improved.

In tuning to reduce I/O operations, pay most attention to calls issued a large number of times. It is more profitable to save 1 second on a call executed 2000 times than to save 5 seconds on a call executed ten times (2000 versus 50 seconds). The DL/I-Call-Summary report (discussed under "DL/I-Call-Summary report" on page 587) describes how many times each DL/I application program call is issued.

Fields in the VSAM-Statistics report

The following example is a sample of a VSAM-Statistics report.

```
IMS MONITOR   ****VSAM STATISTICS****     TRACE START 1989 076  12:42:54     TRACE STOP  1989 076
12:43:07  PAGE  0012

                  DL/I   VSAM     RET    RET   ISRT   ISRT     BFR    BKG
SYN                                USR    NUR
PCBNAME  DDNAME   FUNC  IWAITS    RBA    KEY   ESDS   KSDS    ALT    WTS    PTS   GETS  SCHBFR FOUND
READS   WTS    WTS
-------  ------   ----  -----    ---    ---   ----   ----    ---    ---    ---   ----  ------ -----
_____   ___    ___
DLVNTZ02 DBHVSAM2 STAT   1       1.00   0.00   0.00   0.00   1.00   0.00   1.00   1.00   0.00   1.00
0.00   1.00   0.00
         DD TOTAL
         __ _____        1       1.00   0.00   0.00   0.00   1.00   0.00   1.00   1.00   0.00   1.00
0.00   1.00   0.00
         HIDAM    STAT   4     307.25  36.50   0.50   0.00  35.00   0.00   0.25   120.25 0.00 157.00
0.25   0.75   0.00
         DD TOTAL
         __ _____        4     307.25  36.50   0.50   0.00  35.00   0.00   0.25 120.25   0.00 157.00
0.25   0.75   0.00
         XDLBT04I GU     2      15.00   4.50   0.00   0.00   0.00   0.00   0.00  17.50   0.00  21.50
1.00   0.00   0.00
         DD TOTAL
         __ _____        2      15.00   4.50   0.00   0.00   0.00   0.00   0.00  17.50   0.00  21.50
1.00   0.00   0.00
PCB TOTAL
___ _____                7     180.00  22.14   0.28   0.00  20.14   0.00   0.28  73.85   0.00  96.00
0.42   0.57   0.00
DLVNTZX2 HIDAM    GN     1      15.00   3.00   0.00   0.00   0.00   0.00   0.00  18.00   0.00  20.00
1.00   0.00   0.00
                  GU     1       1.00   0.00   0.00   0.00   0.00   0.00   0.00   1.00   0.00   0.00
1.00   0.00   0.00
         DD TOTAL
         __ _____        2       8.00   1.50   0.00   0.00   0.00   0.00   0.00   9.50   0.00  10.00
1.00   0.00   0.00
         DBHVSAM1 GU     8       0.00   0.12  12.75 520.00   7.50   0.00   0.00   0.12   0.00   0.00
0.87   0.00   0.00
         DD TOTAL
         __ _____        8       0.00   0.12  12.75 520.00   7.50   0.00   0.00   0.12   0.00   0.00
0.87   0.00   0.00
         XDLBT04I GU     6       0.00   0.00   0.00   0.00   0.00   0.00   0.00   0.00   0.00   0.00
1.00   0.00   0.00
         DD TOTAL
         __ _____        6       0.00   0.00   0.00   0.00   0.00   0.00   0.00   0.00   0.00   0.00
1.00   0.00   0.00
         DBHVSAM2 GU     3       0.66   0.00   0.00   0.00   0.00   0.00   0.00   0.66   0.00   0.00
1.00   0.00   0.00
         DD TOTAL
         __ _____        3       0.66   0.00   0.00   0.00   0.00   0.00   0.00   0.66   0.00   0.00
1.00   0.00   0.00
PCB TOTAL
___ _____                19      0.94   0.21   5.36 482.10   3.15   0.00   0.00   1.15   0.00   1.05
0.94   0.00   0.00
BATCH TOTAL
_____ _____              26     49.15   6.11   3.84  83.07   3.11   0.00   0.07  20.73   0.00  26.61
0.80   0.15   0.00
```

The meaning of the various fields in the report is as follows:

TRACE START and TRACE STOP

The trace start and trace stop fields tell you the time when the DB Monitor program was last started and stopped. The time is generated by the time-of-day clock. Clock times are read as follows:

```
Clock time = hh.mm.ss
```

where:

hh
Hours 0 through 23

mm
Minutes

ss
Seconds

If the DB Monitor was on during an entire batch run, the trace start and trace stop time is when the batch run started and stopped. If the DB Monitor was turned on and off more than once in the same batch run, the trace start and trace stop time is when the monitor was last started and stopped.

PCBNAME
This field describes the name of the PCB the report is providing information about. Remember that each application program has one or more PCBs. This field can only be used to identify which application program the report is providing information about if PCB names are unique to application programs.

DDNAME
This field describes the name of the data set the application program is using. Within one report, an application program (PCB) can access more than one data set. The statistics compiled are listed separately for each data set.

DL/I FUNC
This field describes the type of DL/I calls the application program issued.

VSAM IWAITS
This field describes, by type of DL/I call against a specific data set, the number of times IMS had to wait before processing could proceed. When IMS has to wait, it is almost always waiting for an I/O operation to take place, that is, data is being either read from the database to the buffer or written from the buffer back to the database.

The numbers in each column under all remaining fields in the report are averages. They tell the average number of times an activity occurred rather than the specific number of times. Averages are for waits. These numbers are truncated. For example, a value of 0.019 is printed as 0.01.

RET RBA
This field describes how many retrieve by relative byte address (RBA) calls were issued for this subpool. Retrieve by RBA calls are calls issued internally by DL/I. One retrieve by RBA call is issued for each direct-address pointer that must be followed in searching for a segment. For example, a GN call for a dependent segment in an HDAM or PHDAM database uses a series of RBA calls to search for the dependent segment, one call for each direct-address pointer it follows.

One call from an application program can generate more than one retrieve by RBA call. The retrieve by RBA call might or might not require an I/O operation. Because the number in this field does not reflect the number of I/O operations to access a segment, do not use it to judge VSAM performance.

RET KEY
This field describes how many retrieve by key calls were issued for this subpool. Retrieve by key calls are calls issued internally by DL/I. The calls are issued to search a KSDS using a key as a qualification (where key is equal to or greater than X). For example, a GU call for a root segment in a HIDAM or PHIDAM database causes DL/I to issue a retrieve by key call to access the index segment pointing to the requested root segment.

One call from an application program can generate more than one retrieve by key calls. The retrieve by key calls might or might not require an I/O operation. Because the number in this field does not reflect the number of I/O operations to access a segment, do not use it to judge VSAM performance.

ISRT ESDS
This field describes how many of the logical records in your ESDS that were previously empty now contain segments. When a dependent segment is inserted into an ESDS in a HISAM, HIDAM or PHIDAM database, the segment might not fit into a logical record that already contains other

segments. In this case, the segment is put into a new ESDS logical record. When a dependent segment is inserted into a logical record in an ESDS in a HISAM database, other segments in the same logical record might need to be shifted into a new ESDS logical record to make room for the segment being inserted.

Look at this field from one report to the next. It helps you determine when you are running out of logical records in the primary space you have allocated. It is best to avoid using logical records from secondary space because this space is probably not close to the primary space.

ISRT KSDS
This field describes how many of the logical records in your KSDS that were previously empty now contain segments. HISAM databases use a new logical record when a root segment is inserted. HIDAM and PHIDAM index databases use a new logical record for the index segment created when a root segment is inserted.

Look at this field from one report to the next. It helps you determine when you are running out of logical records in the primary space you have allocated. It is best to avoid using logical records from secondary space because this space is probably not close to the primary space. The distance between the two areas of space might cause extra seek time and therefore poor performance. In general, it is best to reorganize your database before you need to use secondary space.

BFR ALT
This field describes how many logical records, while in the buffer pool, were marked as altered. When a segment is inserted or replaced in a logical record, the logical record in the buffer is marked as altered until it is written back to the database.

BKG WTS
If you have specified use of the background write function, this field tells how many times the function was used. Background write, at intervals, writes buffers containing modified data back to the database. It does this so buffers are available for use when an application program needs them. Without background write, if an application program wants to read data into a buffer that already contains modified data, the application program has to wait while the contents of the buffer are written back to the database. The number of times background write was invoked is the same on each subpool report produced during a given execution of the monitor. This is because, once involved, background write writes buffers from all subpools.

Background write is specified in the BGWRT= operand of the OPTIONS statement for the DFSVSAMP or DFSVSMnn data set.

SYN PTS
This field describes how many times checkpoint calls were issued in DL/I programs while the monitor was on.

GETS
This field describes how many times VSAM GET calls were issued. VSAM GET calls are calls issued internally by DL/I. The GET call might be satisfied by data in the buffer pool or it might require that data be read into the buffer pool. Because the number in this field does not reflect the number of I/O operations required to access a segment, do not use it to judge VSAM performance.

SCHBFR
This field describes how many times the HD space management routine issued calls to search for space in which to insert segments.

If, from one monitor report to the next, the number in this field is increasing, it means that space for storing new segments is not available in the most desirable location. Eventually, you must reorganize your database to improve performance. In reorganizing, pay special attention to the operands affecting database space (the BYTES operand in the RMNAME= keyword in the DBD statement and the fbff and fspf operands in the FRSPC= keyword in the DATA SET statement).

FOUND
This field describes how many times a logical record was found in a CI that was already in the buffers. When this occurs, no I/O operations are required to access the desired segments.

If you are trying to improve performance, increase the number of buffers you have allocated. If you increase the number of buffers, you can monitor this field to see if the number in it increases, which indicates improved performance.

READS
This field describes how many times a logical record was not found in a CI that was already in the buffers. When this occurs, an I/O operation is required to read the CI containing the logical record into the buffer pool. Because performance is always better when fewer I/O operations are performed, you might want to increase the number of buffers you have specified to see how that affects the number in this field. Specifying more buffers keeps more CIs (and therefore logical records) in the buffer pool. There is a break-even point in this process, however, where too many buffers are specified, and it takes longer to search and maintain the buffers than it takes to read a CI into the buffer.

The number of buffers is specified in the control statements for the DFSVSAMP or DFSVSMnn data sets.

USR WTS
This field, which indicates user writes, describes the number of times DL/I issued a write request to write data to the database. Write operations are issued when:

- A data set is closed. Database buffers containing data that has been altered by the data set being closed are written to the database.
- Abnormal termination occurs during application program processing. Database buffers containing data that has been altered are written to the database.
- The background write function is invoked. Selected database buffers containing data that has been altered are written to the database.
- A checkpoint call is issued. All altered database buffers are written to the database.

NUR WTS
This field, which indicates nonuser writes, tells how many times a CI had to be read into a buffer containing a logical record with altered data. When this happens, the buffer (because it contains altered data) has to be written back to the database before the new CI can be read into it. This means the application program has to wait while the write operation takes place.

For best performance, ensure that the number in this field is close to zero. This can generally be achieved by turning on the background write function during batch processing and adjusting the number of buffers allocated.

SCRS
This field describes the total number of successful VSAM reads (MOVEPAGE and NON-MOVEPAGE) from hiperspace buffers.

SCWS
This field describes the total number of successful VSAM writes (MOVEPAGE and NON-MOVEPAGE) to hiperspace buffers.

SCRF
This field describes the number of times that a VSAM read request from hiperspace failed, resulting in a read from DASD.

SCWF
This field describes the number of times that a VSAM write request to hiperspace failed, resulting in a write to DASD.

DD TOTAL
This field describes, for a given data set, the overall average number of times an activity occurred (except for the VSAM CALLS field, which tells total number of times).

PCB TOTAL
This field describes, for a given PCB, the overall average number of times an activity occurred (except for the VSAM CALLS field, which tells total number of times).

BATCH TOTAL

This field describes, for the time the monitor was running, the overall average number of times an activity occurred (except for the VSAM CALLS field, which tells total number of times).

Database-Buffer-Pool report

The Database-Buffer-Pool report gives you information about OSAM subpools during processing. One report is produced for all subpools in the buffer pool.

This report and the VSAM-Buffer-Pool report differ in that the VSAM report was produced for each subpool in the buffer pool.

The Database-Buffer-Pool report has no meaning for HSAM, SHSAM, SHISAM, or GSAM databases because none of these databases can use OSAM as the access method.

Using the Database-Buffer-Pool report

The primary usefulness of the Database-Buffer-Pool report is to calculate how many I/O operations were required to read to or write from the OSAM buffer pool.

You might want to increase buffer pool size to see if you can decrease the number of I/O operations. Or, if the number is increasing over time, you might need to reorganize your database.

To calculate the number of I/O operations required to read to or write from the buffer pool, use the following formula:

Total I/O equals the sum of:

1. Blocks read from the database

   Number of blocks read for OSAM specific requests (Number of Read Requests Issued)
2. Blocks written to the database

   Number of blocks written because of buffer steal processing and purge processing (Number of Blocks Written)

Fields in the Database-Buffer-Pool report

The following figure is an example of a Database Buffer Pool report.

```
                    D A T A   B A S E   B U F F E R   P O O L
                                                                                       FIX PREFIX/
BUFFERS           Y/Y
                                                                                       SUBPOOL
ID                  004K
                                                                                       SUBPOOL BUFFER
SIZE         4096
                                                                                       TOTAL BUFFERS IN
SUBPOOL    1000

                                                                               17:08:15           17:10:16

    NUMBER OF LOCATE-TYPE CALLS                                                 1117674
1676213             558539
    NUMBER OF REQUESTS TO CREATE NEW BLOCKS                                           0
0                  0
    NUMBER OF BUFFER ALTER CALLS                                                 215874
322936             107062
    NUMBER OF PURGE CALLS                                                         25077
37454              12377
    NUMBER OF LOCATE-TYPE CALLS, DATA ALREADY IN OSAM POOL                       870306
1301187             430881
    NUMBER OF BUFFERS SEARCHED BY ALL LOCATE-TYPE CALLS                         1258247
1886843             628596
    NUMBER OF READ I/O REQUESTS                                                  238165
360260             122095
    NUMBER OF SINGLE BLOCK WRITES BY BUFFER STEAL ROUTINE                             0
0                  0
    NUMBER OF BLOCKS WRITTEN BY PURGE                                             95057
142413             47356
    NUMBER OF LOCATE CALLS WAITED DUE TO BUSY ID                                    780
```

```
1297              517
    NUMBER OF LOCATE CALLS WAITED DUE TO BUFFER BUSY WRT                        0
0                 0
    NUMBER OF LOCATE CALLS WAITED DUE TO BUFFER BUSY READ                       0
0                 0
    NUMBER OF BUFFER STEAL/PURGE WAITED FOR OWNERSHIP RLSE                    178
261               83
    NUMBER OF BUFFER STEAL REQUESTS WAITED FOR BUFFERS                          0
0                 0
    TOTAL NUMBER OF I/O ERRORS FOR THIS SUBPOOL                                 0
0                 0
    NUMBER OF BUFFERS LOCKED DUE TO WRITE ERRORS                                0
0                 0
```

The meaning of the various fields in the report is as follows:

FIX PREFIX/BUFFERS
: This field indicates the fix options for the buffer prefix/data buffers for this subpool.

SUBPOOL ID
: This field is a 4 character pool ID provided at subpool definition time.

SUBPOOL BUFFER SIZE
: This field indicates the size, in bytes, of the buffers in this subpool.

TOTAL BUFFERS IN SUBPOOL
: This field indicates the total number of buffers in this subpool.

On the following line are time entries that indicate the start trace and end trace times. The start trace and end trace fields tell you the time when the DB Monitor program was last started and stopped.

NUMBER OF LOCATE-TYPE CALLS
: This field indicates the number of locate-type calls for this subpool.

NUMBER OF REQUESTS TO CREATE NEW BLOCKS
: This field indicates the number of times a block had a segment inserted for the first time. When this happens, the block is marked as modified and must eventually be written back to the database.

NUMBER OF BUFFER ALTER CALLS
: This field indicates the number of buffer alter calls for this subpool. This count includes NEW BLOCK and BYTALT calls.

NUMBER OF PURGE CALLS
: This field indicates the number of purge requests for this subpool.

NUMBER OF LOCATE-TYPE CALLS, DATA ALREADY IN SUBPOOL
: This field indicates the number of locate-type calls for this subpool where the data was already in an OSAM pool.

NUMBER OF BUFFERS SEARCHED BY ALL LOCATE-TYPE CALLS
: This field indicates the number of buffers searched by all locate-type calls for this subpool.

NUMBER OF READ I/O REQUESTS
: This field indicates the number of read I/O requests for this subpool.

NUMBER OF SINGLE BLOCK WRITES BY BUFFER STEAL ROUTINE
: This field indicates the number of single block writes initiated by buffer steal routine for this subpool.

NUMBER OF BLOCKS WRITTEN BY PURGE
: This field indicates the number of blocks for this subpool written by purge.

TOTAL NUMBER OF I/O ERRORS FOR THIS SUBPOOL
: This field indicates the total number of I/O errors for this subpool.

NUMBER OF BUFFERS LOCKED DUE TO WRITE ERRORS
: This field indicates how many buffers are currently "frozen" in storage because a permanent I/O error occurred when writing them to the database. When a write operation results in a permanent I/O error, the affected buffers are frozen in storage until the data set is closed or, in an online system, until the system is shut down. Once the data set is closed, or the online system shut down, the buffers are written to the log, and this number returns to 0.

NUMBER OF LOCATE CALLS WAITED DUE TO BUSY ID
: This field indicates the number of locate calls for this subpool which waited due to busy ID.

NUMBER OF LOCATE CALLS WAITED DUE TO BUFFER BUSY WRT
: This field indicates the number of locate calls for this subpool which waited due to buffer busy writing.

NUMBER OF LOCATE CALLS WAITED DUE TO BUFFER BUSY READ
: This field indicates the number of locate calls for this subpool which waited due to buffer busy reading.

NUMBER OF BUFFER STEAL/PURGE WAITED DUE TO BUFFER BUSY READ
: This field indicates the number of locate calls for this subpool which waited for ownership to be released.

NUMBER OF BUFFER STEAL REQUESTS WAITED FOR BUFFERS
: This field indicates the number of buffer steal requests for this subpool which waited because no buffers were available to be stolen.

Program-I/O report

The Program-I/O report tells how long IMS was inactive (no IMS code was being executed) because IMS was waiting for use of a resource or for completion of an event. This time is called IWAIT time and, for all practical purposes, can be considered the time IMS had to wait while an I/O operation took place.

One report is produced each time the DB Monitor is run, and the report describes IWAIT time by PCB and data set name.

All times in the Program-I/O report are in microseconds. A microsecond is one millionth of a second (in other words, 7050 microseconds equals 0.007050 seconds).

Using the Program-I/O report

Using the Program-I/O report, you can correlate IWAIT time with a specific PCB and data set. This allows you to identify databases that are causing a relatively large number of IWAITs.

Because IWAITS are identified by PCB and data set names, you might be able to trace the large number of IWAITs back to a particular application program. If you can, give the application program special attention when tuning for performance. The DL/I-Call-Summary report, described in the following topic, helps you identify specific DL/I calls in an application program that are causing a large number of IWAITs.

Fields in the Program-I/O report

The following example shows a sample of a Program-I/O report.

```
IMS MONITOR   ****PROGRAM I/O****        TRACE START 1989 076  12:42:54     TRACE STOP  1989 076
12:43:07  PAGE  0008
                                    .........IWAIT TIME..........                              DISTR.
         PCB NAME      IWAITS       TOTAL        MEAN     MAXIMUM   DDNAME    MODULE          NO.
         ___ ____      ______       _____        ____     _______   ______    ______          ___
         DLVNTZ02           1       35853       35853       35853   DBHVSAM2   VBH            127
                            4      257649       64412      196028   HIDAM      VBH            128
                            2       79222       39611       62452   XDLBT04I   VBH            129
        PCB TOTAL
        ___ _____           7      372724       53246
         DLVNTZX2           2       57645       28822       40686   HIDAM      VBH            130
                            8      176622       22077       46141   DBHVSAM1   VBH            131
                            6      105340       17556       27843   XDLBT04I   VBH            132
                            3       65296       21765       23458   DBHVSAM2   VBH            133
        PCB TOTAL
        ___ _____          19      404903       21310
BATCH TOTAL
_____ _____                26      777627       29908
```

The meaning of the various fields in the report is as follows:

TRACE START and TRACE STOP
: The trace start and trace stop fields tell you the time when the DB Monitor program was last started and stopped. The time is generated by the time-of-day clock. Clock times are read as follows:

```
Clock time = hh.mm.ss
```

Where:

hh
Hours 0 through 23

mm
Minutes

ss
Seconds

If the DB Monitor program was on during an entire batch run, the trace start and trace stop times is when the batch run started and stopped. If the DB Monitor program was turned on and off more than once in the same batch run, the trace start and trace stop times are the times at which the monitor was last started and stopped.

PCBNAME
This field describes the name of the PCB the report is providing information about. Remember that each application program has one or more PCBs. If PCB names are unique to application programs, this field can only be used to identify which application program the report is providing information about.

IWAITS
This field describes the number of times IMS was inactive.

IWAIT Time
This field describes the elapsed time during which IMS was inactive.

TOTAL
The total time IMS waited

MEAN
The average time IWAIT IMS waited

MAXIMUM
The longest single time IMS waited

DDNAME
This field describes the name of the data set the application program is using. Within one report, an application program (PCB) can access more than one data set. The statistics compiled are listed separately for each data set.

MODULE
This field describes the modules that issued the internal call (in response to a DL/I call) that caused IMS to wait.

DBH
Module DFSDBHR0

DLE
Module DFSDDLE0

VBH
Module DFSDVSM0

DISTR.NUMBER
The distribution number is a reference number to be used in conjunction with the Distribution-Appendix report. For a description of what it means, see "Distribution-Appendix report" on page 590.

PCB Total
For a given PCB this field describes:

IWAITS
The total number of times IMS was inactive

TOTAL
The total time IMS waited

MEAN
The average time per IWAIT

BATCH Total
For this execution of the DB Monitor this field describes:

IWAITS
The total number of times IMS was inactive

TOTAL
The total time IMS waited

MEAN
The average time per IWAIT

DL/I-Call-Summary report

One DL/I-Call-Summary report is produced each time the DB Monitor is run, and the report describes times by PCB name and type of DL/I call.

The DL/I-Call-Summary report tells two things:

- How long IMS was inactive (no IMS code was being executed) because IMS was waiting for use of a resource or for completion of an event. This time is called IWAIT time and, for all practical purposes, can be considered the time IMS had to wait while an I/O operation took place.
- Of the elapsed time during which IMS was inactive, how much of it was actually IWAIT time.

All times in the DL/I-Call-Summary report are in microseconds. A microsecond is one millionth of a second (in other words, 7050 microseconds equals 0.007050 seconds).

Using the DL/I-Call-Summary report

The primary usefulness of the DL/I-Call-Summary report is to track down DL/I calls that are causing a large number of IWAITs.

If the number in the IWAITS/CALL field is relatively high, you want to know why. Because the number in the report is related to a specific DL/I call, segment, and PCB, you can trace the IWAITs back to a specific part of an application program. In addition, by using the Distribution-Appendix report, you can see how many of the DL/I calls being issued are distorting the average in the IWAITS/CALL field. See "Distribution-Appendix report" on page 590 for additional ways in which information from the DL/I-Call-Summary report can be used.

Remember, in tuning to decrease I/O operations, pay most attention to calls issued a large number of times. It is more profitable to save 1 second on a call executed 2000 times than to save 5 seconds on a call executed ten times (2000 versus 50 seconds).

Fields in the DL/I-Call-Summary report

The following example shows a sample DL/I-Call-Summary report.

```
IMS MONITOR   ****DL/I CALL SUMMARY****      TRACE START 1989 076  12:42:54       TRACE STOP 1989 076  2:43:07 PAGE 0009
                                                              (C)                (A)                       (B)
                  CALL  LEV          STAT    DL/I            IWAITS/       ..ELAPSED TIME...       .NOT IWAIT TIME..
DISTRIB.
         PCB NAME FUNC  NO.SEGMENT   CODE   CALLS     IWAITS   CALL          MEAN      MAXIMUM       MEAN      MAXIMUM
NUMBER
         --- ---- ----  ----------   ----   -----     ------   ----          ----      -------       ----      -------
------
         DLVNTZ02 STAT  (00)         GA         1          1    1.00        968281     968281      932428     932428
1 A,B,C
                  GN    (00)         GB         7          0    0.00          5703      29519        5703      29519
4 A,B,C
                  GN    (03)K1Z                 6          0    0.00           165        253         165        253
5 A,B,C
                  GN    (02)K8K5     GA         8          0    0.00           263        888         263        888
6 A,B,C
                  GN    (03)K1Z      GK         3          0    0.00          6844      20272        6844      20272
```

```
7 A,B,C
                  GN   (03)K6Y                   3         0     0.00         140         173         140         173
8 A,B,C
                  GN   (03)K5YK1     GA          3         0     0.00         197         219         197         219
9 A,B,C
                  GN   (04)K1Y       GK          5         0     0.00         306         892         306         892
10 A,B,C
                  GN   (04)K42                   5         0     0.00         121         173         121         173
11 A,B,C
                  GN   (03)K5XK2     GK          5         0     0.00        9951       41900        9951       41900
12 A,B,C
                  GN   (03)K6                    7         0     0.00        1292        7219        1292        7219
13 A,B,C
                  GN   (02)K5        GA          9         0     0.00         160         220         160         220
14 A,B,C
                  GN   (04)K1Y                   7         0     0.00       12008       45991       12008       45991
15 A,B,C
                  GN   (03)K5XK2                 4         0     0.00        6333       20917        6333       20917
16 A,B,C
                  GN   (02)K5                    3         0     0.00         126         138         126         138
17 A,B,C
                  GN   (01)K1        GA          5         0     0.00        6773       23625        6773       23625
18 A,B,C
                  GN   (03)K5YK1                10         0     0.00        9444       30320        9444       30320
19 A,B,C
                  GN   (04)K6X       GK          4         0     0.00         141         165         141         165
20 A,B,C
                  GN   (04)K1X       GK          4         0     0.00        3278       12578        3278       12578
21 A,B,C
                  GN   (04)K4                    4         0     0.00         423        1342         423        1342
22 A,B,C
                  GN   (03)K3K5                  6         0     0.00        1360        6982        1360        6982
23 A,B,C
                  GN   (02)K2                    4         0     0.00         325         910         325         910
24 A,B,C
                  GN   (03)K3K5      GA          2         0     0.00         189         196         189         196
25 A,B,C
                  GN   (04)K1X                   2         0     0.00         134         142         134         142
26 A,B,C
                  GU   (01)K1                    9         0     0.00        3387       26518        3387       26518
27 A,B,C
                  GN   (02)K8K5                  3         0     0.00         179         207         179         207
48 A,B,C
                  GN   (02)K5        GE          1         0     0.00         116         116         116         116
49 A,B,C
                  GU   (03)K5YK1                 8         0     0.00        7752       26664        7752       26664
50 A,B,C
                  GNP  (02)K5        GE          2         0     0.00         108         120         108         120
51 A,B,C
                  GNP  (03)K5YK1                 5         0     0.00        7873       37362        7873       37362
52 A,B,C
                  GU   (04)K1Y                   1         0     0.00         669         669         669         669
56 A,B,C
                  GU   (04)K4                    4         0     0.00       11572       44675       11572       44675
59 A,B,C
                  GU   (02)K5                    6         0     0.00        9025       45363        9025       45363
61 A,B,C
                  GNP  (02)K2        GE          1         0     0.00          87          87          87          87
62 A,B,C
                  GNP  (03)K3K5                  2         0     0.00         165         167         165         167
63 A,B,C
                  GU   (02)K2                    3         0     0.00         431         566         431         566
64 A,B,C
                  GU   (03)K5XK2                 5         0     0.00       13570       52617       13570       52617
65 A,B,C
                  GU   (04)K6X                   3         0     0.00        4892       13242        4892       13242
66 A,B,C
                  GU   (03)K6                    2         0     0.00        1247        1810        1247        1810
67 A,B,C
                  GU   (04)K42                   3         0     0.00         998        1390         998        1390
68 A,B,C
                  DLET (04)K42                   2         0     0.00       46374       49018       46374       49018
72 A,B,C
                  DLET (04)K6X                   2         0     0.00      108321      118802      108321      118802
73 A,B,C
                  DLET (02)K2                    1         0     0.00       44789       44789       44789       44789
74 A,B,C
                  DLET (03)K3K5                  3         0     0.00       87567      121071       87567      121071
75 A,B,C
                  GU   (02)J5        GE          1         0     0.00        1644        1644        1644        1644
111 A,B,C
                  DLET (03)J6                    1         0     0.00       35633       35633       35633       35633
112 A,B,C
                  GU   (02)J9        GE          1         0     0.00         705         705         705         705
113 A,B,C
                  DLET (04)J7J9                  1         0     0.00      245921      245921      245921      245921
114 A,B,C
                  REPL (03)J7PJ6     DA          1         0     0.00         225         225         225         225
115 A,B,C
                  BATCH TOTAL
                  ----- -----              392        22     0.05       22310                   20983
```

The meaning of the various fields in the report is as follows:

TRACE START and TRACE STOP
The trace start and trace stop fields tell you the time when the DB Monitor program was last started and stopped. The time is generated by the time-of-day-clock. Clock times are read as follows:

```
Clock time = hh.mm.ss
```

where:

hh
Hours 0 through 23

mm
Minutes

ss
Seconds

If the DB Monitor was on during an entire batch run, the trace start and trace stop time is when the batch run started and stopped. If the DB Monitor was turned on and off more than once in the same batch run, the trace start and trace stop times are when the monitor was last started and stopped.

PCBNAME
This field describes the name of the PCB the report is providing information about. Remember that each application program has one or more PCBs. If PCB names are unique to application programs, this field can only be used to identify which application program the report is providing information about.

CALL FUNC
This field describes the type of DL/I call the application program issued.

LEV NO.
This field describes the level that was accessed in the hierarchy of the database record to perform the DL/I call. If this field contain zeros, it means position was not established and therefore no level was set. This generally happens when a DL/I call cannot be processed for some reason.

SEGMENT
This field describes the 8-character name of the segment accessed by the DL/I call.

STAT CODE
This field describes the status code returned after the call (if the status code was not blank).

DL/I Calls
This field describes how many times this particular DL/I call was issued and had the five unique characteristics listed in the previous five columns.

IWAITS
This field describes the number of times IMS was inactive.

IWAITS/CALL
This field describes, by DL/I call, the average number of times IMS was inactive.

ELAPSED TIME
This field describes, by DL/I call, the elapsed time for calls.

MEAN
The average elapsed time per DL/I call

MAXIMUM
The longest elapsed time for a single DL/I call

NOT IWAIT TIME
This field describes, by DL/I call, the elapsed time minus the IWAIT time.

MEAN
The average NOT IWAIT TIME

MAXIMUM
The longest single NOT IWAIT TIME

NOT IWAIT TIME includes any time spent by higher priority tasks running in the IMS region. NOT IWAIT TIME might be about equal to total processor time if the IMS database region is the high priority task and no low priority tasks are causing interrupts.

DISTRIB.NUMBER
The distribution number is a reference number to be used in conjunction with the Distribution-Appendix report. For a description of what it means, see "Distribution-Appendix report" on page 590.

C, A, and B
These letters over the IWAITS/CALL, ELAPSED TIME, and NOT IWAIT TIME columns are reference letters to be used in conjunction with the Distribution-Appendix report. For a description, see "Distribution-Appendix report" on page 590.

BATCH Total
For this execution of the DB Monitor this field describes:

DL/I Calls
The total number of DL/I calls

IWAITS
The total number of times IMS was inactive

IWAITS/CALL
The average number of IWAITS per DL/I call

Distribution-Appendix report

Use a Distribution-Appendix report when you suspect some unusual combination of events has occurred, and the times or totals on the Program-I/O or DL/I-Call-Summary report do not give you enough information to highlight the problem.

The Distribution-Appendix report takes specific events for which a time or a total has been generated and distributes the events across ranges. It does this for specific events from the Program-I/O report and the DL/I-Call-Summary report.

To get an idea of when the Distribution-Appendix report is useful, see the following example. Suppose in a DL/I-Call-Summary report, the row of information in the example appears:

```
                                        (C)
PCB        CALL      LEV NO.       DL/I  IWAITS          DISTRIB.
NAME       FUNC      SEGMENT       CALLS CALL     ...    NUMBER
----       ----      -------       ----- ------  ----    --------
DHVBT203   DLET      (01)A1111111  11    8.63            11C
```

The (C) over the IWAITS/CALL field means that this event is detailed on the Distribution-Appendix report. The DISTRIB. NUMBER field says this event is broken down on the Distribution-Appendix report specifically on line 11 C. In this example, the IWAITS/CALL field value of 8.63 is a very high number relative to the other IWAITS/CALL numbers on the DL/I-Call-Summary report. For more information, check line 11 C in the Distribution-Appendix report. It looks like this:

```
11 C ....0....0....1....2....3....4....5....6....7....8....INF
           0    0    0    0    0    2    4    1    3    1
```

The numbers next to line 11 C (0 to INF) are predefined ranges. The numbers beneath line 11 C are the distribution of the event. Look for more information about why the IWAITS/CALL total is high (for DLET calls issued under PCB DHVBT203 against segment A1111111). The distribution of numbers beneath line 11 C says that of the 11 DLET calls issued:

- Two calls caused 5 IWAITs
- Four caused 6 IWAITs
- One caused 7 IWAITs
- Three caused 8 IWAITs

- One caused more than 8 IWAITs

Because one call caused more than 8 IWAITs, this call might be distorting the average in the IWAITS/CALL field in the report. Tune your database to eliminate relatively high IWAITs per DL/I call, investigate the call that required more than 8 IWAITS. In this example, the interval between distributions is 1. For other entries (for example, line 4B in the report in the following example), the intervals are much larger than 1. In these cases, interpret the data as (using line 4B): 1 call fell in the range of 16000 to 32000.

The example is of a Distribution-Appendix report. To read it, remember:

- The lines in the Distribution-Appendix report are always identified by a number (55, 56, and so on) or a number and a character (3A, 3B, and so on). The numbers are the numbers used in the DISTRIBUTION NUMBER fields in the Program-I/O or DL/I-Call-Summary report. The characters are used in the DL/I-Call-Summary report to identify which event is being distributed.
- The numbers next to the line are predefined ranges across which an event can be distributed. These ranges are made appropriate to the event. For example, ranges 0, 1, 2, 3, and so on are appropriate for distributing IWAIT totals for DL/I calls. (In the example used, we wanted to know how many calls required 0, 1, 2, 3, and so on IWAITs.) Ranges such as 0, 1000, 2000, and so on are appropriate for all other events in the Program-I/O and DL/I-Call Summary reports because all other events are times, not totals. These 0, 1000, 2000, and so on numbers are in microseconds. A microsecond is one millionth of a second (in other words, 2000 microseconds equals 0.002000 second). The ranges have default values, but you can redefine the ranges. See "Redefining the default ranges" on page 593 for more information.
- The number below the line is always the number for the event being distributed.

```
IMS MONITOR   ****DISTRIBUTION APPENDIX****    RACE START 1989 076  12:42:54     TRACE STOP 1989 076
12:43:07 PAGE 0014

#
1A..........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                    0          0          0          0          0          0          0
0          0          1

1B..........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                    0          0          0          0          0          0          0
0          0          1

1C..........0..........0..........1..........2..........3..........4..........5..........6..........7....
......8....INF
                    0          1          0          0          0          0          0
0          0          0
#
4A..........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                    5          0          0          1          0          1          0
0          0          0

4B..........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                    5          0          0          1          0          1          0
0          0          0

4C..........0..........0..........1..........2..........3..........4..........5..........6..........7....
......8....INF
                    7          0          0          0          0          0          0
0          0          0
#
5A..........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                    6          0          0          0          0          0          0
0          0          0

5B..........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                    6          0          0          0          0          0          0
0          0          0

5C..........0..........0..........1..........2..........3..........4..........5..........6..........7....
......8....INF
                    6          0          0          0          0          0          0
```

```
0          0          0
#
6A.........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                      8          0          0          0          0          0          0
0          0          0

6B.........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                      8          0          0          0          0          0          0
0          0          0

6C.........0..........0..........1..........2..........3..........4..........5..........6..........7....
......8....INF
                      8          0          0          0          0          0          0
0          0          0
#
7A.........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                      2          0          0          0          0          1          0
0          0          0

7B.........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                      2          0          0          0          0          1          0
0          0          0

7C.........0..........0..........1..........2..........3..........4..........5..........6..........7....
......8....INF
                      3          0          0          0          0          0          0
0          0          0
#
8A.........0.......1000.......2000.......4000.......8000......16000......32000......64000.....128000....
.256000....INF
                      3          0          0          0          0          0          0
0          0          0
⋮
#
131..........0.......2000.......8000......24000......50000.....100000.....150000.....200000.....250000..
...300000....INF
                      0          1          5          2          0          0          0
0          0          0
#
132..........0.......2000.......8000......24000......50000.....100000.....150000.....200000.....250000..
...300000....INF
                      0          0          5          1          0          0          0
0          0          0
#
133..........0.......2000.......8000......24000......50000.....100000.....150000.....200000.....250000..
...300000....INF
                      0          0          3          0          0          0          0
0          0          0
```

The TRACE START and TRACE STOP fields at the top of the report tell you the time when the DB Monitor was last started and stopped. The time is generated by the time-of-day clock. Clock times are read as follows:

```
Clock time = hh.mm.ss
```

Where:

hh
Hours 0 through 23

mm
Minutes

ss
Seconds

If the DB Monitor was on during an entire batch run, the trace start and trace stop times are the times at which the batch run started and stopped. If the DB Monitor was turned on and off more than once in the same batch run, the trace start and trace stop times are the times at which the monitor was last started and stopped.

How to generate the Distribution-Appendix report

The Distribution-Appendix report is not generated automatically when the DB Monitor is run. To get the Distribution-Appendix report, you have to include a DIS input control statement in the analysis control data set.

Events that can be distributed and their default ranges

Certain events, such as the IWAIT time for OSAM, can be distributed in the Program-I/O and DL/I-Call-Summary reports.

The following table shows the events that can be distributed in the Program-I/O and DL/I-Call-Summary reports. Each event has an ID.

Table 52. Events that can be distributed and their IDs

Events that can be distributed	ID
Program-I/O report	
IWAIT time for OSAM	D23
IWAIT time for VSAM	D24
IWAIT time for HSAM	D34
Elapsed time per DL/I call	D11
Not IWAIT time per DL/I call	D12
IWAITs per DL/I call	D13

The following table shows, using this ID, what each predefined set of ranges consists of when the default ranges are used. Notice that the first number in a range always defaults to zero, and the last number always defaults to infinity (INF).

Table 53. Predefined ranges when the default is used

ID	Default ranges
D11,D12	0,1000,2000,4000,8000,16000,32000,64000,128000,256000,INF
D13	0,0,1,2,3,4,5,6,7,8,INF
D23,D24	0,2000,8000,24000,50000,100000,150000,200000,250000,300000,INF
D34	0,2000,4000,8000,16000,32000,64000,96000,128000,160000,INF

Redefining the default ranges

You can redefine the default ranges. To do this, you have to include an input control statement in the analysis control data set for each default range you want to override.

Related reading: The analysis control data set is explained under "IMS Monitor Report Print utility (DFSUTR20)" in *IMS Version 15.2 Database Utilities*.

To override default ranges, you specify control statements in the form Dn n1,n2,...where:

- Dn is the ID of the event to be distributed (D23, for example, is the ID for IWAIT time for the OSAM event in the Program-I/O report)
- n1 is a specific default value. Each of the 10 default values or buckets can be redefined. For example, the Program-I/O report IWAIT time for OSAM could be redefined as follows:

```
D23  0,500,1000,1500,2000,4000,,,100000,500000
```

This would result in the 7th and 8th ranges remaining at their default values (150000 and 200000) and the final range (INF) remaining at its default value.

Monitor-Overhead report

This report describes the overhead required to run the DB Monitor. Because the DB monitor runs while IMS is executing, the monitor's use of resources causes the performance to be slightly less than it is when the monitor is not on.

Fields in the Monitor-Overhead report

The following example shows a sample of a Monitor-Overhead report.

```
IMS MONITOR   ****MONITOR OVERHEAD****        TRACE START 1989 076  12:42:54     TRACE STOP 1989 076
12:43:07 PAGE 0013
MONITOR OVERHEAD
DATA
M
---------------------
                 M
      13144 MILLISECONDS, TRACE INTERVAL
        550 MILLISECONDS, MONITOR MODULE TIME
        853 MONITOR RECORDS WERE PRODUCED
        645 MICROSECONDS PER MONITOR ENTRY
```

The meaning of the various fields in the report is as follows:

TRACE START and TRACE STOP
The trace start and trace stop fields tell you the time when the DB Monitor program was last started and stopped. The time is generated by the time-of-day clock. Clock times are read as follows:

```
Clock time = hh.mm.ss
```

Where:

hh
Hours 0 through 23

mm
Minutes

ss
Seconds

If the DB Monitor was on during an entire batch run, the trace start and trace stop times are the times at which the batch run started and stopped. If the DB Monitor was turned on and off more than once in the same batch run, the trace start and trace stop times are the times at which the monitor was last started and stopped.

MILLISECONDS, TRACE INTERVAL
This field describes how long the monitor was turned on.

MILLISECONDS, MONITOR MODULE TIME
This field describes, during the time the monitor was turned on, how long code in the monitor was actually being executed. (During this time, no IMS code can be executed, so this field tells the real overhead for using the monitor.)

MONITOR RECORDS WERE PRODUCED
This field describes how many records the monitor wrote.

MICROSECONDS PER MONITOR ENTRY
This field describes the average length of time it took the monitor to write a record.

Chapter 48. IMS Monitor reports

The IMS Monitor output reflects the IMS DB/DC environment as a whole. A subset of the reports deals with database calls and buffering.

Interpreting this data for batch application performance and to verify database design are considered separate tasks allied with database administration.

