© 2013 IBM Corporation
IBM Advanced Technical Skills

WBSR85

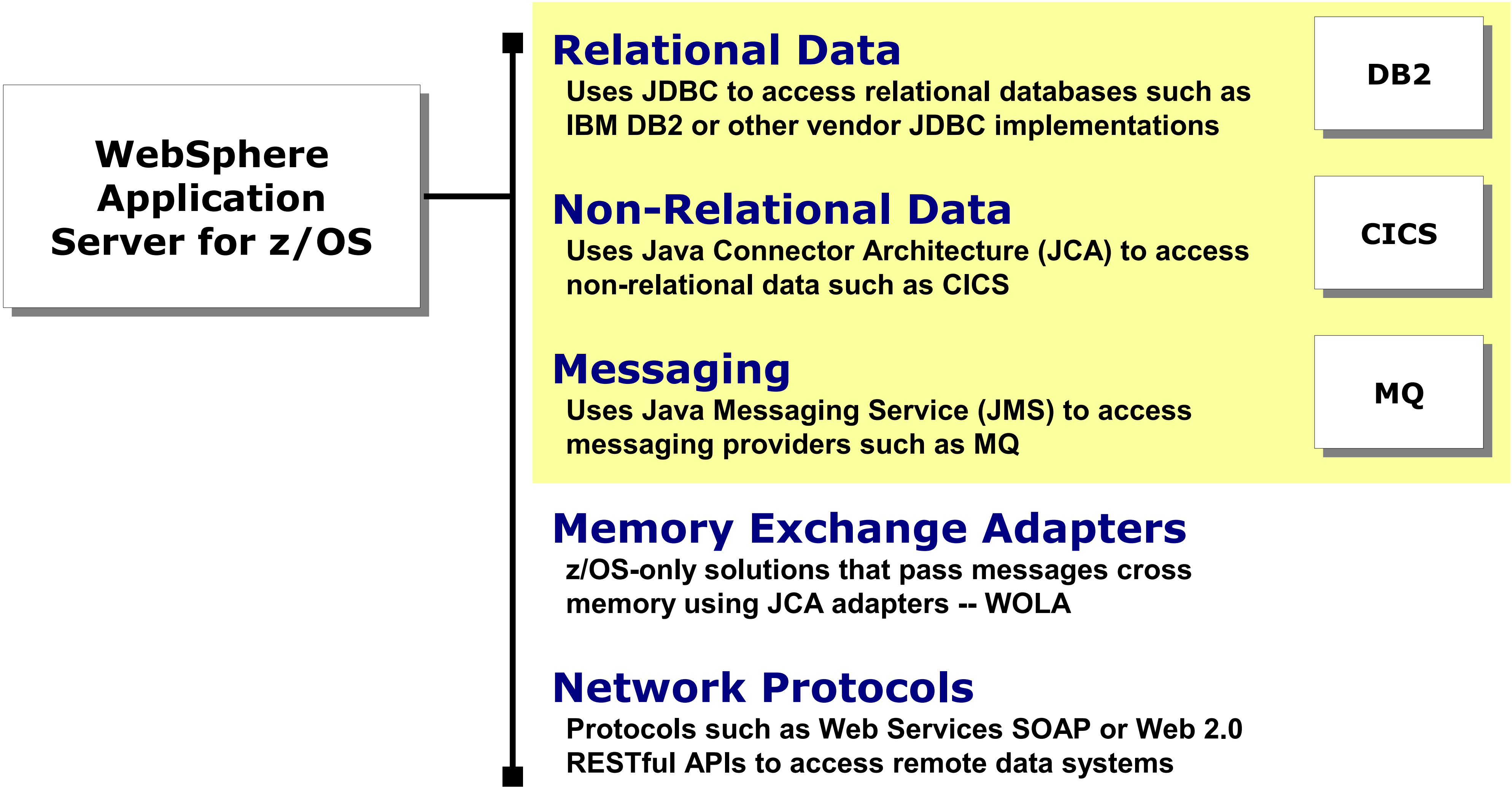
WebSphere Application Server z/OS V8.5

Unit 4 - Accessing z/OS Data

This page intentionally left blank

High Level of Data Access Approaches with WAS z/OS

There are five categories of data access approaches. We'll cover three in this unit and one in the next unit. The fifth we'll leave to other workshops:



WebSphere Application Server for z/OS

Relational Data

Uses JDBC to access relational databases such as IBM DB2 or other vendor JDBC implementations

DB2

Non-Relational Data

Uses Java Connector Architecture (JCA) to access non-relational data such as CICS

CICS

Messaging

Uses Java Messaging Service (JMS) to access messaging providers such as MQ

MQ

Memory Exchange Adapters

z/OS-only solutions that pass messages cross memory using JCA adapters -- WOLA

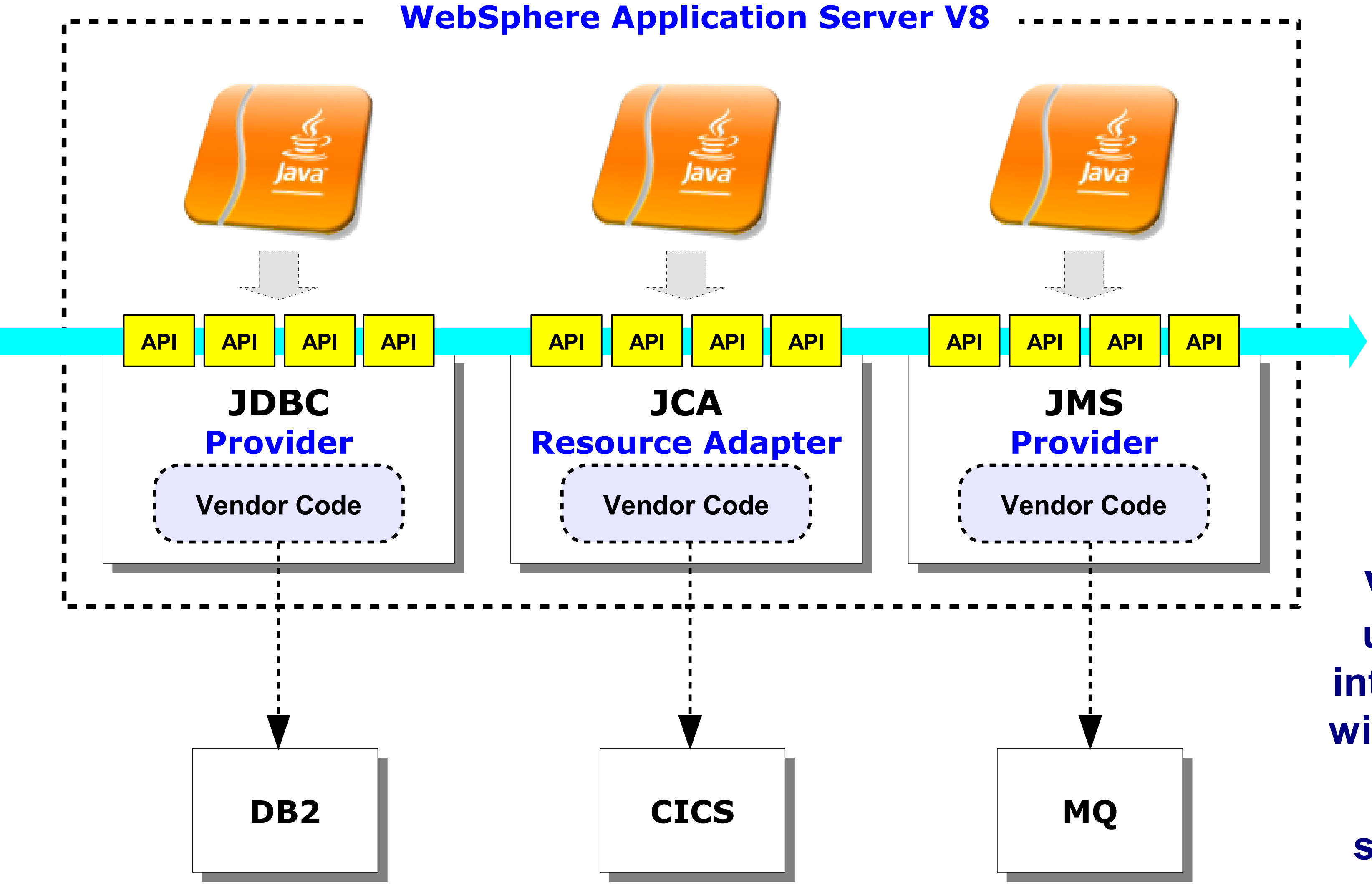
Network Protocols

Protocols such as Web Services SOAP or Web 2.0 RESTful APIs to access remote data systems

Data abstraction and open standards …

Data Abstraction Behind Open Standard Interfaces

The data access approaches all share a common theme -- hiding data subsystem specifics behind standard APIs, with installable code to provide lower level access:



In V8:

JDBC 4.0

JCA 1.6

JMS 1.1

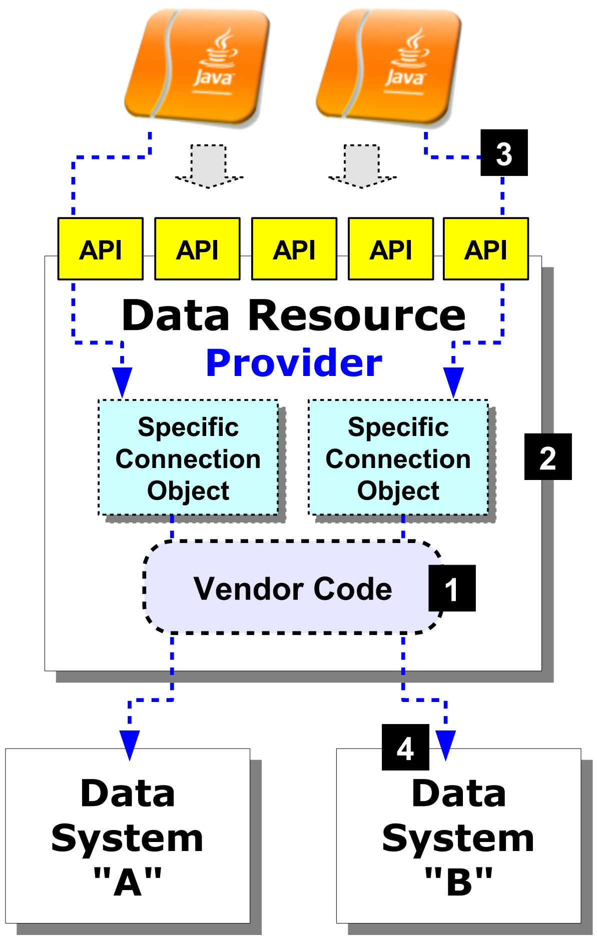
Vendor code has understanding of interaction specifics with the data system

Applications are shielded from this

Connection specifics …

Another Common Theme - Connection Specifics

The "Provider" supplies the vendor code that understands how to work with the data system. Another component is needed - something to tell which data system to talk to:



1. The provider supplies the code that interacts with the specific data resource, as well as a framework for creating specific connection objects
2. The specific connection objects provide details about which data system to connect to and any name, port or other details required
3. Application do a JNDI lookup of the specific connection object
4. Then using that connection they access the data system named in the specific connection object

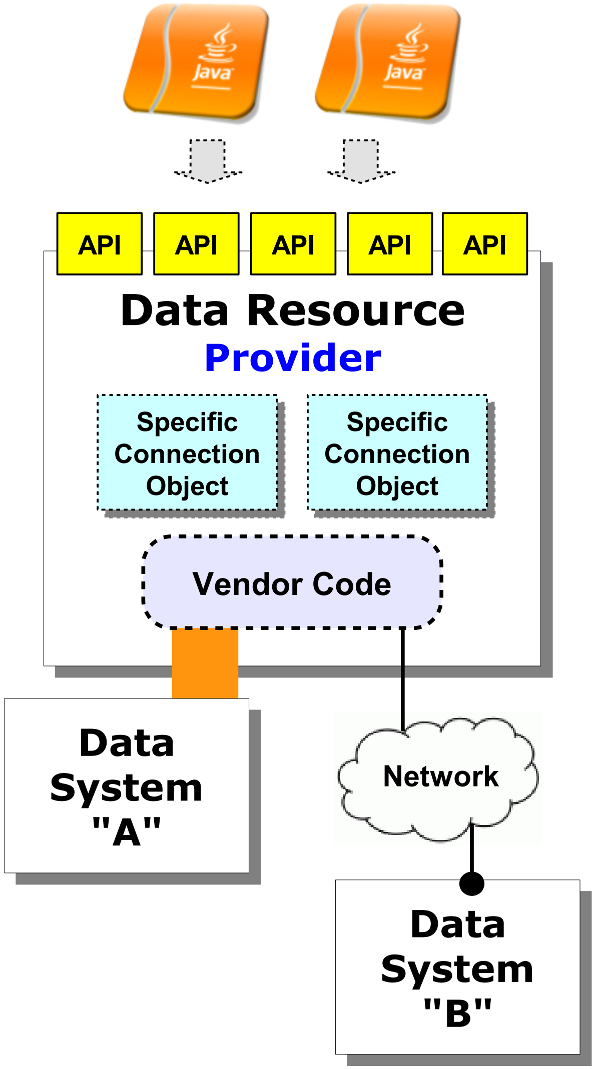
IBM Data System	Name used to refer to the specific connection object in WAS
DB2	"Data Source"
CICS	"Connection Factory"
JMS	"Connection Factory"

Different names … same concept

Local vs. remote connections …

z/OS Theme - Choice of Local or Remote Connection

On z/OS the specific connection details allow for two types of connectivity -- local, which is a cross-memory connection, or remote, which uses TCP/IP:



Cross-Memory Connection

Uses z/OS cross memory services to access the data system:

- DB2 - Type 2 JDBC
- CICS - EXCI
- MQ - Bindings Mode
- Involves Java *and native code execution*
  Which means configuration will involve pointing to native libraries

Network Connection

Accesses the data system via the network and an exchange protocol mapped on TCP/IP:

- DB2 - Type 4 JDBC
- CICS - CTG Gateway or IPIC
- MQ - Client Mode
- Involves Java execution only

