Tivoli software

IBM

Deployment Guide Series
IBM Tivoli Composite Application Manager for WebSphere V6.0

- Analyzing performance of WebSphere-based application
- Providing debugging and problem determination
- Dynamically adjusting monitoring level

Budi Darmawan
Albert Csaszar

Redbooks

ibm.com/redbooks

International Technical Support Organization

Deployment Guide Series: IBM Tivoli Composite Application Manager for WebSphere V6.0

August 2006

SG24-7252-00

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (August 2006)

This edition applies to Version 6.0 of IBM Tivoli Composite Application Manager for WebSphere (product number 5698-A71).

Contents

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®	IBM®	Redbooks (logo) 
CrossWorlds Software®	IMS™	Tivoli Enterprise™
CrossWorlds®	Monitoring On Demand®	Tivoli®
CICS®	MVS™	VTAM®
Database 2™	OMEGAMON®	WebSphere®
DB2 Universal Database™	OS/400®	z/OS®
DB2®	Redbooks™	

The following terms are trademarks of other companies:

Java™ and all Java-based trademarks are trademarks of Sun™ Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX™ is a registered trademark of The Open Group in the United States and other countries.

Linux™ is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

This deployment guide helps you to plan the implementation, installation, and initial usage of IBM Tivoli Composite Application Manager for WebSphere V6.0. The objective of this IBM® Redbook is to use a basic configuration of the product for an initial user to successfully install the product, become familiar with the various working components and how they interoperate. The user can understand some of the product capabilities, and begin to use the product to monitor and manage applications on the IBM WebSphere® Application Server. This guide is similar to a cookbook with easy-to-follow steps with accompanying screen captures that illustrate the necessary tasks required to deploy and work with IBM Tivoli Composite Application Manager for WebSphere V6.0 in this basic configuration.

The instructions contained in this deployment guide target a Windows platform implementation. We do not cover other operating systems such as z/OS®-based data collectors. In addition to the installation steps, sample scenarios are included to highlight some of the ways in which you can use this product to increase the performance and availability of applications running on WebSphere Application Server. As a result, you can use this deployment guide to support an effective proof of concept demonstration of IBM Tivoli Composite Application Manager for WebSphere V6.0.

This book consists of the following chapters:

- Chapter 1, “IBM Tivoli Composite Application Manager for WebSphere V6.0” on page 1 provides an overview of the product.
- Chapter 2, “Planning for IBM Tivoli Composite Application Manager for WebSphere” on page 15 discusses some of the planning considerations and implementation scenarios for IBM Tivoli Composite Application Manager for WebSphere.
- Chapter 3, “Installing IBM Tivoli Composite Application Manager for WebSphere” on page 25 walks through and explains the basic installation steps for the IBM Tivoli Composite Application Manager for WebSphere solution.
- Chapter 4, “Using IBM Tivoli Composite Application Manager for WebSphere” on page 77 demonstrates some sample usage scenarios for IBM Tivoli Composite Application Manager for WebSphere solution.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization (ITSO), Poughkeepsie Center.

Budi Darmawan is a Consulting IT Specialist at ITSO, Poughkeepsie Center. He writes extensively and teaches IBM classes worldwide on all areas of Tivoli® and systems management. Before joining the ITSO seven years ago, Budi worked in IBM Indonesia as a solution architect and lead implementer in Integrated Technology Services. His current interests are Java™ programming, availability management, and automation.

Albert Csaszar is a WebSphere IT Specialist and member of the TecWorks Americas team. His principal focus is in the area of process-based service-oriented architecture (SOA) driven business solutions. In his current role, he primarily develops and delivers WebSphere product focused Proof of Technology workshops across the Americas. He also supports field IT Specialists with customer engagements. Prior to joining TecWorks, Albert spent two years evangelizing process-driven integration in Latin America. Before joining IBM, he worked for CrossWorlds® Software® where the InterChange Server and WebSphere Business Integration (WBI) Adapters were developed.

Thanks to the following people for their contributions to this project:

Rugmony N.
ITSO, Bangalore, India

Elizabeth Purzer
IBM Software Group

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  ibm.com/redbooks

- Send your comments in an email to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYJ; HYJ Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

1

IBM Tivoli Composite Application Manager for WebSphere V6.0

This chapter introduces IBM Tivoli Composite Application Manager for WebSphere V6.0. It is divided into the following sections:

1.1 Composite application management

Computer-based applications have been the lifeblood of modern enterprises. Most business processes are driven by one or more computer applications that promote productivity, automate processing, and minimize human errors. These applications help business users to focus on their business tasks rather than on how to do these tasks. Business processes increasingly span multiple applications. As these processes become more reliant on the applications, these composite applications become increasingly critical. These applications must be available and perform effectively for the business processes to execute well in support of the enterprise.

Historically, most applications were designed and implemented as centralized mainframe-based applications where all the application layers were maintained and managed by a central information technology (IT) department. Today, applications tend to have multiple layers, often distributed across different servers, different platforms, and even consisting of components developed using different technologies. These applications are called *composite* applications. This complicates the management of applications, such as operational settings, problem determination, and performance management.

As business-critical entities applications must be available with adequate response time for users to perform their tasks effectively. With application components spread throughout the enterprise, problem determination and performance management are typically a significant challenge, sometimes a nightmare. When these composite applications do not function as designed or expected, there is no clear path for determining which components have a problem. Sometimes the components themselves may belong to different organizations with their own line of business IT department. Is it a database problem or a network problem? Is the application server experiencing a temporary or degenerative bottleneck that will ultimately bring the system down if it is not investigated and resolved? Is the user's machine creating the problem?

Figure 1-1 shows a typical composite application topology.

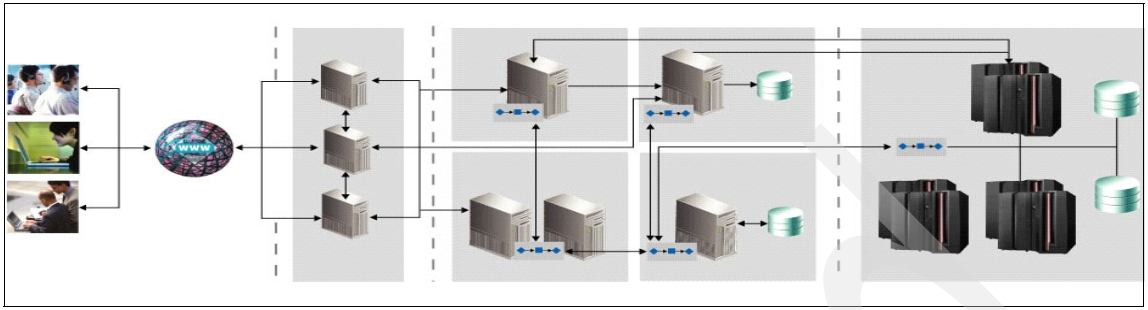


Figure 1-1 Composite application topology

The composite application in Figure 1-1 is used by multiple users from both the Internet and intranet. It consists of multiple application layers, each with its own abstraction layer. Some of the application functionality still resides in the original back-end mainframe as transactions.

Composite applications are regarded as the ultimate application management challenge because they span different application servers that must communicate effectively with each other. This architecture allows modular, flexible application development (changes in one layer may not affect other layers) and scalable distributed deployments, but introduces the complexity of multiple distributed components.

The IBM Tivoli product solution presented in this book is designed to make composite application management as easy as possible by supporting the overall IBM IT Service Management approach. Figure 1-2 illustrates the IBM IT Service Management portfolio.

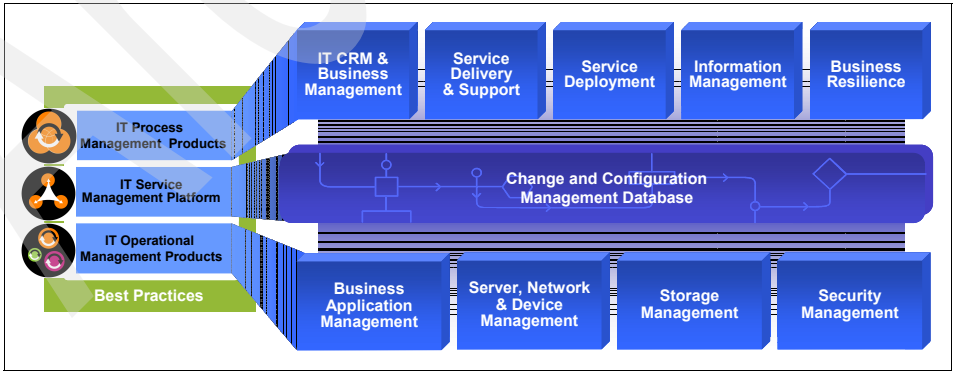


Figure 1-2 IBM IT Service Management

This approach provides IT Infrastructure Library (ITIL) aligned automation work flows. Future offerings will provide an open, standard-based configuration management database (CMDB) based solution as well as a work flow engine.

1.2 Tivoli composite application management solution

The IBM Tivoli Composite Application Manager family resides in the application management pillar of the Tivoli software portfolio. The current application management portfolio consists of the following products:

- IBM Tivoli Composite Application Manager for Response Time Tracking Version 6.0 (V6.0)
- IBM Tivoli Composite Application Manager for Service-Oriented Architecture (SOA) V6.0
- IBM Tivoli Composite Application Manager for WebSphere V6.0
- IBM Tivoli Composite Application Manager for Customer Information Control System (CICS®) V6.0
- IBM Tivoli Composite Application Manager for Information Management System (IMS™) V6.0
- IBM Tivoli OMEGAMON® XE for WebSphere Business Integration V1.1

Figure 1-3 shows the composite application management scope.

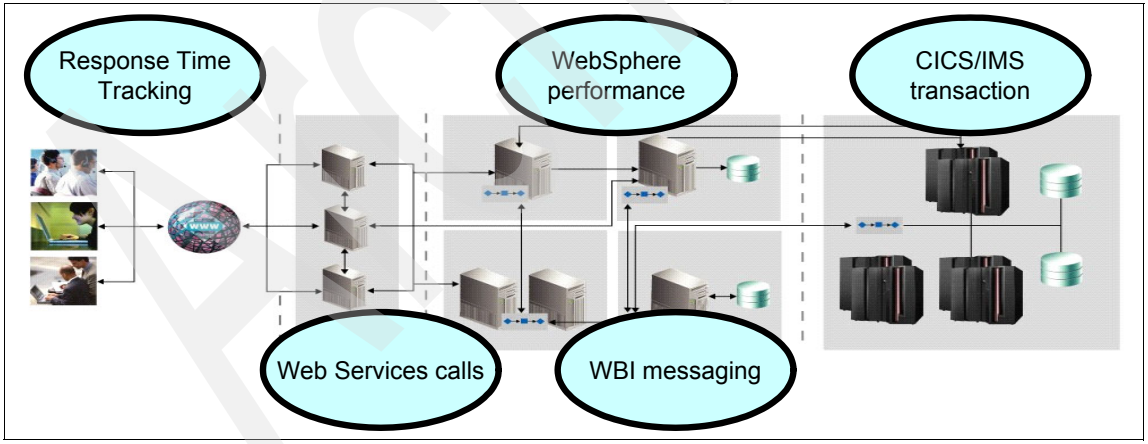


Figure 1-3 Composite application management

The overall composite application can be managed from various perspectives:

- Getting the user perspective of response time and availability with IBM Tivoli Composite Application Manager for Response Time Tracking
- Getting WebSphere middleware performance and analyzing in-depth resource usage perspective through IBM Tivoli Composite Application Manager for WebSphere
- Managing messaging from IBM WebSphere Business Integration MQ Series using OMEGAMON XE for WebSphere Business Integration
- Managing message flow in an SOA environment and collecting metrics for Web services calls using IBM Tivoli Composite Application Manager for SOA
- Providing an integration view with mainframe-based back-end applications such as IMS or CICS using IBM Tivoli Composite Application Manager for CICS Transactions or IBM Tivoli Composite Application Manager for IMS Transactions

For more information about Tivoli application management products, refer to the following redbooks:

- *IBM Tivoli Composite Application Manager V6.0 Family: Installation, Configuration, and Basic Usage*, SG24-7151
- *Implementing IBM Tivoli OMEGAMON XE for WebSphere Business Integration V1.1*, SG24-6768

1.3 IBM Tivoli Composite Application Manager for WebSphere products

IBM Tivoli Composite Application Manager for WebSphere V6.0 has evolved from WebSphere Studio Application Monitor and IBM Tivoli OMEGAMON XE for WebSphere Application Server. This history has resulted in IBM Tivoli Composite Application Manager for WebSphere containing the core features from both products, leveraging the strengths of both these products.

For more information about IBM Tivoli Composite Application Manager for WebSphere, refer to the following Web site:

http://www.ibm.com/software/tivoli/products/composite-application-mgr-websphere/

1.3.1 Features and benefits

IBM Tivoli Composite Application Manager for WebSphere helps increase the performance and availability of business-critical applications by providing real-time problem detection, analysis, and repair. Correlation spanning Java 2 Platform, Enterprise Edition (J2EE™), CICS, and IMS, and diagnostics at the method level can pinpoint code problems to help resolve problems quickly and reduce support and operations costs.

Today's business processes often depend on a number of complex applications. Although most businesses have traditional monitoring tools to manage individual resources at a high level, many lack an integrated solution to automatically monitor, analyze, and resolve problems at the service, transaction, application, and resource levels. As a result, operations and potentially development may take a long time to identify, isolate, and fix composite application problems.

IBM Tivoli Composite Application Manager for WebSphere is an application management tool that helps to maintain the availability and performance of on demand applications. It helps users to quickly pinpoint, in real time, the source of bottlenecks in application code, server resources, and external system dependencies. This product also provides detailed reports that you can use to enhance the performance of your applications. IBM Tivoli Composite Application Manager for WebSphere provides in-depth WebSphere-based application performance analysis and tracing facilities.

IBM Tivoli Composite Application Manager for WebSphere enables multiple levels of analysis to achieve a complete view of the application, depending on the requirement. From a production-level monitoring perspective through detailed heap and method debugging, it digs into Structured Query Language (SQL) performance analysis without the need for database monitors. It can provide SQL information and calls that were made through Java Database Connectivity (JDBC™). It provides a composite status correlation for transactions that use CICS or IMS as the back-end system.

IBM Tivoli Composite Application Manager for WebSphere observes and reports on the health of J2EE-based applications. It tracks the progress of applications as they traverse through J2EE application servers, middleware adapters and transports, database calls, and on to back-end systems such as CICS or IMS that extract business data or invoke mainframe business processes. The tracking of applications produces request traces, where the events in a request's life are recorded and stored in a monitoring repository database.

IBM Tivoli Composite Application Manager for WebSphere captures the processor and the elapsed internal times for event calls and exits, measuring as far down as the processor times consumed and the elapsed internal times charged to individual methods in J2EE classes. The methods or events taking

the most time are marked as an application's parts that deserve attention for runtime improvement studies and code optimizations.

IBM Tivoli Composite Application Manager for WebSphere does not require modification of any J2EE or mainframe application code. Java Virtual Machine Tool Interface (JVMTI) interfaces and primitives, along with WebSphere Performance Management Interface (PMI) and z/OS System Measurement Facility (SMF) 120 records are the principal data sources of IBM Tivoli Composite Application Manager for WebSphere. The monitoring data is collected and analyzed to offer a wealth of information about the health of J2EE applications and their servers.

Many system-level performance metrics are collected and reported about J2EE application servers. The status of the servers and their resources (particularly at vital checkpoints such as processor utilization), memory usage, and the status of internal components such as database connection pools, JVM™ thread pools, Enterprise JavaBeans™ (EJB™) usage, and request processing statistics can be very useful and important for locating real-time problems with J2EE applications. IBM Tivoli Composite Application Manager for WebSphere brings attention to these critical indicators with real-time, graphical displays of their values and trends over time.

1.3.2 Components

IBM Tivoli Composite Application Manager for WebSphere is a distributed performance monitoring application for application servers. Its components are connected through Internet Protocol (IP) communication. IBM Tivoli Composite Application Manager for WebSphere has the following components:

- The managing server

  The central component of IBM Tivoli Composite Application Manager for WebSphere, the managing server, is its heart and brain. It manages and administers the data collectors. It collects and displays various performance information from application servers. A Web-based application is provided to show the monitoring results. This interface is also called the *visualization engine*. See "The managing server" on page 8.

- Data collectors

  The application servers run a component of IBM Tivoli Composite Application Manager for WebSphere called the data collector. Data collectors are collecting agents that run on application servers that are being monitored. They send monitoring information to the managing server and operate independent of each other. See "WebSphere data collector" on page 11.

- The IBM Tivoli Enterprise Monitoring Agent

  It collects information that shows the status of the WebSphere server. It sends this information to the IBM Tivoli Enterprise Monitoring Server for display in the IBM Tivoli Enterprise Portal. The Tivoli Enterprise Monitoring Agent is installed on the individual machines where the data collector resides.

Figure 1-4 shows the overall architecture of IBM Tivoli Composite Application Manager for WebSphere.

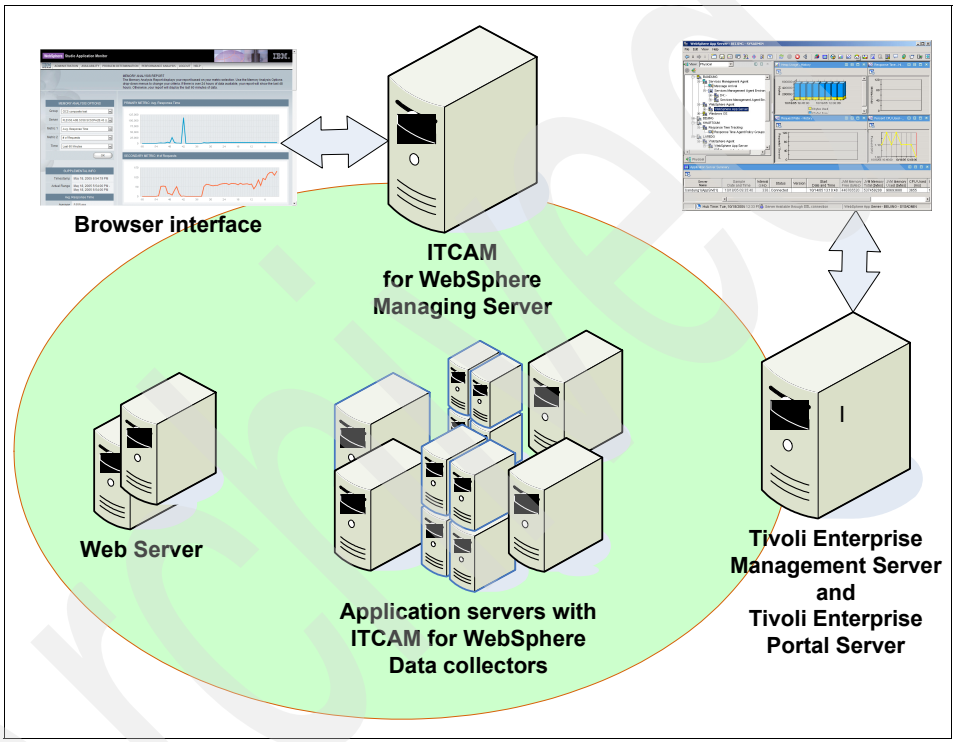


Figure 1-4 IBM Tivoli Composite Application Manager for WebSphere architecture

The managing server

IBM Tivoli Composite Application Manager for WebSphere managing server controls and coordinates data collectors for J2EE, CICS, or IMS servers that run applications.

The managing server consists of the following software:

- The X-Windows Virtual Frame Buffer (Xvfb) graphics package
- Managing server database, IBM DB2 Universal Database (DB2® UDB) or Oracle (on Sun™ Solaris™) for storing collected data relationally

- J2EE server to run the IBM Tivoli Composite Application Manager for WebSphere graphical console application
- An optional Apache Web server such as IBM HTTP Server
- IBM Tivoli Composite Application Manager for WebSphere managing server overseer components, which are a set of Java-based components

The IBM Tivoli Composite Application Manager for WebSphere overseer components are the controlling logic for the managing server. They are:

- Kernels

  The kernels control the managing server. There are always two copies of the kernel running as part of the IBM Tivoli Composite Application Manager for WebSphere managing server. The two copies are required because they support redundancy and failover capabilities. The kernels register components as they join the managing server, periodically renew connections and registrations between components and data collectors, and collect application server instance and component availability information.

- Publishing servers

  The publishing servers receive application and system event data from the data collectors. They gather and compute request-level information about performance metrics such as response times, and implement the trap monitoring and alerts features.

- Archive agents

  The archive agents receive monitoring data from the publishing servers, and store the monitoring data in the IBM Tivoli Composite Application Manager for WebSphere repository.

- Global publishing server

  The global publishing server collects information from the publishing servers. It correlates all parts and pieces of multi-server requests, such as requests from J2EE servers to execute CICS or IMS programs.

- Message dispatcher

  The message dispatcher is a conduit for messages from IBM Tivoli Composite Application Manager for WebSphere using e-mail and Simple Network Management Protocol (SNMP) facilities.

- Polling agents

  The polling agents collect data from Web servers for Apache version 2.0 and later versions.

- Visualization engine

  The visualization engine is a Web-based graphical user interface (GUI) with access to graphics, IBM Tivoli Composite Application Manager for WebSphere performance reports, real-time views of different slices of monitoring data, and access to the IBM Tivoli Composite Application Manager for WebSphere internal commands as well as event-driven functions. It runs on a J2EE server, such as the IBM WebSphere Application Server.

Figure 1-5 shows the conceptual relationship between these components.

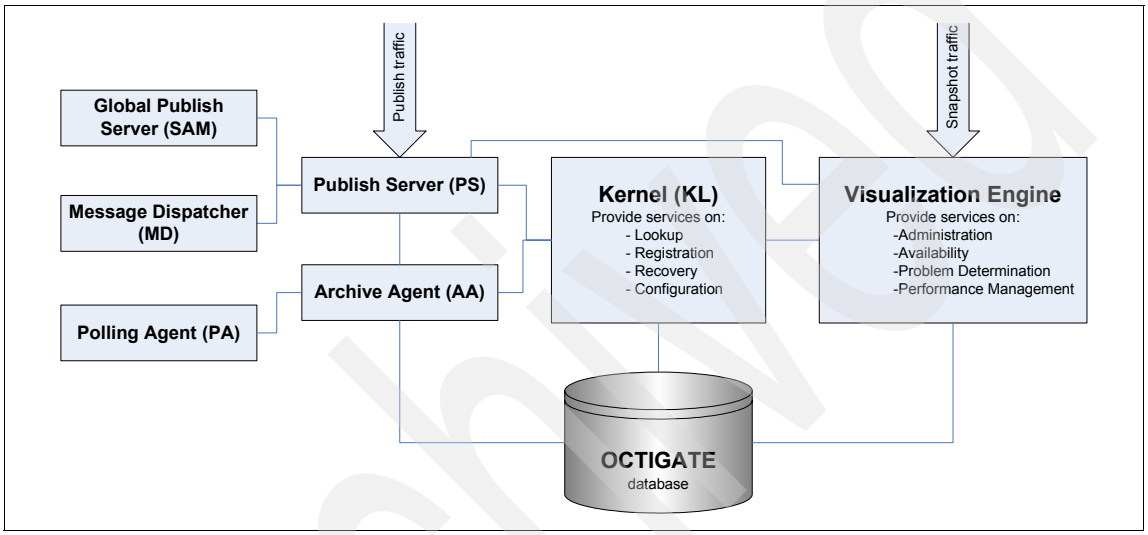


Figure 1-5 Managing server components

After collection by the managing server, the monitoring data is prepared for real-time display within the monitoring console and is inserted into the IBM Tivoli Composite Application Manager for WebSphere data repository. These monitoring server operations are very resource-intensive. Locating the components that comprise IBM Tivoli Composite Application Manager for WebSphere on one or more dedicated servers isolates them from other enterprise activities. This reduces the system resource footprint of IBM Tivoli Composite Application Manager for WebSphere and thus the impact on the monitored composite applications and systems. This design also helps keep the processing overhead of IBM Tivoli Composite Application Manager for WebSphere at levels that are low enough to support 24x7 production system monitoring.

Data from the distributed data collectors is collected by the publishing server and then stored in the OCTIGATE database by the archive agent. The visualization engine reads the database and presents data through the Web console, and

snapshot information such as lock analysis and in-flight transactions are retrieved directly from the data collectors.