Transaction flow and IMS Monitor events

For an overall picture of the events, system activities, and usage of storage areas (buffer pool or data set) for which the IMS Monitor gathers timings, see the following table.

The leftmost column shows, from top to bottom, a sequence of processing events; each event is related to IMS Monitor reported items in the notes that follow the table.

Table 54. Transaction flow and IMS Monitor events description

Flow	Event	Activity	Pool	Data set
1	Wait for poll	Waiting	N/A	N/A
2	Data Transfer	N/A	N/A	N/A
3	Input message processing	Device	CIOP	(For lines)
		MFS	MFP	IMS.FORMATx
		Enqueuing	QBUF	Message queue
4	Input queuing	Waiting	N/A	N/A
5	Scheduling	Scheduling	QBUF	Message queue
		PSB load	PSB, PSBW, DPSB	IMS.ACBLIBx
		DMB load	DMBP	IMS.ACBLIBx
6	Program load	Program load		IMS.PGMLIB
7	Program initialization	Initializing	N/A	N/A
8	Message queue GU	Primed GU call	N/A	N/A
9	Program Execution	DC calls	QBUF	Message queue
	DL/I Elapsed	DB calls	OSAM/VSAM	N/A
	Wait Elapsed	DB I/Os	OSAM/VSAM	Databases
		SPA insert	CWAP	IMS.SPA
10	Output message insert	DC ISRT call	QBUF	Message queue
		DC GU call	QBUF	Message queue
		Sync point (MODE=SNGL)	OSAM/VSAM	Databases
11	Wait for sync point	(Only for MODE=MULT)	N/A	N/A
12	Program termination	Program termination	OSAM/VSAM	Databases
13	Wait for selection	Waiting	QBUF	Message Queue

Table 54. Transaction flow and IMS Monitor events description (continued)

Flow	Event	Activity	Pool	Data set
14	Output message processing	Message send	QBUF	Message queue
		MFS	MFP	IMS.FORMATx
		Device	CIOP	(For lines)
15	Data transfer	N/A	N/A	N/A
16	Output queue processing	Dequeuing	QBUF	Message queue

Notes:

1. The time waiting for poll is not recorded by the IMS Monitor.
2. Line activity in terms of data transmitted and IMS communication sub-task activity are recorded for all messages. Input message activity is not separated from output message activity.
3. During input message processing, the device dependent processing and the use of the communication I/O pool (CIOP) are recorded. If Message Format Service (MFS) is required, the use of the message format buffer pool and any I/O to the active IMS.FORMATA/B library are recorded. The input message is then placed in a message queue buffer (QBUF) with possible I/O to the message queue data sets. The IMS Monitor reporting does not distinguish this activity from output message processing.
4. Time spent waiting on the input queue is not recorded by the IMS Monitor.
5. IMS schedules the processing program into a region and, as part of this action, accesses the message queue or the QBUF pool so that it can present the message to the program. At this time, the required PSB and physical database blocks are made available in the PSB pool and the DMB pool. If they are not already in the pool, they are retrieved from the active IMS.ACBLIBA/B data set. These events are summarized under SCHEDULING AND TERMINATION in IMS Monitor reports.
6. Next, the application program is loaded into the region from IMS.PGMLIB, or initialized if already resident in the region. This processing is included as part of `SCHEDULE TO FIRST CALL` in IMS Monitor reports.
7. The initialization performed by the application program is included in this period of time, up to the time of the first message queue or database call. This processing is considered part of the scheduling process because it is not repeated when multiple transactions are processed for one scheduling event. This processing is recorded as part of the SCHEDULE TO FIRST CALL in IMS Monitor reports.
8. The event of performing the first DL/I GU call to obtain the first message segment is not separately recorded. The processing to prime the application with this message is included with the SCHEDULE TO FIRST CALL.
9. The program elapsed event is measured from the first call to termination of the processing program. The total elapsed time is recorded in IMS Monitor reports as ELAPSED EXECUTION.
   - Each DL/I call, whether DC or DB, is individually recorded along with its use of message queues or database data sets and their respective pools. Each external subsystem call is individually recorded. These events are recorded under CALLS in IMS Monitor reports.
   - When a DL/I call causes I/O activity, the time spent waiting for the data is recorded as WAIT time and as the number of I/Os. When a program's database processing intent and other update activity are in contention, this also contributes to WAIT time. For each external subsystem call, the time spent in external subsystem processing is recorded separately as WAIT time. When processing is suspended because of intent, the elapsed time is recorded in IDLE FOR INTENT IMS Monitor report items. When processing is suspended because the region is designated as wait-for-input and no input message is available, the time spent waiting for the next input message is excluded from elapsed time intervals and wait times. Wait-for-input time appears only in **WFI items on the program I/O report.

   - If the message processing is part of a conversational transaction, the activity in the communications work area pool (CWAP) and the IMS.SPA data set is recorded.
10. The DL/I ISRT call for the response to the message just processed is recorded for the transaction.

    As processing continues, there are synchronization points such as a GU to the message queue for another input message or a checkpoint call. The database and message queue I/O to commit the program processing is recorded. Checkpointing by IMS or by the processing program is recorded separately.
11. If the processing is for multiple messages before synchronization (MODE=MULT), there can be a wait for sync point. This time is not recorded by the IMS Monitor.
12. The processing after exiting from the application program, the program termination events, is included with the initial scheduling time. It is part of SCHEDULING AND TERMINATION.
13. The output message waits for the selection to be transmitted to the terminal. The duration of any wait time on the output queue is not recorded by the IMS Monitor.
14. After the exit from the application program and termination, output message processing events are recorded. These include message format and device dependent processing as well as sending the output message. The IMS Monitor reporting does not distinguish this activity from input message processing.
15. Line activity in terms of data transmitted and IMS communication sub-task activity are recorded for all messages. Output message activity is not separated from input message activity.
16. The processing to remove the output message from the queue is recorded.

IMS Monitor trace event intervals

The IMS Monitor trace interval is bounded by the master terminal operator's use of the **/TRACE** command between the start and stop command entries. The IMS Monitor can also be stopped by the expiration of any interval specified.

The online IMS events are recorded in IMS Monitor records placed in the IMSMON data set. The event timings are related to dependent region activity. The following figure shows the boundaries of the timed event intervals.

The Monitor trace interval includes the following intervals:

- Scheduling and Termination
  - No Messages
  - Block loader busy
  - Intent failures (exclusive intent and data sharing) and Schedule failures (PSB busy and space failure)
  - Sched/Term elapsed
    - NOT-WAIT
    - ACBLIB waits
    - DB Flush waits
    - DB CLOSE waits
- Region occupancy (which overlaps with all of Sched/Term elapsed)
  - Schedule to first call
  - Elapsed execution

The NOT-WAIT time for a region is the elapsed time not accounted for by wait time. Any delay coming from either paging or the processor being dispatched for a higher priority task results in an increase in the NOT-WAIT times.

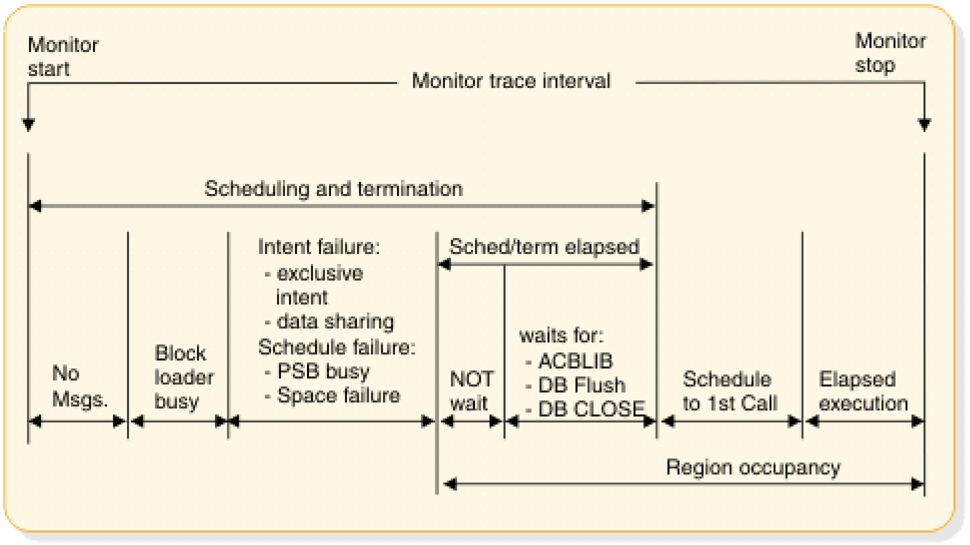


Figure 66. IMS Monitor trace event intervals

Overview of IMS Monitor reports

A list of reports available from data collected by the IMS Monitor, together with the principal performance data they contain, is shown in the following table.

Related reading: For those reports marked by a "DC", see Chapter 50, "IMS Monitor reports for DCCTL," on page 653.

The reports marked "MSC" in the list are only produced when MSC is active. The MSC reports and their interpretation are in the following table.

The order of the reports listed in the following table matches the sequence of the output from the IMS Monitor Report Print Program. The duration or constraints of a monitoring snapshot might not include certain events necessary for an individual report, in which case only report headings or partial data are produced.

Table 55. IMS Monitor reports output sequence and information

Report name	Principal information
System Configuration	Monitor run documentation
Message Queue Pool	Buffering and message I/O per transaction
Database Buffer Pool (DB)	Count of DB calls and I/O per transaction
VSAM Buffer Pool (DB)	Count of inserts and I/Os
Message Format Buffer Pool	Count of blocks fetched and I/Os
Latch Conflict Statistics	IMS internal processing
General Wait Time Events	Wait times for SNAPQ
Region and Jobname	Monitor run documentation
Region Summary	Elapsed times and count of DL/I calls
Region Wait	Wait times
Programs by Region	Elapsed times for region usage
Program Summary	Overall program statistics
Program I/O (DB)	Wait times/PCB

Table 55. IMS Monitor reports output sequence and information (continued)

Report name	Principal information
Communication Summary	Elapsed times for lines
Communication Wait	Wait times by line
Line Functions	Count and size of blocks transmitted
MSC Traffic (MSC)	Count and routing of transactions
MSC Summaries (MSC)	Count of transactions by destination
MSC Queuing Summary (MSC)	Count and queuing time by link
Transaction Queuing	Queue loading statistics
Reports	Count of space failures and deadlocks
Run Profile	Monitor run documentation
Call Summary (DB)	Call counts and timings/segment type
Distribution Appendix	Event frequency distributions

The majority of the data items in IMS Monitor reports are elapsed times. These are normally expressed in microseconds. An entry of 1876534 represents 1.876534 seconds or 1876 milliseconds. Any times that do not follow this convention show the unit of measure on the report.

You can also find counts of events under the heading OCCURRENCES, and some figures that represent the number of bytes.

Documenting the monitoring run

For each trace interval there are several general reports or overall summaries of the processing that took place. You can use these reports as part of your IMS Monitor run documentation.

It is important to record, as accurately as possible the conditions under which the trace was taken. Your documentation can include system status information obtained by the **`/DISPLAY`** command several times before and after the trace, an expected profile of the application program activity, and any desired processing events. The trace interval should represent typical processing loads and not be a biased or inadequate historical record.

Adding to the System-Configuration report data

The first general report titled SYSTEM CONFIGURATION is found under the page heading BUFFER POOL STATISTICS. It shows the modification level of the IMS and z/OS systems. The system configuration output is illustrated in the following section.

Recording the Monitor trace interval

The heading of most IMS Monitor reports carries the trace start and stop times. It is shown in the format `YEAR DAY (Julian) HH:MM:SS`. The overall length of the trace interval is given in milliseconds under the title MONITOR OVERHEAD DATA. The following line shows how many trace records were placed on the IMSMON data set. An example of the monitor trace interval recording is shown in the following example.

```
***IMS MONITOR***  BUFFER POOL STATISTICS   TRACE START 2009 180   5:55:15   TRACE STOP  2009 180
5:59:49  PAGE 0001

                   S Y S T E M   C O N F I G U R A T I O N

                                                                 SYSTEM CONFIGURATION       :  MVS/ESA
                                                                 IMS VERSION                :  11
```

```
                                                                            RELEASE LEVEL            :   1
                                                                            MODIFICATION NUMBER      :   0
```

Completing the Monitor run profile

A compact set of processing ratios is found at the end of the Run-Profile report. The statistics summarize, for the monitor interval, the transaction throughput and the degree of DL/I and I/O activity. An example of the run-profile report is shown in the following example.

The lower part of the Run-Profile report shows several ratios:

Program elapsed time to DL/I elapsed time for each region
DL/I elapsed time to wait time during DL/I processing
Program elapsed time to other subsystem call elapsed time
DL/I elapsed time to other subsystem call elapsed time

Each dependent region is identified by a sequence number, starting at region 1.

```
    IMS MONITOR   **RUN PROFILE**            TRACE START 2009 180   5:55:15     TRACE STOP  2009 180
5:59:49  PAGE 0184
TRACE ELAPSED TIME IN SECONDS.............274.6
TOTAL NUMBER OF MESSAGES DEQUEUED........1403
TOTAL NUMBER OF SCHEDULES.................173
NUMBER OF TRANSACTIONS PER SECOND...........5.1
TOTAL NUMBER OF DL/I CALLS ISSUED.......18632
NUMBER OF DL/I CALLS PER TRANSACTION.......13.2
NUMBER OF OSAM BUFFER POOL I/O'S........11236,    8.0 PER TRANSACTION
NUMBER OF MESSAGE QUEUE POOL I/O'S..........0,    0.0 PER TRANSACTION
NUMBER OF FORMAT BUFFER POOL I/O'S..........0,    0.0 PER TRANSACTION
RATIO OF PROGRAM ELAPSED TO DL/I ELAPSED:
              REGION   1:   1.09
              REGION   2:   1.09
              REGION   3:   1.00
              REGION   4:   1.02
              REGION   5:   1.01
              REGION   6:   1.00
              REGION   7:   1.00
              REGION   8:   1.00
              REGION   9:   1.17
              REGION  10:   1.00
              REGION  11:   1.00
              REGION  49:   1.03
              REGION  50:   1.19
RATIO OF DL/I ELAPSED TO DL/I IWAIT:
              REGION   1: 325.65
              REGION   2:  73.49
              REGION   4: 100.35
              REGION   5:  85.76
              REGION   6:  82.99
              REGION  47:  95.64
              REGION  48:  45.93
              REGION  49:   9.22
```

You can match the regions to the name of the z/OS job using the Region-and-Jobname report. The job names correspond to the step names on the EXEC statements of all the dependent regions started by the operator before the trace was started. The region job names are included on the monitor output page with the heading GENERAL REPORTS, as illustrated in "Detecting checkpoint effects" on page 613.

Some generalized processing ratios are given at the end of several buffer pool statistics reports. You can include them in the documented profile of the trace interval. These are not specific to one application or system resource but can be used as indicators of variation across a series of monitor runs.

The ratios are:

- The total number of OSAM reads + OSAM writes + all waits divided by the total number of transactions.

  From the Message-Queue-Pool report, this ratio indicates on a per transaction basis the physical I/O activity required to handle the message queuing function.
- The total number of OSAM reads + OSAM writes + BISAM reads divided by the total number of transactions.

From the Database-Buffer-Pool report, this ratio indicates on a per transaction basis the physical I/O activity required to handle the database buffering function.

- The total prefetch I/Os + immediate fetch I/Os + directory I/Os divided by the total number of transactions.

  From the Message Format Buffer Pool report, this ratio indicates on a per transaction basis the physical I/O activity required to handle the MFS function.

Verifying IMS Monitor report occurrences

When you examine the output from the IMS Monitor Report Print program, the presence of a report heading does not necessarily mean that appropriate data is listed.

System definition options and utility control statements affect the content of the output as follows:

- The output does not include a Call-Summary report unless a control statement specifies DLI.
- The output does not include a set of Distribution reports unless a control statement specifies DIS or DISTRIBUTION. The column headed DISTRIBUTION NUMBER that occurs on many of the reports contains cross-references to items included in the Distribution reports.
- The output consists of just a Call Summary report if a control statement specifies ONLY DLI.

Because many of the summary reports require system status to calculate the difference between start and end values, and this status is obtained during the /TRACE SET OFF processing, the IMS Monitor execution must end before termination of the IMS control region. If the trace was not stopped properly, the following message is issued:

```
NO QUEUE BUFFER POOL TRACES AT END TIME ON MONITOR LOG TAPE
****QUEUE BUFFER POOL REPORT CANCELLED****
```

Similarly, other summary reports are not produced.

The series of reports with the title BUFFER POOL STATISTICS do not include a VSAM BUFFER POOL section unless one of the databases in IMS.ACBLIB uses the VSAM access method. If VSAM is not used, the following message is issued:

```
NO VSAM BUFFER POOL TRACES ON MONITOR LOG TAPE
****VSAM BUFFER POOL REPORT CANCELLED****
```

The section MESSAGE FORMAT BUFFER POOL is included only if your system definition specifies devices using Message Format Service (MFS).

If the source data used to formulate a particular IMS Monitor report, or a section of that report, has not been recorded by the IMS Monitor during the trace interval, the report contains only the headings.

Monitoring activity in dependent regions

The IMS Monitor gathers timing information for every dependent region identified in the **/TRACE** command active during the trace interval. It records the total of the elapsed times for each event, the maximum individual time encountered, and the average time.

There are three major reports that display timings. The reports and a list of their content are:

- **Region-Summary Report**
  - Scheduling and termination
  - Schedule end to first call
  - Elapsed execution with separate summaries shown for:
    - DL/I calls
    - External subsystem service and command calls
    - External subsystem database access calls

  - Checkpoint processing
  - Region occupancy
- **Region Wait**
  - Waits during scheduling and termination
  - Waits during DL/I calls
  - Waits during external subsystem calls
  - Waits during checkpoint
- **Programs by Region**
  - Elapsed execution
  - Schedule end to first call

These three reports are illustrated in the following examples.

Activities for dependent regions are placed in five categories:

- Elapsed time for scheduling and termination

  The scheduling process includes many preparatory events such as block loading from an active IMS.ACBLIBA/B data set, and obtaining ownership of the PSB. The time required to terminate the region activity after the application program ends is also included.
- Elapsed time from end of schedule to first call

  This time is reserved for application program initialization and housekeeping prior to an initial call (to the message queue, a database, or an external subsystem) that marks the beginning of control program services. It is a measure of processing that is not repeated when multiple transactions are processed in a single scheduling.
- Program elapsed time, including all calls

  This time encompasses the major application program processing, measured from the first call to the return or exit from the program.
- Elapsed time performing DL/I calls

  This time includes all DL/I calls. Each DL/I call event is measured from the time of the call to the return to the application program.
- Elapsed time performing external subsystem calls

  This time includes all external subsystem calls. Each external subsystem event is measured from the time of the call to the return to IMS.

The following example shows a sample of the region-summary report.

```
     IMS MONITOR   ****REGION SUMMARY****             TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0011
                                              (A)                                  (B)
                                 ........ELAPSED TIME.........       NOT IWAIT TIME(ELAPSED-IWAIT)
DISTRIBUTION
               OCCURRENCES       TOTAL        MEAN     MAXIMUM       TOTAL        MEAN     MAXIMUM
NUMBER
               -----------       -----        ----     -------       -----        ----     -------
______
SCHEDULING AND TERMINATION
__________ ___ ___________
**REGION    5            5        4146         829         948        4146         829         948
287A,B
**REGION    6            7        6028         861        1067        6028         861        1067
214A,B
**REGION    8            8        6847         855        1098        6847         855        1098
129A,B
**REGION   10            7        9664        1380        3668        9664        1380        3668
272A,B
**REGION   47            6        5482         913        1021        5482         913        1021
145A,B
**REGION   49            3        2612         870         917        2612         870         917
443A,B
**TOTALS               123      126042        1024                  126042        1024
SCHEDULE TO FIRST CALL
________ __ _____ ____
**REGION    1            1    15479797    15479797    15479797
```

```
555
**REGION     2          1    22376350    22376350    22376350
564
**REGION     3          1    15169488    15169488    15169488
578
**REGION     4          1    48146258    48146258    48146258
584
**REGION    48          1      795351      795351      795351
592
**REGION    49          4     2960425      740106     2951746
442
**REGION    50          1    15713464    15713464    15713464
575
**TOTALS              168   514286738     3061230
ELAPSED EXECUTION
_______ _________
**REGION     1          1   290146255   290146255
290146255                                                    1
**REGION     2          1   252290108   252290108
252290108                                                    2
**REGION     3          1   259496970   259496970
259496970                                                    3
**REGION     4          1   322812716   322812716
322812716                                                    4
**REGION    48          1   273871107   273871107   273871107
48
**REGION    49          4   271703421    67925855   155176058
49
**REGION    50          1   290379922   290379922   290379922
50
**TOTALS              173 14238540145    82303700
DL/I CALLS                                                                                                   IWT/CALL(C)
____ _____                                                                                                   ___________
**REGION     1         60   264626241     4410437    88981490   263813671     4396894    88970053       0.76
247A,B,C
**REGION     2        223   230505269     1033655    61048758   227368742     1019590    61011153       0.73
237A,B,C
**REGION     3         29   257704383     8886358    69000514   257704383     8886358    69000514       0.00
98A,B,C
**REGION     4        792   313735347      396130    52439653   310609035      392183    52439653       0.22
180A,B,C
**REGION    49        592   262886317      444064    30202068   234394017      395935    30159782       2.46
177A,B,C
**REGION    50         36   242591451     6738651    48651260   242591451     6738651    48651260       0.00
289A,B,C
**TOTALS            18632 12386905286      664818             12024562411      645371                   0.97
IDLE FOR INTENT
____ ___ ______
                      NONE
CHECKPOINT
__________
                      NONE
REGION OCCUPANCY
______ _________
**REGION     1    100.0%
**REGION     2    100.0%
**REGION     3    100.0%
**REGION     4    100.0%
**REGION    48    100.0%
**REGION    49    100.0%
**REGION    50    100.0%
```

The following example shows a sample of the region-wait report.

```
     IMS MONITOR   ****REGION IWAIT****               TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0023
                                 .........IWAIT TIME..........                                  DISTRIBUTION
**REGION     5 OCCURRENCES       TOTAL        MEAN     MAXIMUM       FUNCTION  MODULE       NUMBER
  ______     ___________         _____        ____     _______       ________  ______       ______
SCHEDULING + TERMINATION
__________ + ___________
   SUB-TOTAL
   _________
   TOTAL
   _____
DL/I CALLS
____ _____
                        11      181816       16528       24375    DD=IMMSTR2A   DBH            117
                         8      112831       14103       17846    DD=IMMSTR1A   DBH            118
                         5       85460       17092       33717    DD=IMMSTR3A   DBH            119
                         5       58420       11684       14643    DD=IMINDEXA   VBH            120
                         1        4160        4160        4160    INT=DDLTRN24  BLR-64BIT       35
                         1        3623        3623        3623    PSB=BMPFPE02  BLR-64BIT       36
                        12      173866       14488       22152    DD=PRODCNTA   VBH            121
                         3      100576       33525       68373    DD=IMMSTR2B   DBH            428
                         1       17921       17921       17921    DD=IMMSTR3B   DBH            429
                         1       17195       17195       17195    DD=IMMSTR1B   DBH            430
                         1       13577       13577       13577    DD=IMINDEXB   VBH            431
                         3       49928       16642       20396    DD=PRODCNTB   VBH            432
                         4       10973        2743        2787    DD=ITEMACTB   DBH            453
                         2       37680       18840       27664    DD=IAINDEXB   VBH            454
                        49     1500067       30613      138284    DD=INVENTRA   DBH            472
```

```
                   23      345595       15025       27613    DD=VENDORDA   VBH            473
                    1      342952      342952      342952    PI=VENDORDA...1              498
                    1       14612       14612       14612    PI=VNSINDXA...1              499
                    6       69203       11533       19492    DD=VNSINDXA   VBH            500
   TOTAL
   _____          136     3132672       23034
```

The following example shows a sample of the programs-by-region report.

```
     IMS MONITOR   ****PROGRAMS BY REGION****         TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0069
                                                  (A)                               (B)
                                    ELAPSED EXECUTION TIME           SCHEDULING END TO FIRST CALL      DISTRIBUTION
              OCCURRENCES       TOTAL         MEAN     MAXIMUM       TOTAL         MEAN     MAXIMUM       NUMBER
              -----------       -----         ----     -------       -----         ----     -------       ------
**REGION     1
_________  _ 1
    PROGSC6D           1   290146255   290146255   290146255    15479797     15479797    15479797          885A,B
 REGION TOTALS         1   290146255   290146255                15479797     15479797
**REGION     2
_________  _ 2
    PROGIT8C           1   252290108   252290108   252290108    22376350     22376350    22376350          889A,B
 REGION TOTALS         1   252290108   252290108                22376350     22376350
**REGION     3
_________  _ 3
    PROGTS1C           1   259496970   259496970   259496970    15169488     15169488    15169488          893A,B
 REGION TOTALS         1   259496970   259496970                15169488     15169488
**REGION     4
_________  _ 4
    PROGPS3D           1   322812716   322812716   322812716    48146258     48146258    48146258          897A,B
 REGION TOTALS         1   322812716   322812716                48146258     48146258
**REGION     5
_________  _ 5
    PROGPS3A           2    62893103    31446551    40693590        5435         2717        2862          901A,B
    PROGTS1B           1    61794787    61794787    61794787        2790         2790        2790         1271A,B
    PROGPS3B           1    18294458    18294458    18294458        3104         3104        3104         1350A,B
    PROGIT2B           1    36095342    36095342    36095342        2731         2731        2731         1363A,B
    PROGSC2A           1    93902771    93902771    93902771     1667791      1667791     1667791         1401A,B
 REGION TOTALS         6   272980461    45496743                 1681851       280308
**REGION     6
_________  _ 6
    PROGIT1B           2    39000315    19500157    23703429        5286         2643        2801          905A,B
    PROGTS1B           1    34293636    34293636    34293636        3136         3136        3136         1207A,B
    PROGPS3A           1    51887767    51887767    51887767        2534         2534        2534         1278A,B
    PROGPS3B           2    67375031    33687515    40291430    17210570      8605285    17213287         1328A,B
    PROGIT8A           1    69132416    69132416    69132416        3291         3291        3291         1359A,B
    PROGSC4A           1    30165017    30165017    30165017        2571         2571        2571         1433A,B
 REGION TOTALS         8   291854182    36481772                17193752      2149219
**REGION     7
_________  _ 7
    PROGSC2B           1   269618583   269618583   269618583     5047875      5047875     5047875          909A,B
 REGION TOTALS         1   269618583   269618583                 5047875      5047875
**REGION     8
_________  _ 8
    PROGIT8A           1     5181039     5181039     5181039        2928         2928        2928          913A,B
    PROGPS3A           1    27304257    27304257    27304257        3350         3350        3350         1132A,B
    PROGSC4B           1    37286872    37286872    37286872        3009         3009        3009         1255A,B
    PROGIT2A           1    36902995    36902995    36902995        2850         2850        2850         1298A,B
    PROGIT1B           1    30407479    30407479    30407479        2565         2565        2565         1336A,B
    PROGIT1A           3   109875360    36625120    45190114     4279008      1426336     4272096         1357A,B
    PROGIT8B           1    23405220    23405220    23405220        2679         2679        2679         1395A,B
 REGION TOTALS         9   270363222    30040358                 4296389       477376
```

Detecting database processing intent conflicts

The IMS Monitor records the intervals when a region is in an idle state waiting to update a database owned exclusively by another already scheduled application program.

You can see the total, maximum, and average idle times in IDLE FOR INTENT following the DL/I calls. The elapsed time during the unsuccessful scheduling of a program in that region is included in the summary line times for that region.

The region can fail to be scheduled even when ownership of that database is released. The number of times processing is held up by intent failure is separately tallied under the title INTENT FAILURE SUMMARY. The report is illustrated in the following example. In this report you can see which PSBs are in conflict because of exclusive intent for a segment type and the database name in question.

```
INTENT FAILURE SUMMARY
    PSBNAME        DMBNAME      OCCURRENCES
    SSTPSBNM       SSTDMBNM          1
    TOTAL                            1
```

Examining the effects of checkpoints

The checkpoint line of the Region Summary report at the end of the region-by-region summary, shows: the number of times that a system checkpoint was taken during the monitor interval, the elapsed times, and the not-wait times.

Checkpoint processing can be initiated by the control program at a specified frequency determined by the number of records placed on the system log. Other checkpoints can be caused by operator commands.

The wait time experienced during checkpoints is reported at the end of the first region summary on the Region Wait report. You can detect delays for each combination of DD name and module code. Typical entries here are for the message queue data sets and the restart data set. If a wait for storage is the cause, the entry under the FUNCTION column is STG.= followed by the identification of the pool.

Measuring region occupancy

A measure of region activity is the percentage of region occupancy. This is broadly the ratio of the elapsed time a region is performing processing to the trace interval.

The region occupancy time does not include those times when no messages are available, when the block loading is delayed, or when the PSB cannot be used. The last section in the Region Summary report lists all active regions for which timed events were collected and shows the calculated percentages of region occupancy.

Monitoring application program elapsed time

The IMS Monitor can record measurements of elapsed times for each transaction and scheduling of an application program. It does this during the monitored interval while other programs are executing concurrently.

Elapsed times are calculated from the start of the first DL/I (or other) call to the end of that program. You can distinguish between time spent in application code and in DL/I processing. The following figure illustrates the event intervals.

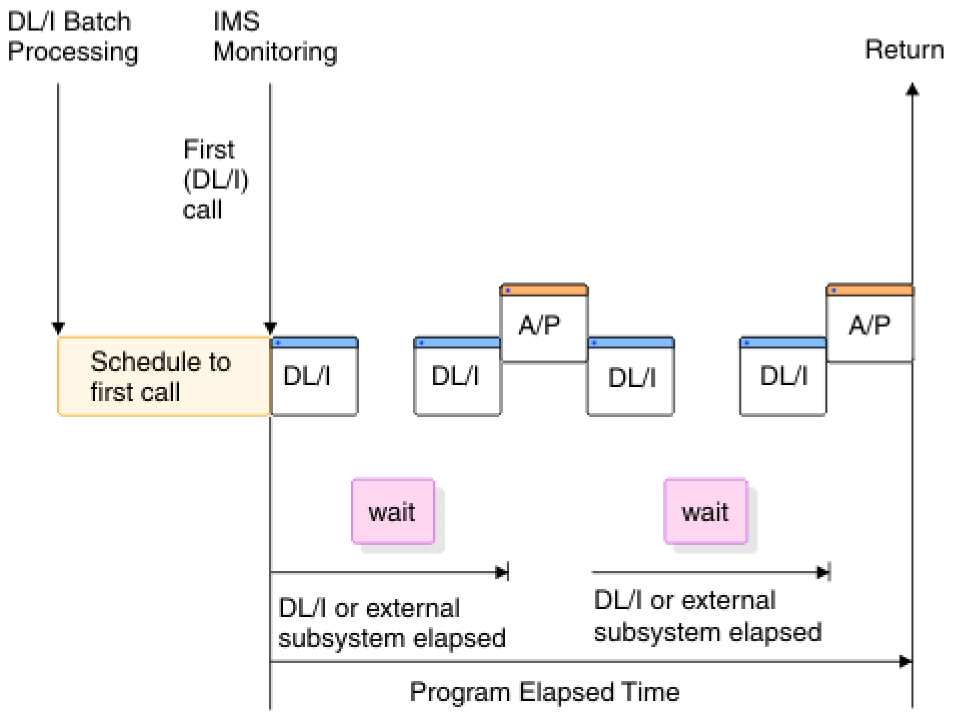


Figure 67. Event intervals for time in application code and DL/I processing

Within the elapsed time for a DL/I call, the wait time to obtain segment data is recorded separately. Similarly, within the elapsed time for an external subsystem call, the processing time in the external subsystem is recorded separately as the wait time. For regions designated as wait-for-input, the time spent waiting for input messages is excluded from values shown in the reports. The application processing (A/P) time includes many kinds of subsidiary service beyond the machine cycles expended by the program object code—such as subroutine loading, I/O to z/OS data sets, and any overlay processing. If the program is waiting to be dispatched or requires paging before it can use real storage, these delays are also accounted for in application program processing time. Because a program can execute many transactions for each schedule, the elapsed time from schedule to first call is recorded separately. This time covers the initialization performed by the application program and includes the time for loading the program.

The elapsed times are given in the Program Summary report. The following example is a sample of the report. Programs are identified by their PSB name on individual lines in the report. Each line gives a summary of the activity for that PSB during the measured interval. The total number of schedules, DL/I calls, transactions completed (dequeued), and waits for DL/I call I/O and external subsystem processing are given. The report line gives calculated average times for:

- Elapsed time per schedule
- Processor time per schedule
- Schedule to first DL/I call per schedule
- Elapsed time per transaction

The report also includes:

- Frequencies for calls per transaction
- I/O waits per DL/I call
- Waits per external subsystem call
- Transactions dequeued per schedule

A TOTALS line summarizes all activity for the PSBs active during the monitored interval. (The PSB DUMMY line reconciles any incomplete scheduling caused by a region stopping during scheduling or for a program that experiences a pseudo abend.)

```
     IMS MONITOR   ****PROGRAM SUMMARY****             TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0075
                                                              (A)........(B)........          (A)........(B)........
                                                       I/O   TRAN.       CPU       .        ELAPSED
SCHED.TO    .        ELAPSED
            NO.    TRANS.            CALLS     I/O   IWAITS  DEQD.      TIME    DISTR.       TIME      1ST CALL
DISTR.       TIME
PSBNAME  SCHEDS.    DEQ.     CALLS  /TRAN    IWAITS  /CALL  /SCH.     /SCHED.    NO.       /SCHED.     /SCHED.
NO.       /TRANS.
-------  -------    ----     -----  -----    ------  -----  -----     -------    ---       -------     -------
---       -------

PROGSC6D       1      13        60    4.6        46    0.7   13.0       10010   884A,B   290146255    15479797
886A,B    22318942
PROGIT8C       3      17       225   13.2       166    0.7    5.6       90592   888A,B   256617508    73283259
890A,B    45285442
PROGTS1C       2      25        47    1.8         0    0.0   12.5       10010   892A,B   239190808     7586234
894A,B    19135264
PROGPS3D       1      23       792   34.4       182    0.2   23.0       10010   896A,B   322812716    48146258
898A,B    14035335
PROGPS3A      13      36      1246   34.6       267    0.2    2.7       49782   900A,B    32801812     2228611
902A,B    11845098
PROGIT1B      11      21        99    4.7         0    0.0    1.9        6341   904A,B    23212388     2036217
906A,B    12158870
PROGSC2B       7     155      3068   19.7      1845    0.6   22.1      346112   908A,B    93655514      789390
910A,B     4229603
PROGIT8A      12      28       434   15.5       293    0.6    2.3       34350   912A,B    30196795     1745815
914A,B    12941483
PROGPS2C       1      10       179   17.9       205    1.1   10.0       10010   916A,B   221024429    53642029
918A,B    22102442
PROGTS1B       8      20        54    2.7         0    0.0    2.5        5447   920A,B    39943245        2895
922A,B    15977298
PROGPS3C       1      14       468   33.4       117    0.2   14.0       10010   924A,B   310644485    35978027
926A,B    22188891
PROGIT1C       1       9        32    3.5         0    0.0    9.0       10010   930A,B   304892631    30226173
932A,B    33876959
PROGSC2C       1       9       160   17.7       101    0.6    9.0       10010   934A,B   296909110    22242652
936A,B    32989901
PROGIT2B       8      21       393   18.7        63    0.1    2.6       21703   938A,B    35126671     1798496
940A,B    13381589
PROGIT2C       6      17       211   12.4        39    0.1    2.8       13312   942A,B   288883508    50698467
944A,B   101958885
```

```
PROGTS1D      2     26       50    1.9        0    0.0   13.0      10010   950A,B   284944505    10613350
952A,B    21918808
PROGPS3B      8     22      770   35.0      169    0.2    2.7      35737   954A,B    38016279     2149158
956A,B    13824101
PROGIT1A     11     24      106    4.4        0    0.0    2.1       7925   958A,B    30883486     1935855
960A,B    14154931
PROGSC4A      9    163     1775   10.8     5101    2.8   18.1     235921   963A,B    62172947     3011199
965A,B     3432862
PROGSC6C      1     10       44    4.4       38    0.8   10.0      10010   967A,B   228098334    46568124
969A,B    22809833
PROGPS2B     11     28      557   19.8      604    1.0    2.5      35069   971A,B    33309266     1181831
973A,B    13085783
PROGIT8D      1     12      175   14.5      133    0.7   12.0      10010   975A,B   253392289    21274169
977A,B    21116024
PROGSC4C      1     10       98    9.8      349    3.5   10.0      10010   979A,B   248736332    25930126
981A,B    24873633
PROGSC6A      7    157      789    5.0      457    0.5   22.4      11703   983A,B    73936039      115979
985A,B     3296511
PROGIT2A      7     22      430   19.5       71    0.1    3.1      28529   987A,B    37905001        2982
989A,B    12060682
PROGSC2D      1     15      280   18.6      180    0.6   15.0      10010   991A,B   316194222    41527764
993A,B    21079614
PROGPS2A      6     25      490   19.6      548    1.1    4.1      43177   995A,B    58277945     2467506
997A,B    13986707
PROGSC2A      5    121     2363   19.5     1420    0.6   24.2     276187  1001A,B    88906184     6022954
1003A,B    3673809
PROGIT2D      1     20      361   18.0       62    0.1   20.0      10010  1005A,B   386092737   111426279
1007A,B   19304636
PROGSC4B     10    131     1421   10.8     4115    2.8   13.1     617016  1011A,B    53826667     2632409
1013A,B    4108905
PROGSC4D      1     19      197   10.3      668    3.3   19.0      10010  1020A,B   227999124    46667334
1022A,B   11999953
PROGPS2D      1     13      240   18.4      291    1.2   13.0      10010  1025A,B   327602445    52935987
1027A,B   25200188
PROGSC6B      5    140      694    4.9      395    0.5   28.0      16884  1032A,B    78994223     3290769
1034A,B    2821222
PROGIT1D      1     10       36    3.6        0    0.0   10.0      10010  1041A,B   290379922    15713464
1043A,B   29037992
PROGIT8B      8     17      288   16.9      190    0.6    2.1      33436  1259A,B    35223857        2902
1261A,B   16575932
**TOTALS    173   1403    18632   13.2    18115    0.9    8.1      90328             82303700
2972755          10148638
```

You can use the Call-Summary report to examine the detail of the call processing for each program. The report is itemized by type of call and summarized for the monitor interval. An extract from the multipage output is given in the following example. The calls using an I/O PCB are given first and sub-totaled. Then, the total calls of each type, against each database PCB and each external subsystem, are listed. The PSB TOTAL line marks the end of data for each program.

```
     IMS MONITOR   ****CALL SUMMARY****              TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0186
                                                             (C)              (A)                      (B)
                  CALL  LEV         STAT                   IWAITS/       ..ELAPSED TIME...       .NOT IWAIT TIME..
DISTRIB.
PSB NAME PCB NAME FUNC  NO.SEGMENT  CODE   CALLS     IWAITS   CALL         MEAN      MAXIMUM       MEAN      MAXIMUM
NUMBER
--- ---- --- ---- ----  ----------  ----   -----     ------   ----         ----      -------       ----      -------
______
PROGSC6B I/O PCB  ISRT (  )                  138          0     0.00        372        1240         372
1240       598A,B,C
                  GU   (  )                  134        133     0.99    2600917    20974615     2587532
20962866       602A,B,C
                  (GU) (  )                    3          0     0.00         15          16          15
16       716A,B,C
                  ASRT (  )                    3          0     0.00        330         333         330
333       869A,B,C
                  GU   (  )         QC         2          1     0.50   17639806    21219588    17634776
21209529       870A,B,C
                  I/O PCB SUBTOTAL
                  ___ ___ ________           280        134     0.47    1370910                 1364469
         INVENTRB DLET (03)IN060SUP          138          0     0.00        813        1289         813
1289       599A,B,C
                  GNP  (03)IN060SUP          138          7     0.05       2112      112589        1047
112589       600A,B,C
                  GU   (01)IN010PAR          138        254     1.84      29511       75356        1195
19229       601A,B,C
                  DL/I PCB SUBTOTAL
                  ____ ___ ________          414        261     0.63      10812                    1018
                  PSB TOTAL
                  ___ _____                  694        395     0.56     559555                  551114
PROGSC2A I/O PCB  ISRT (  )                  118          0     0.00        381        1496         381
1496       603A,B,C
                  GU   (  )                  114        284     2.49    3304809    21784513     3164423
21664181       632A,B,C
                  (GU) (  )                    2          0     0.00         17          18          17
18       781A,B,C
                  ASRT (  )                    3          0     0.00        367         444         367
444       871A,B,C
                  GU   (  )         QC         2          5     2.50   19931897    20045206    19799530
```

```
19925277       872A,B,C
                  I/O PCB SUBTOTAL
                  ___ ___ ________           239       289     1.20     1743339                 1675270
         LOGVENDA REPL (03)IN040SLQ          118         0     0.00         268         804         268
804       604A,B,C
                  GNP  (03)IN040SLQ          118         5     0.04         899       16995         218
305       605A,B,C
                  REPL (02)VN030PAR          826         0     0.00         805        1578         805
1578       606A,B,C
                  GNP  (02)VN030PAR          826       873     1.05       19321       94521         456
1363       607A,B,C
                  REPL (01)VN020REO          118        58     0.49        8879       48076         832
1682       623A,B,C
                  GU   (01)VN020REO          118       195     1.65       31688      360775        1300
1746       625A,B,C
                  DL/I PCB SUBTOTAL
                  ____ ___ ________         2124      1131     0.53       10145                     636
                  PSB TOTAL
                  ___ _____                 2363      1420     0.60      185445                  170013
PROGSC2D I/O PCB  ISRT (  )                   14         0     0.00         377         621         377
621       608A,B,C
                  GU   (  )                   14        36     2.57    22360408    52048566    22221852
51901313       634A,B,C
                  I/O PCB SUBTOTAL
                  ___ ___ ________            28        36     1.28    11180393                11111115
         LOGVENDD REPL (03)IN040SLQ           14         0     0.00         263         328         263
328       609A,B,C
                  GNP  (03)IN040SLQ           14         1     0.07        1407       16889         223
307       610A,B,C
                  REPL (02)VN030PAR           98         0     0.00         820        1015         820
1015       611A,B,C
APOL1    I/O PCB  ASRT (..)........            3
                  (GU) (..)........            3
                  INQY (..)........            2
                  I/O PCB SUBTOTAL
                  ___ ___ ________             8
         OTMDEST1 ICAL (..)........ ..         4
                  DL/I PCB SUBTOTAL
                  ____ ___ ________            4
                  PSB TOTAL
                  ___ _____                   12
```

Monitoring I/O for application program DL/I Calls

The IMS Monitor report shows the total number of I/O occurrences and the total time the occurrences took for each application program executed during a monitored interval.