Relational …

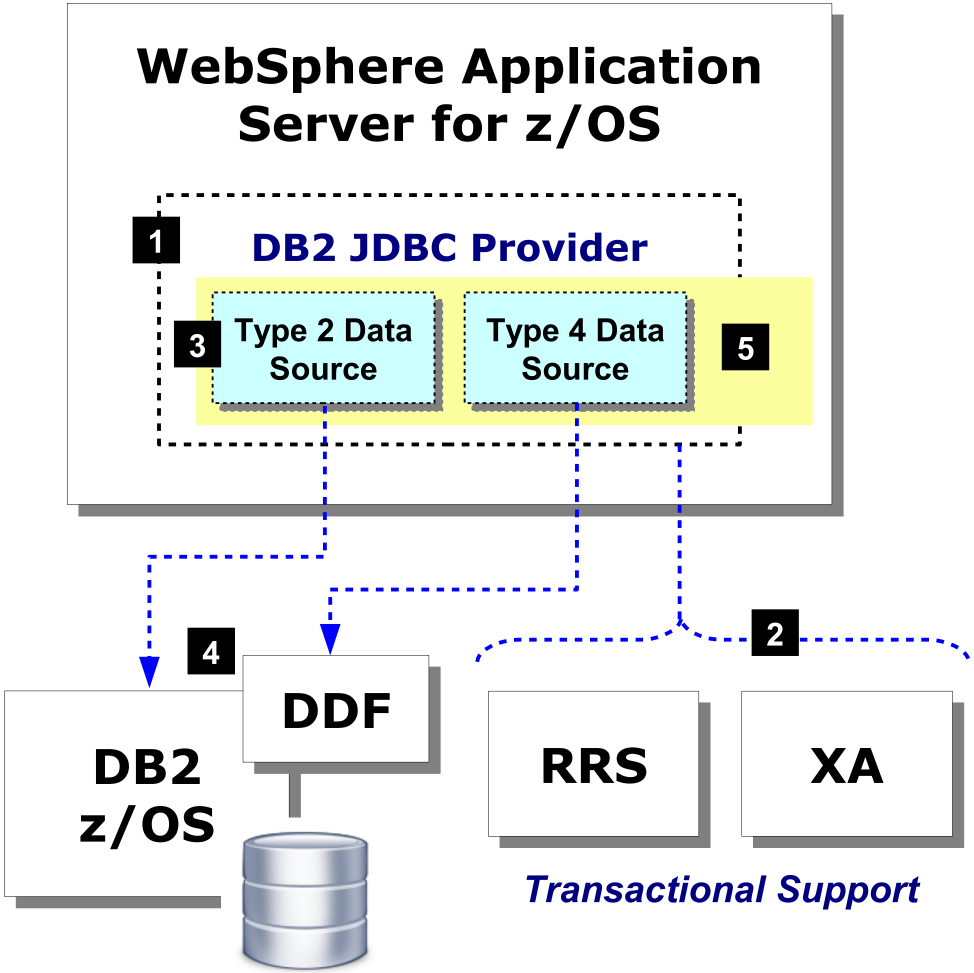
© 2013 IBM Corporation

Relational Data Access

JDBC

Framework of This Section's Discussion

There are five major areas of discussion within this JDBC sub-section of the unit:



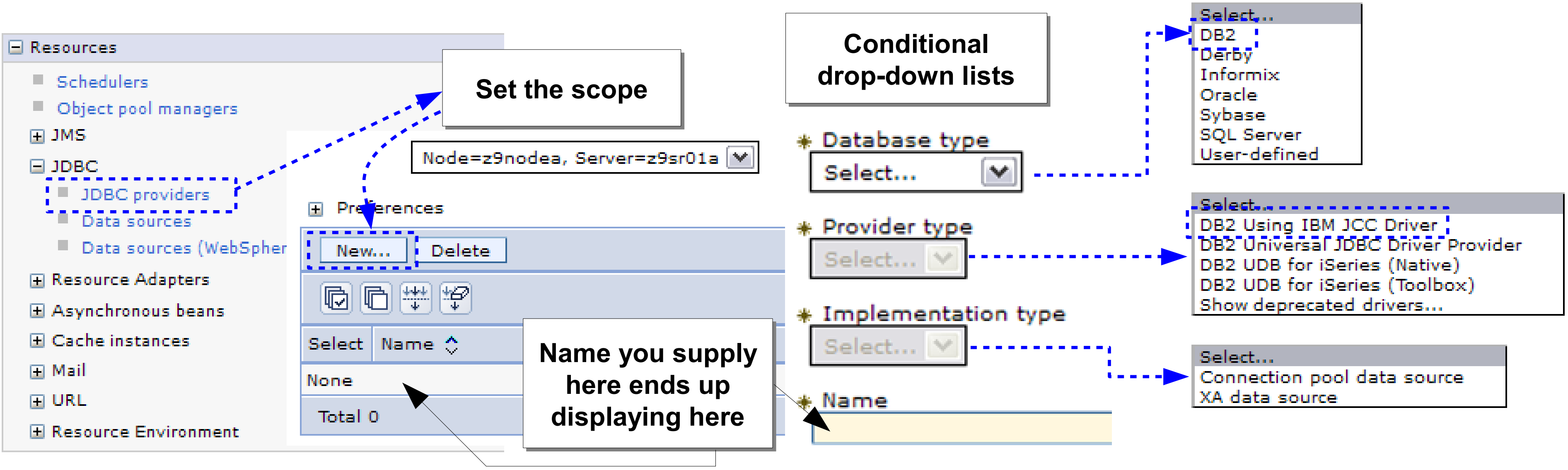
1. **Configuration of Provider**
   Where driver is located, Admin Console panels used to install and configure
2. **Transaction support based on "Implementation Type" selected**
   1 phase or 2 phase, RRS or XA Partner Logs
3. **Configuration of Data Sources**
   Admin Console panels used to configure
4. **Implications of Type 2 v Type 4**
   Specifically, identity assertion
5. **The new failover capabilities of WAS V8**
   Ability to automatically fail over and fail back

JDBC provider …

© 2013 IBM Corporation

Configuring the JDBC Provider for DB2 z/OS

This is a relatively simple process involving a few panels … but some interesting implications are surfaced by the choices made:



Set the scope

Conditional drop-down lists

Name you supply here ends up displaying here

DB2 Using IBM JCC Driver

Contains the JDBC 4.0 specification support

Backwards compatible so applications written to JDBC 3.0 will work with this driver

For WAS z/OS V7 and later this is recommended provider for IBM z/OS DB2, provided your DB2 has the db2jcc4.jar file (which indicates this driver is present).

DB2 Universal JDBC Driver Provider

JDBC 3.0 specification support

Connection Pool Data Source

If data source is Type 4, then 1 Phase Commit only

If data source is Type 2, then 2 Phase Commit with RRS

XA Data Source

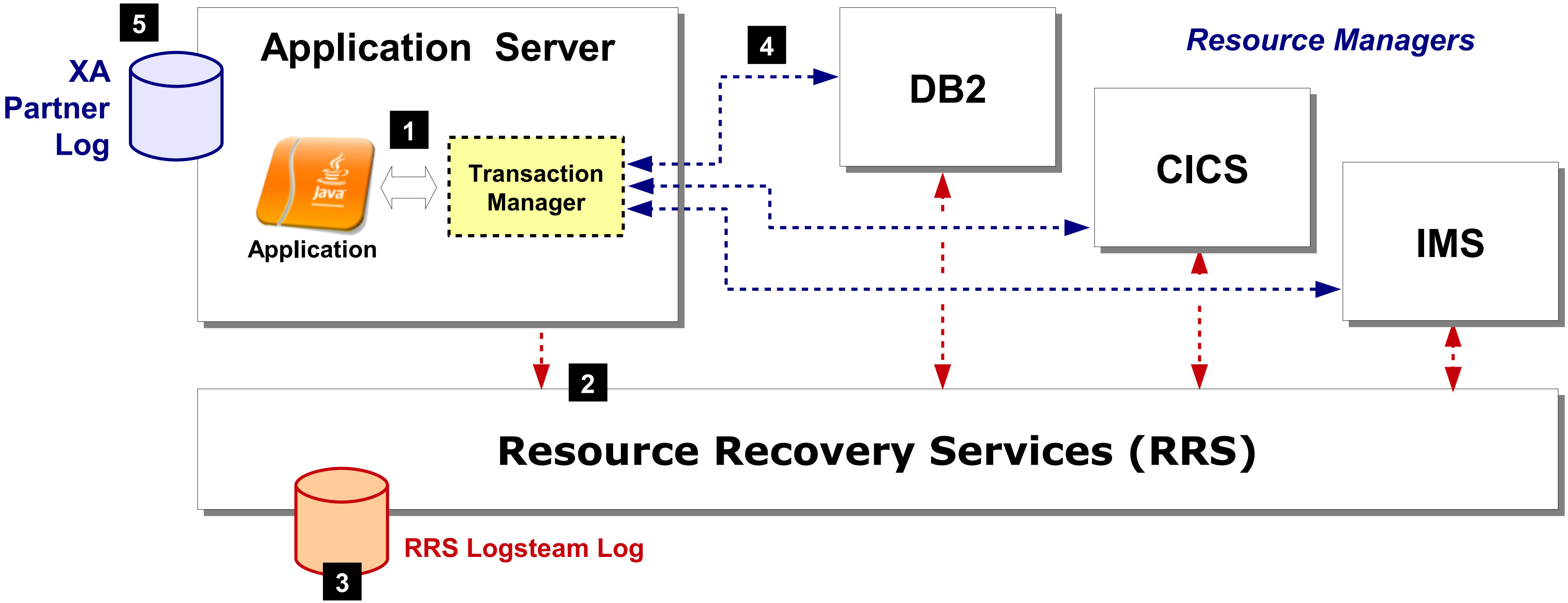
If data source is Type 4, then 2 Phase Commit with XA

Type 2 data sources are not supported under this implementation type

Transaction support …

Brief Discussion of Transaction Support

The previous chart mentioned RRS and XA as the two means of supporting global transactions from WAS into other resource managers ...



RRS is a facility of z/OS. It is Sysplex aware. The RRS log may be maintained in Sysplex-shared data structures. This allows cross-Sysplex Two-Phase Commit (2PC) processing across instances of WAS and resource managers.

XA is an open standard for distributed Two-Phase Commit. The transaction logs are maintained by WAS.

The "Implementation Type" setting on Provider determines which is used

Provider code supplied by DB2 …

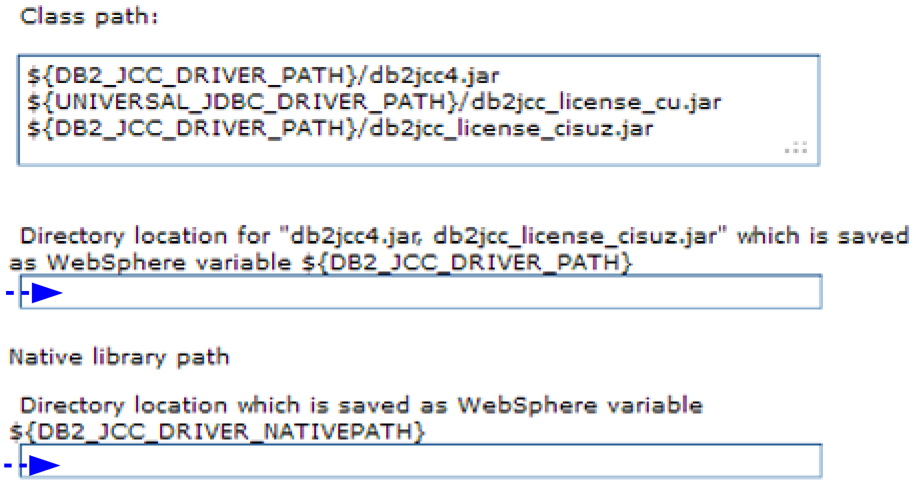
© 2013 IBM Corporation

The Provider Code Supplied by DB2

WAS z/OS does not ship with the provider code ... you point to it in the DB2 directories in the WAS configuration panels:

```
/<mount_point>/db21010/jdbc
          |
          |-- /classes/
          |      db2jcc.jar
          |      db2jcc_javax.jar
          |      db2jcc_license_cisuz.jar
          |      db2jcc4.jar
          |      sqlj.zip
          |      sqlj4.zip
          |
          |-- /lib/
                 libdb2jcct2zos.so
                 libdb2jcct2zos_64.so
                 libdb2jcct2zos4.so
                 libdb2jcct2zos4_64.so
```

JDBC Provider Configuration Panel in WAS:



Class path:

```
${DB2_JCC_DRIVER_PATH}/db2jcc4.jar
${UNIVERSAL_JDBC_DRIVER_PATH}/db2jcc_license_cu.jar
${DB2_JCC_DRIVER_PATH}/db2jcc_license_cisuz.jar
```

Directory location for "db2jcc4.jar, db2jcc_license_cisuz.jar" which is saved as WebSphere variable ${DB2_JCC_DRIVER_PATH}

Native library path

Directory location which is saved as WebSphere variable ${DB2_JCC_DRIVER_NATIVEPATH}

WAS then puts your values into the environment variables. Upon next restart of the server it can find and load the specified JDBC driver.

