

WebSphere Application Server for z/OS

Introducing the IBM Support Assistant for WAS on z/OS

Setting up the IBM Support Assistant (ISA) Diagnostic Tools for WebSphere Application Server
on z/OS

A document planned for on-going updates as additional tools are discovered and understood

This document can be found on the web at:
www.ibm.com/support/techdocs
Search for document number **WP101575** under the category of "White Papers"

Version Date: March 9, 2010

See "Document Change History" on page 37 for a description of the changes in this version of the document

Written and provided by the WebSphere Application Server for z/OS team at the
IBM Washington Systems Center

Many thanks to **Pete Robbins, Howard Hellyer, and Steve Poole** on the **IBM Monitoring and Diagnostic Tools for Java** team, and **Russell Wright** on the **IBM Support Assistant Tools** development team.

Also to **Louis Wilen** of the Washington Systems Center, and **Ed McCarthy** from TechWorks, IBM AP Software Group.

The WebSphere Application Server for z/OS support team at the Washington Systems Center consists of: **John Hutchinson, Mike Kearney, Louis Wilen, Lee-Win Tai, Steve Matulevich, Mike Loos** and **Don Bagwell**.

Mike Cox, Distinguished Engineer, serves as technical consultant and advisor for all of our activities.

For questions or comments regarding this document, send e-mail to John Hutchinson at hutchjm@us.ibm.com

Table of Contents

Executive Overview	4
Diagnostic Tools in the ISA	4
Overview of Diagnostic Tools for z/OS	5
Installing the ISA and Tool Add-ons	6
Individual ISA Tool Discussions	7
Real time diagnostic tools	7
Health Center (IBM Monitoring and Diagnostic Tools for Java)	7
Garbage Collection analysis tools	9
Steps to Gather Garbage Collection Measurement Data	9
GC and Memory Visualizer (IBM Monitoring and Diagnostic Tools for Java) - GCMV	10
Pattern Modeling Tool for Java GC (PMAT)	12
Memory Dump analysis tools	13
Memory Dump Diagnostic for Java – MDD4J	14
Memory Analyzer (IBM Monitoring and Diagnostic Tools for Java).....	17
Dump Analyzer (IBM Monitoring and Diagnostic Tools for Java)	20
Here is an example of the Default Dump Report”.....	22
HeapAnalyzer	23
Thread & Monitor Dump Analyzer for Java - TMDA	24
Trace analysis tools	25
Trace and Request Analyzer for WebSphere Application Server.....	25
Log Analyzer.....	26
Configuration analysis tools	28
Visual Configuration Explorer - VCE	28
Working with Memory Dumps on z/OS	31
Java Dumps.....	31
z/OS System (SVC) Dumps	31
Other Dumps	31
zOSMF	32
Working with Traces on z/OS	33
WebSphere Traces.....	33
ffdc logs – "first failure data capture"	33
wsadmin.sh traces.....	33
Configuration script logs	33
JDBC traces	33
Browsing log files of traces in ascii.....	33
Additional tracing documentation	34
Performance Observations and Memory Requirements	35
ISA Documentation and other Resources	35
Other resources:.....	35
Appendix A – Other Diagnostic Tools	36
WebSphere “built-in” tools	36
z/OS Diagnostic Tools – See the z/OS InfoCenter or Library.....	36
JinsightLive for IBM System z	36
Details of Document Change History	37

Executive Overview

Many valuable diagnostic tools have been developed to aid in trouble-shooting WebSphere Server and Java applications. Recently, many of them have been provided through the complimentary M Support Assistant (ISA) for use on your workstation. This paper will examine the tools that work with WebSphere on z/OS, showing you how to install and configure them, and the basics on how to operate them.

Detailed instructions on how to effectively use them so solve problems may be found in other documents that we will point you to.

Diagnostic Tools in the ISA

The ISA workbench is actually a “Launch Pad” for many other tools, some of which were previously available as separate downloads from alphaWorks or developerWorks. Here are the Tools described in this paper. Four of them are provided by the IBM Centre for Java Technology Development for testing Java 2 applications. See the home page for IBM Monitoring and Diagnostic Tools for Java™ - <http://www.ibm.com/developerworks/java/jdk/tools/index.html>

See the ISA Tools Add-ons List for a complete list of diagnostic tools available for ISA.

Real-time analysis

- Health Center (IBM Monitoring and Diagnostic Tools for Java™)

Garbage Collection analysis

- GC and Memory Visualizer (IBM Monitoring and Diagnostic Tools for Java – GCMV)
- Pattern Modeling Tool for Java GC (PMAT)

Dump analysis

- Memory Analyzer (IBM Monitoring and Diagnostic Tools for Java)
- Dump Analyzer (IBM Monitoring and Diagnostic Tools for Java)
- Memory Dump Diagnostic for Java (MDD4J)
- HeapAnalyzer (*New!*)
- Thread & Monitor Dump Analyzer for Java (TMDA)

Trace analysis

- Trace and Request Analyzer for WAS
- Log Analyzer

Configuration analysis

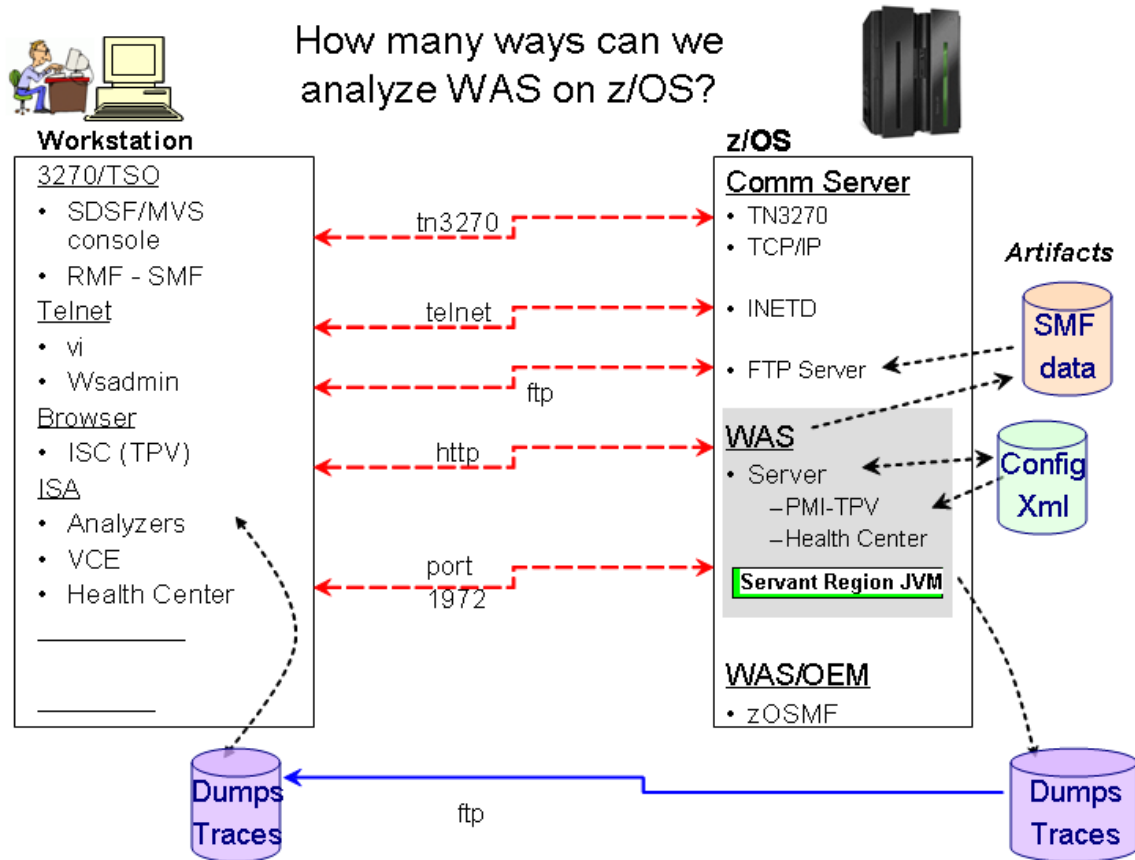
- Visual Configuration Explorer (VCE)

ISA also has some “data gathering” functions, but they are not yet appropriate for z/OS.

(There are also many more tools available, which we have not studied, or are not applicable to the z/OS environment.)

Overview of Diagnostic Tools for z/OS

To put these ISA tools in perspective, here is a picture of many more tools available to access diagnostic data from your workstation:



As you can see, there are many more tools than just the ones that come with the IBM Support Assistant. Here is a quick list of many tools and what kind of diagnostic data they can access:

TSO/ISPF/SDSF Applications

- Display Active WAS Servers and browse logs on JES SPOOL (incl. RMF/SMF performance data)
- Issue MVS Display and Modify commands and view the results and messages in SYSLOG
- Turn Java tracing on and off, and view trace logs.
- RMF - Monitors to view performance data from SMF
- IPCS – Interactive Problem Control System
- DB2 Administrative Utility – Browse and manage DB2 objects

Telnet Client

- View logs & traces in the HFS filesystem, & Rumage around in the Configuration HFS (xml files)
- UNIX commands, netstat, FTP, wsadmin.sh scripting

Web Browsers

- ISC (Integrated Solutions Console) - Monitor and Activate tracing, Performance Viewer
- z/OS Management Facility (zOSMF) – PD Incident Log Manager & Comm Server Configuration tool

ISA Tools

- Real-time analyzer - Health Center
- GC verbose trace analyzers
- Memory/Dump analyzers for java heap, core dumps and MVS SVC dumps
- Trace and Log Analyzers
- Visual Configuration Explorer (VCE)

Installing the ISA and Tool Add-ons

The installation of the ISA and its analysis tools is well documented in several websites shown below, so we will not repeat the instructions here.

Download from <http://www.ibm.com/software/support/isa/>

The IBM Education Assistant (IEA) has several tutorials on installing and using the ISA at: http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.selfassist/selfassist/ISAv41_Task.html

Once you have the ISA installed on your workstation, it is quite easy to download specific diagnostic tools as “add-ons”. There are over 100 Product Add-ons and a growing list of tool add-ons.

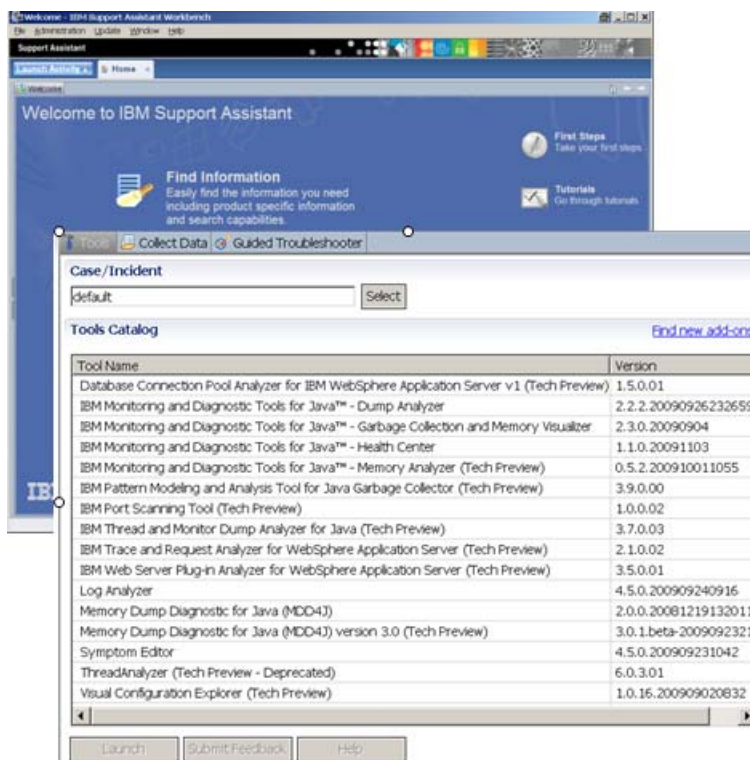
From the Support Assistant Workbench, click **Update > Find New... > Tools Add-ons**.

In the window that pops up, select the tools that you would like to install and follow the on-screen instructions to complete the installation. When the installation is completed, the ISA Workbench will restart.

There are also “Product Add-ons” which can be installed for WebSphere,

Reference: “How to Install and Run Tools through the IBM Support Assistant:” at <http://www.ibm.com/support/docview.wss?uid=swg27013279>

Here is the Home page and launch page for “analyzing problems”:



Individual ISA Tool Discussions

This section describes how to install and use the tools we found useful for WAS on z/OS.

Author Comments:

- There are many more tools available with ISA, which are not applicable to the z/OS environment.
- The ISA also has some “data gathering” functions, but they are not yet appropriate for z/OS.
- These tools are from many different developers, and have unique user interfaces for accessing artifacts and producing reports.