WebSphere data collector

The data collectors use native system services, and they are tailored for the particular environments where they execute. The data collectors for z/OS systems are written to take advantage of services on z/OS, such as IBM Multiple Virtual Storage (MVS™) Cross-Memory Services and address space fencing, which are not available on distributed systems.

Data collectors have two agents:

- Command agent

  The command agent collects requests from other components for information about EJB invocations, database connection pools, thread pools, stack traces, memory analyses, and heap dumps.

- Event agent

  The event agent provides data to the publishing servers according to polling frequencies. This data includes system initialization data, application request-level data, and application method-level data.

Collectively, these agents and other data collector routines unleash the probes, package the monitoring data into Java formats, and deliver this data to the managing server. The data collectors send the probes into the application servers to analyze the applications' performance. The probes collect the monitoring data and feed it to transport routines that in turn route the data to the managing server. The managing server processes this data for appropriate display in the IBM Tivoli Composite Application Manager for WebSphere console and for storage in the OCTIGATE repository. This relieves the processing burden of IBM Tivoli Composite Application Manager for WebSphere from the application servers as much as possible. The data collectors and probes are not designed to analyze or interpret data, but to collect it and route it as quickly as possible to the managing server where the analysis is performed.

The data sources employed by IBM Tivoli Composite Application Manager for WebSphere are:

- JVMTI garbage collection data, method trace, stack trace, processor time, and heap dump
- Java Management Extension (JMX™) system resources
- SMF system resources (z/OS only)
- PMI system resources

- Operating system (OS) services, platform processor, and its environment
- Byte Code Modification (BCM) instrumentation of some classes

The data collector in a J2EE server runs as a custom service called *am*. Figure 1-6 shows the conceptual data collector structure.

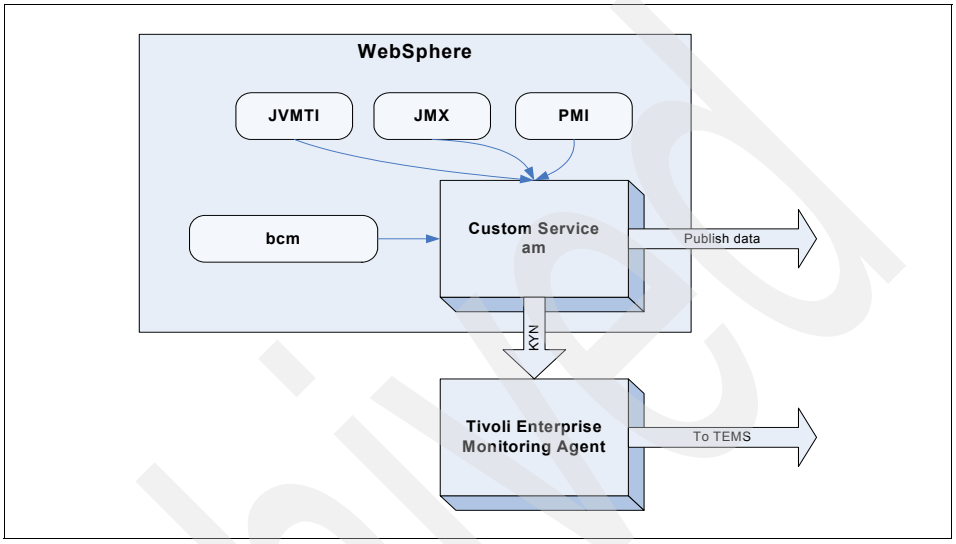


Figure 1-6 J2EE data collector structure

Tivoli Enterprise Monitoring Agent

The Tivoli Enterprise Monitoring Agent allows WebSphere performance information to be relayed to the Tivoli Enterprise Monitoring Server for display using Tivoli Enterprise Portal. This facility replaces the data collector mechanism employed by OMEGAMON for WebSphere Application Server. The Tivoli Enterprise Monitoring Agent communicates with the data collector in the local machine, retrieves performance information, and reports this information to the Tivoli Enterprise Monitoring Server.

For more information about the Tivoli Enterprise Monitoring Server and IBM Tivoli Monitoring V6.1 architecture, refer to *Getting Started with IBM Tivoli Monitoring 6.1 on Distributed Environments*, SG24-7143.

1.3.3 Overview of supported platform

For a complete platform coverage list, refer to the following Web site:

http://publib.boulder.ibm.com/tividd/td/ITCAMWAS/prereq60/en_US/HTML/itcam6.html

Table 1-1 provides an overview of the supported platform for IBM Tivoli Composite Application Manager for WebSphere V6.

Table 1-1 IBM Tivoli Composite Application Manager for WebSphere supported platform overview

Component	Software
Managing server operating system	► IBM AIX® V5.2, V5.3 ► Solaris 9 and 10 ► Hewlitt-Packard UNIX 11iv1 (HP-UX 11iv1) ► Red Hat Enterprise Linux 3.0 (RHEL 3.0) and RHEL 4.0 for xLinux, iLinux, and zLinux ► SUSE Linux Enterprise Server 8 (SLES8) and SLE9 for xLinux, iLinux, and zLinux ► Windows 2000 Server or Advanced Server with Service Pack 4 (SP4) ► Windows 2003 Server Standard Edition (SE) and Enterprise Edition (EE) ► Windows 2003 R2 Server SE and EE
Managing server database	► IBM DB2 V7.2 EE/EEE Fix Pack 11 (FP 11) ► DB2 V8.1 ESE FP6 ► DB2 V8.2 ESE ► Oracle 9i S/E R2 9.2+ ► Oracle 10*g*
Managing server WebSphere	► WebSphere Application Server V5.1.1 and later versions ► WebSphere Application Server V6.*x*
Data collector platform	► Windows 2000 Pro SP4 ► Windows 2000 Server or Advanced Server with SP4 ► Windows 2003 Server SE/EE ► AIX 5.2 and 5.3 ► RHEL 3.0 and 4.0 ► SLES 8 and 9 ► Solaris 8, 9, 10, Solaris 9 Cluster ► HP-UX 11iv1 ► Red Flag Advanced Server 4.0 (RFAS 4.0) and RFAS 4.1 ► IBM Operating System/400 (OS400®) V5R2 and V5R3 ► IBM z/OS V1.4, V1.5, V1.6, or V1.7
CICS	► CICS Transaction Gateway (CTG) V5.0.1 or V5.1 and V1.3, V2.2, V2.3, or V3.1
IMS	► V7.1, V8.1, and V9.1

2

Planning for IBM Tivoli Composite Application Manager for WebSphere

This chapter discusses the implementation options and solution configuration for IBM Tivoli Composite Application Manager for WebSphere. It consists of the following sections:

2.1 Implementation issues

The implementation of IBM Tivoli Composite Application Manager for WebSphere requires the configuration of the managing server and data collector. The implementation options depend on which edition of the product you use and the size of the installation.

This chapter discusses the prerequisites and preparation steps required before installing the product. For a complete and up-to-date prerequisite list, refer to the following Web site:

http://publib.boulder.ibm.com/tividd/td/ITCAMWAS/prereq60/en_US/HTML/itcam6.html

2.2 Managing server hardware platform

The hardware platform selection of the IBM Tivoli Composite Application Manager for WebSphere V6.0 managing server is influenced by several implementation factors. They are:

- 2.2.1, “Supported operating system” on page 16
- 2.2.2, “File system requirements” on page 17
- 2.2.3, “Hardware sizing consideration” on page 18

2.2.1 Supported operating system

The supported operating systems for the IBM Tivoli Composite Application Manager for WebSphere managing server are:

The supported operating systems for managing server are:

- AIX V5.2 and V5.3
- Solaris 9 and 10
- Hewlitt-Packard UNIX 11i v1 (HP-UX11i v1)
- Red Hat Enterprise Linux 3.0 (RHEL 3.0) and RHEL 4.0 for xLinux, iLinux, and zLinux
- SUSE Linux Enterprise Server 8 (SLES8) and SLE9 for xLinux, iLinux, and zLinux
- Windows 2000 Server or Advanced Server with Service Pack 4 (SP4)
- Windows 2003 Server Standard Edition (SE) and Enterprise Edition (EE)
- Windows 2003 R2 Server SE and EE

Note: If you have Windows 2003 Server with SP1, Windows 2003 R2 Server, or a later version, you have to apply the hotfix provided by Microsoft at the following Web site:

http://support.microsoft.com/default.aspx?scid=kb;en-us;899522

The Windows implementation of IBM Tivoli Composite Application Manager for WebSphere managing server has a slightly larger memory footprint than when installed on UNIX. Because the managing server components are constructed from a set of Java processes, memory is an important factor to consider. Larger implementations benefit by using a UNIX or Linux installed managing server and must therefore be considered. We recommend a minimum memory of 4 gigabytes (GB) for a production level managing server. As the load and processing requirements grow, you may have to adjust the available memory accordingly.

2.2.2 File system requirements

The installation of IBM Tivoli Composite Application Manager for WebSphere managing server requires a set of file systems. On Windows operating systems, these file systems are directory structures on a physical drive.

Table 2-1 shows the necessary file systems.

Table 2-1 File system requirement

File system usage	Typical path	Size requirement
DB2 binaries	► UNIX: /opt/db2_08_01 ► Windows: C:\PROGRA~1\SQLLIB	500 megabytes (MB)
DB2 database	► UNIX: /home/db2inst1 ► Windows: C:\PROGRA~1\SQLLIB	1 GB or more
WebSphere path	► UNIX: /opt/WebSphere ► Windows: C:\PROGRA~1\IBM\WebSphere	450 MB
Tivoli common directory	► UNIX: /var/ibm/tivoli/common ► Windows: C:\PROGRA~1\ibm\tivoli\common	200 MB or more
IBM Tivoli Composite Application Manager for WebSphere	► UNIX: /usr/lpp/IBM/ITCAM/WebSphere ► Windows: C:\PROGRA~1\IBM\ITCAM\WebSphere	250 MB

You may have to adjust the numbers in Table 2-1 for production environments. Typically, the size requirement for the DB2 database grows as a function of the monitored transaction volumes and the installation's data retention policy. Other

principal areas where size requirements vary widely include the log files stored in the Tivoli common directory. These log files are very useful for debugging purposes.

2.2.3 Hardware sizing consideration

The managing server workload is determined by several factors, such as:

- The number of data collectors that connect to the managing server. Each data collector generates information that must be processed by the managing server.
- The monitoring level for the data collectors. IBM Tivoli Composite Application Manager for WebSphere is typically run with Level 1 monitoring for a production environment. However, when more information is required, you can modify this level on the fly to collect Level 2 or Level 3 information. While Level 1 provides Servlet and Enterprise JavaBeans (EJB) invocation information, Level 2 and Level 3 collect class-level and method-level information.
- The transaction rates for each data collector. Each transaction that is processed by the application server generates information that is transmitted to the managing server.
- The sampling rate of the transactions. Although the managing server receives data for all transactions, it only saves a certain percentage of these transactions to the repository database.

The processing requirements for the managing server involve the usage of processor, memory, and disk input/output (I/O). The managing server is typically constrained by memory rather than processor or disk I/O requirements.

2.3 Managing server software requirements

The managing server requires several prerequisite software components:

- 2.3.1, “Database” on page 19
- 2.3.2, “Web application server” on page 19
- 2.3.3, “Operating system specific requirements” on page 19

2.3.1 Database

The managing server stores the performance information in a relational database system. The databases that are supported with the managing server are:

- DB2 V7.2 EE/EEE Fix Pack 11 (FP11)
- DB2 V8.1 ESE FP6
- DB2 V8.2 ESE
- Oracle 9i S/E R2 9.2+
- Oracle 10*g*

The database is called the OCTIGATE database and is written to by the archive agent component of the managing server. The visualization engine Web enterprise application retrieves information from this database to be displayed using the Web console.

The embedded installation wizard simplifies the installation of IBM Tivoli Composite Application Manager for WebSphere and its prerequisites. It installs IBM DB2 Universal Database™ V8.2 and IBM WebSphere Application Server V6.0.1. If you use a different supported version of DB2, you cannot use the simplified embedded installation method. This deployment guide demonstrates only the embedded installation method.

2.3.2 Web application server

The managing server Web interface is provided using a Java 2 Platform, Enterprise Edition (J2EE) compliant application. This requires the application to be deployed to a Web application server. IBM Tivoli Composite Application Manager for WebSphere supports the following WebSphere levels:

- WebSphere Application Server V5.1.*x*
- WebSphere Application Server V6.*x*

The embedded installation installs WebSphere Application Server V6.0 and upgrades it with FP1. Other levels of WebSphere Application Server are supported, but you cannot use them for embedded installation.

2.3.3 Operating system specific requirements

Other requirements for a managing server vary for different operating systems. The product requires some facilities that are not installed by default or may require additional components.

For Windows operating system, the IBM Tivoli Composite Application Manager for WebSphere managing server requires the installation of Microsoft Services for UNIX. Microsoft Services for UNIX allows a UNIX-like environment to run on

the Windows platform, such as executing UNIX shell scripts or running some basic UNIX programs.

For other operating systems requirements, refer to the IBM Tivoli Composite Application Manager for WebSphere documentation page at:

http://publib.boulder.ibm.com/tividd/td/ITCAMWAS/prereq60/en_US/HTML/itcam6.html

2.4 Networking requirements

From the networking point of view, IBM Tivoli Composite Application Manager for WebSphere as a distributed application may require some additional consideration, such as:

- 2.4.1, “Port usage” on page 21
- 2.4.2, “Communication traffic” on page 22
- 2.4.3, “Communication security” on page 22

2.4.1 Port usage

IBM Tivoli Composite Application Manager for WebSphere uses several ports for communication between the data collectors and the managing server. Figure 2-1 shows the port usage.

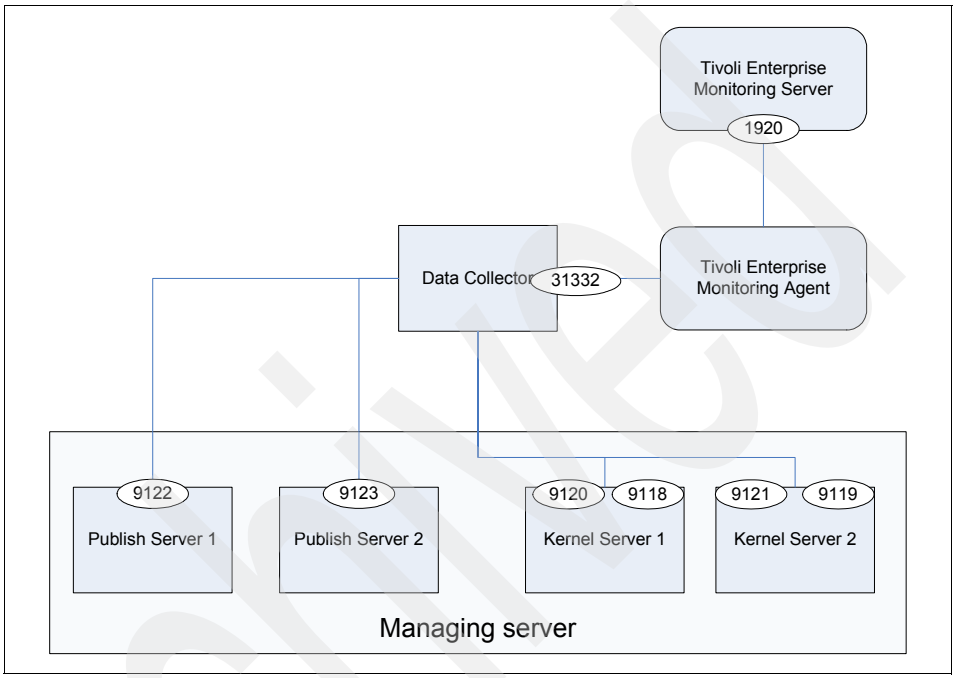


Figure 2-1 Port usage

You may require an additional port to be opened in the firewall rule. You can configure the data collector ports using a configuration file. However, the data collector requires communication to at least six ports of the managing server (four for kernel and two for publish server). If you install more than one data collector on a node, you must open additional ports and assign unique values for each in the respective datacollector.properties configuration files.

We do not use or configure a firewall for this basic deployment guide. For more information, refer to the IBM Tivoli Composite Application Manager for WebSphere documentation page at:

http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.itcamwas.doc/toc.xml

Note: For a complex installation where firewall security is an issue, you can use an optional component, the port consolidator, to minimize the number of open ports required in the firewall.

2.4.2 Communication traffic

The traffic between the data collector and the managing server happens at the following times:

- At initialization time, when the data collector downloads its configuration from the managing server. The configuration information can be in the order of several KB.
- Transaction information, when the data collector informs the managing server of each transaction start and stop. Each notification contains the data bytes of the data collector and the information that explains the data. This is the bulk of the communication load, especially if the monitoring is performed in Level 3.
- Server activity display and in-flight request search that is triggered by the visualization engine request to collect in-flight task information.

2.4.3 Communication security

The communication between the data collector and the managing server can potentially reveal the structure of the WebSphere application. If you need to encrypt this information, IBM Tivoli Composite Application Manager for WebSphere provides Secure Sockets Layer (SSL) encryption mechanism. You have to deploy SSL certificates for all the data collectors and the managing server in which you want SSL to be enabled.

2.5 Data collector considerations

The data collectors are installed in each application server to be monitored. The data collector installation modifies the WebSphere Application Server setting. It defines new custom services and creates a set of new variables. This may impact updates or fixes to the WebSphere Application Server.

The new custom services is started using the -Xrun switch. For a Java Virtual Machine (JVM), there can be only one -Xrun switch argument. Therefore, if there are any other services that must be run using -Xrun switch such as the IBM Tivoli Composite Application Manager for Response Time Tracking, the data collector cannot be installed.

Note: Coexistence with the IBM Tivoli Composite Application Manager for Response Time Tracking J2EE monitoring component can be achieved if you install the IBM Tivoli Composite Application Manager for WebSphere data collector first. The IBM Tivoli Composite Application Manager for Response Time Tracking J2EE monitoring component can use -Xrunvirt, which allows multiple custom services to be launched from a single -Xrun switch.

The data collector is installed in a two-part process: the installation of the data collector files and the creation of data collector instance inside the WebSphere Application Server. You can install the data collector on multiple WebSphere Application Servers in a single machine.

2.6 Deployment options

Based on the size of the deployment, we consider the following scenarios:

- 2.6.1, “Proof of concept deployment” on page 23
- 2.6.2, “Simple production deployment” on page 24
- 2.6.3, “Large-scale environment deployment” on page 24

2.6.1 Proof of concept deployment

This kind of deployment squeezes the product into a small environment with low intermittent load. The product is installed in the shortest possible path. This deployment is performed when the performance is of low concern.

You may have to perform this kind of deployment on a less than recommended hardware feature. You have to perform some tuning action to make the product run as expected. This includes reducing the amount of pre-allocated Java heap for the managing server processes and removing the redundancy of the archive agent and publishing server.

To reduce the memory requirement of IBM Tivoli Composite Application Manager for WebSphere, consider the following:

- Do not start the second archive agent (AA2) and publishing server (PS2). Remove references to AA2 and PS2 from am-start.sh and am-stop.sh.
- Do not start an unnecessary process:
  - The message dispatcher may be stopped if you do not use the trap-and-alert feature.
  - The polling server may be stopped if you do not monitor the Apache Web servers.

  - The global publishing server may be stopped if you do not correlate transaction running in different WebSphere applications.
- Reduce the memory requirement for the kernel components. These memory sizes are stored in setenv.sh program.

2.6.2 Simple production deployment

In this type of deployment, you must meet all the hardware and software requirements. The product is expected to run continuously for a long period of time and it has to meet a relevant production performance objective.

You can perform this installation with the default installation methods. Some parameter changes may be necessary to cope with the workload and to tune the solution. A typical configuration allows monitoring for 50 application servers. This includes default memory sizes. Additional application servers or transaction loads may require that you boost performance by increasing the memory size configurations of the publishing server and the archive agent.

2.6.3 Large-scale environment deployment

For a large-scale deployment, such as an environment with hundreds of application servers or with total transaction rates higher than 100 transactions per second, you must implement IBM Tivoli Composite Application Manager for WebSphere a little differently.

- Performance considerations imply that the servers should be extended across multiple machines.
- Extensive database processing requires the use of a separate database server.
- Additional deployment considerations may include a mass installation method instead of the manual wizard-based installation approach.
- Maintenance can also be a major pain point for large-scale deployments. It may not be feasible to update a large number of data collectors manually. You must schedule patch installation to critical production servers and consider the impact of this. For more consideration, refer to *Large Scale Implementation for IBM Tivoli Composite Application Manager for WebSphere*, REDP-4162

3

Installing IBM Tivoli Composite Application Manager for WebSphere

In this chapter, we provide step-by-step instructions for installing IBM Tivoli Composite Application Manager for WebSphere. These instructions also include some basic initial customization steps that you must perform. Chapter 4, “Using IBM Tivoli Composite Application Manager for WebSphere” on page 77 provides more usage samples and scenarios. This chapter consists of the following sections:

3.1 Installation overview

There are several deployment schemes that we can choose, as discussed in 2.6, “Deployment options” on page 23. This deployment guide chooses to document in detail the deployment for a simple production environment discussed in 2.6.2, “Simple production deployment” on page 24. We consider this to be the most common deployment schema that customers will have.

You can also use this guide for the proof of concept deployment option that we discussed in 2.6.1, “Proof of concept deployment” on page 23. However, you have to perform some additional modifications and configuration changes to ensure that the limited capacity of the proof of concept system does not hinder demonstration of the product’s capability.

A description of the large-scale deployment that we discussed in 2.6.3, “Large-scale environment deployment” on page 24 is beyond the scope of this deployment guide. Much more analysis and installation design must be performed before even starting the product installation. For more information about large-scale deployment, refer to *Large Scale Implementation for IBM Tivoli Composite Application Manager for WebSphere*, REDP-4162 for more consideration.

The installation process that we discuss in this chapter consists of the following steps:

1. Installing the managing server. You must install the managing server before any other IBM Tivoli Composite Application Manager for WebSphere component. We provide step-by-step instructions for installing the managing server on Windows platforms in 3.2, “Installing IBM Tivoli Composite Application Manager for WebSphere managing server” on page 27.
2. After the managing server is up and running, you can begin deploying the data collectors. The data collectors are installed on the WebSphere Application Server that hosts composite application components. Each application server must have its own data collector instance installed. We provide step-by-step instructions for the data collector installation in 3.3, “Installing IBM Tivoli Composite Application Manager for WebSphere data collector” on page 51.
3. After you install and connect the managing server and the data collectors, you have to configure them. You have to perform additional administrative tasks to define users and server groups. We discuss these tasks in 3.4, “Defining data collectors, server groups, and users” on page 61.
4. Optionally, you can install the IBM Tivoli Composite Application Manager for WebSphere’s Tivoli Enterprise Monitoring Agent. This allows WebSphere metrics to be forwarded to Tivoli Enterprise Monitoring Server for display

using the Tivoli Enterprise Portal. Outside the basic scope of this guide, Appendix A, “Tivoli Enterprise Monitoring Agent component” on page 127 provides step-by-step instructions for this optional installation procedure.

3.2 Installing IBM Tivoli Composite Application Manager for WebSphere managing server

We perform the managing server installation on Windows platforms using the embedded installation method. There are several prerequisite steps that you must perform before installing the managing server. This section assumes that you have installed Windows 2000 server with the latest fix pack that conforms to the machine requirement for the IBM Tivoli Composite Application Manager for WebSphere managing server. We discuss the following tasks in this section:

- 3.2.1, “Defining users and access rights” on page 27
- 3.2.2, “Installing Microsoft Services for UNIX” on page 33
- 3.2.3, “Running the IBM Tivoli Composite Application Manager for WebSphere installation wizard” on page 39

3.2.1 Defining users and access rights

IBM Tivoli Composite Application Manager for WebSphere requires several user IDs. Some of them are created by the installation, and you must create some of them before installing the product. Furthermore, the user ID that performs the installation may require some additional access rights to be able to successfully install the IBM Tivoli Composite Application Manager for WebSphere managing server.

Creating the installation user

To create a new user ID with membership in the Administrators group, perform the following steps:

1. Log in as the administrator for the system.
2. To open the Computer Management window, select **Start** → **Administrative Tools** → **Computer Management**.