PDSE

DSN1010.SDSNLINK	APF
DSN1010.SDSNLOAD	APF
DSN1010.SDSNLOD2	APF

If using the Type 2 native drivers then servant regions must have access to the PDSE modules as well

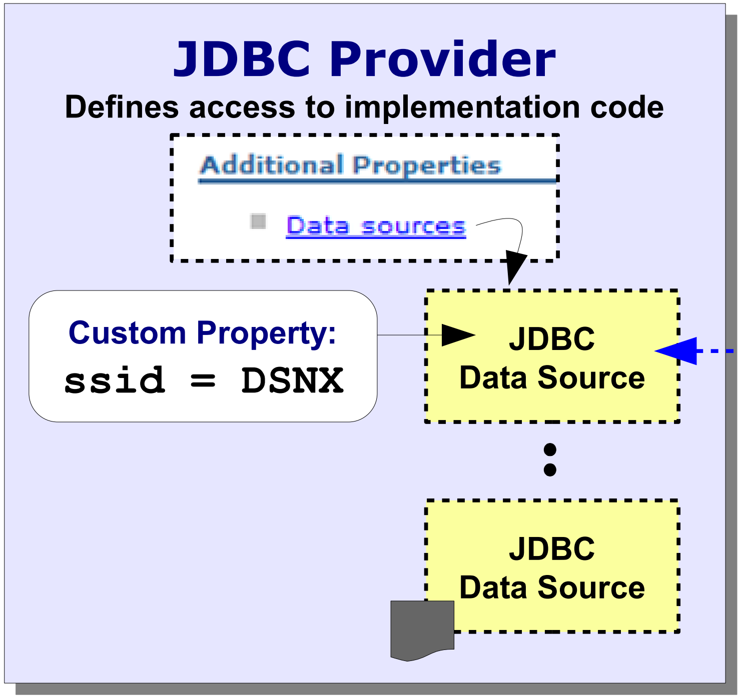
STEPLIB or Linklist

Lab systems have these in Linklist so no STEPLIB is necessary

Data sources …

JDBC Data Sources -- Specific Connection Information

The data source is defined under the provider and has information about how to connect to the desired DB2 instance:



JDBC Provider

Defines access to implementation code

Additional Properties

Data sources

Custom Property:

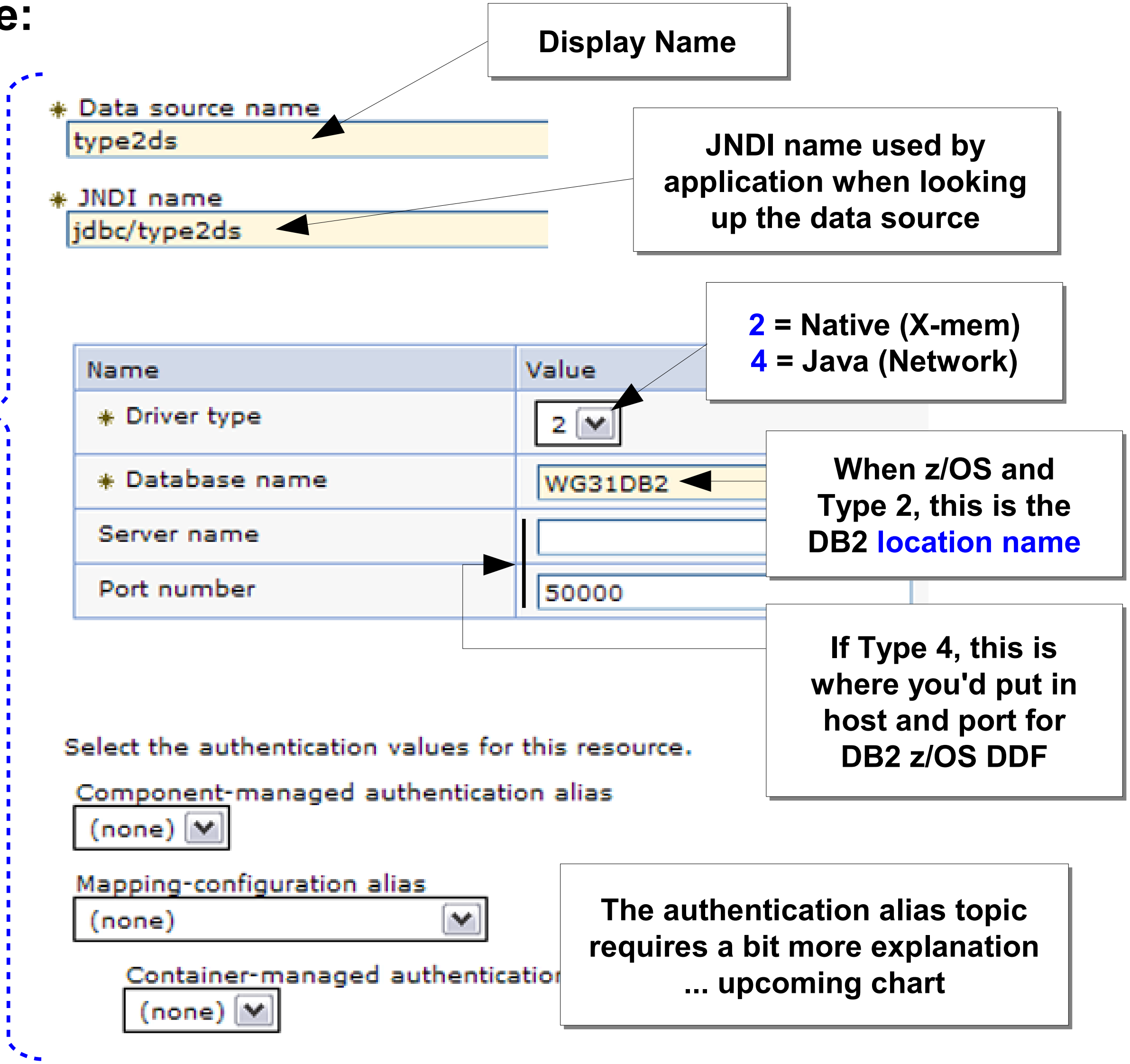
`ssid = DSNX`

JDBC Data Source

:

JDBC Data Source

More data sources possible, each with a separate set of connection specifics



Display Name

* Data source name

type2ds

JNDI name used by application when looking up the data source

* JNDI name

jdbc/type2ds

2 = Native (X-mem)
4 = Java (Network)

Name	Value
* Driver type	2
* Database name	WG31DB2
Server name	
Port number	50000

When z/OS and Type 2, this is the DB2 location name

If Type 4, this is where you'd put in host and port for DB2 z/OS DDF

Select the authentication values for this resource.

Component-managed authentication alias: (none)

Mapping-configuration alias: (none)

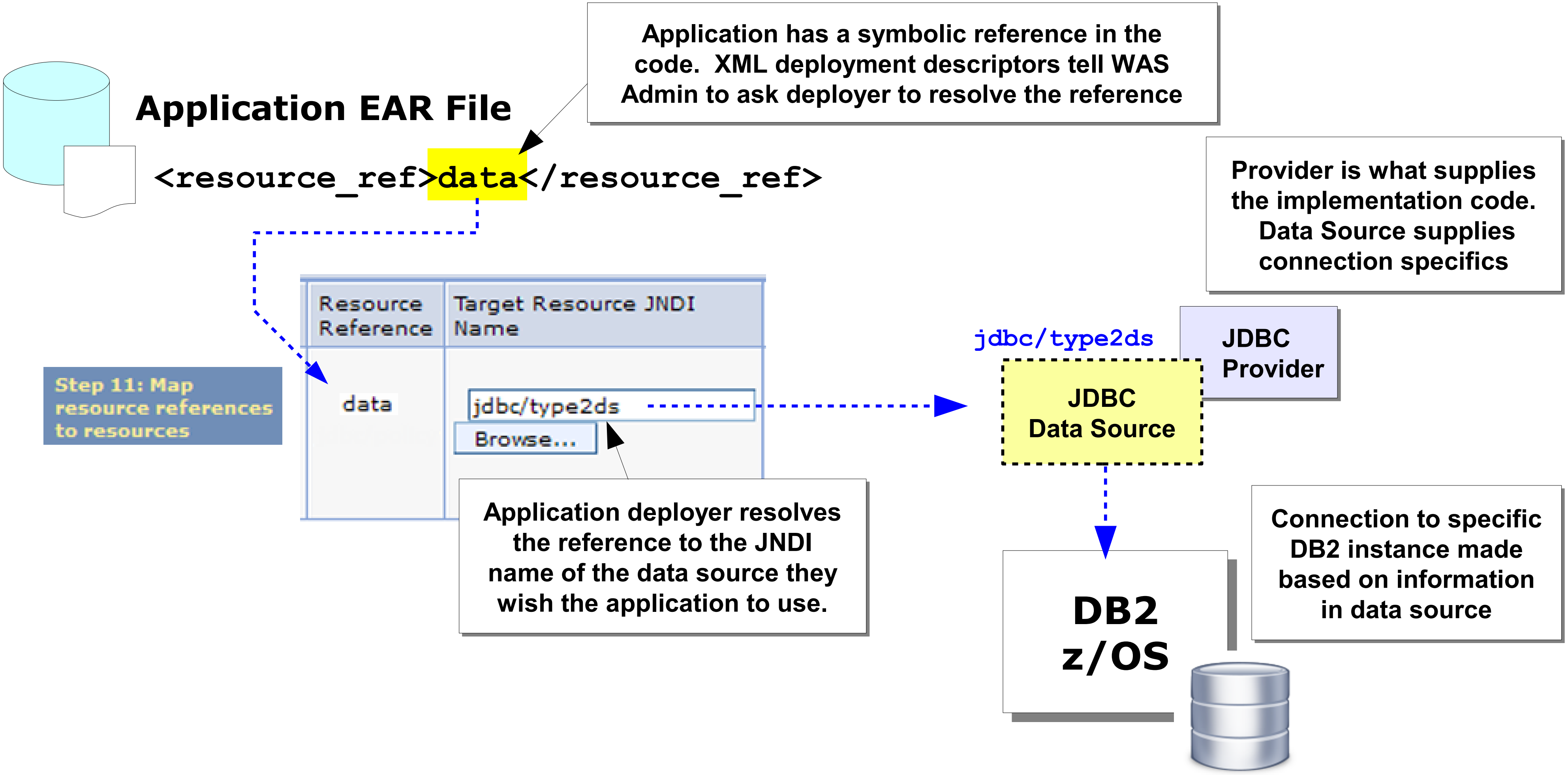
Container-managed authenticatio: (none)

The authentication alias topic requires a bit more explanation ... upcoming chart

Application lookup of JNDI …

Application Lookup of Data Source JNDI

Application "resource references" are bound to data source JNDI names ... that's the sequence of associations that ultimately provides JDBC connection



Application has a symbolic reference in the code. XML deployment descriptors tell WAS Admin to ask deployer to resolve the reference

Application EAR File

```
<resource_ref>data</resource_ref>
```

Provider is what supplies the implementation code. Data Source supplies connection specifics

Step 11: Map resource references to resources

Resource Reference	Target Resource JNDI Name
data	jdbc/type2ds Browse...

Application deployer resolves the reference to the JNDI name of the data source they wish the application to use.

jdbc/type2ds

JDBC Provider

JDBC Data Source

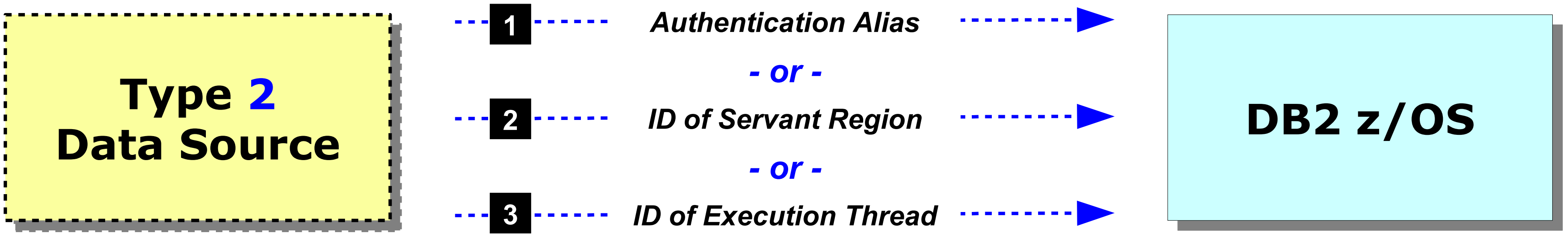
DB2 z/OS

Connection to specific DB2 instance made based on information in data source

Identity assertion …

Identity Assertion from WAS into DB2

There's a few different options depending on if Type 2 or Type 4:



Type 2
Data Source

1 Authentication Alias

- or -

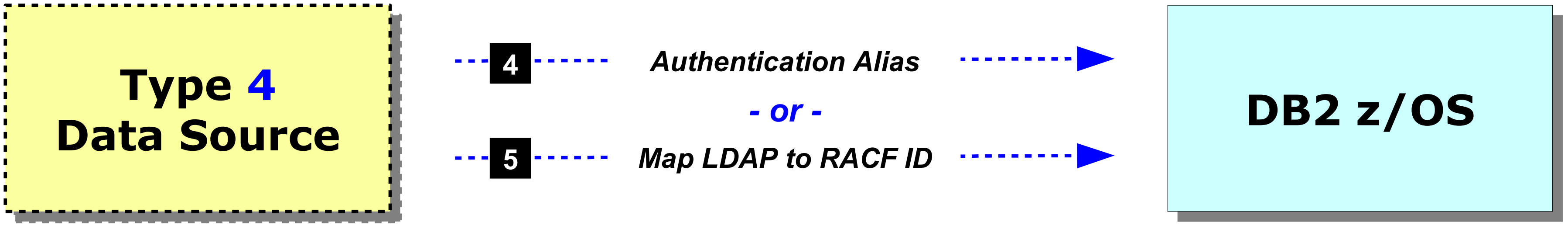
2 ID of Servant Region

- or -

3 ID of Execution Thread

DB2 z/OS

1. An alias is a hard-coded userid/password pair that WAS passes on request
2. If no alias then Type 2 uses the ID of the WAS servant region
3. Use RunAs roles and map ID of execution thread to request into DB2



Type 4
Data Source

4 Authentication Alias

- or -

5 Map LDAP to RACF ID

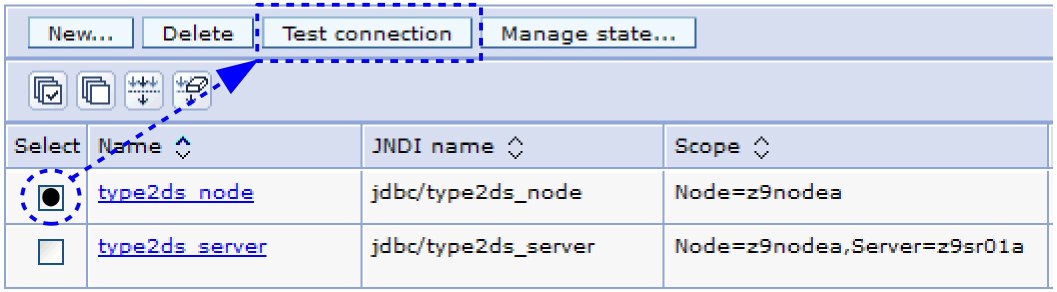
DB2 z/OS

4. An alias is a hard-coded userid/password pair that WAS passes on request
5. New function that allows a distributed LDAP identity to be mapped to a RACF identity
   Function shipped in z/OS 1.13 and rolled back to 1.11. Required DB2 z/OS V10 to use.