The Program-I/O report gives these two totals for all PSBs active during the monitored interval and includes the detailed breakdown of the I/O wait time as it was incurred by each PCB used by the program.

The report shows any contention experienced during application program processing. Each type of conflict and the number of times it occurred are recorded for each I/O PCB or database PCB. The report shows the total wait time, the highest wait experienced, and the average time. Subtotals are given for each PCB under a PSB, and for all PCBs under each PSB.

The DDN/FUNC column lists the data set DD name. The MODULE column uses a code to indicate the source of the contention. The types of conflicts and codes are shown as follows.

- **Message handling**

Code	Conflict
DBH	OSAM I/O for message queues
MFS	MFS format library directory
PMM	Message format buffer pool space or control block I/O
QMG	Message queue management

- **Scheduling**

Code
  Conflict

BLR
Load/read from ACBLIB

MSC
MPP region initialization

SMN
Virtual storage management

- **Database access**

Code	**Conflict**
DBH	OSAM I/O
DLE	DL/I functions
VBH	VSAM interface
(Physical segment code)	Program isolation

For external subsystem calls, the elapsed time to complete the processing is considered wait time. The DDN/FUNC column indicates the external subsystem call function, as shown as follows:

- **External subsystems**

Code	**Subsystem call function**
AB0	ABORT
CT0	Create thread
D50	Terminate identify or thread, signoff
D80	INIT
I30	Identify, command, echo, terminate
I30	Identify, terminate subsystem
I50	INIT
I60	Resolve-in-doubt
PR0	Subsystem-not-operational
P10	Commit prepare (Phase 1)
P20	Commit continue (Phase 2)
SO0	Signon
SI0	Identify

The following example show a sample of the report.

```
    IMS MONITOR   ****PROGRAM I/O****         TRACE START 1993 022   14:00:18    TRACE STOP  1993 022
14:02:20  PAGE 0088
                                    .........IWAIT TIME..........
PSBNAME  PCB NAME      IWAITS       TOTAL        MEAN     MAXIMUM   DDN/FUNC  MODULE
-------  --- ----      ------       -----        ----     -------   --------  ------

PROGHR1A I/O PCB          122     2341116       19189       70795   HOTELDBA   DBH
                           34    24177936      711115     3950160   **W F I
                           40    23652665      591316     2668917   **W F I
                            5       67613       13522       21214   SHMSG      QMG
                            4      110363       27590       60486   QBLKS      QMG

        PCB TOTAL
        ___ _____         131     2519092       19229
PSB TOTAL
___ _____                 305     6725063       20049

PROGDE1A TRMNALDA          20      624677       31233       68252   TRMNALDA   VBH
                            1      275811      275811     275811 PI TRMNALDA....

        PCB TOTAL
        ___ _____          21      900488       42880
         I/O PCB           16      488812       30550       79980   TRMNALDA   VBH
                            1       16118       16118       16118   SHMSG      QMG

        PCB TOTAL
        ___ _____          17      504930       29701
         TABLEDBA          16      290471       18154       33254   TABLEDA    DBH

        PCB TOTAL
        ___ _____          16      290471       18154

PSB TOTAL
___ _____                  54     1695889       31405

PROGHR2B HOTELDBB           8      698384       87298      184475   HOTELDBB   DBH
                            4     5820650     1455162     1455278 PI HOSINDXB....
                            4     4481024     1120256     1209075 PI HOTELDBB....
                            2      260817      130408      232750   HOSINDOB   VBH
                            7      106623       15231       16410   HOSINDXB   VBH
                            1       15366       15366       15366   HOTELDBD   DBH

        PCB TOTAL
        ___ _____          26    11382864      437802
PSB TOTAL
___ _____                  26    11382864      437802

PROGHR2A HOTELDBA          17      655801       38576      366108   HOSINDXA   VBH
                           73     1836721       25160       82141   HOTELDBA   DBH
                            2       54663       27331       41975   HOTELDBD   DBH
                            1        9887        9887        9887   HOTELDBC   DBH
                            2      851042      845635      845635   HOSINDOA   VBH

        PCB TOTAL
        ___ _____          95     3408114       35874
         I/O PCB           20      575847       28792       74227   HOTELDBA   DBH
                           21      370390       17637       43153   HOSINDXA   VBH
IMS MONITOR  ****PROGRAM I/O****  TRACE START 1993 022   14:00:18    TRACE STOP  1993 022  14:02:20
PAGE 0089
                                    .........IWAIT TIME..........
PSBNAME  PCB NAME      IWAITS       TOTAL        MEAN     MAXIMUM   DDN/FUNC  MODULE
-------  --- ----      ------       -----        ----     -------   --------  ------

PROGHR2A I/O PCB            5     4654544      930908     2020043  **W F I
                            8    32796604     4099575     9328891  **W F I

        PCB TOTAL
        ___ _____          41      946237       23078
PSB TOTAL
___ _____                 136     4354351       32017

PROGPS2A LOGIMA            89     2046670       22996       73593   IMMSTR3A   VBH
                          612    53886417       88049      185674   IMMSTR1A   VBH
                            3       44906       14968       20788   IMINDEXA   VBH

        PCB TOTAL
        ___ _____         704    55977993       79514
                          469    11742900       25038      170337   COMPOSDA   DBH
                          329     8198418       24919       91422   CPINDEXA   VBH
        PCB TOTAL
        --- -----         798    19941318       24989
         I/O PCB            3       47511       15837       20806   SHMSG      QMG

        PCB TOTAL
        --- -----           3       47511       15837
PSB TOTAL
--- -----                1505    75966822       50476

PROGSC6C I/O PCB           52     2698602       51896      473763   INVENTRC   VBH
                            4       70921       17730       34241   SHMSG      QMG
                            3       50699       16899       24724   QBLKS      QMG

        PCB TOTAL
        --- -----          59     2820222       47800
                           55     2666884       48488      210752   INVENTRC   VBH
                           50      797587       15951       41706   ININDEXC   VBH
                            1      119253      119253      119253 PI INVENTRC...1
                            1        8634        8634        8634   INVENTRB   VBH
                            2       83947       41973       53936   INVENTRA   VBH

        PCB TOTAL
        --- -----         109     3676305       33727
PSB TOTAL
--- -----                 168     6496527       38669

PROGHR2D I/O PCB           21     2285296      108823      199223   HOTELDBD   DBH
                           28      762370       27227      111860   HOSINDXD   VBH
                            1       11685       11685       11685   SHMSG      QMG

        PCB TOTAL
        --- -----          50     3059351       61187
         HOTELDBD          96     6279107       65407      139032   HOTELDBD   DBH

         MONITOR   ****PROGRAM I/O****   TRACE START 1993 022   14:00:18    TRACE STOP  1993 022
14:02:20  PAGE 0090
                                  .........IWAIT TIME..........
PSBNAME  PCB NAME      IWAITS       TOTAL        MEAN     MAXIMUM   DDN/FUNC  MODULE
-------  --- ----      ------       -----        ----     -------   --------  ------

PROGHR2D HOTELDBD          31     2130585       68728      769130   HOSINDXD   VBH
                            3      115999       38666       56394   HOTELDBA   DBH
                            2       69833       34916       43470   HOTELDBC   DBH
                            2       41430       20715       28020   HOSINDOD   VBH
                            4     5515374     1378843     1458884 PI HOSINDXD....
                            4     3997017      999254     1026228 PI HOTELDBD....

PCB TOTAL
--- -----                 142    18149345      127812
PSB TOTAL
--- -----                 192    21208696      110461
```

The I/O waits for the calls to the I/O PCB, are grouped as the first entries for a PSB. For DL/I calls, the data set for which the I/O took place is indicated under the DDN/FUNC heading, and the module code indicates what type of conflict caused the wait. For external subsystem calls, the function is indicated under the DDN/FUNC heading and the module code indicates the source of the call entry.

Names other than LGMSG and SHMSG can appear in the DDN/FUNC column for I/O PCBs. An example is a checkpoint call issued by an application program (using an I/O PCB) which causes a database buffer to be written.

If the program is designated as wait-for-input and has to wait for the input of the next message, the wait entry is marked **WFI under the DDN/FUNC heading and no entry appears in the MODULE column. The time spent waiting for the next input message is shown under wait time. **WFI entries are shown for information only and their values are not used to compute statistics.

Contention for the same physical segment in a database causes a wait on behalf of program isolation. This is shown in the DDN/FUNC column, on the PCB line, by the entry PIdmb, where dmb is the DMB of the physical data set. The MODULE column identifies the segment type using the physical segment code assigned by DBD generation.

When an application is accessing a database using VSAM as the access method, DL/I calls do not generally result in an I/O wait. A MODULE column entry of VBH indicates that interface to VSAM occurred and there was an I/O wait.

A seemingly unrelated entry can occur under the DDN/FUNC column for a database PCB. An example is a retrieval call to a database (DB-A) that causes a buffer to be purged in order to make room for that retrieved data. If the buffer contents included data belonging to another database (DB-B), the I/O entry in the report shows the DD name for DB-B as being in conflict for PCB access to DB-A.

Monitoring MFS activity

You can obtain a summary of all activity that occurs for management of message format buffer pool use from the Message Format Buffer Pool report.

The report is illustrated in the following example. The data shows the counts at the start and end of the trace interval and their difference.

When message formatting occurs, the appropriate message blocks must reside in the message format buffer pool, a DIF/MID pair for input or a DOF/MOD pair for output. If the blocks are not already in the buffer, I/O to the active IMS.FORMATA/B library must occur. Block retrieval can involve a prior directory lookup, or be direct, using an index kept in the pool.

Many of the counts reveal details of internal event management. When there is no directory entry for a block this implies extra directory lookup I/O.

```
***I M S  M O N I T O R***  BUFFER POOL STATISTICS  TRACE START 1993 130   5:55:15  TRACE STOP  1993
130   5:59:49  PAGE 0007

                   M E S S A G E   F O R M A T   B U F F E R   P O O L

                                                                         5:55:15            5:59:49
                                                                      START TRACE         END
TRACE        DIFFERENCE

    NUMBER OF P/F REQUESTS                                                     0
0                  0
    NUMBER OF I/F REQUESTS                                                    18
20                  2
    NUMBER OF I/F I/O'S                                                        2
2                  0
    NUMBER OF TIMES POOL COMPRESS WOULD BE SUCCESSFUL                          0
0                  0
    NUMBER OF DIRECTORY I/O OPERATIONS                                         2
2                  0
    NUMBER OF TIMES BLOCK WASHED FOR FRE                                       0
0                  0
    NUMBER OF TIMES P/F REQUEST IGNORED                                        0
0                  0
    NUMBER OF F/B REQUESTS                                                    18
20                  2
    NUMBER OF TIMES F/B REQUEST IGNORED                                        0
0                  0
    NUMBER OF TIMES I/F ON F/B QUEUE                                          16
18                  2
    NUMBER OF TIMES I/F ON I/F QUEUE                                           0
0                  0
    NUMBER OF TIMES F/B ON I/F QUEUE                                          18
20                  2
    NUMBER OF TIMES P/F ON I/F QUEUE                                           0
0                  0
    NUMBER OF TIMES P/F ON F/B QUEUE                                           0
0                  0
    NUMBER OF TIMES THERE WAS NO DIR ENTR FOR A BLOCK                          0
0                  0
    NUMBER OF TIMES I/O ERRORS POINT OR READ MACRO                             0
0                  0
    NUMBER OF IMMEDIATE I/O REQUESTS WAITED DUE TO MAXIMUM I/O                 0
0                  0
    NUMBER OF REQUESTS SATISFIED BY INDEX/DYNAMIC DIRECTORY                    0
0                  0

    QUOTIENT :  IMMEDIATE FETCH I/O'S + DIRECTORY I/O'S OPERATIONS =    0.00
                ----------------------------------------------------------------------------------
                                            TOTAL NUMBER OF TRANSACTIONS
```

Monitoring message queue handling

A key resource that directly affects the efficiency of transaction processing is the message queue pool and the management of the I/O to the message queues. You can examine the activity by looking at the Message Queue Pool report.

The following example illustrates the message-queue-pool report. Counts of activities are given at start and end of the trace interval and as the differences between start and end numbers.

```
***I M S  M O N I T O R***  BUFFER POOL STATISTICS  TRACE START 1993 130   5:55:15  TRACE STOP  1993 
130   5:59:49  PAGE 0002

                   M E S S A G E   Q U E U E   P O O L

                                                                            5:55:15            5:59:49
                                                                         START TRACE        END
TRACE        DIFFERENCE

    NUMBER OF LOCATE CALLS FROM QMGR                                         54204
68436             14232
    NUMBER OF RECORD RELEASE CALLS FROM QMGR                                 16431
20738              4307
    NUMBER OF LOCATE AND ALTER CALLS FROM QMGR                              131593
164744             33151
    NUMBER OF REQUESTS TO PURGE THE Q POOL                                       2
2                 0
    NUMBER OF ADDRESS TO DRRN TRANSLATION REQUESTS                           21351
27076              5725
    NUMBER OF REQUESTS TO WAIT FROM QMGR                                         0
0                 0
    NUMBER OF READ REQUESTS                                                    962
962                 0
    NUMBER OF WRITE REQUESTS(TOTAL)                                            499
499                 0
    NUMBER OF WRITES DONE BY PURGE                                             499
499                 0
    NUMBER OF WAITS FOR PURGE COMPLETION                                         1
1                 0
    NUMBER OF WAITS BECAUSE NO BUFFER AVAILABLE                                  0
0                 0
    NUMBER OF WAITS FOR OTHER DECB TO READ THIS BUFFER                         823
823                 0
    NUMBER OF WAITS FOR OTHER DECB TO WRITE THIS BUFFER                          0
0                 0
    NUMBER OF WAITS FOR CONFLICTING END DEQ BUFFER REQ                           0
0                 0
    NUMBER OF PSBS UNCHAINED FROM BUFFERS                                        0
0                 0
    NUMBER OF CALLS TO QMGR.(TOTAL)                                          48164
62213             14049
    NUMBER OF CALLS TO REPOSITION A LOST BUFFER                                  0
0                 0
    NUMBER OF CALLS TO ENQ A MESSAGE                                         10583
13441              2858
    NUMBER OF CALLS TO DEQ ONE OR MORE MESSAGE                                6321
7767              1446
    NUMBER OF CALLS TO CANCEL INPUT OR OUTPUT                                  119
121                 2

    QUOTIENT :  TOTAL NUMBER OF OSAM READS + OSAM WRITES + ALL IWAITS =     0.00
                -----------------------------------------------------
                              TOTAL NUMBER OF TRANSACTIONS
```

Detecting checkpoint effects

When a checkpoint command specifies SNAPQ, the current status of all message queues is written to the system log. This prevents any message handling on behalf of queue management.

The General Iwait Time Events records the wait time incurred by the SNAPQ. The following example shows the activity on the summary line QMGR SNAPQ CHECK. The number of occurrences is given with the total, average, and maximum wait times.

```
      IMS MONITOR  ** GENERAL REPORTS **    TRACE START 1993 130   5:55:15    TRACE STOP  1993 130 
5:59:49  PAGE 0009
                GENERAL IWAIT TIME EVENTS
```

```
    EVENT                                 ..........IWAIT TIME...........    DISTRIBUTION
    IWAITS          OCCURRENCES        TOTAL         MEAN      MAXIMUM       NUMBER
    ______          ___________        _____         ____      _______       ______
QMGR SNAPQ CHECK              0            0            0            0            0
                REGION AND JOBNAME REPORT
REG. NO.   JOB NAME
________   ________
     1     MPR1A100
     2     MPR1A209
     3     MPR1A210
     4     MPR1A211
     5     MPR1A103
     6     MPR1A101
     7     MPR1A115
     8     MPR1A116
     9     MPR1A216
    10     MPR1A200
    11     MPR1A217
    12     MPR1A119
    13     MPR1A218
    14     MPR1A219
    15     MPR1A104
    16     MPR1A220
    17     MPR1A203
    18     MPR1A123
    19     MPR1A222
    20     MPR1A105
    21     MPR1A124
    22     MPR1A223
    23     MPR1A107
    24     MPR1A224
    25     MPR1A106
    26     MPR1A206
    27     MPR1A205
    28     MPR1A108
    29     MPR1A109
    30     MPR1A208
    31     MPR1A111
    32     MPR1A112
    33     MPR1A113
    34     MPR1A204
    35     MPR1A114
    36     MPR1A102
    48     MPR1A121
    49     MPR1A122
    50     MPR1A221
```

Transaction Queuing report

In addition to monitoring the efficiency of message handling, you can monitor the service provided for each application, by looking at the size of the transaction queues at each scheduling of their processing programs.

The Transaction-Queuing report shown in the following example records, for each transaction, the minimum, average, and maximum counts at scheduling time. The total number of dequeued transactions (or transactions that have been fully processed) during the monitored interval is given for each transaction code. The average number of transactions processed for each scheduling is given in the DEQUEUED MEAN column.

```
     IMS MONITOR   ****TRANSACTION QUEUING****  TRACE START 1993 130   5:55:15  TRACE STOP  1993 130
5:59:49  PAGE 0181
                                                         (B)                     (A)
                   NUMBER      NUMBER      ..ON QUEUE  WHEN  SCHEDULED.....   DEQUED      DISTRIBUTION
TRANSACTION        DEQUED      SCHEDS.     MINIMUM     MEAN         MAXIMUM    MEAN        NUMBER
___________        ______      ______      _______     ____         _______    ____        ______

 SC6X                13           1           0        0.00            0      13.00         883A,B
 IT8W                17           3           0        0.00            0       5.66         887A,B
 TS1Z                16           1           0        0.00            0      16.00         891A,B
 PS3X                23           1           0        0.00            0      23.00         895A,B
 PS3Y                17           7           0        0.00            0       2.42         899A,B
 IT1V                11           6           0        0.00            0       1.83         903A,B
 SC2Z               143           2           0        0.00            0      71.50         907A,B
 IT8U                12           7           0        0.00            0       1.71         911A,B
 PS2W                10           1           0        0.00            0      10.00         915A,B
 TS1U                12           4           0        0.00            0       3.00         919A,B
```

```
  PS3W                14          1           0       0.00           0       14.00          923A,B
  IT8Y                16          5           0       0.00           0        3.20          927A,B
  IT1W                 9          1           0       0.00           0        9.00          929A,B
  SC2W                 9          1           0       0.00           0        9.00          933A,B
  IT2V                13          5           0       0.00           0        2.60          937A,B
  IT2W                17          6           0       0.00           0        2.83          941A,B
  TS1V                 9          1           0       0.00           0        9.00          945A,B
  SC2V                12          5           0       0.00           0        2.40          947A,B
  TS1W                11          1           0       0.00           0       11.00          949A,B
  PS3V                13          3           0       0.00           0        4.33          953A,B
  IT1U                 9          6           0       0.00           0        1.50          957A,B
  SC4U                11          5           0       0.00           0        2.20          962A,B
  SC6W                10          1           0       0.00           0       10.00          966A,B
  PS2V                 8          6           0       0.00           0        1.33          970A,B
  IT8X                12          1           0       0.00           0       12.00          974A,B
  SC4W                10          1           0       0.00           0       10.00          978A,B
  SC6U                14          6           0       0.00           0        2.33          982A,B
  IT2Y                 9          3           0       0.00           0        3.00          986A,B
  SC2X                15          1           0       0.00           0       15.00          990A,B
  PS2Y                17          2           0       0.00           0        8.50          994A,B
  SC4Y               152          4           0       0.50           1       38.00          998A,B
  SC2Y               106          2           0       0.00           0       53.00         1000A,B
  IT2X                20          1           0       0.00           0       20.00         1004A,B
  SC2U                15          3           0       0.00           0        5.00         1008A,B
  SC4Z               123          5           0       0.60           1       24.60         1010A,B
  TS1X                15          1           0       0.00           0       15.00         1015A,B
  SC4X                19          1           0       0.00           0       19.00         1019A,B
  PS2X                13          1           0       0.00           0       13.00         1024A,B
  PS2Z                20          5           0       0.00           0        4.00         1028A,B
  SC6Z               130          1           0       0.00           0      130.00         1031A,B
  SC6V                10          4           0       0.00           0        2.50         1035A,B
  SC6Y               143          1           0       0.00           0      143.00         1037A,B
  IT1X                10          1           0       0.00           0       10.00         1040A,B
  PS3U                19          6           0       0.00           0        3.16         1131A,B
  IT2U                13          4           0       0.00           0        3.25         1146A,B
```

Monitoring database buffers

One of the key resources in an online system is the database buffer pool. The efficiency of DL/I call service depends on the presence of the required database logical record in the buffer, so that segment retrieval does not require additional I/O.

This is especially true for HOLD calls with intervening database calls prior to a replace call.

Use the **QUERY POOL** command to view usage statistics for Fast Path buffer pools.

You can assess the general efficiency of the full-function pool management using the Database Buffer Pool report shown in the following example. The event counts on this report are not specific to a particular database or program but represent the pressure for use of the database pool.

Related reading: For more information about the Database Buffer Pool reports, see "Database-Buffer-Pool report" on page 583.

If any of your databases use VSAM as the access method, the IMS Monitor produces a series of reports headed VSAM BUFFER POOL, one for each subpool. The following example shows one of these reports.

```
                    D A T A   B A S E   B U F F E R   P O O L

                                                                                    FIX PREFIX/
BUFFERS              Y/Y
                                                                                    SUBPOOL
ID                     004K
                                                                                    SUBPOOL BUFFER
SIZE              4096
                                                                                    TOTAL BUFFERS IN
SUBPOOL         1000


                                                                             17:08:15          17:10:16

    NUMBER OF LOCATE-TYPE CALLS                                               1117674
1676213             558539
    NUMBER OF REQUESTS TO CREATE NEW BLOCKS                                         0
0                  0
    NUMBER OF BUFFER ALTER CALLS                                               215874
322936             107062
```

```
    NUMBER OF PURGE CALLS                                                    25077
37454             12377
    NUMBER OF LOCATE-TYPE CALLS, DATA ALREADY IN OSAM POOL                  870306
1301187            430881
    NUMBER OF BUFFERS SEARCHED BY ALL LOCATE-TYPE CALLS                    1258247
1886843            628596
    NUMBER OF READ I/O REQUESTS                                             238165
360260            122095
    NUMBER OF SINGLE BLOCK WRITES BY BUFFER STEAL ROUTINE                        0
0                 0
    NUMBER OF BLOCKS WRITTEN BY PURGE                                        95057
142413             47356
    NUMBER OF LOCATE CALLS WAITED DUE TO BUSY ID                               780
1297               517
    NUMBER OF LOCATE CALLS WAITED DUE TO BUFFER BUSY WRT                         0
0                 0
    NUMBER OF LOCATE CALLS WAITED DUE TO BUFFER BUSY READ                        0
0                 0
    NUMBER OF BUFFER STEAL/PURGE WAITED FOR OWNERSHIP RLSE                     178
261                83
    NUMBER OF BUFFER STEAL REQUESTS WAITED FOR BUFFERS                           0
0                 0
    TOTAL NUMBER OF I/O ERRORS FOR THIS SUBPOOL                                  0
0                 0
    NUMBER OF BUFFERS LOCKED DUE TO WRITE ERRORS                                 0
0                 0

    QUOTIENT :  TOTAL NUMBER OF OSAM READS + OSAM WRITES =                 6.98
                ------------------------------------------------------
                              TOTAL NUMBER OF TRANSACTIONS
```

The following example shows a sample of the VSAM-Buffer-Pool report.

```
***I M S  M O N I T O R***  BUFFER POOL STATISTICS

                   V S A M   B U F F E R   P O O L
                                                                                         FIX INDEX/BLOCK/
DATA          N/Y/N
                                                                                         SHARED RESOURCE
POOL ID        VPL1
                                                                                         SHARED RESOURCE
POOL TYPE         D
                                                                                         SUBPOOL
ID                         2
                                                                                         SUBPOOL BUFFER
SIZE            4096
                                                                                         NUMBER
HIPERSPACE BUFFERS        50
                                                                                         TOTAL BUFFERS
IN SUBPOOL       1000

                                                                           17:08:15           17:10:16

    NUMBER OF RETRIEVE BY RBA CALLS RECEIVED BY BUF HNDLR                       152
330               178
    NUMBER OF RETRIEVE BY KEY CALLS                                         117780
178424             60644
    NUMBER OF LOGICAL RECORDS INSERTED INTO ESDS                               132
310               178
    NUMBER OF LOGICAL RECORDS INSERTED INTO KSDS                              6460
9853              3393
    NUMBER OF LOGICAL RECORDS ALTERED IN THIS SUBPOOL                            0
0                 0
    NUMBER OF TIMES BACKGROUND WRITE FUNCTION INVOKED                            0
0                 0
    NUMBER OF SYNCHRONIZATION CALLS RECEIVED                                 18566
27923              9357
    NUMBER OF WRITE ERROR BUFFERS CURRENTLY IN THE SUBPOOL                       0
0                 0
    LARGEST NUMBER OF WRITE ERRORS IN THE SUBPOOL                                0
0                 0
    NUMBER OF VSAM GET CALLS ISSUED                                         124648
189220             64572
    NUMBER OF VSAM SCHBFR CALLS ISSUED                                           0
0                 0
    NUMBER OF TIMES CTRL INTERVAL REQUESTED ALREADY IN POOL                  33662
51088             17426
    NUMBER OF CRTL INTERVALS READ FROM EXTERNAL STORAGE                      91169
138505             47336
```

```
    NUMBER OF VSAM WRITES INITIATED BY IMS                              6022
9251               3229
    NUMBER OF VSAM WRITES TO MAKE SPACE IN THE POOL                          0
0                  0
    NUMBER OF VSAM READS FROM HIPERSPACE BUFFERS                             0
0                  0
    NUMBER OF VSAM WRITES FROM HIPERSPACE BUFFERS                           50
50                  0
    NUMBER OF FAILED VSAM READS FROM HIPERSPACE BUFFERS                      0
0                  0
    NUMBER OF FAILED VSAM WRITES FROM HIPERSPACE BUFFERS                     0
0                  0

    QUOTIENT :  TOTAL NUMBER OF VSAM READS + VSAM WRITES =                 2.08
                ______________________________________________________
                              TOTAL NUMBER OF TRANSACTIONS
```

Monitoring line activity

You can obtain a summary of all occurrences of activity for each node that handles message traffic during the monitored interval. The elapsed times and not-wait times are given in categories of total, mean, and maximum times for each communication line in the Communication-Summary report.

The following example illustrates this report.

You must match which physical devices are using the line to the Stage 1 output from system definition. The line numbers are assigned sequentially, according to their physical occurrence in the Stage 1 input deck.

If your online system specifies the prefetch option for MFS blocks in the control region JCL, the last line of the report contains the statistics for all prefetch events.

```
     IMS MONITOR   ****COMMUNICATION SUMMARY****  TRACE START 1993 130   5:55:15  TRACE STOP  1993 130
5:59:49  PAGE 0089
                                                             (A)                                  (B)
     NODE OR                                         ........ELAPSED TIME.........       NOT IWAIT TIME(ELAPSED-
IWAIT)    DISTRIBUTION
   LINE NUMBER       OCCURRENCES       TOTAL        MEAN      MAXIMUM       TOTAL         MEAN
MAXIMUM      NUMBER

   ---- ------       -----------       -----        ----      -------       -----         ----
-------      ------
     PMTO1A                    3        2396         798         1547        2396          798
1547        1467A,B
        19                   182       92155         506         1106       92155          506
1106        1493A,B
         2                    59        2280          38           41        2280           38
41        1515A,B
      TOTAL
      _____                  244       96831         396                    96831          396
```

You can also investigate the amount of data transmitted across nodes with the Line-Functions report. The following example illustrates this report. The report distinguishes between input data and output data. The number of blocks of data and the average and maximum size of the blocks are recorded for data received by IMS and for transmitted data.

This report also includes a measure of how inactive the lines are. An inactive interval is assumed to be the difference between the time that marks the end of the last input block received and the starting time for output transmission. These occurrences of inactivity are termed turnaround intervals, and the report cumulates the number of occurrences as well as the average and maximum times associated with these intervals.

If the line is being used by an MFS-supported terminal, a count of the number of requests for next page for a multipage message is recorded.

If link traffic for coupled multiple systems is recorded, a set of three reports follows the Line Functions report. These are described in "IMS Monitor reports for MSC" on page 627.

```
     IMS MONITOR   ****LINE FUNCTIONS***              TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0091
                                 (A).....................(B)..............                  ......................
                                 MEAN     MAX.           MEAN    MAX.     .         TURN         .                    .
```

```
  NODE OR    DEVICE  RECEIVE RECEIVE RECEIVE   TRANS.  TRANS.  TRANS.  DIST.      AROUND       MEAN         MAX.
DIST.   PAGING
LINE NUMBER  TYPE    BLOCKS BLKSIZE BLKSIZE   BLOCKS BLKSIZE BLKSIZE NUMBER    INTERVALS    INTERVAL    INTERVAL
NUMB.  REQUESTS
---- ------  ----    ------ ------- -------   ------ ------- ------- ------    ---------    --------    --------
-----  --------
  PMT01A    3270V         1      29      29        2     170     171   1468A,B         3         798        1547
1466          0
       19   XXXX                                                                     182         506        1106
1492          0
        2   LOC SYS                                                                   59          38          41
1514          0
ALL LINES
--- -----                 1      29                2     170                         244
396                            0
```

Monitoring message handling efficiency

The IMS Monitor produces both summary and detailed information on asynchronous processing that takes place in the IMS control region. Asynchronous processing occurs when data transmitted from VTAM arrives. Application program responses also result in asynchronous processing.

The space in four major buffer pools and access to format, SPA, and message queue data sets are managed for the total communications traffic. Wait times are recorded when contention for pool space or I/O interrupts the processing of any of the communication tasks triggered by line activity. This information is contained in the Communication Wait report, shown in the following example.

This report is similar to the Communication-Summary report because the line number identifies the series of communication processing tasks.

```
     IMS MONITOR   ****COMMUNICATION IWAIT*****        TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0090
  NODE OR                      ..........IWAIT TIME..........                                       DIST.
LINE NUMBER  OCCURRENCES       TOTAL        MEAN     MAXIMUM     FUNCTION           BLKSIZE  MODULE       NO.
---- ------  -----------       -----        ----     -------     --------           -------  ------      ----
ALL LINES...
PREFETCH I/O
-------- ---        NONE
```

IMS internal resource usage

There are several summary reports that you can use to examine the level of internal contention for resources.

The following list gives a brief explanation of these reports.

Pool space failure

The Pool Space Failure Summary report gives the number of times in each region a given amount of storage was unavailable. It shows the number of bytes, the identification of the pool, and the number of occurrences when storage was unavailable. You can use this summary to determine if you need to increase the buffer pool allocation, either by a system definition change or by overriding the number of buffers in the EXEC statements in the JCL.

The format of the report is shown in the following example.

```
POOL  SPACE  FAILURE  SUMMARY

           POOL  ID     BYTES REQ.    OCCURRENCES
           DLMP            8888             1
           DLDP            7777             1
      TOTAL                                 2
```

Programs experiencing deadlock

Each time a pair of programs reaches a deadlock over ownership of a segment in a given database data set, the Deadlock-Event-Summary report records the occurrence. Each line in the report shows the two PSBs involved and indicates which is given processing right-of-way (REQ-ING PSB) and which has to reprocess after dynamic backout occurs (LOSING PSB). The report is illustrated in the following example.

```
DEADLOCK EVENT SUMMARY

     REQ-ING PSB    LOSING PSB    DMBNAME    OCCURRENCES
     PSBNAME1       TPPSBRE3      DBASEBAL        1
  TOTAL                                           1
```

IMS latch conflict

The basic serialization of the task processing in IMS is controlled by ownership of an IMS latch. When different programs are executing, they compete for the ownership. If they wait for the resource, the one possessing the latch has to post the other ITASK waiting for it. Use the Latch Conflict Statistics report to judge the level of contention for a resource.

The latch names, which are abbreviations for the different types of resources being serialized, are as follows:

Abbreviation	**Latch Name**
DISP	SYS/DISPATCHER
DCSL	DC/CHECKPOINT DC SYSTEM
LUML	DC/LU 6.2 LUM
CONV	DC/CONVERSATION CHECKPT
TERM	DC/TERMINAL
LUBT	DC/LU62 LUB-TIB CHAIN
SCHD	TM/SCHEDULING
TCTB	TM/TCT BLOCK
APSB	TM/ALLOCATE PSB (BLK MVR)
PDRB	TM/PDIR BLOCK (BLK MVR)
PSBP	TM/PSB POOL (BLK MVR)
DMBP	TM/DMB POOL (BLK MVR)
PSBB	TM/PSB BLOCK (BLK MVR)
DMBB	TM/DMB BLOCK (BLK MVR)
PDRP	TM/PDIR POOL (BLK MVR)
DBAU	TM/DBRC AUTH (BLK MVR)
DDRB	TM/DDIR BLOCK (BLK MVR)
DDRP	TM/DDIR POOL (BLK MVR)

DBBP	DB/OSAM BUFFER POOL
DBLR	DB/DFSDBLR0 MODULE
SUBQ	TM/TM SUBQUEUES
DBSL	DB/DB CHECKPOINT
USER	DC/USER
CCTL	SYS/DBCTL RESOURCE
VTCB	SYS/CBTS VTCB POOL
VLQB	SYS/CBTS LQB POOL
CBTS	SYS/CBTS POOLS (ALL)
BLKM	TM/SMB QUEUE HASH TABLE
QMGR	SYS/QUEUE MANAGER
QBSL	SYS/QUEUE BUFFER
SMGT	SYS/STORAGE MANAGEMENT
DBLK	SYS/DEPENDENT REGION
XCNQ	DB/EXCLUSIVE ENQ/DEQ
ACTL	SYS/STATISTICS
LOGL	SYS/LOGGER

When a system checkpoint is taken during the time the monitor is active, latch conflict statistics are reset to zero, thus corrupting the values presented in this report. If this situation exists, the following message will be inserted at the top of the report:

```
**** A CHECKPOINT OCCURRED DURING MONITOR RUN ****
****  LATCH CONFLICT STATISTICS ARE INVALID   ****
****      SEE UTILITIES REFERENCE MANUAL      ****
```

However, if the master terminal operator issues the **/CHECKPOINT** command with the STATISTICS keyword parameter, latch conflict statistics are reset to zero, but the IMS monitor is not notified. Therefore, DFSUTR20 cannot detect that the statistics have been corrupted and does not issue this message.

Recommendation: Do not issue statistics checkpoints while the monitor is running.

The different types of latches and the counters that exhibit the level of contention are given in the Latch Conflict Statistics report. The following example shows a sample of the latch-conflict-statistics report. The entries are organized according to the latch names.

```
         IMS MONITOR   ** GENERAL REPORTS **       TRACE START 1993 209...

                 LATCH CONFLICT STATISTICS

LATCH        COUNT                AT          AT
NAMES        FIELD             START         END        DIFF.

LOGL      CONTENTIONS              0           0           0

SMGT      CONTENTIONS              0           0           0

XCNQ      CONTENTIONS              0           0           0

ACTL      CONTENTIONS              0           0           0

CBTS      CONTENTIONS              0           0           0

DBLK      CONTENTIONS              0           0           0
```

Using frequency distributions from IMS Monitor output

The reports derived from the IMS Monitor data records contain many summary lines where the mean time is given. If you are interested in the distribution of those timed events, rather than just average and maximum times, you can request the Report Print utility to individually record the events in a frequency distribution across a range of intervals.

Some distributions are not time dependent, such as those for transaction queue loads or transmitted block sizes.

The following tables show the major IMS Monitor reports and the type of frequency distributions generated for each report. Each type results in several distributions, depending on how many entries are in each section of the report. For each type of frequency distribution, the data is cumulated in suitable intervals or ranges. The set of ranges used for each type is given an identifier, shown in the ID column.

- The following table shows the report distributions sorted by region summary.

Table 56. Distribution reports by region summary

Report name	**ID**	**Description**
Schedule end to 1st DL/I call	D1	Elapsed time
Elapsed execution time	D2	Not wait time
DL/I calls	D3	N/A
	D4	N/A
External Subsystem calls	D5	Elapsed time
Waits per DL/I call	D6	Not wait time
Idle for intent	D43	Elapsed time
Checkpoint	D7	N/A
	D8	N/A
	D20	Elapsed time
	D21	Not wait time

- The following table shows the report distributions sorted by program region.

Table 57. Report distributions by program region

Report name	ID	Description
Elapsed execution time	D30	N/A
Schedule end to 1st DL/I call	D31	N/A

- The following table shows the report distributions sorted by program summary.

Table 58. Report distributions by program summary

Report name	ID	Description
Processor time per schedule	D15	N/A
Transactions dequeued per schedule	D14	N/A
Elapsed time per schedule	D9	N/A
Schedule end to 1st DL/I call	D10	N/A

- The following table shows the report distributions sorted by communication summary.

Table 59. Report distributions by communication summary

Report name	ID	Description
Line elapsed time	D18	N/A
Line not wait time	D19	N/A

- The following table shows the report distributions sorted by line functions.

Table 60. Report distributions by line functions

Report name	ID	Description
Received block length	D36	N/A
Transmitted block length	D37	N/A
Inactive intervals	D38	N/A

- The following table shows the report distributions sorted by MSC queuing summary.

Table 61. Report distributions by MSC queuing summary

Report name	ID	Description
Time in queue	D39	N/A

- The following table shows the report distributions sorted by transaction queuing.

Table 62. Report distributions by transaction queuing

Report name	ID	Description
Transactions on queue at schedule	D17	N/A
Transactions dequeued per schedule	D16	N/A
Prefetch format blocks	D28	Elapsed time
	D29	Not wait time

- The following table shows the report distributions sorted by call summary.

Table 63. Report distributions by call summary

Report name	ID	Description
PSB waits per DL/I call	D13	N/A
PSB waits per external subsystem call	D44	N/A
PSB elapsed time per call	D11	N/A
PSB not wait time per call	D12	N/A
PSB external subsystem calls	D45	Elapsed time

- The following table shows some distributions derived from buffer pool statistics for wait times.

Table 64. Wait time distributions

Function	ID	Module key
Storage	D22	SMN
OSAM I/O	D23	DBH
VSAM I/O	D24	VBH
Scheduler internal	D25	MSC
Queue manager I/O	D26	QMG
Block loader I/O	D27	BLR
MFS block I/O	D32	MFS
MFS directory I/O	D33	MFS
HSAM I/O	D34	DIE
Format Buffer Pool Space	D35	PMM
PI enqueue	D40	None
QMGR SNAPQ Check	D42	None

Related reference

“Default values of distribution definitions” on page 625
Using an identifier provided in the frequency distribution tables and the Wait Time Distributions table, you can determine the default end points for the distribution by locating it in the following list.

How to get a frequency distribution output

To request the IMS Monitor Report Print utility to gather distribution data, you must include a DIS input control statement. This causes all report items with an entry under a column headed MEAN to have a corresponding frequency distribution as part of the Distribution Appendix.

Each report line includes an identifying reference number under the column headed Distribution Number so that you can locate the distribution data in the appendix, flagged by that same number.

How frequency distribution ranges are defined

A set of ten intervals is defined for each summary line and the occurrences falling in each interval are cumulated. The interval ranges are preset with default end points. The default end points are chosen so that they are suitable to the event.

For example, the end points, for DL/I call elapsed time are: 0, 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000, 256000, INF (all times are in milliseconds). The lower limit of the first interval always defaults to zero, and the upper limit of the tenth interval is infinity (INF).

Although several types of distribution can use the same set of end points, each type is assigned a distribution identifier. You can use this to redefine the end points. To override the default end points, include an input control statement to the Report Print utility. The statement specifies the type of distribution identifier and gives the desired end point values. For example, the DL/I call elapsed time end points could be respecified by:

```
D5   0,500,1000,1500,2000,4000,,,100000,500000
```

The values of the unspecified end points remain at their default values of 32000 and 64000 as does the last (INF).

The following example, which is a sample page taken from a Distribution Appendix, shows how individual distributions are numbered, and how ranges vary with the type of distribution. The lines are arranged in pairs, with the second one recording the cumulated counts.

```
   IMS MONITOR  ****DISTRIBUTION APPENDIX****   TRACE START 1993 130   5:55:15    TRACE STOP  1993 130
5:59:49  PAGE 0200

#
1..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          1
#
2..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          1
#
3..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          1
#
4..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          1
#
5..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          6
#
6..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          8
#
7..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          1
#
8..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          9
#
9..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                   0          0          0          0          0          0          0
0          0          1
#
10..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                   1          0          0          0          0          0          0
0          0          7
#
11..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                   0          0          0          0          0          0          0
0          0          1
#
12..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
```

```
                  0         0         0         0         0         0         0
0         0       8
#
13..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                  0         0         0         0         0         0         0
0         0       1
#
14..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                  0         0         0         0         0         0         0
0         0       1
#
15..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                  1         0         0         0         0         0         0
0         0       8
#
16..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                  0         0         0         0         0         0         0
0         0       1
```

Default values of distribution definitions

Using an identifier provided in the frequency distribution tables and the Wait Time Distributions table, you can determine the default end points for the distribution by locating it in the following list.