3. Expand the **Local Users and Groups** folder, right-click **Users** and select **New User**, as shown in Figure 3-1.

> **Note:** In this example, we create a user with the name itcamadm to perform the installation. This name is not mandatory, you can use any name appropriate for your environment. This user will be the IBM Tivoli Composite Application Manager for WebSphere administrator.

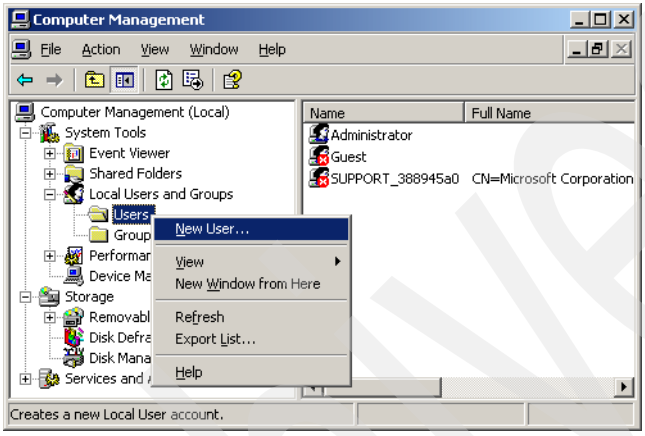


Figure 3-1 Creating the installation user

4. In the New User window (Figure 3-2), perform these tasks:
   a. Enter the user name `itcamadm` for the installation and administration user.
   b. Enter a password.
   c. Deselect the box for **User must change password at next logon**.
   d. Click **Create**.
   e. Click **Close** when done.

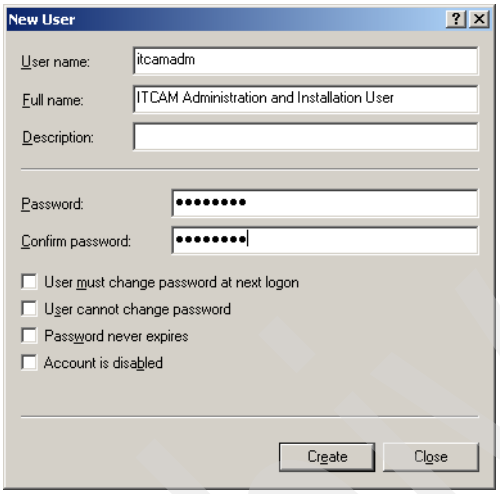


Figure 3-2 Setting the installation user password

5. Select the **Groups** folder in the tree view, right-click **Administrators** in the detail view, and select **Add to Group**, as shown in Figure 3-3.

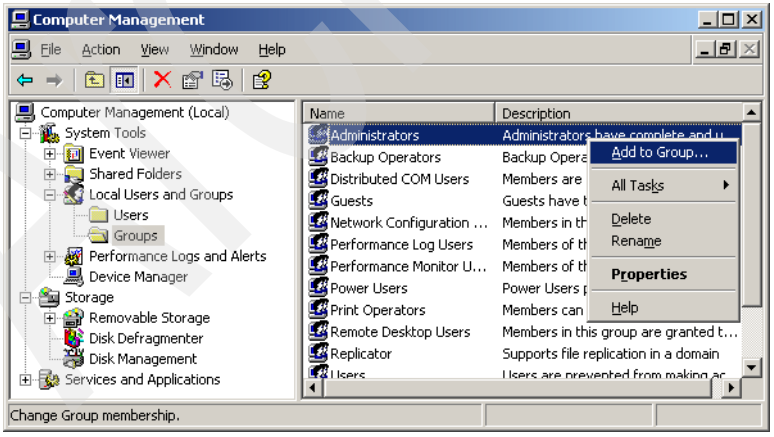


Figure 3-3 Adding to the administrators group

6. Click the **Add** button, type the user name just created, `itcamadm`, and click **OK**.
7. In the Select Users window, click **Check Names**, as shown in Figure 3-4. Click **OK**.

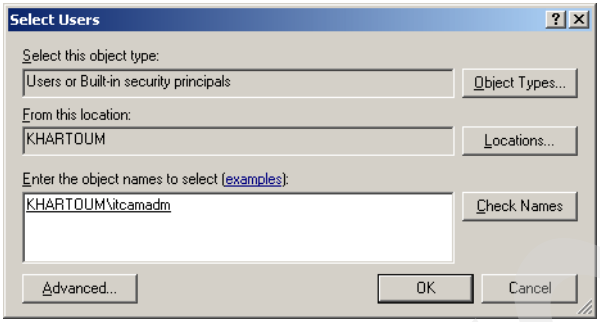


Figure 3-4 Adding the installation user to the administrators group

8. Click **OK** again to close the Administrators Properties window. We have now added itcamadm to the administrators group.
9. Close the Computer Management window by selecting **File** → **Exit**.

Modifying the installer user access rights

After you create the installation and administration user, itcamadm, it is necessary to assign the following additional access rights to this user:

- Act as part of the operating system
- Replace process level token
- Log on as a service

To do this, perform the following steps:

1. Open the Local Security Settings window by selecting **Start** → **Administrative Tools** → **Local Security Policy**. Alternatively, run `secpol.msc` from the command line.
2. Expand the **Local Policies** folder and select the **User Rights Assignment** folder.

3. In the detail view on the right, double-click the **Act as part of the Operating System** policy, as shown in Figure 3-5.

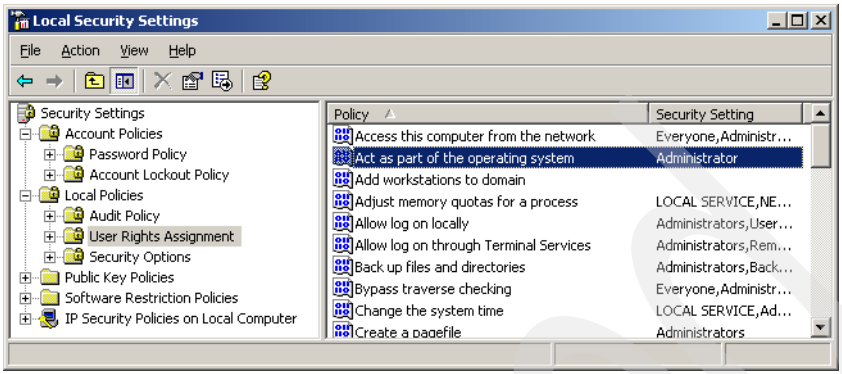


Figure 3-5 Adding local security settings

4. Click the **Add User or Group** button.
5. Type the user name, `itcamadm`. Click **Check Names**. Click **OK**.
6. Verify that the Administrator is also in the list with this policy. If not, add it as well. The results look similar to Figure 3-6.

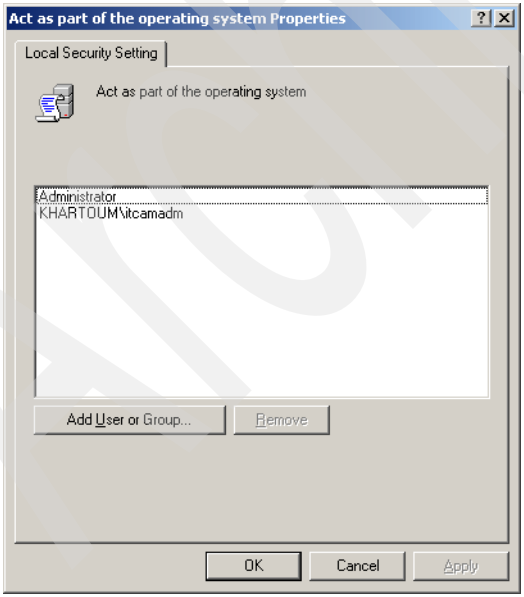


Figure 3-6 Adding the Act as part of the operating system policy

7. Add itcamadm to the Log on as a Service policy using similar steps. In the Local Security Settings window (Figure 3-5), double-click the policy **Log on as a Service**.
8. Click the **Add User or Group** button.
9. Type the user name, `itcamadm`. Click **Check Names**. Click **OK**.

   The results look similar to Figure 3-7.

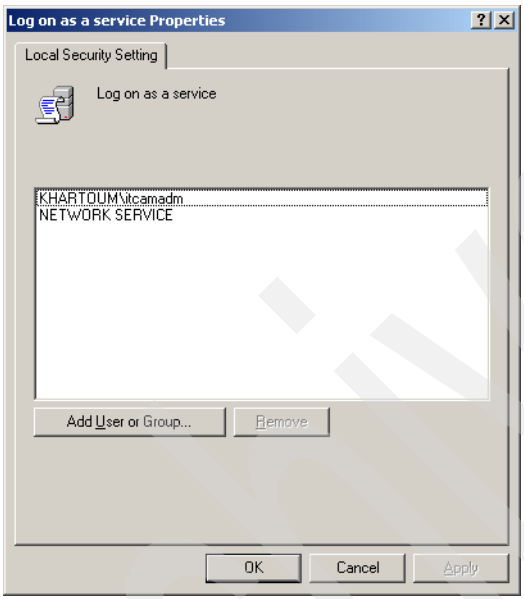


Figure 3-7 Adding the Log on as a service policy

10. Close all the windows.
11. Restart the computer.
12. Log in as the newly created managing server installation and administration user, itcamadm.

Changing the user mode

If you are running a terminal server or use a terminal server session, you may have to enable the administrator user to perform the installation. To perform this, use the **`change user /install`** command.

Figure 3-8 shows the sample execution of the change user commands.

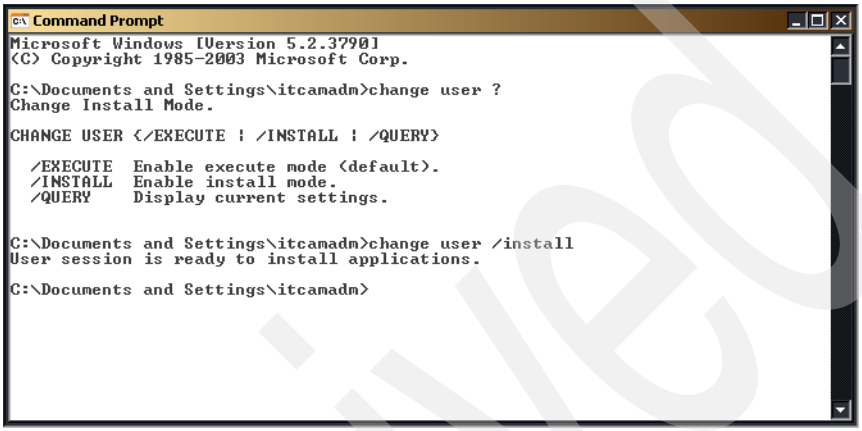


Figure 3-8 Change user command

3.2.2 Installing Microsoft Services for UNIX

Download and install Microsoft Services for UNIX before installing the IBM Tivoli Composite Application Manager for WebSphere managing server. Microsoft Services for UNIX is available at the following Web site:

`http://www.microsoft.com/windowsserversystem/sfu/`

The download is a compressed file package SFU35SEL_EN.exe. When you run it, the installation files are generated. The extraction dialog is shown in Figure 3-9.

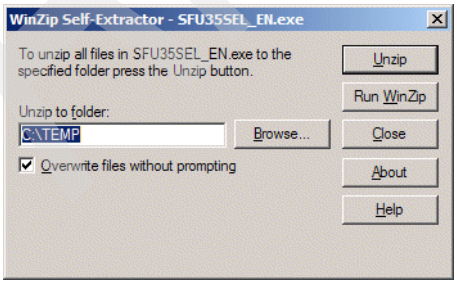


Figure 3-9 Extraction dialog

To install Microsoft Services for UNIX, perform the following steps:

1. Start the installation program by running setup.exe from the installation directory. The initial dialog is shown in Figure 3-10. Click **Next**.

Figure 3-10 Welcome dialog

2. Enter your user name and organization information, as shown in Figure 3-11. Click **Next**.

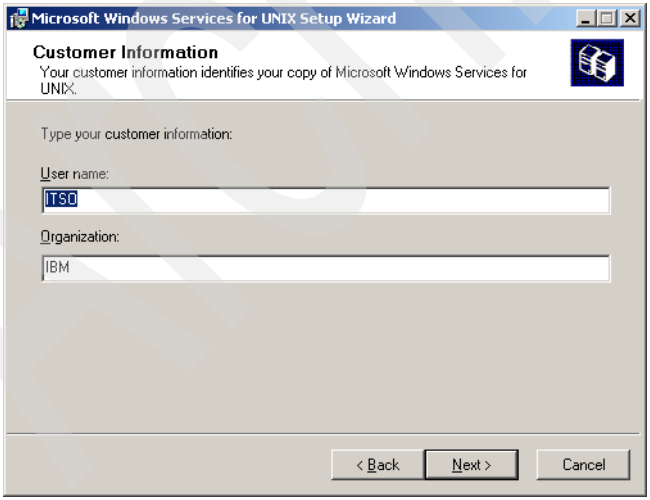


Figure 3-11 User name and organization

3. Read and select the accept license agreement button, if appropriate, as shown in Figure 3-12. Click **Next**.

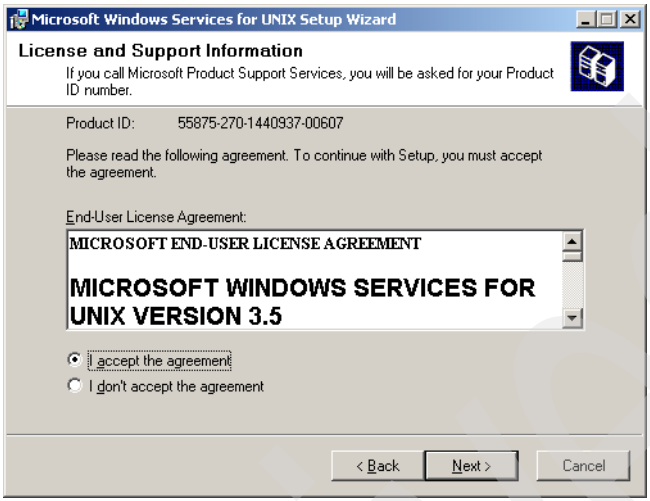


Figure 3-12 License agreement

4. Select **Standard Installation**, as shown in Figure 3-13. Click **Next**.

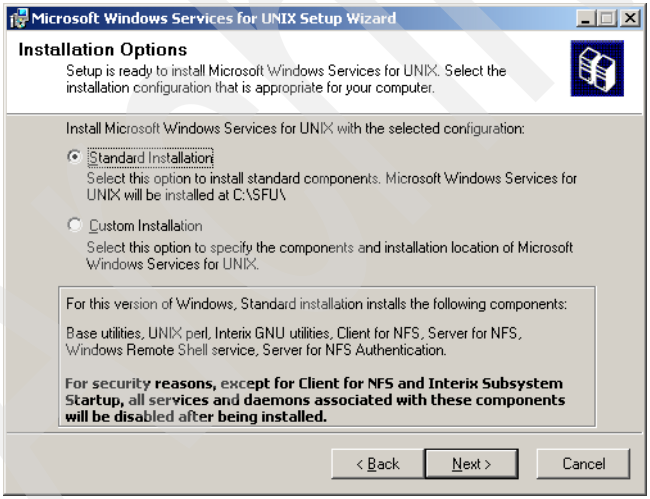


Figure 3-13 Selecting the installation method

5. In the security setting dialog (Figure 3-14), do not check any boxes. Click **Next**.

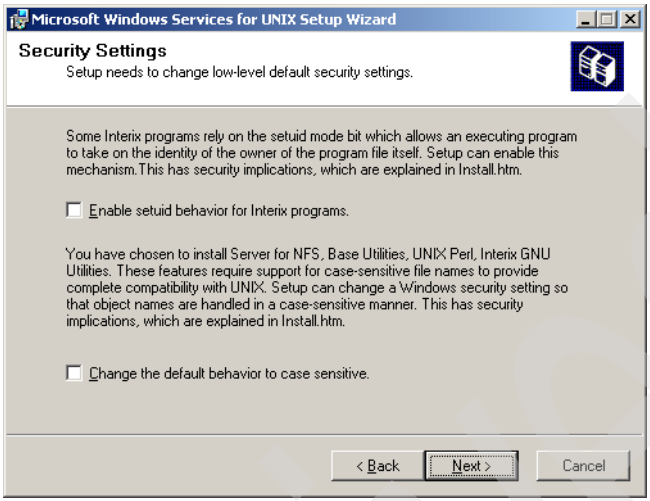


Figure 3-14 Security setting

6. As the users that we use are local users, select the default **Local User Name Mapping Server**, as shown in Figure 3-15. Click **Next**.

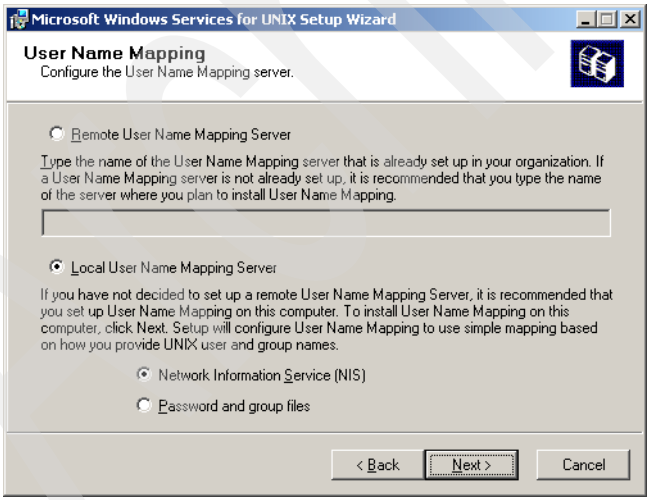


Figure 3-15 User name mapping

7. Keep the default for user name mapping dialog shown in Figure 3-16. Click **Next**.

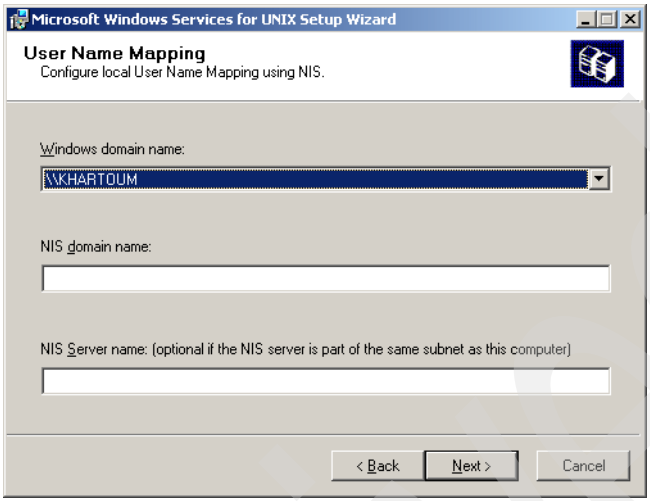


Figure 3-16 Mapping options

8. The installation will proceed and transfer the files. When the installation is complete, the window shown in Figure 3-17 opens. Click **Finish**.

Figure 3-17 Installation completed

9. After the installation of Microsoft Services for UNIX is complete, you can add a new path to your system path environment variable. From **Control Panel** → **System**, select the **Advanced** tab and click **Environment Variables**.

10. In the Environment Variable window, in the lower half of the window for System variables, scroll down to select the **Path** variable and click **Edit**. Enter additional path, as shown in Figure 3-18. Click **OK**.

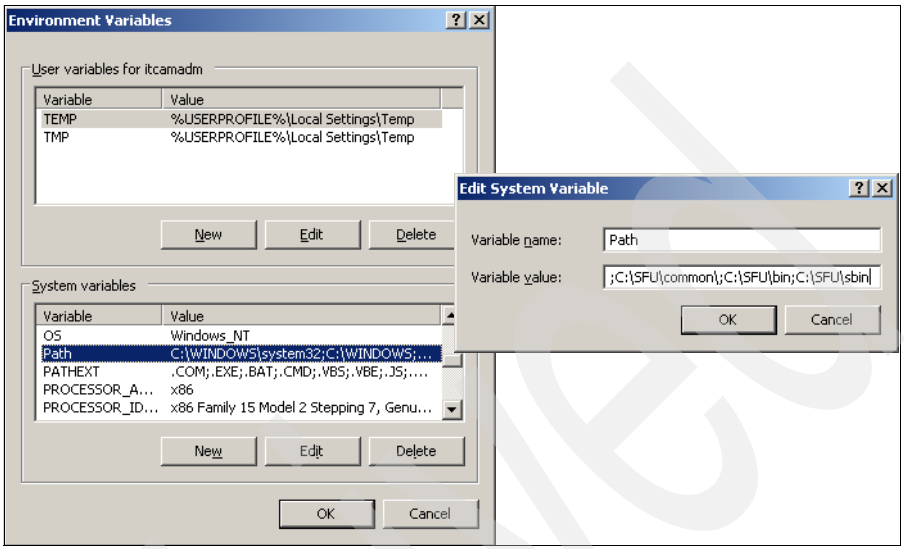


Figure 3-18 Modifying the system path

Now that you have installed Microsoft Services for UNIX, you can start the installation of the IBM Tivoli Composite Application Manager for WebSphere managing server.

3.2.3 Running the IBM Tivoli Composite Application Manager for WebSphere installation wizard

If you install the managing server from a physical CD-ROM, the installation wizard will copy the CD-ROM content to a temporary directory so that the embedded installation can use the CD-ROM to load additional products such as DB2 Universal Database or WebSphere Application Server.

Perform the following steps:

1. Start the installation wizard by using the launch pad (Figure 3-19). Click the **Install ITCAM** button.

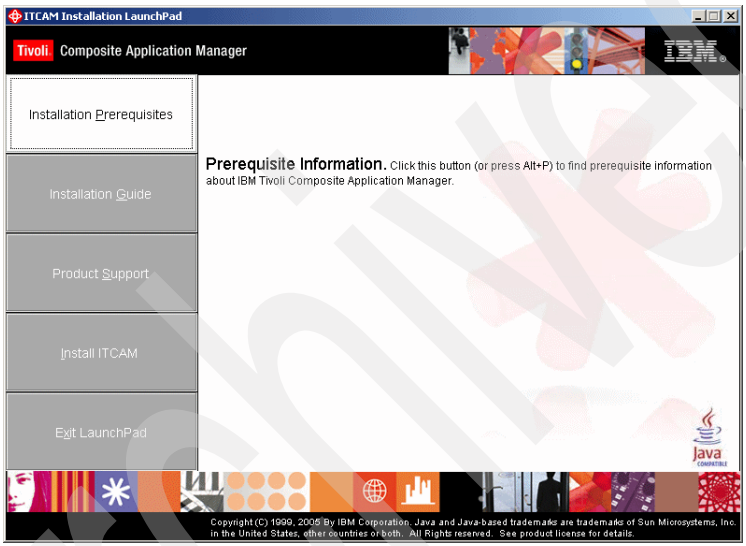


Figure 3-19 IBM Tivoli Composite Application Manager for WebSphere launch pad

2. The IBM Tivoli Composite Application Manager for WebSphere installation wizard welcome dialog is shown in Figure 3-20. Click **Next**.