Test Connection button …

The "Test Connection" Button

Will perform a rudimentary connection test … its success depends on the scope of the JDBC Provider

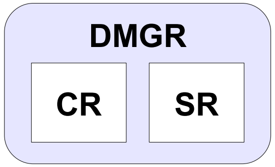


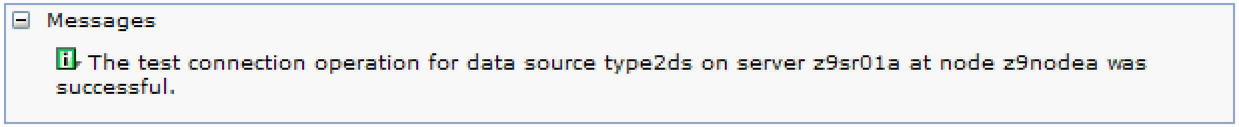
Select	Name	JNDI name	Scope
●	type2ds_node	jdbc/type2ds_node	Node=z9nodea
□	type2ds_server	jdbc/type2ds_server	Node=z9nodea,Server=z9sr01a

The test is executed from the servant region JVM … so the question is whether the server implied from the scope has a servant region

Scope=Cell

Test run from DMGR which has servant region

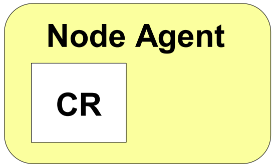


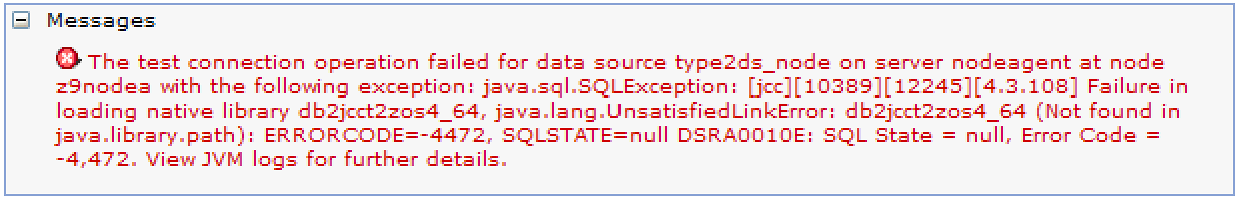


The test connection operation for data source type2ds on server z9sr01a at node z9nodea was successful.

Scope=Node

Test run from Node Agent which does not have servant

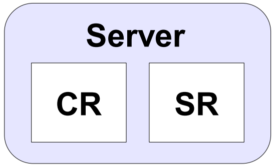


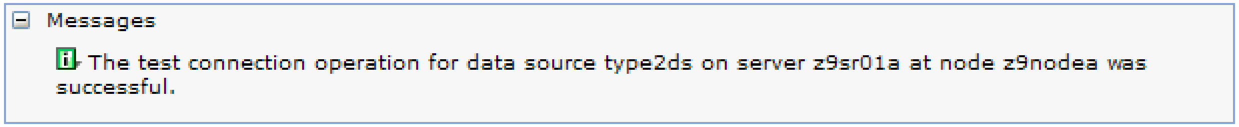


The test connection operation failed for data source type2ds_node on server nodeagent at node z9nodea with the following exception: java.sql.SQLException: [jcc][10389][12245][4.3.108] Failure in loading native library db2jcct2zos4_64, java.lang.UnsatisfiedLinkError: db2jcct2zos4_64 (Not found in java.library.path): ERRORCODE=-4472, SQLSTATE=null DSRA0010E: SQL State = null, Error Code = -4,472. View JVM logs for further details.

Scope=Server

Test run from Server which has servant



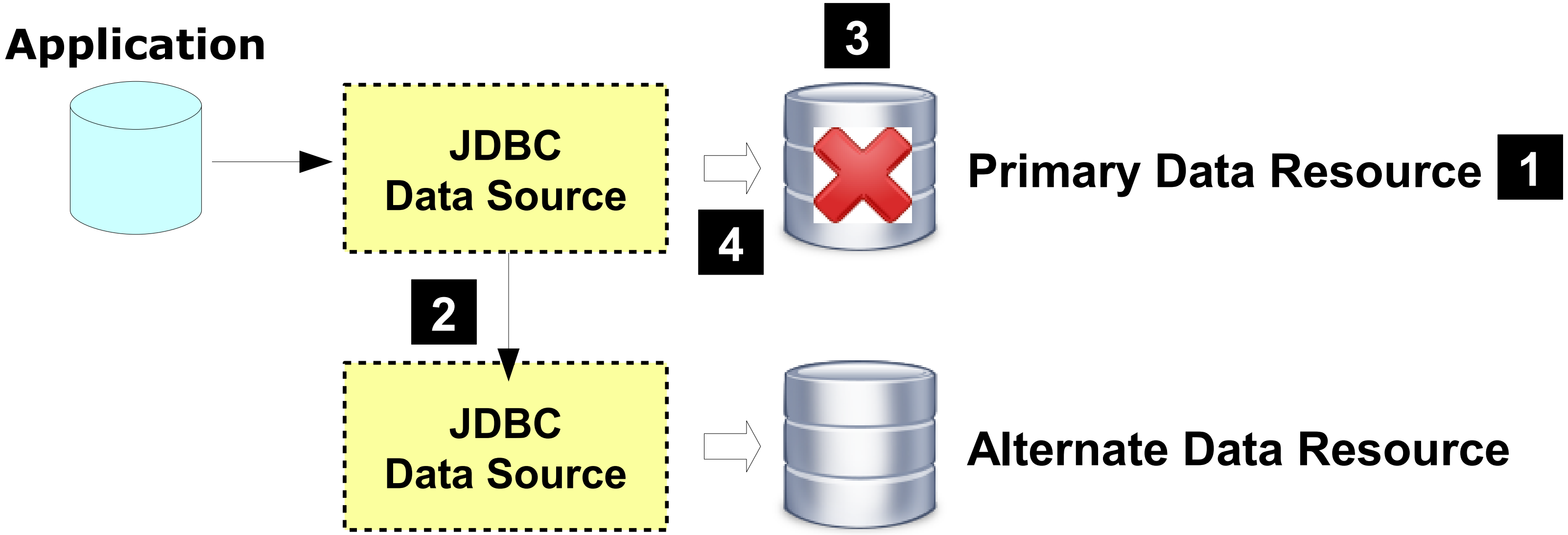


The test connection operation for data source type2ds on server z9sr01a at node z9nodea was successful.

Data resource failover …

Data Resource Failover - Four Questions

The new function in WAS V8 (all platforms) is designed to address four questions related to data resource failover and failback:

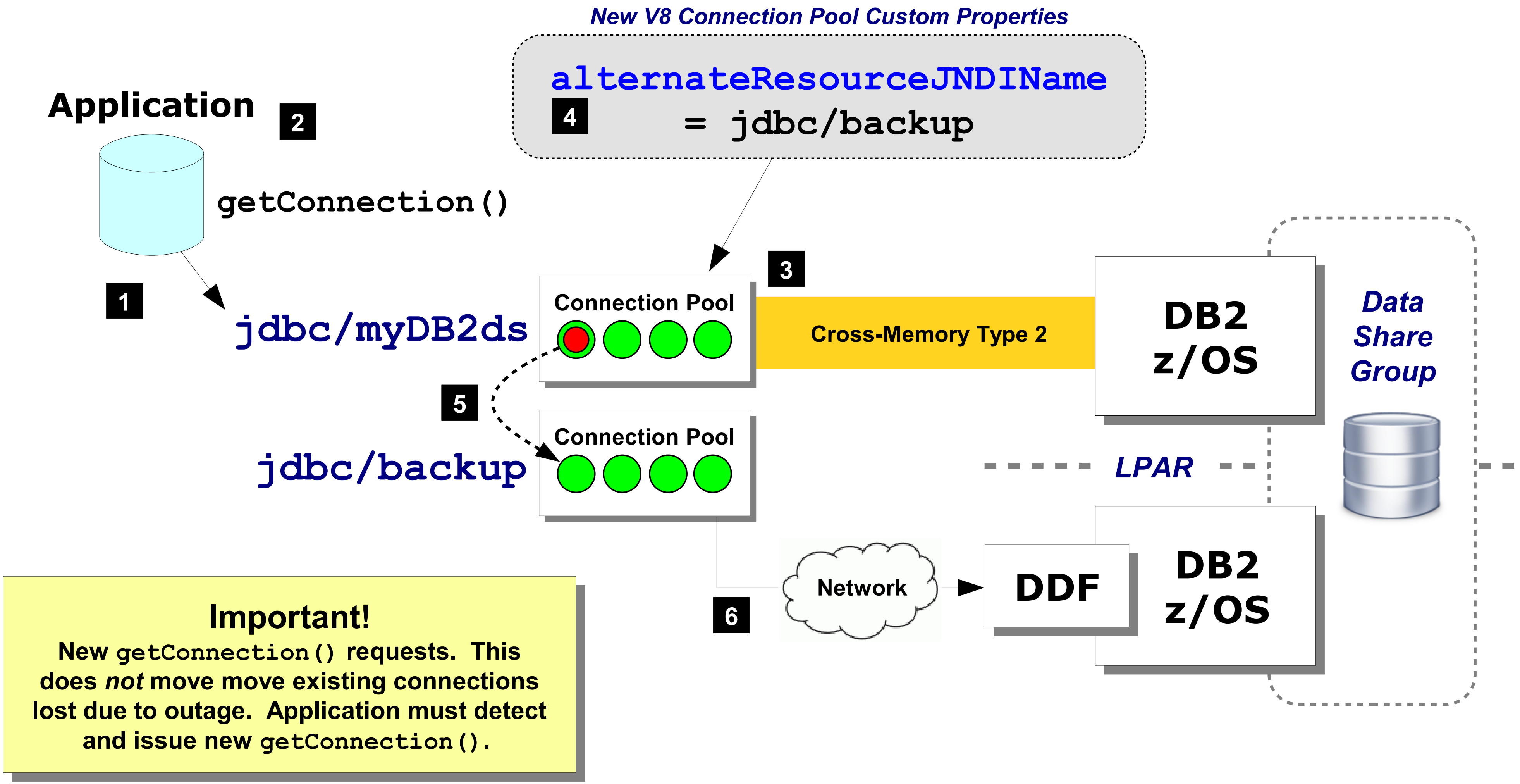


1. **Has the primary data resource failed?**
   We'll discuss the mechanism used to trigger the failover function
2. **What alternative data resource is available?**
   This is defined with a new variable
3. **Has the primary data resource recovered?**
   A test for primary resource recovery is made
4. **Should failback to the primary be manual or automatic?**
   You may not want automatic failback … there are ways to control this

Essentials of failover support …

Essentials of Resource Failover

A new environment variable is used to define an "alternate JNDI" for use when the primary JNDI experiences `getConnection()` problems:



New V8 Connection Pool Custom Properties

```
alternateResourceJNDIName
        = jdbc/backup
```

Important!

New `getConnection()` requests. This does *not* move move existing connections lost due to outage. Application must detect and issue new `getConnection()`.

InfoCenter `cdat_dsfailover`

Other custom properties …

Other Connection Pool Custom Properties

Four other connection pool custom properties are also made available:

`failureThreshold`

Determines the number of consecutive `getConnection()` failures are needed to trigger the failover processing
Integer, Default = 5

`resourceAvailabilityTestRetryInterval`

After failover has occurred, this determines the frequency of polling to see if the primary resource has recovered
Integer, Default = 10 seconds

`enablePartialResourceAdapterFailoverSupport`

Indicates that automatic failover is permitted but automatic failback is disabled
Boolean, Default = False

`disableResourceFailOver`
`disableResourceFailBack`

Disables automatic failover or failback. Used to allow configuration of failover values, but control using z/OS `MODIFY`
Boolean, Default = False

InfoCenter `cdat_dsfailover`

z/OS MODIFY …

z/OS MODIFY Control of Failover and Failback

The following MODIFY commands will act upon a server where the connection pool custom property `alternateResourceJNDIName` has previously been configured:

Manual Failover to Alternate and Failback to Primary

```
F <server>,FAILOVER,'<JNDI Name>'
F <server>,FAILBACK,'<JNDI Name>'
```

The JNDI name is that of the primary data source. Never the defined alternate data source.

Note the *single quotes* enclosing the JDNI name

Manual Disable or Enable of Automatic Failover / Failback

```
F <server>,DISABLEFAILOVER,'<JNDI Name>'
F <server>,ENABLEFAILOVER,'<JNDI Name>'
```

The JNDI name is that of the primary data source. Never the defined alternate data source.

These MODIFY commands override connection pool custom properties you may have set of enable and disable of failover and failback

InfoCenter `rxml_mvsmodify`

z/OS Action Notification …

z/OS failureNotificationActionCode

These define actions to take when the primary is unreachable *and* any defined alternate JNDI resources are also unreachable:

```
failureNotificationActionCode = 1 | 2 | 3
```

1 Issue a BBOJ0130I message, but take no other action

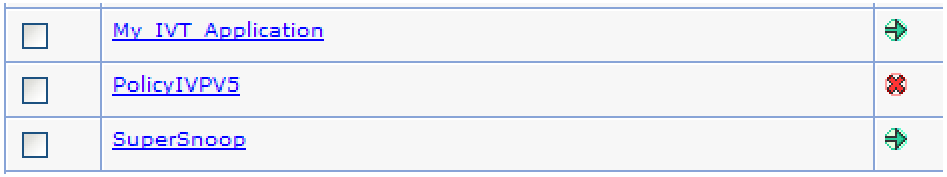
```
BBOJ0130I: CONNECTION MANAGEMENT IN A SERVANT REGION DETECTED THAT THE
RESOURCE IDENTIFIED BY JNDI NAME jdbc/type2ds IS DISCONNECTED FROM SERVER
z9cell/z9nodea/Z9SR01/z9sr01a. ACTION TAKEN: NONE.
```

2 Issue PAUSELISTENERS for the server; RESUMELISTENERS when resource is back

```
ACTION TAKEN: PAUSING LISTENERS.
BBOO0222I: ZAIO0002I: z/OS asynchronous IO TCP Channel TCP_1 has stopped
listening on host * port 10065.
 :
BBOO0222I: ZAIO0002I: z/OS asynchronous IO TCP Channel TCP_4 has stopped
listening on host * port 10068.
```

Front-end routing devices will detect loss of listener ports and route to other members of a cluster

3 Stop applications using failed resource; restart applications when resource is back



□	My IVT Application	➜
□	PolicyIVPV5	☒
□	SuperSnoop	➜

Makes affected application unavailable but leaves intact other applications in the server

Non-relational …

Non-Relational Data Access

CICS

The Role of the CICS Transaction Gateway Product

Connectivity from WAS to CICS requires the CICS Transaction Gateway product to provide the necessary software function. There are several components of CTG:

CICS Transaction Gateway for Multiplatforms
http://publib.boulder.ibm.com/infocenter/cicstgmp/v8r1/index.jsp

Windows AIX Linux

CICS Transaction Gateway for z/OS
http://publib.boulder.ibm.com/infocenter/cicstgzo/v8r1/index.jsp

z/OS

- **The most recent version is V8.1**
- **Two key components:**

Java Connector Architecture (JCA) compliant resource adapter

This is a package of code that *installs into the WAS runtime environment.* It provides the open standard application interface and code to interact with CICS. The bundle is packaged as a "RAR" file (Resource ARchive).