D1, D2, D5, D6, D9, D10, D11, D12, D15 D18, D19, D20, D21, D22, D25, D27 D28, D29, D30, D31, D43, and D45
: 0, 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000, 256000, INF

D3
: 0, 50000, 100000, 150000, 200000, 250000, 300000, 350000, 400000, 450000, INF

D4
: 0, 200000, 400000, 600000, 800000, 1000000, 1200000, 1400000, 1600000, 1800000, INF

D7, D13, and D44
: 0, 0, 1, 2, 3, 4, 5, 6, 7, 8, INF

D8
: 0, 100000, 200000, 300000, 400000, 500000, 600000, 700000, 800000, 900000, INF

D14, D16, D17
: 0, 1, 2, 3, 4, 5, 10, 15, 30, 90, INF

D23, D24, D26, D32, D40, D42
: 0, 2000, 8000, 24000, 50000, 100000, 150000, 200000, 250000, 300000, INF

D33, D34, D35
: 0, 2000, 4000, 8000, 16000, 32000, 64000, 96000, 128000, 160000, INF

D36, D37
: 0, 10, 20, 40, 80, 100, 200, 400, 800, 1000, INF

D38
: 0, 1000, 10000, 100000, 200000, 500000, 800000, 1000000, 1500000, 2000000, INF

D39
: 0, 1000, 5000, 10000, 50000, 100000, 500000, 1000000, 5000000, 10000000, INF

Related concepts

“Using frequency distributions from IMS Monitor output” on page 621
The reports derived from the IMS Monitor data records contain many summary lines where the mean time is given. If you are interested in the distribution of those timed events, rather than just average and

maximum times, you can request the Report Print utility to individually record the events in a frequency distribution across a range of intervals.

Interpreting the Distribution Appendix

You can use the detailed output in the Distribution Appendix when you suspect an unusual combination of events was reported in a report summary line. Usually, the average and maximum times or counts are sufficient to highlight a resource usage problem.

However, if you suspect the mean value to be masking an unusual distribution you can draw on the detail contained in the IMS Monitor output records.

For example, suppose you are investigating a change in the scheduling algorithm for a particular transaction and need to know how many transactions were able to be processed for each scheduling of an application program. The following figure shows a possible histogram for the processed transactions:

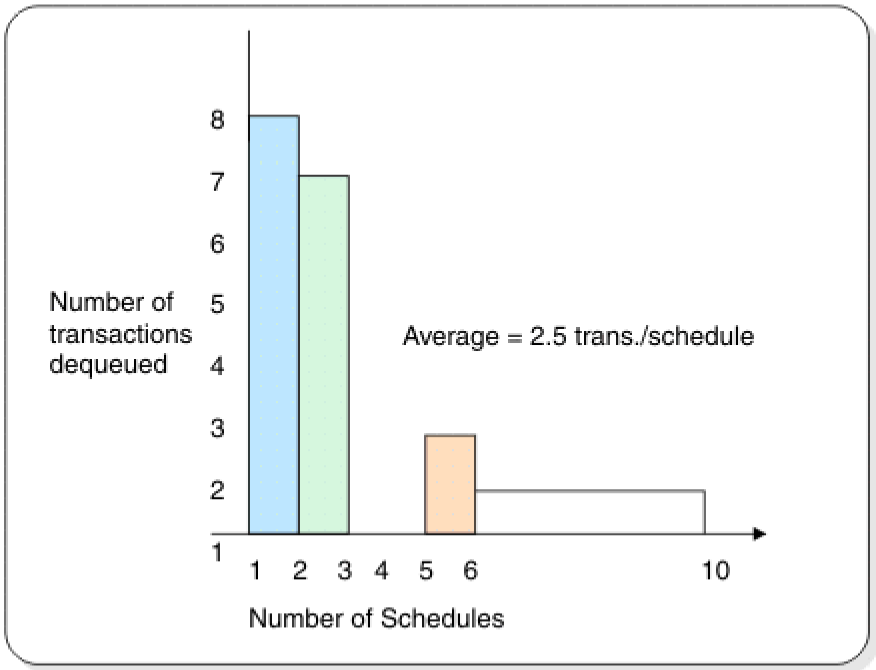


Figure 68. Number of transactions processed for each scheduling of an application program

The Average=2.5 transactions pre schedule. The distribution in the figure suggests that many schedules were able to process only one or two transactions, and few schedules significantly exhausted the queue. The distribution data for the histogram is as follows:

```
Number of schedules    1    2    3    4    5    6-10    >10

Transactions dequeued  8    7    0    0    2    1       0
```

The Distribution Appendix presents the histogram data in the form of two lines:

- The first line shows the intervals, prefixed by a cross reference to an individual line on the earlier output.
- The second line gives the number of events occurring in those intervals.

This data appears as follows:

```
# 950B...0...1...2...3...4...5...10...15...30...90...INF
            8   7   0   0   2   1    0    0    0    0
```

The cross reference 950B points to a unique report line. For example, the Transaction-Queuing report on the appropriate line for the transaction of interest show, 950A,B under the column headed DISTRIBUTION NUMBER. Use the reference number 950B to locate the data in the Distribution Appendix. The 950A reference points to the data for the number of transactions in the queue at schedule time.

IMS Monitor reports for MSC

The IMS Monitor Report Print program includes three reports that highlight message events caused by Multiple Systems Coupling.

- MSC Traffic report

  This report shows the enqueue and dequeue counts of messages that use the various link paths for the monitor interval.
- MSC Summary report

  This report shows summaries of:

  – The traffic queues for each input transaction name
  – The traffic queues for each destination name
  – The traffic queues for each link number
  – The traffic queues for each destination system
- MSC Queuing Summary report

  This report is generated only when intersystem messages are queued on the local system before being sent to the destination system. The local system must be an intermediate system. This report shows:

  – Maximum time messages spend in queues
  – Average time messages spend in queues
  – Maximum queue lengths
  – Maximum queue counts
  – Total number of messages queued for all links the local system participates in

All three of the reports can have entries in the Distribution Appendix. You can examine the frequency distributions of the traffic if you suspect unusual transmission patterns.

Related concepts

"Overview of IMS Monitor reports" on page 654
A list of reports available from data collected by the IMS Monitor, together with the principal performance data they contain, is shown in the following table.

"Monitoring line activity" on page 670
You can obtain a summary of all occurrences of activity for each node that handles message traffic during the monitored interval. The elapsed times and NOT-WAIT times are given in categories of total, mean, and maximum times for each communication line in the Communication Summary report.

Related tasks

Monitoring and tuning multiple systems (Communications and Connections)

MSC Traffic report

The MSC Traffic report reveals the individual queue loads for all traffic between partner systems of which the monitored system is the local system. The report lists all the unique system identification numbers (SIDs) that are defined for communications for that local system.

It then summarizes the total messages queued and dequeued for each combination of the following variables:

- Input name (terminal or program that was a message source)
- Destination name (terminal or program)
- Input system (SID)
- Destination system (SID)
- Link number
- Link type (CTC, MTM, TCP/IP, or VTAM)

The following example illustrates the MSC-traffic report. If a message originates in the local system, its presence is accounted for in the dequeue counts only. Messages with local destinations appear only in the enqueue count.

```
     IMS MONITOR  ****MSC TRAFFIC REPORT****      TRACE START 1993 130   5:55:15    TRACE STOP  1993
130   5:59:49  PAGE 0151
LOCAL SID VALUES = 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115
 INPUT     DESTIN.   INPUT  DEST.  LINK  LINK  ENQUEUE  DEQUEUE
  NAME      NAME      SID    SID    NO.  TYPE    COUNT    COUNT
  ____      ____      ___    ___    __   ____    _____    _____
DSWT3685  DSWT3685     2      2     2    C-C       0        1
          SC6Y         2    102     2    C-C       1        0
DSWT6161  DSWT6161     3      3     3    C-C       0        1
          SC6Z         3    103     3    C-C       1        0
DSWT3838  DSWT3838     2      2     2    C-C       0        1
          SC6Y         2    102     2    C-C       1        0
DSWT4618  DSWT4618     3      3     3    C-C       0        1
          SC6Z         3    103     3    C-C       1        0
DSWT3903  DSWT3903     3      3     3    C-C       0        1
          SC2Z         3    103     3    C-C       1        0
DSWT5418  DSWT5418     3      3     3    C-C       0        1
          SC4Z         3    103     3    C-C       1        0
DSWT4673  DSWT4673     3      3     3    C-C       0        1
          SC2Z         3    103     3    C-C       1        0
DSWT5141  DSWT5141    45     45    45   VTAM       0        1
          PS3X        45    145    45   VTAM       1        0
DSWT4391  DSWT4391     2      2     2    C-C       0        1
          SC4Y         2    102     2    C-C       1        0
DSWT3324  DSWT3324    17     17    17   VTAM       0        1
          IT1Y        17    117    17   VTAM       1        0
DSWT4781  DSWT4781     3      3     3    C-C       0        1
          SC4Z         3    103     3    C-C       1        0
DSWT3525  DSWT3525     3      3     3    C-C       0        1
          SC6Z         3    103     3    C-C       1        0
DSWT4542  DSWT4542     2      2     2    C-C       0        1
          SC6Y         2    102     2    C-C       1        0
DSWT5796  DSWT5796     3      3     3    C-C       0        1
          SC2Z         3    103     3    C-C       1        0
DSWT4782  DSWT4782     3      3     3    C-C       0        1
          SC6Z         3    103     3    C-C       1        0
DSWT4633  DSWT4633     2      2     2    C-C       0        1
          SC6Y         2    102     2    C-C       1        0
DSWT3655  DSWT3655    12     12    12   VTAM       0        1
          SC6U        12    112    12   VTAM       1        0
DSWT3892  DSWT3892     2      2     2    C-C       0        1
          SC6Y         2    102     2    C-C       1        0
DSWT3338  DSWT3338     4      4     4   VTAM       0        1
          SC2U         4    104     4   VTAM       1        0
DSWT4681  DSWT4681     3      3     3    C-C       0        1
          SC4Z         3    103     3    C-C       1        0
DSWT4482  DSWT4482     2      2     2    C-C       0        1
          SC6Y         2    102     2    C-C       1        0
DSWT4902  DSWT4902     3      3     3    C-C       0        1
          SC2Z         3    103     3    C-C       1        0
DSWT4558  DSWT4558     3      3     3    C-C       0        1
DSWT4558  SC6Z         3    103     3    C-C       1        0
DSWT3925  DSWT3925     2      2     2    C-C       0        1
TOTAL TRAFFIC
                                                  1353     1359
_____ _______
```

Related tasks

Monitoring and tuning multiple systems (Communications and Connections)

MSC Summaries report

The MSC Summaries report shows you the enqueue and dequeue activity for messages that are handled by the local system but are part of the Multiple System Coupling (MSC) traffic.

The report format is illustrated in the following example.

The first set of queuing counts shows how many transactions of each type were entered in the monitor interval, and how many were subsequently dequeued.

The second set of counts summarizes the total traffic for each destination name. You can distinguish the primary transactions and responses by the resource names and examine the relative servicing of the link transmissions using the difference between the enqueue and dequeue counts.

The third set of counts lists the active links by link number, and you can determine if there is buildup on the link by the difference in the enqueue and dequeue counts.

The fourth set of counts records the traffic that is going to other systems by all link paths. You can judge by the difference in enqueue and dequeue counts whether the overall pattern of link priorities to one or more systems is causing buildup of cross-system traffic.

```
     IMS MONITOR   ****MSC SUMMARIES****       TRACE START 1993 130   5:55:15   TRACE STOP  1993 130
5:59:49  PAGE 0178
<--SUMMARY BY INPUT NAME-->|<--SUMMARY BY DESTINATION NAME-->|<--SUMMARY BY LOGICAL LINK-->|<--SUMMARY
BY DEST. SYS. ID-->
                           |                                  |                              |
 INPUT    ENQUEUE  DEQUEUE |   DESTIN.   ENQUEUE   DEQUEUE   |    LINK  ENQUEUE   DEQUEUE  |    DEST.
ENQUEUE   DEQUEUE
  NAME      COUNT    COUNT |    NAME       COUNT     COUNT   |    NO.     COUNT     COUNT  |     SID
COUNT     COUNT
  ----      -----   -----          ----       -----     -----         ---     -----     -----
---     -----     -----
                           |                                  |                              |
DSWT3577       1      1    |  DSWT4358         0         1   |                              |
DSWT4048       1      1    |  DSWT5988         0         1   |                              |
DSWT5216       1      1    |  DSWT5457         0         1   |                              |
DSWT4776       1      1    |  DSWT5187         0         1   |                              |
DSWT5496       1      1    |  DSWT3312         0         2   |                              |
DSWT5277       1      1    |  DSWT5604         0         1   |                              |
DSWT5711       1      1    |  DSWT3347         0         1   |                              |
DSWT5274       1      1    |  DSWT5338         0         1   |                              |
DSWT5807       1      1    |  DSWT3268         0         1   |                              |
DSWT3685       1      1    |  DSWT3676         0         1   |                              |
DSWT6161       1      1    |  DSWT5428         0         1   |                              |
DSWT3838       1      1    |  DSWT5395         0         1   |                              |
DSWT4618       1      1    |  DSWT4168         0         1   |                              |
DSWT3903       1      1    |  DSWT5061         0         1   |                              |
DSWT5418       1      1    |  DSWT3511         0         1   |                              |
DSWT4673       1      1    |  DSWT3363         0         1   |                              |
DSWT5141       1      1    |  DSWT3674         0         1   |                              |
DSWT4391       1      1    |  DSWT4467         0         1   |                              |
DSWT3324       1      1    |  DSWT4501         0         1   |                              |
DSWT4781       1      1    |  DSWT5037         0         1   |                              |
DSWT3525       1      1    |  DSWT4298         0         1   |                              |
DSWT4542       1      1    |  DSWT5778         0         1   |                              |
DSWT5796       1      1    |  DSWT4003         0         1   |                              |
DSWT4782       1      1    |  DSWT3988         0         1   |                              |
DSWT4633       1      1    |  DSWT4217         0         1   |                              |
DSWT3655       1      1    |  DSWT6135         0         1   |                              |
DSWT3892       1      1    |  DSWT5147         0         1   |                              |
DSWT3338       1      1    |  DSWT5381         0         1   |                              |
DSWT4681       1      1    |  DSWT5593         0         1   |                              |
DSWT4482       1      1    |  DSWT3304         0         1   |                              |
DSWT4902       1      1    |  DSWT5081         0         1   |                              |
DSWT4558       1      1    |  DSWT4671         0         1   |                              |
                           |  DSWT3655         0         1   |                              |
                           |  DSWT3892         0         1   |                              |
                           |  DSWT3338         0         1   |                              |
                           |  DSWT4681         0         1   |                              |
                           |  DSWT4482         0         1   |                              |
                           |  DSWT4902         0         1   |                              |
                           |  DSWT4558         0         1   |                              |
                           |  DSWT3925         0         1   |                              |
```

Related tasks

Monitoring and tuning multiple systems (Communications and Connections)

MSC Queuing Summary report

The MSC Queuing Summary report provides information about intersystem message traffic only. You can use the sample of traffic recorded in the IMS Monitor interval to examine the maximum and average time messages spend in queues waiting to be sent on active links.

You can detect whether the link priorities are causing undue delay of primary messages through the intermediate system, or whether there is a build up of responses. The report shows the logical link paths for this system which is an intermediate system. Each incoming link number shows the number of messages that are queued before transmission on their specified outward bound link number. The maximum queue count is given as well as the maximum and average time on the intermediate system queues.

The following example shows the MCS queuing summary report.

```
    IMS MONITOR  ****MSC QUEUING SUMMARY****    TRACE START 2011 063, 10:33:13  TRACE STOP
2011 063, 10:34:54
PAGE 0022

ENQUE.....      DEQUE.....                MAX.Q   MAX.    MEAN
LINK NO.  TYPE  LINK NO.  TYPE  MESSAGES  LENGTH  Q TIME  Q TIME


-------   ----  -------   ----  --------  ------  ------ ------

     13   VTAM        6   M-M          2       1    2342   1234

TOTALS...                              2                   1234
```

Related tasks

Monitoring and tuning multiple systems (Communications and Connections)

Chapter 49. IMS Monitor reports for DBCTL

DBCTL monitoring provides data about the processing that occurs when a CCTL transaction accesses DBCTL databases. The CCTL gains this access using database resource adapter (DRA) requests.

This topic describes:

- The events that the IMS Monitor collects
- The content of the reports produced by the IMS Monitor Report Print Program

Monitoring has different meanings for DBCTL and DB/DC. For DB/DC, the end user enters the transaction on a terminal. The transaction is processed by IMS and then returns a result to the user. Transaction characteristics that are monitored include total response time and the occurrences of resource contentions (for example, PSB schedule wait time, and database I/Os).

DBCTL has neither transactions nor terminal end users. It does, however, work on behalf of transactions entered by CCTL terminal users.

A typical sequence of these DRA requests would be:

1. A SCHED request to get a PSB scheduled in DBCTL
2. A DL/I request to make database calls
3. A sync-point request, COMMTERM, to commit the updates and release the PSB

The DBCTL process that encompasses these requests is called a unit of recovery (UOR).

DBCTL provides monitoring data about UORs, such as: total time UOR existed, wait time for PSB schedule, and I/Os during database calls. This information is very similar to IMS transaction monitor data. In a DBCTL-CCTL system, however, the UOR data represents only part of the total processing of a CCTL transaction. Therefore, CCTL monitor data is necessary to get a total view of CCTL transaction performance.

DBCTL does not change the format or usage of the IMS monitor reports. There are reports and fields within reports that are not applicable to DBCTL. Generally, these are in the transaction manager and communication areas. There are some fields that are interpreted differently in a DBCTL environment.

For reports that do not apply to DBCTL, either a heading without data is shown or no report is generated. These reports are:

- Message Queue Pool report
- Message Format Buffer Pool report
- Communication Summary report
- Communication IWAIT report
- Line Functions report
- MSC Traffic report
- MSC Summaries report
- MSC Queuing Summary report

The term *region* in IMS Monitor reports refers to a PST assigned to a specific dependent region that processes specific IMS transactions. In DBCTL monitor reports the term *region* still applies to a PST. A PST can service one CCTL thread (transaction) at a time. However, CCTL threads change, resulting in one PST servicing many different CCTL transactions. Since multiple CCTLs can connect to DBCTL, the PST can actually service transactions from different CCTLs.

All of the threads built for a CCTL carry the job name of the CCTL. This appears as the same job name for many regions in the General Reports.

Within a trace interval, a thread can be assigned to multiple CCTLs, but it can only be assigned to one CCTL at any instant of time. So, depending on the number of CCTLs attached to DBCTL, the Region Summary reports can show:

- One region with only one job name.
- One region with different job names.
- Multiple regions with different job names. Some regions can have the same job name and some can have different job names.
- Multiple regions with only one job name.

Any monitor report for a region is a summary of all the CCTLs a thread served during the trace interval (for example, the elapsed time for all CCTLs that a thread has been assigned to during the trace interval).

The reports generated by the IMS Monitor are the same for BMPs and non-message BMPs.

UOR elapsed times are spent in DBCTL, not in the DRA. The time spent in the DRA is considered part of the CCTL, therefore the DRA time is not reported by any DBCTL statistics.

IMS Monitor trace event intervals

The IMS Monitor trace interval is defined by the master terminal operator's use of the /TRACE command between the start and stop command entries. The online IMS events are recorded in IMS Monitor records placed in the IMSMON data set.

The event timings are related to dependent region activity. The following figure shows the boundaries of the timed event intervals.

The Monitor trace interval includes the following intervals:

- Scheduling and Termination
  - Block loader busy
  - Intent failures (exclusive intent and data sharing) and Schedule failures (PSB busy and space failure)
  - Sched/Term elapsed
    - NOT-WAIT
    - ACBLIB waits
    - DB Flush waits
    - DB CLOSE waits
- Region occupancy (which overlaps with all of Sched/Term elapsed)
  - Schedule to first call
  - Elapsed execution

The NOT-WAIT time for a region is the elapsed time not accounted for by wait time. Any delay coming from either paging or the processor being dispatched for a higher priority task results in an increase in the NOT-WAIT times.

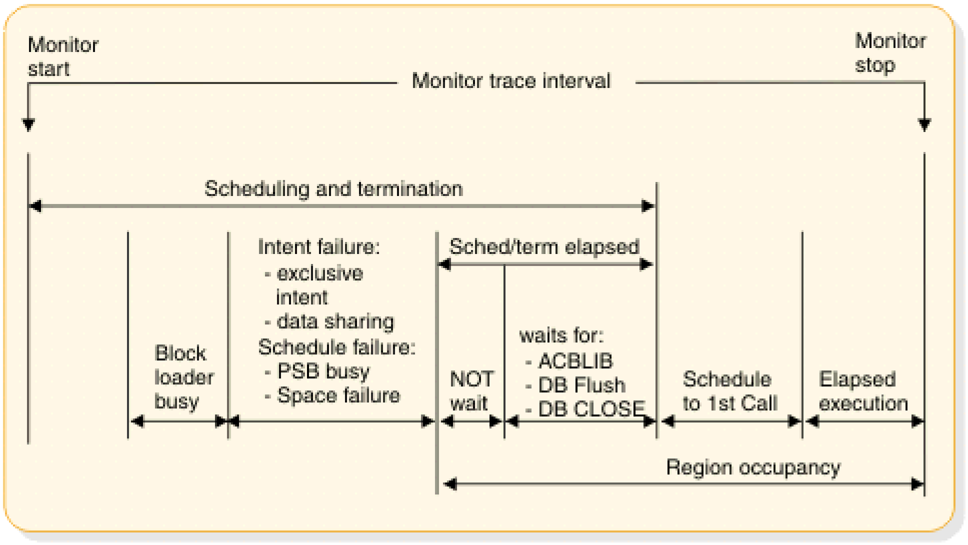


Figure 69. IMS Monitor trace event intervals

Overview of IMS Monitor reports

A list of reports available from data collected by the IMS Monitor, together with the principal performance data they contain, is shown in the following table.

Related reading: For a description of those reports marked "DB", see Chapter 47, “DB Monitor reports,” on page 573.

The order of the reports listed in the table matches the sequence of the output from the IMS Monitor Report Print program. The duration of a monitoring snapshot might not include certain events necessary for an individual report, in which case only report headings or partial data are produced.

Table 65. IMS Monitor reports output sequence and information

Report name	Principal information
System Configuration	Monitor run documentation
Database Buffer Pool (DB)	Count of DB calls and I/O per transaction
VSAM Buffer Pool (DB)	Count of inserts and I/Os
Latch Conflict Statistics	IMS internal processing
Region and Jobname	Monitor run documentation
Region Summary	Elapsed times and count of DL/I calls
Region Wait	Wait times
Programs by Region	Elapsed times for region usage
Program Summary	Overall program statistics
Program I/O (DB)	Wait times/PCB
Reports	Count of space failures and deadlocks
Run Profile	Monitor run documentation
Call Summary (DB)	Call counts and timings/segment type
Distribution Appendix	Event frequency distributions

The majority of the data items in IMS Monitor reports are elapsed times. These are normally expressed in microseconds. An entry of 1876534 is 1.876534 seconds or 1876 milliseconds. Any times that do not follow this convention show the unit of measure on the report.

You can also find counts of events under the heading OCCURRENCES, and some figures that represent the number of bytes.

Documenting the monitoring run

For each trace interval, several general reports or overall summaries are generated for the processing that took place. You can use these reports as part of your IMS Monitor run documentation.

It is important to record, as accurately as possible, the conditions under which the trace was taken. Your documentation can include system status information (obtained by the **/DISPLAY** command) several times before and after the trace, an expected profile of the CCTL transaction activity, and any desired processing events. The trace interval should represent typical processing loads and not be a biased or inadequate historical record.

Adding to the System Configuration report data

The first general report (titled SYSTEM CONFIGURATION) is found under the page heading BUFFER POOL STATISTICS. It shows the modification level of the IMS and z/OS systems. You can choose to add a list of IMS APARs applied and include the service levels of the application programs, especially if the latter are not permanent programs or are part of a staged implementation. The system configuration output is illustrated in the following section.

Recording the Monitor trace interval

The heading of most IMS Monitor reports carries the trace start and stop times. It is shown in the format YEAR DAY (Julian) HH:MM:SS. The overall length of the trace interval is given in milliseconds under the title MONITOR OVERHEAD DATA. The following example shows how many trace records were placed on the IMSMON data set. An example of the monitor trace interval recording is shown in the following figure.

```
***IMS  MONITOR***  BUFFER POOL STATISTICS   TRACE START 1993 130   5:55:15   TRACE STOP  1993 130
5:59:49  PAGE 0001

                   S Y S T E M   C O N F I G U R A T I O N



                                                 SYSTEM CONFIGURATION      :
                                                    IMS VERSION            :   4
                                                 RELEASE LEVEL             :
                                                 MODIFICATION NUMBER       :
```

Completing the Monitor run profile

A compact set of processing ratios will be found at the end of the Run Profile report. The statistics summarize, for the monitor interval, the UOR throughput and the degree of DL/I and I/O activity. An example of the report is shown in the following example.

```
 IMS MONITOR   **RUN PROFILE**              TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0184
TRACE ELAPSED TIME IN SECONDS.............274.6
TOTAL NUMBER OF MESSAGES DEQUEUED........1403
TOTAL NUMBER OF SCHEDULES.................173
NUMBER OF TRANSACTIONS PER SECOND...........5.1
TOTAL NUMBER OF DL/I CALLS ISSUED.......18632
NUMBER OF DL/I CALLS PER TRANSACTION.......13.2
NUMBER OF OSAM BUFFER POOL I/O'S........11236,    8.0 PER TRANSACTION
NUMBER OF MESSAGE QUEUE POOL I/O'S..........0,    0.0 PER TRANSACTION
NUMBER OF FORMAT BUFFER POOL I/O'S..........0,    0.0 PER TRANSACTION
RATIO OF PROGRAM ELAPSED TO DL/I ELAPSED:
              REGION   1:   1.09
              REGION   2:   1.09
              REGION   3:   1.00
```

```
               REGION   4:   1.02
               REGION   5:   1.01
               REGION   6:   1.00
               REGION   7:   1.00
               REGION   8:   1.00
               REGION   9:   1.17
               REGION  10:   1.00
               REGION  11:   1.00
               REGION  49:   1.03
               REGION  50:   1.19
 RATIO OF DL/I ELAPSED TO DL/I IWAIT:
               REGION   1: 325.65
               REGION   2:  73.49
               REGION   4: 100.35
               REGION   5:  85.76
               REGION   6:  82.99
               REGION  47:  95.64
               REGION  48:  45.93
               REGION  49:   9.22
```

The lower half of the Run Profile report shows several ratios:

- Program elapsed time to DL/I elapsed time for each region
- DL/I elapsed time to wait time during DL/I processing
- Program elapsed time to other subsystem call elapsed time
- DL/I elapsed time to other subsystem call elapsed time

Each region is identified by a sequence number, starting at region 1.

There are some generalized processing ratios that are given at the end of several buffer pool statistics reports. You can include them in the documented profile of the trace interval. These are not specific to one UOR or system resource but can be used as indicators of variation across a series of monitor runs. The ratios are:

- The total number of OSAM reads + OSAM writes + all waits divided by the total number of transactions.

  From the Message Queue Pool report, this ratio indicates on a per transaction basis the physical I/O activity required to handle the scheduling function.
- The total number of OSAM reads + OSAM writes + BISAM reads divided by the total number of transactions.

  From the Database Buffer Pool report, this ratio indicates on a per transaction basis the physical I/O activity required to handle the database buffering function.

Verifying IMS Monitor report occurrences

When you examine the output from the IMS Monitor Report Print Program, the presence of a report heading does not necessarily mean that appropriate data will be listed. System definition options and utility control statements also affect the content of the output as follows:

- The output does not include a Call Summary report unless a control statement specifies DLI.
- The output does not include a set of Distribution reports unless a control statement specifies DIS or DISTRIBUTION. The column headed DISTRIBUTION NUMBER that occurs on many of the reports contains cross-references to items included in the Distribution reports.
- The output consists of just a Call Summary report if a control statement specified ONLY DLI.

Because many of the summary reports require system status to calculate the difference between start and end values, and this status is obtained during the /TRACE SET OFF processing, the IMS Monitor execution must end before termination of the IMS control region. If the trace was not stopped properly, the following message is issued:

```
NO DATABASE BUFFER POOL TRACES AT END TIME ON MONITOR LOG TAPE
****DATABASE BUFFER POOL REPORT CANCELLED****
```

Similarly, other summary reports are not produced.

The series of reports titled Buffer Pool Statistics do not include a VSAM Buffer Pool section unless the database in IMS.ACBLIB uses the VSAM access method. If VSAM is not used, the following message is issued:

```
NO VSAM BUFFER POOL TRACES ON MONITOR LOG TAPE
****VSAM BUFFER POOL REPORT CANCELLED****
```

If the source data used to formulate a particular IMS Monitor report, or a section of that report, has not been recorded by the IMS Monitor during the trace interval, the report contains only the headings.

Monitoring activity in dependent regions

The IMS Monitor gathers timing information for every dependent region identified in the **/TRACE** command (a CCTL thread) active during the trace interval. It records the total of the elapsed times for each event, the maximum individual time encountered, and the average time.

There are three major reports that display timings. The reports and a list of their content are:

- **Region Summary Report**
  - Scheduling and termination
  - Schedule end to first call
  - Elapsed execution with separate summaries shown for:
    - DL/I calls
    - External subsystem service and command calls
    - External subsystem database access calls
    - Checkpoint processing
    - Region occupancy
- **Region Wait**
  - Waits during scheduling and termination
  - Waits during DL/I calls
  - Waits during external subsystem calls
  - Waits during checkpoint
- **Programs by Region**
  - Elapsed execution
  - Schedule end to first call

In this report, "program name" is the PSB name for the UOR.

These three reports are illustrated in the following examples.

Activities in dependent regions are placed in five timing categories:

- Elapsed time for scheduling and termination

  The scheduling process includes many preparatory events such as block loading from an active IMS.ACBLIBA/B data set, and obtaining ownership of the PSB. The time required to terminate is the time it takes DBCTL to complete this process after receiving a request to terminate the UOR.
- Elapsed time from end of schedule to first call

  This is the time from when DBCTL completes scheduling until the time DBCTL reviews the first DL/I call. Events that occur during this time are all outside of DBCTL, either in the database resource adapter (DRA) or the CCTL.
- Program elapsed time, including all calls

  This time encompasses the major UOR processing, measured from the first DL/I call to the call that terminates a UOR.

- Elapsed time performing DL/I calls

  This time includes all DL/I calls. The time in DBCTL is recorded and summed.

The following example shows a region summary report.

```
 IMS MONITOR ****REGION SUMMARY****      TRACE START 1993 130   5:55:15     TRACE STOP  1993 130   5:59:49  PAGE 0011
                                             (A)                                 (B)
                                 ........ELAPSED TIME.........       NOT IWAIT TIME(ELAPSED-IWAIT)
DISTRIBUTION
               OCCURRENCES       TOTAL        MEAN     MAXIMUM       TOTAL        MEAN     MAXIMUM
NUMBER
               -----------       -----        ----     -------       -----        ----     -------
______
SCHEDULING AND TERMINATION
__________ ___ ___________
**REGION     5           5        4146         829         948        4146         829         948
287A,B
**REGION     6           7        6028         861        1067        6028         861        1067
214A,B
**REGION     8           8        6847         855        1098        6847         855        1098
129A,B
**REGION    10           7        9664        1380        3668        9664        1380        3668
272A,B
**REGION    47           6        5482         913        1021        5482         913        1021
145A,B
**REGION    49           3        2612         870         917        2612         870         917
443A,B
**TOTALS               123      126042        1024                  126042        1024
SCHEDULE TO FIRST CALL
________ __ _____ ____
**REGION     1           1    15479797    15479797    15479797
555
**REGION     2           1    22376350    22376350    22376350
564
**REGION     3           1    15169488    15169488    15169488
578
**REGION     4           1    48146258    48146258    48146258
584
**REGION    48           1      795351      795351      795351
592
**REGION    49           4     2960425      740106     2951746
442
**REGION    50           1    15713464    15713464    15713464
575
**TOTALS               168   514286738     3061230
ELAPSED EXECUTION
_______ _________
**REGION     1           1   290146255   290146255
290146255                                                           1
**REGION     2           1   252290108   252290108
252290108                                                           2
**REGION     3           1   259496970   259496970
259496970                                                           3
**REGION     4           1   322812716   322812716
322812716                                                           4
**REGION    48           1   273871107   273871107   273871107
48
**REGION    49           4   271703421    67925855   155176058
49
**REGION    50           1   290379922   290379922   290379922
50
**TOTALS               173 14238540145    82303700
DL/I CALLS                                                                                          IWT/CALL(C)
____ _____                                                                                          ___________
**REGION     1          60   264626241     4410437    88981490   263813671     4396894    88970053       0.76
247A,B,C
**REGION     2         223   230505269     1033655    61048758   227368742     1019590    61011153       0.73
237A,B,C
**REGION     3          29   257704383     8886358    69000514   257704383     8886358    69000514       0.00
98A,B,C
**REGION     4         792   313735347      396130    52439653   310609035      392183    52439653       0.22
180A,B,C
**REGION    49         592   262886317      444064    30202068   234394017      395935    30159782       2.46
177A,B,C
**REGION    50          36   242591451     6738651    48651260   242591451     6738651    48651260       0.00
289A,B,C
**TOTALS             18632 12386905286      664818             12024562411      645371                   0.97
IDLE FOR INTENT
---- --- ------
                      NONE
CHECKPOINT
__________            NONE
REGION OCCUPANCY
______ _________
**REGION     1    100.0%
**REGION     2    100.0%
**REGION     3    100.0%
**REGION     4    100.0%
**REGION    48    100.0%
```

```
**REGION    49    100.0%
**REGION    50    100.0%
```

The following example shows a region wait report.

```
 IMS MONITOR  ****REGION IWAIT****        TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0023
                                 .........IWAIT TIME..........                           DISTRIBUTION
**REGION     5 OCCURRENCES       TOTAL        MEAN     MAXIMUM       FUNCTION  MODULE      NUMBER
  ______      ___________       _____        ____     _______       ________  ______      ______
SCHEDULING + TERMINATION
__________   ___________
   SUB-TOTAL
   _________
   TOTAL
   _____
DL/I CALLS
____ _____
                        11      181816       16528       24375    DD=IMMSTR2A   DBH            117
                         1        1667        1667        1667    PSB=BMPFPE06  BLR-64BIT       11
                         1        1635        1635        1635    INT=BMPFPE06  BLR-64BIT       12
                         8      112831       14103       17846    DD=IMMSTR1A   DBH            118
                         5       85460       17092       33717    DD=IMMSTR3A   DBH            119
                         5       58420       11684       14643    DD=IMINDEXA   VBH            120
                        12      173866       14488       22152    DD=PRODCNTA   VBH            121
                         3      100576       33525       68373    DD=IMMSTR2B   DBH            428
                         1       17921       17921       17921    DD=IMMSTR3B   DBH            429
                         1       17195       17195       17195    DD=IMMSTR1B   DBH            430
                         1       13577       13577       13577    DD=IMINDEXB   VBH            431
                         3       49928       16642       20396    DD=PRODCNTB   VBH            432
                         4       10973        2743        2787    DD=ITEMACTB   DBH            453
                         2       37680       18840       27664    DD=IAINDEXB   VBH            454
                        49     1500067       30613      138284    DD=INVENTRA   DBH            472
                        23      345595       15025       27613    DD=VENDORDA   VBH            473
                         1      342952      342952      342952    PI=VENDORDA...1              498
                         1       14612       14612       14612    PI=VNSINDXA...1              499
                         6       69203       11533       19492    DD=VNSINDXA   VBH            500
   TOTAL
   _____               136     3132672       23034
```

The following example shows a programs-by-region report.

```
 IMS MONITOR  ****PROGRAMS BY REGION****    TRACE START 1993 130   5:55:15    TRACE STOP  1993 130
5:59:49  PAGE 0069
                                             (A)                                 (B)
                                  ELAPSED EXECUTION TIME            SCHEDULING END TO FIRST CALL
DISTRIBUTION
               OCCURRENCES       TOTAL        MEAN     MAXIMUM       TOTAL        MEAN     MAXIMUM
NUMBER
               ___________       _____        ____     _______       _____        ____     _______
______
**REGION     1
________   _ 1
    PROGSC6D             1   290146255   290146255   290146255    15479797    15479797
15479797         885A,B
 REGION TOTALS           1   290146255   290146255                15479797    15479797
**REGION     2
________   _ 2
    PROGIT8C             1   252290108   252290108   252290108    22376350    22376350
22376350         889A,B
 REGION TOTALS           1   252290108   252290108                22376350    22376350
**REGION     3
________   _ 3
    PROGTS1C             1   259496970   259496970   259496970    15169488    15169488
15169488         893A,B
 REGION TOTALS           1   259496970   259496970                15169488    15169488
**REGION     4
________   _ 4
    PROGPS3D             1   322812716   322812716   322812716    48146258    48146258
48146258         897A,B
 REGION TOTALS           1   322812716   322812716                48146258    48146258
**REGION     5
________   _ 5
    PROGPS3A             2    62893103    31446551    40693590        5435        2717
2862         901A,B
    PROGTS1B             1    61794787    61794787    61794787        2790        2790
2790        1271A,B
    PROGPS3B             1    18294458    18294458    18294458        3104        3104
3104        1350A,B
```

```
    PROGIT2B             1    36095342    36095342    36095342        2731        2731
2731        1363A,B
    PROGSC2A             1    93902771    93902771    93902771     1667791     1667791
1667791        1401A,B
 REGION TOTALS           6   272980461    45496743                 1681851      280308
**REGION     6
________   _ 6
    PROGIT1B             2    39000315    19500157    23703429        5286        2643
2801        905A,B
    PROGTS1B             1    34293636    34293636    34293636        3136        3136
3136        1207A,B
    PROGPS3A             1    51887767    51887767    51887767        2534        2534
2534        1278A,B
    PROGPS3B             2    67375031    33687515    40291430    17210570     8605285
17213287        1328A,B
    PROGIT8A             1    69132416    69132416    69132416        3291        3291
3291        1359A,B
    PROGSC4A             1    30165017    30165017    30165017        2571        2571
2571        1433A,B
 REGION TOTALS           8   291854182    36481772                17193752     2149219
**REGION     7
________   _ 7
    PROGSC2B             1   269618583   269618583   269618583     5047875     5047875
5047875         909A,B
 REGION TOTALS           1   269618583   269618583                 5047875     5047875
**REGION     8
________   _ 8
    PROGIT8A             1     5181039     5181039     5181039        2928        2928
2928        913A,B
    PROGPS3A             1    27304257    27304257    27304257        3350        3350
3350        1132A,B
    PROGSC4B             1    37286872    37286872    37286872        3009        3009
3009        1255A,B
    PROGIT2A             1    36902995    36902995    36902995        2850        2850
2850        1298A,B
    PROGIT1B             1    30407479    30407479    30407479        2565        2565
2565        1336A,B
    PROGIT1A             3   109875360    36625120    45190114     4279008     1426336
4272096        1357A,B
    PROGIT8B             1    23405220    23405220    23405220        2679        2679
2679        1395A,B
 REGION TOTALS           9   270363222    30040358                 4296389      477376
```

Detecting database processing intent conflicts

The IMS Monitor records the intervals when a region is in an idle state waiting to update a database owned exclusively by another already scheduled application program.

You can see the total, maximum, and average idle times in IDLE FOR INTENT following the DL/I calls. The elapsed time during the unsuccessful scheduling of a program in that region is included in the summary line times for that region.

The region can fail to be scheduled even when ownership of that database is released. The number of times processing is held up by intent failure is separately tallied under the title INTENT FAILURE SUMMARY. The report is illustrated in “Detecting database processing intent conflicts” on page 604. This report shows which PSBs are in conflict because of exclusive intent for a segment type and the database name in question.

Examining the effects of checkpoints

The checkpoint line of the Region Summary report at the end of the region 0 summary, shows the following:

- The number of times that a system checkpoint is taken during the monitor interval
- The elapsed times
- The not-wait times

Checkpoint processing can be initiated by the control program at a specified frequency determined by the number of records placed on the system log. Other checkpoints can be caused by operator commands.

The wait time experienced during checkpoints is reported at the end of the first region summary on the Region Wait report. You can detect delays for each combination of DD name and module code. Typical entries here are for the message queue data sets and the restart data set. If a wait for storage is the cause, the entry under the FUNCTION column is STG.= followed by the identification of the pool.

Measuring region occupancy

Region occupancy shows the ratio of elapsed time a PST spent processing UORs to the total time of the monitor interval.

Monitoring application program elapsed time

The IMS Monitor can record measurements of elapsed times for each UOR. It does this during the monitored interval while other UORs are executing concurrently. Elapsed times are calculated from the start of the first DL/I (or other) call to the end of that program.

You can distinguish between time spent in application code and in DL/I processing. The event intervals are illustrated in the following figure.