Real time diagnostic tools

The Health Center diagnostic tool is the only real-time ISA tool. Other tools in this category are:

- The performance viewer in the ISC
- SDSF and RMF from a TSO/ISPF session

Health Center (IBM Monitoring and Diagnostic Tools for Java)

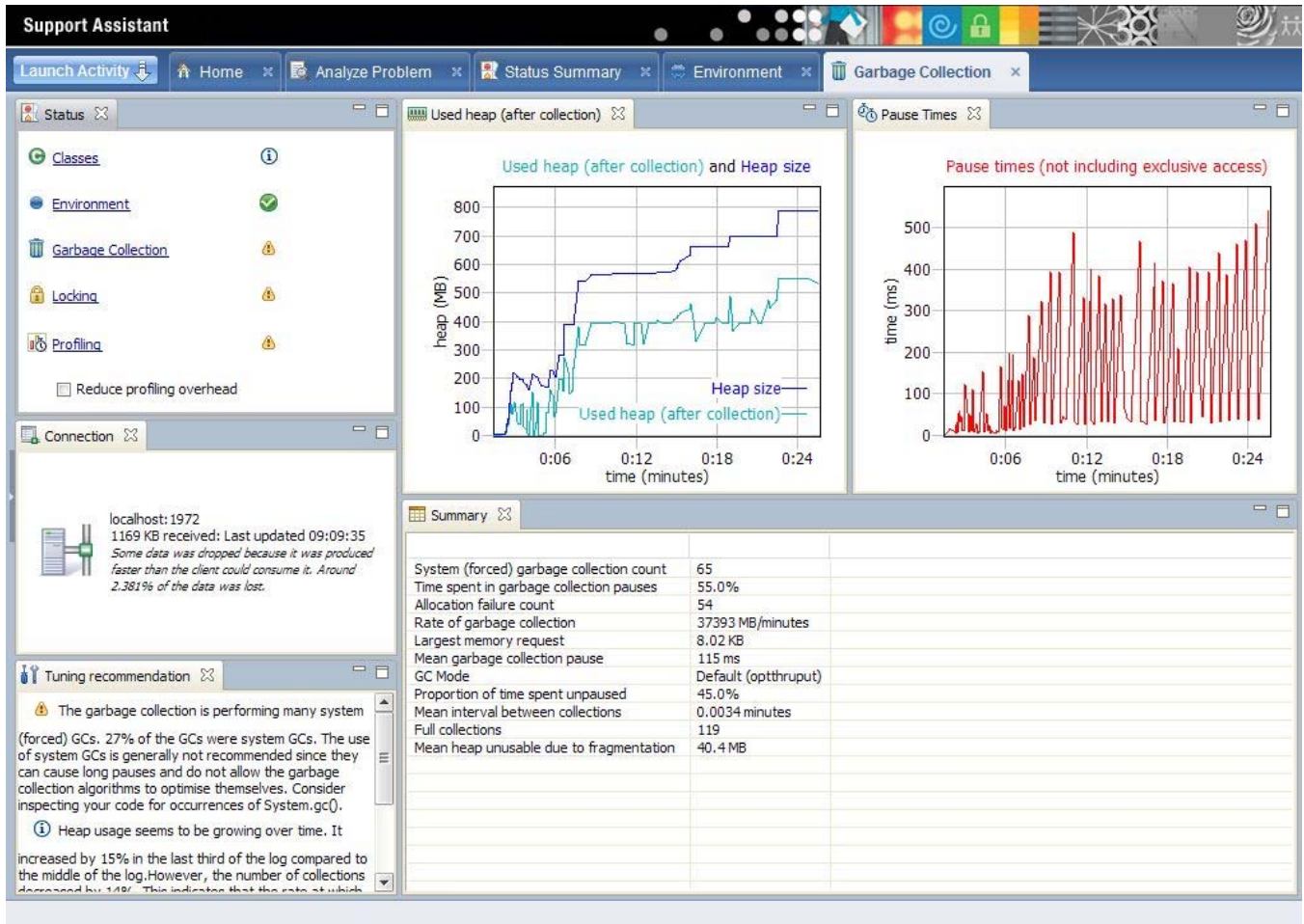
The IBM Monitoring and Diagnostic Tools for Java - Health Center is a lightweight tool that monitors active IBM Virtual Machines for Java with minimal performance overhead. The Health Center suggests live tuning recommendations for garbage collection, profiles methods including call stacks, and highlights contended locks.

Since this tool allows you to examine live, running WebSphere servers, you must set this up on your workstation, AND enable your WebSphere server for monitoring.

To install the Health Center tool Add-on, we recommend you consult these documentation resources:

- IBM WebSphere Support Authority article at www.ibm.com/developerworks/websphere/techjournal/0901_supauth/0901_supauth
- YouTube videos
 - Overview of Health Center features: www.youtube.com/watch?v=5Tcktcl0qxs
 - Install IBM Support Assistant and Health Center: www.youtube.com/watch?v=6WjE9U0jvEk
 - How to enable a Java application for live monitoring by the Health Center: www.youtube.com/watch?v=Hdp0mJ13NLQ
- Enabling the Health Center in a WAS server is easy if your JDK is recent (Java6 SR5, or Java5 SR10 and above):
 - Add the `-Xhealthcenter` property to the servant JVM properties.
- Using the Health Center is quite easy; launch it from the ISA, and specify the host address the port number (1972 is the default) and un-check “Scan for 100 Ports.”
- Views:
 - Classes – timeline shows the class loading frequency.
 - Environment – Verify variables and Classpath settings
 - Garbage Collection – Graphs show Heap use, size, and Pause times.
 - Locking – Identify Lock Contention and Average Lock Holding times.
 - Profiling – Lower overhead than Jinsight Live for z, but not as rich.

Here is a screen-shot of the Garbage Collection view:



For additional guidance on using these views to diagnose problems, see the following:

- ISA Help: “Tool: IBM® Monitoring and Diagnostic Tools for Java™ - Health Center”
- developerWorks article: <http://www.ibm.com/developerworks/java/library/j-ibmtools5/>
- Presentation “Low overhead performance monitoring for your JVM with IBM Monitoring and Diagnostic Tools for Java - Health Center” by Brian Peacock, IBM JTC: http://www.websphereusergroup.org.uk/uploadedfiles/mtg26/35_Consumability_Tools.pdf

Presentation “IBM Monitoring and Diagnostic Tools for Java” by Dave Nice, IBM SWG-PTC: www.ibm.com/support/docview.wss?uid=swg27016069&aid=1

Garbage Collection analysis tools

Garbage collection (GC) in the JVM can be a significant bottleneck in your application server. Turning on verboseGC in the WAS server's servant address space will create messages in the SYSOUT DD file (by default) that can help you tune the heap size and GC policy to improve performance. You can view these messages or use these ISA tools to visualize them. In addition to these tools, MDD4J can analyze java heap dumps and help identify causes of memory leaks.

These ISA tools are available to help visualize these reports:

- GC and Memory Visualizer (IBM Monitoring and Diagnostic Tools for Java) - GCMV
- Pattern Modeling Tool for Java GC - PMAT

Other tools outside the ISA include:

- Performance viewer in the ISC (Integrated Solutions Console)

Steps to Gather Garbage Collection Measurement Data

To enable visualization or analysis of your JVM heap size and garbage collection activity with these tools, perform the following steps:

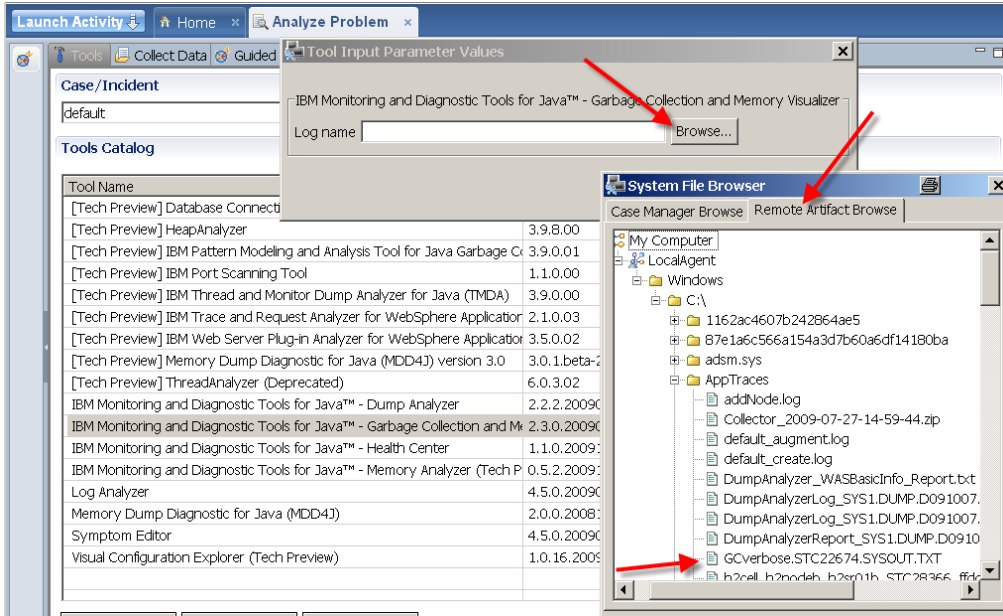
- Enable verbose GC in your ISC: Application servers -> <server_name> -> Process definitions -> Servant -> Java Virtual Machine , and click on the box labeled "Verbose garbage collection". Click "OK", Save the changes, and re-start your server.
- Verify the verbose GC setting using SDSF by looking for the <af and <gc xml structures in the servant's SYSOUT log.
- Drive an application in your server. (Use your own, or the sample applications provided in this paper.) You can use Jmeter, the MS Web Application Stress tool or your own to drive a load.
- At the end of the test, copy the verboseGC output from the SYSOUT log to a file (in ASCII) on your workstation.
- Launch one of the GC verbose visualizers (GCMV or PMAT) and browse to the file you uploaded.

See Techdoc WP101612 "*Getting started with analysis of GC, Heapdumps and Javacores For WebSphere on z/OS*" for a good white paper to help you gather verbose GC and use these tools.

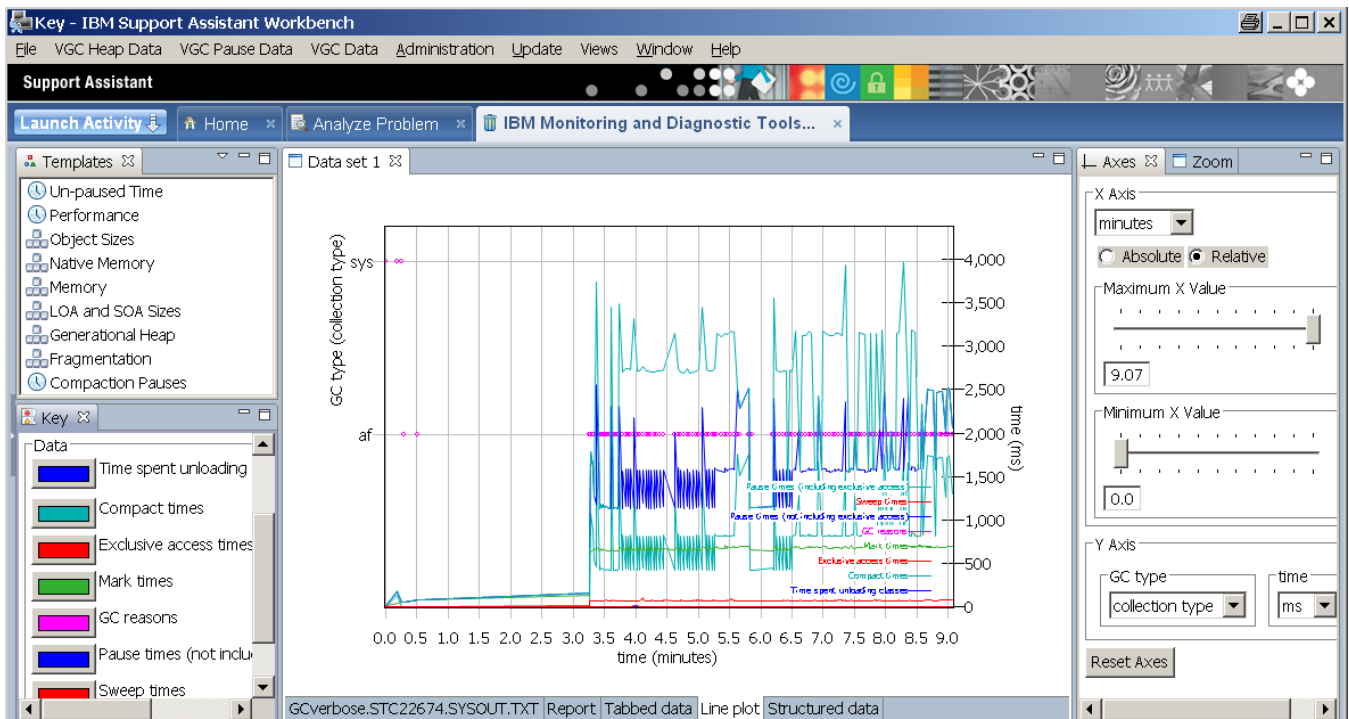
GC and Memory Visualizer (IBM Monitoring and Diagnostic Tools for Java) - GCMV

IBM Monitoring and Diagnostic Tools for Java™ - Garbage Collection and Memory Visualizer (GCMV) provides analysis and views of your application's verbose gc output. GCMV displays the data in both graphical and tabulated form. It provides a clear summary and interprets the information to produce a series of tuning recommendations, and it can save data to HTML, JPEG or .csv files (for export to spreadsheets). Multiple logs can also be viewed and compared. See ISA “Help” and <http://www.ibm.com/developerworks/java/jdk/tools/gcmv> for more information.

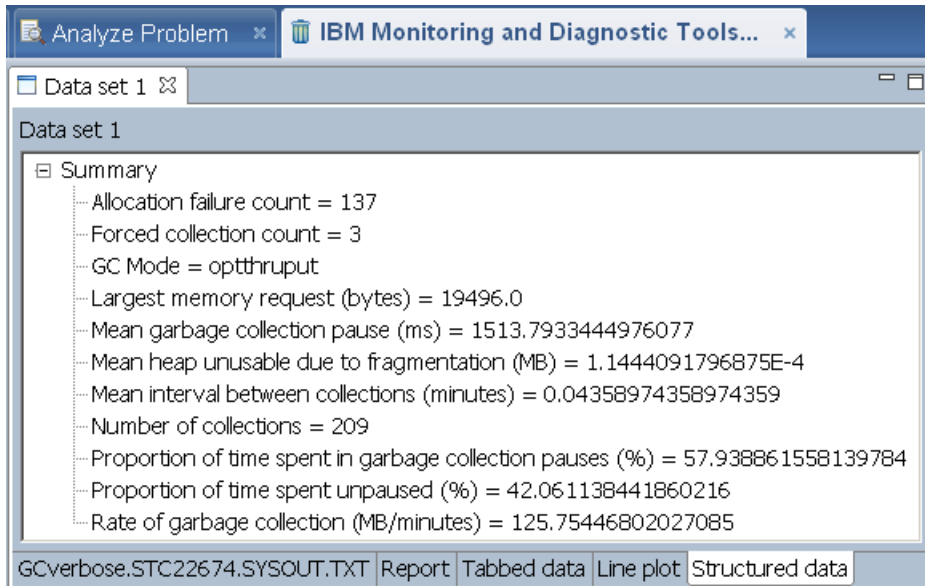
- Launch GCMV from the ISA, and open the file you uploaded.



- Click on “Next >” to see the Performance Line plot:



- Click on the “Summary” tab to see the summary:



GCMV provides interfaces to analyze the data to "drill down" into the causes of trends or data points of interest. The graphical interface provides the following capabilities:

- View the raw log, tabulated data and graphs
- Zoom and crop graphs
- Select data points in line plots and view in raw data
- Customize the graph by adding/removing data and changing display units
- Compare output from multiple logs
- Save data to jpeg or .csv files for export to spreadsheets, or for future snalysis with GCMV.

Pattern Modeling Tool for Java GC (PMAT)

IBM Pattern Modeling and Analysis Tool for Java Garbage Collector (PMAT) parses IBM verbose GC trace, analyzes Java heap usage, and recommends key configurations based on pattern modeling of Java heap usage. Only verbose GC traces generated from IBM JDKs are supported. (PMAT is a Technology Preview.)