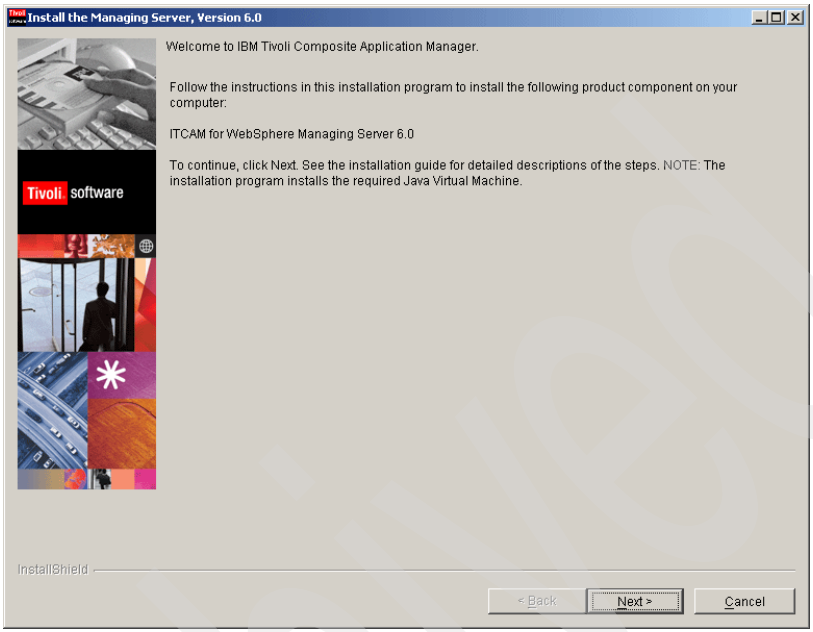


Figure 3-20 Welcome dialog

3. Accept the software license agreement as shown in Figure 3-21. Click **Next**.

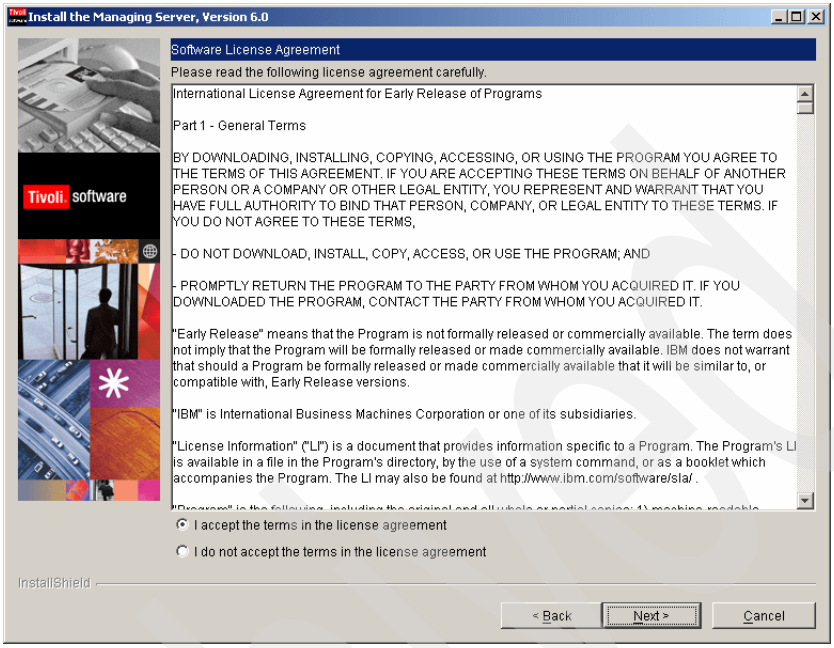


Figure 3-21 Software license agreement

4. The dialog in Figure 3-22 asks you whether the installation is performed from a CD-ROM. For CD-ROM installation, the wizard copies the installation image to a temporary directory and restarts itself. We are installing the product from a local drive. Select **No**. Click **Next**.

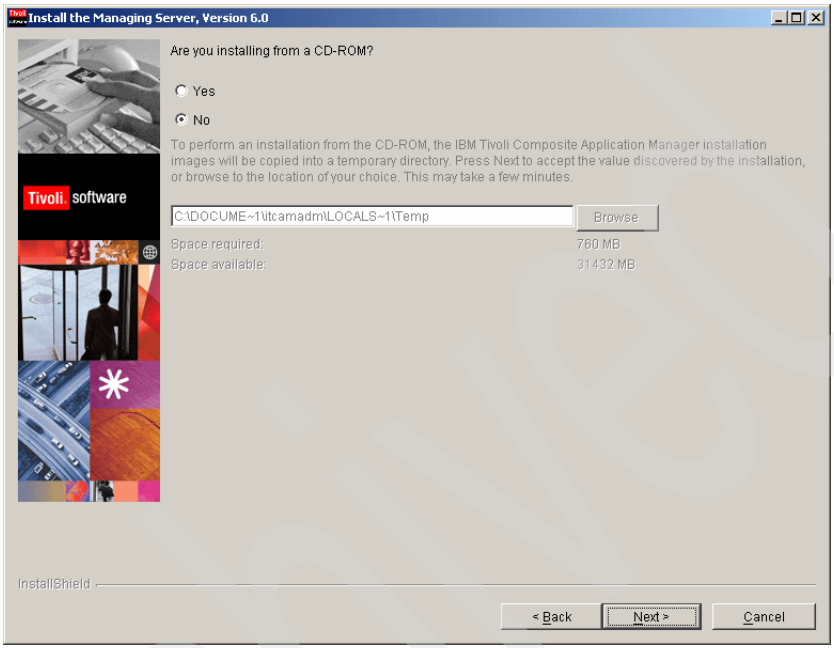


Figure 3-22 Media selection dialog

5. Enter the path that you want to install IBM Tivoli Composite Application Manager for WebSphere on, as shown in Figure 3-23. The default path for Windows system is C:\Program Files\IBM\itcam\WebSphere\MS. Click **Next**.

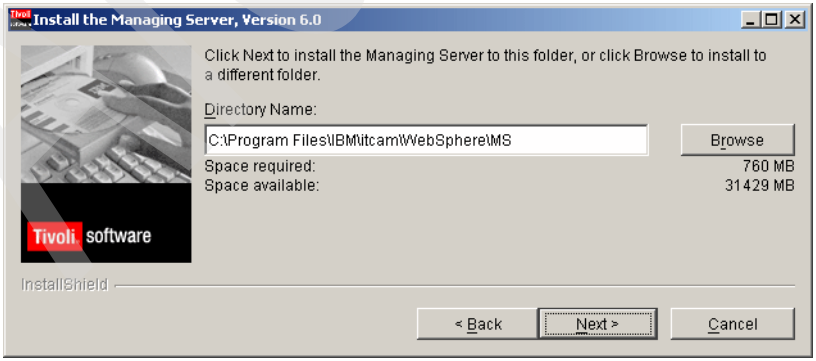


Figure 3-23 Directory path

Note: In this guide, we refer to this managing server installation location as $AM_HOME.

6. For an embedded complete installation, select all the features in the IBM Tivoli Composite Application Manager for WebSphere managing server, as shown in Figure 3-24. Optionally, you can install the visualization engine and database tables separately. Click **Next**.

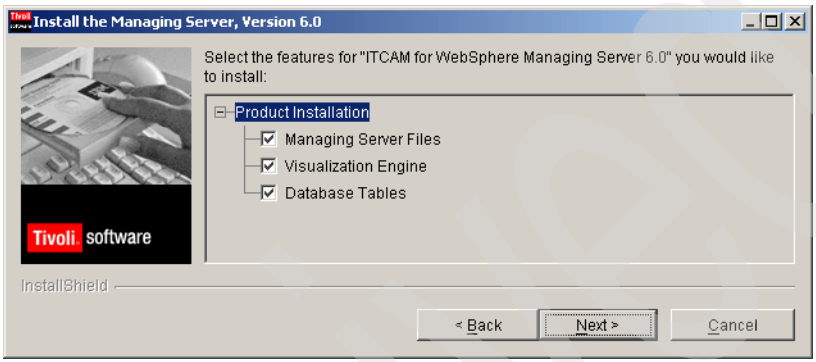


Figure 3-24 Component list

7. Select **Install DB2**, as shown in Figure 3-25. Click **Next**. You can also use an existing DB2 or Oracle database, if you already have them installed and the managing server has access to these database.

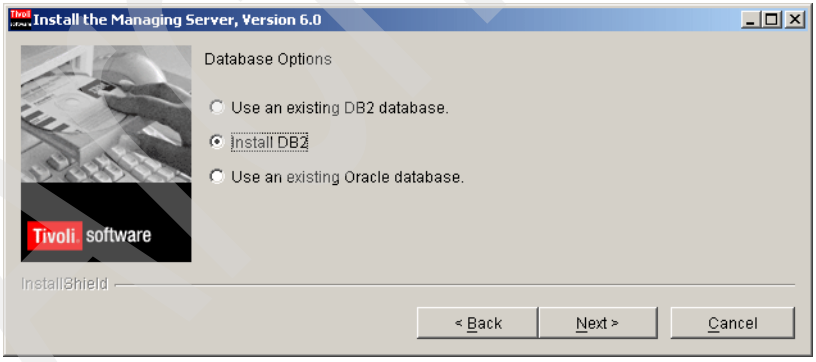


Figure 3-25 Database installation option

8. Because we install DB2 from scratch, the wizard asks for some DB2-related information. It asks for the DB2 instance user and administration user, typically these users are called dasusr1 and db2inst1. Enter the information in the User Name, Password, and Verify Password fields, as shown in Figure 3-26. Click **Next**.

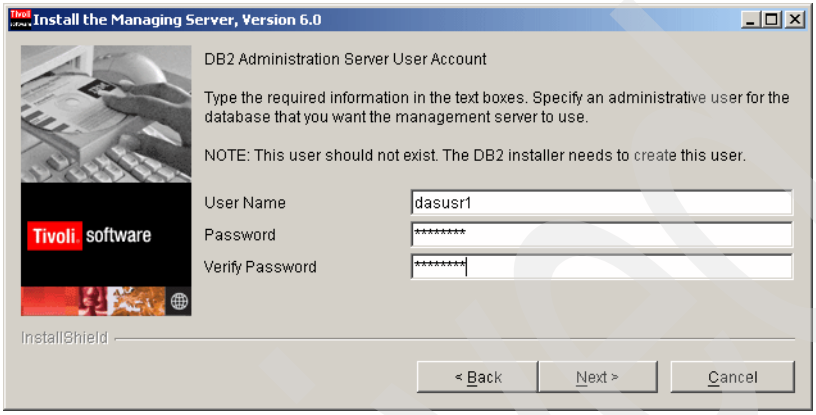


Figure 3-26 Selecting the DB2 user

9. For DB2 installation, you must provide the source of DB2 installation image. Figure 3-27 shows the dialog for DB2 installation image. Note that you must have the setup.exe of DB2 in the path. Click **Next**.

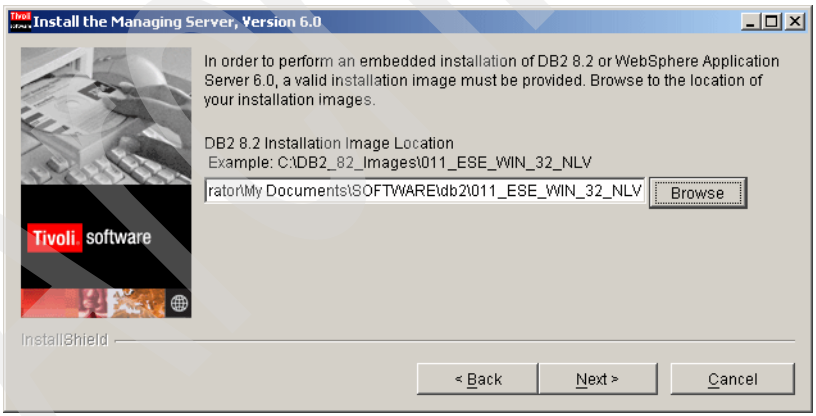


Figure 3-27 DB2 installation location

10. The DB2 installation starts as shown in Figure 3-28. This will take some time.

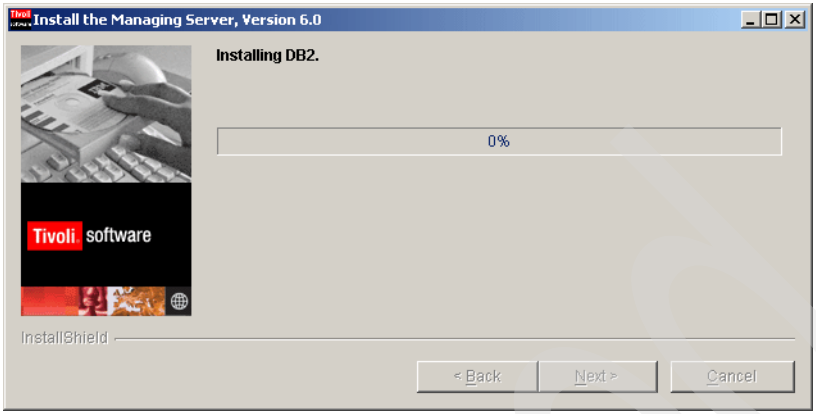


Figure 3-28 DB2 installation progress

11. After the DB2 installation is complete, the wizard collects information for creating the OCTIGATE database. Figure 3-29 shows the dialog. Most of the information here is prefilled because DB2 is installed using the embedded installation. You must supply an existing user ID for the Admin User prompt, this will be the user to log in initially to the Web console. Type `itcamadm` for this. Click **Next**.

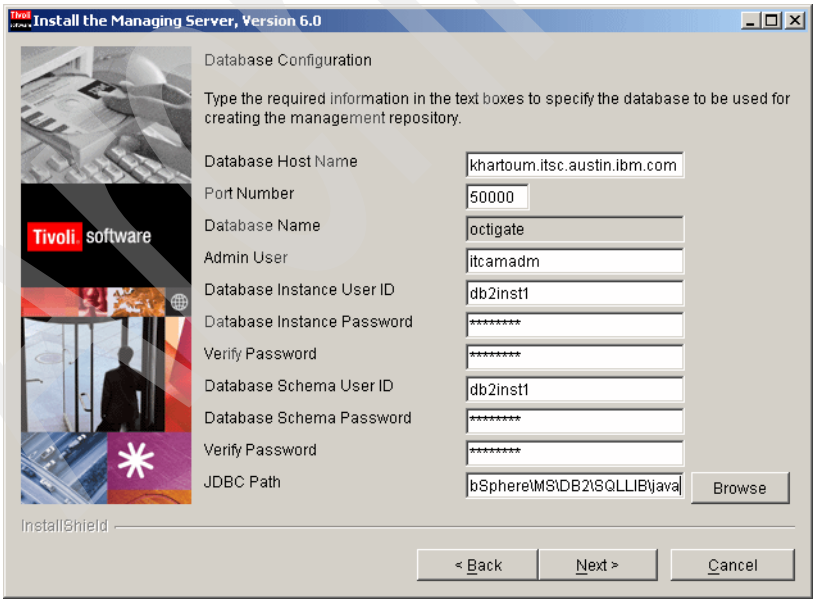


Figure 3-29 Creating database parameter

12. After the OCTIGATE database is created, the installer searches for the WebSphere Application Server installation. In this case, Figure 3-30 shows that we do not have WebSphere installed. Click **Next**.

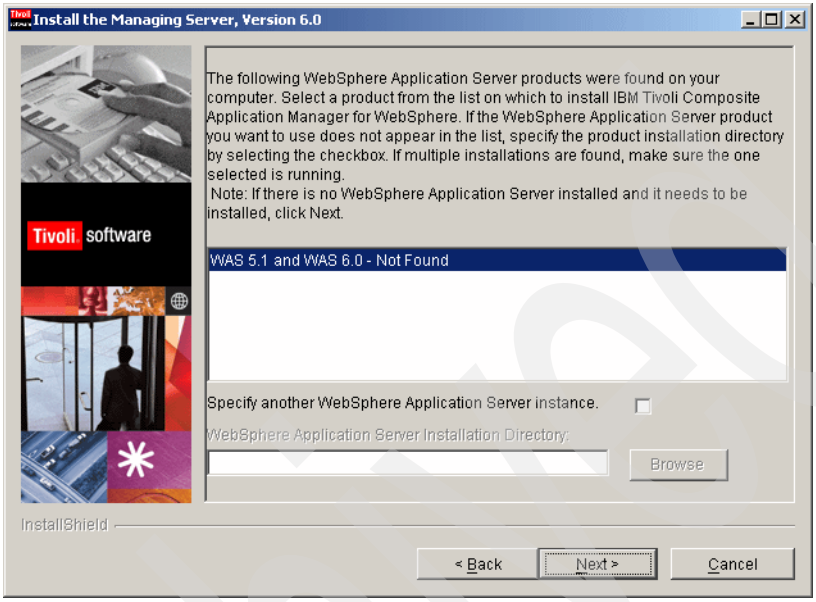


Figure 3-30 WebSphere not found dialog

13. Because we do not have any previous installation of WebSphere Application Server, the dialog in Figure 3-31 shows the WebSphere Application Server new installation parameters. Enter the password in the Password and Verify Password fields. Click **Next**.

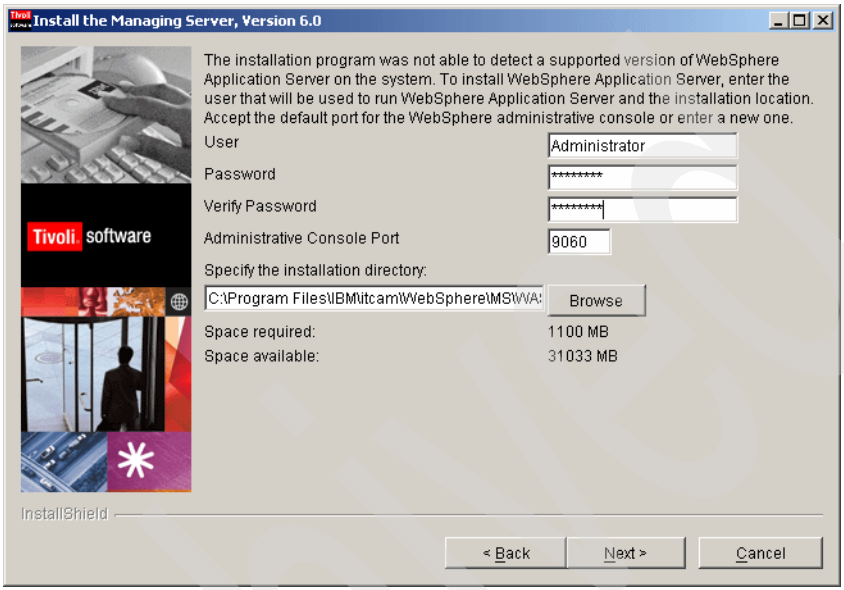


Figure 3-31 New installation of WebSphere

14. You must provide the installation source for both WebSphere Application Server V6.0 and refresh pack 1, as shown in Figure 3-32. Click **Next**.

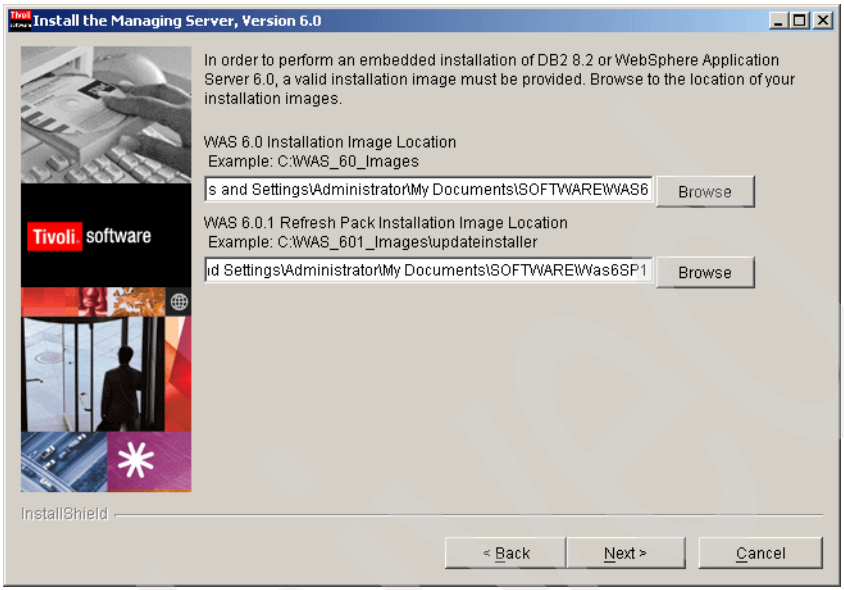


Figure 3-32 WebSphere Application Server installation sources

15. After you install all the prerequisites (DB2 Universal Database and WebSphere Application Server), you can start the actual IBM Tivoli Composite Application Manager for WebSphere managing server. You must specify two kernel machines for IBM Tivoli Composite Application Manager for WebSphere as shown in Figure 3-33. Click **Next**.

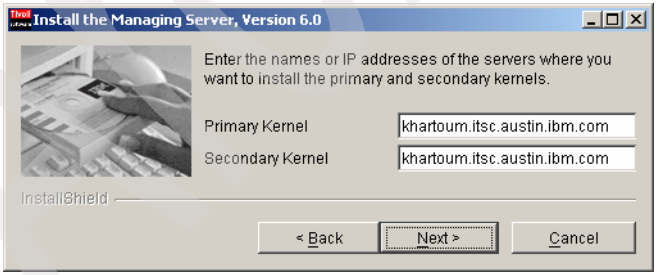


Figure 3-33 Kernel information

16. Each component of the managing server uses a Transmission Control Protocol Internet Protocol (TCP/IP) port to communicate with other components. The dialog in Figure 3-34 lists the port usages for IBM Tivoli Composite Application Manager for WebSphere managing server. You can change any of these port definitions. Click **Next**.

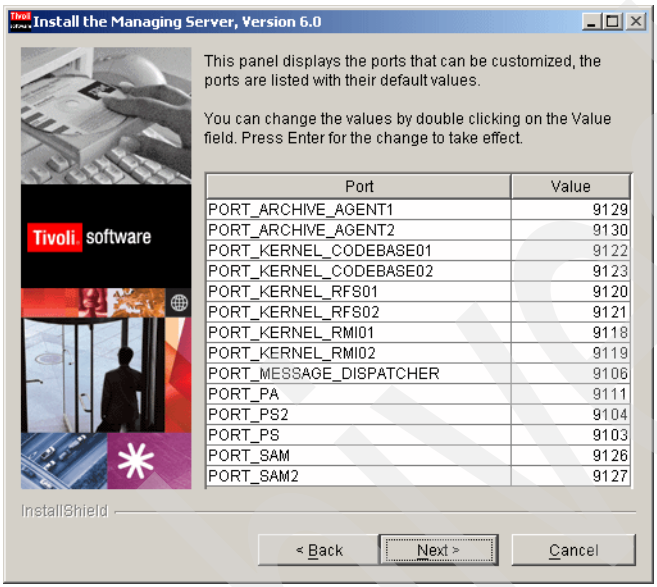


Figure 3-34 Port information

17. Before the installation begins, the summary information is displayed as shown in Figure 3-35. Click **Next** and the installer starts transferring the product files.

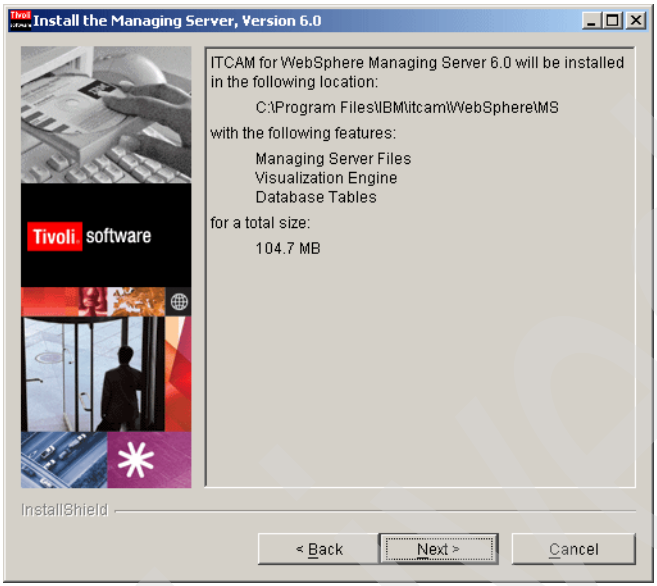


Figure 3-35 Installation summary

18. Figure 3-36 shows that the installation is complete. Click **Finish**.

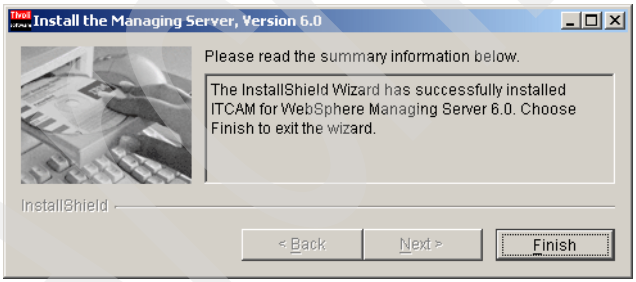


Figure 3-36 Installation complete

3.3 Installing IBM Tivoli Composite Application Manager for WebSphere data collector

The data collector installation is a two-step process. The first step installs the software on the WebSphere Application Server system. The second step configures the data collector for a specific application server instance on that system. The following sections detail these steps:

- 3.3.1, “Installing the data collector” on page 51
- 3.3.2, “Configuring the data collector” on page 58
- 3.3.3, “Configuring additional application server instances” on page 60

3.3.1 Installing the data collector

The following procedure installs the IBM Tivoli Composite Application Manager for WebSphere data collector on a WebSphere Application Server system. Install the data collector on a machine other than the one hosting the IBM Tivoli Composite Application Manager for WebSphere managing server.