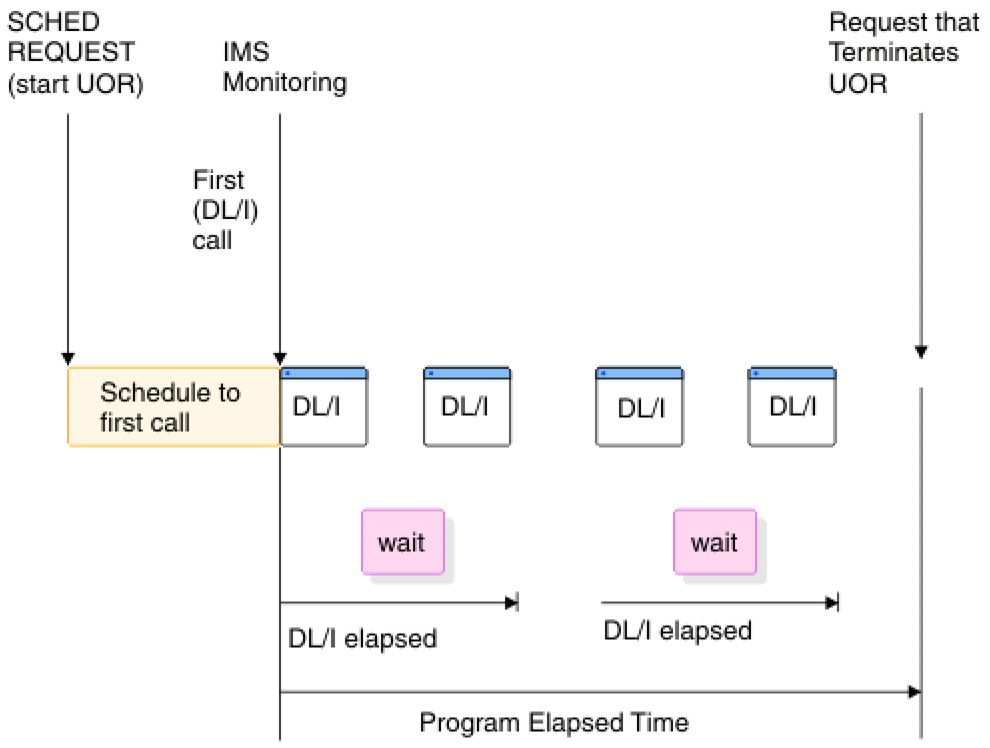


Figure 70. Event intervals

Within the elapsed time for a DL/I call, the wait time to obtain segment data is recorded separately. Also, the elapsed time from schedule to first call is separately recorded. This time covers the processing in the CCTL and the database resource adapter (DRA).

The elapsed times are given in the Program Summary report. The following example shows the report. Programs are identified by their PSB name on individual lines in the report. Each line gives a summary of the activity for that PSB during the measured interval. The total number of schedules, DL/I calls, transactions dequeued, and waits for a DL/I call for I/O are given. The report line gives calculated average times for elapsed time per schedule, processor time per schedule, schedule to first DL/I call per schedule, and elapsed time per transaction. Frequencies for calls per transaction, I/O waits per DL/I call, and transactions dequeued per schedule are also given. A TOTALS line summarizes all activity for the PSBs active during the monitored interval. (The PSB DUMMY line is a reconciliation for any incomplete scheduling caused by a region being stopped during scheduling or for a program that experiences a pseudo-abend.)

In this report, *transaction* and *schedule* can be interpreted as UOR.

```
     IMS MONITOR   ****PROGRAM SUMMARY****            TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0075
                                                             (A)........(B)........         (A)........(B)........
                                                      I/O   TRAN.       CPU       .        ELAPSED
SCHED.TO    .        ELAPSED
            NO.    TRANS.           CALLS     I/O   IWAITS  DEQD.      TIME    DISTR.       TIME      1ST CALL
DISTR.       TIME
PSBNAME  SCHEDS.    DEQ.     CALLS  /TRAN    IWAITS  /CALL  /SCH.     /SCHED.    NO.       /SCHED.     /SCHED.
NO.       /TRANS.
-------  -------    ----     -----  -----    ------  -----  -----     -------    ---       -------     -------
---       -------

PROGSC6D       1      13        60    4.6        46    0.7   13.0       10010   884A,B   290146255    15479797
886A,B    22318942
PROGIT8C       3      17       225   13.2       166    0.7    5.6       90592   888A,B   256617508    73283259
890A,B    45285442
PROGTS1C       2      25        47    1.8         0    0.0   12.5       10010   892A,B   239190808     7586234
894A,B    19135264
PROGPS3D       1      23       792   34.4       182    0.2   23.0       10010   896A,B   322812716    48146258
898A,B    14035335
PROGPS3A      13      36      1246   34.6       267    0.2    2.7       49782   900A,B    32801812     2228611
902A,B    11845098
PROGIT1B      11      21        99    4.7         0    0.0    1.9        6341   904A,B    23212388     2036217
906A,B    12158870
PROGSC2B       7     155      3068   19.7      1845    0.6   22.1      346112   908A,B    93655514      789390
910A,B     4229603
PROGIT8A      12      28       434   15.5       293    0.6    2.3       34350   912A,B    30196795     1745815
914A,B    12941483
PROGPS2C       1      10       179   17.9       205    1.1   10.0       10010   916A,B   221024429    53642029
918A,B    22102442
PROGTS1B       8      20        54    2.7         0    0.0    2.5        5447   920A,B    39943245        2895
922A,B    15977298
PROGPS3C       1      14       468   33.4       117    0.2   14.0       10010   924A,B   310644485    35978027
926A,B    22188891
PROGIT1C       1       9        32    3.5         0    0.0    9.0       10010   930A,B   304892631    30226173
932A,B    33876959
PROGSC2C       1       9       160   17.7       101    0.6    9.0       10010   934A,B   296909110    22242652
936A,B    32989901
PROGIT2B       8      21       393   18.7        63    0.1    2.6       21703   938A,B    35126671     1798496
940A,B    13381589
PROGIT2C       6      17       211   12.4        39    0.1    2.8       13312   942A,B   288883508    50698467
944A,B   101958885
PROGTS1D       2      26        50    1.9         0    0.0   13.0       10010   950A,B   284944505    10613350
952A,B    21918808
PROGPS3B       8      22       770   35.0       169    0.2    2.7       35737   954A,B    38016279     2149158
956A,B    13824101
PROGIT1A      11      24       106    4.4         0    0.0    2.1        7925   958A,B    30883486     1935855
960A,B    14154931
PROGSC4A       9     163      1775   10.8      5101    2.8   18.1      235921   963A,B    62172947     3011199
965A,B     3432862
PROGSC6C       1      10        44    4.4        38    0.8   10.0       10010   967A,B   228098334    46568124
969A,B    22809833
PROGPS2B      11      28       557   19.8       604    1.0    2.5       35069   971A,B    33309266     1181831
973A,B    13085783
PROGIT8D       1      12       175   14.5       133    0.7   12.0       10010   975A,B   253392289    21274169
977A,B    21116024
PROGSC4C       1      10        98    9.8       349    3.5   10.0       10010   979A,B   248736332    25930126
981A,B    24873633
PROGSC6A       7     157       789    5.0       457    0.5   22.4       11703   983A,B    73936039      115979
985A,B     3296511
PROGIT2A       7      22       430   19.5        71    0.1    3.1       28529   987A,B    37905001        2982
989A,B    12060682
PROGSC2D       1      15       280   18.6       180    0.6   15.0       10010   991A,B   316194222    41527764
993A,B    21079614
PROGPS2A       6      25       490   19.6       548    1.1    4.1       43177   995A,B    58277945     2467506
997A,B    13986707
PROGSC2A       5     121      2363   19.5      1420    0.6   24.2      276187  1001A,B    88906184     6022954
1003A,B    3673809
PROGIT2D       1      20       361   18.0        62    0.1   20.0       10010  1005A,B   386092737   111426279
1007A,B   19304636
PROGSC4B      10     131      1421   10.8      4115    2.8   13.1      617016  1011A,B    53826667     2632409
1013A,B    4108905
PROGSC4D       1      19       197   10.3       668    3.3   19.0       10010  1020A,B   227999124    46667334
1022A,B   11999953
PROGPS2D       1      13       240   18.4       291    1.2   13.0       10010  1025A,B   327602445    52935987
1027A,B   25200188
PROGSC6B       5     140       694    4.9       395    0.5   28.0       16884  1032A,B    78994223     3290769
1034A,B    2821222
PROGIT1D       1      10        36    3.6         0    0.0   10.0       10010  1041A,B   290379922    15713464
1043A,B   29037992
PROGIT8B       8      17       288   16.9       190    0.6    2.1       33436  1259A,B    35223857        2902
1261A,B   16575932
**TOTALS     173    1403     18632   13.2     18115    0.9    8.1       90328             82303700
2972755             10148638
```

To examine the detail of the call processing for each program (itemized by type of call and summarized for the monitor interval), you can use the Call Summary report. An extract from the multipage output is given in the following example. The calls using an I/O PCB are given first and sub-totaled. Then, the total calls, of each type, against each database PCB and each external subsystem are listed. The PSB TOTAL line marks the end of data for each program.

```
     IMS MONITOR   ****CALL SUMMARY****               TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0186
                                                                (C)              (A)                      (B)
                  CALL  LEV         STAT                      IWAITS/       ..ELAPSED TIME...       .NOT IWAIT TIME..
DISTRIB.
PSB NAME PCB NAME FUNC  NO.SEGMENT  CODE   CALLS     IWAITS   CALL         MEAN      MAXIMUM       MEAN      MAXIMUM
NUMBER
--- ---- --- ---- ----  ----------  ----   -----     ------   ----         ----      -------       ----      -------
------
PROGSC6B I/O PCB  ISRT (  )                  138          0     0.00          372         1240          372
1240       598A,B,C
                  GU   (  )                  134        133     0.99      2600917     20974615      2587532
20962866       602A,B,C
                  (GU) (  )                    3          0     0.00           15           16           15
16       716A,B,C
                  ASRT (  )                    3          0     0.00          330          333          330
333       869A,B,C
                  GU   (  )         QC         2          1     0.50     17639806     21219588     17634776
21209529       870A,B,C
                  I/O PCB SUBTOTAL
                  ___ ___ ________           280        134     0.47      1370910                   1364469
         INVENTRB DLET (03)IN060SUP          138          0     0.00          813         1289          813
1289       599A,B,C
                  GNP  (03)IN060SUP          138          7     0.05         2112       112589         1047
112589       600A,B,C
                  GU   (01)IN010PAR          138        254     1.84        29511        75356         1195
19229       601A,B,C
                  DL/I PCB SUBTOTAL
                  ____ ___ ________          414        261     0.63        10812                      1018
                  PSB TOTAL
                  ___ _____                  694        395     0.56       559555                    551114
PROGSC2A I/O PCB  ISRT (  )                  118          0     0.00          381         1496          381
1496       603A,B,C
                  GU   (  )                  114        284     2.49      3304809     21784513      3164423
21664181       632A,B,C
                  (GU) (  )                    2          0     0.00           17           18           17
18       781A,B,C
                  ASRT (  )                    3          0     0.00          367          444          367
444       871A,B,C
                  GU   (  )         QC         2          5     2.50     19931897     20045206     19799530
19925277       872A,B,C
                  I/O PCB SUBTOTAL
                  ___ ___ ________           239        289     1.20      1743339                   1675270
         LOGVENDA REPL (03)IN040SLQ          118          0     0.00          268          804          268
804       604A,B,C
                  GNP  (03)IN040SLQ          118          5     0.04          899        16995          218
305       605A,B,C
                  REPL (02)VN030PAR          826          0     0.00          805         1578          805
1578       606A,B,C
                  GNP  (02)VN030PAR          826        873     1.05        19321        94521          456
1363       607A,B,C
                  REPL (01)VN020REO          118         58     0.49         8879        48076          832
1682       623A,B,C
                  GU   (01)VN020REO          118        195     1.65        31688       360775         1300
1746       625A,B,C
                  DL/I PCB SUBTOTAL
                  ____ ___ ________         2124       1131     0.53        10145                       636
                  PSB TOTAL
                  ___ _____                 2363       1420     0.60       185445                    170013
PROGSC2D I/O PCB  ISRT (  )                   14          0     0.00          377          621          377
621       608A,B,C
                  GU   (  )                   14         36     2.57     22360408     52048566     22221852
51901313       634A,B,C
                  I/O PCB SUBTOTAL
                  ___ ___ ________            28         36     1.28     11180393                  11111115
         LOGVENDD REPL (03)IN040SLQ           14          0     0.00          263          328          263
328       609A,B,C
                  GNP  (03)IN040SLQ           14          1     0.07         1407        16889          223
307       610A,B,C
                  REPL (02)VN030PAR           98          0     0.00          820         1015          820
1015       611A,B,C
```

Monitoring I/O for application program DL/I calls

The IMS Monitor report shows the total number of I/O occurrences and the total time the occurrences took for each UOR during a monitored interval.

The Program I/O report gives these two totals for all PSBs active during the monitored interval and includes the detailed breakdown of the I/O wait time as it was incurred by each PCB used by the program.

The report shows any contention experienced during application program processing. Each type of conflict and the number of times it occurred are recorded for each I/O PCB or database PCB. The report shows the total wait time, the highest wait experienced, and the average time. Subtotals are given for each PCB under a PSB, and for all PCBs under each PSB.

The DDN/FUNC column list the data set by DD name. The MODULE column uses a code to indicate the source of the contention. The types of conflicts and codes are shown as follows. Any codes that appear apply to IMS only.

- **Scheduling**

Code	**Conflict**
BLR	Load/read from ACBLIB
SMN	Virtual storage management

- **Database access**

Code	**Conflict**
DBH	OSAM I/O
DLE	DL/I functions
VBH	VSAM interface
(Physical segment code)	Program isolation

The I/O waits for the calls to the I/O PCB are grouped as the first entries for a PSB. For DL/I calls, the data set for which the I/O took place is indicated under the DDN/FUNC heading, and the module code tells you what type of conflict caused the wait.

Contention for the same physical segment in a database causes a wait on behalf of program isolation. This is shown in the DDN/FUNC column, on the PCB line, by the entry PIdmb, where dmb is the DMB of the physical data set. The MODULE column identifies the segment type using the physical segment code assigned by DBD generation.

When an application is accessing a database using VSAM as the access method, DL/I calls do not generally result in an I/O wait. A MODULE column entry of VBH indicates that interface to VSAM occurred and there was an I/O wait.

A seemingly unrelated entry can occur under the DDN/FUNC column for a database PCB. An example is a retrieval call to a database (DB-A) that causes a buffer to be purged in order to make room for that retrieved data. If the buffer contents included data belonging to another database (DB-B), the I/O entry in the report shows the DD name for DB-B as being in conflict for PCB access to DB-A.

Transaction Queuing report

The Transaction Queuing Report in the following example lists the transactions for DBCTL. Each transaction name is an 8-byte transaction ID specified by the CCTL on the schedule request or the CCTL ID.

A transaction ID from CICS, when used as the transaction manager, is composed of a 4-byte CICS transaction name plus a 4-byte CICS identifier. If the CCTL does not specify the transaction ID, DBCTL takes the CCTL region ID obtained at connection time as the default. In this report for DBCTL, the transaction NUMBER DEQUEUED is the number of schedules, and the ON QUEUE WHEN SCHEDULED is always zero, because the IMS message queues are not involved.

```
 IMS MONITOR   ****TRANSACTION QUEUING****   TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0181
                                                             (B)                    (A)
```

```
                   NUMBER      NUMBER      ..ON QUEUE  WHEN  SCHEDULED.....   DEQUED      DISTRIBUTION
TRANSACTION        DEQUED      SCHEDS.     MINIMUM     MEAN          MAXIMUM   MEAN         NUMBER
-----------        ------      ------      -------     ----          -------   ----         ------

 SC6X                13           1           0        0.00            0       13.00         883A,B
 IT8W                17           3           0        0.00            0        5.66         887A,B
 TS1Z                16           1           0        0.00            0       16.00         891A,B
 PS3X                23           1           0        0.00            0       23.00         895A,B
 PS3Y                17           7           0        0.00            0        2.42         899A,B
 IT1V                11           6           0        0.00            0        1.83         903A,B
 SC2Z               143           2           0        0.00            0       71.50         907A,B
 IT8U                12           7           0        0.00            0        1.71         911A,B
 PS2W                10           1           0        0.00            0       10.00         915A,B
 TS1U                12           4           0        0.00            0        3.00         919A,B
 PS3W                14           1           0        0.00            0       14.00         923A,B
 IT8Y                16           5           0        0.00            0        3.20         927A,B
 IT1W                 9           1           0        0.00            0        9.00         929A,B
 SC2W                 9           1           0        0.00            0        9.00         933A,B
 IT2V                13           5           0        0.00            0        2.60         937A,B
 IT2W                17           6           0        0.00            0        2.83         941A,B
 TS1V                 9           1           0        0.00            0        9.00         945A,B
 SC2V                12           5           0        0.00            0        2.40         947A,B
 TS1W                11           1           0        0.00            0       11.00         949A,B
 PS3V                13           3           0        0.00            0        4.33         953A,B
 IT1U                 9           6           0        0.00            0        1.50         957A,B
 SC4U                11           5           0        0.00            0        2.20         962A,B
 SC6W                10           1           0        0.00            0       10.00         966A,B
 PS2V                 8           6           0        0.00            0        1.33         970A,B
 IT8X                12           1           0        0.00            0       12.00         974A,B
 SC4W                10           1           0        0.00            0       10.00         978A,B
 SC6U                14           6           0        0.00            0        2.33         982A,B
 IT2Y                 9           3           0        0.00            0        3.00         986A,B
 SC2X                15           1           0        0.00            0       15.00         990A,B
 PS2Y                17           2           0        0.00            0        8.50         994A,B
 SC4Y               152           4           0        0.50            1       38.00         998A,B
 SC2Y               106           2           0        0.00            0       53.00        1000A,B
 IT2X                20           1           0        0.00            0       20.00        1004A,B
 SC2U                15           3           0        0.00            0        5.00        1008A,B
 SC4Z               123           5           0        0.60            1       24.60        1010A,B
 TS1X                15           1           0        0.00            0       15.00        1015A,B
 SC4X                19           1           0        0.00            0       19.00        1019A,B
 PS2X                13           1           0        0.00            0       13.00        1024A,B
 PS2Z                20           5           0        0.00            0        4.00        1028A,B
 SC6Z               130           1           0        0.00            0      130.00        1031A,B
 SC6V                10           4           0        0.00            0        2.50        1035A,B
 SC6Y               143           1           0        0.00            0      143.00        1037A,B
 IT1X                10           1           0        0.00            0       10.00        1040A,B
 PS3U                19           6           0        0.00            0        3.16        1131A,B
 IT2U                13           4           0        0.00            0        3.25        1146A,B
```

Monitoring database buffers

One of the key resources in an online system is the database buffer pool. The efficiency of DL/I call service depends on the presence of the required database logical record in the buffer, so that segment retrieval does not require additional I/O.

This is especially true for HOLD calls with intervening database calls prior to a replace call.

Use the **QUERY POOL** command to view usage statistics for Fast Path buffer pools.

You can assess the general efficiency of the full-function pool management using the Database Buffer Pool report shown in the following example. The event counts on this report are not specific to a particular database or program but represent the pressure for use of the database pool.

Related reading: Refer to "Database-Buffer-Pool report" on page 583 for more information on the Database Buffer Pool reports.

```
                    D A T A   B A S E   B U F F E R   P O O L

                                                                                 FIX PREFIX/
BUFFERS                   Y/Y
                                                                                 SUBPOOL
ID                          004K
                                                                                 SUBPOOL BUFFER
SIZE                 4096
                                                                                 TOTAL BUFFERS IN
```

```
SUBPOOL        1000


                                                                              16:09:59          16:25:10
                                                                            START TRACE        END
TRACE        DIFFERENCE

    NUMBER OF LOCATE-TYPE CALLS                                                 407636
4296793           3889157
    NUMBER OF REQUESTS TO CREATE NEW BLOCKS                                          1
7                 6
    NUMBER OF BUFFER ALTER CALLS                                                 75006
819359            744353
    NUMBER OF PURGE CALLS                                                         9137
93881             84744
    NUMBER OF LOCATE-TYPE CALLS, DATA ALREADY IN OSAM POOL                      313896
3317264           3003368
    NUMBER OF BUFFERS SEARCHED BY ALL LOCATE-TYPE CALLS                         453364
4779327           4325963
    NUMBER OF READ I/O REQUESTS                                                  86881
904487            817606
    NUMBER OF SINGLE BLOCK WRITES BY BUFFER STEAL ROUTINE                            0
0                 0
    NUMBER OF BLOCKS WRITTEN BY PURGE                                            32629
360434            327805
    NUMBER OF LOCATE CALLS WAITED DUE TO BUSY ID                                   281
3173              2892
    NUMBER OF LOCATE CALLS WAITED DUE TO BUFFER BUSY WRT                             6
180               174
    NUMBER OF LOCATE CALLS WAITED DUE TO BUFFER BUSY READ                            0
0                 0
    NUMBER OF BUFFER STEAL/PURGE WAITED FOR OWNERSHIP RLSE                          43
483               440
    NUMBER OF BUFFER STEAL REQUESTS WAITED FOR BUFFERS                               0
0                 0
    TOTAL NUMBER OF I/O ERRORS FOR THIS SUBPOOL                                      0
0                 0
    NUMBER OF BUFFERS LOCKED DUE TO WRITE ERRORS                                     0
0                 0

    QUOTIENT :  TOTAL NUMBER OF OSAM READS + OSAM WRITES =                     7.02
                ------------------------------------------------------
                              TOTAL NUMBER OF TRANSACTIONS
```

If any of your databases use VSAM as access method, the IMS Monitor produces a series of reports headed VSAM BUFFER POOL, one for each subpool. The following example shows one of these reports.

```
***I M S  M O N I T O R***  BUFFER POOL STATISTICS

                   V S A M   B U F F E R   P O O L
                                                                                                  FIX INDEX/BLOCK/
DATA          N/Y/N
                                                                                                  SHARED RESOURCE
POOL ID        VPL1
                                                                                                  SHARED RESOURCE
POOL TYPE         D
                                                                                                  SUBPOOL
ID                        2
                                                                                                  SUBPOOL BUFFER
SIZE            4096
                                                                                                  NUMBER
HIPERSPACE BUFFERS         0
                                                                                                  TOTAL BUFFERS IN
SUBPOOL          4

                                                                              16:09:59          16:25:10
                                                                            START TRACE        END
TRACE        DIFFERENCE

    NUMBER OF RETRIEVE BY RBA CALLS RECEIVED BY BUF HNDLR                          432
6029              5597
    NUMBER OF RETRIEVE BY KEY CALLS                                              40857
443840            402983
    NUMBER OF LOGICAL RECORDS INSERTED INTO ESDS                                   414
6011              5597
    NUMBER OF LOGICAL RECORDS INSERTED INTO KSDS                                  2132
25266             23134
    NUMBER OF LOGICAL RECORDS ALTERED IN THIS SUBPOOL                                0
0                 0
```

```
    NUMBER OF TIMES BACKGROUND WRITE FUNCTION INVOKED                           0
0                 0
    NUMBER OF SYNCHRONIZATION CALLS RECEIVED                                 6494
70963             64469
    NUMBER OF WRITE ERROR BUFFERS CURRENTLY IN THE SUBPOOL                      0
0                 0
    LARGEST NUMBER OF WRITE ERRORS IN THE SUBPOOL                               0
0                 0
    NUMBER OF VSAM GET CALLS ISSUED                                         44249
487181            442932
    NUMBER OF VSAM SCHBFR CALLS ISSUED                                          0
0                 0
    NUMBER OF TIMES CTRL INTERVAL REQUESTED ALREADY IN POOL                 11886
129668            117782
    NUMBER OF CRTL INTERVALS READ FROM EXTERNAL STORAGE                     32842
363635            330793
    NUMBER OF VSAM WRITES INITIATED BY IMS                                   2370
29208             26838
    NUMBER OF VSAM WRITES TO MAKE SPACE IN THE POOL                             0
0                 0
    NUMBER OF VSAM READS FROM HIPERSPACE BUFFERS                                0
0                 0
    NUMBER OF VSAM WRITES FROM HIPERSPACE BUFFERS                               0
0                 0
    NUMBER OF FAILED VSAM READS FROM HIPERSPACE BUFFERS                         0
0                 0
    NUMBER OF FAILED VSAM WRITES FROM HIPERSPACE BUFFERS                        0
0                 0

    QUOTIENT :  TOTAL NUMBER OF VSAM READS + VSAM WRITES =                  2.19
                ------------------------------------------------------
                              TOTAL NUMBER OF TRANSACTIONS
```

IMS internal resource usage

There are several summary reports that you can use to examine the level of internal contention for resources.

The following list provides a brief description of these reports.

Pool space failure report

The Pool Space Failure Summary report gives the number of times (in each region) a given amount of storage was unavailable. It shows the number of bytes, the identification of the pool, and the number of occurrences when storage was unavailable. You can use this summary to determine if you need to increase the buffer pool allocation, either by a system definition change or by overriding the number of buffers in the EXEC statements in the JCL.

The format of the report is shown in the following example.

```
POOL SPACE FAILURE SUMMARY

          POOL ID    BYTES REQ.    OCCURRENCES
          DLMP          8888            1
          DLDP          7777            1
     TOTAL                              2
```

Programs experiencing deadlock report

The Deadlock Event Summary report records each time a pair of programs reaches a deadlock over ownership of a segment in a given database data set. Each line in the report shows the two PSBs involved and indicates which is given processing right-of-way (REQ-ING PSB) and which has to reprocess after dynamic backout has occurred (LOSING PSB). The deadlock event summary report is illustrated in the following example.

```
DEADLOCK EVENT SUMMARY

     REQ-ING PSB    LOSING PSB    DMBNAME    OCCURRENCES
     PSBNAME1       TPPSBRE3      DBASEBAL       1
  TOTAL                                          1
```

IMS latch conflict report

The basic serialization of the task processing in IMS is controlled by ownership of an IMS latch. When different programs are executing, they compete for the ownership. If they wait for the resource, the one possessing the latch has to post the other ITASK waiting for it. Use the Latch Conflict Statistics report to judge the level of contention for a resource.

The different types of latches and the counters that exhibit the level of contention are given in the Latch Conflict Statistics report. The following example shows a sample latch conflict statistics report. The entries are organized according to the latch names.

When a system checkpoint is taken during the time the monitor is active, latch conflict statistics are reset to zero, thus corrupting the values presented in this report. If this situation exists, the following message will be inserted at the top of the report:

```
**** A CHECKPOINT OCCURRED DURING MONITOR RUN ****
****  LATCH CONFLICT STATISTICS ARE INVALID   ****
****      SEE UTILITIES REFERENCE MANUAL      ****
```

However, if the master terminal operator issues the **/CHECKPOINT** command with the STATISTICS keyword parameter, latch conflict statistics are reset to zero, but the IMS monitor is not notified. Therefore, DFSUTR20 cannot detect that the statistics have been corrupted and will not issue this message.

Recommendation: Do not issue statistics checkpoints while the monitor is running.

```
       IMS MONITOR   ** GENERAL REPORTS **         TRACE START 1993 209...

                 LATCH CONFLICT STATISTICS

LATCH       COUNT                AT           AT
NAMES       FIELD             START          END          DIFF.

LOGL     CONTENTIONS               0            0            0

SMGT     CONTENTIONS               0            0            0

XCNQ     CONTENTIONS               0            0            0

ACTL     CONTENTIONS               0            0            0


CBTS     CONTENTIONS               0            0            0

DBLK     CONTENTIONS               0            0            0
```

Using frequency distributions from IMS Monitor output

The reports derived from the IMS Monitor data records contain many summary lines where the mean time is given.

If you are interested in the distribution of those timed events, rather than just average and maximum times, you can request the Report Print utility to individually record the events in a frequency distribution across a range of intervals. Some distributions are not time dependent, such as those for transaction queue loads or transmitted block sizes.

- The following table shows the report distributions sorted by Region Summary.

Table 66. Report distributions by region summary

Report name	ID	Description
Scheduling and Termination	D1	Elapsed time
	D2	Not wait time
Schedule end to 1st DL/I call	D3	N/A
Elapsed execution time	D4	N/A

Table 66. Report distributions by region summary (continued)

Report name	ID	Description
DL/I calls	D5	Elapsed time
	D6	Not wait time
External Subsystem calls	D43	Elapsed time
Waits per DL/I call	D7	N/A
Idle for intent	D8	N/A
Checkpoint	D20	Elapsed time
	D21	Not wait time

- The following table shows the report distributions sorted by Programs Region.

Table 67. Report distributions by programs region

Report name	ID	Description
Elapsed execution time	D30	N/A
Schedule and to 1st DL/I call	D31	N/A

- The following table shows the report distributions by Program Summary.

Table 68. Report distributions by program summary

Report name	ID	Description
Processor time per schedule	D15	N/A
Transactions dequeued per schedule	D14	N/A
Elapsed time per schedule	D9	N/A
Schedule end to 1st DL/I call	D10	N/A

- The following table shows the report distributions sorted by Call Summary.

Table 69. Report distributions by call summary

Report name	ID	Description
PSB waits per DL/I call	D13	N/A
PSB waits per external subsystem call	D44	N/A
PSB elapsed time per call	D11	N/A
PSB not wait time per call	D12	N/A
PSB external subsystem calls	D45	N/A
	N/A	Elapsed time

- The following table lists some distributions derived from buffer pool statistics for wait times.

Table 70. Wait time distributions

Function	ID	Module key
Storage	D22	SMN
OSAM I/O	D23	DBH
VSAM I/O	D24	VBH

Table 70. Wait time distributions (continued)

Function	ID	Module key
Block loader I/O	D27	BLR
HSAM I/O	D34	DIE
PI enqueue	D40	None

How to get a frequency distribution output

To request the IMS Monitor Report Print utility to gather distribution data, you must include a DIS input control statement. This causes all report items with an entry under a column headed MEAN to have a corresponding frequency distribution as part of the Distribution Appendix. Also, each report line includes an identifying reference number under the column headed DISTRIBUTION NUMBER so that you can locate the distribution data in the appendix, flagged by that same number.

The following tables show the major IMS Monitor reports and the type of frequency distributions generated for each report. Each type results in several distributions, depending on how many entries are in each section of the report. For each type of frequency distribution, the data is cumulated in suitable intervals or ranges. The set of ranges used for each type is given an identifier, shown in the ID column.

How frequency distribution ranges are defined

A set of ten intervals is defined for each summary line and the occurrences falling in each interval are cumulated. The interval ranges are preset with default end points. For example, the end points, for DL/I call elapsed time are: 0, 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000, 256000, INF (all times are in milliseconds). The default end points are chosen so that they are suitable to the event. The lower limit of the first interval always defaults to zero, and the upper limit of the tenth interval is infinity (INF).

Although several types of distribution can use the same set of end points, each type is assigned a distribution identifier. You can use this to redefine the end points. To override the default end points include an input control statement to the Report Print utility. The statement specifies the type of distribution identifier and gives the desired end point values. For example, the DL/I call elapsed time end points could be respecified by:

```
D5   0,500,1000,1500,2000,4000,,,100000,500000
```

The values of the unspecified end points remain at their default values of 32000 and 64000 as does the last (INF).

The following example, which is a sample page taken from a Distribution Appendix, shows how individual distributions are numbered and how ranges vary with the type of distribution. The lines are arranged in pairs, with the second one recording the cumulated counts.

```
     IMS MONITOR   ****DISTRIBUTION APPENDIX****       TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0200

#
1..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                     0          0          0          0          0          0          0          0
0          1
#
2..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                     0          0          0          0          0          0          0          0
0          1
#
3..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                     0          0          0          0          0          0          0          0
0          1
#
4..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                     0          0          0          0          0          0          0          0
0          1
#
5..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                     0          0          0          0          0          0          0          0
0          6
#
```

```
6..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          8
#
7..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          1
#
8..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          9
#
9..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          1
#
10..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      1          0          0          0          0          0          0          0
0          7
#
11..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          1
#
12..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          8
#
13..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          1
#
14..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          1
#
15..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      1          0          0          0          0          0          0          0
0          8
#
16..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....1800000....INF
                      0          0          0          0          0          0          0          0
0          1
```

Default values of distribution definitions

Using an identifier provided in the frequency distribution tables and the Wait Time Distributions table, you can determine the default end points for the distribution by locating it in the following list:

D1, D2, D5, D6, D9, D10, D11, D12, D15 D18, D19, D20, D21, D22, D25, D27 D28, D29, D30, D31, D43, and D45
: 0, 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000, 256000, INF

D3
: 0, 50000, 100000, 150000, 200000, 250000, 300000, 350000, 400000, 450000, INF

D4
: 0, 200000, 400000, 600000, 800000, 1000000, 1200000, 1400000, 1600000, 1800000, INF

D7, D13, and D44
: 0, 0, 1, 2, 3, 4, 5, 6, 7, 8, INF

D8
: 0, 100000, 200000, 300000, 400000, 500000, 600000, 700000, 800000, 900000, INF

D14, D16, D17
: 0, 1, 2, 3, 4, 5, 10, 15, 30, 90, INF

D23, D24, D26, D32, D40, D42
: 0, 2000, 8000, 24000, 50000, 100000, 150000, 200000, 250000, 300000, INF

D33, D34, D35
: 0, 2000, 4000, 8000, 16000, 32000, 64000, 96000, 128000, 160000, INF

D36, D37
: 0, 10, 20, 40, 80, 100, 200, 400, 800, 1000, INF

D38
: 0, 1000, 10000, 100000, 200000, 500000, 800000, 1000000, 1500000, 2000000, INF

D39
0, 1000, 5000, 10000, 50000, 100000, 500000, 1000000, 5000000, 10000000, INF

Interpreting Distribution Appendix output

You can use the detailed output in the Distribution Appendix when you suspect an unusual combination of events was reported in a report summary line.

Usually, the average and maximum times or counts are sufficient to highlight a resource usage problem. However, if you suspect the mean value to be masking an unusual distribution you can draw on the detail contained in the IMS Monitor output records.

Chapter 50. IMS Monitor reports for DCCTL

DCCTL is a transaction management subsystem that has no database components. With the external subsystem (ESS) attach facility, it provides the transaction management capability for non-IMS database subsystems.

This topic describes:

- The events that the IMS Monitor collects
- The content of the reports produced by the IMS Monitor Report Print Program in a DCCTL environment

DCCTL does not change the format or usage of the IMS Monitor reports. There are reports, and fields within reports, that contain information specific to databases, and these are not applicable to the DCCTL environment. Reports that do not apply to DCCTL appear as a heading without data, or are not produced. The reports that do not apply to DCCTL include:

- Database Buffer Pool report
- VSAM Buffer Pool (DB) report
- Call Summary (DB) report
- Program I/O (DB) report

For a detailed look at the events, system activities, and use of storage areas (buffer pool or data set) for which timings are gathered by the IMS Monitor, see "Transaction flow and IMS Monitor events" on page 595.

IMS Monitor trace event intervals

The IMS Monitor trace interval is defined by the master terminal operator's use of the **/TRACE** command between the start and stop command entries. The online IMS events are recorded in IMS Monitor records placed in the IMSMON data set.

The event timings are related to dependent region activity. The following figure shows the boundaries of the timed event intervals.

The Monitor trace interval includes the following intervals:

- Scheduling and Termination
  - Block loader busy
  - Schedule failures (PSB busy and space failure)
  - Sched/Term elapsed
    - NOT-WAIT
    - ACBLIB waits
- Region occupancy (which overlaps with all of Sched/Term elapsed)
  - Schedule to first call
  - Elapsed execution

The NOT-WAIT time for a region is the elapsed time not accounted for by wait time. Any delay coming from either paging or the processor being dispatched for a higher priority task results in an increase in the `NOT-WAIT` times.

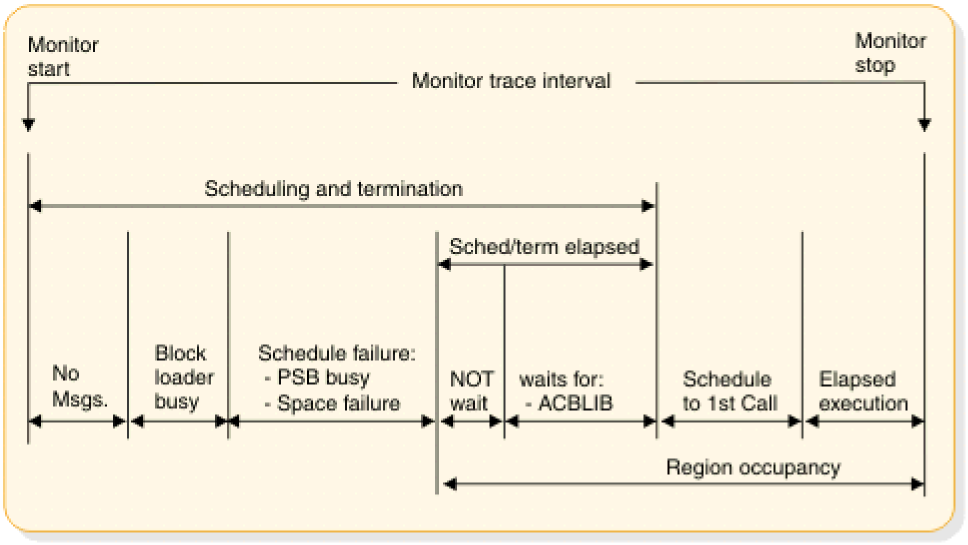


Figure 71. IMS Monitor trace event intervals

Overview of IMS Monitor reports

A list of reports available from data collected by the IMS Monitor, together with the principal performance data they contain, is shown in the following table.

The reports marked "MSC" in the list are only produced when MSC is active.

The order of the reports listed in the table matches the sequence of the output from the IMS Monitor Report Print Program. The duration of a monitoring snapshot might not include certain events necessary for an individual report, in which case only report headings or partial data are produced.

Table 71. Output sequence and information from IMS Monitor reports

Report name	Principal information
System Configuration	Monitor run documentation
Message Queue Pool	Buffering and message I/O per transaction
Message Format Buffer Pool	Count of I/Os
Latch Conflict Statistics	IMS internal processing
General Wait Time Events	Wait times for SNAPQ
Region and Jobname	Monitor run documentation
Region Summary	Elapsed times and count of DL/I calls (DC)
Region Wait	Wait times
Programs by Region	Elapsed times for region usage
Program Summary	Overall program statistics
Communication Summary	Elapsed times for lines
Communication Wait	Wait times by line
Line Functions	Count and size of blocks transmitted
MSC Traffic (MSC)	Count and routing of transactions
MSC Summaries (MSC)	Count of transactions by destination
MSC Queuing Summary (MSC)	Count and queuing time by link

Table 71. Output sequence and information from IMS Monitor reports (continued)

Report name	Principal information
Transaction Queuing	Queue loading statistics
Reports	Count of space failures and deadlocks
Run Profile	Monitor run documentation
Distribution Appendix	Event frequency distributions

The majority of the data items in IMS Monitor reports are elapsed times. These are normally expressed in microseconds. An entry of 1876534 is 1.876534 seconds or 1876 milliseconds. Any times that do not follow this convention show the unit of measure on the report.

You can also find counts of events under the heading OCCURRENCES, and figures that represent the number of bytes.

Related concepts

"IMS Monitor reports for MSC" on page 627
The IMS Monitor Report Print program includes three reports that highlight message events caused by Multiple Systems Coupling.

Documenting the monitoring run

For each trace interval there are several general reports or overall summaries of the processing that took place. You can use these reports as part of your IMS Monitor run documentation.

It is important to record as accurately as possible the conditions under which the trace was taken. Your documentation can include system status information obtained by the **/DISPLAY** command several times before and after the trace, an expected profile of the application program activity, and any desired processing events. The trace interval should represent typical processing loads and not be a biased or inadequate historical record.

Adding to the System Configuration report data

The first general report titled SYSTEM CONFIGURATION is found under the page heading BUFFER POOL STATISTICS. It shows the modification level of the IMS and z/OS systems. You can add a list of IMS APARs applied and include the service levels of the application programs, especially if the latter are not permanent programs or are part of a staged implementation. The system configuration output is illustrated in the following section.

Recording the Monitor trace interval

The heading of most IMS Monitor reports carries the trace start and stop times. It is shown in the format YEAR DAY (Julian) HH:MM:SS. The overall length of the trace interval is given in seconds under the heading TRACE ELAPSED TIME IN SECONDS. The following line shows how many trace records were placed on the IMS.MON data set. An example of the monitor trace interval recording is shown in the following example.

```
***IMS MONITOR***  BUFFER POOL STATISTICS   TRACE START 1993 130   5:55:15   TRACE STOP  1993 130
5:59:49  PAGE 0001

                   S Y S T E M   C O N F I G U R A T I O N



                                                 SYSTEM CONFIGURATION      :
                                                    IMS VERSION            :   4
                                                 RELEASE LEVEL             :
                                                 MODIFICATION NUMBER       :
```

Completing the Monitor run profile

A compact set of processing ratios is found at the end of the Run Profile report. The statistics summarize, for the monitor interval, the transaction throughput and the degree of DL/I and I/O activity. An example of the report is shown in the following example. In a DCCTL environment, DL/I activity is restricted to data communications calls and calls to GSAM databases. Database calls to other types of DL/I databases are not supported in DCCTL.

The lower part of the Run Profile report shows several ratios:

- Program elapsed time to DL/I elapsed time for each region
- DL/I elapsed time to wait time during DL/I processing
- Program elapsed time to other subsystem call elapsed time
- DL/I elapsed time to other subsystem call elapsed time

Each dependent region is identified by a sequence number, starting at region 1.

```
   IMS MONITOR   **RUN PROFILE**              TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0184
TRACE ELAPSED TIME IN SECONDS.......274.6
TOTAL NUMBER OF MESSAGES DEQUEUED....1403
TOTAL NUMBER OF SCHEDULES        .....173
NUMBER OF TRANSACTIONS PER SECOND    5.1
TOTAL NUMBER OF DL/I CALLS ISSUED....18632
NUMBER OF DL/I CALLS PER TRANSACTION   13.2
NUMBER OF OSAM BUFFER POOL I/O'S.           0,    0.0 PER TRANSACTION
NUMBER OF MESSAGE QUEUE POOL I/O'S..........0,    0.0 PER TRANSACTION
NUMBER OF FORMAT BUFFER POOL I/O'S..........0,    0.0 PER TRANSACTION
RATIO OF PROGRAM ELAPSED TO DL/I ELAPSED:
              REGION   1:   1.09
              REGION   2:   1.09
              REGION   3:   1.00
              REGION   4:   1.02
              REGION   5:   1.01
              REGION   6:   1.00
              REGION   7:   1.00
              REGION   8:   1.00
              REGION   9:   1.17
              REGION  10:   1.00
              REGION  11:   1.00
              REGION  49:   1.03
              REGION  50:   1.19
RATIO OF DL/I ELAPSED TO DL/I IWAIT:
              REGION   1: 325.65
              REGION   2:  73.49
              REGION   4: 100.35
              REGION   5:  85.76
              REGION   6:  82.99
              REGION  47:  95.64
              REGION  48:  45.93
              REGION  49:   9.22
```

You can match the regions to the z/OS job name using the Region and Jobname report. The job names correspond to the step names on the EXEC statements of all the dependent regions started by the operator before the trace was started. The region job names are included on the monitor output page with the heading GENERAL REPORTS, as illustrated in "Detecting checkpoint effects" on page 669.