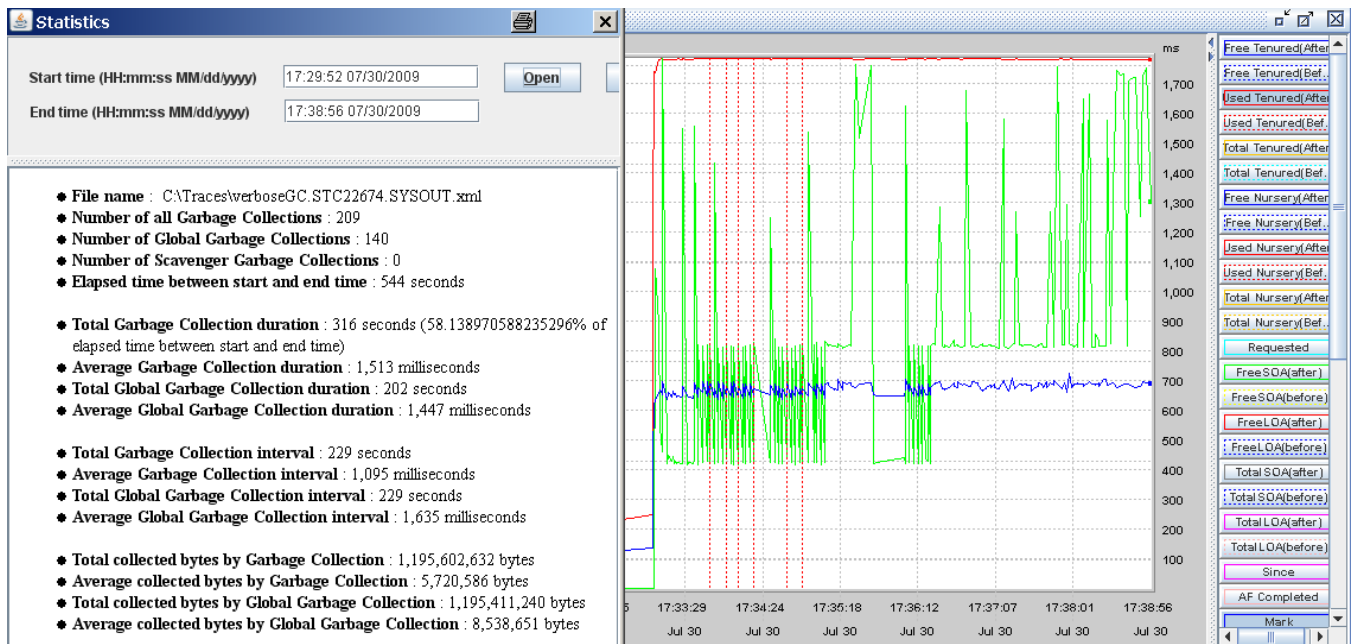
This tool provides a different perspective from GCMV that can be useful in visualizing garbage collection. It provides several views, such as:

- GC analysis
- GC Table View
- AF summary
- GC usage summary
- GC duration summary
- GC Graph View
- GC trend analysis
- Zoom in/out/selection/center of graph view

Launch PMAT and browse to the file you uploaded (similar to the “Browse” or “File > Open” tabs in GVMV).

Note: PMAT does not run inside the ISA workspace as GCMV does, but launches a separate window for analysis.

Here is a screen-shot showing a summary and graph of the activity from PMAT:



Further information is available in a Webcast replay: "How to analyze verbosegc trace with IBM Pattern Modeling and Analysis Tool for IBM Java Garbage Collector" at <http://www.ibm.com/support/docview.wss?uid=swg27007240>

Memory Dump analysis tools

Various forms of memory dumps from your JVM heap or z/OS system can be analyzed to help you diagnose memory problems and tune your java runtime. Here are the kinds of dumps that can be created from a WebSphere server on z/OS. See the section at the end of this paper titled “**Working with Dumps on z/OS**”

Java Dumps:

- Heapdump
- Javacore (this is in plain human-readable text)
- JavaTDUMP
- Snap

z/OS Dumps

- SYSABEND
- WebSphere ABEND dumps
- CEE Dumps
- System (SVC) Dumps

This paper introduces the following tools available with the ISA. (Most of these deal with only one type of dump.)

- Memory Dump Diagnostic for Java (MDD4J) - Heapdumps
- Memory Analyzer (*IBM Monitoring and Diagnostic Tools for Java*) – Heapdumps, System dumps
- Dump Analyzer (*IBM Monitoring and Diagnostic Tools for Java*) - System dumps
- HeapAnalyzer (*New for ISA! Previously available on AlphaWorks*) - Heapdumps
- Thread & Monitor Dump Analyzer for Java (TMDA) - Javacores

Also see Techdoc WP101612 “*Getting started with analysis of GC, Heapdumps and Javacores For WebSphere on z/OS*” for a good white paper to help you use these tools.

Another tool that analyzes CTRACES and System Dumps is IPCS. See “*Formatting CTRACE data with an IPCS dialog*” in the WebSphere Application Server for z/OS Informaiton Center.

Memory Dump Diagnostic for Java – MDD4J

This tool can be used to provide a first pass analysis of the contents of a heap dump (or a comparison between two heap dumps), and to identify the most common causes of memory leaks. It also provides several high-level analysis functions to investigate the overall footprint of a JVM and provide indications on how to reduce that footprint.

There are different “types” of analyses:

- “Baseline” heap dump – taken when application is started.
- “Primary” heap dump – when a problem occurs.
- “Comparative” heap dumps – analyze differences between two heap dumps.

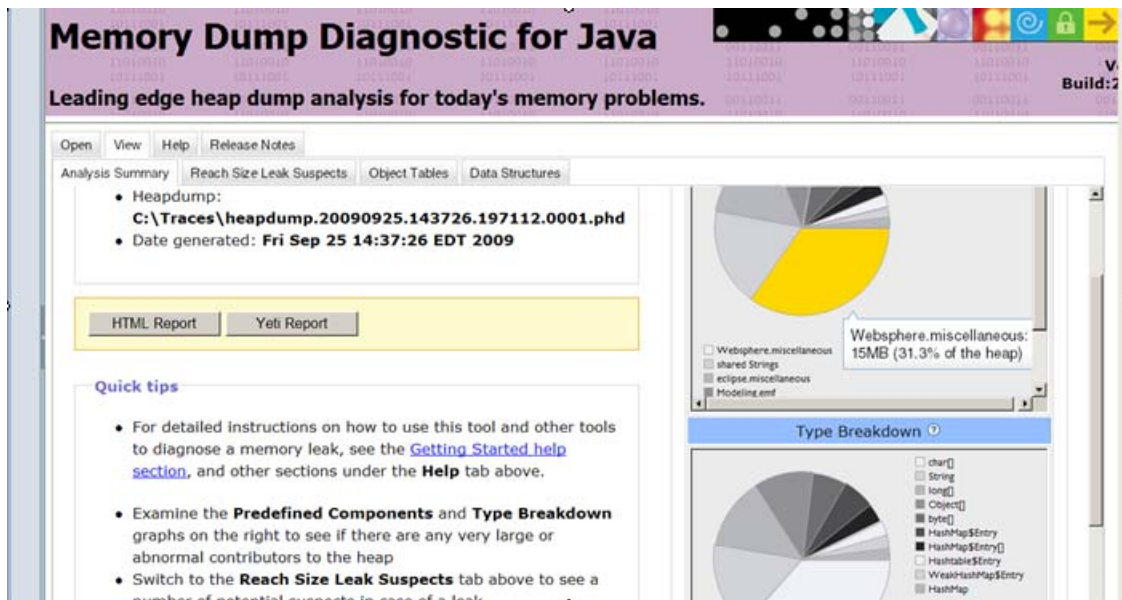
This tool is not intended to provide in-depth, low-level, examinations of the entire object graph within the JVM. For this, we recommend that you use a tool such as the Memory Analyzer.

There are some terms you need to understand when using MDD4J:

- “Leak Root” – object holding a reference to chain of objects leading to the “leak container”
- “Leak Container” – object owning all “leaking objects”
- “Leaking Unit” - Object in data structure with multiple instances present
- “Owner Chain” – chain of objects starting from a leak root object to leak container?
- “Contents” – data responsible for consuming most of the heap
- “Reach Size” – sum of objects reachable from a given object
- “Drop in Reach Size” – difference between the reach size of an object and reach size of a child

There are two versions of MDD4J available with the ISA: V2 and V3 (tech preview).

Here is a sample screen shot of the summary report from MDD4J V.3:



Guided Activity Assistant.

- “Summary” – Basic Heap Info & Contents Summary.
- “Reach Size Leak Objects” – Cumulative size of all objects.
- “Object/Class” leak suspects.

- “Data Structures view” – Understand relations & Track Changes.
 “Growing”, “Shrinking”, & “Steady” helpful when comparing 2 heap dumps

Reports: (HTML or Yeti)

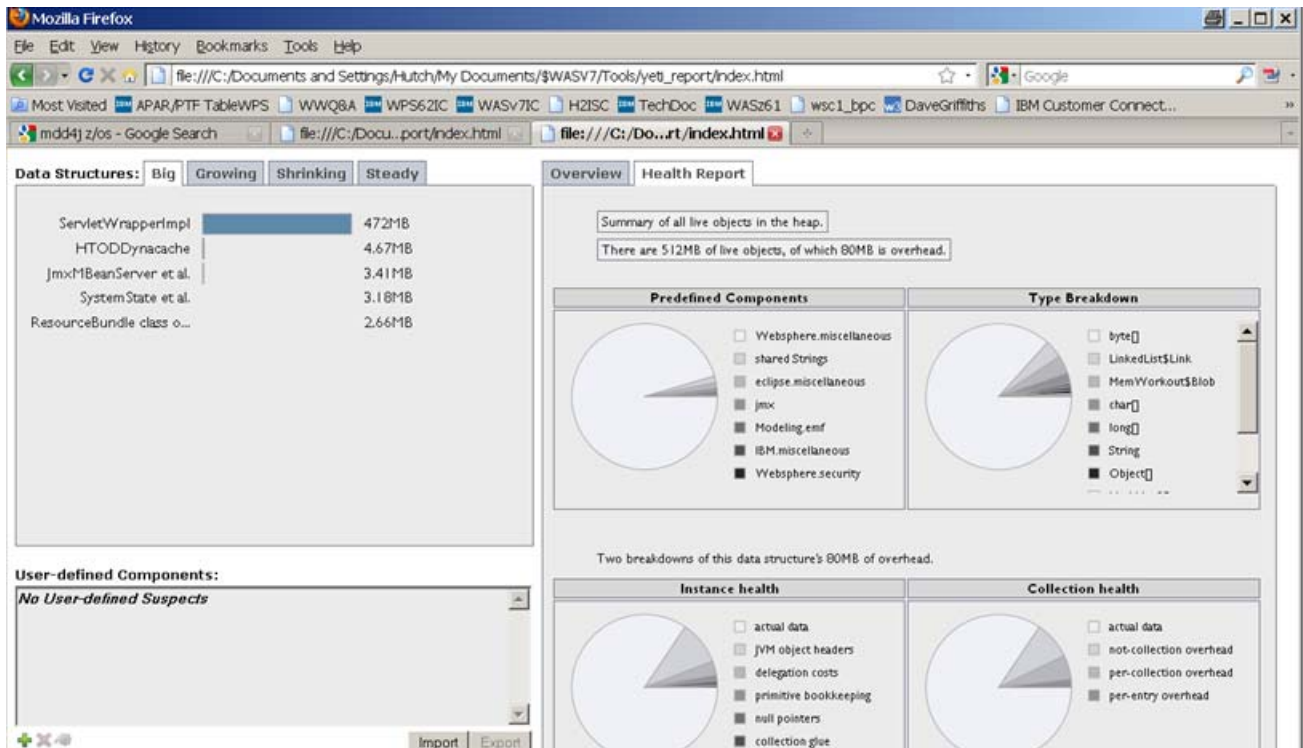
- Save the analysis as a single HTML file
- Save entire Yeti analysis in a zip file – then extract & open the index.html file.

For documentation see:

- www.ibm.com/developerworks/websphere/techjournal/0909_supauth/0909_supauth.html
- Techdoc WP101612 – “Getting started with analysis of GC, Heapdumps and Javacores For WebSphere on z/OS”

Yeti reports:

- MDD4J can also create a “Yeti Report” using an analysis engine based on “Yeti” technology. “Yeti” takes as input one or more Java heap snapshots, and produces reports showing categorizations and details of the data structures and objects in the JVM heap.



You can extract these reports as an html or zip file. (For a zip file, extract it and launch the index.html file.)

- Overview
- Health Report
- Content Graphs – Type & Field layout views

Analysis Summary Report

Here are some of the reports you can view in html form:

Primary heap file info:

C:\AppTraces\heapdump.20091013.140817.196909.0011.phd

Heap Contents Summary

Aggregated Data Structure Leak Suspects

Object/Class Leak Suspects

Big Data Structures

Whole Heap Health Measures

Big Data Structures

	Description	Size	Reachable Size	Shared Size
<input type="checkbox"/>	com.ibm.ws.security.web.WebAppCache class object	446kB	230MB	229MB
<input type="checkbox"/>	com.ibm.ws.webcontainer.servlet.ServletWrapperImpl	207MB	207MB	0 bytes
<input type="checkbox"/>	com.ibm.ws.cache.HTODDynaCache	8.16MB	8.16MB	0 bytes
<input type="checkbox"/>	com.ibm.ws.management.descriptor.MBeanDescriptorManager	191kB	5.36MB	5.17MB
<input type="checkbox"/>	org.eclipse.emf.ecore.impl.EPackageRegistryImpl class object	16kB	3.69MB	3.67MB
<input type="checkbox"/>	org.eclipse.emf.ecore.impl.EPackageRegistryImpl	1.47MB	3.02MB	1.55MB
<input type="checkbox"/>	com.ibm.ws.naming.ipbase.NameSpace	183kB	2.23MB	2.05MB
<input type="checkbox"/>	com.ibm.ws.naming.ipbase.NameSpace	47kB	2.11MB	2.07MB
<input type="checkbox"/>	com.ibm.ws390.orb.CommonBridge class object	360 bytes	1.75MB	1.75MB
<input type="checkbox"/>	javax.management.modelmbean.ModelMBeanOperationInfo	1.6MB	1.65MB	52kB

The analysis result includes the leak suspects and list of types and packages contributing the most to the heap size in a table format. It also provides a sophisticated data structure analysis view, based on the Yeti technology initially developed by IBM Research. For details about Yeti, see

[http://domino.research.ibm.com/comm/research_people.nsf/pages/nickmitchell.pubs.html/\\$FILE/yeti-ecoop2009.pdf](http://domino.research.ibm.com/comm/research_people.nsf/pages/nickmitchell.pubs.html/$FILE/yeti-ecoop2009.pdf)

If the simple approach offered by this tool does not suffice to diagnose your particular memory leak or other memory-related issue, we recommend that you follow-up with a more in-depth investigation using a special tool for this purpose, such as the Memory Analyzer Tool.