Note: Installing the data collector on the same WebSphere Application Server instance as the one used by the managing server is not supported. If you have to install the data collector and managing server on the same machine, you must create a new instance of the WebSphere Application Server for the data collector prior to this installation.

To install the data collector, perform the following steps:

1. Verify that the WebSphere Application Server is running.
2. Start the Installation by using the **`setup_DC_w32.exe`** command either from the data collector installation media or a copy of the image in a local directory. The installation wizard opens with a welcome dialog, as shown in Figure 3-37. Click **Next**.

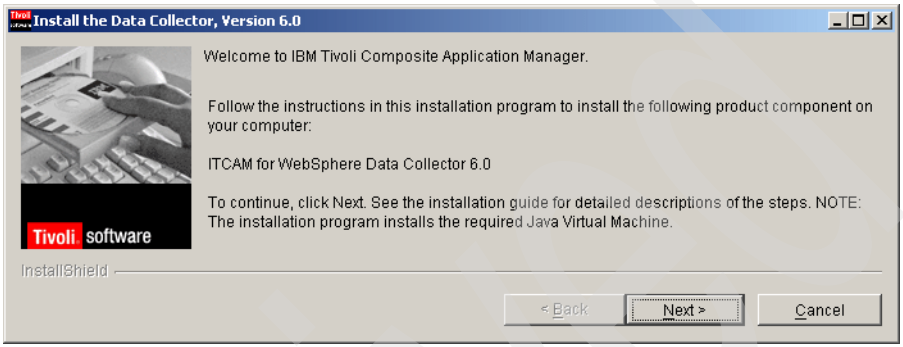


Figure 3-37 Welcome dialog

3. Select only the Application Monitor interface option, as shown in Figure 3-38.

   Do *not* select the IBM Tivoli Enterprise™ Portal interface option unless the IBM Tivoli Enterprise Monitoring V6.1 environment is already installed and available. The installation of the data collector used for Tivoli Enterprise Portal is covered in Appendix A, “Tivoli Enterprise Monitoring Agent component” on page 127. Click **Next**.

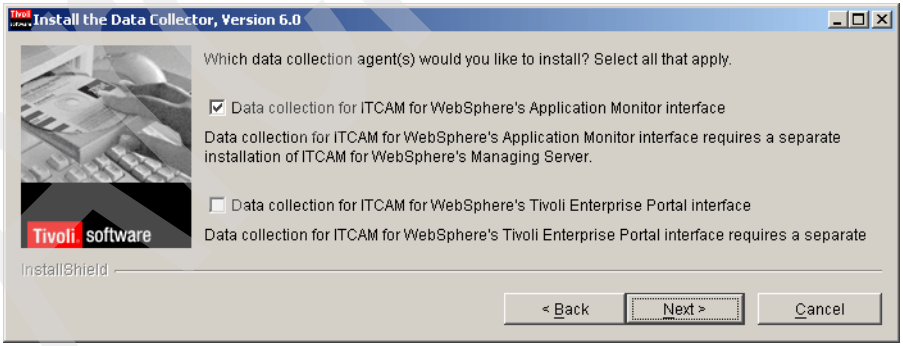


Figure 3-38 Selecting the installation components

4. Accept the license agreement, as shown in Figure 3-39. Click **Next**.

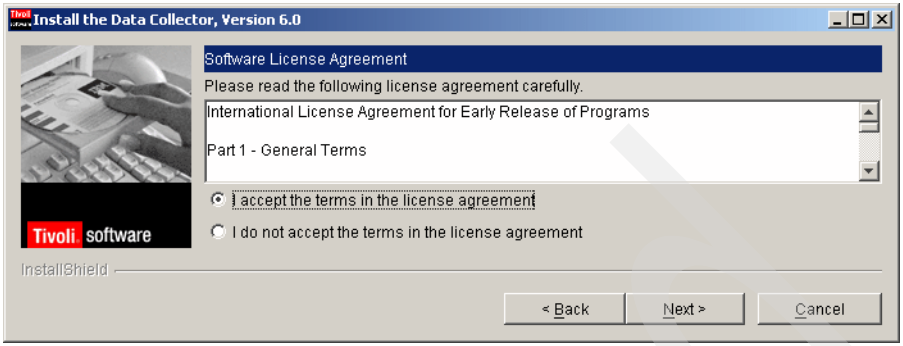


Figure 3-39 License agreement

5. The default prefilled installation path provided for Windows environments is displayed as `C:\Program Files\IBM\itcam\WebSphere\DC`. This default path will work. Because we do not want to have a space in the default installation path, it is eliminated. This is becoming a best practice for installing software on Windows. The install process will create the directory if it does not already exist. Enter `C:\IBM\itcam\WebSphere\DC`, as shown in Figure 3-40. Click **Next**.

> **Note:** In this guide, we refer to this data collector installation location C:\IBM\itcam\WebSphere\DC as $DC_HOME.

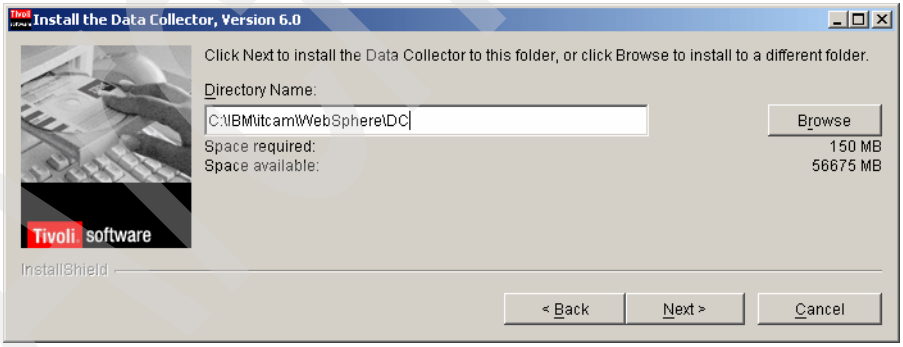


Figure 3-40 Selecting the installation path

6. Select **WebSphere Application Server** as the type of application server, as shown in Figure 3-41. WebSphere Portal Server is also an option but this is not covered in this guide.

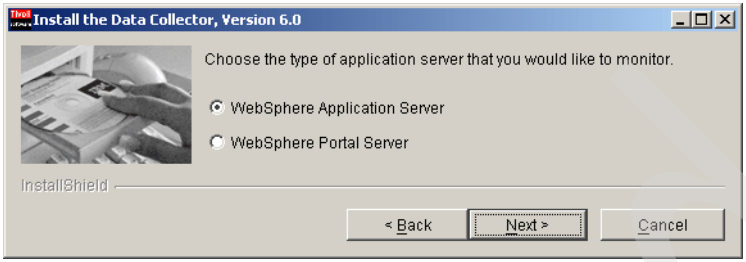


Figure 3-41 Selecting WebSphere type

7. The wizard detects any existing WebSphere Application Server installations and their associated profiles on this machine. Choose the application server profile to be instrumented with a data collector, as shown in Figure 3-42. Click **Next**.

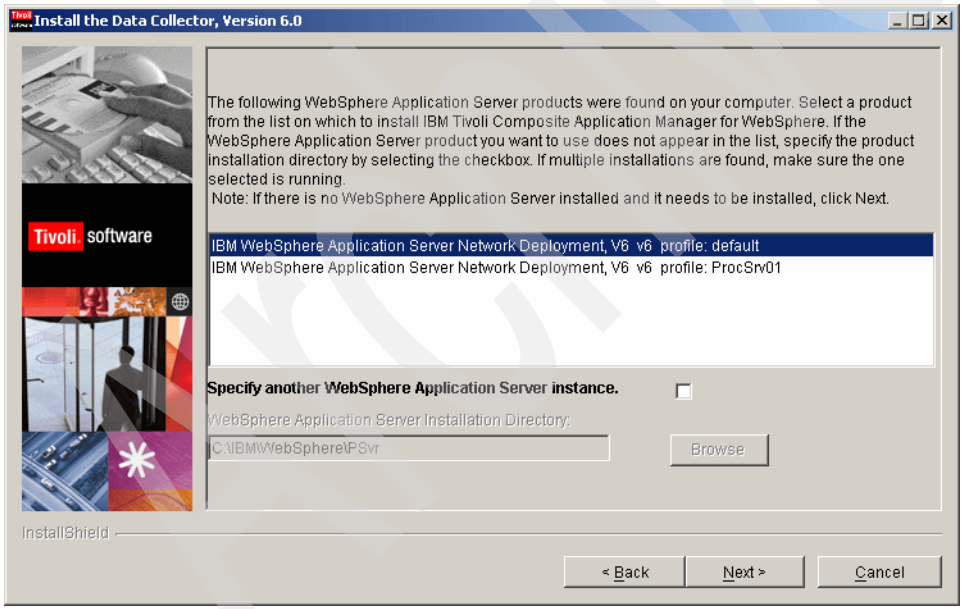


Figure 3-42 Existing WebSphere installation

8. Verify the gathered information in the new dialog window. The installed WebSphere Application Server is used to identify the Java environment that the IBM Tivoli Composite Application Manager for WebSphere data collector will use. This information is prepopulated, as shown in Figure 3-43. Click **Next**.

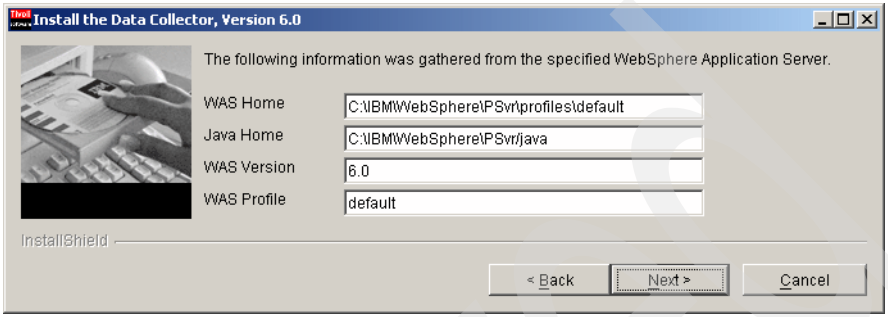


Figure 3-43 WebSphere properties

9. Enter the remote IBM Tivoli Composite Application Manager for WebSphere managing server connection properties for both the primary and secondary kernels, as shown in Figure 3-44. Click **Next**.

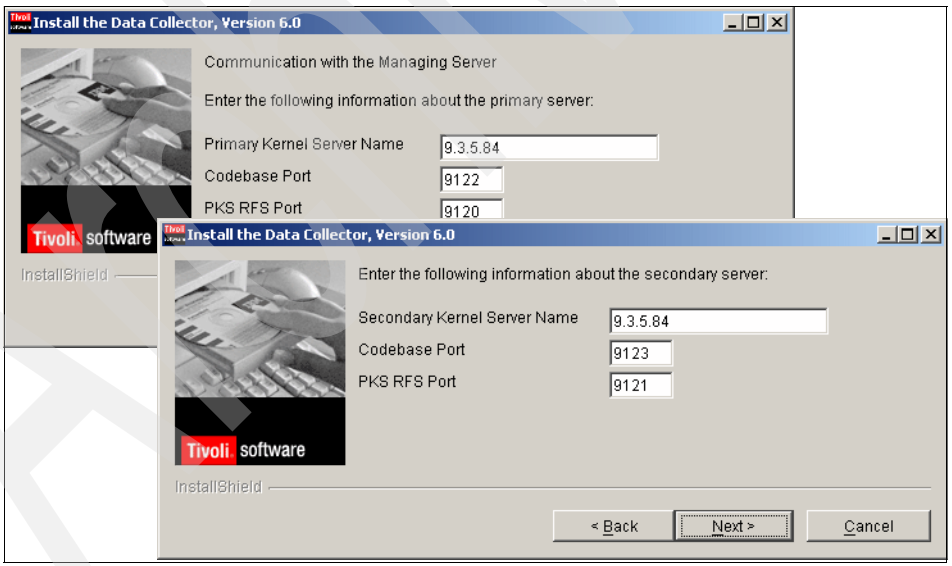


Figure 3-44 Managing server properties

10. In the next dialog window, specify the installation path of the managing server, as shown in Figure 3-45. Some instrumentation jar files will be downloaded from the managing server installation using this path. Click **Next**.

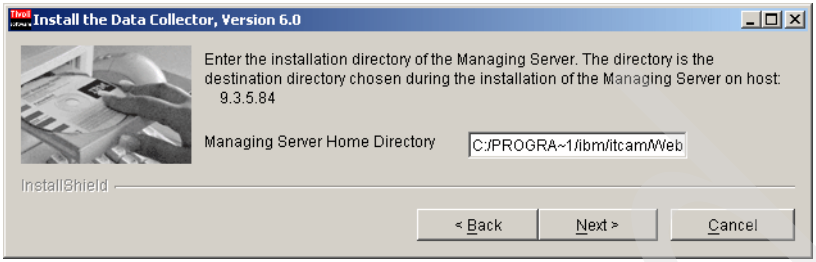


Figure 3-45 Managing server installation path

11. Some core installation summary information is displayed in the next window, as shown in Figure 3-46. Click **Next** to install the data collector.

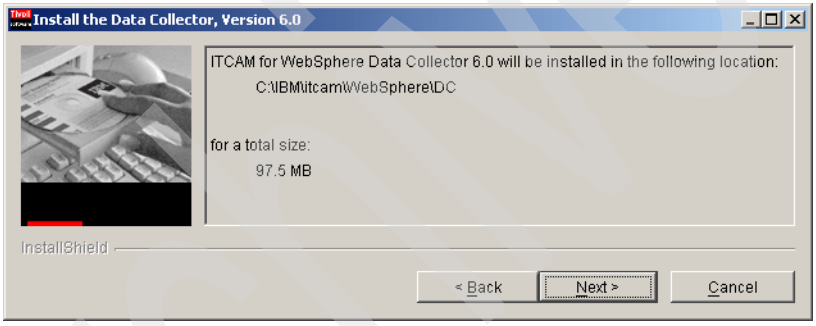


Figure 3-46 Installation summary

12. When the installation is complete, the resulting window (Figure 3-47) provides an option to either proceed and configure the data collector or to defer the configuration until later. Select **Launch the Configuration Tool**. Click **Next**.

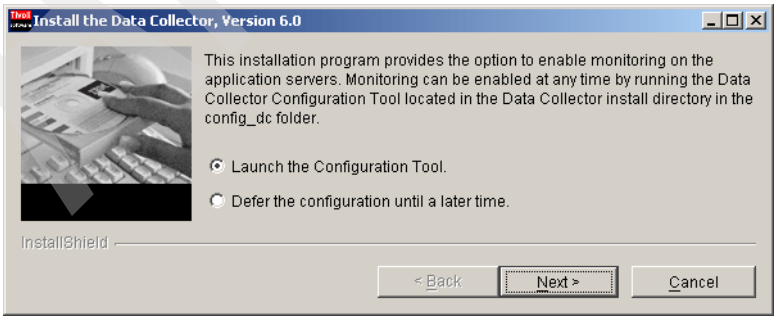


Figure 3-47 Launching the configuration tool

The wizard launches the data collector configuration wizard. We discuss this in 3.3.2, “Configuring the data collector” on page 58.

Note: You can manually launch the data collector configuration wizard using the appropriate script in the $DC_HOME/config_dc directory.

13. When the configuration process is complete, a dialog box from the data collector installation process opens, as shown in Figure 3-48. Click **Finish** to exit.

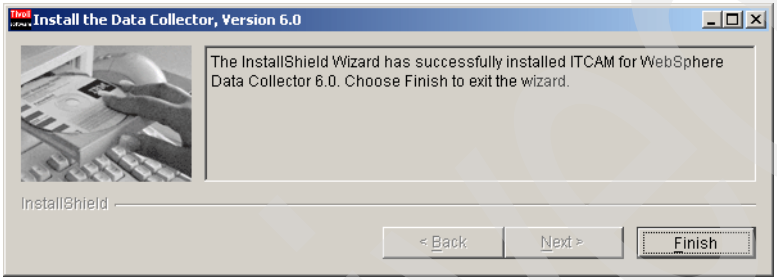


Figure 3-48 Installation completed

This completes the installation of the data collector.

3.3.2 Configuring the data collector

After you install the data collector on a WebSphere Application Server system, as described in 3.3.1, “Installing the data collector” on page 51, you have to configure the data collector.

1. The configuration tool welcome dialog opens as shown in Figure 3-49. This can be a continuation from the installation wizard or a new configuration. Click **Next**.

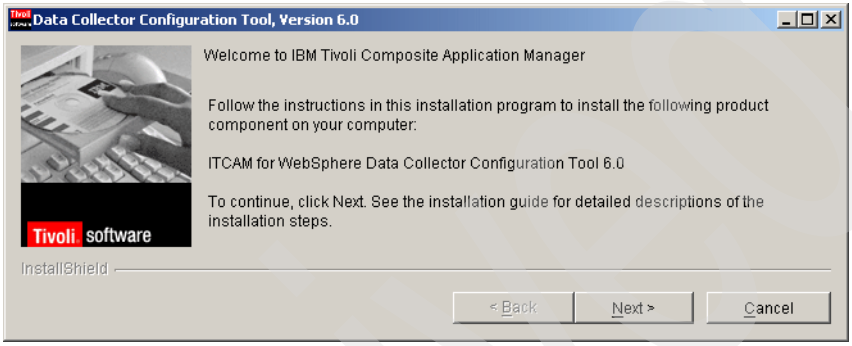


Figure 3-49 Configuration tool welcome dialog

2. The next dialog (Figure 3-50) shows both the configure and unconfigure options. As this is an initial installation, select the **Configure servers for data collection** option. Click **Next**.

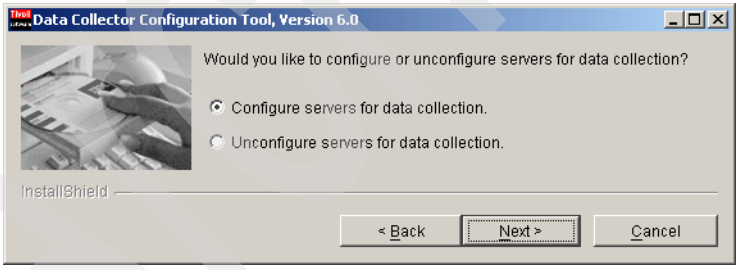


Figure 3-50 Configuration option dialog

3. The configuration tool has to access the unique SOAP Connector port belonging to a specific WebSphere Application Server instance to configure it for data collection and monitoring. Figure 3-51 shows the dialog where you have to enter this connection information. Click **Next**.

Note: For a WebSphere Application Server network deploy installation with multiple nodes, use the SOAP port for the deployment manager node. This configuration topic is beyond the scope of this guide.

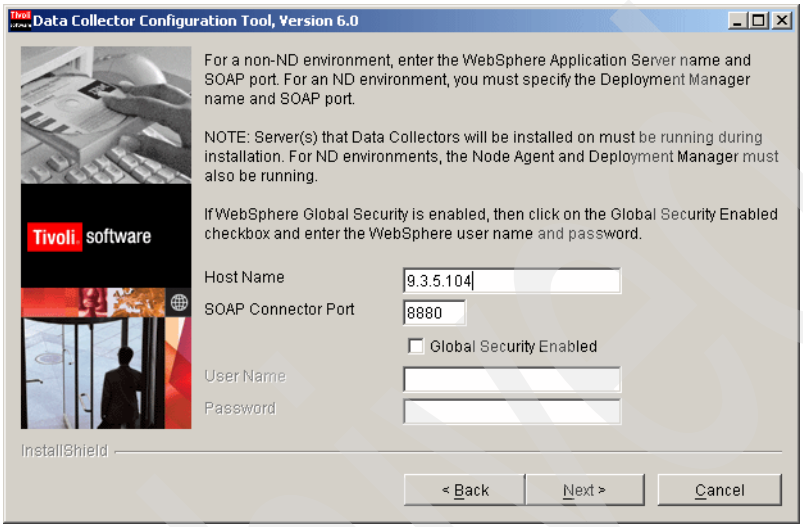


Figure 3-51 SOAP connection setup

4. The wizard retrieves the necessary information from the targeted WebSphere Application Server instance through the specified SOAP Connector port. This information is presented in a tree view, as shown in Figure 3-52. Select the application server instance and click **Next**.

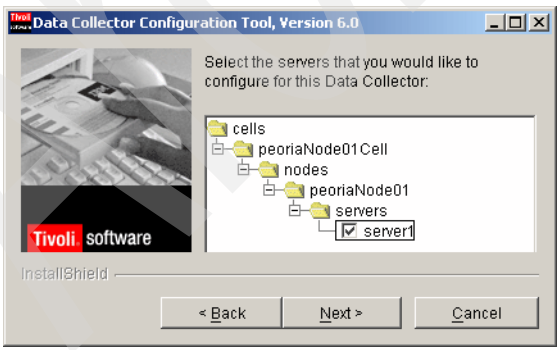


Figure 3-52 Selecting the application server instance

5. When the configuration is complete, a configuration summary dialog opens, as shown in Figure 3-53. Click **Finish** to exit the data collector configuration wizard.

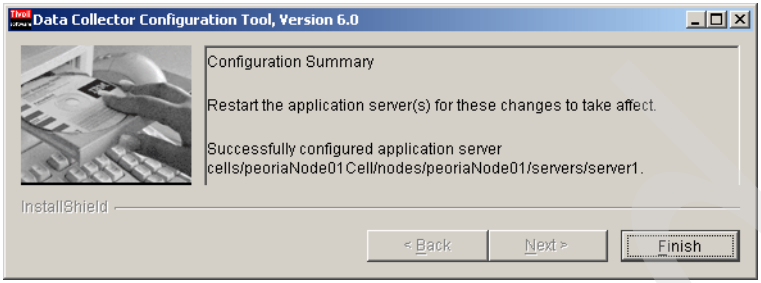


Figure 3-53 Configuration completed

6. Restart the targeted application server instance. This activates its newly enhanced IBM Tivoli Composite Application Manager for WebSphere monitoring and data collection capabilities.

You have successfully installed the IBM Tivoli Composite Application Manager for WebSphere data collector in a WebSphere Application Server environment. You have also configured a specific application server instance to allow monitoring and data collection.

3.3.3 Configuring additional application server instances

A computer system with IBM WebSphere Application Server installed will almost always have more than one application server instance running on it. Application server instances may be referred to as application servers or Java Virtual Machines (JVMs) periodically. With the IBM Tivoli Composite Application Manager for WebSphere data collector installed once for the WebSphere Application Server environment itself, just run the data connector configuration tool multiple times to configure multiple, distinct application server instances.

We used the data collector configuration tool earlier to enable monitoring and data collection for the server1 application server instance. To configure another application server instance in the same WebSphere Application Server system, perform the following steps:

1. Determine the SOAP Connector port for the application server instance that has to be configured for data collection and monitoring.

   The SOAP Connector port number is contained in a file called *serverindex.xml* located in the application server instance's own directory

structure. For WebSphere Application Server Version 6.*x,* this file is located under the following path:

WAS_home\appserver_instance\profiles\profile_name\config\cells\<*cell_name*>\nodes\<*node_name*>\serverindex.xml

Performing a search for this file in your environment may be the easiest way to locate it. The port number is located in the SOAP_CONNECTOR_ADDRESS property.

Note: For WebSphere Application Server V5, the server.xml file contains the SOAP Connector port information. The following path contains this file: WAS_home\config\cells\<*cell_name*>\nodes\<*node_name*>\servers\<*server_name*>\server.xml.

2. Launch the data collector configuration wizard. You can manually launch the data collector configuration wizard using the appropriate script in the $DC_HOME/config_dc directory.
3. After the data collector configuration tool starts and presents the configuration tool welcome page, begin at step 1 on page 58 and complete each step. Enter the new SOAP Connector information for the application server instance that is being configured.

To summarize, WebSphere Application Server installations can host multiple application server instances with each instance capable of hosting its own Java 2 Platform, Enterprise Edition (J2EE) applications and components. Each instance is a separate JVM that can be monitored using IBM Tivoli Composite Application Manager for WebSphere. Although the data collector installation occurred only once for the WebSphere Application Server, you must configure each application server instance separately using its own assigned SOAP connection information.