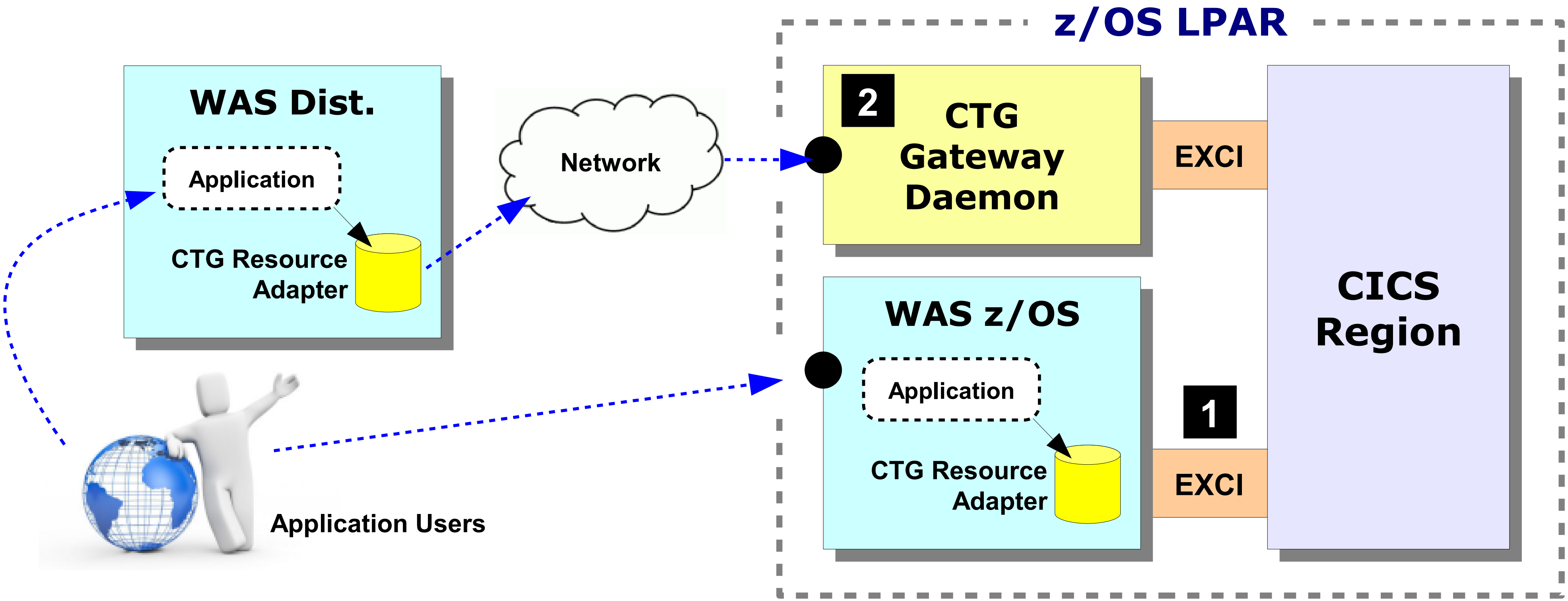
Code to run as a started "Gateway Daemon" process or task

The Gateway Daemon provides an intermediary agent for clients to connect to; the Gateway Daemon then communicates with the CICS region to complete the connection

Two topologies …

Two Simple Topologies … to Start the Discussion

There are several variations on topologies and it can be a bit confusing at first. Let's start with two relatively simple examples to set some context:



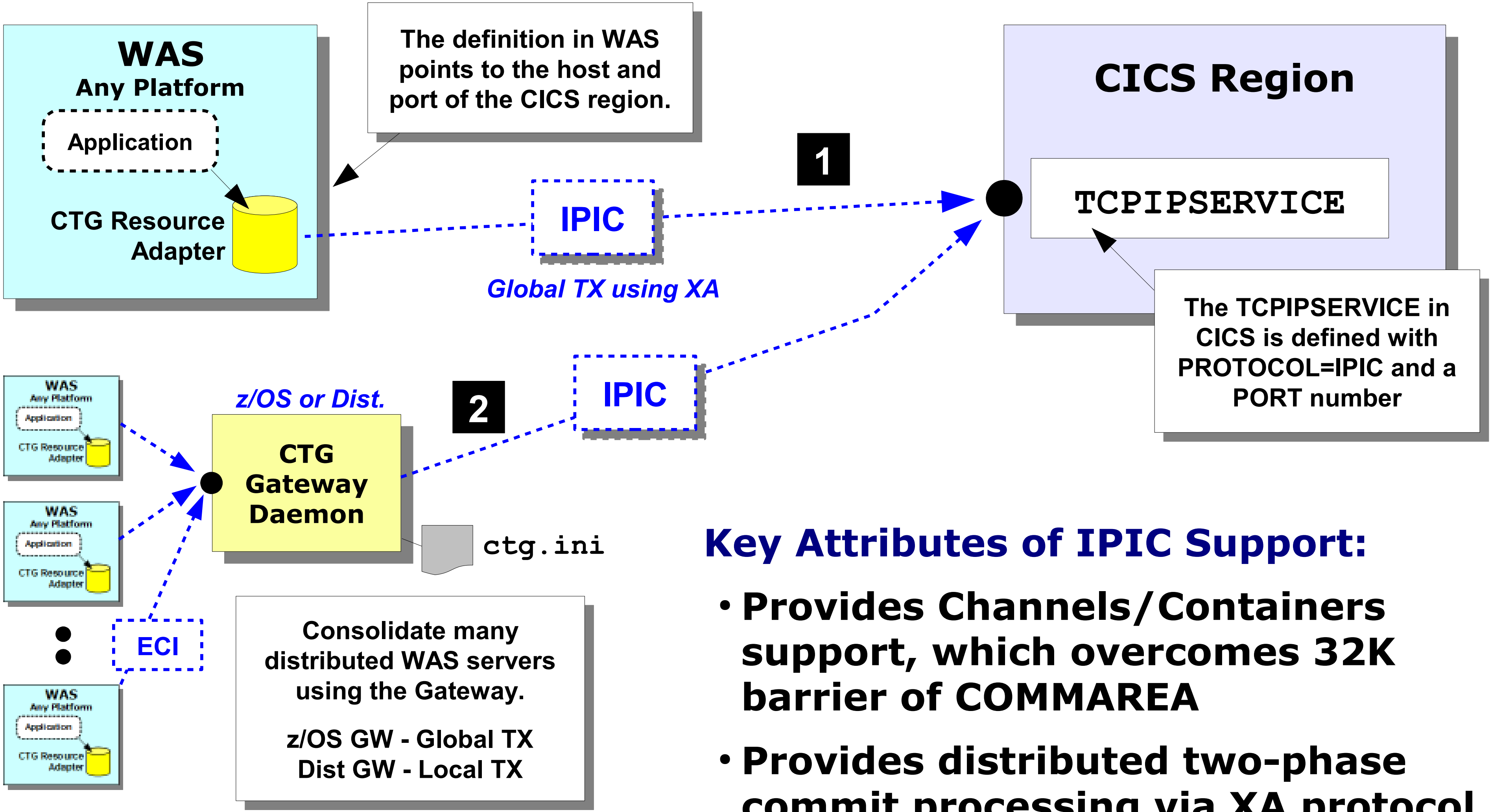
1. **WAS uses RAR to access CICS with EXCI**
   This is known as "Local Mode" in CTG terminology
2. **WAS uses RAR and TCP to access Gateway; Gateway uses EXCI to access the CICS region**
   This is known as "Remote Mode" in CTG terminology

We need to introduce IPIC

IPIC …

CICS and IPIC

IPIC is a CICS program call protocol that maps on TCP/IP (or SSL). There are two modes -- "local" and "remote":



The definition in WAS points to the host and port of the CICS region.

The TCPIPSERVICE in CICS is defined with PROTOCOL=IPIC and a PORT number

Consolidate many distributed WAS servers using the Gateway.

z/OS GW - Global TX
Dist GW - Local TX

Key Attributes of IPIC Support:

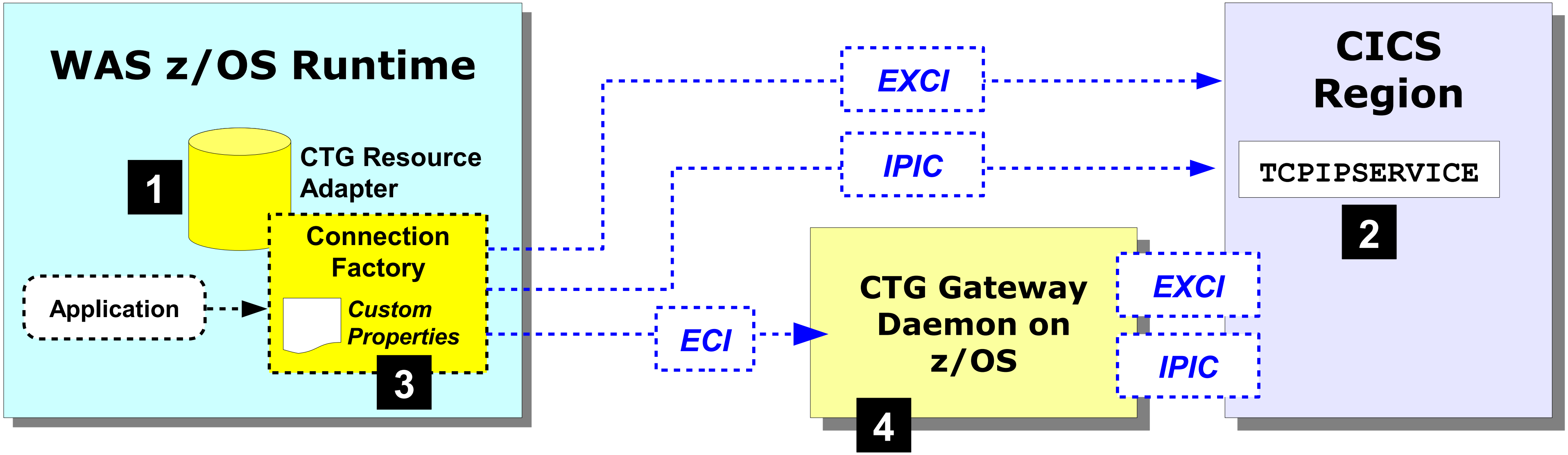
- **Provides Channels/Containers support, which overcomes 32K barrier of COMMAREA**
- **Provides distributed two-phase commit processing via XA protocol**

CTG InfoCenter `ipicconfig`

Our focus …

Many Options ... Our Focus Will Be on WAS z/OS

This workshop is focused on WAS z/OS, so our discussion of CTG for access to CICS will be on z/OS-related topologies:



1. **The installation of the CTG Resource Adapter**

   Which is the starting point to providing WAS-to-CICS connectivity

2. **The basics of the `TCPIPSERVICE` and `IPCONN` definitions in CICS**

   To show the interrelationship between values there and what's coded on the connection factories

3. **The configuration of JCA Connection Factories**

   In particular the configuration of the custom properties in support of the connection types -- EXCI, IPIC or to Gateway Daemon

4. **An overview of the CTG Gateway Daemon**

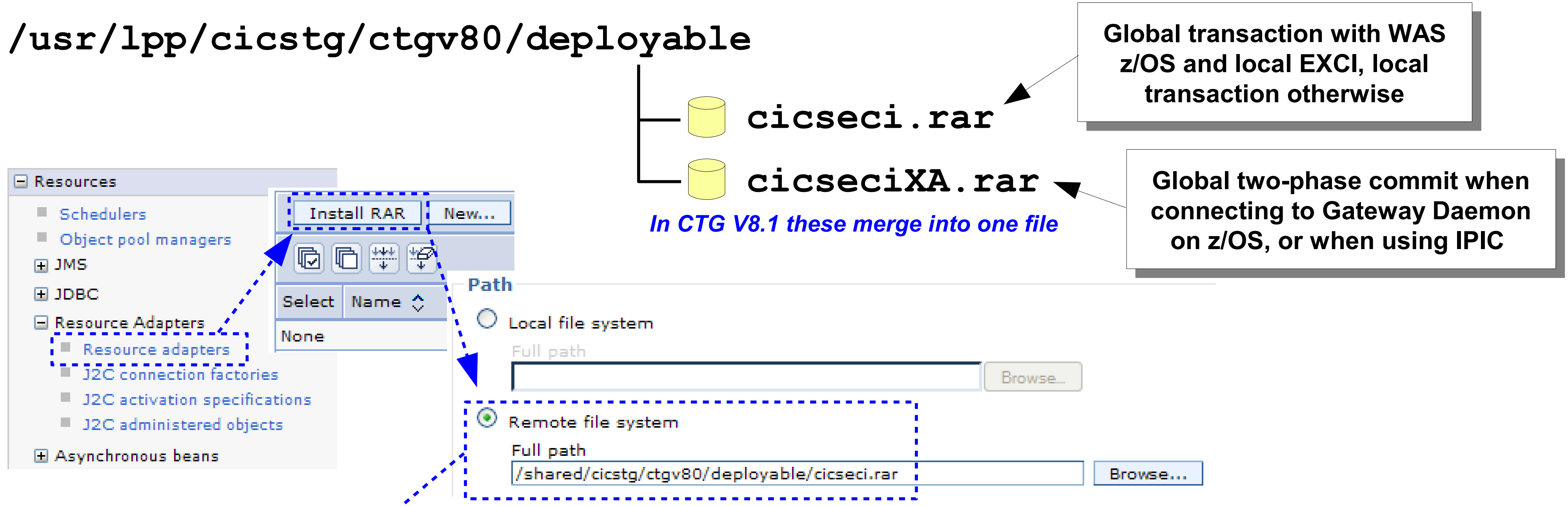
   To give you a sense for the structure and configuration settings of the Gateway Daemon

CTG RAR file …

© 2013 IBM Corporation

The CTG Resource Adapter RAR File

The RAR (Resource ARchive) is the adapter in its installable packaging format. You use the Admin Console to install that RAR file into the WAS runtime environment:



```
/usr/lpp/cicstg/ctgv80/deployable
                                  ├── cicseci.rar
                                  └── cicseciXA.rar
```

Global transaction with WAS z/OS and local EXCI, local transaction otherwise

Global two-phase commit when connecting to Gateway Daemon on z/OS, or when using IPIC

In CTG V8.1 these merge into one file

Path

○ Local file system

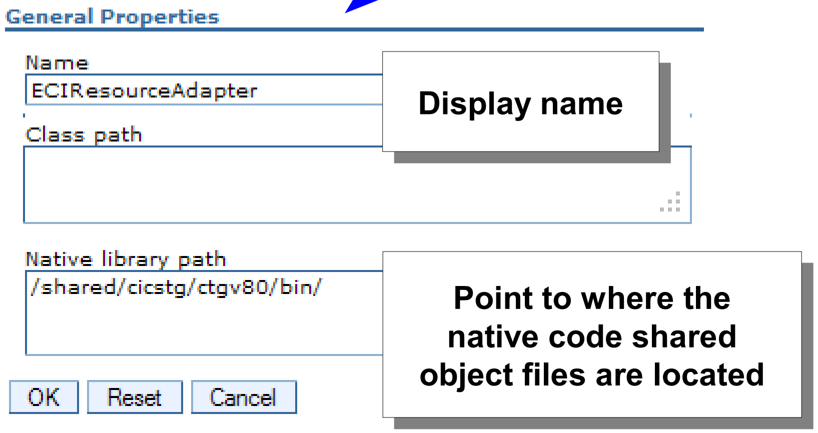
Full path

Browse...