Memory Analyzer (IBM Monitoring and Diagnostic Tools for Java)

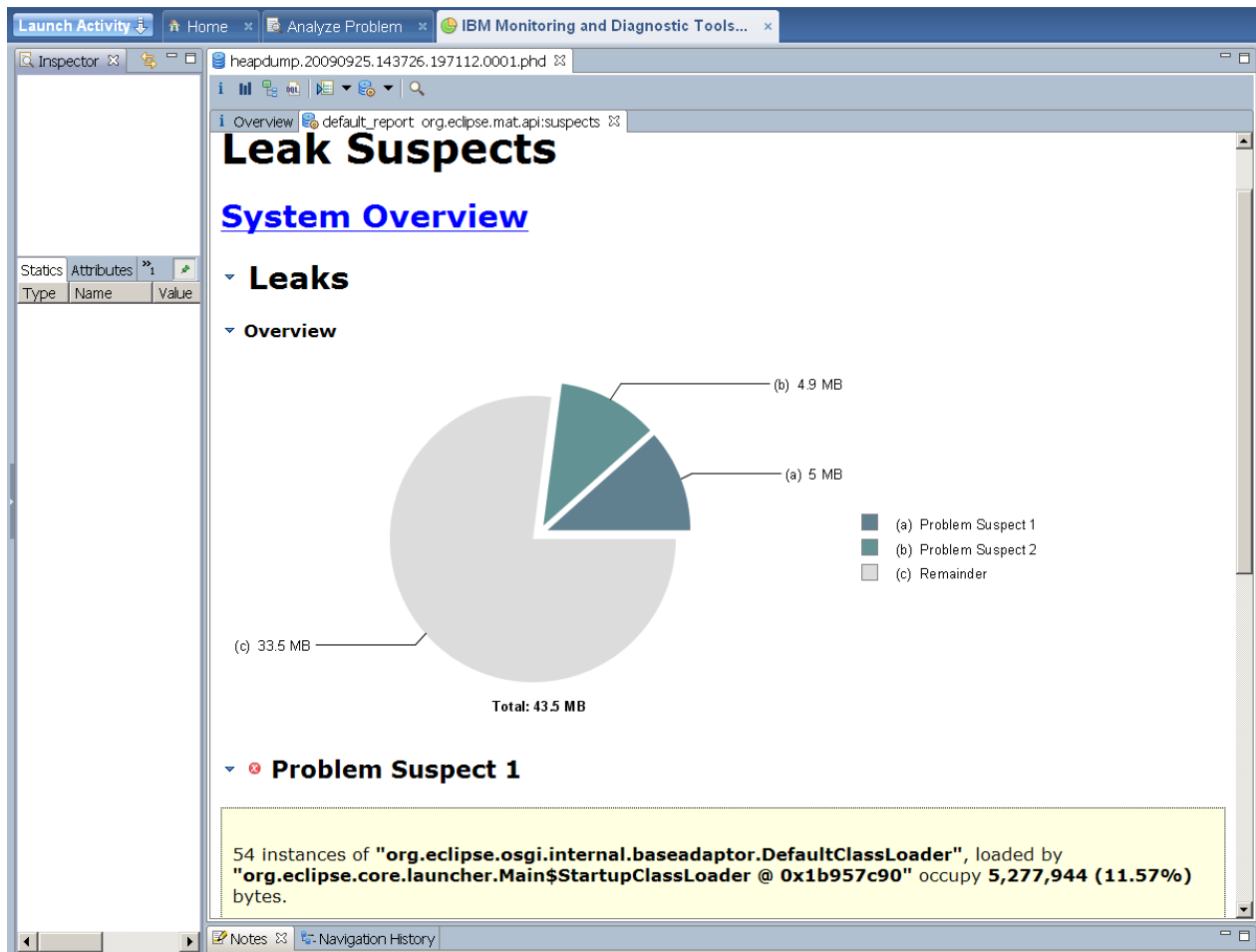
IBM Monitoring and Diagnostic Tools for Java™ - Memory Analyzer is a Java heap analyzer that helps you find memory leaks and reduce memory consumption. It works with java heapdumps (phd), gzipped heap dumps, and jextracted system dumps (z/OS SVC dumps processed by the jextract tool.) Javacore dumps are also handled, but there may not be as much useful information in them.

Note: This tool is based on the open source “Eclipse Memory Analyzer Tool” which is available from <http://www.eclipse.org/mat> and updated with the DTFJ Plug-in for the IBM JVM.

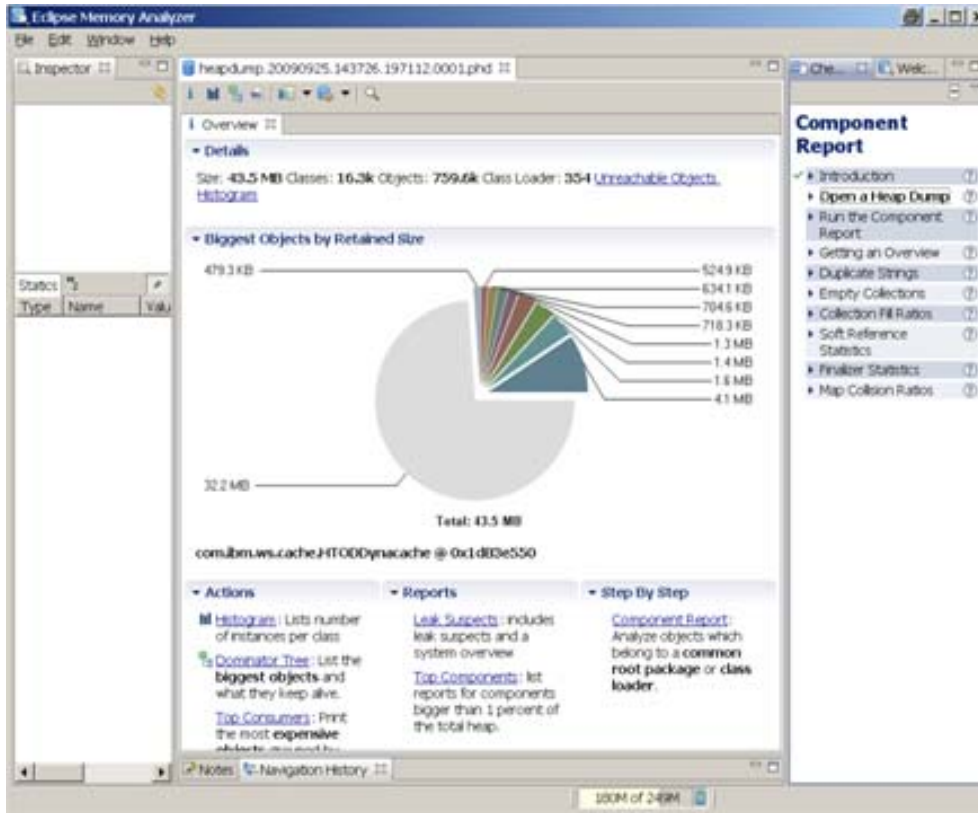
To analyze a memory dump, click on “File > Open > formats. From there, you can use the “Getting Started wizard” to view various reports:

- Overview: Leak suspects, Component reports
- Navigate: TOC, Fly over, Click on . . .
- Histogram (tables)
- Context (Incoming References, Outgoing References)
- Group (Package, Class, Class Loader)
- Dominator Tree - Objects keeping others
- Path to GC roots
- Leak reports

Here is an example of the Overview screen:



Here is an example of the “Biggest objects by Retained Size” screen:



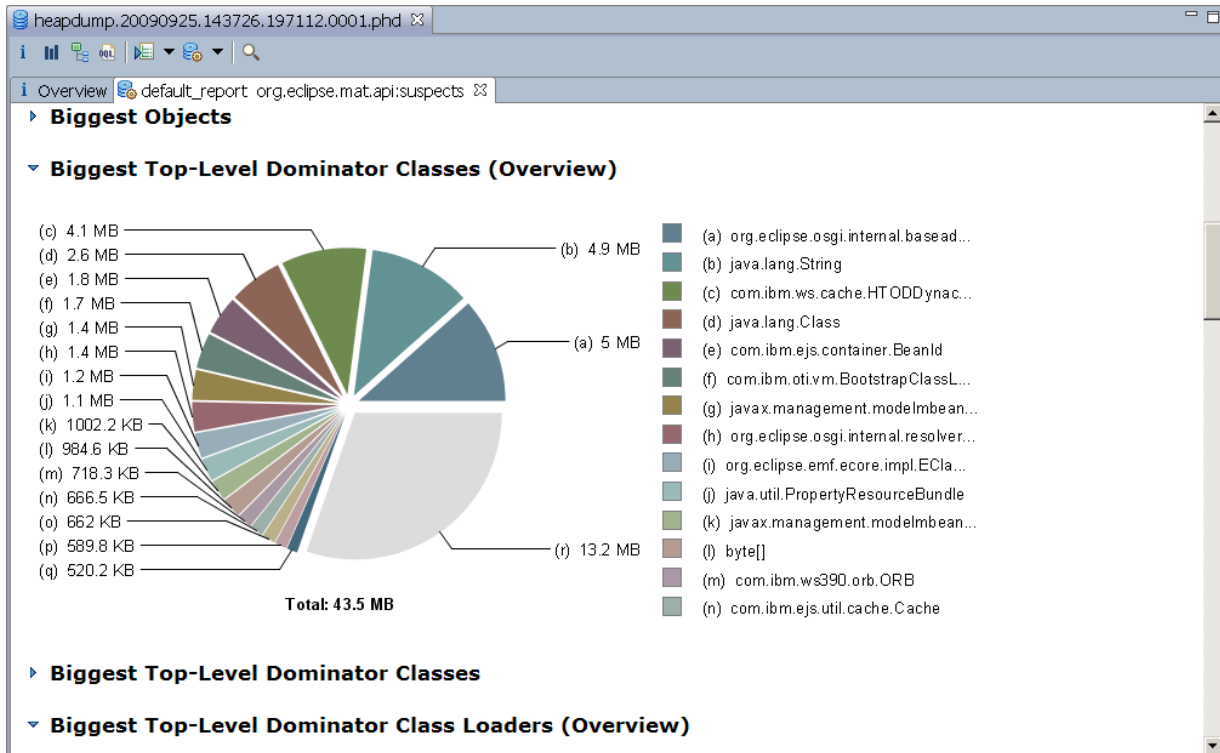
Here is list of several reports that can be found in the Table of Contents:

The screenshot shows the "Table Of Contents" page in the Eclipse Memory Analyzer. The table lists various reports and their sub-items:

- System Overview
 - Heap Dump Overview
 - System Properties
 - Thread Overview
- Top Consumers
 - Biggest Objects (Overview)
 - Biggest Objects
 - Biggest Top-Level Dominator Classes (Overview)
 - Biggest Top-Level Dominator Classes
 - Biggest Top-Level Dominator Class Loaders (Overview)
 - Biggest Top-Level Dominator Class Loaders
 - Biggest Top-Level Dominator Packages
- Class Histogram
- Leaks
 - Overview
 - Problem Suspect 1
 - Description
 - Problem Suspect 2
 - Description

The footer of the page reads "Table Of Contents" and "Created by Memory Analyzer".

Click on “Biggest Objects” to see a break-down:



Here is a sample use case where a servlet was (intentionally) leaking memory. A heapdump was generated, and we analyzed it with the Memory Analyzer:

- Click on ‘File’ --> ‘Open Heap Dump’ --> ‘Leak Suspects Report’ --> Under “Problem Suspect 1” --> “Shortest Paths To the Accumulation Point” and “Accumulated Objects” where we found the suspect “MemoryLeak.”

Class Name	Shallow Heap	Retained Heap
java.util.LinkedList\$Link @ 0x83401c470	48	209,384,520
java.util.LinkedList @ 0x83401c448	40	209,384,560
com.ibm.washington.tai.MemWorkout @ 0x83401b3b0	64	209,384,624
com.ibm.washington.tai.MemoryLeak @ 0x833f91150	40	209,384,664
com.ibm.ws.webcontainer.servlet.ServletWrapperImpl @ 0x831eab8e8	240	209,385,000
com.ibm.ws.webcontainer.webapp.WebAppServletInvocationEvent @ 0x833a75828 Unknown	88	88
com.ibm.oti.vm.BootstrapClassLoader @ 0x827f42910 System Class	200	128,824
java.util.LinkedList\$Link @ 0x833a6d880 >	48	48
com.ibm.ws.util.ClauseNode @ 0x831ec8590 >	56	432
Σ Total: 3 entries		

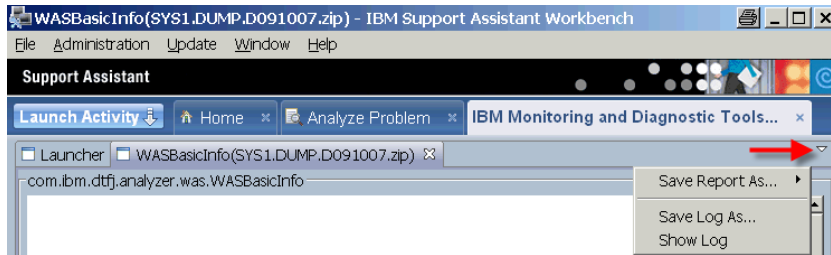
For more information about the Memory Analyzer, see the ISA “Help” pages, and <http://www.ibm.com/developerworks/java/jdk/tools/memoryanalyzer>

Dump Analyzer (IBM Monitoring and Diagnostic Tools for Java)