3.4 Defining data collectors, server groups, and users

After you install and configure the data collector and restart the application server, the data collector automatically connects to the remote managing server. There are some initial setup tasks that you have to perform from the IBM Tivoli Composite Application Manager for WebSphere managing server system console:

- 3.4.1, “Starting the IBM Tivoli Composite Application Manager for WebSphere console” on page 62
- 3.4.2, “Activating data collection” on page 62

- 3.4.3, “Defining server groups” on page 69
- 3.4.4, “Defining operators” on page 73

3.4.1 Starting the IBM Tivoli Composite Application Manager for WebSphere console

Perform these initial setup tasks working from the system on which you have installed the managing server components that comprise the IBM Tivoli Composite Application Manager for WebSphere solution.

1. Start the managing server if it is not already started for this Windows installation. Using the Windows Graphical User Interface (GUI), select **Start** → **All Programs** → **ITCAMfWS** → **am-start.sh**. Alternatively, from the command line, you can start the Micorsoft Services for UNIX environment and issue the **`am-start.sh`** command. The command that you issue using the Windows GUI does the same function. You can see this by right-clicking the am-start.sh item in the Windows GUI menu path shown in the next section.
2. Double-click the desktop icon **ITCAMfWAS** to start the browser-based managing server console. Alternatively, enter the following URL in your browser to start this console:

   `http://`*`MShost`*`:9080/am/home`
3. The welcome and login screen for IBM Tivoli Composite Application Manager for WebSphere opens. Enter the user and password established when you installed the managing server. See “Creating the installation user” on page 27. The user we created in the lab as the administrator for this installation is itcamadm.

3.4.2 Activating data collection

You have to complete several administrative tasks before the data collectors can begin sending application server monitoring information to the managing server for display. These are:

- Assigning a configuration profile for the data collector. See “Data collector configuration” on page 63.
- Defining the monitoring level and sampling rate for the data collector. See “Monitoring level and sampling rate” on page 66.

Data collector configuration

A configuration is a profile that assigns a specific monitoring definition for a data collector.

1. From the menu bar at the top of the browser console, select **ADMINISTRATION → Server Management → Data Collector Configuration**.
2. Click the **Unconfigured Data Collectors** link to view the newly installed data collector. (If the data collector you just installed is not listed, see 4.3, “IBM Tivoli Composite Application Manager for WebSphere problem determination” on page 123.)
3. Figure 3-54 shows the unconfigured data collector window. Configuring the data collector means that we assign the data collector to one of the three default profiles in the managing server. Select the **J2EE Default** profile using the drop-down menu. Select the appropriate application server where the data collector is installed. Click **Apply**.

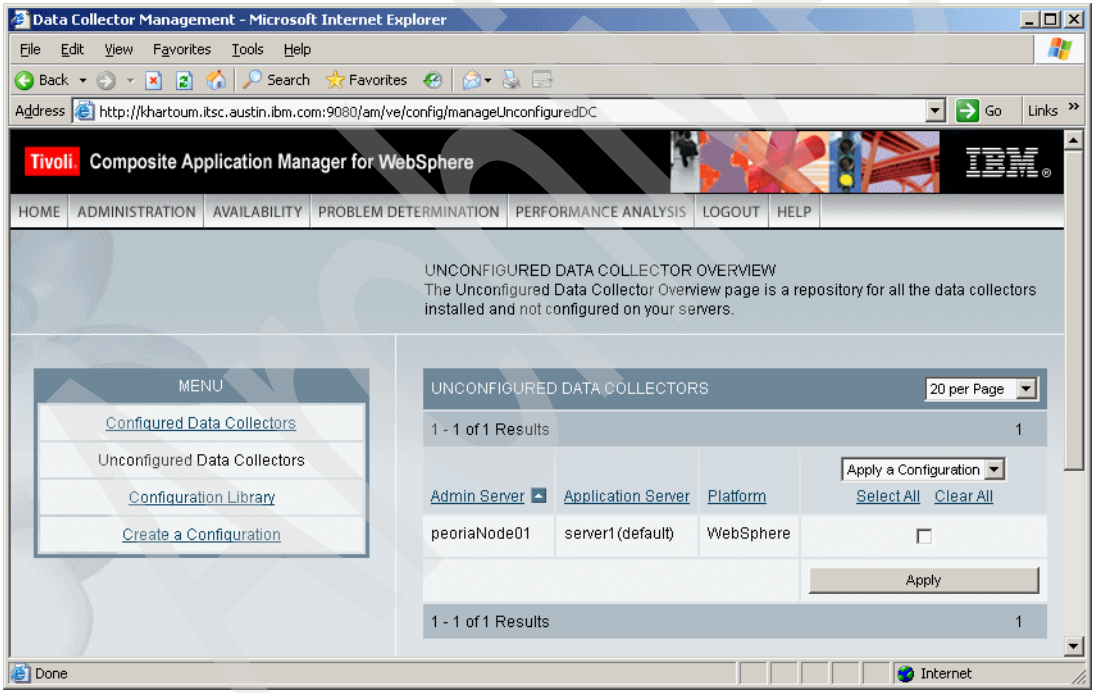


Figure 3-54 Unconfigured data collector

4. The Configured Data Collectors page opens with the new data collector added to the list of already configured data collectors. See Figure 3-55. Click the **Configuration Library** link in the menu box to the left.

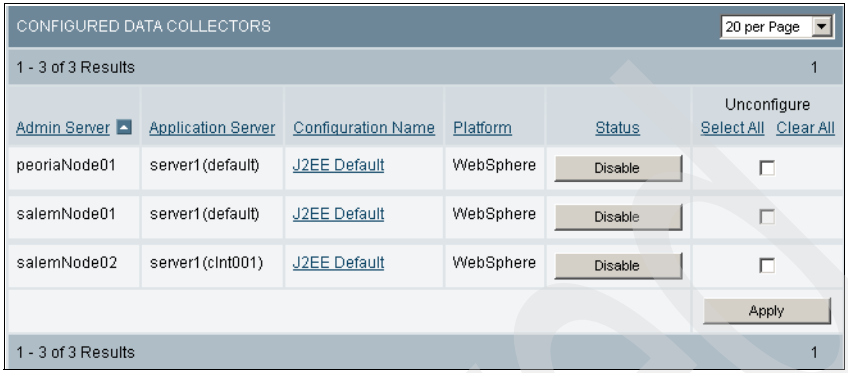
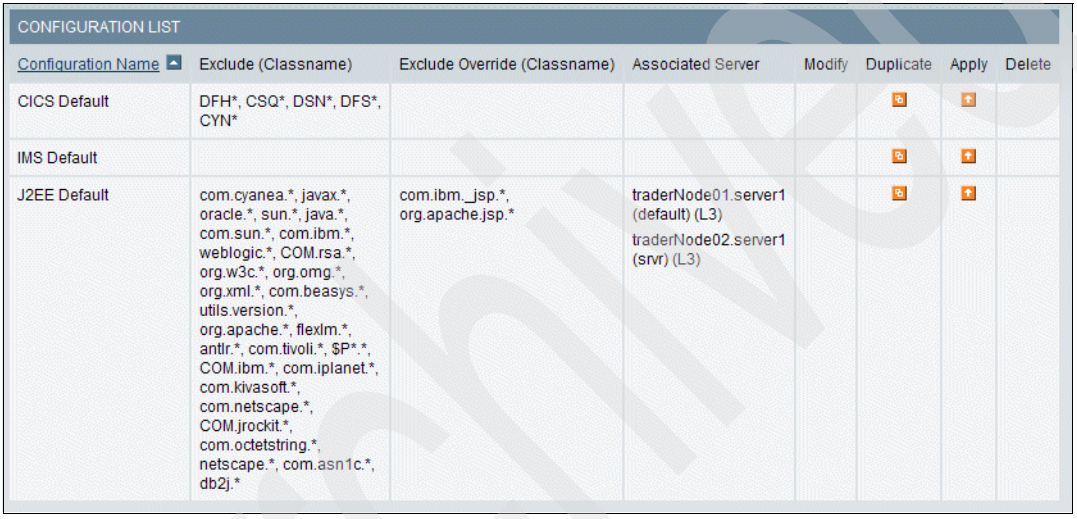


Figure 3-55 Data collector configuration

This shows the default profiles for Customer Information Control System (CICS), Information Management System (IMS), and J2EE, as shown in Figure 3-56. These default profiles list the transaction names or class names to be excluded when processing information in the data collector. These configurations should be adequate for most installations. If necessary, you can perform further customization by updating the configuration files.

Note: The association between a data collector instance and a profile is eternal, that is, if we change the profile, all its historic monitoring data will be lost.



CONFIGURATION LIST

Configuration Name	Exclude (Classname)	Exclude Override (Classname)	Associated Server	Modify	Duplicate	Apply	Delete
CICS Default	DFH*, CSQ*, DSN*, DFS*, CYN*						
IMS Default							
J2EE Default	com.cyanea.*, javax.*, oracle.*, sun.*, java.*, com.sun.*, com.ibm.*, weblogic.*, COM.rsa.*, org.w3c.*, org.omg.*, org.xml.*, com.beasys.*, utils.version.*, org.apache.*, flexlm.*, antlr.*, com.tivoli.*, $P*.*, COM.ibm.*, com.iplanet.*, com.kivasoft.*, com.netscape.*, COM.jrockit.*, com.octetstring.*, netscape.*, com.asn1c.*, db2j.*	com.ibm._jsp.*, org.apache.jsp.*	traderNode01.server1 (default) (L3) traderNode02.server1 (srvr) (L3)				

Figure 3-56 Default monitoring profile

You have configured the data collector with the default J2EE profile.

Monitoring level and sampling rate

Now that you have configured the data collector, you can view or change the monitoring level and sampling rates that the IBM Tivoli Composite Application Manager for WebSphere managing server uses.

The following list describes the different monitoring levels available:

- Level 1 (L1): Production Mode

  This monitoring level provides availability management, system resources and basic request-level data. It least affects the central processing unit (CPU) overhead per transaction and is appropriate for servers that are not malfunctioning.

- Level 2 (L2): Problem Determination Mode

  This monitoring level provides production-level monitoring plus advanced request data, including external component and CPU information, as well as additional monitoring fields and functions. In this mode, you can view component traces. These are traces that show J2EE request-related events that are made to external services. You must use this level when you suspect a problem, or when you have to capture data about external events but do not need all the method-level data.

- Level 3 (L3): Tracing Mode

  This is the most powerful monitoring level because only this level utilizes all reporting elements available. For example, in L3 the server activity display shows additional data for the following columns: Accumulated CPU, Last Known Class Name, Last Known Method, and Last Known action. In addition, on the Request Detail page, the Method Trace with Structured Query Language (SQL) statements are also available. L3 has inherently higher overhead than the other monitoring levels. Therefore, you must use this level for servers that have been selected for diagnostics and detailed workload characterization. It is advisable to schedule L3 for a very short period of time when investigating issues on a production application server.

Note: You must set the monitoring level to either Problem Determination Mode (L2) or Tracing Mode (L3) to retrieve information about lock contentions and lock acquisitions.

To view or change monitoring levels, perform the following steps:

1. To view the default monitoring level, polling frequency, as well as the sampling rates specified for each monitoring level option, from the menu bar at the top of the console, select **ADMINISTRATION** → **Managing Server** → **System Properties**. Figure 3-57 shows an example.

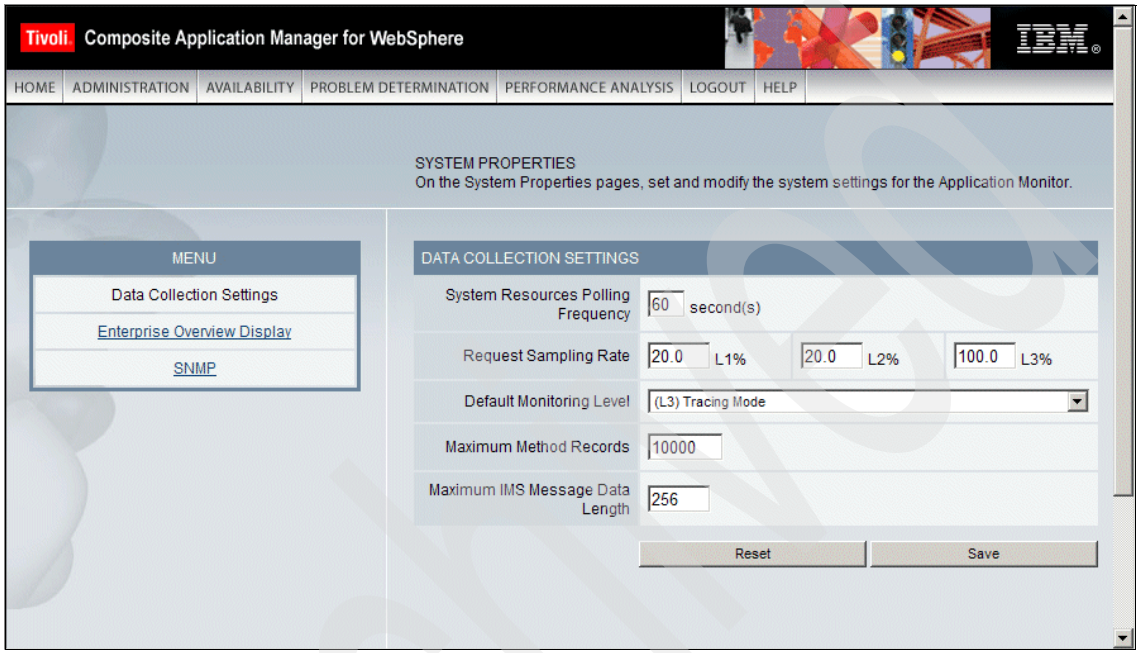


Figure 3-57 Default monitoring level

Depending on a composite application's complexity and transaction level, it may be necessary to adjust the sampling rates. The default sampling rate of 2% assumes a very busy application server. We want to specify an appropriately high sampling rate to capture statistically significant information about the transactions, and yet not overload the database with too much similar, statistically irrelevant, information.

Note: This is a rule of thumb example for calculating an initial sampling rate. We require at least five transactions per workload transaction mix per minute. If we have five different invoked transactions throughout the day, we will want to collect 25 transactions per minute (five times five equals 25). If the application server has approximately 200 total hits per minute, we set the sampling rate to 13%. In this example, 25 divided by 200 is approximately 13%.

2. Calculate an appropriate estimated sampling rate for the remote WebSphere Application Server to be monitored. See the previous Note.

   In the lab environment where we developed this guide, we used a Trader application to generate adjustable transaction loads. See Appendix B, “Trader application usage” on page 139 to obtain the additional material. See Appendix A, “Tivoli Enterprise Monitoring Agent component” on page 127 for information about how to use this Trader application. In the development environment for this guide, we used monitoring level L3 with 100% sampling rate.

   Important: When you monitor a production application server, do *not* use L3 for more than a few minutes because this may adversely affect performance of the application server. You can also use scheduling to dynamically collect L3 information periodically.

3. Select **ADMINISTRATION** → **Monitoring On Demand®** from the menu to view each individual application server nodes monitoring level.
4. A dialog similar to that shown in Figure 3-58 opens. Click the orange arrow icon for the appropriate node.

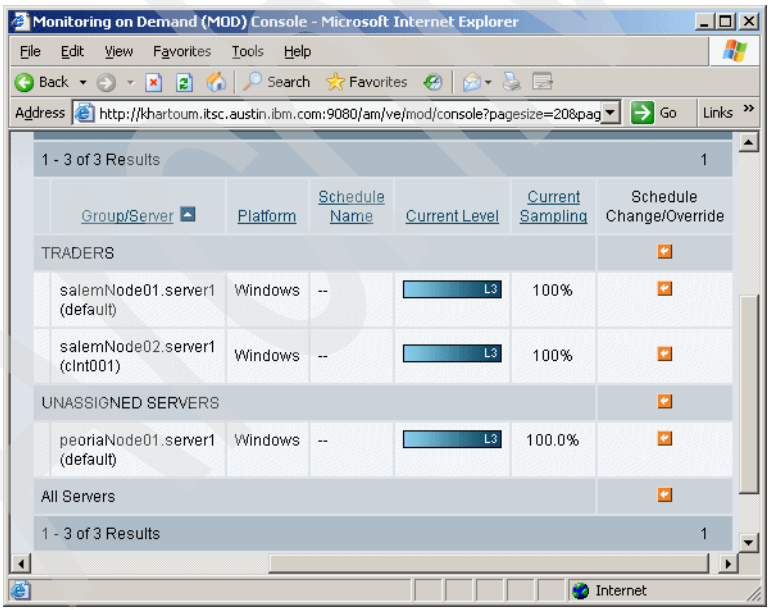


Figure 3-58 Monitoring On Demand

5. You can change these values, or create or change a scheduled time for collecting information, as shown in Figure 3-59. Click **OK** to close this window.

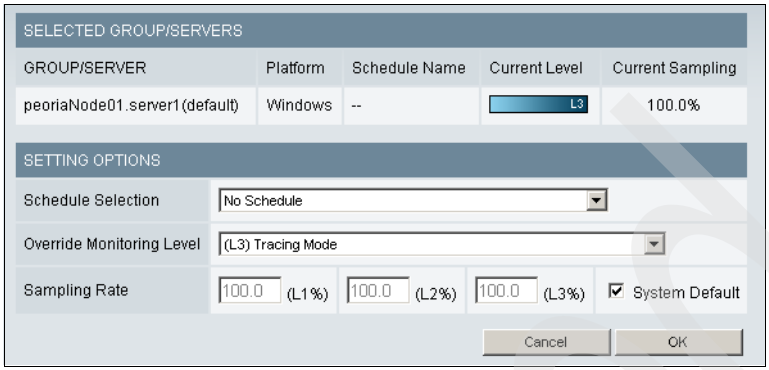


Figure 3-59 Changing the monitoring level

You have reviewed the newly installed and configured data collector's monitoring and sampling rates and possibly adjusted them for your environment.

3.4.3 Defining server groups

IBM Tivoli Composite Application Manager for WebSphere supports the grouping and aggregation of information obtained from instrumented application servers. Server groups provide this functionality. These groupings make the viewing and reporting of aggregated information, as well as assigning access to IBM Tivoli Composite Application Manager for WebSphere information easy and effective. Some basic examples of helpful server groupings are provided in the following list:

- The enterprise overview allows users to understand the behavior of their application servers at a glance, by graphically displaying throughput and response time of server groups. In this case, the grouping of servers helps to provide a high-level overview for a complex multi-server environment.
- Using server groups for report generation allows us to run reports against a combined group of servers instead of running separate reports against each individual application server.
- Another option that server grouping gives us is a more granular authorization capability because we can grant access to servers at the group level.
- Server grouping is also helpful if we want to change the monitoring level of several servers at the same time.

Further examples of useful server grouping strategies include grouping by cell or node, operating system, physical location, or by responsibility. For instance, we

can group application servers according to the teams of people responsible for managing them and then grant access through these different server groupings to the appropriate team members.

Initially all the instrumented and configured application servers reside in a generic group called *unassigned servers*. We assign them to server groups that we create.

1. Click the menu item **HOME** and click the **Group** tab. A list of the unassigned servers is shown similar to Figure 3-60.

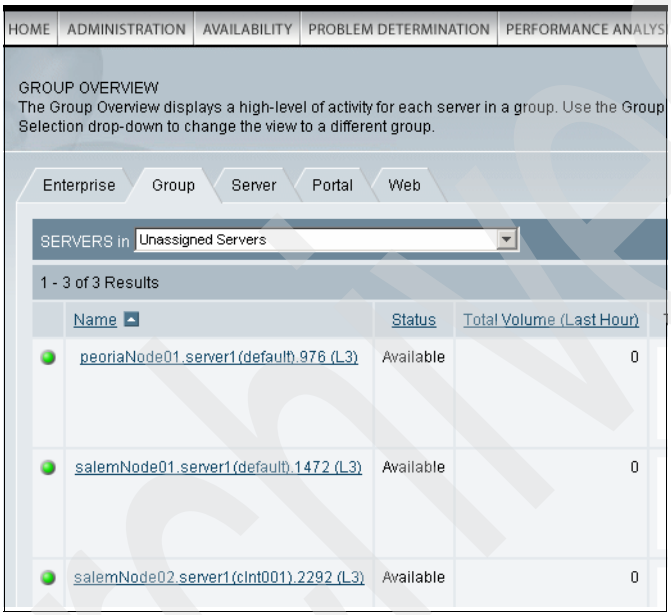


Figure 3-60 Unassigned servers

2. From the menu, select **ADMINISTRATION** → **Server Management** → **Server Groups.**
3. Click the link **Create Groups** to create a new group.
4. Enter `All` for the group name. Scroll down to the Group Members section and select, then add, all the available servers to the Servers in Group window by clicking the **Add** button. Click **Save** to create this server group.

To create helpful server groupings consider:

- The number of servers to monitor: Does the grouping make sense.
- The organizational and processes that are used for monitoring the environment: Who is responsible for what servers and applications, who needs to have access to what server information.
- How reporting is used: What reports are necessary, what type, and how many reports are manual and how many are scheduled.

In the lab environment, we created three server groups called *All*, *Trader*, and *WPS on WAS*, as shown in Figure 3-61.

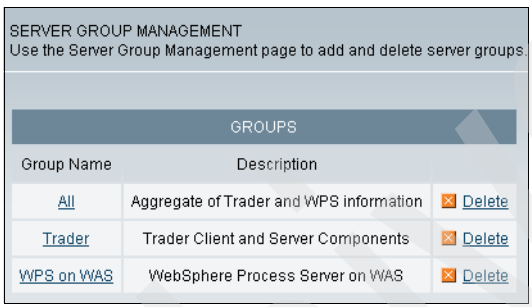


Figure 3-61 Server groups

In a shared server environment, depending on the requirements for accuracy in the enterprise overview perspective, grouping servers from an application point of view can be an effective option.

In Figure 3-62, the group Trader_CICS_distr is an example of a server group that is grouped by application. IBM Tivoli Composite Application Manager for WebSphere allows a server to be assigned as a member of several groups. This allows the creation of server groups from an application point of view, which includes transactions that share servers with other applications. In this example, there are several WebSphere Application Servers that access the same CICS and IMS systems. These application servers are members of several groups. For instance, the IMS server is part of the group *IMS distributed* and also the group *IMS zOS*.

Note: Using *shared* servers in server groups may cause misinterpretations in the enterprise overview because transactions are not distinguished at component level. For example, every server group that contains the same shared CICS server will see all transactions from this CICS server. The number of requests and response times aggregated in the Enterprise View tab will include transactions triggered by servers that are not part of the group.

Figure 3-62 shows a grouping of servers from an application perspective. Note the performance patterns for groups IMS_distributed and Trader_CICS_distributed: There have not been any IMS activities during this period of time. The explanation for this possibly misleading information is that both CICS and IMS transactions run on the same WebSphere Application Server.