● Remote file system

Full path: /shared/cicstg/ctgv80/deployable/cicseci.rar

Browse...



General Properties

Name: ECIResourceAdapter

Class path:

Native library path: /shared/cicstg/ctgv80/bin/

Display name

Point to where the native code shared object files are located

Before we get to the definition of the Connection Factories, let's take a brief look at the definitions inside of CICS to support IPIC

TCPIPSERVICE and IPCONN …

CICS Definitions in Support of IPIC Usage

Two elements to this -- the TCPIPSERVICE definition and the IPCONN definition. Values you provide here are used in the JCA Connection Factory definition ...

TCPIPSERVICE Definition

```
CEDA  View TCpipservice( SRVTCPX  )
 TCpipservice     : SRVTCPX
 GROup            : IPICX
 DEScription      :
 Urm              : DFHISAIP
 POrtnumber       : 10099
 STatus           : Open
 PROtocol         : IPic
 TRansaction      : CISS
 Backlog          : 00005
 TSqprefix        :
 Host             : ANY
  (Mixed Case)    :
 Ipaddress        : ANY
 SOcketclose      : No
 MAXPersist       : No
```

The name of the service

Definition of the procotocl for the service

This service is listening on port 10099 on any of the TCP hosts defined to the system on which the CICS region resides

IPCONN Definition

```
CEDA  View Ipconn( IPCONX   )
 Ipconn           : IPCONX
 Group            : IPICX
 DEScription      :
IPIC CONNECTION IDENTIFIERS
 APplid           : IPCONX
 Networkid        : IPCONNET
 Host             :
  (Mixed Case)    :
 Port             : No
 Tcpipservice     : SRVTCPX
IPIC CONNECTION PROPERTIES
 Receivecount     : 000
 SENdcount        : 000
 Queuelimit       : No
 MAxqtime         : No
```

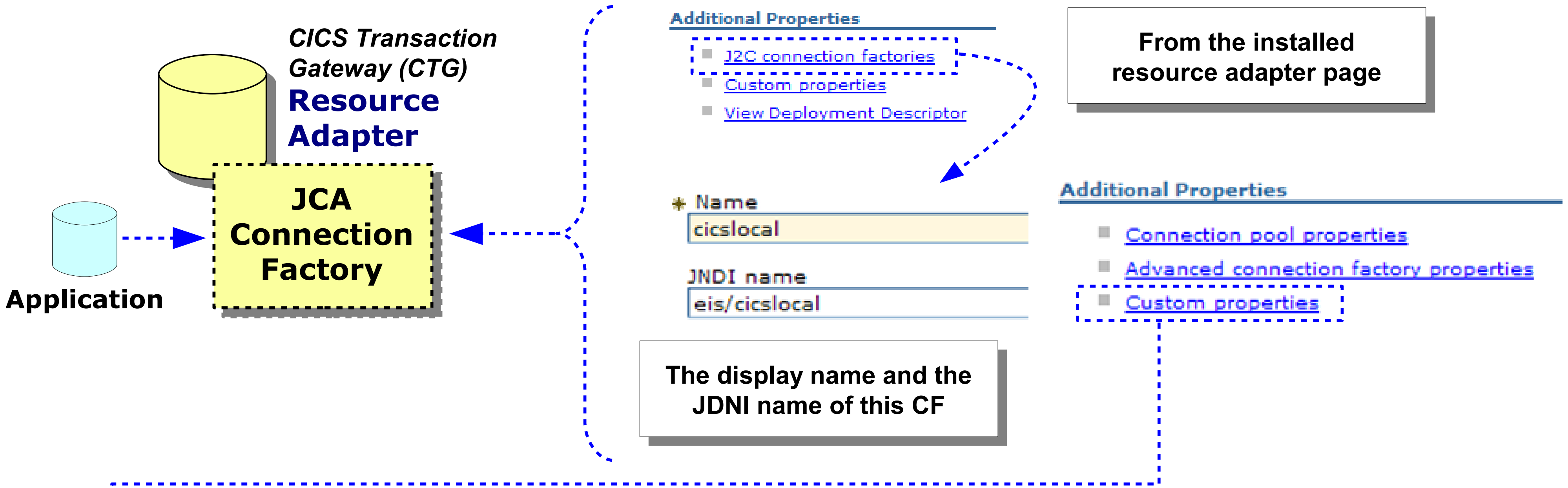
These values are used on the connection factory definition when using IPIC to connect to the CICS region

The JCA Connection Factory may now be configured to communicate with defined TCPIPSERVICE/IPCONN

JCA connection factories …

JCA Connection Factories

Connection Factories (CFs) provide the specifics for the connection to CICS:



Additional Properties
- J2C connection factories
- Custom properties
- View Deployment Descriptor

From the installed resource adapter page

* Name: cicslocal

JNDI name: eis/cicslocal

The display name and the JDNI name of this CF

Additional Properties
- Connection pool properties
- Advanced connection factory properties
- Custom properties

	IPIC	*To GW Daemon*	*Local EXCI*
Applid	*IPCONN APPLID*		
ApplidQualifier	*IPCONN NETWORKID*		
ConnectionURL		tcp://*gw_host* ssl://*gw_host*	local:
PortNumber		*gw_port*	
ServerName	tcp://*host:port* ssl://*host:port*		*CICS applid*

Other custom properties exist ... these are the key properties we'll focus on

CTG Gateway Daemon …

CTG Gateway Daemon

Here's a brief overview of the essential structure of the Gateway Daemon task:

CTGV8MT.SCTGSAMP

Supplied sample library

CTGJOB

Sample JCL that launches CTG Gateway Daemon

```
//CTGJOB    JOB (0),MSGCLASS=X,CLASS=A,NOTIFY=&SYSUID,REGION=500M
//CTG       EXEC PGM=CTGBATCH,
//          PARM='/shared/cicstg/ctgv80/bin/ctgstart -noinput '
//STEPLIB   DD DSN=CTGV8MT.SCTGLOAD,DISP=SHR
//STDENV    DD DSN=USER1.WAS.CNTL(CTGENV),DISP=SHR
```

CTGENV Member:

- Pointer to `ctg.ini` file used
- Pointer to Java installation
- Other definitions

ctg.ini File

- TCP information, including listen port
- CICS APPLID information if EXCI
- IPIC information is using IPIC to connect to CICS region
- Other definitions

Consult the CTG InfoCenter for specifics on customizing CTG Gateway Daemon

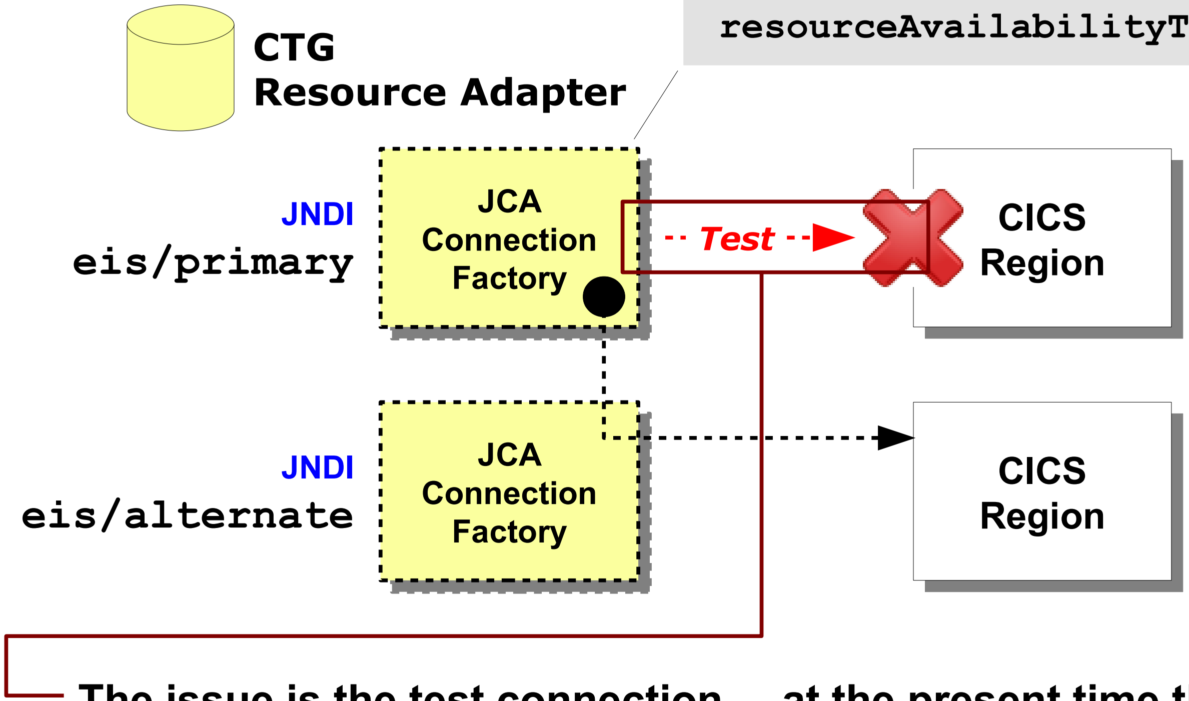
Resource failover/failback …

Resource Failover and Failback -- Work with CICS?

The resource failover methodology we explored for JDBC applies here as well, with one notable exception -- automatic failback:

Connection Pool Custom Properties

```
alternateResourceJNDIName = eis/alternate
failureThreshold = 5
resourceAvailabilityTestRetryInterval = 10 seconds
```



The same connection pool custom properties that we discussed earlier are applicable to the CICS environment as well

The same mechanism for failover applies -- `getConnection()` failures triggers failover to defined alternative connection factory

The issue is the test connection ... at the present time the CTG resource adapter code "test connection" process will always indicate a positive, even when CICS is not there.

Use `enablePartialResourceAdapterFailoverSupport` or `disableResourceFailBack` to prevent failback polling. Then use `MODIFY` failback when you know CICS region is truly back.

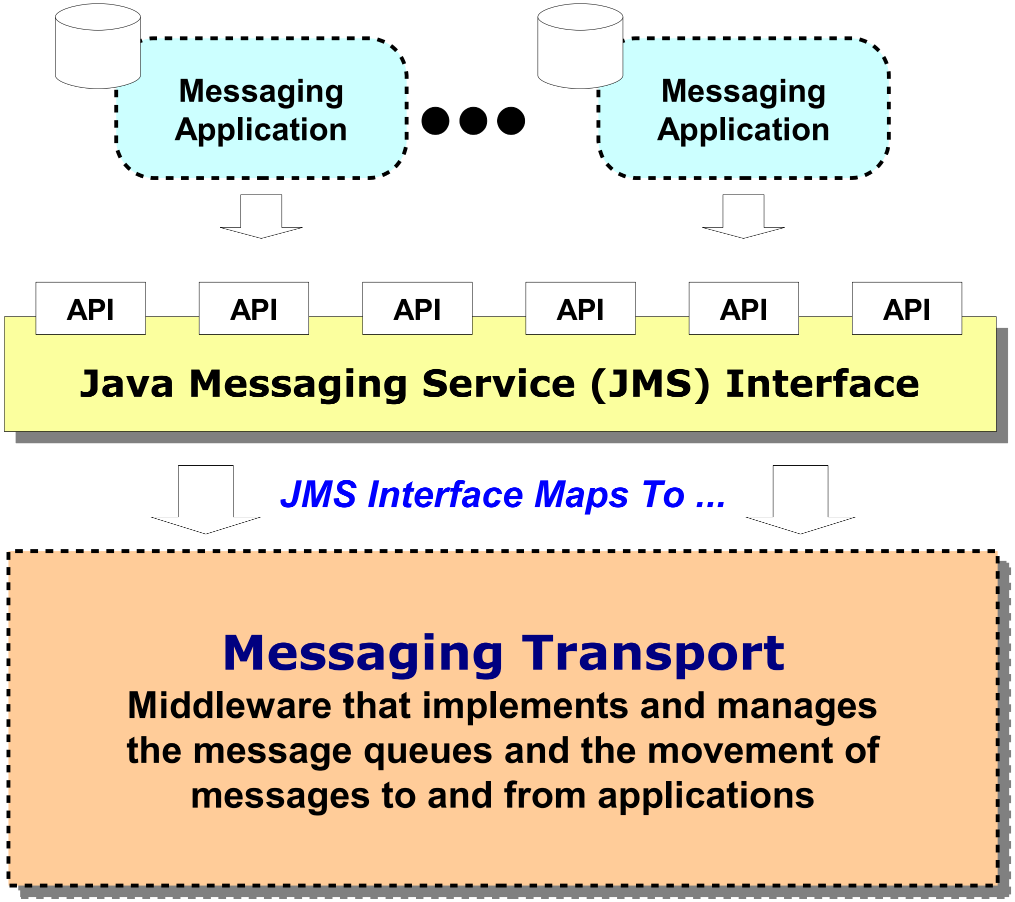
Messaging …

JMS and Messaging

With a focus on MQ

The Difference Between JMS and Messaging Transport

JMS is a messaging *interface* while MQ is an example of a messaging transport that may be used under the interface:



Messaging Application
●●●
Messaging Application
API
API
API
API
API
API
Java Messaging Service (JMS) Interface
JMS Interface Maps To ...