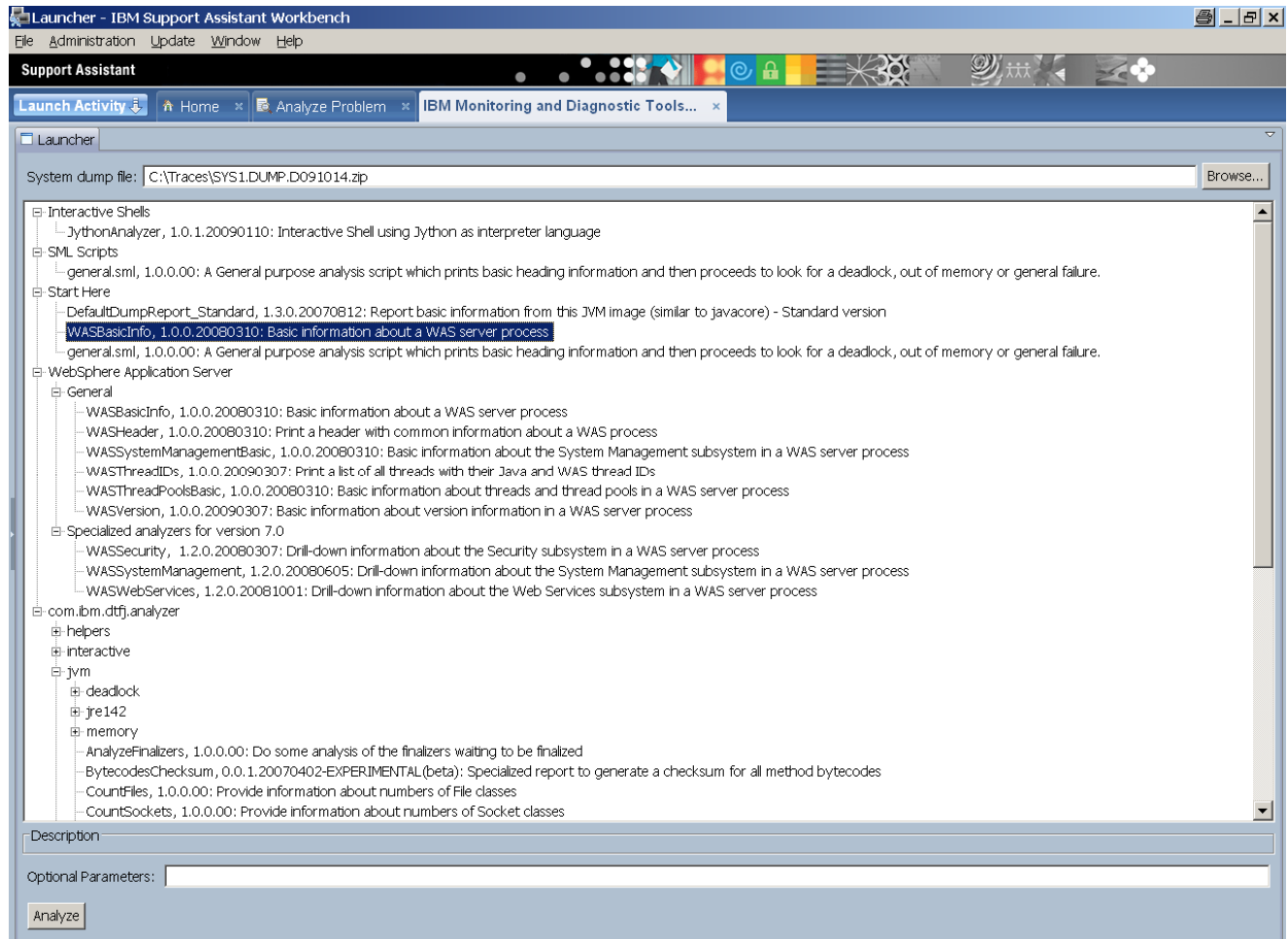
This tool is an extension of the `jdumpview` tool found in the IBM SDK, and runs a variety of specialized analyzers to extract information from a JVM system dump (System dump on z/OS.)

This works with system dumps, not heap or javacore dumps. The dumps must first be processed by the `jextract` tool found in the `/java/bin/` directory. See section “Working with Dumps on z/OS” later in this paper.

Note: If you have errors when invoking the Dump Analyzer, go to the “Show Log” button in the little downward-pointing triangle - **Context menu** (pull-down menu in the upper-right corner of the screen.)

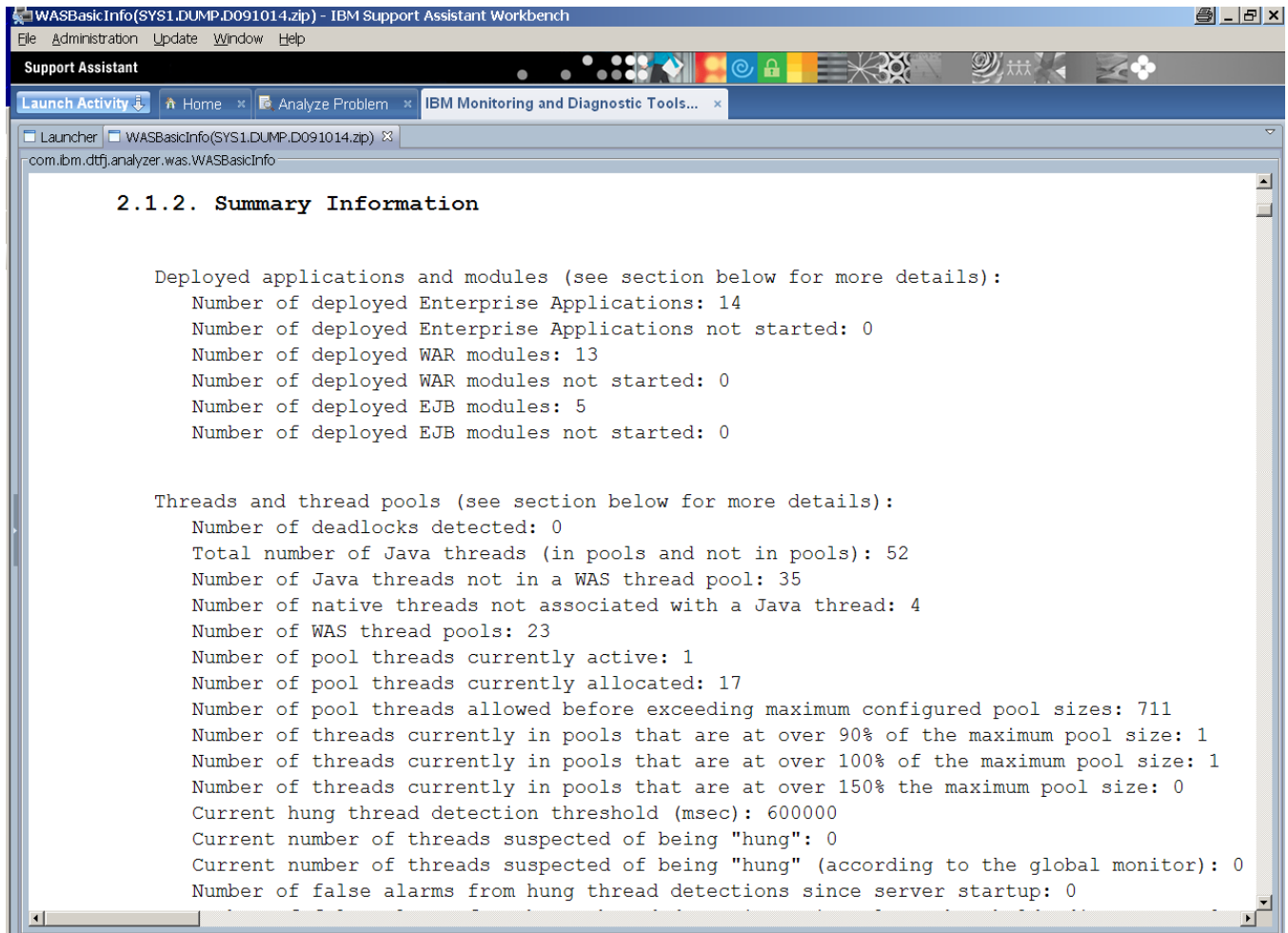


There are many analysis reports available, as you can see from this menu:



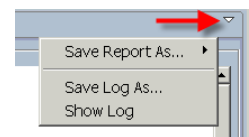
Click on “Browse” to access a dump file processed by `jextract` such as `SYS1.DUMP.D091014.zip` and select an analysis option(?).

Below is a screen-shot of the Dump Analyzer output showing the “WASBasicInfo screen which is actually many screens long.



There are no “pretty pictures” graphs or pie charts, but the reports provide an organized view of many things in the JVM heap.

You can save these reports as HTML, XML, or as a TXT file, using the same tiny downward-pointing triangle in the upper-right corner; click on “Save Log as...”



Dump Analyzer Sample Reports

Here is an example of the Default Dump Report”:

```
2. Analysis results
  DumpAnalyzer V:2.2.2.20090926232659 : Start analysis of C:\AppTraces\SYS1.DUMP.D091014.zip

2.1. Results from Analyzer=com.ibm.dtfj.analyzer.jvm.DefaultDumpReport_Standard
  Analyzer full name: com.ibm.dtfj.analyzer.jvm.DefaultDumpReport_Standard
  Analyzer version: 1.3.0.20070812
  Analyzer description: Report basic information from this JVM image (similar to javacore)

2.1.1. Image and runtime information
  Now reporting on runtime: 0.0.0
  Image: (no identity)
  Time of dump: Wed Oct 14 02:28:14 EDT 2009
  System Type: z/OS           System SubType: 01.10.00
  Processor Type: s390x       Processor SubType:
  Number of Processors: 2
  Installed Memory: 6442450944
  Host Name: wsc2
  IP address: /9.82.24.70
  This Image contains: 1 address spaces; 1 processes; 1 runtimes

Process: PID:0x30208
  Executable: main
  Command line: [<null>]
  Pointer size (bits): 64
  Signal that triggered this dump: 0 ((no signal info available, or dump was not triggered)
  Current Thread: 0x14b06000

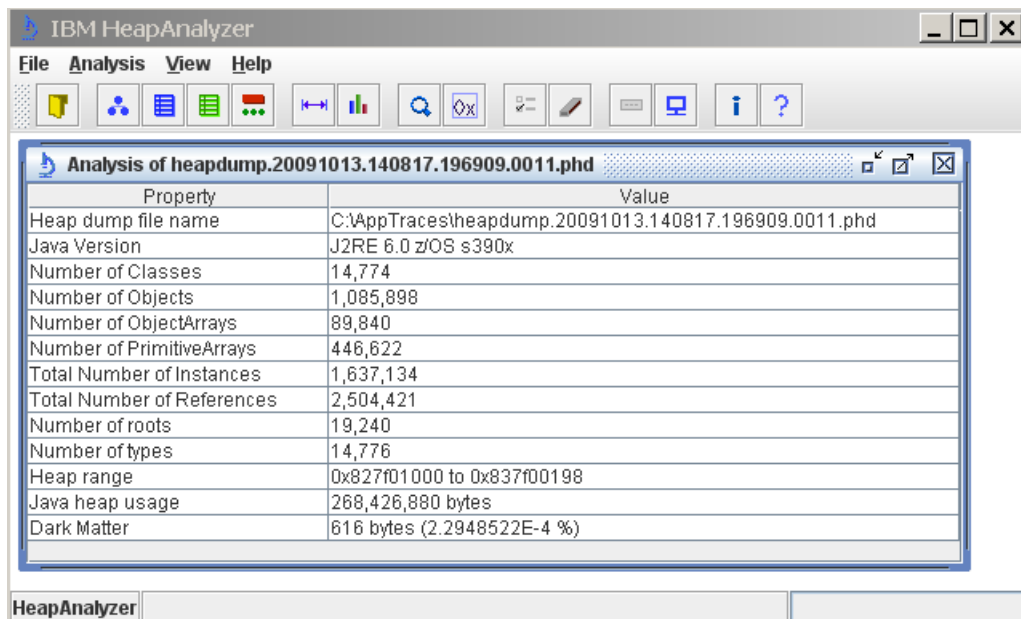
Java Runtime: JavaVM@0x000000080B178548
  Java Version:Java(TM) SE Runtime Environment(build pmz6460sr5ifx-20090623_02 (SR5))IBM J9
```


HeapAnalyzer

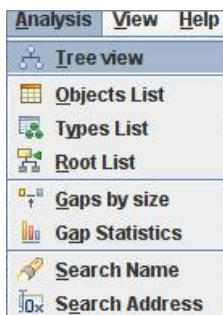
The IBM HeapAnalyzer provides a graphical way to analyze javacores (heapdumps). It was originally provided by the alphaWorks web site, but is now available through the IBM Support Assistant.

HeapAnalyzer analyzes Java heap dumps by parsing the heap dump, creating directional graphs, transforming them into directional trees, and executing a heuristic search engine to find memory leaks and excessive heap usage.

As with the other ISA tools, Install the HeapAnalyzer from the ISA “Tools Add-Ons” button (which will restart your ISA window.) Then launch the HeapAnalyzer, and browse to the heapdump file you want analyzed. The HeapAnalyzer will open in a new window outside the ISA window, and the initial screen will look something like this:



You can use the tool bar icons to list the various displays:



See Help in ISA for detailed explanations of the reports. Additional documentation about the HeapAnalyzer can be found in the ISA Help pages, and on the web at:

<http://www.ibm.com/support/docview.wss?uid=swg21410980>

<http://www.ibm.com/support/docview.wss?uid=swg27017427>

Thread & Monitor Dump Analyzer for Java - TMDA

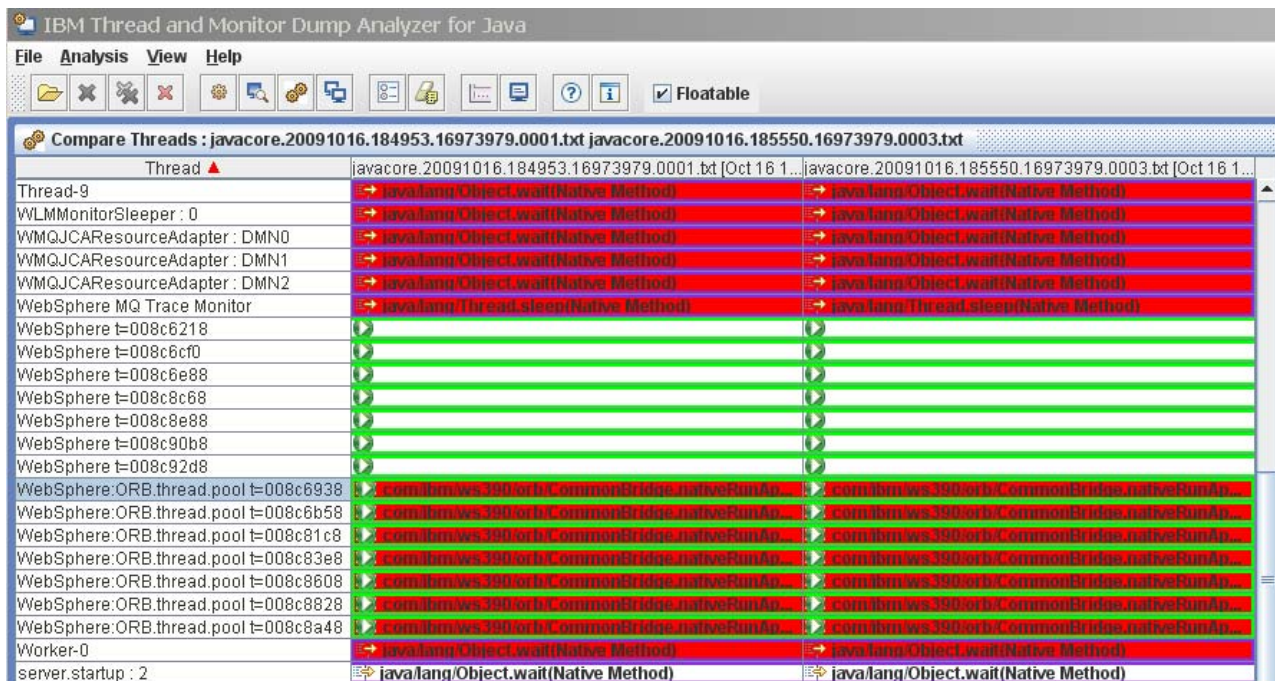
IBM Thread and Monitor Dump Analyzer for Java allow identification of hangs, deadlocks, resource contention, and bottlenecks in Java threads. (This tool is a tech preview.)