There are some generalized processing ratios that are given at the end of several buffer pool statistics reports. You can include them in the documented profile of the trace interval. These are not specific to one application or system resource but can be used as indicators of variation across a series of monitor runs. In DCCTL, the ratios are:

- All waits divided by the total number of transactions

  This value can be found on the Message Queue Pool Report in "Monitoring message queue handling" on page 668. This ratio indicates on a per transaction basis the physical I/O activity required to handle the message queuing function.

- The total prefetch I/Os + immediate fetch I/Os + directory I/Os divided by the total number of transactions

This value also appears on the example shown in "Monitoring MFS activity" on page 667. This ratio indicates on a per transaction basis the physical I/O activity required to handle the MFS function during the trace period.

Verifying IMS Monitor report occurrences

When you examine the output from the IMS Monitor Report Print program, do not assume that the presence of a report heading implies that appropriate data is listed. System definition options and utility control statements affect the content of the output as follows:

- The output does not include a Call Summary report unless a control statement specifies DLI.
- The output does not include a set of Distribution reports unless a control statement specifies DIS or DISTRIBUTION. The column headed DISTRIBUTION NUMBER that occurs on many of the reports contains cross-references to items included in the Distribution reports.
- The output consists of just a Call Summary report if a control statement specifies ONLY DLI.

Because many of the summary reports require system status to calculate the difference between start and end values, and this status is obtained during /TRACE SET OFF processing, the IMS Monitor execution must end before termination of the IMS control region. If the trace was not stopped properly, the following message is issued:

```
NO QUEUE BUFFER POOL TRACES AT END TIME ON MONITOR LOG TAPE
****QUEUE BUFFER POOL REPORT CANCELLED****
```

Similarly, other summary reports are not produced.

The section MESSAGE FORMAT BUFFER POOL is included only if your system definition specifies devices using Message Format Service (MFS).

If the source data used to formulate a particular IMS Monitor report, or a section of that report, has not been recorded by the IMS Monitor during the trace interval, the report contains only the headings.

Monitoring activity in dependent regions

The IMS Monitor gathers timing information for every dependent region identified in the **`/trace`** command active during the trace interval. It records the total of the elapsed times for each event, the time for the longest event encountered, and the average time for all recorded events.

There are three major reports that display timings. The reports and a list of their content are:

Region Summary Report

- Scheduling and termination
- Schedule end to first call
- Elapsed execution with separate summaries shown for:
  - DL/I calls
  - External subsystem service and command calls
  - External subsystem database access calls
  - Checkpoint processing
  - Region occupancy

Region Wait

- Waits during scheduling and termination
- Waits during DL/I calls
- Waits during external subsystem calls
- Waits during checkpoint

Programs by Region

- Elapsed execution
- Schedule end to first call

These three reports are illustrated in the following examples.

Activities in dependent regions are placed in five timing categories:

- Elapsed time for scheduling and termination

  The scheduling process includes many preparatory events such as block loading from an active IMS.ACBLIBA/B data set and obtaining ownership of the PSB. The time required to terminate the region activity after the application program ends is also included.
- Elapsed time from end of schedule to first call

  This time is reserved for application program initialization and housekeeping prior to an initial call (to the message queue, or an external subsystem) that marks the beginning of control program services. It is a measure of processing that is not repeated when multiple transactions are processed in a single scheduling.
- Program elapsed time, including all calls

  This time encompasses the major application program processing and is measured from the first call to the return to or exit from the program.
- Elapsed time performing DL/I calls

  This time includes all DL/I calls. Each DL/I call event is measured from the time of the call to the return to the application program.
- Elapsed time performing external subsystem calls

  This time includes all external subsystem calls. Each external subsystem event is measured from the time of the call to the return to IMS.

The following example shows a sample of the region summary report.

```
    IMS MONITOR   ****REGION SUMMARY****              TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0011
                                              (A)                                  (B)
                                 ........ELAPSED TIME.........       NOT IWAIT TIME(ELAPSED-IWAIT)
DISTRIBUTION
              OCCURRENCES       TOTAL        MEAN     MAXIMUM       TOTAL        MEAN     MAXIMUM
NUMBER
              -----------       -----        ----     -------       -----        ----     -------
______
SCHEDULING AND TERMINATION
__________ ___ ___________
**REGION    5           5        4146         829         948        4146         829         948
287A,B

**REGION     6           7        6028         861        1067        6028         861        1067
214A,B
**REGION     8           8        6847         855        1098        6847         855        1098
129A,B
**REGION    10           7        9664        1380        3668        9664        1380        3668
272A,B
**REGION    47           6        5482         913        1021        5482         913        1021
145A,B
**REGION    49           3        2612         870         917        2612         870         917
443A,B
**TOTALS               123      126042        1024                   126042        1024
SCHEDULE TO FIRST CALL
________ __ _____ ____
**REGION     1           1    15479797    15479797    15479797
555
**REGION     2           1    22376350    22376350    22376350
564
**REGION     3           1    15169488    15169488    15169488
578
**REGION     4           1    48146258    48146258    48146258
584
**REGION    48           1      795351      795351      795351
592
**REGION    49           4     2960425      740106     2951746
442
**REGION    50           1    15713464    15713464    15713464
575
**TOTALS               168   514286738     3061230
```

```
ELAPSED EXECUTION
_______ _________
**REGION     1           1   290146255   290146255
290146255                                                           1
**REGION     2           1   252290108   252290108
252290108                                                           2
**REGION     3           1   259496970   259496970
259496970                                                           3
**REGION     4           1   322812716   322812716
322812716                                                           4
**REGION    48           1   273871107   273871107   273871107
48
**REGION    49           4   271703421    67925855   155176058
49
**REGION    50           1   290379922   290379922   290379922
50
**TOTALS               173 14238540145    82303700
DL/I CALLS                                                                                          IWT/CALL(C)
____ _____                                                                                          ___________
**REGION     1          60   264626241     4410437    88981490   263813671     4396894    88970053       0.76
247A,B,C
**REGION     2         223   230505269     1033655    61048758   227368742     1019590    61011153       0.73
237A,B,C
**REGION     3          29   257704383     8886358    69000514   257704383     8886358    69000514       0.00
98A,B,C
**REGION     4         792   313735347      396130    52439653   310609035      392183    52439653       0.22
180A,B,C
**REGION    49         592   262886317      444064    30202068   234394017      395935    30159782       2.46
177A,B,C
**REGION    50          36   242591451     6738651    48651260   242591451     6738651    48651260       0.00
289A,B,C
**TOTALS             18632 12386905286      664818             12024562411      645371                   0.97
IDLE FOR INTENT
____ ___ ______
                      NONE
CHECKPOINT
__________            NONE
REGION OCCUPANCY
______ _________
**REGION     1    100.0%
**REGION     2    100.0%
**REGION     3    100.0%
**REGION     4    100.0%
**REGION    48    100.0%
**REGION    49    100.0%
**REGION    50    100.0%
```

The following example shows a sample of the region wait report.

```
   IMS MONITOR   ****REGION IWAIT****         TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0023
                                 .........IWAIT TIME..........                           DISTRIBUTION
**REGION     5 OCCURRENCES       TOTAL        MEAN     MAXIMUM       FUNCTION  MODULE       NUMBER
  ______     ___________         _____        ____     _______       ________  ______       ______
SCHEDULING + TERMINATION
__________   ___________
   SUB-TOTAL
   _________
   TOTAL
   _____
DL/I CALLS
____ _____
                        11      181816       16528       24375    DD=IMMSTR2A   DBH            117
                         8      112831       14103       17846    DD=IMMSTR1A   DBH            118
                         5       85460       17092       33717    DD=IMMSTR3A   DBH            119
                         1        1633        1633        1633    INT=BMPFPE05  BLR-64BIT       14
                         1        4242        4242        4242    PSB=DDLTRN17  BLR-64BIT       20
                         5       58420       11684       14643    DD=IMINDEXA   VBH            120
                        12      173866       14488       22152    DD=PRODCNTA   VBH            121
                         3      100576       33525       68373    DD=IMMSTR2B   DBH            428
                         1       17921       17921       17921    DD=IMMSTR3B   DBH            429
                         1       17195       17195       17195    DD=IMMSTR1B   DBH            430
                         1       13577       13577       13577    DD=IMINDEXB   VBH            431
                         3       49928       16642       20396    DD=PRODCNTB   VBH            432
                         4       10973        2743        2787    DD=ITEMACTB   DBH            453
                         2       37680       18840       27664    DD=IAINDEXB   VBH            454
                        49     1500067       30613      138284    DD=INVENTRA   DBH            472
                        23      345595       15025       27613    DD=VENDORDA   VBH            473
                         1      342952      342952      342952    PI=VENDORDA...1              498
                         1       14612       14612       14612    PI=VNSINDXA...1              499
                         6       69203       11533       19492    DD=VNSINDXA   VBH            500
   TOTAL
   _____               136     3132672       23034
```

The following example shows a sample of the programs-by-region report.

```
   IMS MONITOR   ****PROGRAMS BY REGION****     TRACE START 1993 130   5:55:15     TRACE STOP  1993 
130   5:59:49  PAGE 0069
                                             (A)                                    (B)
                                  ELAPSED EXECUTION TIME           SCHEDULING END TO FIRST CALL 
DISTRIBUTION
               OCCURRENCES       TOTAL        MEAN     MAXIMUM       TOTAL        MEAN     MAXIMUM 
NUMBER
               -----------       -----        ----     -------       -----        ----     -------
______
**REGION     1
________   _ 1
    PROGSC6D             1   290146255   290146255   290146255    15479797    15479797 
15479797         885A,B
 REGION TOTALS           1   290146255   290146255                15479797    15479797
**REGION     2
________   _ 2
    PROGIT8C             1   252290108   252290108   252290108    22376350    22376350 
22376350         889A,B
 REGION TOTALS           1   252290108   252290108                22376350    22376350
**REGION     3
________   _ 3
    PROGTS1C             1   259496970   259496970   259496970    15169488    15169488 
15169488         893A,B
 REGION TOTALS           1   259496970   259496970                15169488    15169488
**REGION     4
________   _ 4
    PROGPS3D             1   322812716   322812716   322812716    48146258    48146258 
48146258         897A,B
 REGION TOTALS           1   322812716   322812716                48146258    48146258
**REGION     5
________   _ 5
    PROGPS3A             2    62893103    31446551    40693590        5435        2717 
2862         901A,B
    PROGTS1B             1    61794787    61794787    61794787        2790        2790 
2790        1271A,B
    PROGPS3B             1    18294458    18294458    18294458        3104        3104 
3104        1350A,B
    PROGIT2B             1    36095342    36095342    36095342        2731        2731 
2731        1363A,B
    PROGSC2A             1    93902771    93902771    93902771     1667791     1667791 
1667791        1401A,B
 REGION TOTALS           6   272980461    45496743                 1681851      280308
**REGION     6
________   _ 6
    PROGIT1B             2    39000315    19500157    23703429        5286        2643 
2801         905A,B
    PROGTS1B             1    34293636    34293636    34293636        3136        3136 
3136        1207A,B
    PROGPS3A             1    51887767    51887767    51887767        2534        2534 
2534        1278A,B
    PROGPS3B             2    67375031    33687515    40291430    17210570     8605285 
17213287        1328A,B
    PROGIT8A             1    69132416    69132416    69132416        3291        3291 
3291        1359A,B
    PROGSC4A             1    30165017    30165017    30165017        2571        2571 
2571        1433A,B
 REGION TOTALS           8   291854182    36481772                17193752     2149219
**REGION     7
________   _ 7
    PROGSC2B             1   269618583   269618583   269618583     5047875     5047875 
5047875         909A,B
 REGION TOTALS           1   269618583   269618583                 5047875     5047875
**REGION     8
________   _ 8
    PROGIT8A             1     5181039     5181039     5181039        2928        2928 
2928         913A,B
    PROGPS3A             1    27304257    27304257    27304257        3350        3350 
3350        1132A,B
    PROGSC4B             1    37286872    37286872    37286872        3009        3009 
3009        1255A,B
    PROGIT2A             1    36902995    36902995    36902995        2850        2850 
2850        1298A,B
    PROGIT1B             1    30407479    30407479    30407479        2565        2565 
2565        1336A,B
    PROGIT1A             3   109875360    36625120    45190114     4279008     1426336 
4272096        1357A,B
    PROGIT8B             1    23405220    23405220    23405220        2679        2679
```

```
2679         1395A,B
 REGION TOTALS          9   270363222    30040358                 4296389      477376
```

Examining the effects of checkpoints

Checkpoint processing can be initiated by the control program at a specified frequency determined by the number of records placed on the system log. Other checkpoints can be caused by operator commands.

The checkpoint line of the Region Summary report at the end of the region 0 summary shows the following:

- The number of system checkpoint taken during the monitor interval
- The elapsed times
- The not-wait times

The wait time experienced during checkpoints is reported at the end of the first region summary on the Region Wait report. You can detect delays for each combination of DD name and module code. Typical entries here are for the message queue data sets and the restart data set. If an wait for storage is the cause, the entry under the FUNCTION column is STG.=, followed by the identification of the pool.

Measuring region occupancy

A measure of region activity is the percentage of region occupancy. This is broadly the ratio of the elapsed time a region is performing processing to the trace interval.

The region occupancy time does not include those times when no messages are available, when the block loading is delayed, or when the PSB cannot be used. The last section in the Region Summary report lists all active regions for which timed events were collected and shows the calculated percentage region occupancies.

Monitoring application program elapsed time

The IMS Monitor can record measurements of elapsed times for each transaction and scheduling of an application program. It does this during the monitored interval while other programs are executing concurrently.

Elapsed times are calculated from the start of the first DL/I (or other) call to the end of that program. You can distinguish between time spent in application code and in DL/I processing. The event intervals are illustrated in the following figure.

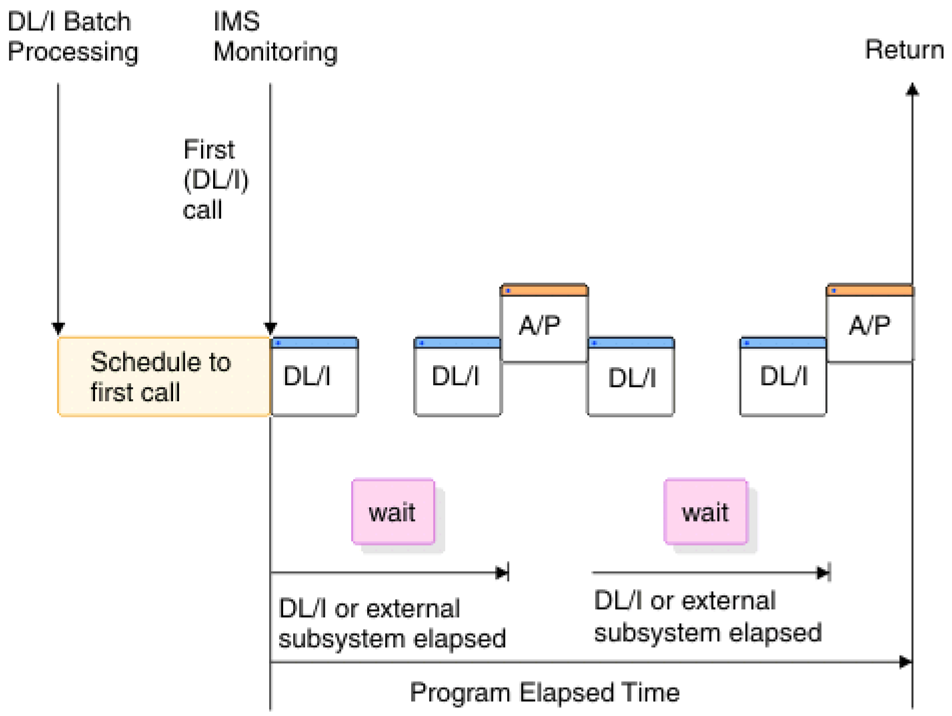


Figure 72. Elapsed Time event intervals

Within the elapsed time for a DL/I call, the wait time to obtain segment data is recorded separately. Similarly, within the elapsed time for an external subsystem call, the processing time in the external subsystem is recorded separately as the wait time. The application processing (A/P) time includes many kinds of subsidiary service beyond the machine cycles expended by the program object code—such as subroutine loading, I/O to z/OS data sets, and any overlay processing. If the program is waiting to be dispatched or requires paging before it can use the real storage, these delays are also accounted for in application program processing. Because a program can execute many transactions for each schedule, the elapsed time from schedule to first call is recorded separately. This time covers the initialization performed by the application program and also includes the time for loading the program.

The elapsed times are given in the Program Summary report. The following example is a sample of the program summary report. Programs are identified by their PSB name on individual lines in the report. Each line gives a summary of the activity for that PSB during the measured interval. The total number of schedules, DL/I calls, transactions completed (dequeued), and waits for DL/I call I/O calls, the and external subsystem processing are given. The report line gives calculated average times for:

- Elapsed time per schedule
- Processor time per schedule
- Schedule to first DL/I call per schedule
- Elapsed time per transaction

Frequencies for calls per transaction, I/O waits per DL/I call, waits per external subsystem call, and transactions dequeued per schedule are also given. A TOTALS line summarizes all activity for the PSBs active during the monitored interval. (The PSB DUMMY line reconciles any incomplete scheduling caused by a region stopping during scheduling or for a program that experiences a pseudo abend.)

```
     IMS MONITOR   ****PROGRAM SUMMARY****            TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0075
                                                                 (A)........(B)........          (A)........(B)........
                                                       I/O   TRAN.       CPU        .          ELAPSED
SCHED.TO     .       ELAPSED
            NO.    TRANS.            CALLS     I/O   IWAITS  DEQD.      TIME    DISTR.       TIME      1ST CALL
DISTR.       TIME
PSBNAME  SCHEDS.     DEQ.     CALLS  /TRAN    IWAITS  /CALL  /SCH.     /SCHED.    NO.       /SCHED.     /SCHED.
NO.       /TRANS.
-------  -------     ----     -----  -----    ------  -----  -----     -------    ---       -------     -------
---       -------
```

```
PROGSC6D       1      13         60    4.6         46    0.7   13.0       10010   884A,B   290146255    15479797
886A,B    22318942
PROGIT8C       3      17        225   13.2        166    0.7    5.6       90592   888A,B   256617508    73283259
890A,B    45285442
PROGTS1C       2      25         47    1.8          0    0.0   12.5       10010   892A,B   239190808     7586234
894A,B    19135264
PROGPS3D       1      23        792   34.4        182    0.2   23.0       10010   896A,B   322812716    48146258
898A,B    14035335
PROGPS3A      13      36       1246   34.6        267    0.2    2.7       49782   900A,B    32801812     2228611
902A,B    11845098
PROGIT1B      11      21         99    4.7          0    0.0    1.9        6341   904A,B    23212388     2036217
906A,B    12158870
PROGSC2B       7     155       3068   19.7       1845    0.6   22.1      346112   908A,B    93655514      789390
910A,B     4229603
PROGIT8A      12      28        434   15.5        293    0.6    2.3       34350   912A,B    30196795     1745815
914A,B    12941483
PROGPS2C       1      10        179   17.9        205    1.1   10.0       10010   916A,B   221024429    53642029
918A,B    22102442
PROGTS1B       8      20         54    2.7          0    0.0    2.5        5447   920A,B    39943245        2895
922A,B    15977298
PROGPS3C       1      14        468   33.4        117    0.2   14.0       10010   924A,B   310644485    35978027
926A,B    22188891
PROGIT1C       1       9         32    3.5          0    0.0    9.0       10010   930A,B   304892631    30226173
932A,B    33876959
PROGSC2C       1       9        160   17.7        101    0.6    9.0       10010   934A,B   296909110    22242652
936A,B    32989901
PROGIT2B       8      21        393   18.7         63    0.1    2.6       21703   938A,B    35126671     1798496
940A,B    13381589
PROGIT2C       6      17        211   12.4         39    0.1    2.8       13312   942A,B   288883508    50698467
944A,B   101958885
PROGTS1D       2      26         50    1.9          0    0.0   13.0       10010   950A,B   284944505    10613350
952A,B    21918808
PROGPS3B       8      22        770   35.0        169    0.2    2.7       35737   954A,B    38016279     2149158
956A,B    13824101
PROGIT1A      11      24        106    4.4          0    0.0    2.1        7925   958A,B    30883486     1935855
960A,B    14154931
PROGSC4A       9     163       1775   10.8       5101    2.8   18.1      235921   963A,B    62172947     3011199
965A,B     3432862
PROGSC6C       1      10         44    4.4         38    0.8   10.0       10010   967A,B   228098334    46568124
969A,B    22809833
PROGPS2B      11      28        557   19.8        604    1.0    2.5       35069   971A,B    33309266     1181831
973A,B    13085783
PROGIT8D       1      12        175   14.5        133    0.7   12.0       10010   975A,B   253392289    21274169
977A,B    21116024
PROGSC4C       1      10         98    9.8        349    3.5   10.0       10010   979A,B   248736332    25930126
981A,B    24873633
PROGSC6A       7     157        789    5.0        457    0.5   22.4       11703   983A,B    73936039      115979
985A,B     3296511
PROGIT2A       7      22        430   19.5         71    0.1    3.1       28529   987A,B    37905001        2982
989A,B    12060682
PROGSC2D       1      15        280   18.6        180    0.6   15.0       10010   991A,B   316194222    41527764
993A,B    21079614
PROGPS2A       6      25        490   19.6        548    1.1    4.1       43177   995A,B    58277945     2467506
997A,B    13986707
PROGSC2A       5     121       2363   19.5       1420    0.6   24.2      276187  1001A,B    88906184     6022954
1003A,B     3673809
PROGIT2D       1      20        361   18.0         62    0.1   20.0       10010  1005A,B   386092737   111426279
1007A,B    19304636
PROGSC4B      10     131       1421   10.8       4115    2.8   13.1      617016  1011A,B    53826667     2632409
1013A,B     4108905
PROGSC4D       1      19        197   10.3        668    3.3   19.0       10010  1020A,B   227999124    46667334
1022A,B    11999953
PROGPS2D       1      13        240   18.4        291    1.2   13.0       10010  1025A,B   327602445    52935987
1027A,B    25200188
PROGSC6B       5     140        694    4.9        395    0.5   28.0       16884  1032A,B    78994223     3290769
1034A,B     2821222
PROGIT1D       1      10         36    3.6          0    0.0   10.0       10010  1041A,B   290379922    15713464
1043A,B    29037992
PROGIT8B       8      17        288   16.9        190    0.6    2.1       33436  1259A,B    35223857        2902
1261A,B    16575932
**TOTALS     173    1403      18632   13.2      18115    0.9    8.1       90328             82303700
2972755            10148638
```

You can use the Call Summary report to examine the detail of the call processing for each program, itemized by type or call and summarized for the monitor interval. An extract from the multi-page output is given in the following call summary report example. The calls using an I/O PCB are given first and sub-totaled. Then, the total calls, of each type, against each database PCB and each external subsystem are listed. The PSB TOTAL line marks the end of data for each program.

```
   IMS MONITOR   ****CALL SUMMARY****        TRACE START 1993 130   5:55:15     TRACE STOP  1993 130   5:59:49  PAGE
0186
                                                               (C)            (A)                       (B)
                  CALL  LEV         STAT                     IWAITS/       ..ELAPSED TIME...       .NOT IWAIT TIME..
DISTRIB.
PSB NAME PCB NAME FUNC  NO.SEGMENT  CODE   CALLS     IWAITS   CALL         MEAN      MAXIMUM       MEAN      MAXIMUM
NUMBER
--- ---- --- ---- ----  ----------  ----   -----     ------   ----         ----      -------       ----      -------
------
PROGSC6B I/O PCB  ISRT (  )                  138          0     0.00         372        1240          372        1240
```

```
598A,B,C
                        GU    (  )                 134        133     0.99     2600917    20974615     2587532    20962866
602A,B,C
                        (GU)  (  )                   3          0     0.00          15          16          15          16
716A,B,C
                        ASRT  (  )                   3          0     0.00         330         333         330         333
869A,B,C
                        GU    (  )         QC        2          1     0.50    17639806    21219588    17634776    21209529
870A,B,C
                        I/O PCB SUBTOTAL
                        ___ ___ ________           280        134     0.47     1370910                 1364469
                      PSB TOTAL
                        ___ _____                  280        134     0.47     1370910                 1364469
PROGSC2A I/O PCB        ISRT  (  )                 118          0     0.00         381        1496         381        1496
603A,B,C
                        GU    (  )                 114        284     2.49     3304809    21784513     3164423    21664181
632A,B,C
                        (GU)  (  )                   2          0     0.00          17          18          17          18
781A,B,C
                        ASRT  (  )                   3          0     0.00         367         444         367         444
871A,B,C
                        GU    (  )         QC        2          5     2.50    19931897    20045206    19799530    19925277
872A,B,C
                        I/O PCB SUBTOTAL
                        ___ ___ ________           239        289     1.20     1743339                 1675270
                        PSB TOTAL
                        ___ _____                  239        289     1.20     1743339                 1675270
PROGSC2D I/O PCB        ISRT  (  )                  14          0     0.00         377         621         377         621
608A,B,C
                        GU    (  )                  14         36     2.57    22360408    52048566    22221852    51901313
634A,B,C
                        I/O PCB SUBTOTAL
                        ___ ___ ________            28         36     1.28    11180393                11111115
```

Monitoring I/O for application program DL/I calls

The IMS Monitor report shows the total number of I/O occurrences and the total time the occurrences took for each application program executed during a monitored interval. The Program I/O report gives these two totals for all PSBs active during the monitored interval and includes the detailed breakdown of the I/O wait time as it was incurred by each PCB used by the program.

The detail of the report reveals much of the contention experienced during application program processing. Each type of conflict and the number of times it occurred are recorded for each I/O PCB. The report shows the total wait time, the highest wait experienced, and the average time. Subtotals are given for each PCB under a PSB, and for all PCBs under each PSB.

The DDN/FUNC column lists the data set by DD name. The MODULE column uses a code to indicate the source of the contention. The types of conflicts and codes are shown as follows:

- **Message handling**

Code	**Conflict**
MFS	MFS format library directory
PMM	Message format buffer pool space or control block I/O
QMG	Message queue management

- **Scheduling**

Code	**Conflict**
BLR	Load/read from ACBLIB
MSC	MPP region initialization
SMN	Virtual storage management

For external subsystem calls, the elapsed time to complete the processing is considered wait time. The DDN/FUNC column indicates the external subsystem call function, as follows:

- **External subsystems**

Code	**Subsystem call function**
AB0	ABORT
CT0	Create thread
D50	Terminate identify or thread, signoff
D80	INIT
I30	Identify, command, echo, terminate
I30	Identify, terminate subsystem
I50	INIT
I60	Resolve-in-doubt
PR0	Subsystem-not-operational
P10	Commit prepare (Phase 1)
P20	Commit continue (Phase 2)
SO0	Signon
SI0	Identify

The following example shows as sample of the program I/O report.

```
   IMS MONITOR   ****PROGRAM I/O****      TRACE START 1993 022  14:00:18   TRACE STOP 1993 022  14:02:20
PAGE 0088
                                    .........IWAIT TIME..........
PSBNAME  PCB NAME      IWAITS       TOTAL        MEAN     MAXIMUM   DDN/FUNC  MODULE
-------  --- ----      ------       -----        ----     -------   --------  ------

PROGHR1A I/O PCB          122     2341116       19189       70795   HOTELDBA   DBH
                           34    24177936      711115     3950160   **W F I
                           40    23652665      591316     2668917   **W F I
                            5       67613       13522       21214   SHMSG      QMG
                            4      110363       27590       60486   QBLKS      QMG

        PCB TOTAL
        ___ _____         131     2519092       19229
PSB TOTAL
___ _____                 305     6725063       20049
PROGDE1A TRMNALDA          20      624677       31233       68252   TRMNALDA   VBH
                            1      275811      275811      275811 PI TRMNALDA....

        PCB TOTAL
        ___ _____          21      900488       42880
         I/O PCB           16      488812       30550       79980   TRMNALDA   VBH
                            1       16118       16118       16118   SHMSG      QMG

        PCB TOTAL
        ___ _____          17      504930       29701
         TABLEDBA          16      290471       18154       33254   TABLEDA    DBH
        PCB TOTAL
        ___ _____          16      290471       18154
PSB TOTAL
___ _____                  54     1695889       31405
PROGHR2B HOTELDBB           8      698384       87298      184475   HOTELDBB   DBH
                            4     5820650     1455162     1455278 PI HOSINDXB....
                            4     4481024     1120256     1209075 PI HOTELDBB....
                            2      260817      130408      232750   HOSINDOB   VBH
                            7      106623       15231       16410   HOSINDXB   VBH
                            1       15366       15366       15366   HOTELDBD   DBH

        PCB TOTAL
        ___ _____          26    11382864      437802
PSB TOTAL
___ _____                  26    11382864      437802
PROGHR2A HOTELDBA          17      655801       38576      366108   HOSINDXA   VBH
                           73     1836721       25160       82141   HOTELDBA   DBH
                            2       54663       27331       41975   HOTELDBD   DBH
                            1        9887        9887        9887   HOTELDBC   DBH
                            2      851042      845635      845635   HOSINDOA   VBH

        PCB TOTAL
        ___ _____          95     3408114       35874
         I/O PCB           20      575847       28792       74227   HOTELDBA   DBH
                           21      370390       17637       43153   HOSINDXA   VBH
IMS MONITOR ****PROGRAM I/O**** TRACE START 1993 022 14:00:18  TRACE STOP  1993 022  14:02:20  PAGE 0089
                                    .........IWAIT TIME..........
PSBNAME  PCB NAME      IWAITS       TOTAL        MEAN     MAXIMUM   DDN/FUNC  MODULE
-------  --- ----      ------       -----        ----     -------   --------  ------

PROGHR2A I/O PCB            5     4654544      930908     2020043  **W F I
                            8    32796604     4099575     9328891  **W F I

        PCB TOTAL
        ___ _____          41      946237       23078
PSB TOTAL
___ _____                 136     4354351       32017
PROGPS2A LOGIMA            89     2046670       22996       73593   IMMSTR3A   VBH
                          612    53886417       88049      185674   IMMSTR1A   VBH
                            3       44906       14968       20788   IMINDEXA   VBH

        PCB TOTAL
        --- -----         704    55977993       79514
                          469    11742900       25038      170337   COMPOSDA   DBH
                          329     8198418       24919       91422   CPINDEXA   VBH

        PCB TOTAL
        ___ _____         798    19941318       24989
         I/O PCB            3       47511       15837       20806   SHMSG      QMG

        PCB TOTAL
        ___ _____           3       47511       15837
PSB TOTAL
___ _____                1505    75966822       50476
PROGSC6C I/O PCB           52     2698602       51896      473763   INVENTRC   VBH
                            4       70921       17730       34241   SHMSG      QMG
                            3       50699       16899       24724   QBLKS      QMG

        PCB TOTAL
        --- -----          59     2820222       47800
                           55     2666884       48488      210752   INVENTRC   VBH
                           50      797587       15951       41706   ININDEXC   VBH
                            1      119253      119253      119253 PI INVENTRC...1
                            1        8634        8634        8634   INVENTRB   VBH
                            2       83947       41973       53936   INVENTRA   VBH

        PCB TOTAL
        ___ _____         109     3676305       33727
PSB TOTAL
___ _____                 168     6496527       38669
PROGHR2D I/O PCB           21     2285296      108823      199223   HOTELDBD   DBH
                           28      762370       27227      111860   HOSINDXD   VBH
                            1       11685       11685       11685   SHMSG      QMG

        PCB TOTAL
        ___ _____          50     3059351       61187
         HOTELDBD          96     6279107       65407      139032   HOTELDBD   DBH
  MONITOR   ****PROGRAM I/O**** TRACE START 1993 022 14:00:18 TRACE STOP  1993 022  14:02:20  PAGE 0090
                                    .........IWAIT TIME..........
PSBNAME  PCB NAME      IWAITS       TOTAL        MEAN     MAXIMUM   DDN/FUNC  MODULE
-------  --- ----      ------       -----        ----     -------   --------  ------
```

```
PROGHR2D HOTELDBD           31     2130585       68728       769130   HOSINDXD   VBH
                             3      115999       38666        56394   HOTELDBA   DBH
                             2       69833       34916        43470   HOTELDBC   DBH
                             2       41430       20715        28020   HOSINDOD   VBH
                             4     5515374     1378843      1458884 PI HOSINDXD....
                             4     3997017      999254      1026228 PI HOTELDBD....

PCB TOTAL
___ _____                  142    18149345      127812
PSB TOTAL
___ _____                  192    21208696      110461
```

The I/O waits for the calls to the I/O PCB are grouped as the first entries for a PSB. For DC DL/I calls, the data set for which the I/O took place is indicated under the DDN/FUNC heading, and the module code indicates what type of conflict caused the wait. For external subsystem calls, the function is indicated under the DDN/FUNC heading, and the module code indicates the source of the call entry.

Names other than LGMSG and SHMSG can appear under the DDN/FUNC column for I/O PCBs.

If the program is designated as wait for input and has to wait for the input of the next message, the wait entry is marked **WFI under the DDN/FUNC heading and no entry appears in the MODULE column. The time spent waiting for the next input message is shown under wait time. **WFI entries are shown for information only and their values are not used to compute statistics.

Monitoring MFS activity

You can obtain a summary of all activity that occurs for management of message format buffer pool use from the Message Format Buffer Pool report.

The report is illustrated in the following example. The data shows the counts at the start and end of the trace interval and their difference.

When message formatting occurs, the appropriate message blocks must reside in the message format buffer pool, a DIF/MID pair for input or a DOF/MOD pair for output. If the blocks are not already in the buffer, I/O to the active IMSFORMATA/B library must occur. Block retrieval can involve a prior directory lookup, or be direct, using an index kept in the pool.

Many of the counts reveal details of internal event management. The number of times there is no directory entry for a block implies extra directory lookup I/O. Delays caused by unavailable FRE entries are recorded as request-ignored counts.

```
***IMS  MONITOR***  BUFFER POOL STATISTICS   TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0007

                   M E S S A G E   F O R M A T   B U F F E R   P O O L

                                                                           5:55:15           5:59:49
                                                                        START TRACE        END
TRACE        DIFFERENCE

    NUMBER OF P/F REQUESTS                                                      0
0                  0
    NUMBER OF I/F REQUESTS                                                     18
20                  2
    NUMBER OF I/F I/O'S                                                         2
2                  0
    NUMBER OF TIMES POOL COMPRESS WOULD BE SUCCESSFUL                           0
0                  0
    NUMBER OF DIRECTORY I/O OPERATIONS                                          2
2                  0
    NUMBER OF TIMES BLOCK WASHED FOR FRE                                        0
0                  0
    NUMBER OF TIMES P/F REQUEST IGNORED                                         0
0                  0
    NUMBER OF F/B REQUESTS                                                     18
20                  2
    NUMBER OF TIMES F/B REQUEST IGNORED                                         0
0                  0
    NUMBER OF TIMES I/F ON F/B QUEUE                                           16
18                  2
    NUMBER OF TIMES I/F ON I/F QUEUE                                            0
0                  0
```

```
    NUMBER OF TIMES F/B ON I/F QUEUE                                            18
20                 2
    NUMBER OF TIMES P/F ON I/F QUEUE                                             0
0                 0
    NUMBER OF TIMES P/F ON F/B QUEUE                                             0
0                 0
    NUMBER OF TIMES THERE WAS NO DIR ENTR FOR A BLOCK                            0
0                 0
    NUMBER OF TIMES I/O ERRORS POINT OR READ MACRO                               0
0                 0
    NUMBER OF IMMEDIATE I/O REQUESTS WAITED DUE TO MAXIMUM I/O                   0
0                 0
    NUMBER OF REQUESTS SATISFIED BY INDEX/DYNAMIC DIRECTORY                      0
0                 0

    QUOTIENT :  IMMEDIATE FETCH I/O'S + DIRECTORY I/O'S OPERATIONS =    0.00
                -----------------------------------------------------------------------------------
                                            TOTAL NUMBER OF TRANSACTIONS
```

Monitoring message queue handling

A key resource that directly affects the efficiency of transaction processing is the message queue pool and the management of the I/O to the message queues. You can examine the activity by looking at the Message Queue Pool report.

The following example illustrates the message queue pool report contents. Counts of activities are given at start and end of the trace interval and as the differences between start and end numbers.

```
***IMS  MONITOR***  BUFFER POOL STATISTICS    TRACE START 1993 130   5:55:15     TRACE STOP  1993 130
5:59:49  PAGE 0002

                   M E S S A G E   Q U E U E   P O O L

                                                                           5:55:15            5:59:49
                                                                        START TRACE          END
TRACE         DIFFERENCE

    NUMBER OF LOCATE CALLS FROM QMGR                                        54204
68436              14232
    NUMBER OF RECORD RELEASE CALLS FROM QMGR                                16431
20738               4307
    NUMBER OF LOCATE AND ALTER CALLS FROM QMGR                             131593
164744              33151
    NUMBER OF REQUESTS TO PURGE THE Q POOL                                      2
2                  0
    NUMBER OF ADDRESS TO DRRN TRANSLATION REQUESTS                          21351
27076               5725
    NUMBER OF REQUESTS TO WAIT FROM QMGR                                        0
0                  0
    NUMBER OF READ REQUESTS                                                   962
962                 0
    NUMBER OF WRITE REQUESTS(TOTAL)                                           499
499                 0
    NUMBER OF WRITES DONE BY PURGE                                            499
499                 0
    NUMBER OF WAITS FOR PURGE COMPLETION                                        1
1                  0
    NUMBER OF WAITS BECAUSE NO BUFFER AVAILABLE                                 0
0                  0
    NUMBER OF WAITS FOR OTHER DECB TO READ THIS BUFFER                        823
823                 0
    NUMBER OF WAITS FOR OTHER DECB TO WRITE THIS BUFFER                         0
0                  0
    NUMBER OF WAITS FOR CONFLICTING END DEQ BUFFER REQ                          0
0                  0
    NUMBER OF PSBS UNCHAINED FROM BUFFERS                                       0
0                  0
    NUMBER OF CALLS TO QMGR.(TOTAL)                                         48164
62213              14049
    NUMBER OF CALLS TO REPOSITION A LOST BUFFER                                 0
0                  0
    NUMBER OF CALLS TO ENQ A MESSAGE                                        10583
13441               2858
    NUMBER OF CALLS TO DEQ ONE OR MORE MESSAGE                               6321
7767               1446
    NUMBER OF CALLS TO CANCEL INPUT OR OUTPUT                                 119
121                 2
```

```
    QUOTIENT :  TOTAL NUMBER OF OSAM READS + OSAM WRITES + ALL IWAITS =     0.00
                -----------------------------------------------------
                              TOTAL NUMBER OF TRANSACTIONS
```

Detecting checkpoint effects

When a checkpoint command specifies SNAPQ, the current status of all message queues is written to the system log. This prevents any message handling on behalf of queue management.

General Iwait Time Events report records the wait time incurred by the SNAPQ. The following example shows the activity on the summary line QMGR SNAPQ CHECK. The number of occurrences is given with the total, average, and maximum wait times.

```
        IMS MONITOR   ** GENERAL REPORTS **        TRACE START 1993 130   5:55:15   TRACE STOP  1993
130   5:59:49  PAGE 0009
                GENERAL IWAIT TIME EVENTS
    EVENT                              ..........IWAIT TIME...........    DISTRIBUTION
    IWAITS          OCCURRENCES        TOTAL         MEAN      MAXIMUM       NUMBER
    ______          ___________        _____         ____      _______       ______
QMGR SNAPQ CHECK              0            0             0            0            0
                REGION AND JOBNAME REPORT
REG. NO.   JOB NAME
________   ________
     1     MPR1A100
     2     MPR1A209
     3     MPR1A210
     4     MPR1A211
     5     MPR1A103
     6     MPR1A101
     7     MPR1A115
     8     MPR1A116
     9     MPR1A216
    10     MPR1A200
    11     MPR1A217
    12     MPR1A119
    13     MPR1A218
    14     MPR1A219
    15     MPR1A104
    16     MPR1A220
    17     MPR1A203
    18     MPR1A123
    19     MPR1A222
    20     MPR1A105
    21     MPR1A124
    22     MPR1A223
    23     MPR1A107
    24     MPR1A224
    25     MPR1A106
    26     MPR1A206
    27     MPR1A205
    28     MPR1A108
    29     MPR1A109
    30     MPR1A208
    31     MPR1A111
    32     MPR1A112
    33     MPR1A113
    34     MPR1A204
    35     MPR1A114
    36     MPR1A102
    48     MPR1A121
    49     MPR1A122
    50     MPR1A221
```

Transaction Queuing report

In addition to monitoring the efficiency of message handling, you can monitor the service provided for each application, by looking at the size of the transaction queues at each scheduling of their processing programs.