Messaging Transport

Middleware that implements and manages the message queues and the movement of messages to and from applications

Open standards above this line ... applications are unaware of the underlying message transport

Vendor-supplied transports below this line

IBM WebSphere MQ

The focus of this unit. An existing MQ infrastructure may be used as the transport under the JMS interface

WebSphere SIBus

The built-in all-Java messaging transport mechanism of WebSphere Application Server

Other Vendor Transports

WAS, as a Java EE server runtime, supports other vendor transports as well

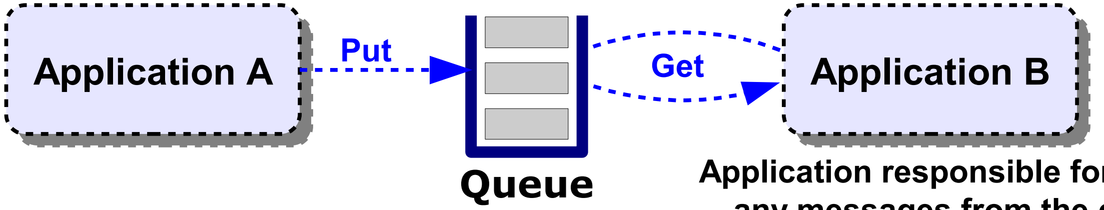
Application types …

Types of Messaging Applications

When discussing JMS configuration within WebSphere Application Server, it's helpful to keep straight three basic types of messaging applications:

Point-to-Point (or PUT/GET)

Very common model where an application puts a message on a queue and another application pulls the message

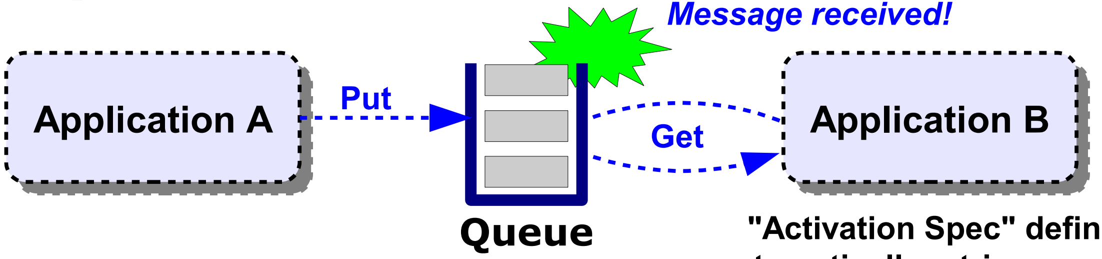


Application responsible for pulling any messages from the queue

Message Driven Bean (MDB)

Receipt of a message in the assigned queue triggers a process to automatically get the message and invoke the Java bean

TechDocs `WP101792`

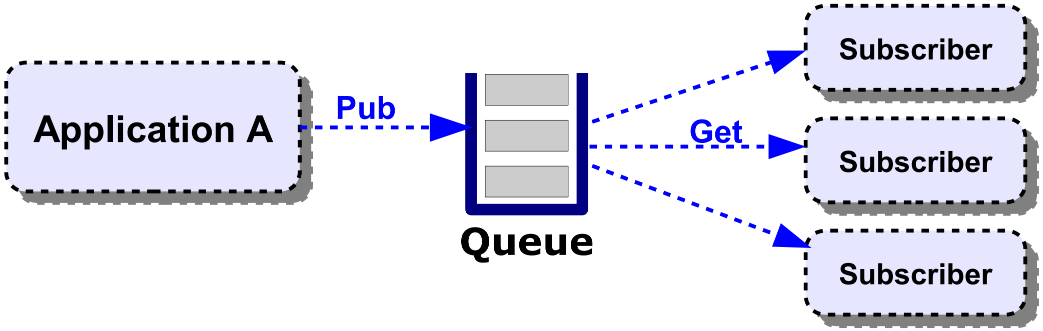


Message received!

"Activation Spec" defined to automatically retrieve messages

Pub / Sub

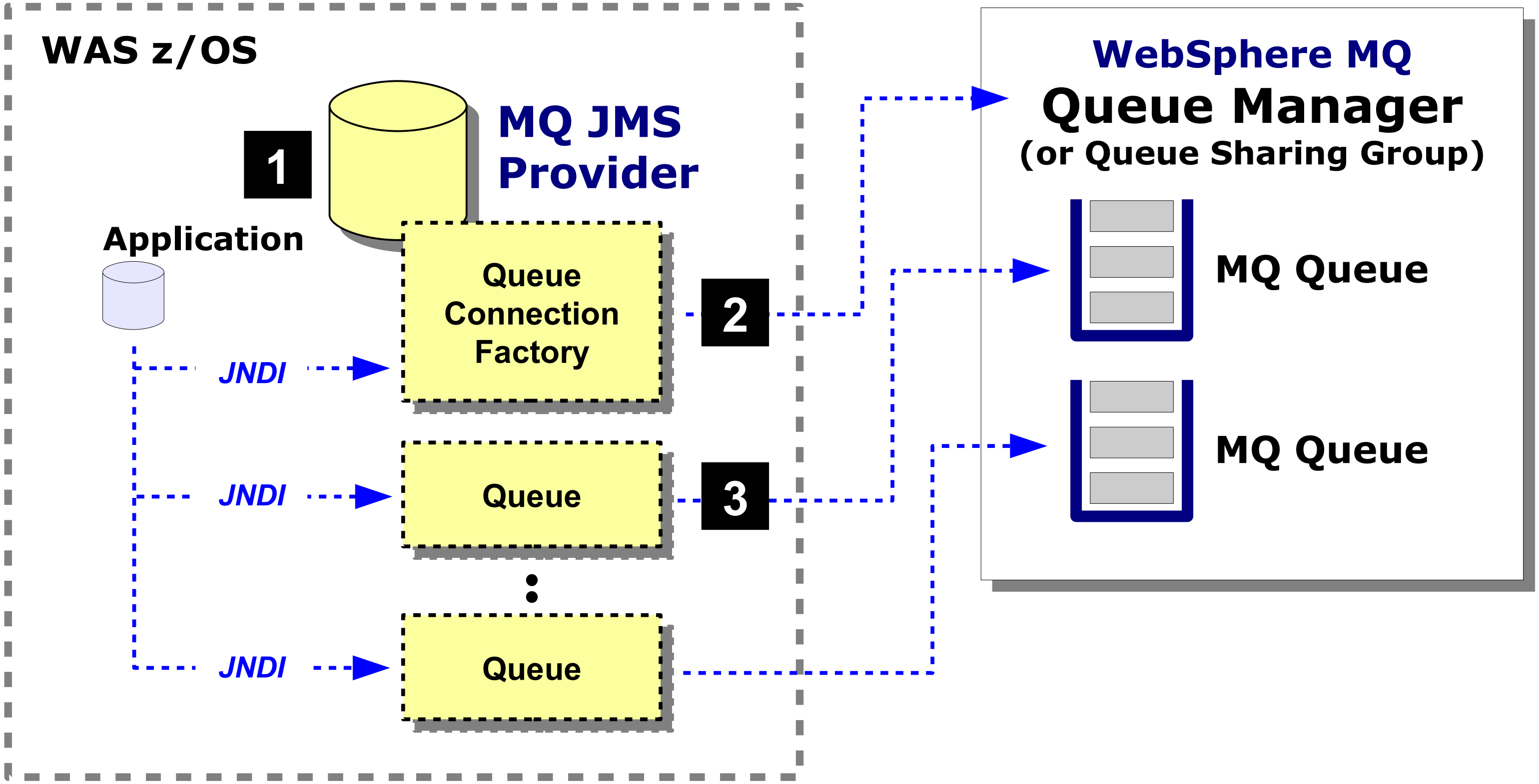
Applications "subscribe" to a queue. Provider "publishes" messages to the queue, and subscribers (one to many) pull from the subscription queue



Key concepts …

Key Concepts of WAS and MQ Messaging

Some of the concepts are similar to JDBC and JCA, but there are some differences:



1 The MQ JMS Provider supplies the code needed to access MQ

Unlike JDBC or JCA, the provider code is supplied with WAS itself. It does not need to be installed.

2 Queue Connection Factory provides specifics about connecting to MQ

Typically a Queue Manager, but may be a queue sharing group. Two modes: binding and client.

3 Queue definitions provide abstraction of physical queue in QMGR

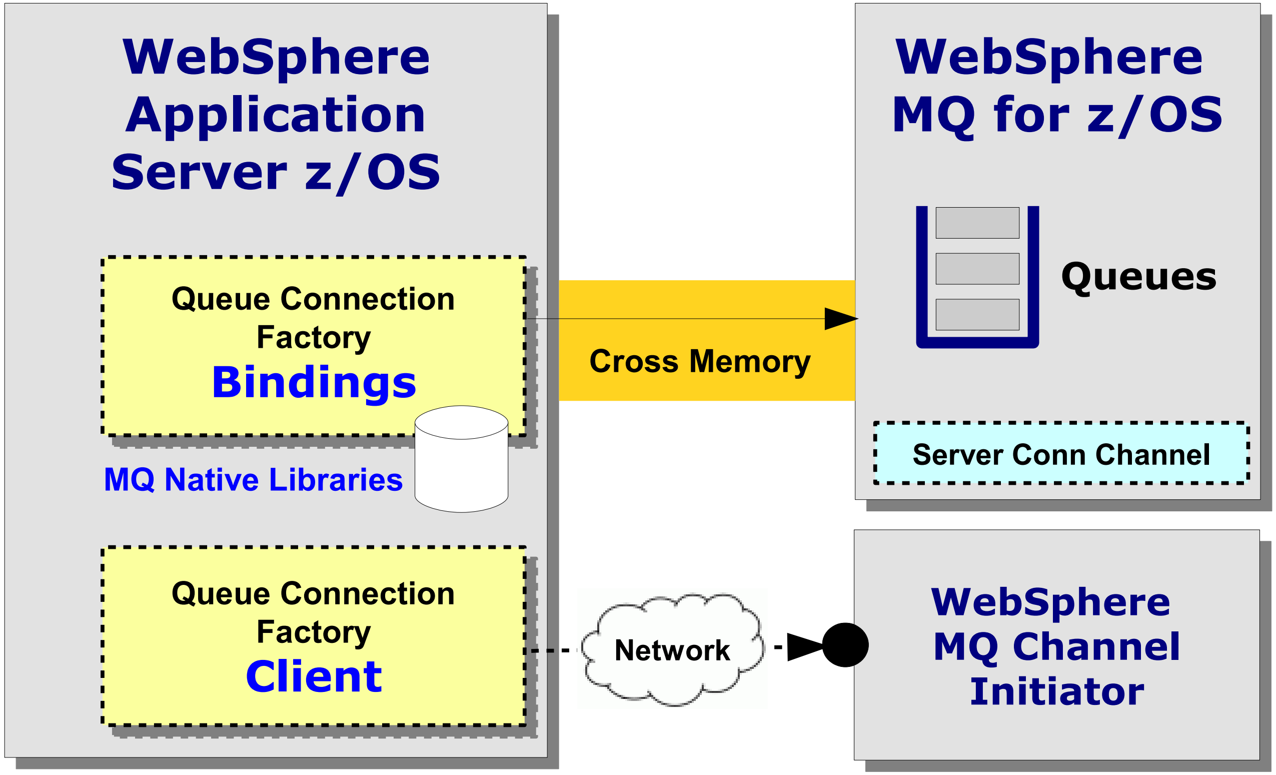
Applications may require multiple queues so it is common to have multiple JMS queue definitions

Bindings, client …

Bindings Mode and Client Mode

The "local" and "remote" theme is carried to MQ connections as well ...

When configuring bindings you'll need to provide the path to the native shared object libraries as well as STEPLIB or LINKLIST to the SCSQANLE, SCSQANLU and SCSQAUTH data sets

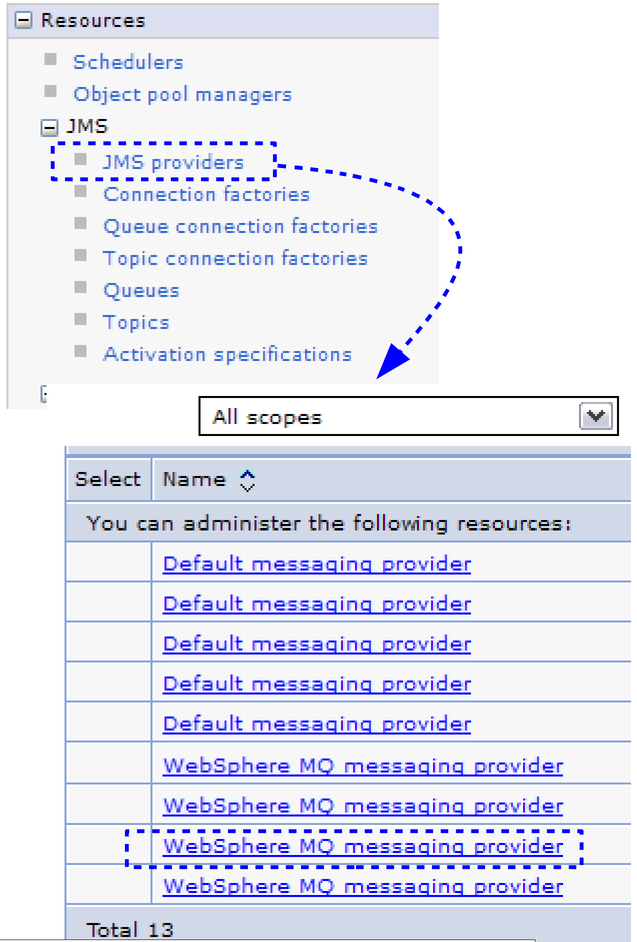


When configuring client you'll need to know the host and port of the MQ channel initiator as well as the Server Connection Channel to use when connecting

JMS MQ provider ...