This works with javacore dumps. Here are some of the features it provides:

- Summary of Javacore
- Thread detail view
- Monitor detail view
- List of hang suspects
- Thread compare view
- Thread comparison summary
- Monitor lock compare view
- Garbage collector statistics for IBM JVM

See the Webcast presentation at: www.ibm.com/support/docview.wss?uid=swg27011855&aid=1

Here's an example comparing two javacore dumps from the same servant:



Note: This does not have anything to do with the TMDA tool, but a good discussion on analyzing threads in WebSphere on z/OS using java core dumps and output from the 'D OMVS' command, is in Techdoc WP101474 - "Threads and excessive CPU consumption in WebSphere on z/OS" at www.ibm.com/support/techdocs/atmastr.nsf/WebIndex/WP101474 .

Trace analysis tools

WebSphere formats the trace and message logs differently on z/OS than it does on other platforms. The JVM System.out and STDOUT streams are redirected to the SYSPRINT ddname, and the System.err and STDERR streams are redirected to the SYSOUT ddname, and are usually routed to print files on JES SPOOL (although some installations route these to files in the HFS filesystem.)

These streams are formatted to fit into 132-character wide print buffers, so they cannot be displayed with the tools that work with traces and logs on the WebSphere distributed platforms.

Trace and Request Analyzer for WebSphere Application Server

This tool is designed to detect delays and hangs in WebSphere trace and HTTP plug-in traces.

(I got this to work with the `addNode.log` file. Doesn't show much with other `wsadmin.sh` logs, or the logs in WAS for z/OS that usually go to JES SPOOL.)

Here is the "Gap Analysis" report for the `addNode.log` file:

Time(...)	Trace	Line Num...	Fi...
72,494	[9/8/08 15:12:38:376 EDT] 00000000 AdminTool A BBOO0222: ADMU0022: Node Agent launched. Waiting for initialization status.	107	a...
	[9/8/08 15:13:50:870 EDT] 00000000 AdminTool A BBOO0222: ADMU0030: Node Agent initialization completed successfully. Process id is: 000002ac00000002	108	a...
55,457	[9/8/08 15:11:12:337 EDT] 00000000 AdminTool A BBOO0222: ADMU0120: isclite on BLA will not be uploaded since it already exists in the target repository.	38	a...
	[9/8/08 15:12:07:794 EDT] 00000000 AdminTool A BBOO0222: ADMU0016: Synchronizing configuration between node and cell.	39	a...
29,879	[9/8/08 15:12:07:794 EDT] 00000000 AdminTool A BBOO0222: ADMU0016: Synchronizing configuration between node and cell.	39	a...
	[9/8/08 15:12:37:673 EDT] 00000000 AdminTool A BBOO0222: ADMU0018: Launching Node Agent process for node: h2nodeb	105	a...
29,249	[9/8/08 15:10:34:944 EDT] 00000000 AdminTool A BBOO0222: ADMU0009: Successfully connected to Deployment Manager Server: wsc2.washington.ibm.com:24010	21	a...
	[9/8/08 15:11:04:193 EDT] 00000000 AdminTool A BBOO0222: ADMU0505: Servers found in configuration:	22	a...
6,967	[9/8/08 15:12:27:595 EDT] 00000001 FileRepositor A BBOO0222: ADMR0011: Document cells/h2cell/PolicySetsWWS-I-RSP/policySet.xml is deleted.	102	a...
	[9/8/08 15:12:34:562 EDT] 00000001 NodeSyncTask I com.ibm.ws.management.sync.NodeSyncTask doSync ADMS0003: The configuration synchronization completed su...	103	a...
4,920	[9/8/08 15:10:25:076 EDT] 00000000 ModelMgr I BBOO0222: WSVR0801: Initializing all server configuration models	18	a...
	[9/8/08 15:10:29:996 EDT] 00000000 SSLConfigMana I BBOO0222: CWPFI0027: Disabling default hostname verification for HTTPS URL connections.	19	a...
4,503	[9/8/08 15:10:29:996 EDT] 00000000 SSLConfigMana I BBOO0222: CWPFI0027: Disabling default hostname verification for HTTPS URL connections.	19	a...
	[9/8/08 15:10:34:499 EDT] 00000000 AdminTool A BBOO0222: ADMU0001: Begin federation of node h2nodeb with Deployment Manager at wsc2.washington.ibm.com:...	20	a...
3,266	[9/8/08 15:11:05:883 EDT] 00000000 AdminTool A BBOO0222: ADMU0015: Backing up the original cell repository.	27	a...
	[9/8/08 15:11:09:149 EDT] 00000000 AdminTool A BBOO0222: ADMU0012: Creating Node Agent configuration for node: h2nodeb	28	a...
1,279	[9/8/08 15:11:04:293 EDT] 00000000 AdminTool A BBOO0222: ADMU2010: Stopping all server processes for node h2nodeb	24	a...
	[9/8/08 15:11:05:572 EDT] 00000000 AdminTool A BBOO0222: ADMU0512: Server h2sr01b cannot be reached. It appears to be stopped.	25	a...
768	[9/8/08 15:11:10:527 EDT] 00000000 AdminTool A BBOO0222: ADMU0014: Adding node h2nodeb configuration to cell: h2cell	32	a...
	[9/8/08 15:11:11:295 EDT] 00000000 AdminTool A BBOO0222: ADMU0120: WebSphereWSDM on CU will not be uploaded since it already exists in the target repository.	33	a...
693	[9/8/08 15:11:09:149 EDT] 00000000 AdminTool A BBOO0222: ADMU0012: Creating Node Agent configuration for node: h2nodeb	28	a...
	[9/8/08 15:11:09:842 EDT] 00000000 AdminTool A BBOO0222: ADMU0120: WebSphereWSDM.ear will not be uploaded since it already exists in the target repository.	29	a...
686	[9/8/08 15:12:21:688 EDT] 00000001 FileRepositor W BBOO0221W: ADMR0014W: The system is overwriting document cells/h2cell/variables.xml by request.	48	a...
	[9/8/08 15:12:22:374 EDT] 00000001 FileRepositor A BBOO0222: ADMR0010: Document cells/h2cell/variables.xml is modified.	49	a...
669	[9/8/08 15:12:37:707 EDT] 00000000 AdminTool A BBOO0222: ADMU0020: Reading configuration for Node Agent process: nodeagent	106	a...

Log Analyzer

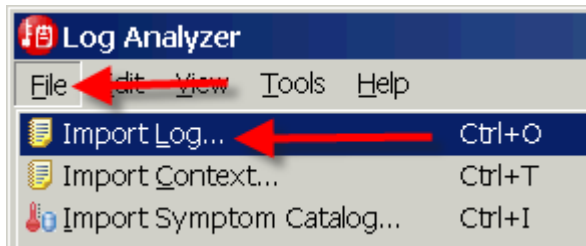
The Log Analyzer is a graphical user interface to browse, analyze, and correlate log events produced by many different products.

For WAS on z/OS, it works with wsadmin.sh logs, verboseGC traces, but cannot process logs such as those usually written to the SYSOUT and SYSPRINT files on JES SPOOL.

- When you launch it from the ISA, the Log Analyzer opens in a new window

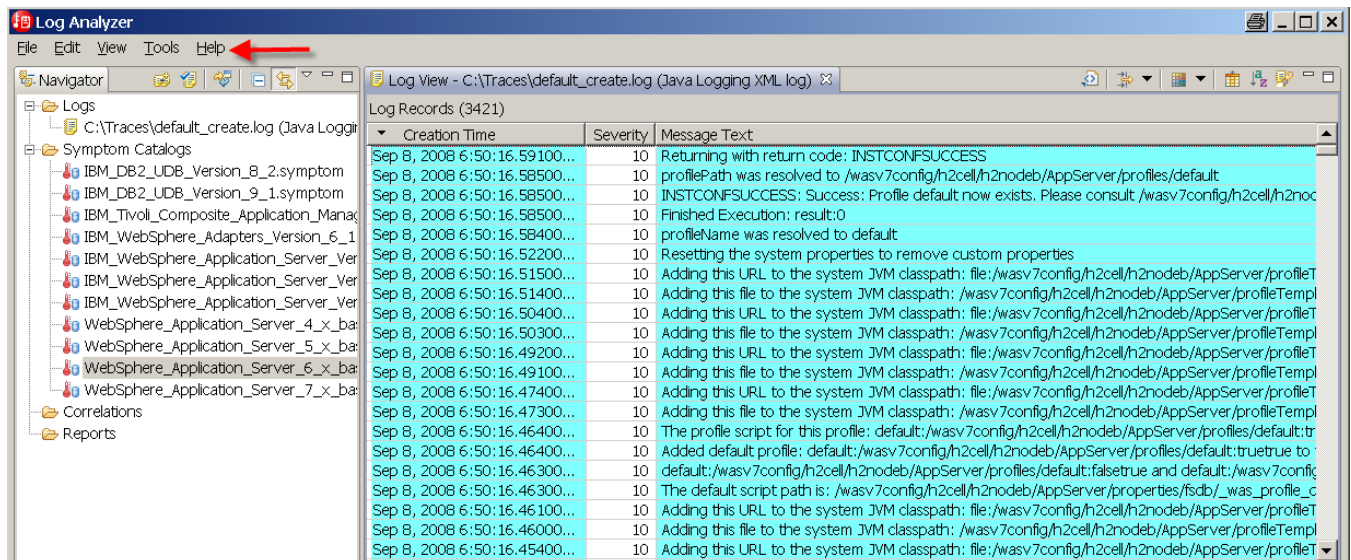


- Click on File --> Import Log --> ..from local system



- Select log file --> Symptom Catalog (WAS V.7) --> Finish

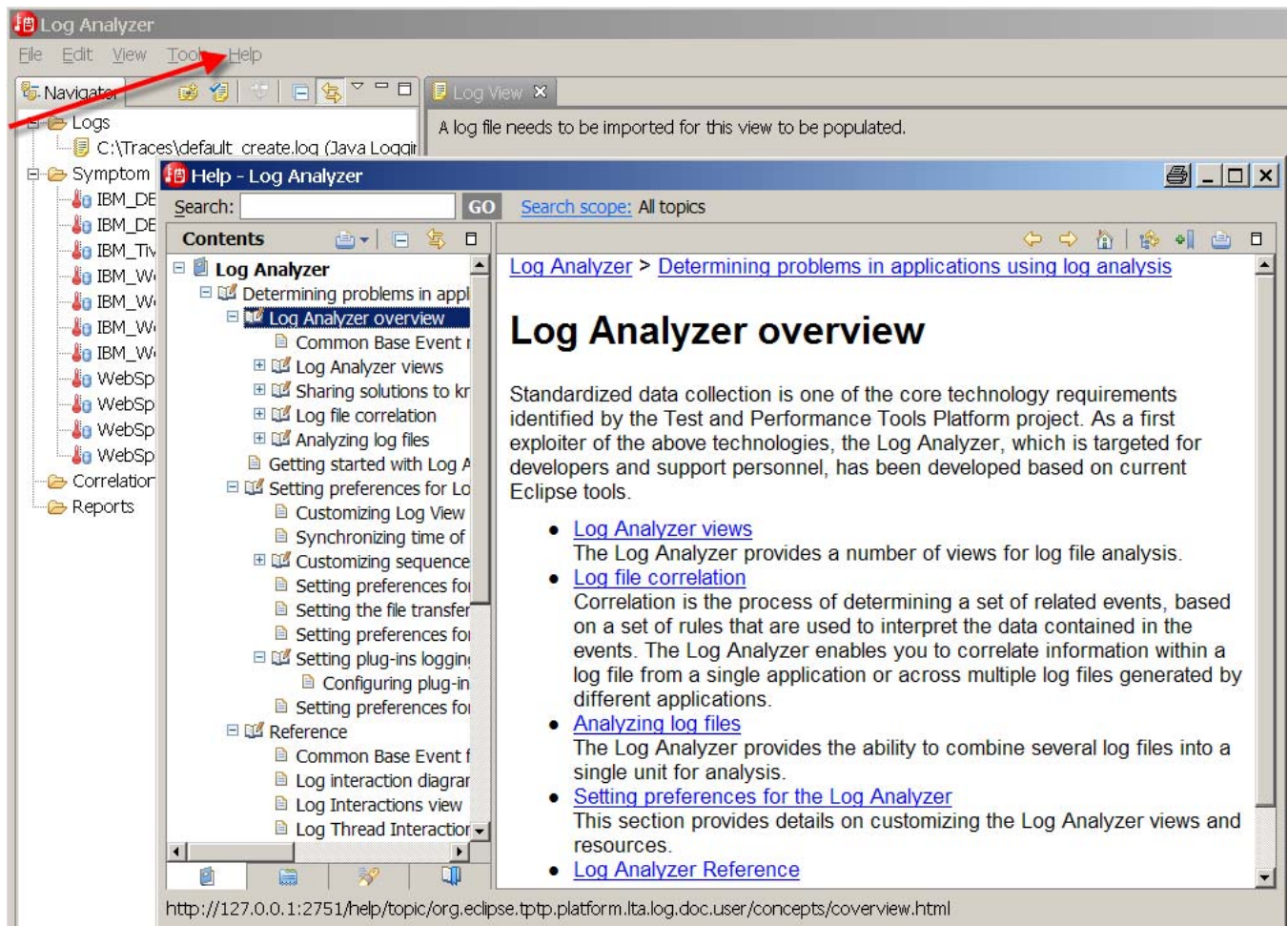
- The Log Analyzer will then “Analyze” the imported log file and display the log entries in tabular format,



The Log Analyzer provides a number of views for log file analysis.