The Transaction Queuing report shown in the following example records, for each transaction, the minimum, average, and maximum counts at scheduling time. The total number of dequeued transactions (or transactions that have been fully processed) during the monitored interval is given for each transaction code. The average number of transactions processed for each scheduling is given in the DEQUEUED MEAN column.

```
   IMS MONITOR   ****TRANSACTION QUEUING****    TRACE START 1993 130   5:55:15     TRACE STOP  1993
130   5:59:49  PAGE 0181
                                                       (B)                   (A)
                   NUMBER      NUMBER      ..ON QUEUE WHEN SCHEDULED.....   DEQUED      DISTRIBUTION
TRANSACTION        DEQUED      SCHEDS.     MINIMUM    MEAN        MAXIMUM    MEAN        NUMBER

-----------        ------      ------      -------    ----        -------    ----        ------

 SC6X                13           1           0       0.00           0      13.00         883A,B
 IT8W                17           3           0       0.00           0       5.66         887A,B
 TS1Z                16           1           0       0.00           0      16.00         891A,B
 PS3X                23           1           0       0.00           0      23.00         895A,B
 PS3Y                17           7           0       0.00           0       2.42         899A,B
 IT1V                11           6           0       0.00           0       1.83         903A,B
 SC2Z               143           2           0       0.00           0      71.50         907A,B
 IT8U                12           7           0       0.00           0       1.71         911A,B
 PS2W                10           1           0       0.00           0      10.00         915A,B
 TS1U                12           4           0       0.00           0       3.00         919A,B
 PS3W                14           1           0       0.00           0      14.00         923A,B
 IT8Y                16           5           0       0.00           0       3.20         927A,B
 IT1W                 9           1           0       0.00           0       9.00         929A,B
 SC2W                 9           1           0       0.00           0       9.00         933A,B
 IT2V                13           5           0       0.00           0       2.60         937A,B
 IT2W                17           6           0       0.00           0       2.83         941A,B
 TS1V                 9           1           0       0.00           0       9.00         945A,B
 SC2V                12           5           0       0.00           0       2.40         947A,B
 TS1W                11           1           0       0.00           0      11.00         949A,B
 PS3V                13           3           0       0.00           0       4.33         953A,B
 IT1U                 9           6           0       0.00           0       1.50         957A,B
 SC4U                11           5           0       0.00           0       2.20         962A,B
 SC6W                10           1           0       0.00           0      10.00         966A,B
 PS2V                 8           6           0       0.00           0       1.33         970A,B
 IT8X                12           1           0       0.00           0      12.00         974A,B
 SC4W                10           1           0       0.00           0      10.00         978A,B
 SC6U                14           6           0       0.00           0       2.33         982A,B
 IT2Y                 9           3           0       0.00           0       3.00         986A,B
 SC2X                15           1           0       0.00           0      15.00         990A,B
 PS2Y                17           2           0       0.00           0       8.50         994A,B
 SC4Y               152           4           0       0.50           1      38.00         998A,B
 SC2Y               106           2           0       0.00           0      53.00        1000A,B
 IT2X                20           1           0       0.00           0      20.00        1004A,B
 SC2U                15           3           0       0.00           0       5.00        1008A,B
 SC4Z               123           5           0       0.60           1      24.60        1010A,B
 TS1X                15           1           0       0.00           0      15.00        1015A,B
 SC4X                19           1           0       0.00           0      19.00        1019A,B
 PS2X                13           1           0       0.00           0      13.00        1024A,B
 PS2Z                20           5           0       0.00           0       4.00        1028A,B
 SC6Z               130           1           0       0.00           0     130.00        1031A,B
 SC6V                10           4           0       0.00           0       2.50        1035A,B
 SC6Y               143           1           0       0.00           0     143.00        1037A,B
 IT1X                10           1           0       0.00           0      10.00        1040A,B
 PS3U                19           6           0       0.00           0       3.16        1131A,B
 IT2U                13           4           0       0.00           0       3.25        1146A,B
```

Monitoring line activity

You can obtain a summary of all occurrences of activity for each node that handles message traffic during the monitored interval. The elapsed times and NOT-WAIT times are given in categories of total, mean, and maximum times for each communication line in the Communication Summary report.

The following example shows a sample of the communication summary report.

Requirement: You must match which physical devices are using the line to the Stage 1 output from system definition. The line numbers are assigned sequentially, according to their physical occurrence in the Stage 1 input deck.

If your online system specifies the prefetch option for MFS blocks in the control region JCL, the last line of the report contains the statistics for all prefetch events.

```
  IMS MONITOR   ****COMMUNICATION SUMMARY****   TRACE START 1993 130   5:55:15   TRACE STOP  1993 130
5:59:49  PAGE 0089
                                                   (A)                                 (B)
     NODE OR                          ........ELAPSED TIME.........       NOT IWAIT TIME(ELAPSED-
IWAIT)    DISTRIBUTION
   LINE NUMBER       OCCURRENCES       TOTAL         MEAN     MAXIMUM       TOTAL         MEAN
MAXIMUM      NUMBER
   ---- ------       -----------       -----         ----     -------       -----         ----
-------      ------
     PMTO1A                    3        2396          798        1547        2396          798
1547        1467A,B
         19                  182       92155          506        1106       92155          506
1106        1493A,B
          2                   59        2280           38          41        2280           38
41        1515A,B
      TOTAL
      -----                  244       96831          396                   96831          396
```

You can also investigate the amount of data transmitted across nodes with the Line Functions report. The following example shows a sample of a line-function report. The report distinguishes between input data and output data. The number of blocks of data and the average and maximum size of the blocks are recorded for data received by IMS and for transmitted data.

This report also includes a measure of how inactive the lines are. An inactive interval is assumed to be the difference between the time that marks the end of the last input block received and the starting time for output transmission. These occurrences of inactivity are termed turnaround intervals, and the report cumulates the number of occurrences as well as the average and maximum times associated with these intervals.

If the line is being used by an MFS supported terminal, a count of the number of requests for next page for a multi-page message is recorded.

If link traffic for coupled multiple systems is recorded, a set of three reports follows the Line Functions report.

```
  IMS MONITOR   ****LINE FUNCTIONS***      TRACE START 1993 130   5:55:15     TRACE STOP  1993 130   5:59:49  PAGE 0091
                          (A)......................(B)..............                       .....................
                            MEAN     MAX.                MEAN    MAX.      .          TURN         .                    .
  NODE OR   DEVICE  RECEIVE RECEIVE RECEIVE   TRANS.  TRANS.  TRANS.  DIST.       AROUND       MEAN         MAX.
DIST.   PAGING
LINE NUMBER  TYPE    BLOCKS BLKSIZE BLKSIZE   BLOCKS BLKSIZE BLKSIZE NUMBER    INTERVALS    INTERVAL    INTERVAL
NUMB.  REQUESTS
---- ------  ----    ------ ------- -------   ------ ------- ------- ------    ---------    --------    --------
-----  --------
  PMTO1A    3270V         1      29      29        2     170     171   1468A,B         3         798        1547
1466        0
      19    XXXX                                                                      182         506        1106
1492        0
       2    LOC SYS                                                                    59          38          41
1514        0
ALL LINES
___ _____                 1      29                2     170                          244
396                 0
```

Related concepts

"IMS Monitor reports for MSC" on page 627
The IMS Monitor Report Print program includes three reports that highlight message events caused by Multiple Systems Coupling.

Monitoring message handling efficiency

The IMS Monitor produces both summary and detailed information on asynchronous processing in the IMS control region. The arrival of data transmitted from terminals triggers the processing.

Application program responses also result in processing. The space in four major buffer pools and access to format, SPA, and message queue data sets are managed for the total communications traffic. Wait times are recorded when contention for pool space or I/O interrupts the processing of any of the communication tasks triggered by line activity. This information is contained in the Communication Wait report. The following example illustrates this report.

This report is complementary to the Communication Summary report in that the line number is used as an identification for the series of communication processing tasks.

```
 IMS MONITOR  ****COMMUNICATION IWAIT*****   TRACE START 1993 130  5:55:15   TRACE STOP  1993 130
5:59:49  PAGE 0090
```

```
  NODE OR                        .........IWAIT TIME..........
DIST.
LINE NUMBER  OCCURRENCES       TOTAL        MEAN      MAXIMUM     FUNCTION           BLKSIZE
MODULE       NO.
---- ------  -----------       -----        ----      -------     --------           -------  ------
----
ALL LINES...
PREFETCH I/O
-------- ---         NONE
```

IMS internal resource usage

There are several summary reports that you can use to examine the level of internal contention for resources.

The following list gives a brief explanation of these reports.

Pool Space Failure Summary report

The Pool Space Failure Summary report gives the number of times in each region a given amount of storage was unavailable. It shows the number of bytes and the identification of the pool as well as the number of occurrences of this failure to obtain storage. You can use this summary to determine whether you need to increase the buffer pool allocation by a system definition change or by overriding the number of buffers in the EXEC statements in the JCL.

The format of the report is shown in the following example.

```
POOL SPACE FAILURE SUMMARY

          POOL ID    BYTES REQ.   OCCURRENCES
          DLMP           8888           1
     TOTAL                              1
```

Latch Conflict Statistics report

The basic serialization of the task processing in IMS is controlled by ownership of an IMS latch. When different programs are executing, they compete for the ownership. If they wait for the resource, the one possessing the latch has to post the other ITASK waiting for it. You can judge the level of contention for a resource and then investigate a set of changes to relieve the pressure.

The different types of latches and the counters that exhibit the level of contention are given in the Latch Conflict Statistics report. The following figure is an example of this report. The entries are organized according to the latch names.

When a system checkpoint is taken during the time the monitor is active, latch conflict statistics are reset to zero, thus corrupting the values presented in this report. If this situation exists, the following message will be inserted at the top of the report:

```
**** A CHECKPOINT OCCURRED DURING MONITOR RUN ****
****  LATCH CONFLICT STATISTICS ARE INVALID   ****
****      SEE UTILITIES REFERENCE MANUAL      ****
```

However, if the master terminal operator issues the **/CHECKPOINT** command with the STATISTICS keyword parameter, latch conflict statistics are reset to zero, but the IMS monitor is not notified. Therefore, DFSUTR20 cannot detect that the statistics have been corrupted and does not issue this message.

Recommendation: Do not issue statistics checkpoints while the Monitor is running.

The counters are primarily concerned with storage management and logging services. The statistics recorded are the number of times contentions occur, that is, the resource waits for a latch.

```
         IMS MONITOR   ** GENERAL REPORTS **    TRACE START 1993 209...

                 LATCH CONFLICT STATISTICS

LATCH       COUNT                 AT          AT
NAMES       FIELD           START       END        DIFF.
```

```
LOGL      CONTENTIONS                 0           0           0

SMGT      CONTENTIONS                 0           0           0

XCNQ      CONTENTIONS                 0           0           0

ACTL      CONTENTIONS                 0           0           0


CBTS      CONTENTIONS                 0           0           0

DBLK      CONTENTIONS                 0           0           0
```

Using frequency distributions from IMS Monitor output

The reports that are derived from the IMS Monitor data records contain many summary lines where the mean time is given. If you are interested in the distribution of those timed events, rather than just average and maximum times, you can request the Report Print utility to individually record the events in a frequency distribution across a range of intervals.

Some distributions are not time dependent, such as those for transaction queue loads or transmitted block sizes.

The following tables show the major IMS Monitor reports and the type of frequency distributions generated for each report. Each type results in several distributions, depending on how many entries are in each section of the report. For each type of frequency distribution the data is cumulated in suitable intervals or ranges. The set of ranges used for each type is given an identifier, shown in the ID column.

- The following table shows the report distributions sorted by Region Summary.

Table 72. Report distributions by region summary

Report name	ID	Description
Scheduling and Termination	D1	Elapsed time
	D2	Not wait time
Schedule end to 1st DL/I call	D3	N/A
Elapsed execution time DL/I calls	D4	N/A
	D5	Elapsed time
External Subsystem calls	D6	Not wait time
Waits per DL/I call	D43	Elapsed time
Idle for intent Checkpoint	D7	N/A
	D8	N/A
	D20	Elapsed time
	D21	Not wait time

- The following table shows the report distributions Programs Region.

Table 73. Report distributions by program region

Report name	ID	Description
Elapsed execution time	D30	N/A
Schedule and to 1st DL/I call	D31	N/A

- The following table shows the report distributions sorted by Program Summary.

Table 74. Report distributions by program summary

Report name	ID	Description
Processor time per schedule	D15	N/A
Transactions dequeued per schedule	D14	N/A
Elapsed time per schedule	D9	N/A
Schedule end to 1st DL/I call	D10	N/A

- The following table shows the report distributions sorted by Communication Summary.

Table 75. Report distributions by communication summary

Report name	ID	Description
Line elapsed time	D18	N/A
Line not wait time	D19	N/A

- Table 76 on page 674 shows the report distributions sorted by Line Functions.

Table 76. Report distributions by line functions

Report Name	ID	Description
Received block length	D36	N/A
Transmitted block length	D37	N/A
Inactive intervals	D38	N/A

- The following table shows the report distributions sorted by MSC Queuing Summary.

Table 77. Report distributions by MSC queuing summary

Report name	ID	Description
Time in queue	D39	N/A

- The following table shows the report distributions sorted by Transaction Queuing.

Table 78. Report distributions by transaction queuing

Report name	ID	Description
Transactions on queue at schedule	D17	N/A
Transactions dequeued per schedule	D16	N/A
Prefetch format blocks	D28	Elapsed time
	D29	Not wait time

- The following table shows the report distributions sorted by Call Summary.

Table 79. Report distributions by call summary queuing

Report name	ID	Description
PSB waits per DL/I call	D13	N/A
PSB waits per external subsystem call	D44	N/A
PSB elapsed time per call	D11	N/A
PSB not wait time per call	D12	N/A
PSB external subsystem calls	D45	Elapsed time

- The following table lists some distributions derived from buffer pool statistics for wait times.

Table 80. Wait time distributions

Function	ID	Module key
Storage	D22	SMN
Scheduler internal	D25	MSC
Queue manager I/O	D26	QMG
Block loader I/O	D27	BLR
MFS block I/O	D32	MFS
MFS directory I/O	D33	MFS
Format buffer pool space	D35	PMM
QMGR SNAPQ check	D42	None

How to get a frequency distribution output

To request the IMS Monitor Report Print utility to gather distribution data, include a DIS input control statement. This causes all report items with an entry under a column headed MEAN to have a corresponding frequency distribution as part of the Distribution Appendix report. Each report line includes an identifying reference number under the column headed Distribution Number. You can use the reference number to locate the distribution data flagged by that number in the appendix.

How frequency distribution ranges are defined

A set of ten intervals is defined for each summary line and the occurrences falling in each interval are cumulated. The interval ranges are preset with default end points. For example, the end points, for DL/I call elapsed time are: 0, 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000, 256000, INF (all times are in milliseconds). The default end points are chosen so that they are suitable to the event. The lower limit of the first interval always defaults to zero, and the upper limit of the tenth interval is infinity (INF).

Although several types of distribution can use the same set of end points, each type is assigned a distribution identifier. You can use this to redefine the end points. To override the default end points you include an input control statement to the Report Print utility. The statement specifies the type of distribution identifier and gives the desired end point values.

The DL/I call elapsed time end points could be respecified by:

```
D5   0,500,1000,1500,2000,4000,,,100000,500000
```

The values of the unspecified end points remains at their default values of 32000 and 64000 as does the last (INF).

The following example shows a sample page from the Distribution Appendix report, which gives an example of how ranges vary with the type of distribution. The lines are arranged in pairs, with the second one recording the cumulated counts.

```
  IMS MONITOR  ****DISTRIBUTION APPENDIX****   TRACE START 1993 130   5:55:15   TRACE STOP 1993 130
5:59:49 PAGE 0200

#
1..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                     0          0          0          0          0          0          0
0          0          1
#
2..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                     0          0          0          0          0          0          0
0          0          1
```

```
#
3..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                    0          0          0          0          0          0          0
0         0         1
#
4..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                    0          0          0          0          0          0          0
0         0         1
#
5..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                    0          0          0          0          0          0          0
0         0         6
#
6..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                    0          0          0          0          0          0          0
0         0         8
#
7..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                    0          0          0          0          0          0          0
0         0         1
#
8..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                    0          0          0          0          0          0          0
0         0         9
#
9..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000....
1800000....INF
                    0          0          0          0          0          0          0
0         0         1
#
10..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                    1          0          0          0          0          0          0
0         0         7
#
11..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                    0          0          0          0          0          0          0
0         0         1
#
12..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                    0          0          0          0          0          0          0
0         0         8
#
13..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                    0          0          0          0          0          0          0
0         0         1
#
14..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                    0          0          0          0          0          0          0
0         0         1
#
15..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                    1          0          0          0          0          0          0
0         0         8
#
16..........0.....200000.....400000.....600000.....800000....1000000....1200000....1400000....1600000...
.1800000....INF
                    0          0          0          0          0          0          0
0         0         1
```

Default values of distribution definitions

Using an identifier provided in the frequency distribution tables and the wait time distributions table you can determine the default end points for the distribution by locating it in the following list:

D1, D2, D5, D6, D9, D10, D11, D12, D15 D18, D19, D20, D21, D22, D25, D27, D28, D29, D30, D31, D43, and D45
: 0, 1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000, 256000, INF

D3
: 0, 50000, 100000, 150000, 200000, 250000, 300000, 350000, 400000, 450000, INF

D4
: 0, 200000, 400000, 600000, 800000, 1000000, 1200000, 1400000, 1600000, 1800000, INF

D7, D13, D44
: 0, 0, 1, 2, 3, 4, 5, 6, 7, 8, INF

D8
: 0, 100000, 200000, 300000, 400000, 500000, 600000, 700000, 800000, 900000, INF

D14, D16, D17
: 0, 1, 2, 3, 4, 5, 10, 15, 30, 90, INF

D23, D24, D26, D32, D40, D42
: 0, 2000, 8000, 24000, 50000, 100000, 150000, 200000, 250000, 300000, INF

D33, D34, D35
: 0, 2000, 4000, 8000, 16000, 32000, 64000, 96000, 128000, 160000, INF

D36, D37
: 0, 10, 20, 40, 80, 100, 200, 400, 800, 1000, INF

D38
: 0, 1000, 10000, 100000, 200000, 500000, 800000, 1000000, 1500000, 2000000, INF

D39
: 0, 1000, 5000, 10000, 50000, 100000, 500000, 1000000, 5000000, 10000000, INF

Interpreting Distribution Appendix output

You can use the detailed output in the Distribution Appendix when you suspect an unusual combination of events was reported in a report summary line. Usually, the average and maximum times or counts are sufficient to highlight a resource usage problem.

However, if you suspect the mean value to be masking an unusual distribution you can draw on the detail contained in the IMS Monitor output records.

For example, suppose you are investigating a change in the scheduling algorithm for a particular transaction and need to know how many transactions were able to be processed for each scheduling of an application program. The following figure shows the processed transactions in a histogram:

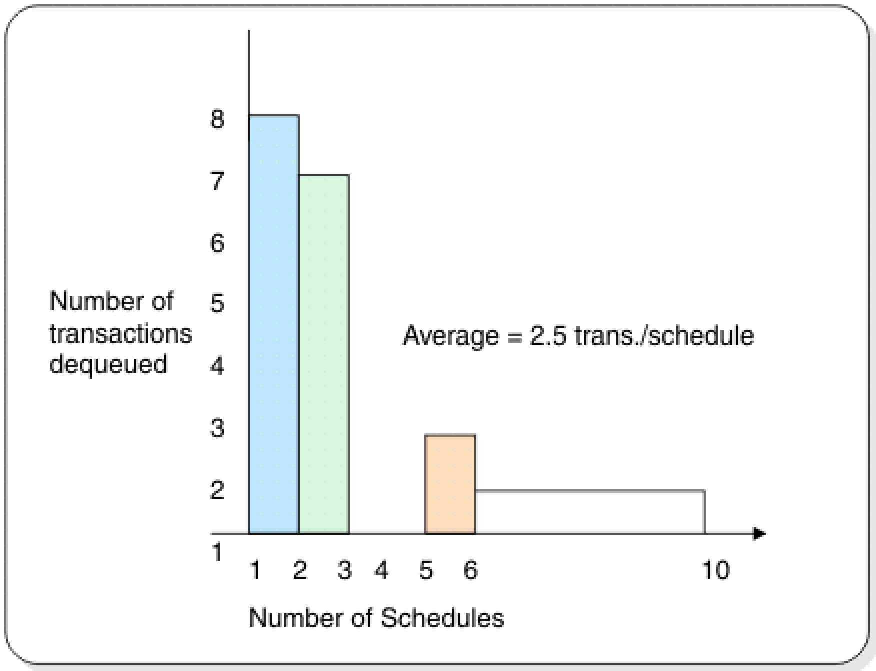


Figure 73. Number of transactions processed for each scheduling of an application program

The average is 2.5 transactions per schedule. The distribution in the figure suggests that many schedules were able to process only one or two transactions, and few schedules significantly exhausted the queue. The distribution data for the histogram is as follows:

```
Number of schedules    1    2    3    4    5    6-10    >10

Transactions dequeued  8    7    0    0    2    1       0
```

The Distribution Appendix presents the histogram data in the form of two lines:

- The first line shows the intervals, prefixed by a cross reference to an individual line on the earlier output.
- The second line gives the number of events occurring in those intervals.

This data appears as follows:

```
# 950B...0...1...2...3...4...5...10...15...30...90...INF
            8   7   0   0   2   1    0    0    0    0
```

The cross reference 950B points to a unique report line. For example, the Transaction Queuing report on the appropriate line for the transaction of interest shows 950A,B under the column headed DISTRIBUTION NUMBER. Use the reference number 950B to locate the data in the Distribution Appendix. The 950A reference points to the data for the number of transactions in the queue at schedule time.

Chapter 51. //DFSSTAT reports

The //DFSSTAT reports show you how many DB and DC calls are issued by an application program and describe buffering activity during the application's execution. The reports are written when the application program terminates.

To get the reports, you must put a //DFSSTAT DD statement in the JCL of your batch region or online dependent region. The following is an example of a //DFSSTAT DD statement.

```
//DFSSTAT DD   SYSOUT=A
```

Recommendation: Although it is supported, do not include the //DFSSTAT DD statement in the JCL for an MPP region. If you do not include the //DFSSTAT DD statement, you avoid the overhead and large amount of output that results from creating one set of reports each time a short-running MPP terminates.

The following topics describe the reports that //DFSSTAT creates.

Related concepts

"Tools for detailed monitoring" on page 367
Many of the monitoring tools you can use to collect detailed data are also used for general diagnostics. The principal tool provided by IMS for collecting and analyzing data is the IMS Monitor, which allows you to monitor online subsystems.

PST-Accounting report

This report shows how many DB and DC calls are issued by an application program.

Fields in the PST-Accounting report

The following example shows a PST-Accounting report and shows the names of each field. Each field in this report represents one type of DB or DC call. For example, the DB GU CALLS field shows how many database Get Unique calls were issued by the application.

```
             *** PST ACCOUNTING STATISTICS ***
DB GU CALLS                2
DB GN CALLS                2
DB GNP CALLS               1
DB GHU CALLS               1
DB GHN CALLS               1
DB GHNP CALLS              1
DB ISRT CALLS              2
DB DLET CALLS              1
DB REPL CALLS              1
DB CALLS (TOTAL)          12
DB DEQ CALLS               1
DB RLSE CALLS              0
MSG GU CALLS               2
MSG GN CALLS               2
MSG CHNG CALLS             4
MSG ISRT CALLS             8
MSG PURGE CALLS            4
MSG CMD CALLS              1
MSG GCMD CALLS             1
MSG AUTH CALLS             1
MSG SETO CALLS             4
SYS APSB CALLS             0
SYS DPSB CALLS             0
SYS GMSG CALLS             2
SYS ICMD CALLS             1
SYS RCMD CALLS             2
SYS CHKP CALLS             0
SYS XRST CALLS             0
SYS ROLB CALLS             1
SYS ROLS CALLS             2
SYS SETS CALLS             1
SYS SETU CALLS             1
SYS INIT CALLS             1
SYS INQY CALLS              3
SYS LOG CALLS               1
```

VSAM-Buffer-Pool report (for batch regions only)

The VSAM-Buffer-Pool report describes VSAM buffer pool activity during the execution of an application program. A separate report is written for each VSAM subpool. The last VSAM-Buffer-Pool report summarizes the buffering activity in all the VSAM subpools used by the application.

Using the VSAM-Buffer-Pool report

This report is written only for applications that you run in batch regions.

The primary use of the VSAM-Buffer-Pool report, an example is shown in the following example, is to see how many I/O operations were issued in each VSAM subpool.

```
                   *** VSAM BUFFER POOL STATISTICS ***
 FIX INDEX/BLOCK/DATA                                  N/Y/N
 SHARED RESOURCE POOL ID                                VPL1
 SHARED RESOURCE POOL TYPE                                 D
 SUBPOOL BUFFER SIZE                                   4,096
 TOTAL BUFFERS IN SUBPOOL                              1,000
 TOTAL HIPERSPACE BUFFERS IN SUBPOOL                      50


 RETRIEVE BY RBA CALLS                                   370
 RETRIEVE BY KEY CALLS                                187583
 LOGICAL RECORDS INSERTED INTO ESDS                      310
 LOGICAL RECORDS INSERTED INTO KSDS                     9823
 LOGICAL RECORDS ALTERED IN THIS SUBPOOL                   0
 TIMES BACKGROUND WRITE FUNCTION INVOKED                   0
 SYNCHRONIZATION CALLS RECEIVED                        29923
 PERM WRT ERROR BUFFS NOW IN THE SUBPOOL                   0
 LARGEST NBR OF PERM ERR BUFFS EVEN IN THE SUBPL           0
 VSAM GET CALLS ISSUED                                     0
 VSAM SCHBFR CALLS ISSUED                             189290
 CONTROL INTERVAL REQUESTED ALREADY IN POOL                0
*CONTROL INTERVAL READ FROM EXTERNAL STORAGE           51238
*VSAM WRITES INITIATED BY IMS                         138637
*VSAM WRITES TO MAKE SPACE IN THE POOL                  9288
 VSAM READS FROM HIPERSPACE BUFFERS                        0
 VSAM WRITES FROM HIPERSPACE BUFFERS                       0
 FAILED VSAM READS FROM HIPERSPACE BUFFERS                 0
 FAILED VSAM WRITES TO HIPERSPACE BUFFERS                  0

*TOTAL I/O OPERATIONS                                 199163
```

Fields in the VSAM-Buffer-Pool report

The VSAM-Buffer-Pool report is identical to the VSAM-Buffer-Pool report written by the DB monitor, with the following exceptions:

- Although the field names in the DB Monitor's VSAM Buffer Pool report are preceded by "NUMBER OF", the fields in both reports have the same meaning.
- The //DFSSTAT VSAM-Buffer-Pool report does not keep track of the start trace and end trace times. This is unnecessary because information is always gathered for the //DFSSTAT reports from the beginning to the ending of the application's execution.
- The //DFSSTAT VSAM-Buffer-Pool report contains a "TOTAL I/O OPERATIONS" field, which is the sum of the following:
  - The number of times a CI was read into the buffer from the database (CONTROL INTERVAL READ FROM EXTERNAL STORAGE field in the report)
  - The number of times a buffer was written to the database (VSAM WRITES INITIATED BY IMS field in the report)

- The number of times a buffer was written to the database so a new CI could be read into the buffer (VSAM WRITES TO MAKE SPACE IN THE POOL field in the report).
- The //DFSSTAT VSAM-Buffer-Pool report includes a summary report. The summary report is preceded by SUBPOOL BUFFER SIZE=ALL. It contains a summary of read and write information for all VSAM Buffer Pool reports.

The fields that represent I/O operations are highlighted on the left by an asterisk (*).

Related reading: See "VSAM-Buffer-Pool report" on page 573 for a description of the various fields in the report.

OSAM-Buffer-Pool report (for batch regions only)

The OSAM-Buffer-Pool report describes the OSAM buffer pool activity during an application's execution. This report is only written for programs running in batch regions.

Using the OSAM-Buffer-Pool report

The primary use of the OSAM-Buffer-Pool report, an example is shown in the following example, is to see how many OSAM I/O operations were issued. This report does not, however, show Sequential Buffering (SB) related information.

```
    *** OSAM DATA BASE BUFFER POOL STATISTICS ***
 FIX BLOCK DATA                                      Y/Y
 SUBPOOL ID                                         004K
 SUBPOOL BUFFER SIZE                                4096
 TOTAL BUFFERS IN SUBPOOL                           1000


 LOCATE-TYPE CALLS                               1765296
 REQUESTS TO CREATE NEW BLOCKS                         0
 BUFFER ALTER CALLS                               340800
 PURGE CALLS                                       39371
 LOCATE-TYPE CALLS, DATA ALREADY IN SUBPOOL      1370897
 BUFFERS SEARCHED BY ALL LOCATE-TYPE CALLS       1987604
*READ I/O REQUESTS                                375355
*SINGLE BLOCK WRITES BY BUFFER STEAL RTN               0
*BLOCKS WRITTEN BY PURGE                          150284
 TOTAL NBR OF I/O ERRORS FOR THIS SUBPOOL              0
 BUFFERS LOCKED DUE TO WRITE ERRORS                    0
 LOCATE CALLS WAITED DUE TO BUSY ID                 1431
 LOCATE CALLS WAITED DUE TO BUFR BUSY WRITE            0
 LOCATE CALLS WAITED DUE TO BUFR BUSY READ             0
 BUFR STEAL/PURGE WAITED FOR OWNERSHIP RLSE          296
 BUFFER STEAL REQUEST WAITED FOR BUFFERS               0

*TOTAL I/O OPERATIONS                             525639
```

Fields in the OSAM-Buffer-Pool report

The figure is an example of an OSAM-Buffer-Pool report.

The OSAM-Buffer-Pool report is identical to the Database-Buffer-Pool report written by the DB monitor, with the following exceptions:

- The field names in the IMS Monitor's Database-Buffer-Pool report are preceded by "NUMBER OF", although the fields have the same meaning in both reports.
- The //DFSSTAT OSAM-Buffer-Pool report does not keep track of the start trace and end trace times. This is unnecessary because information is always gathered for the //DFSSTAT reports from the beginning to the ending of the application's execution.
- In addition, the //DFSSTAT OSAM-Buffer-Pool report contains a "TOTAL I/O OPERATIONS" field, which equals the sum of the following fields:
  - READ REQUESTS ISSUED
  - OSAM WRITES ISSUED

- – QUEUED WRITES ISSUED
- – FORMAT LOGICAL CYLINDER REQUESTS
- – BISAM READS OR QISAM SETLS.

These fields represent I/O operations and are highlighted on the left by an asterisk (*).

Sequential-Buffering-Summary report

The Sequential-Buffering-Summary report provides an overview of SB-related information for the application. (VSAM-related information is not included.)

Using the Sequential-Buffering-Summary report

From an SB Summary report, an example is shown in the following example, you can determine if the application benefited from the use of SB. When using this report, pay particular attention to these fields:

- NBR BLOCKS READ SEQUENTIALLY and PCT OF TOTAL
- PERCENT READ PER SEARCH REQUEST

```
*** SEQUENTIAL BUFFERING SUMMARY FOR THE APPLICATION ***

DFSSBUX0 DISALLOWED USAGE OF SB:                        NO
DFSSBUX0 REQUESTED CONDITIONAL SB ACTIVATION:           NO
AT LEAST ONE SB= KEYWORD IN PSB:                       YES
AT LEAST ONE SBPARM CONTROL STMT FOR APPLICATION:       NO
SBPARM CONTROL CARD(S) READ FROM //DFSCTL:             YES
AT LEAST ONE SBPARM PSB= SPECIFIED THAT MATCHED PSB: YES
AT LEAST ONE SBPARM DB= SPECIFIED THAT MATCHED DB:   YES
AT LEAST ONE SBPARM PCB= SPECIFIED THAT MATCHED PCB:  NO
AT LEAST ONE SBPARM DD= SPECIFIED THAT MATCHED DD:    NO

NUMBER OF SEARCH REQUESTS ISSUED BY OSAM BH:
      SEARCH                                         2,213
NUMBER OF READ I/O:
      RANDOM READ                                      686
      SEQUENTIAL READ                                  652
NUMBER OF BLOCKS READ:
      TOTAL NUMBER BLOCKS READ                       7,206
      NBR BLOCKS READ AT RANDOM                        686        PCT OF TOTAL:
      NBR BLOCKS READ SEQUENTIALLY                   6,520        PCT OF TOTAL:  9
PERCENT READ PER SEARCH REQUEST                      60.46
NUMBER OF SEQUENTIAL I/O ERRORS                          0
```

Fields in the Sequential-Buffering Summary report

The example shows a Sequential-Buffering-Summary report.

The first part of this report shows why Sequential Buffering was or was not used. This part of the report describes whether:

- A SBONLINE control card was provided in DFSVSMxx (this applies only to IMS DC environments).
- A **/STOP SB** command was in effect when the application program started (this applies only to IMS DC environments).
- The SB Initialization Exit Routine (DFSSBUX0) disallowed use of SB.
- The SB Initialization Exit Routine (DFSSBUX0) requested conditional activation of SB by default.
- At least one SB= keyword was provided during PSBGEN.
- The //DFSCTL file contained at least one SBPARM control statement that applied to the application program.
- SBPARM control cards have been read. If the answer is Yes, the following statistics indicate what SBPARM keywords were used. This can be helpful in determining why sequential buffering was or was not used for the application program.

- At least one PSB= keyword was specified on an SBPARM control card and it matched the PSB used by the application.
- At least one DB= keyword was specified on an SBPARM control card where the PSB matched or was not specified, and the database matched one used by the application.
- At least one PCB= keyword was specified on an SBPARM control card where the PSB and DB matched or were not specified, and the PCB name matched one used by the application.
- At least one DD= keyword was specified on an SBPARM control card where the PSB, DB, and PSB matched or were not specified, and the DD name matched one used by the application.
- Whether SBPARM control cards have been read. If the answer is "yes," the following statistics indicate what SBPARM keywords were used. This is helpful in determining why sequential buffering was or was not used for the application program.
- At least one PSB= keyword was specified on a SBPARM control card and it matched the PSB used by the application.

The other fields in the report are as follows:

NUMBER OF SEARCH REQUESTS ISSUED BY THE OSAM BH

This field shows you how many times the OSAM buffer handler asked the SB buffer handler to search the SB buffer pools for a specific OSAM block.

The value in this field is equal to the number of OSAM random read I/O operations that would have been issued without SB.

NUMBER OF READ I/O

These fields show you the number of OSAM random and sequential read I/O operations it took to satisfy requests made by the application program. The sum of these two numbers is the total number of OSAM read I/O operations issued on behalf of the application. You can subtract this sum from the NUMBER OF SEARCH REQUESTS ISSUED BY THE OSAM BH field to calculate how many read I/O operations you saved by using SB.

NUMBER OF BLOCKS READ

These fields tell you how many OSAM data set blocks were read to satisfy requests from the application program. These fields show you:

- The total number of blocks read
- The number and percentage of blocks read with a random read
- The number and percentage of blocks read with a sequential read

If the percentage of blocks read with a sequential read is high, SB probably helped reduce the elapsed time of the application program.

PERCENT READ PER SEARCH REQUEST

This field shows you the number of read I/O operations issued by the SB buffer handler expressed as a percentage of the number of times the OSAM buffer handler asked the SB buffer handler to search for a block.

A low percentage indicates that many of the search requests were satisfied without issuing an I/O operation. Therefore, a low number in this field shows that SB probably helped reduce the elapsed time of the application program.

NUMBER OF SEQUENTIAL I/O ERRORS

This field describes the number of sequential reads that resulted in I/O errors. When an I/O error is detected during a sequential read, IMS increments this field and marks the 10 SB buffers involved in the read as invalid. Then IMS issues a random read for the block that was requested by the OSAM buffer handler.

Sequential Buffering Detail report

This report gives you detailed information about how SB was used for a particular SB buffer pool. A separate report is created for each SB buffer pool used by the application program.

Each report consists of three pages, A, B, and C. Summary information is contained on page A. More detailed information is found on pages B and C.

Each of the following topics contains an example of the relevant page (A, B or C) of the Sequential-Buffering-Detail report.

Fields on page A of the Sequential Buffering Detail report

The following example shows page A of the report.

```
//DFSSTAT STATISTICS FOR: JOB=OSBTC01   STEP=STEP1   .          PGM=DFSDDLT0  PSB=PBVDSALR  DATE=93.058
TIME=09.39
---------------------------------------------------------------------------------------------------------
---------------
                       *** SB DETAIL STATISTICS   (PAGE A) ***
PSB                 PBVDSALR
DB                  DBOVLFPC
PCB
DB-PCB NBR                 1
DSG-CB NBR                 1
DD                  VLOSAM01
DB-ORG              HDAM
DD-TYPE             *PSDATA
NBR OF BUFSETS             4
COMPARE-OPTION IS ACTIVE
** NUMBER OF SEARCH REQUESTS ISSUED BY OSAM BH:
      SEARCH                                           2,213
** NUMBER OF READ I/O:
      TOTAL                                            1,338
      RANDOM READ                                        686
      SYNCHRONOUS SEQUENTIAL READ                        555
      OVERLAPPED  SEQUENTIAL READ                         97
** NUMBER OF BLOCKS READ:
      TOTAL                                            7,206
      RANDOM READ                                        686        PCT OF TOTAL:   9.51
      SYNCHRONOUS SEQUENTIAL READ                      5,550        PCT OF TOTAL:  77.01
      OVERLAPPED  SEQUENTIAL READ                        970        PCT OF TOTAL:  13.46
** AVERAGE I/O WAIT TIMES (MILLIS):
      RANDOM READ                                      15.70
      SYNCHRONOUS SEQUENTIAL READ                      18.03
      OVERLAPPED  SEQUENTIAL READ                        .26
```

You can use the first topic of this page to identify the SB buffer pool, database PCB, and DD name this report applies to. There can be more than one SB buffer pool (and, thus, more than one report) created for a particular database PCB and DD name. This can happen if the PCB is involved in a logical relationship.

This topic contains the following information:

- The PSB name.
- The DBD name (as coded in the PCB macro during PSBGEN).
- The PCB label (as coded in the PCB macro during PSBGEN).
- The relative number of the database PCB within the PSB.
- A unique identifier of the data set group control block within the database PCB. (You can use this number to uniquely identify the SB buffer pool when more than one SB buffer pool has been created for the same PCB and DD name.)
- The DD name.
- The type of database organization (HDAM, HIDAM, PHDAM, PHIDAM, or HISAM).
- The type of database data set. This can be one of three values:
  - *INDX indicates this report applies to a data set used as an index.
  - *PSDATA indicates this report applies to a data set containing data accessed according to its primary sequence.

  - *SSDATA indicates this report applies to a data set containing data accessed according to a secondary index sequence or by crossing a logical relationship.
- The number of buffer sets in the SB buffer pool. The default number of buffer sets is four. You might have changed this default, however, with a SBPARM control statement or in the SB Initialization Exit routine.

NUMBER OF SEARCH REQUESTS ISSUED BY OSAM BH
This field shows you the number of OSAM random read I/O operations that would have been issued without using SB.

NUMBER OF READ I/O
These fields show you the number of read I/O operations that were actually issued in this SB buffer pool. The fields tell you:

- The total number of OSAM read I/O operations
- The number of random reads
- The number of synchronous sequential reads
- The number of overlapped (asynchronous) sequential reads

You can use these fields to calculate the percentage of each of the three types of read I/O operations that were used for this database PCB and DD name. You can also subtract the total number of OSAM read I/O operations from the NUMBER OF SEARCH REQUESTS ISSUED BY OSAM BH field to determine the number of read I/O operations you saved by using SB.

NUMBER OF BLOCKS READ
These fields show you how many blocks were read by each type of read I/O operation. The fields tell you:

- The total number of blocks read
- The number and percentage of blocks read by random reads
- The number and percentage of blocks read by synchronous sequential reads
- The number and percentage of blocks read by overlapped (asynchronous) sequential reads

A high percentage of blocks read with sequential reads indicates that SB probably helped the application program run faster, at least while processing this database PCB and data set.

A high percentage of blocks read with random reads, however, might indicate:

- A large amount of random processing by the application program
- The OSAM data set associated with this database PCB needs reorganizing

If many blocks were read with random reads, and if the program was processing the database sequentially, you can sometimes get better buffering performance by increasing the number of buffer sets (for example, increase the value of the BUFSETS parameter on the SBPARM control statement in //DFSCTL). After increasing the number of buffer sets, observe how the number reported in the NUMBER OF READ I/O field changes the next time the application is executed.

A small percentage of blocks read at random sometimes indicates that you can reduce the number of buffer sets to save virtual storage space.

AVERAGE I/O WAIT TIMES (MILLIS)
These fields tell you the average I/O wait times for each of the three types of read I/O operations. These times show the average time the application program waited for a read I/O operation to complete before it could process the data being read.

The times in these fields are in milliseconds. A millisecond is one thousandth of a second (in other words, 50 milliseconds equals 0.050 seconds).

These fields show you:

- The average I/O wait time for random reads
- The average I/O wait time for synchronous sequential reads

- The average I/O wait time for overlapped (asynchronous) sequential reads

These times are only measured when SB is active or IMS is monitoring the I/O reference pattern.

Fields on page B of the Sequential Buffering Detail report

The following example shows page B of the report.

```
--------------------------------------------------------------------------------
                      *** SB DETAIL STATISTICS: REFERENCE STATISTICS   (PAGE B) ***
** REFERENCES IN BUFFER-SETS:
      RATIO                                  .23
** REFERENCES IN RANDOM SRAN CBS:
      RATIO                                  .20
** RANDOM SRAN CBS WHICH HAVE BEEN CONVERTED:
      NUMBER                                   0
      PCT OF STOLEN RANDOM SRAN              .00

*****          DISTRIBUTION OF REFERENCES IN BUFFER-SETS          *****
----------------------------------------------------------------------
REFERENCE COUNT   NBR OF OCCURRENCES   PCT OF OCCURRENCES   ACCUMUL. PCT
              0                    3                  .46            .46
              1                  551                84.50          84.96
              2                    0                  .00          84.96
              3                    0                  .00          84.96
              4                    0                  .00          84.96
              5                    0                  .00          84.96
              6                    0                  .00          84.96
              7                    0                  .00          84.96
              8                    0                  .00          84.96
              9                    4                  .61          85.58
         =>  10                   94                14.41         100.00

*****        DISTRIBUTION OF REFERENCES IN RANDOM SRAN CBS        *****
----------------------------------------------------------------------
REFERENCE COUNT   NBR OF OCCURRENCES   PCT OF OCCURRENCES   ACCUMUL. PCT
              0                    0                  .00            .00
              1                    0                  .00            .00
              2                    1               100.00         100.00
```

This page can be used to evaluate the efficiency of the SB algorithm that monitored the I/O reference pattern for this SB buffer pool. By analyzing this page, you can answer these questions:

- How efficient were the decisions to issue sequential reads? This question can be answered by the REFERENCES IN BUFFER SETS field.
- How efficient were the decisions to issue random reads? This question can be answered by the REFERENCES IN RANDOM SRAN CBS field and the RANDOM SRAN CBS WHICH HAVE BEEN CONVERTED field.