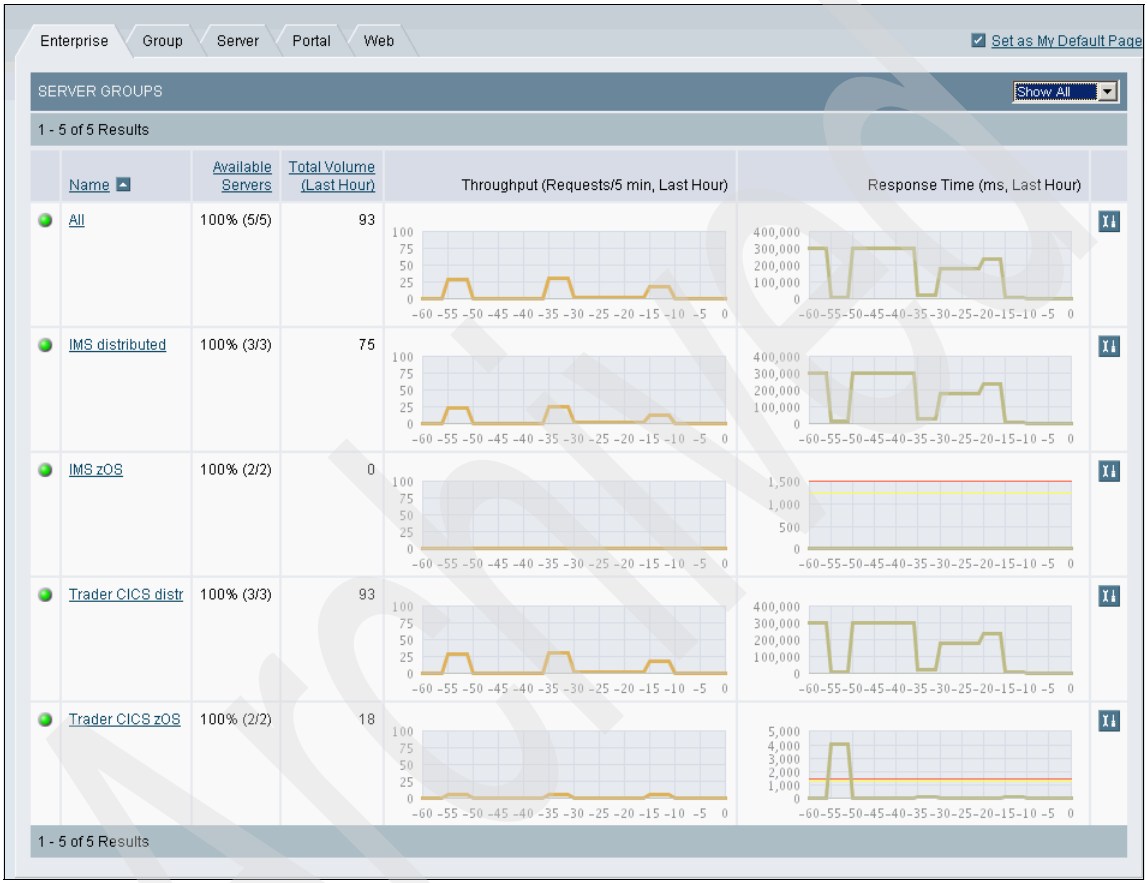


Figure 3-62 Server group by application

You have now defined server groups and assigned data collector information sources to them in a meaningful and useful way. This server grouping capability enables effective viewing, analysis, and reporting of monitoring information in unique customer environments.

3.4.4 Defining operators

So far we have only defined the administrator user called itcamadm during the installation process. See “Creating the installation user” on page 27. In a multi-user IBM Tivoli Composite Application Manager for WebSphere environment, it is advisable to create multiple separate users. To define a new user for IBM Tivoli Composite Application Manager for WebSphere, perform the following steps:

1. Create a user ID on the operating system where the IBM Tivoli Composite Application Manager for WebSphere managing server is installed:
   - Use computer management on Windows system
   - Use **`smit`** for AIX system
   - Use **`adduser`** command on Linux system
   - Alternatively, use the appropriate tools for your server platform

   Our managing server is installed on Windows. Therefore, we define an additional user, oper1 using the computer manager utility and assigning a password that never expires. See Figure 3-63.

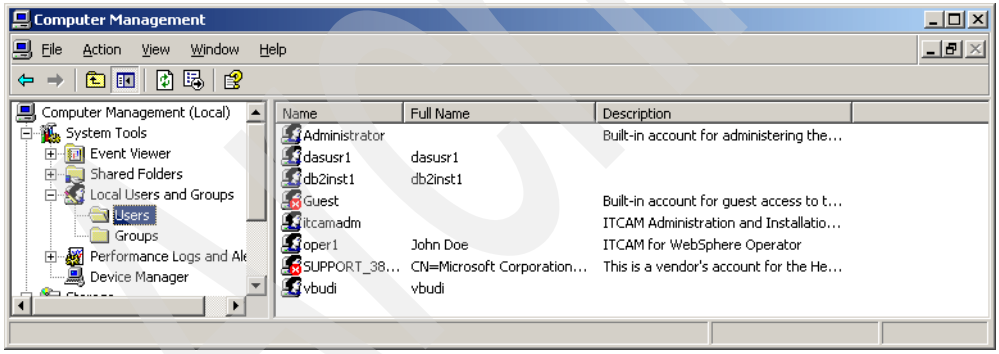


Figure 3-63 Operating system user create

2. Review the different user roles available. This is performed from the Web console. From the menu, select **ADMINISTRATION** → **Account Management** → **Role Configuration** to review the different default roles available. Note that an operator cannot create new users or server groups and so on. These roles can be modified for a customer’s environment, if necessary.

3. Define an IBM Tivoli Composite Application Manager for WebSphere user. Select **ADMINISTRATION** → **Account Management** → **User Profiles**.
4. Enter the required information in the fields marked with an asterisk. Assign one or more defined server groups to this user using the Group Access portion of the window in the bottom section, as shown in Figure 3-64. In our example, this operator has only been assigned access to the Traders server group with the role of Operator.

   Click **Save** to create this new operator. You have created a new operator with appropriate access privileges.

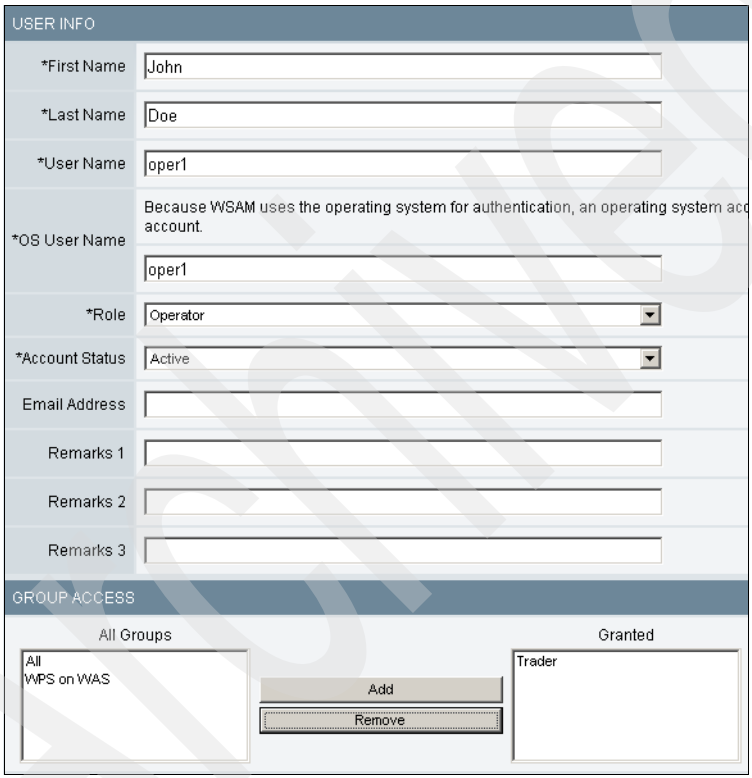


Figure 3-64 Creating the IBM Tivoli Composite Application Manager operator

Note: The operator name in IBM Tivoli Composite Application Manager for WebSphere and the operating system user name do not have to be the same. You can also define many IBM Tivoli Composite Application Manager for WebSphere operators with unique passwords using the same operating system user.

This may be confusing for administrative purposes. We recommend that you use a unique operating system user for each IBM Tivoli Composite Application Manager for WebSphere operator. Furthermore, the IBM Tivoli Composite Application Manager for WebSphere operator must have the same password as the operating system login to avoid any confusion.

It is important to consider how IBM Tivoli Composite Application Manager for WebSphere will be used in your environment and who will use it. The solution produces reports that can be relevant to operational managers as well as service or application development managers. Identifying the roles and responsibilities is an important consideration because this affects the customization of IBM Tivoli Composite Application Manager for WebSphere, especially in features such as server groups.

4

Using IBM Tivoli Composite Application Manager for WebSphere

In this chapter, we describe some of the methods to use IBM Tivoli Composite Application Manager for WebSphere. The sections are:

4.1 Monitoring WebSphere Application Server

This section discusses some of the methods to use IBM Tivoli Composite Application Manager for WebSphere when monitoring the IBM WebSphere Application Server. We discuss the following:

- 4.1.1, “Working with the Web console” on page 78
- 4.1.2, “Use case scenarios” on page 96

Tip: It is useful to have a Java 2 Platform, Enterprise Edition (J2EE) application running and also have some historic data collected from one or more instrumented application servers to effectively work with the information discussed in this section. IBM Tivoli Composite Application Manager for WebSphere is already installed, see “Installing IBM Tivoli Composite Application Manager for WebSphere” on page 25. If you do not have an application installed on your data collector instrumented application server, a simple application called *Trader* is available. You can install and run it prior to working through these sections.

See Appendix C, “Additional material” on page 143 for instructions about how to download and install the Trader application.

See Appendix B, “Trader application usage” on page 139 for instructions about how to use the Trader application.

4.1.1 Working with the Web console

The Web console for IBM Tivoli Composite Application Manager for WebSphere is also called the *visualization engine*. There are different ways to use this Web console. Two typical usage scenarios for IBM Tivoli Composite Application Manager for WebSphere are: To monitor currently running application servers from a first line high-level operations perspective to detect whether a problem is developing or has occurred. The second usage scenario is to perform a detailed analysis of the application server performance after or preferably before a problem has started to impact the composite application.

This section discusses these two Web console usage scenarios in the following sections:

- “Monitoring application servers” on page 79
- “Analyzing application server statistics” on page 83

It is advisable for operations personnel to leverage both the monitoring and analysis capabilities of IBM Tivoli Composite Application Manager for WebSphere. Regular high-level monitoring, as well as in-depth analysis, keeps

the systems up and running, detects potential developing problems, and provides tools and reports. The second line operations, deployment, and development teams can use these tools and reports to help them quickly determine root cause, fix, and improve the monitored applications.

Monitoring application servers

When operations personnel monitor their production WebSphere Application Servers, their first objective is to observe and monitor, from a high level, the overall performance and health of their systems. They do not yet have to focus on specific transactions or components that comprise these composite applications from this high-level monitoring perspective.

Typical application server monitoring for operators has several levels of detail available. Operator assignments and server group configurations affect the display of the available information. The typical monitoring levels are:

- Enterprise monitoring

  The operator sees the aggregate performance of all application servers and their associated applications.

- Group monitoring

  The operator sees an overview of the grouped application servers in an assigned server group including their aggregate transaction volume and average response time.

- Server monitoring

  The operator sees an individual application server's performance with detailed performance indicators for that application server.

The Web console for IBM Tivoli Composite Application Manager for WebSphere provides three tabs called Enterprise, Group, and Server that allow an operator to quickly switch between these monitoring levels. Click the **HOME** menu item for the Web console view with these three tabs. An operator can customize this initial page to provide a specific initial monitoring level view by clicking the Set as My Default Page link.

Enterprise monitoring

The enterprise monitoring view shows an overview of all server groups that are assigned to an operator. This view aggregates response time average and total transaction rates with warnings and critical thresholds for each metric shown as red and yellow lines. Figure 4-1 shows an example of an enterprise monitoring page view.

Figure 4-1 Enterprise monitoring

Sometimes the enterprise view is not realistic enough. This is because a view of all application servers in an enterprise potentially combines different, unrelated applications and their performance profiles into one view.

Server group monitoring

The server group monitoring view may be more realistic for an operator. This view shows a server group where the servers are typically grouped together from a common application perspective. The server group monitoring view looks similar to the enterprise monitoring view. This page also shows whether the data collector in each application server is online. Figure 4-2 shows an example of server group monitoring.

Figure 4-2 Server group monitoring

This view can be useful for detecting an application's issues and to begin narrowing down possible problems. However, a display page with more than 10 server entries may be too crowded and can impact effective monitoring by an operator.

Application server monitoring

The detailed application server monitoring view may be necessary for a specific, critical application server. Figure 4-3 shows an example of an application server view that provides additional server-specific performance information that is not available from the enterprise or group monitoring views.

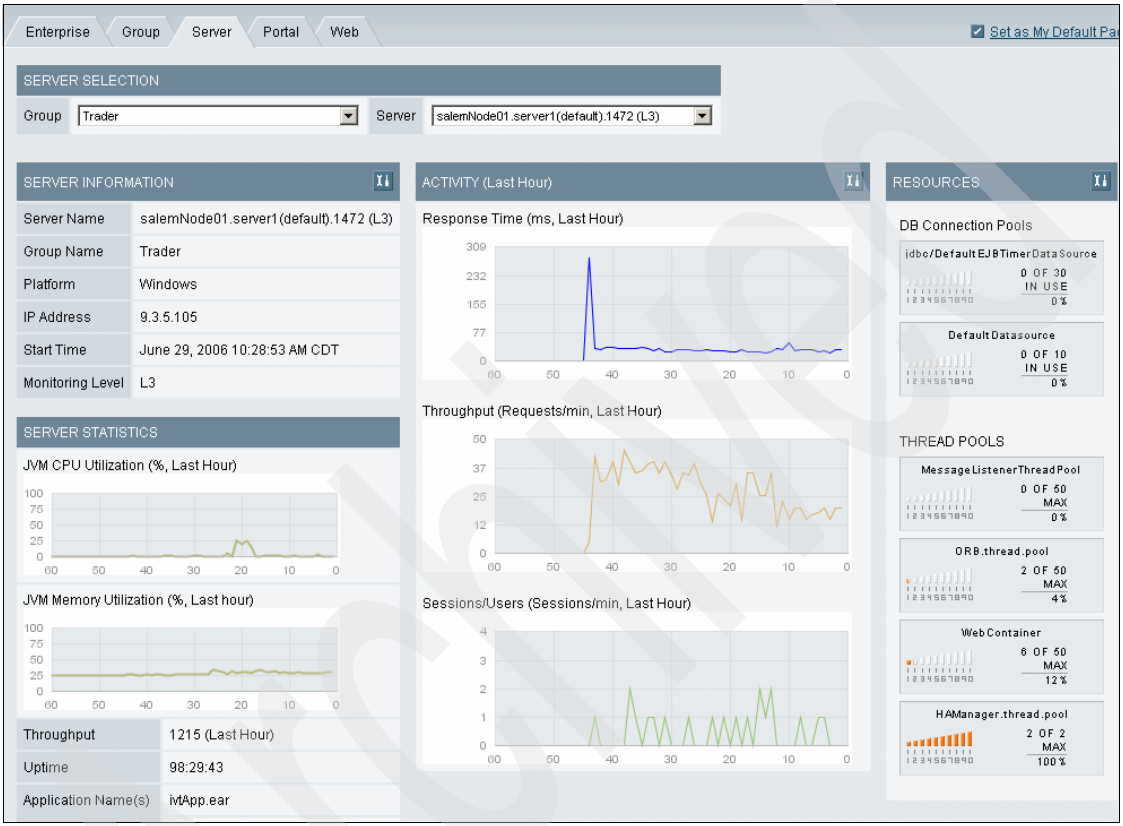


Figure 4-3 Application server performance

To drill down further and access more detailed information subviews, click any of the three tool icons ( ) available for each of the Server Information, Activity, and Resources area windows.

Click the tool icon for Server Information to enable access to the following detailed information:

- Runtime environment check
- Runtime environment comparison
- Server statistics overview

Click the tool icon for Activity for the following detailed information:

- Server statistics overview
- Server activity display
- Memory analysis
- Heap analysis
- Memory leak

Click the tool icon for Resources to access the following detailed information:

- System resources
- System resource comparison
- Server statistics overview
- Java Virtual Machine (JVM) thread display

The detailed application server monitoring view and its associated informational subviews are most useful for second-line operations members to diagnose specific application issues and narrow down detected problems to their root causes. To summarize, with these tab-based enterprise, group, and server views, an operator can easily switch between and monitor application servers at the level that is appropriate for their objectives. These views enable an operator to efficiently detect and acquire the necessary information to proactively correct and thus prevent an impending problem.

Note: These monitoring view summary windows automatically refresh every 60 seconds. They do not time out the user. This may be an issue in some environments that are particularly security conscious. Click **LOGOUT** from the main menu to end the session proactively.

Analyzing application server statistics

In this second usage scenario section, the focus is on analyzing application statistics. The analysis of previous transaction information is used to uncover detailed performance trends or other issues. Detailed analysis is useful for detecting developing problems as early as possible. You can use this information to make tuning improvements for certain transactions down to the method level, and providing development with insight into how they may redesign the application, as well as providing capacity planning information among other things for applications deployed and running on one or more WebSphere application servers.

IBM Tivoli Composite Application Manager for WebSphere provides a rich set of reports that can either be queried online using the Web console or saved and scheduled to run during off hours. The reports are also useful for identifying application bottlenecks quickly and also to get a better understanding of the

application flow. Among the many reporting capabilities available, this basic guide focuses only on some of the principal reports. They are:

- “Top reports” on page 84
- “Request/transaction reports” on page 87
- “Server reports” on page 92

Top reports

Top reports show potential bottlenecks in the application system. These reports can be used to identify worst performers and also to begin to investigate them further by drilling into the information reported. The top reports delivered with Table 4-1 lists the transactions and the required monitor levels for IBM Tivoli Composite Application Manager for WebSphere.

Table 4-1 Top report transactions

Transaction	Description	Required monitoring level
Top Requests Used	Most called requests with the number of calls	Level 1 (L1)
Top Methods Used	Most called methods including the number of calls	L3
Top Slowest Requests	Slowest requests and their average response time	L1
Top Slowest Methods	Slowest methods and their average response time	L3
Top central processing unit (CPU) Intensive Requests	Requests sorted by total CPU time	L1
Top CPU Intensive Methods	Methods sorted by total CPU time	L3
Top Structured Query Language (SQL) Intensive Requests	Requests sorted by number of SQL calls	L2
Top SQL Intensive Methods	Methods sorted by number of SQL calls	L3
Top Tables Used	The database tables called most often and the number of calls	L1
Top SQL Used	Top five SQL call types and number of calls	L2

To access the top reports wizard and create the report views, perform the following steps:

1. From the menu, select **PERFORMANCE ANALYSIS** → **Create Application Report** → **Top Reports**. A series of questions guides you through the process of selecting and creating these various report views.
2. The first question asks you whether this is to be a recurring report, as shown in Figure 4-4. The default selection is No. Click **Next**.

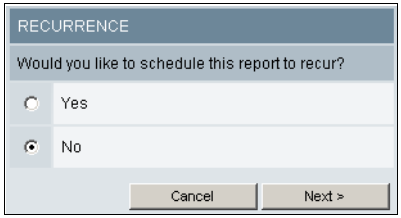


Figure 4-4 Top report: Recurrence

3. In the Server Selection page (Figure 4-5), the wizard provides the option to select the server group and either all or a specific server's information to be included in the top report. Click **Next**.

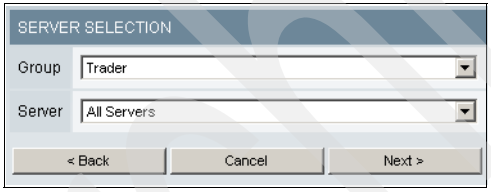


Figure 4-5 Top report: Server selection

4. The wizard provides a drop-down menu containing the list of available top reports. Select a report and specify the time range for the report, using advanced filtering capabilities, if necessary. The default report is Top Requests Used, as shown in Figure 4-6. Click **View Report** to generate the requested report.

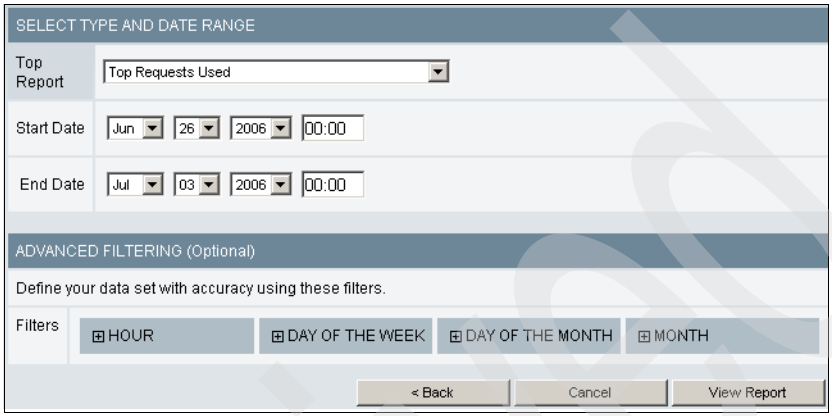


Figure 4-6 Top report: Selecting type and date range

Figure 4-7 presents a sample top requests report. After the report is generated, you can save the report or modify it using hyperlinks in the menu box to the left of the report. Additionally, the icons  , seen to the right in the Report Properties window, represent the options for e-mailing a link or a PDF of this report, as well as saving the report as a comma-separated values file.

Note: PDF generation is inactive until your site completes the iText integration instructions in “Appendix F” of the *IBM Tivoli Composite Application Manager for WebSphere Installation and Customization Guide*, GC32-9506.

REPORT PROPERTIES

Report Name	Untitled
Report Type	Top Requests Used Analysis
Report Period	Jun 26, 2006 12:00 AM to Jul 3, 2006 12:00 AM
Server Scope	Server: All Servers on Group: Trader

TOP REPORTS DATA TABLE — 20 per Page

1 - 20 of 97 Results — 1 2 3 4 5 Next > Last > >

RANK	REQUEST/TRANSACTION NAME	COUNT
1	/TraderDBServices/services/TraderDBServices	39512
2	/TraderClientWeb/BuySellServlet	11234
3	/TraderClientWeb/ListCompanyServlet	5728
4	/TraderClientWeb/GetQuotesServlet	5674
5	/TraderClientWeb/LogoutServlet	5559
6	/TraderClientMem/BuySellServlet	3805
7	/TraderClientMem/ListCompanyServlet	1938
8	/TraderClientMem/GetQuotesServlet	1936

Figure 4-7 Top report: Top requests used

Begin with top reports to identify the top players in your environment. In the development phase, these reports can be run at regular intervals to understand application behavior and pinpoint possible bottlenecks for the developers to focus on.

Request/transaction reports

Request/transaction analysis reports provide a high-level overview of the behavior of the application server. Several reports are available: Trend report, decomposition report, request report detail (including detail, summary and worst performers), and trace report. Each of these reports provides more specific data for understanding the application's performance at every level.

The following report profiles can be useful:

- Volume against aggregate hour: To see the transaction distribution in a day
- Response time against aggregate hour: To see the correlation of response time and time of day
- Volume against time series in an hour: For checking a problem daily

To access the request/transaction analysis report, perform the following steps:

1. From the menu bar, select **PERFORMANCE ANALYSIS** → **Create Application Report** → **Request/Transaction**. A series of questions guides you through the process of selecting and creating the possible report views.

2. The first question asks you whether this is to be a recurring report, as shown in Figure 4-8. The default selection is No. Click **Next**.

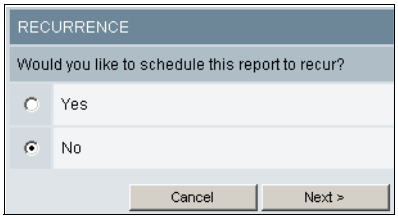


Figure 4-8 Request/transaction reports: Recurrence

3. In the Server Selection page (Figure 4-9), the wizard provides an opportunity to select the server group and either all or a specific server's information for the request/transaction report. Click **Next**.

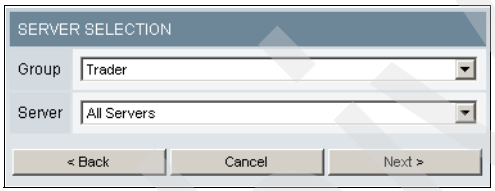


Figure 4-9 Request/transaction reports: Server selection

4. The Report Filtering Options page opens. Set the following options to filter the records returned in the report:
   - Metric: The item you want to measure. The options are: Throughput per Second, Throughput per Minute, Throughput per Hour, Response Time, or CPU Time.
   - Request/Transaction Type: The options are: All, EJB, JavaServer™ Pages™ (JSP™), Servlet, Customer Information Control System (CICS), Open Transaction Manager Access (OTMA), Virtual Telecommunications Access Method (VTAM®), Basic Telecommunication Access Method (BTAM), Advanced Program to Program Communication (APPC), or Portal.
   - Request/Transaction Name: Unless you know exactly what the request string is, leave this field blank to return all requests. Only enter a value if you know the specific request/transaction name.

   Click **Next** to continue creating the report.

5. The Date Range Settings page opens. Set the parameters to restrict the data as appropriate. You can specify detailed time ranges from hours to months using the optional advanced filtering capabilities. The Graphing Option portion of the wizard window, at the bottom, allows you to set the graph's x-axis value. The options include:

   – Time Series in Hour
   – Time Series in Day
   – Time Series in Week
   – Time Series in Month
   – Aggregate Minute of the Hour
   – Aggregate Hour of the Day
   – Aggregate Day of the Week
   – Aggregate Month of the Year

   The default x-axis is set to Aggregate Hour of the Day, as shown in Figure 4-10. Click **View Report** to generate the requested report.