Configuring the JMS Provider

The provider is very simple to configure ... the key is the connection factories and the queue definitions are access from the provider properties screen:



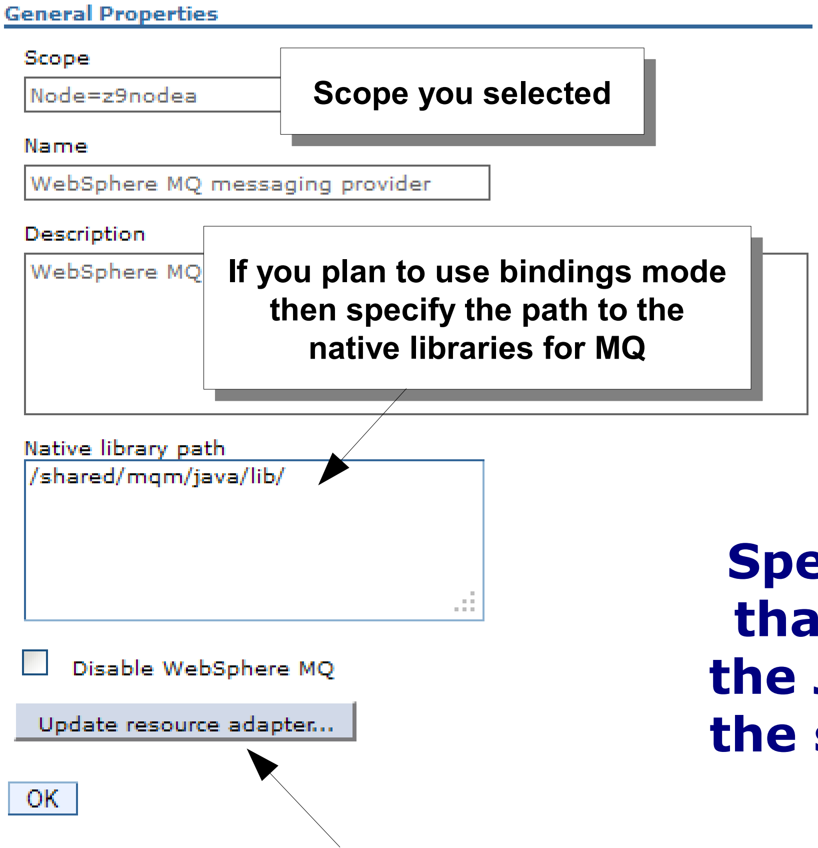
⊟ Resources
- Schedulers
- Object pool managers

⊟ JMS
- JMS providers
- Connection factories
- Queue connection factories
- Topic connection factories
- Queues
- Topics
- Activation specifications

All scopes

Select	Name
You can administer the following resources:	
	Default messaging provider
	Default messaging provider
	Default messaging provider
	Default messaging provider
	Default messaging provider
	WebSphere MQ messaging provider
	WebSphere MQ messaging provider
	WebSphere MQ messaging provider
	WebSphere MQ messaging provider
Total 13	

Pick the provider link associated with the scope you wish ... cell, node or server level



General Properties

Scope: Node=z9nodea

Name: WebSphere MQ messaging provider

Description: WebSphere MQ

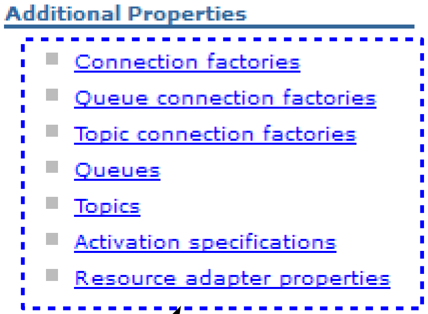
Native library path: /shared/mqm/java/lib/

□ Disable WebSphere MQ

Scope you selected

If you plan to use bindings mode then specify the path to the native libraries for MQ

InfoCenter indicates not to use this "update" button unless directed by IBM support. Normal WAS maintenance brings in updates to JMS MQ provider



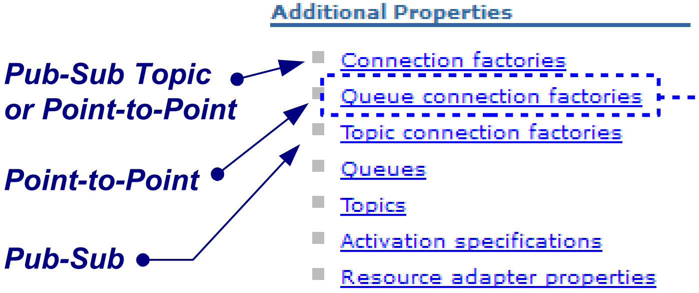
Additional Properties
- Connection factories
- Queue connection factories
- Topic connection factories
- Queues
- Topics
- Activation specifications
- Resource adapter properties

Specific definitions that will fall under the JMS provider for the scope you chose

Queue connection factories ...

Queue Connection Factories

These provide the information on how to connect to the MQ queue manager:



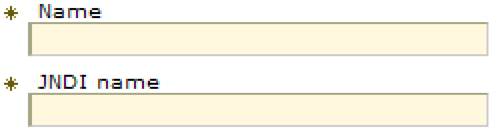
Pub-Sub Topic or Point-to-Point

Point-to-Point

Pub-Sub

Additional Properties

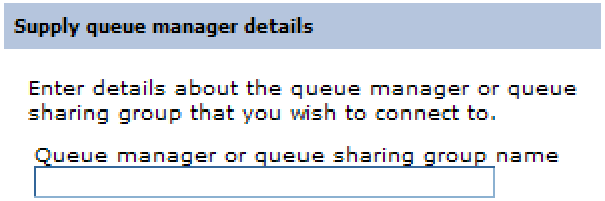
- Connection factories
- Queue connection factories
- Topic connection factories
- Queues
- Topics
- Activation specifications
- Resource adapter properties



* Name

* JNDI name

Provide a display name and a JNDI name

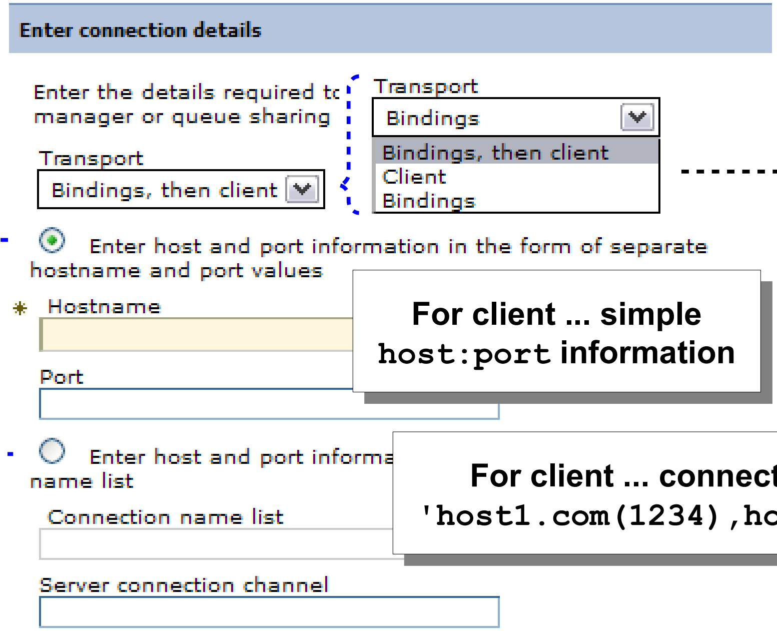


Supply queue manager details

Enter details about the queue manager or queue sharing group that you wish to connect to.

Queue manager or queue sharing group name

Provide the name of the QMGR or the QSG



Enter connection details

Enter the details required tc manager or queue sharing

Transport
Bindings, then client

Transport
Bindings
- Bindings, then client
- Client
- Bindings

○ Enter host and port information in the form of separate hostname and port values

* Hostname

Port

or

○ Enter host and port informa name list

Connection name list

Server connection channel

For client ... simple `host:port` information

For client ... connection name list: `'host1.com(1234),host2.com(4321)'`

For client ... server connection channel to use

If Bindings ...

Then just the Queue Manager name is needed. The host and port fields are grayed-out.

If Client ...

Choose either host/port or connection list, then supply the server connection channel information

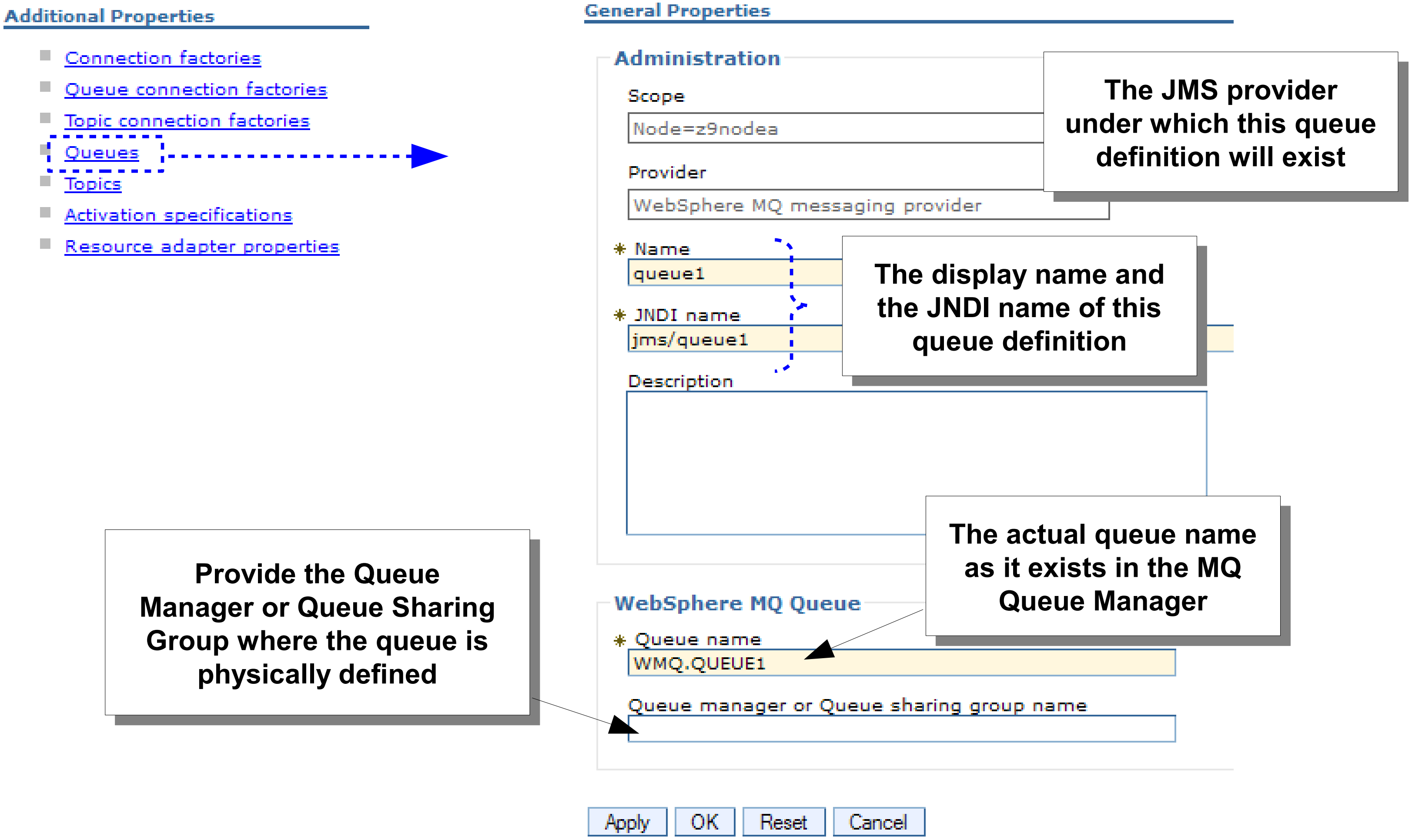
If Bindings, then client ...

Choose either host/port or connection list, then supply the server connection channel information

Queue definitions …

Queue Definitions

You would have as many queue definitions as you have queues in MQ that you wish applications to use:


Additional Properties
Connection factories
Queue connection factories
Topic connection factories
Queues
Topics
Activation specifications
Resource adapter properties

General Properties

Administration

Scope: Node=z9nodea

Provider: WebSphere MQ messaging provider

* Name: queue1

* JNDI name: jms/queue1

Description:

WebSphere MQ Queue

* Queue name: WMQ.QUEUE1

Queue manager or Queue sharing group name:

The JMS provider under which this queue definition will exist

The display name and the JNDI name of this queue definition

The actual queue name as it exists in the MQ Queue Manager

Provide the Queue Manager or Queue Sharing Group where the queue is physically defined

Application JNDI lookups …

Application Access to JMS Resources

This involves mapping the `<resource-ref>` tags in the application deployment descriptors to the JMS definitions you've created:

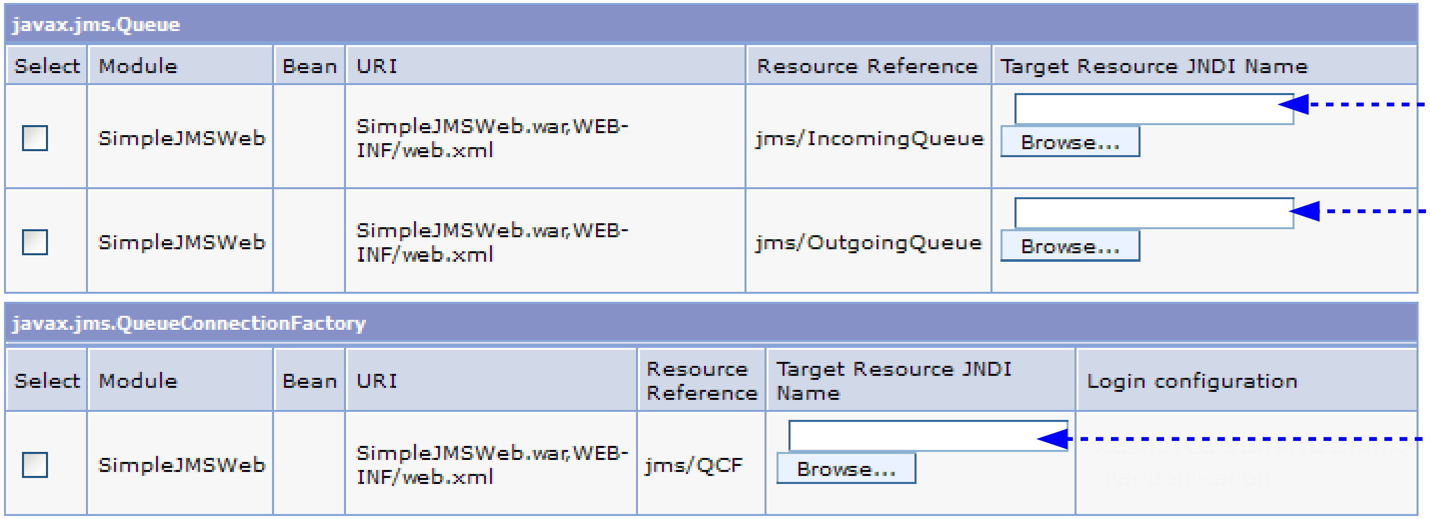
SimpleJMS.ear

Messaging App Used in Upcoming Lab

```
<resource-ref id="ResourceRef_1074045272521">
   <res-ref-name>jms/QCF</res-ref-name>
   <res-type>javax.jms.QueueConnectionFactory</res-type>
</resource-ref>
<resource-ref id="ResourceRef_1074045272531">
   <res-ref-name>jms/IncomingQueue</res-ref-name>
   <res-type>javax.jms.Queue</res-type>
</resource-ref>
<resource-ref id="ResourceRef_1074045272551">
   <res-ref-name>jms/OutgoingQueue</res-ref-name>
   <res-type>javax.jms.Queue</res-type>
</resource-ref>
```

Step 6 Map resource references to resources

Admin Console prompts you to map resource refs to the JMS JNDI definitions created



javax.jms.Queue

Select	Module	Bean	URI	Resource Reference	Target Resource JNDI Name
□	SimpleJMSWeb		SimpleJMSWeb.war,WEB-INF/web.xml	jms/IncomingQueue	Browse...
□	SimpleJMSWeb		SimpleJMSWeb.war,WEB-INF/web.xml	jms/OutgoingQueue	Browse...

javax.jms.QueueConnectionFactory

Select	Module	Bean	URI	Resource Reference	Target Resource JNDI Name	Login configuration
□	SimpleJMSWeb		SimpleJMSWeb.war,WEB-INF/web.xml	jms/QCF	Browse...	

© 2013 IBM Corporation