The fields on Page B reflect buffering activity when SB was active. Buffering activity when IMS was only monitoring the I/O reference pattern is not included.

REFERENCES IN BUFFER SETS

This field shows you how many times blocks read with a sequential read were referenced by the OSAM buffer handler. This ratio is calculated as follows:

The number of references in buffer sets divided by the number of blocks read by sequential read.

For example, a ratio of 1.00 means that, on the average, each block read by a sequential read was referenced once. A ratio of 0.50 means that, on the average, only half the blocks read by a sequential read were referenced once.

In general, a high ratio (for example, 0.85) shows you that the decisions to issue sequential reads were efficient and probably helped reduce the execution time of the application. A low ratio (for example, 0.30) shows you that the benefit of issuing sequential reads was smaller because many of the blocks read were never referenced.

REFERENCES IN RANDOM SRAN CBS

This field shows you the effectiveness of the decisions to issue random reads. To understand what this ratio means, you must first understand how the SB buffer handler uses control blocks to track the I/O reference pattern.

The SB buffer handler uses a control block called SB range (SRAN) to track the I/O reference pattern within a range of 10 consecutive blocks. Each SRAN has a reference counter that shows how many times the OSAM buffer handler requested a block contained in the set of 10 blocks tracked by the SRAN.

There are two types of SRAN control blocks. One type, called a *sequential* SRAN, maintains counts for 10 consecutive blocks that have been read with a sequential read. A second type, called a *random* SRAN, maintains counts for 10 consecutive blocks that have been referenced in a random pattern. There is one sequential SRAN for each buffer set in an SB buffer pool; the number of random SRANs is twice the number of sequential SRANs.

The SRANs are chained together on a "use chain," that is, they are chained together in the order in which they have been most or least recently used. Each type of SRAN has its own use chain. Most recently used SRANs are at the top of the use chain; least recently used SRANs are at the bottom. Whenever a SRAN is needed to track a set of blocks that are not currently being tracked, the SB buffer handler selects a SRAN from the bottom of the use chain. This is because ranges of consecutive blocks that have not been referenced recently are less likely to be referenced again in the near future.

Each time a random SRAN is reused, the reference count maintained in the SRAN is recorded for later use. When the application program terminates, these reference counts are used to calculate the number in this field. The number is calculated as follows:

Ratio = X / (Y * 10)

where,

X

Total number of random references in random SRANs (this number is usually equal or close to the number of random reads in the sets of blocks tracked by random SRANs)

Y

Total number of reused random SRANs

A low ratio (for example, 0.15) in this field means that on the average, the SB buffer handler correctly recognized random I/O reference patterns. A ratio of 0.15 means that the SB buffer handler issued an average of 1.5 random reads for blocks tracked by a random SRAN. Issuing 1.5 random reads for a set of 10 consecutive blocks is normally more efficient then issuing one sequential read for the 10 consecutive blocks.

A high ratio (for example, 0.50) in this field probably indicates that the SB buffer handler often mistook sequential reference patterns as random reference patterns. A ratio of 0.50 means that the SB buffer handler issued an average of 5 random reads for blocks tracked by a random SRAN. Issuing 5 random reads for a set of 10 consecutive blocks is normally less efficient than issuing one sequential read for the 10 consecutive blocks.

RANDOM SRAN CBS WHICH HAVE BEEN CONVERTED

These fields are another measurement of the SB buffer handler's effectiveness in analyzing the I/O reference patterns.

Sometimes the SB buffer handler interprets a reference to a set of consecutive blocks as a random pattern, then, after some additional references, detects that the set of blocks is actually being referenced sequentially. In this case, the SB buffer handler first issues several random reads in a set of consecutive blocks and later issues a sequential read for the same set of blocks. When this occurs, the random SRAN that tracks the set of blocks is converted to a sequential SRAN. If the SB buffer handler had originally read the set of blocks with a sequential read, the cost of issuing several random reads would have been avoided.

The fields in this topic are:

NUMBER
This indicates how many times random SRANs were converted to sequential SRANs during the application program's execution.

PCT OF STOLEN RANDOM SRAN
The value in this field expresses the number of converted SRANs as a percentage of the number of times the SB buffer handler had to acquire a random SRAN from the use chain. A high percentage (for example, 40 percent) probably indicates that the SB buffer handler often mistook a sequential reference pattern as random reference pattern.

DISTRIBUTION OF REFERENCES IN BUFFER SETS
This table shows you more detailed information about the ratio reported in the REFERENCES IN BUFFER SETS field.

For example, the REFERENCES IN BUFFER SETS field might indicate that 80 percent (0.80) of the blocks in the SB buffer sets were referenced. What does this mean? It could mean that 20 percent of the time none of the blocks in a buffer set were referenced. Or instead, it could mean that 100 percent of the time only 8 out of 10 blocks in the buffer sets were referenced. You can answer this question by analyzing this table.

The fields in this table are:

REFERENCE COUNT
This column lists the number of references to blocks in a buffer set. A zero in this column means that no references were made to blocks in a buffer set, a one means that one reference was made, and so on. The "=> 10" in the last row of this column means that 10 or more references were made.

NBR OF OCCURRENCES
This column shows you how many buffer sets were referenced the number of times shown in the REFERENCE COUNT column. For example, in the figure, the first number in this column is 3, which says that 3 buffer sets were never referenced. The second number says that 551 buffer sets were referenced once.

PCT OF OCCURRENCES
This column shows you the value in the NBR OF OCCURRENCES column expressed as a percentage of the number of times buffer sets were reused. For example, in the figure, the first number in this column is 0.46, which says that 0.46 percent of the reused buffer sets were never referenced. The second number says that 84.50 percent of the reused buffer sets were referenced once.

ACCUMUL. PCT
This column shows you the accumulated PCT OF OCCURRENCES from zero through the current reference count. For example, in the figure, the third number in this column is 84.96, which says that 84.96 percent of the reused buffer sets were referenced two or less times.

DISTRIBUTION OF REFERENCES IN RANDOM SRAN CBS
This table shows you more detailed information about the ratio reported in the REFERENCES IN RANDOM SRAN CBS field. The table is similar to the DISTRIBUTION OF REFERENCES IN BUFFER SETS table, except that it shows information about references tracked by random SRAN control blocks.

REFERENCE COUNT
This column lists the number of references to blocks tracked by random SRANs. A zero in this column means that no references were made and, therefore, no random reads were issued for blocks tracked by a random SRAN. A "one" means that one reference was made to blocks tracked by a random SRAN (on the average, one reference for every tenth block tracked by the random SRAN).

NBR OF OCCURRENCES
This column shows you how many random SRANs were referenced the number of times shown in the REFERENCE COUNT column.

PCT OF OCCURRENCES
This column shows you the NBR OF OCCURRENCES expressed as a percentage of the number of times that random SRANs were reused.

ACCUMUL. PCT
This column shows you the accumulated PCT OF OCCURRENCES from zero through the current reference count.

Fields on page C of the Sequential Buffering Detail report

The following example shows page C of the report.

```
//DFSSTAT STATISTICS FOR: JOB=OSBTC01   STEP=STEP1   .          PGM=DFSDDLT0  PSB=PBVDSALR  DATE=93.058
TIME=09.39
----------------------------------------------------------------------------------------------------------
-------
                       *** SB DETAIL STATISTICS: INTERNAL COUNTERS AND VALUES    (PAGE C) ***
** DEACTIVATIONS:
      NBR OF SB-DEACTIVATION                                 1
      NBR OF MONITORING-DEACTIVATION                         0
** RESULTS OF EVALUATION OF SEQUENTIALITY:
      NBR POSITIVE RESULTS                                   3
      NBR NEGATIVE RESULTS                                   1
** RESULTS OF EVALUATION OF ACTIVITY RATE:
      NBR POSITIVE RESULTS                                   4
      NBR NEGATIVE RESULTS                                   0
** NBR RANDOM READ:
      DURING SEQUENTIAL BUFFERING PHASES                    27
      DURING "MONITORING-ONLY" PHASES                      659
      WHILE NOT MONITORING REFERENCE PATTERN                 0
** NBR RANDOM READS WITH SEQUENTIAL REFERENCE PATTERN:
      ACCESS TO INVALID BUFFERS                             25
      ACCESS AT DATA SET END                                 2
** NBR OF BUFFERING POSITIONS:                           2,213
** INTERNAL SB-ALGORITHM VALUES:
      SDSGBPTR: BLOCKS PER TRACK                            31
      SDSGNBRB: BLOCKS PER BUFSET                           10
      SDSGSCST: RELATIVE SEQ I/O COSTS                    1.36
      SDSGSINB: SIZE OF NEIGHBORHOOD                         2
      SDSGTHR1: THRESHOLD CURRENT+1                          2
      SDSGTHR2: THRESHOLD OVERLAP                            3
      SDSGTHR3: THRESHOLD NEIGBH                            11
```

Page C consists of the following topics:

DEACTIVATIONS
The fields in this topic are:

NBR OF SB DEACTIVATION
This field shows how many times SB was deactivated.

NBR OF MONITORING DEACTIVATION
This field shows how many times I/O reference monitoring was deactivated.

Deactivation of I/O reference monitoring can occur for one of two reasons:

- If several consecutive periodical evaluations of the buffering process show that it is not worthwhile to use SB
- If you set a limit on the SB buffer space by specifying the MAXSB keyword in the SBONLINE control statement. The use of SB is restricted when this limit is reached.

RESULTS OF EVALUATION OF SEQUENTIALITY
This topic and the next topic, RESULTS OF EVALUATION OF ACTIVITY RATE, help explain why SB was deactivated (or was not activated) during the application program's execution.

The decision to activate and deactivate SB is made by a periodical evaluation of the buffering process for a particular DB-PCB/DSG control block pair. The evaluation is based on the following criteria:

- Sequentiality
- Activity rate

If the results for both tests are positive (in other words, if the results of both tests recommend use of SB), IMS will activate SB (if not active) or will continue to use SB. If at least one of the test results is negative, then IMS will deactivate SB (if active) or will continue not to use SB. After several decisions not to use SB, IMS can also deactivate monitoring of the I/O reference pattern.

The fields in this topic are:

NBR POSITIVE RESULTS
This field shows you how many times periodical evaluation of the I/O reference pattern detected enough of a sequential reference pattern to warrant use of SB.

NBR NEGATIVE RESULTS
This field shows you how many times periodical evaluation of the I/O reference pattern did not detect enough of a sequential reference pattern to warrant use of SB.

RESULTS OF EVALUATION OF ACTIVITY RATE
The fields in this topic are:

NBR POSITIVE RESULTS
This shows you how many times periodical evaluation detected an I/O activity rate high enough to warrant use of SB.

NBR NEGATIVE RESULTS
This shows how many times periodical evaluation determined that the I/O activity rate was not high enough to warrant use of SB.

NBR RANDOM READ
This topic shows you how many random reads were issued during each of the following types of buffering phases:

DURING SEQUENTIAL BUFFERING PHASES
This field shows you how many random reads were issued while SB was active.

DURING "MONITORING ONLY" PHASES
This field shows you how many random reads were issued while SB was not active and IMS was still monitoring the I/O reference pattern.

WHILE NOT MONITORING REFERENCE PATTERN
This field shows you how many random reads were issued while SB was not active and IMS was not monitoring the I/O reference pattern.

NBR RANDOM READS WITH SEQUENTIAL REFERENCE PATTERN
This topic shows you how many times random reads were issued even though the I/O reference pattern was sequential. These counters are updated only when SB is active.

The fields in this topic are:

ACCESS TO INVALID BUFFERS
This field shows how many times a random read was issued because the contents of an SB buffer was invalid and could not be used. Some possible reasons for marking an SB buffer invalid are:

- Your IMS system is running in a block-level sharing environment. In a block-level sharing environment, more than one IMS system can read from and write to the same database. For example, if IMS system "A" reads a block into a buffer and IMS system "B" updates that block while the block is still in system "A's" buffer, system "A's" buffer will be marked as invalid.
- The activation and deactivation of SB during periodical evaluations marks SB buffers as invalid.
- A block was being written by another online application in the same IMS subsystem at the same time the SB buffer handler was reading it.
- The SB buffer was marked invalid because of I/O errors.

ACCESS AT DATA SET END
This field shows how many times the SB buffer handler issued a random read because the set of 10 blocks containing the referenced block was not completely formatted. SB never issues a sequential read at the end of a data set if the last set of consecutive blocks is not completely formatted.

NBR OF BUFFERING POSITIONS

This field shows how many times the SB buffer handler assumed the application program issued a DL/I call requesting a new position in the database. For example, most GU calls qualified on the key field of a root segment other than the current root segment will cause this counter to be incremented. This counter is maintained only while SB is active and IMS is monitoring the I/O reference pattern.

A high value in this field can indicate a large amount of logical random processing by the application. If other fields seem to indicate that the application did not benefit from SB, this field can explain why.

INTERNAL SB ALGORITHM VALUES

This topic shows the values of internal counters. They are included to help IMS development during SB problem determination.

Chapter 52. Statistical-analysis, log-transaction reports, and analyzing log records

IMS provides several utilities that extract data from IMS system logs.

- The Statistical Analysis utility produces summary reports of message activity, reports for lines and terminals, and the Message Select and List Report.
- The Log Transaction Analysis utility gives detailed information of individual transaction and processing activities.

Statistical analysis utility reports

The input data for the Statistics Analysis utility is a set of IMS log data sets, or user data sets created by the Log Archive utility.

Each input data set can consist of multiple volumes. Several data sets can be concatenated and you can include data sets from multiple IMS systems connected through shared queues or MSC.

Restriction: You cannot use system log output from a batch system.

You can use the Transaction Analysis utility to obtain new system logs with reduced content to save processing time.

The Statistical Analysis utility has six control statements you can use to select a subset of transaction activity.

Transaction code control statement
: You can use this control statement to select a specific transaction code or groups of transaction codes.

Symbolic terminal name control statement
: You can use this control statement to specify the LTERM name or a generic name. For example, L3270M selects all messages originating from or directed to that LTERM. A generic name of L3270* could select L3270M and L3270B messages, the comparison being based on the characters preceding the *.

: You can further qualify the output LTERM so that only messages to a given symbolic name resulting from the input LTERM specified are selected.

Time control statement
: You can specify an interval as a criterion for selection. You give the start and stop times in the form YYDDD and HHMM (Julian day and clock time in minutes). This range criterion is applied to all messages selected by transaction code and terminal specifications.

Message select output order statement
: Use this control statement to determine which order to list messages. This control statement affects only the content of the Message Select and List or Copy output.

Nonprintable character control statement
: If you anticipate non-printable characters in the message text, you can specify they be printed in hexadecimal format (first character above the second). Otherwise, the characters appear as blanks.

For more information about the Statistical Analysis utility (DFSISTS0), see *IMS Version 15.2 System Utilities*.

Calculating transaction loads

There are two reports produced by the Statistical Analysis utility that summarize the distribution of transaction activity across a 24-hour day. Input and output message distributions are separately tabulated for each transaction code and for each device.

A further report shows the response times for each transaction type expressed as percentiles. The reported data is dependent on the selection of system log data that makes up the utility input. The scope of the report can be further limited by selecting a subset of transactions and line traffic as well as the reporting interval.

The following figure shows the format of the Line and Terminal report. For each device on a line the LTERM name is given and a pair of rows of results for send and receive activity is given. The "Total Messages" column is followed by the total and average size of the messages in bytes. A series of hourly intervals divides the 24-hour day and counts of the transaction active in those intervals are recorded.

```
         L I N E   A N D   T E R M I N A L   R E P O R T                  D A T E  06/05/07                      P A G
E 00002
               TOTAL        TOTAL          AVG              HOURLY        DISTRIBUTION
 NODE    R/S   MESSAGES    CHARACTERS      SIZE  00-07 07-08 08-09 09-10 10-11 11-12 12-13 13-14 14-15 15-16 16-17
17-18 18-19 19-24
CTRL     R            5           782       156     0     0     0     0     0     5     0     0     0     0     0
0     0     0
         S           10         1,123       112     0     0     0     0     0    10     0     0     0     0     0
0     0     0
CTRLA01  R            3           365       121     0     0     0     0     0     3     0     0     0     0     0
0     0     0
         S            2           276       138     0     0     0     0     0     2     0     0     0     0     0
0     0     0
CTRLA02  R            2           186        93     0     0     0     0     0     2     0     0     0     0     0
0     0     0
         S            4           306        76     0     0     0     0     0     4     0     0     0     0     0
0     0     0
L62MVS1  R            9           366        40     0     0     0     0     0     9     0     0     0     0     0
0     0     0
         S            9           483        53     0     0     0     0     0     9     0     0     0     0     0
0     0     0
SEGUNDO  S           28         1,842        65     0     0     0     0     0    28     0     0     0     0     0
0     0     0
 SYSTEM  R           19         1,699        89     0     0     0     0     0    19     0     0     0     0     0
0     0     0
 TOTALS  S           53         4,030        76     0     0     0     0     0    53     0     0     0     0     0
0     0     0
```

Key: R/S—Received/Sent

Entries for devices restricted to input only or output only traffic show only one reporting line. The following example shows the format of the Transaction report. This organizes the data like the Line and Terminal report, except that it is ordered by transaction code.

```
         T R A N S A C T I O N   R E P O R T                              D A T E  06/05/07                      P A G
E 00003
TRANSACTION    TOTAL        TOTAL          AVG              HOURLY        DISTRIBUTION
   CODE  R/S   MESSAGES    CHARACTERS      SIZE  00-07 07-08 08-09 09-10 10-11 11-12 12-13 13-14 14-15 15-16 16-17
17-18 18-19 19-24
CONV21V0 R            5           246        49     0     0     0     0     0     5     0     0     0     0     0
0     0     0
         S            4           298        74     0     0     0     0     0     4     0     0     0     0     0
0     0     0
SMQR21C0 R            3            90        30     0     0     0     0     0     3     0     0     0     0     0
0     0     0
         S            2            50        25     0     0     0     0     0     2     0     0     0     0     0
0     0     0
TRAN21V0 R            1            30        30     0     0     0     0     0     1     0     0     0     0     0
0     0     0
         S            1            25        25     0     0     0     0     0     1     0     0     0     0     0
0     0     0
 SYSTEM  R            9           366        40     0     0     0     0     0     9     0     0     0     0     0
0     0     0
 TOTALS  S            7           373        53     0     0     0     0     0     7     0     0     0     0     0
0     0     0
```

The following example shows the format of the Transaction report. This report gives the longest and shortest response times for each transaction code for the data selected as input from a set of system logs. Four columns record the percentile response times in seconds. The 25th, 50th, 75th, and 95th percentiles are given. For example, a response time within the 50th percentile is greater than or equal to 50% of the total number of response times processed for that transaction. The first line of response times given is from the completion of the receipt of the input message until the response message is successfully dequeued. In the event that an output message takes a significantly long time to be completely received at the terminal, a second line shows the receipt to the time the response message is started.

```
     T R A N S A C T I O N   R E S P O N S E   R E P O R T       D A T E  06/05/07            P A G
E 00004
TRANSACTION      TOTAL          LONGEST          95%            75%            50%
25%         SHORTEST
   CODE        RESPONSES       RESPONSE       RESPONSE       RESPONSE       RESPONSE
RESPONSE       RESPONSE
  CONV21V0            4            .36S           .31S           .31S           .24S
.19S           .19S
                    4          .36S           .31S           .31S           .24S
.19S           .19S
  SMQR21C0            2            .51S           .33S           .33S           .33S
.33S           .33S
                    2          .51S           .33S           .33S           .33S
.33S           .33S
  TRAN21V0            1            .36S           .36S           .36S           .36S
.36S           .36S
                    1          .36S           .36S           .36S           .36S
.36S           .36S
```

Assessing program-to-program traffic

When a message processing program directs an output message to another program, that secondary transaction is queued. The transaction code is sometimes unique for convenience of the processing program's logic.

Otherwise, the secondary transaction is queued along with any messages from terminal origin.

You can use the two Messages—Program-to-Program reports to separately count transaction traffic. The following example illustrates the two tabulations. The column headed "Destination" appears above a list of transaction codes that were queued to another program. The originating program is not identified. The column headed "Transaction Code" appears above a list of the initial transaction codes that invoked the programs that issued the secondary transactions.

If you had program-to-program switches during conversational transaction processing, these will be included in the lists.

```
        MESSAGES--PROGRAM TO PROGRAM        DATE 06/05/07

                    TOTAL
    DESTINATION     MESSAGES

    ELEANOR             1
    SW1050              1
    T2741N1             1
    T2742N3             1

        MESSAGES--PROGRAM TO PROGRAM        DATE 06/05/07

    TRANSACTION     TOTAL
    CODE            MESSAGES

    TA10                107
```

Obtaining counts of unsent messages

The two reports titled Messages—Queued-But-Not-Sent summarize how many output messages were still in the message queues for interval covered by the input tapes.

The reports are illustrated in the following example. Command responses that were not sent to the terminal are indicated by (IMSSYS). The entry of NOTAVA indicates "no transaction available". This would be the case if an output message were generated for an input not recorded in the system log input data or by a command input from the same terminal not recorded.

```
       MESSAGES - QUEUED  BUT  NOT  SENT     DATE  06/05/07    PAGE 00001
                    TOTAL
DESTINATION      MESSAGES
CTRLA01                 1
SEGUNDO                37
```

Auditing critical transactions

You can use the optional Messages report produced by the Statistical Analysis utility with the DFSIST40 program to examine the input and output data for specific transaction codes in detail. This allows you to audit exactly what was in an input message and possibly examine the output content for errors.

The report is illustrated in the following example.

```
            MESSAGES

INPUT  SEG=001 LEN=020*  CONV12V0 Y  c
*
INPUT  SEG=002 LEN=023*MESSAGE TO CONV12V0  >
*
PGM SW SEG=001 LEN=066*  CONV21V0 Y     THIS DATA INSERTED TO THE SPA OF CONV21V0  c
*
PGM SW SEG=002 LEN=042*MESSAGE TO CONV21V0                     >
*
OUTPUT SEG=001 LEN=066*           Y       THIS TERMINATEED TO THE SPA OF CONV21V0  c
*
OUTPUT SEG=002 LEN=035*RESPONSE TO INPUT TERMINAL       >
*
INPUT   TRANSACTION                 SEQ                                   OUTPUT
SEQ
PREFIX     CODE       LTERM         NO            DATE     TIME        PREFIX   LTERM
NO            DATE     TIME
         CONV12V0     TNDSE        00001         07.053  12.37.18               TNDSE
00001         07.053  12.37.19
```

Log transaction analysis utility reports

You can obtain detailed data at the individual transaction level by using the Log Transaction Analysis utility (DFSILTA0). Although the data is not summarized by this utility, the detail report lines bring together many information items that help you assess the service given to a transaction type and the effect of the scheduling algorithm. The report shows actual response data because input data is the IMS log.

If you do not process the entire IMS log, data is presented from a starting checkpoint to a cutoff point. You limit the sample of transaction processing to be analyzed by specifying start time and duration in minutes, or you can give the number of checkpoints to be included after the starting checkpoint. Canceled messages are omitted.

The format of the Log-Analysis report and the full list of data items for each report detail line are shown in "Examining scheduling activity" on page 697. Times are given to the nearest tenth of a second and are elapsed times. You can see that the Processing Types field is a key description item.

Using the starting position and lengths of the fields in the report detail records you can specify a sort order for the second step in the utility execution.

The sort control statement to cause a report to be sequenced by message class and transaction priority is:

```
SORT FIELDS = (18,3,CH,A,16,1,CH,A)
```

You can also use the option of creating a DASD data set of the detail report records. Your installation could then develop an analysis program to extract and summarize the data.

Examining scheduling activity

Using the data extracted by the Log Transaction Analysis utility you can examine the effect of your scheduling algorithm.

Each occurrence of a transaction primarily indicates:

- Message priority and message class
- Time in input queue
- Time to process
- Time in output queue
- Total time-in-system (measured between completion of message queue input to retrieval for output)

You can look at the processing type 'S' entries to see the send and receive times. You can sort the detail report lines by transaction code and look at any critical transactions requiring rapid response time. Refer to the following example for details on each table line item.

```
12.50.11 JOB00386  $HASP373 DFSILTA4 STARTED - INIT 4    - CLASS K - SYS STL1
12.50.12 JOB00386  SMF000I  DFSILTA4    ILTA        DFSILTA0    0000
12.50.12 JOB00386  $HASP395 DFSILTA4 ENDED
------ JES2 JOB STATISTICS ------
  05 JUN 2007 JOB EXECUTION DATE
           30 CARDS READ
          143 SYSOUT PRINT RECORDS
            0 SYSOUT PUNCH RECORDS
           10 SYSOUT SPOOL KBYTES
         0.01 MINUTES EXECUTION TIME
        1 //DFSILTA4 JOB 'TERRY',CLASS=K,MSGCLASS=A,MSGLEVEL=(1,1),               JOB00386
          // REGION=0M,TIME=1440,
          // USER=USRID01,PASSWORD=
          /*ROUTE PRINT THISCPU/IMSTST45
        2 //JOBLIB   DD DSN=IMSTESTL.TNUC0,DISP=SHR
        3 //         DD DSN=IMSBLD.I10RTS17.CRESLIB,DISP=SHR
        4 //ILTA     EXEC PGM=DFSILTA0
        5 //HEADING  DD SYSOUT=A
        6 //PRINTER  DD SYSOUT=A
        7 //SYSUDUMP DD SYSOUT=A
        8 //REPORT   DD DUMMY
        9 //TITLE    DD *
          //*
          //* LOG INPUT FOLLOWS...
          //*
       10 //LOGIN0A  DD DSN=IMSTESTL.SLDSP.IMSA.TYTY,DISP=SHR,
          //         UNIT=SYSDA,VOL=SER=DSHR03
       11 //LOGIN01  DD DSN=IMSTESTL.SLDSP.IMS1.TYTY,DISP=SHR,
          //         UNIT=SYSDA,VOL=SER=DSHR03
       12 //LOGIN02  DD DSN=IMSTESTL.SLDSP.IMS2.TYTY,DISP=SHR,
          //         UNIT=SYSDA,VOL=SER=DSHR03
       13 //LOGIN03  DD DSN=IMSTESTL.SLDSP.IMS3.TYTY,DISP=SHR,
          //         UNIT=SYSDA,VOL=SER=DSHR03
ICH70001I USRID01  LAST ACCESS AT 12:50:09 ON TUESDAY, JUNE 5, 2007
IEF236I ALLOC. FOR DFSILTA4 ILTA
IEF237I 04A1 ALLOCATED TO JOBLIB
IEF237I 04B4 ALLOCATED TO
IEF237I JES2 ALLOCATED TO HEADING
IEF237I JES2 ALLOCATED TO PRINTER
IEF237I JES2 ALLOCATED TO SYSUDUMP
IEF237I DMY  ALLOCATED TO REPORT
IEF237I JES2 ALLOCATED TO TITLE
IEF237I 04A9 ALLOCATED TO LOGIN0A
IEF237I 04A9 ALLOCATED TO LOGIN01
IEF237I 04A9 ALLOCATED TO LOGIN02
IEF237I 04A9 ALLOCATED TO LOGIN03
IEF237I 04A9 ALLOCATED TO LOGIN04
IEF237I 04A9 ALLOCATED TO LOGIN05
IEF237I 04A9 ALLOCATED TO LOGIN036
IEF142I DFSILTA4 ILTA - STEP WAS EXECUTED - COND CODE 0000
IEF285I   USRID01.DFSILTA4.JOB00386.D0000102.?         SYSOUT
IEF285I   USRID01.DFSILTA4.JOB00386.D0000103.?         SYSOUT
IEF285I   USRID01.DFSILTA4.JOB00386.D0000104.?         SYSOUT
IEF285I   USRID01.DFSILTA4.JOB00386.D0000101.?         SYSIN
IEF285I   IMSTESTL.SLDSP.IMSA.TYTY                     KEPT
IEF285I   VOL SER NOS= DSHR03.
IEF285I   IMSTESTL.SLDSP.IMS1.TYTY                     KEPT
IEF285I   VOL SER NOS= DSHR03.
IEF285I   IMSTESTL.SLDSP.IMS2.TYTY                     KEPT
IEF285I   VOL SER NOS= DSHR03.
IEF285I   IMSTESTL.SLDSP.IMS3.TYTY                     KEPT
IEF285I   VOL SER NOS= DSHR03.
IEF285I   IMSTESTL.SLDSP2.IMS1.TYTY                    KEPT
IEF285I   VOL SER NOS= DSHR03.
IEF285I   IMSTESTL.SLDSP2.IMS2.TYTY                    KEPT
IEF285I   VOL SER NOS= DSHR03.
```

```
IEF285I   IMSTESTL.SLDSP2.IMS3.TYTY                    KEPT
IEF285I   VOL SER NOS= DSHR03.
IEF373I STEP/ILTA    /START 2007156.1250
IEF374I STEP/ILTA    /STOP  2007156.1250 CPU    0MIN 00.02SEC SRB    0MIN 00.01SEC VIRT    16K SYS   296K EXT     808K
SYS   11924K
IEF285I   IMSTESTL.TNUC0                               KEPT
IEF285I   VOL SER NOS= USER01.
IEF285I   IMSBLD.I10RTS17.CRESLIB                      KEPT
IEF285I   VOL SER NOS= MVSS16.
IEF375I  JOB/DFSILTA4/START 2007156.1250
IEF376I  JOB/DFSILTA4/STOP  2007156.1250 CPU    0MIN 00.02SEC SRB    0MIN 00.01SEC
 SPECIFIED START TIME IS 16:53:47.3
 TO END OF FILE PROCESSED 9 TRANSACTIONS THIS RUN
 REPORT CONTAINS 14 COMPLETE RECORD SETS
 IMS LOG DATA FOR IMS3 STARTS AT TIME = 12:47:41.8, DATE = 2007.156
     JOB      STEP    DR     CLASSES
     NAME     NAME    ID ***************
   MPP1     MPP        1   1   2   3   4
   MPP2     MPP        1   1   2   3   4
   MPP3     MPP        1   1   2   3   4
 IMS LOG DATA FOR IMS2 STARTS AT TIME = 12:46:39.5, DATE = 2007.156
 IMS LOG DATA FOR IMS1 STARTS AT TIME = 12:34:26.8, DATE = 2007.156
 IMS LOG DATA FOR IMSA STARTS AT TIME = 12:36:56.8, DATE = 2007.156
 SEQ    TRANS  P   C  ***IN***   ***OUT**  P   PGM     DR SMB*ENQ  MSG*SCHD CNT*ENQ  MSG*END  CNT*GU  SYS IN Q   PROC
OUT Q  TOTAL
 NBR    CODE   R   L  LTERM      LTERM      T   NAME    ID HHMMSST  HHMMSST  HHMMSST  HHMMSST  HHMMSST  ID SSSST  SSSST
SSSST  SSSST
------------------------------------------------------------------------------------------------------------------------
------------
00001  CHKPT   0001************************               1234270  1234270  1234270  1234270  1234270 1
00002  CHKPT   0001************************               1236569  1236569  1236569  1236569  1236569 A
00003  CHKPT   0001************************               1239165  1239165  1239165  1239165  1239165 2
00004  CHKPT   0001************************               1240111  1240111  1240111  1240111  1240111 3
00005                 CTRLA02    CTRLA02    M                                1241134           1241135 2
2                   1
00006                 CTRL       CTRL       M                                1241260           1241263 3
3                   2
00007                 CTRLA02    CTRLA02    M                                1241398           1241401 2
2                   2
00008                 CTRL       CTRL       M                                1241523           1241526 3
3                   2
00009                 CTRL       CTRL       M                                1242176           1242179 3
3                   3
00010                 CTRL       CTRL       M                                1242240           1242244 3
3                   3
00011 TRAN21V0 1   1  L62MVS1    L62MVS1   S PGM2V0     1 1242411  1242411  1242413  1242413  1242414 131     0
2      1      3
00012 SMQR21C0 1   1  L62MVS1    L62MVS1   S PGM2C0     1 1242419  1242419  1242421  1242421  1242424 131     0
1      3      5
00013                 CTRLA01    CTRLA01    M                                1242582           1242585 1
1                   2
00014                 CTRLA01    CTRLA01    M                                1243136           1243138 1
1                   1
00015 SMQR21C0 1   1  L62MVS1    L62MVS1   S PGM2C0     1 1243147  1243147  1243149  1243149  1243150 131     0
1      1      3
00016 SMQR21C0 1   1  L62MVS1              T PGM2C0     1 1243279  1243279           1243281           13      0
1             2
00017                 CTRL       CTRL       M                                1243450           1243453 3
3                   2
00018 CONV21V0 1   1  L62MVS1              D CPGM2V0    1 1243498  1243499           1243506           13      0
7             8
00019 CONV21V0 1   1  L62MVS1    L62MVS1   C CPGM2V0    1 1243545  1243545  1243546  1243546  1243546 133     0
0      0 ******
00020 CONV21V0 1   1  L62MVS1    L62MVS1   C CPGM2V0    1 1243577  1243577  1243577  1243577  1243577 133     0
0      0 ******
00021 CONV21V0 1   1  L62MVS1    L62MVS1   C CPGM2V0    1 1244049  1244049  1244049  1244049  1244049 133     0
0      0 ******
00022 CONV21V0 1   1  L62MVS1    L62MVS1   C CPGM2V0    1 1244080  1244080  1244081  1244081  1244081 133     0
0      0 ******
00026  CHKPT   0002************************               1248342  1248342  1248342  1248342  1248342 A
00027  CHKPT   0002************************               1249068  1249068  1249068  1249068  1249068 1
00028  CHKPT   0002************************               1249365  1249365  1249365  1249365  1249365 2
00029  CHKPT   0002************************               1250062  1250062  1250062  1250062  1250062 3
```

Table 81. Log Analysis report line format

Identification	Starting position	Length	Note
Sequence Number	1	5	1
Transaction Code	7	8	
Priority of Transaction (PR)	16	1	
Class of Transaction (CL)	18	3	

Table 81. Log Analysis report line format (continued)

Identification	**Starting position**	**Length**	**Note**
Input LTERM name	22	8	
Output LTERM name	33	8	
Processing Type (PT)	44	1	2
Program Name	46	8	
Dependent Region ID	55	3	
Time of SMB Enqueue (Transaction received)	59	7	3
Time of Message Schedule or GU	68	7	3
Time of CNT Enqueue (Message put on output queue)	77	7	3
Time of Program End or Next Message GU	86	7	3
Time of CNT GU (Output message starts to terminal)	95	7	3
System IDs (from LOGINxxx DD statement)	103	3	
Time in Input Queue	106	6	4, 5
Time Processing	113	6	5, 6
Time in Output Queue	120	6	5, 7
Total Time	127	6	5, 8

Key to the table:

1. Starting position 1 is a carriage control character which alters the starting positions of the fields when producing a report on disk.
2. Processing types:

   A
   Abended transaction

   C
   Conversational Send/Receive Processing

   D
   Transmit Only Conversational Processing

   F
   /FORMAT entered (Transaction Code Field has MODNAME)

   M
   Message Switch

   O
   Region Occupancy (A region is occupied by a program that is processing transactions that existed in the input queue before the start checkpoint has encountered or a program scheduled by an unrecoverable message.)

   P
   Program Switch Send/Receive Processing

   Q
   Transmit Only Program Switch Processing

   S
   Send/Receive Processing

T
: Transmit Only Processing

X
: Conversational Program Switch, Send/Receive Processing

Y
: Transmit Only Conversational Program Switch Processing

3. Time HHMMSST
4. Input queue time is from SMB enqueue to message schedule.
5. Time SSSST or OVRFLW (If the total seconds exceeds the field size, OVRFLW is printed).
6. If the wait-for-input (WFI) system option is used, the time processing field also includes the wait time between transactions.
7. Output queue time is from CNT enqueue to CNT GU.
8. Total Time is from SMB enqueue to CNT dequeue. The total time spans the complete transaction.

IMS accounting information

The nature of accounting methods varies a great deal among data processing installations. The IMS Transaction Manager presents special difficulties, because many and varied transactions are processed by a partnership of the control region and dependent regions.

Further, operationally discrete applications can be served concurrently.

For installations with IMS-dedicated processors, the overall cost of hardware and support functions is often charged back based on predicted and actual use by contributing groups. For shared systems, the processor usage can be the base for proportional cost.

Although IMS does not have an explicit accounting function, the individual events that make up the processing activity are recorded on the IMS log in considerable detail. Analysis of IMS log records can be used as a basis for charge-back algorithms. You can, for example, obtain a report from the Statistical Analysis utility of the number of transactions and the average number of DL/I calls for each transaction.

Another source of resource usage figures is the reports produced as a result of IMS Monitor data collection. Samples of processing activity can be taken on a regular basis. Accounting algorithms can use, for example, processor utilization by program.

Both of these approaches require further manipulation and calibration of the resource indicators.

If the DL/I address space option is used (LSO=S), accounting procedures based on SMF data will be affected. SMF statistics for IMS system data sets and Fast Path databases are accounted to the control region procedure. Full function databases are accounted to the DL/I address space procedure.

Using the Application Accounting report

The Statistical Analysis utility produces an Application-Accounting report that you can use to assess machine charges. The following breakdown is provided for each transaction and for each program:

- The number of messages with related total and average processor time in seconds
- The number and type of DL/I message calls
- The number and type of DL/I database calls

The following example illustrates the output format.

```
         A P P L I C A T I O N   A C C O U N T I N G   R E P O R T        D A T E  06/05/07                      P A G
E 00005
PROGRAM TRANSACTION    MESSAGE- - - - COUNTS    DATA - - - - - - - - - - BASE - - - - - - - - - COUNTS  CC OR RC    TOT
PROG      AVG
 NAME      CODE   PRI  QTY    GU    GN  ISRT    GU    GN   GNP   GHU   GHN  GHNP  ISRT  DLET  REPL   NOT 0       CPU
TIME      TIME
CPGM2V0  CONV21V0  1     5     5    10     9     0     0     0     0     0     0     0     0     0     0
0.0S      0.000S
PGM2C0   SMQR21C0  1     3     3     3     2     0     0     0     0     0     0     0     0     0     0
0.0S      0.000S
```

```
PGM2V0   TRAN21V0  1     1     1     1     1     0     0     0     0     0     0     0     0     0     0
0.0S      0.000S
ICE143I
```

Using IMS transaction profiles

You can use a composite picture of each transaction as a basis for estimating usage. The IMS Transaction profile can contain DL/I call requirements by type of call and possibly items derived from path length for other processing blocks. You can weight the message count to allow for heavy DL/I use by the transaction. Transaction statistics can be obtained on a regular basis from **/DISPLAY** output, for example at end-of-day or before shutdown.

The profiles should characterize the IMS workload in such a way that growth trends and major deviations from the predicted load can be traced to the transaction codes responsible.

Notices

Programming interface information

This information documents Product-sensitive Programming Interface and Associated Guidance Information and General-use Programming Interface and Associated Guidance Information.

Product-sensitive Programming Interfaces allow the customer installation to perform tasks such as diagnosing, modifying, monitoring, repairing, tailoring, or tuning of this software product. Use of such interfaces creates dependencies on the detailed design or implementation of the IBM software product. Product-sensitive Programming Interfaces should be used only for these specialized purposes. Because of their dependencies on detailed design and implementation, it is to be expected that programs written to such interfaces may need to be changed in order to run with new product releases or versions, or as a result of service. Product-sensitive Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a section or topic, or by a Product-sensitive programming interface label. IBM requires that the preceding statement, and any statement in this information that refers to the preceding statement, be included in any whole or partial copy made of the information described by such a statement.

General-use programming interfaces allow the customer to write programs that obtain the services of IMS. General-use Programming Interface and Associated Guidance Information is identified where it occurs, either by an introductory statement to a section or topic or by a General-use programming interface label.

Trademarks

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Terms and conditions for product documentation

Permissions for the use of these publications are granted subject to the following terms and conditions.

Applicability

These terms and conditions are in addition to any terms of use for the IBM website.

Personal use

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

Commercial use

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

IBM Online Privacy Statement

IBM Software products, including software as a service solutions, (“Software Offerings”) may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering’s use of cookies is set forth below.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

To learn more, see IBM Privacy Statement.

Bibliography

This bibliography lists all of the publications in the IMS 15.2 library.

Title	**Acronym**
IMS Version 15.2 Application Programming	APG
IMS Version 15.2 Application Programming APIs	APR
IMS Version 15.2 Commands, Volume 1: IMS Commands A-M	CR1
IMS Version 15.2 Commands, Volume 2: IMS Commands N-V	CR2
IMS Version 15.2 Commands, Volume 3: IMS Component and z/OS Commands	CR3
IMS Version 15.2 Communications and Connections	CCG
IMS Version 15.2 Database Administration	DAG
IMS Version 15.2 Database Utilities	DUR
IMS Version 15.2 Diagnosis	DGR
IMS Version 15.2 Exit Routines	ERR
IMS Version 15.2 Installation	INS
IMS Version 15.2 Licensed Program Specifications	LPS
IMS Version 15.2 Messages and Codes, Volume 1: DFS Messages	MC1
IMS Version 15.2 Messages and Codes, Volume 2: Non-DFS Messages	MC2
IMS Version 15.2 Messages and Codes, Volume 3: IMS Abend Codes	MC3
IMS Version 15.2 Messages and Codes, Volume 4: IMS Component Codes	MC4
IMS Version 15.2 Operations and Automation	OAG
IMS Version 15.2 Release Planning	RPG
IMS Version 15.2 System Administration	SAG
IMS Version 15.2 System Definition	SDG
IMS Version 15.2 System Programming APIs	SPR
IMS Version 15.2 System Utilities	SUR

Index

Product Number:	5635-A06
	5655-DS5
	5655-TM4