- Log view showing the properties and values of each log record.
- Navigator view is a tree structure view that allows you to navigate and manage logs, log correlations, and symptom catalogs.
- Log Interactions view displays interactions among log records that occur during the execution of an application. This view shows correlations between log records from a single log file or across multiple log files.
- Log Thread Interactions view showing interactions among log events which occur on different threads, which participate in the execution of an application.
- Symptom Analysis Results view

Note: More extensive help for the Log Analyzer is available from the “Help” pull-down in the Log Analyzer window, as seen below.



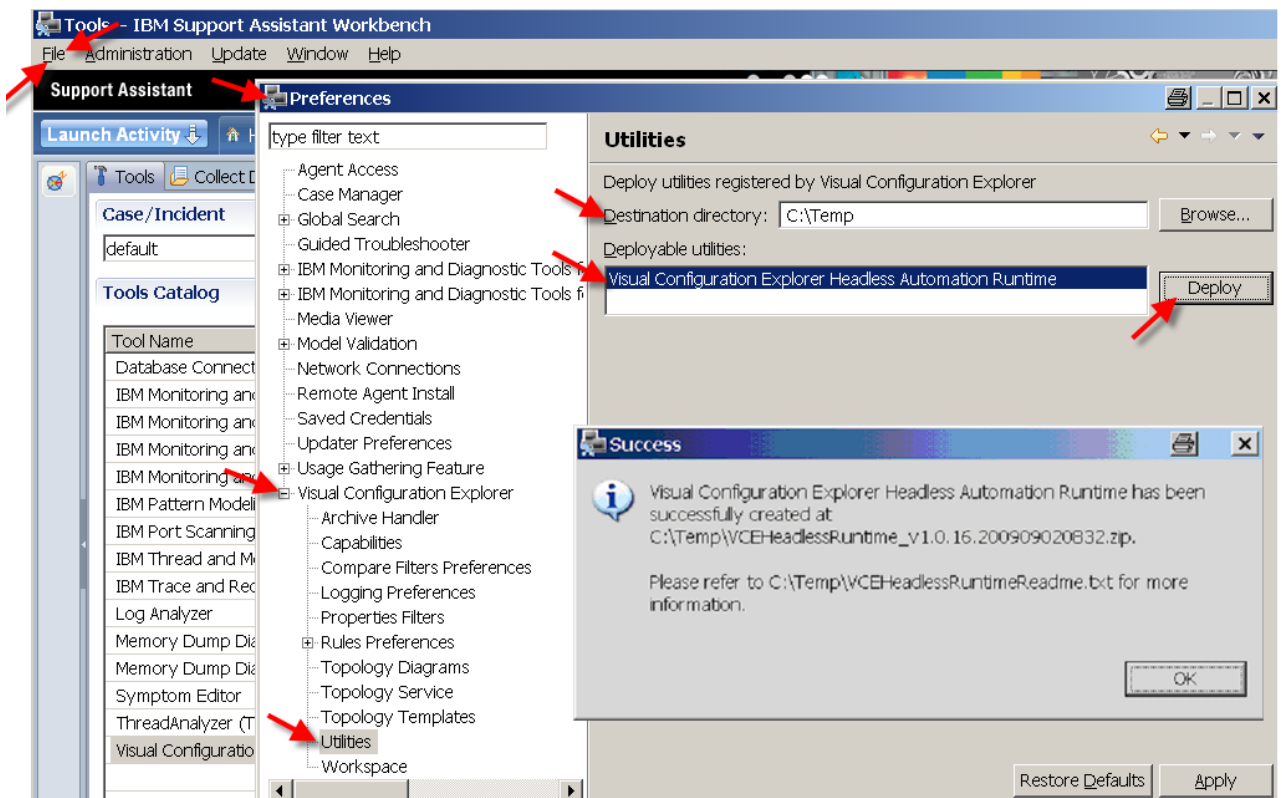
Configuration analysis tools

Visual Configuration Explorer - VCE

VCE can be used to visualize, explore, and analyze configuration from WebSphere servers and other sources.

Set up VCE on z/OS by following these steps:

1. Start the IBM Support Assistant, and select Update --> Find new --> Product Add-Ons, then select Other --> Visual Configuration Explorer tool. The ISA will restart after installation.
2. Use the “Headless Deploy” Utility from the ISA Home Page, select File --> Preferences --> Visual Configuration Explorer --> Utilities tab. Then select the Destination directory, the “Visual Configuration Explorer Headless Automation Runtime” and click on “Deploy”.



You will see the “Success” pop-up, and the VCEHeadlessRuntime_v1.0.16.xx.zip file and the VCEHeadlessRuntimeReadme.txt will appear in your destination directory:

3. Using FTP, transfer the zip file in binary from to your working directory in the z/OS USS, and print the Readme.txt document. (Note that the zip file is about 16 Mb in size.)
4. Telnet into your working directory and un-jar (i.e., unzip) the zip file with the following jar command. (The java bin directory must be in your PATH settings.)

```
jar -xvf VCEHeadlessRuntime_v1.0.16.200909020832.zip
```

A "vce" subdirectory will be created from which you can launch your operations.

For more information about the VCE Headless Automation Runtime, please see the built-in help of the VCE in the ISA.



Capture the VCE Export file: Run the Configuration Exporter for each cell you want to “Visualize.”

1. Set the WAS_HOME variable to the home of the deployment manager of the cell you wish to “visualize” using the setupCmdLine.sh script found in the <profile_root>/bin directory.
2. Run the Exporter using the following java command in your ./vce sub-directory:

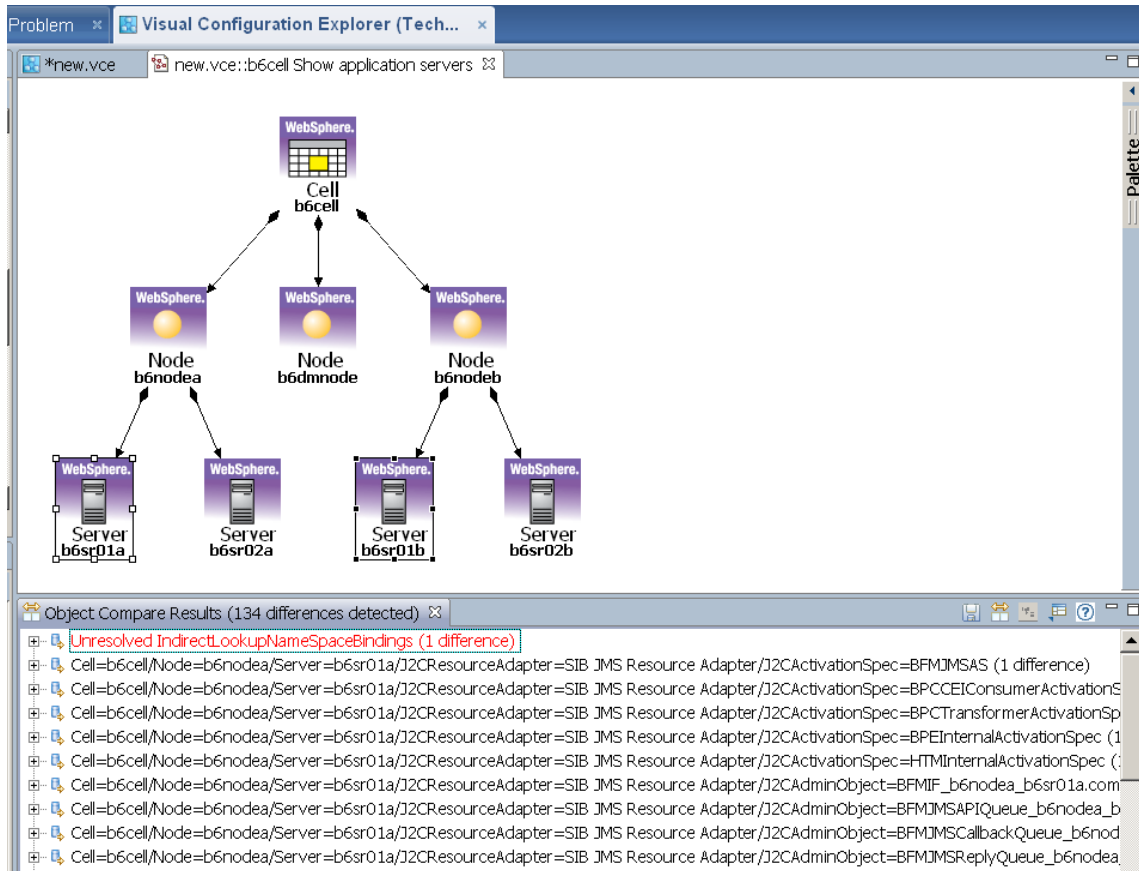
```
java -jar startup.jar -buildfile wasexporter/wasexport.xml -Dwas.root=$WAS_HOME
```
3. FTP the websphere.configuration file from the ./vce/wasexporter/output/ sub-directory to your workstation (in binary). Re-name each configuration file to reflect the name of the cell.

Repeat the previous three steps (1, 2, and 3) for each cell you wish to “Visualize.” The data is now collected and ready to be visually explored. For more details see the “Information for WebSphere Configuration Exporter” Readme document.

Using the VCE Visualizer: Use the “Quick Start” menu or “File > VCE Explorer” pull-down as follows:

- Create a new ‘Workspace’ using “File” pull-down -> VCE Explorer -> ‘New’ or ‘Open Workspace’
- Add one or more “Configuration” files: [Add Configuration](#)  tool
- Select a new Diagram: [New Diagram](#)  tool
- In the left-hand configuration panel, expand the structure and select a Server or Cluster.
- Drag it over to the ‘Diagram’ dialog.
- Right-click on the object and select “Show Parents” or “Show Children”.

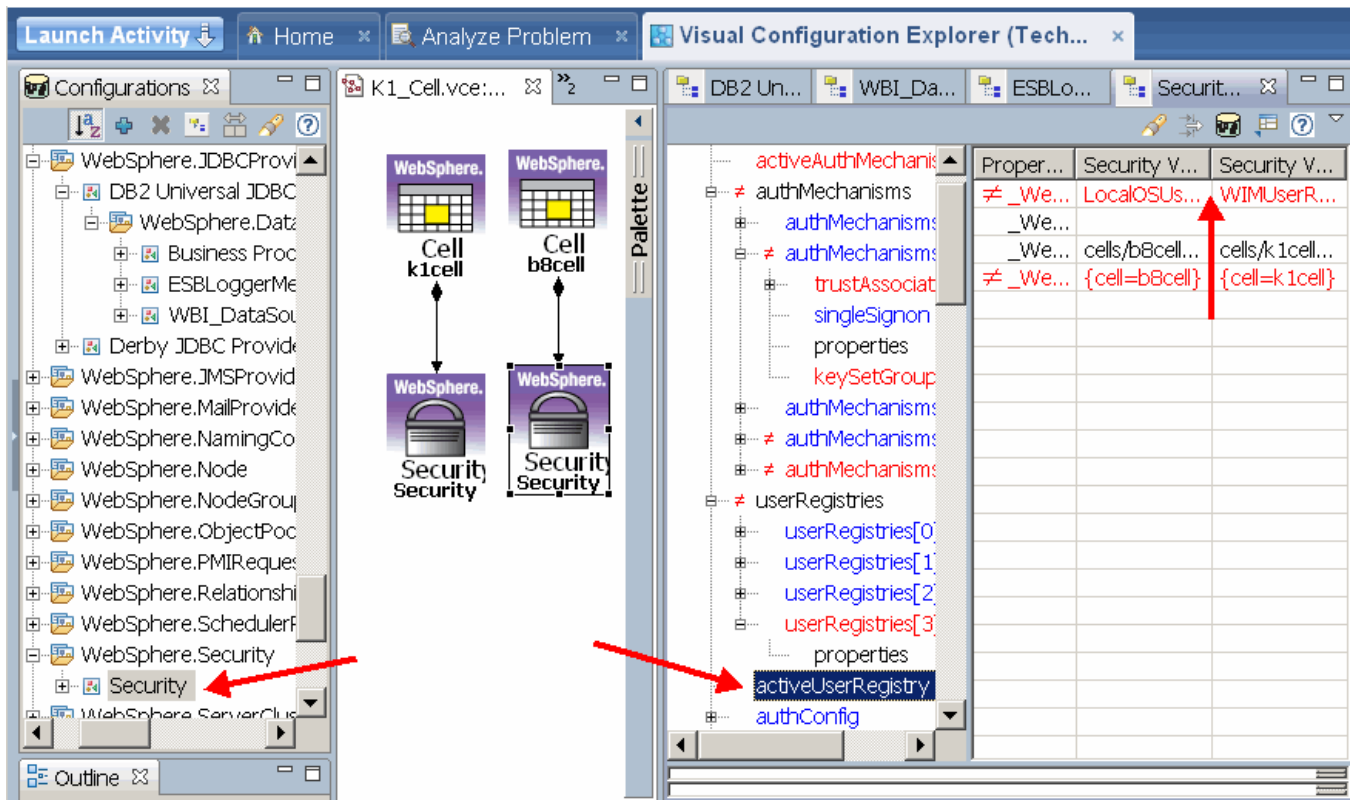
Here is an example of the servers (and their “parents”) in the b6cell:



To compare two configurations, use the following steps:

- Drag the servers or other objects into the Diagram.
- Right-click on “Show Properties Node.”
It will appear in a new panel on the right.
- Select a comparable object and select “Compare Objects”

Here is an example of the VCE visualization screen with Object Comparison of the Security panels below:



For more information about using VCE, see the following:

- ❑ SHARE Presentation by Mickey Scott:
http://ew.share.org/client_files/callpapers/attach/SHARE_in_Denver/S1153MS154246.pdf
- ❑ Developerworks paper by the developers of VCE:
http://www.ibm.com/developerworks/websphere/techjournal/0710_supauth/0710_supauth.html

Working with Memory Dumps on z/OS

There are many different kinds of dumps

Java Dumps

- These are usually created in the servant or controller's owner's home directory. This is usually at `/var/WebSphere/home/<cell_name>/<address_space_group_owner>` (See examples below.) These directories can fill up with dumps if you don't periodically delete them.
- The following MVS console commands can be used to capture java dumps:
 - Heap: MVS command: F <server_name>,HEAPDUMP
 - Core: F <server_name>,JAVACORE
 - JVM TDUMP: F <server_name>,JAVATDUMP
- They will also be created in the servant's userID's home case of a JVM failure, such as an OutOfMemory condition. You can see these messages in the SYSLOG when they occur:

```
+JVMDUMP032I JVM requested Java dump using '/SYSB/var/WebSphere/home/h2cell/H2SRG/
javacore.20091007.171936.197388.0009.txt' in response to an event

+JVMDUMP032I JVM requested Snap dump using '/SYSB/var/WebSphere/home/h2cell/H2SRG/
Snap.20091007.171953.197388.0010.trc' in response to an event

+JVMDUMP032I JVM requested Heap dump using '/SYSB/var/WebSphere/home/h2cell/H2SRG/
heapdump.20091007.171953.197388.0011.phd' in response to an event
```

z/OS System (SVC) Dumps

SVC dumps (also called MVS System dumps) go to MVS data sets named:

```
SYS1.DUMP.Dyyymmdd.Thhmmss.sysname.Snnnn
```

To capture a system dump, first setup and verify the z/OS Dump Options with these commands from the MVS console:

```
CD SET,SDUMP=(RGN)
```

D DUMP,O and you should see the following options in the IEE857I messages:

```
SDUMP- ADD OPTIONS (LSQA,RGN,TRT)
```

Then issue the MVS Dump console command, and respond to the WTOR (write-to-operator-with-reply) message:

```
Dump COMM='title'
```

```
R nn,jobname=<startedTask_JobName>
```

Other Dumps

- SYSABEND Dumps usually go to the JES SPOOL
- WebSphere ABEND dumps usually go to the `>/var/home/...`
- CEE Dumps usually go to the CEEDUMP DD on JES spool

Use jextract to convert system dumps for the Dump Analyzer

`jextract` is shipped with the IBM JVM in the `./java64/bin/` directory, and documented in as part of the dump viewer documentation in the Java 6 Diagnostic Guide. (Not the "Java for z/OS

InfoCenter.”) Note that jextract (and jdmpview) are designed to work with 32-bit and 64-bit JVMs, and currently do not work with the 31-bit JVM on z/OS.

Point directly to the Dump dataset:

```
/shared/zWebSphere/V7R0/java64/J6.0_64/bin/jextract SYS1.DUMP.D091014.T102814.SYSB.S00003
Loading dump file...
Read memory image from SYS1.DUMP.D091014.T102814.SYSB.S00003
Set debug scratch space size to 8 MB
VM set to 000000080B178548
Dumping JExtract file to SYS1.DUMP.D091014.T102814.SYSB.S00003.xml
<!-- extracting gpf state --> <!-- 5ms -->
<!-- extracting host network data --> <!-- 9ms -->
<!-- extracting classes --> <!-- 183200ms -->
<!-- extracting monitors --> ..<!-- 188234ms -->
<!-- extracting threads --> .<!-- 195727ms -->
<!-- extracting trace buffers --> <!-- 195740ms -->
<!-- extracting roots -->.<!-- 199500ms -->
<!-- extracting objects --> .<!-- 268266ms -->
Finished writing jextract XML file in 268268ms
Warning: found 14 inconsistencies in the dump file. Further information has been written to the
jextract XML file
Creating archive file: SYS1.DUMP.D091014.T102814.SYSB.S00003.zip
Adding "SYS1.DUMP.D091014.T102814.SYSB.S00003"
Adding "SYS1.DUMP.D091014.T102814.SYSB.S00003.xml"
Adding "/shared/zWebSphere/V7R0/java64/J6.0_64/lib/TraceFormat.dat"
Adding "/shared/zWebSphere/V7R0/java64/J6.0_64/lib/J9TraceFormat.dat"
jextract complete..
```

FTP the resulting .zip file to your workstation.

You may have to specify these arguments if you get these messages:

```
export J9DBGEXT_SCRATCH_SIZE=8
jextract -J-DJavaio.tmpdir=/u/hutch/largezfs-Xmx1024 SYS1.DUMP.D091014.
```

Using jdmpview to view the SYSTEM dump (instead of the Dump Analyzer)

jdmpview is shipped with the IBM JVM in the ../java64/bin/ directory, and currently only works with the 64-bit JVM on z/OS.

jdmpview writes the dump and the XML file to the /tmp/ directory, then unpacks the .zip.

If the USS /tmp disk space is limited that might be a problem. A workaround would be to unzip the dump yourself by hand somewhere where you do have space, then run jdmpview directly on the core (the original SYSTEM dump) and xml files. You do not need the xxx.dat files; all you need is the xml file in addition to the original dump, so you can run jdmpview with the –core and –xml parameters:

```
jdmpview -xml SYS1.DUMP.D091014.T102814.SYSB.S00003.xml -core SYS1.DUMP.D091014.T102814.SYSB.S00003
```

Documentation for jextract and jdmpview can be found in the IBM java Information Center. For java 6.0, this is at <http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp>

zOSMF

The IBM® z/OS® Management Facility (z/OSMF) V1.11 (5655-S28) is a new product for z/OS providing support for a modern, Web-browser-based management console for z/OS. It is configured in a WebSphere for z/OS OEM Edition which is also provided in z/OS V1.11.

This initial release of z/OS Management Facility provides:

- Incident Log capability to help facilitate problem data management tasks.
- Configuration Assistant for z/OS Comm. Server to help configure TCP/IP networking policies.

See Techdoc PRS3796 “z/OS System Management Facility (z/OSMF) & WSC User Experience” for more information.

Working with Traces on z/OS

There are also many different kinds of traces (and some of them are different from those captured on other platforms).

WebSphere Traces

Operator informational and error messages are normally directed to JES Spool datasets.

Examples of these include:

- RAS Traces & output from System.out.println are directed to the SYSPRINT DD file.
- Error log output, output from System.err.println, and verbose GC directed to SYSOUT
- Errors caught by LE or the Java run-time are written to the CEEDUMP file.

Java tracing: Enable java tracing dynamically with the MVS Modify (F) command. Here are some examples:

- F server,tracejava='com.ibm.*=all'
- F server,tracedetail=(3,4)
- F server,TRACEINIT (This resets tracing to the initial level when the server was started.)

See the Information Center or TD103695 on Techdocs for more details. (They may be re-directed to HFS files.)

ffdc logs – "first failure data capture"

Messages are written (in ASCII) to the servers' SYSPRINT file showing the name & location, i.e.,:

```
ExtendedMessage: FFDC0010I: FFDC closed incident stream file
/wasv6config/b6cell/nodeb/AppServer/profiles/default/logs/ffdc/
b6cell_b6nodeb_b6sr01b_STC32711_B6SR01BS_06.10.23_20.07.03_0.txt
```

wsadmin.sh traces

Edit the {profile_root}/properties/wsadmin.properties file, and uncomment the line:

```
#com.ibm.ws.scripting.traceString=com.ibm.*=all=enabled
```

The trace output will be directed to the file specified in the following property in the same file:

```
com.ibm.ws.scripting.traceFile=/wasv6config/b6cell/nodea/AppServer/profiles/default/logs/wsadmin.traceout
```

Then run the script and go look in the {profile_root}/logs/wsadmin.traceout file.

Configuration script logs

Many configuration shell scripts for WebSphere write log files and traces to the following directories:

{app_server_root}/logs or {profile_root}/logs or to subdirectories within these.

(e.g., /wasv6config/b6cell/nodea/AppServer/logs or
/wasv6config/b6cell/nodea/AppServer/profiles/default/logs

JDBC traces

Specify the following parameter in the jdbc properties file (normally named the

db2.jcc.propertiesFile): `db2.jcc.override.traceFile=/tmp/B6SR_tracejdbc.txt`

See the "DB2 Universal Database for z/OS Application Programming Guide and Reference For JAVA™" at <http://publib.boulder.ibm.com/infocenter/dzichelp/v2r2/index.jsp>

Browsing log files of traces in ascii

- In vi or PuTTY, use viascii or a2e in commands such as: `cat bpeconfig.log | a2e | pg`
- ISPF – use option 3.17 – the “z/OS UNIX Directory List Utility” in z/OS 1.8 and later.
- Convert them to EBCDIC with the ICONV UNIX command:
`iconv -f ISO8859-1 -t IBM-1047 xxx.log > xxx.log.ebc`
- Use FTP Client: Download to your workstation or 'View' in 'Binary' mode.

Additional tracing documentation

See the Information Center or the following Techdocs articles on managing operator and trace messages:

- TD103695 “Managing Operator Message Routing in WebSphere for z/OS Servers”
- WP101342 “Understanding SMF Record Type 120, Subtype 9”
- WP101233 “WAS for z/OS V6.1 Configuration Options for Handling Application Dispatch Timeouts”
- WP101374 “WebSphere Application Server for z/OS V7 - Dispatch Timeout Improvements”
- WP101138 - Hidden Gems: Lesser Known Features of WebSphere Application Server on z/OS
- WP101464 - Hidden Gems 2: More Great But Little-Known Features of WAS on z/OS

Performance Observations and Memory Requirements

The ISA workbench is an eclipsed-based tool built on java. In general, it initializes quickly (10 to 15 seconds) and takes about 175 Mb of memory on your workstation before you launch any of the ISA tools.

You can change the heapsize used by the eclipse rpc launcher by adding this parameter to the `rpclauncher.properties` file:

```
jvmarg.Xmx=-Xmx768m
```

Some ISA tools, such as the memory/dump analyzers, take over 700 Mb, so they may run slowly if your workstation is processor or memory constrained. (Or, your other workstation applications may be impacted.)

ISA Documentation and other Resources

- ISA Information can be accessed from the ISA Home Page using a local eclipse viewer. Click on the “Find Information” button on the ISA home page. You will next see three tabs:
 1. “Search Information” based on IBMSoftware Support Documents, developerworks, Newsgroups and Forums, Product information Centers, (and Google.)
 2. “Media Viewer” tab – accessed from the Find Information Page
 3. “Product Information” tab
- IEA (IBM Education Assistant) tutorials:
 - http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.selfassist/selfassist/ISAv41_Task.html
- YouTube has 3 “How To” Demos for the ISA Health Center

Other ISA Resources:

- Search the Web with “isa health center site:ibm.com”
- ISA websites: www.ibm.com/support/docview.wss?rs=3455&uid=swg27012682 and www.ibm.com/support/docview.wss?uid=swg27013116
- ISA Download website: www.ibm.com/software/support/isa/download.html
- IBM Support Assistant V4.1.1 ReadMe: www.ibm.com/support/docview.wss?uid=swg27016886
- IBM Support Assistant Team: www.ibm.com/software/support/isa
- ISA Forum: www.ibm.com/developerworks/forums/forum.jspa?forumID=935
- Comments/feedback to: IBMSA@us.ibm.com

Other resources:

- IBM Diagnostics Guides: www.ibm.com/developerworks/java/jdk/diagnosis/index.html or <http://download.boulder.ibm.com/ibmdl/pub/software/dw/jdk/diagnosis/diag60.pdf>
- IBM Java Information Centers at <http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp>
- developerWorks at www.ibm.com/developerworks/websphere
- alphaWorks at www.alphaworks.ibm.com

Appendix A – Other Diagnostic Tools

This document focuses on the tools for diagnosing java problems in WAS on z/OS, most of which can be found in the ISA. There are many other tools to help you with trouble-shooting problems that are listed here.

WebSphere “built-in” tools

See the WebSphere Information Centers for details.

- Traces - Modify MVS commands (see section above “Working with Traces on z/OS”)
- Modify server,display, . . . (MVS commands)
- Integrated Solutions Console (ISC) – Also called the Administrative Console - Performance Monitor (PMI) & Viewer
- `serviceInfo.sh` – run it from the `<AppServer>/bin/` directory to see the service level of all the products installed in the node’s runtime.
- `dumpNameSpace` to display the JNDI context roots
- `threadmonitor` (WAS V7) to configure the hang detection policy
- `wsadmin` scripting to enable tracing, and dump threads
- WAS Error Log

z/OS Diagnostic Tools – See the z/OS InfoCenter or Library

- RMF Monitors
- SMF Records
- CFRM Policy
- IPCS – Interactive Problem Control System

The all-powerful MVS Modify command

- F <server_name>,HELP
 - TRACEALL - Set overall trace level
 - TRACEBASIC - Set basic trace components
 - TRACEDetail - Set detailed trace components
 - TRACESPECIFIC - Set specific trace points
 - TRACEINIT - Reset to initial trace settings
 - TRACENONE - Turn off all tracing
 - JAVACORE - Generate jvm core dump
 - HEAPDUMP - Generate jvm heap dump
 - JAVATDUMP - Generate jvm tdump
 - TRACEJAVA - Set java trace options
- F <server_name>,DISPLAY,HELP
 - LISTENERS - Display listeners
 - CONNECTIONS - Display connection information
 - TRACE - Display information about trace settings
 - JVMHEAP - Display jvm heap statistics
 - WORK - Display work elements
 - ERRLOG - Display the last 10 entries in the error log
 - THREADS - Display thread status

JinsightLive for IBM System z

See <http://www.alphaworks.ibm.com/tech/jinsightlive>

Details of Document Change History

Check the date in the footer of the document for the version of the document.

<i>October 28, 2009</i>	Original document – Updated with Louis Wilen’s comments.
<i>October 29, 2009</i>	Draft updated with comments from Pete Robbins (MAT), and Howard Hellyer & Russell Wright (GCMV).
<i>November 12, 2009</i>	Updated with more comments from Russell Wright.
<i>December 3, 2009</i>	Updated with minor corrections.
<i>December 10, 2009</i>	Added HeapAnalyzer, “Working with Traces on z/OS”, and minor corrections.
<i>December 17, 2009</i>	Added references to Techdoc WP101612 “Getting started with analysis of GC, Heapdumps and Javacores For WebSphere on z/OS”
<i>January 3, 2010</i>	Added material for Memory and Dump Analysis tools, and minor editorial corrections.
<i>March 3, 2010</i>	Corrected material for running the VCE “headless” installer.
<i>March 9, 2010</i>	Enhanced directions for using the VCE Vizualizer to Compare two objects.

End of WP101575