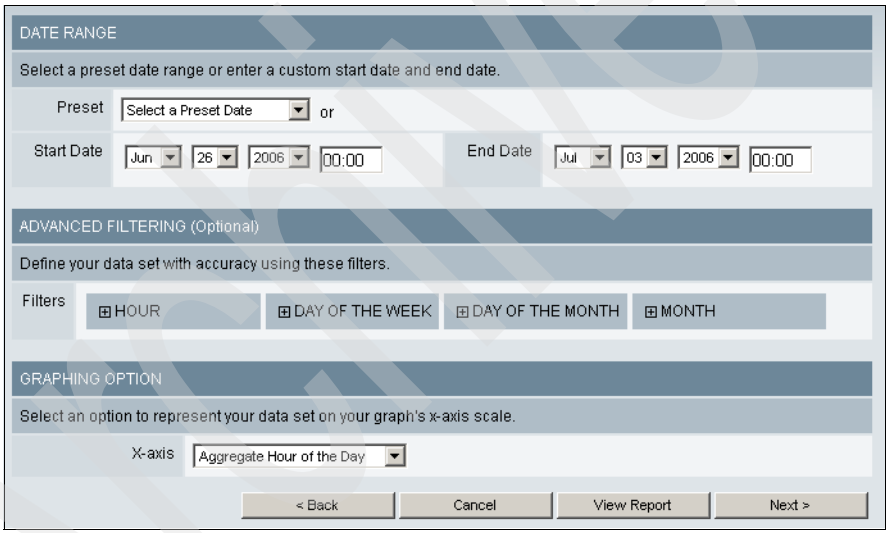


Figure 4-10 Request/transaction reports: Date range and x-axis

Figure 4-11 shows a sample trend report. After you generate the report, you can save or modify it using the menu box to the left of the report. Additionally, the icons  , to the right of the Report Properties window, represent the options for e-mailing a link or a PDF file, generating a PDF file of this report, as well as saving the report as a comma-separated values file.

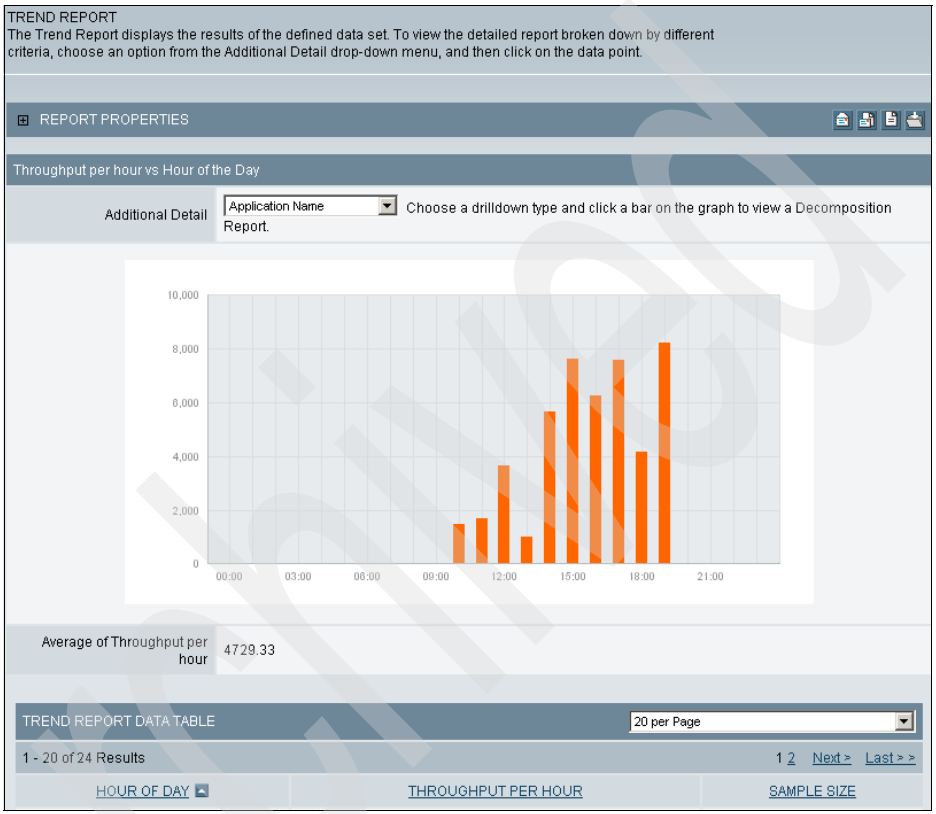


Figure 4-11 Request/transaction reports: Trend report

There are two ways to get additional detailed information from this trend report. This information is provided in the form of a decomposition report.

- Drill down by selecting either Application Name, Request/Transaction Type, or Server using the drop-down box next to the Additional Detail field at the top of the graph. Click a bar on the graph to view a decomposition report of that information.
- At the bottom of the report is a Trend Report Data Table field, which contains a column of time values that are provided in the report graph. Select one of the hyperlinked time elements from this table to generate a decomposition report.

Decomposition reports provide a breakdown of the trend report information. A sample decomposition report of transactions is shown in Figure 4-12.

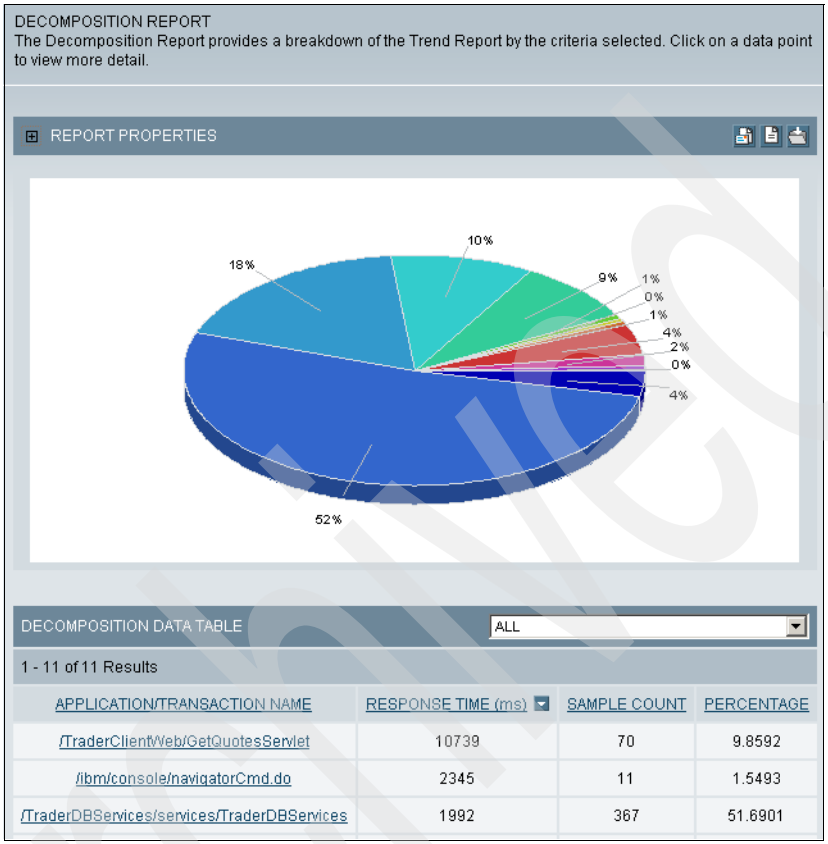
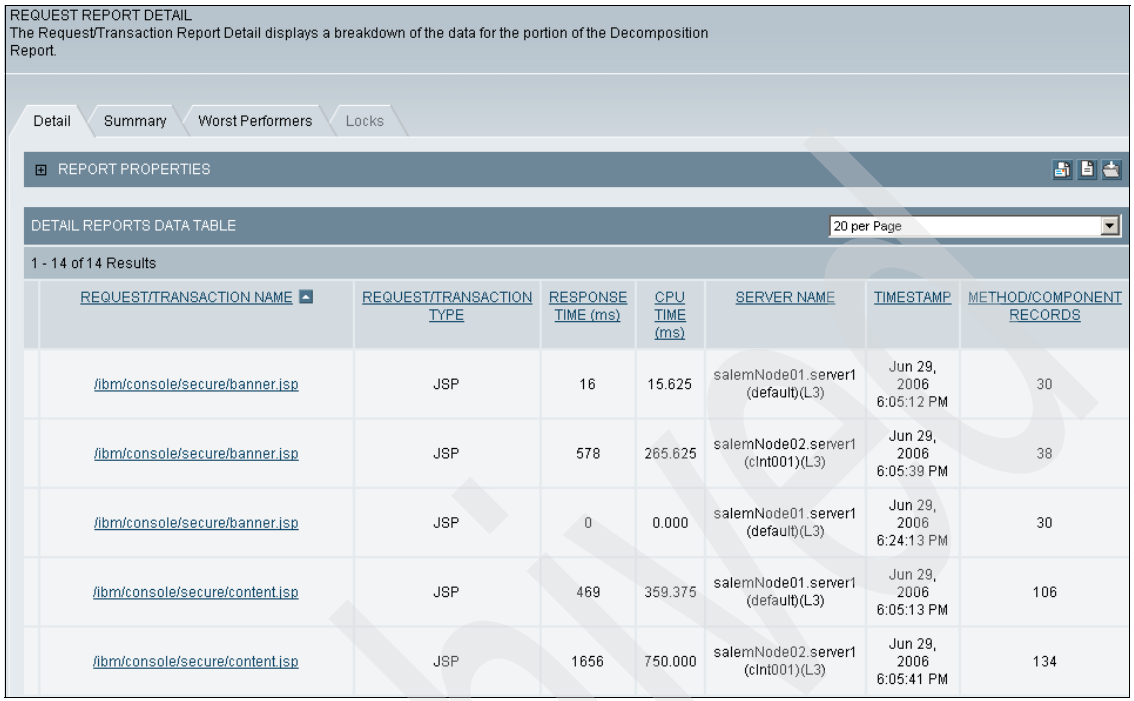


Figure 4-12 Decomposition report

When you click either a specific slice of this decomposition report graph or one of the hyperlinked Request/Transaction Type elements in its table view at the bottom, you can view transaction-specific details for further investigation and analysis. This Request Report Detail table report view displays a breakdown of the transaction data for the portion of the decomposition report that you selected. When you select the appropriate tab, it enables a detail, summary, or worst performers list of transactions by name.

Figure 4-13 shows an example of a request/transaction report detail view.


REQUEST REPORT DETAIL
The Request/Transaction Report Detail displays a breakdown of the data for the portion of the Decomposition Report.

Detail | Summary | Worst Performers | Locks

REPORT PROPERTIES

DETAIL REPORTS DATA TABLE — 20 per Page

1 - 14 of 14 Results

REQUEST/TRANSACTION NAME	REQUEST/TRANSACTION TYPE	RESPONSE TIME (ms)	CPU TIME (ms)	SERVER NAME	TIMESTAMP	METHOD/COMPONENT RECORDS
/ibm/console/secure/banner.jsp	JSP	16	15.625	salemNode01.server1 (default)(L3)	Jun 29, 2006 6:05:12 PM	30
/ibm/console/secure/banner.jsp	JSP	578	265.625	salemNode02.server1 (clnt001)(L3)	Jun 29, 2006 6:05:39 PM	38
/ibm/console/secure/banner.jsp	JSP	0	0.000	salemNode01.server1 (default)(L3)	Jun 29, 2006 6:24:13 PM	30
/ibm/console/secure/content.jsp	JSP	469	359.375	salemNode01.server1 (default)(L3)	Jun 29, 2006 6:05:13 PM	106
/ibm/console/secure/content.jsp	JSP	1656	750.000	salemNode02.server1 (clnt001)(L3)	Jun 29, 2006 6:05:41 PM	134

Figure 4-13 Request/transaction detail report

Through hyperlinks in these reports, you can drill further into the monitoring data collected by IBM Tivoli Composite Application Manager for WebSphere. It is possible to choose a specific transaction and profile it against time.

Server reports

Server reports provide a high-level overview of the behavior of the application server. These reports show resource usage of an application server allowing an IBM Tivoli Composite Application Manager for WebSphere user to anticipate bottleneck or capacity problems. The collection of server reports provided by IBM Tivoli Composite Application Manager for WebSphere are:

- System resource: The system resource analysis report provides information about the utilization of the memory, and database connection pools for the application servers. You can generate and analyze trend reports and decomposition reports after you define the system resource analysis report.
- Server availability: The server availability analysis report shows the percentage of the server availability. In a group situation, availability is defined as the total amount of time when one or more servers assigned to a server group available is divided by the total elapsed time.

- Capacity analysis: The capacity analysis report provides the necessary information to evaluate the capacity of your systems using supply and demand metrics.

This basic guide describes how to generate only the capacity analysis report. Perform the following steps:

1. From the menu bar, select **PERFORMANCE ANALYSIS** → **Create Server Reports** → **Capacity Analysis** to initiate the report wizard. A series of questions guides you through the process of selecting and then creating one of the possible report dimension views.
2. The first question asks which server group and specific server to report on, as shown in Figure 4-14. Click **Next**.

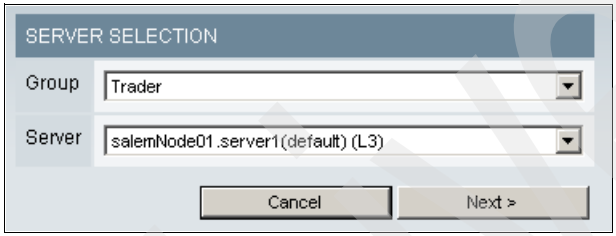


Figure 4-14 Capacity analysis report: Server selection

3. The wizard asks which performance metrics to assign for each axis. The options for the x-axis demand metric are:
   - Throughput per Minute
   - Users

   The options for the y-axis supply metric are:
   - System CPU (%)
   - JVM/Process CPU (%)
   - JVM/Process memory (megabyte (MB))
   - Thread Pool
   - Connection Pool
   - Response Time (millisecond (ms))

Select **Users** for the x-axis and **JVM/Process Memory (MB)** for the y-axis, as shown in Figure 4-15. Click **Next** to continue creating the report.

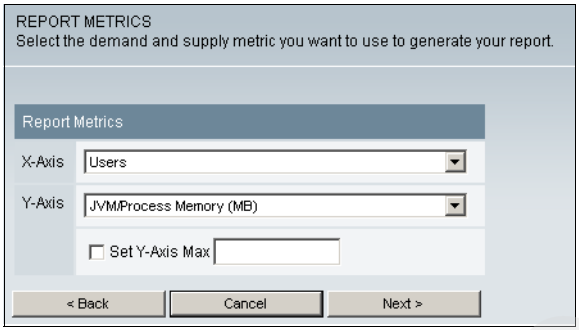


Figure 4-15 Capacity analysis report: Report metics

4. The Date Range Settings page opens. Set the parameters to restrict the data as appropriate. You can specify detailed time ranges from hours to months using the optional advanced filtering capabilities. See Figure 4-16. Click **View Report** to generate the requested report.

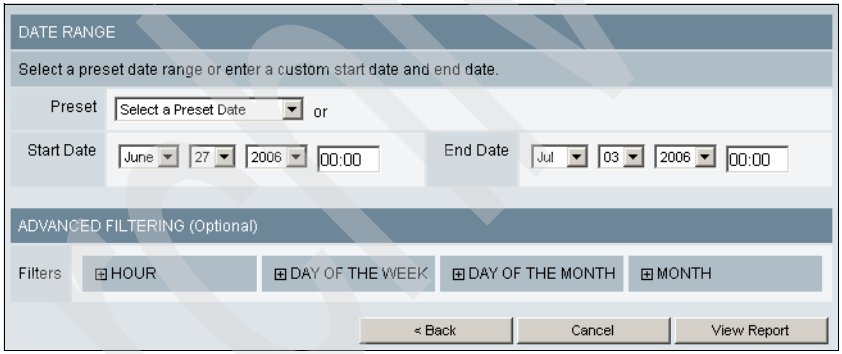


Figure 4-16 Capacity analysis report: Date range

An example of this type of report can be seen in Figure 4-17. The menu box to the left of the report enables you to modify the report further or save it by clicking the appropriate hyperlink. As with other reports, the Graph Data section allows you to sort the information. To do this, click a column heading such as Users and click the sort icon to specify the sort order.

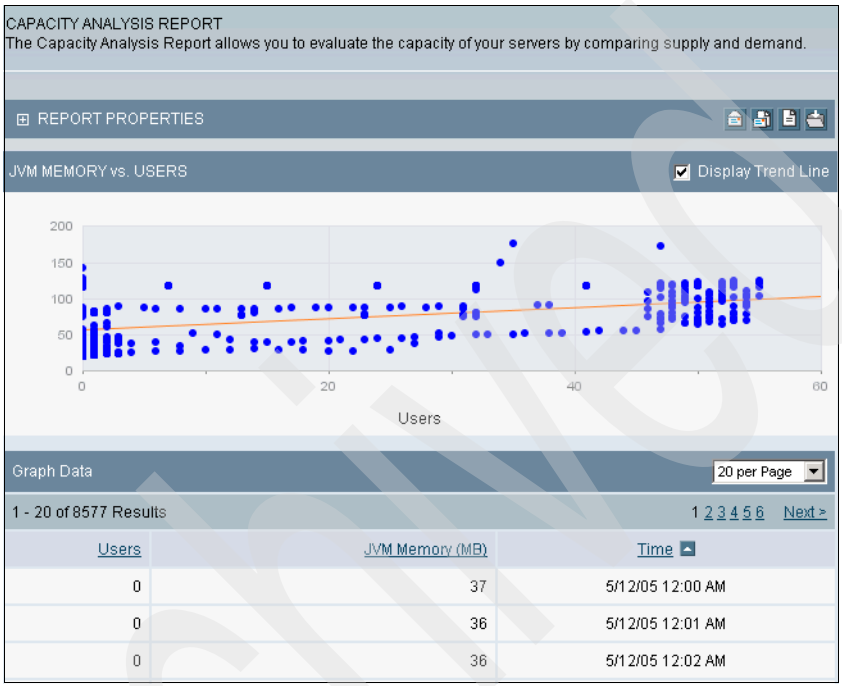


Figure 4-17 Capacity analysis report: JVM memory versus users

To summarize this section, you can leverage the Web console for IBM Tivoli Composite Application Manager for WebSphere in different ways. The previous subsections covered two typical usage scenarios. We covered how to access the three high-level operations monitoring views and their capabilities.

We also examined how to access a subset of the available detailed analysis reports. These reports are used for drilling into historic data to detect developing problems or to determine root causes of known problems. This detailed analysis approach provides the ability to proactively look for trends or indicators of developing and existing problems in applications, composite applications, the WebSphere application server they run on, as well as the hardware servers themselves.

4.1.2 Use case scenarios

This section presents some use case scenarios that represent possible ways in which you can use the IBM Tivoli Composite Application Manager for WebSphere capabilities to investigate some typical application server issues.

Note: In this section, not all of the utilities, reports, views, perspectives, or capabilities provided with IBM Tivoli Composite Application Manager for WebSphere are detailed with screen captures and steps. The reader is encouraged to further investigate any newly introduced capabilities mentioned in this section.

The following IBM Tivoli Composite Application Manager for WebSphere capabilities are presented with sample use case scenarios:

System Overview

Systems Overview allows you to evaluate the availability of your entire system by looking at recent performance trends.

- Verifying customer response time complaints

  In this scenario, customer service receives complaints that your company's Web sites are responding slowly. As one of the administrators of the servers, the inquiry comes to your attention. When you open the Enterprise Overview page, you immediately see that three of your production servers are no longer available. You also verify that the response time has degraded.

- Diagnosing an application problem

  Customers complain that they cannot place orders. As one of your company's administrators, you open the Enterprise Overview page and see that all the servers are up. You also find the group that appears to have the highest response time, and drill down to the server overview page where you see that a database connection pool is saturated.

Server Statistics Overview

Server Statistics Overview helps you to compare activity and related platform data across servers so that you can recognize problems.

- Investigating an unresponsive system

  Your first line of support receives calls that some parts of the system are not responding. The support team opens the Server Statistics Overview page and immediately sees that one server displays the red icon representing the unavailable status. The support team also determines that the unavailable server has to be restarted to return the system to full functionality.

- Monitoring proactively

  As the administrator of production systems, you have set appropriate thresholds for the fields displayed on the Server Statistics Overview page. During your regular monitoring, you see that the Paging Rate threshold is being crossed. The increase in paging rate probably means an increase in overhead. You can now increase memory, add servers, or take a similar course of action to keep production running smoothly.

In-Flight Request Search

Use In-Flight Request Search to improve your chances of locating a malfunctioning application in a server farm. In-Flight Request Search provides a snapshot of the transactions in progress, showing you hanging transactions.

- Investigating a hanging transaction

  Customers call and complain that they are having trouble completing transactions. You select In-Flight Request Search to locate a hanging transaction. On finding one, you view a method trace for the transaction. You can also see that the transaction is waiting for the return of a specific SQL call. You forward the method trace to a database administrator for further analysis.

- Isolating a problem with CPU utilization

  After viewing the Server Statistics Overview page, you notice that the CPU utilization is very high. You select In-Flight Request Search to see whether a transaction is present. It appears that the system is churning on a transaction.

Through a method trace, you suspect that the transaction is looping. You forward the method trace to a developer for further analysis.

Server Activity

Server Activity Display helps you to troubleshoot and fix hanging requests and evaluate the current performance of your applications.

- Troubleshooting an application that hangs

  You receive reports that several users of application Z cannot update their user preferences: Application Z times out after a minute of not responding. You look for application Z requests that have long resident times in the Active Requests tab of the Server Activity Display. View the Request Detail for one of these requests to determine why or where it is hanging.

- Understanding immediate workload

  When you perform normal monitoring of your servers, you notice that a server's average response time has recently increased, with no appreciable change in throughput. You start looking at the Recent Requests tab of the Server Activity Display to see what the most recently completed requests have been on that server. You can verify whether the requests are uniformly slow, or if there is variation among requests. This may help you to isolate whether it is a problem with the server (uniformly slow), or with an application (certain requests are slow). You can verify whether the slow requests are CPU-heavy, or if they are spending too many moments idle.

Recent Activity

Use Recent Activity to discover problems related to memory or other resources.

Evaluating the impact of garbage collection (GC): You suspect that frequent GC calls are affecting the performance of a server. Therefore, you select Recent Activity and set up the first graph to display the Number of Garbage Collections metric for the last 48 hours. In the second graph, you roll through the different metrics that are possibly affected by frequent GC.

Memory Diagnosis

You can gain insight into the JVM heap and memory information through Memory Diagnosis. Use this information to tune the JVM parameters, assess your resources, and locate evidence of memory leaks.

- Detecting a memory leak

  After creating a Memory Analysis report that compares JVM Heap Size to Average Response Time, you believe that there is a memory leak. You access the Memory Leak feature to check whether the amount of uncollected memory is increasing. You set up a candidate for the server in question. This

tells the system to collect heap data now and again after a specified amount of time. Then you can compare the heap data for the two periods of time to determine whether there is evidence of a memory leak.

- Supporting the need to purchase new servers

  The year-end budget is due and you have to project whether you have to buy more servers for your environment. You create a Memory Analysis report during peak usage and compare JVM Heap Size to the Number of Sessions. The number of servers is close to maxing out the current environment. As a capacity planner, you recommend that the company must increase the number of servers currently servicing the environment based on this supporting information.

JVM Thread Display

Use JVM Thread Display to view all the threads running within the JVM of an application server.

Alleviating high server response time: You are asked to investigate server A where the response time and JVM CPU% are higher than expected, but throughput is normal. You do not see any active requests in the In-Flight Request Search. Therefore, you suspect that there may be threads running outside the application server. You access the JVM Thread Display and notice a couple of suspect threads. After taking a thread dump for the JVM, you determine the details of the current thread that is misbehaving and either reprioritize or cancel the thread.

Software Consistency Check

Use Software Consistency Check to troubleshoot aberrant servers in an otherwise homogenous server group.

Comparing non-functioning servers to working servers: After an upgrade to Application B, which is deployed on multiple servers, requests on Server D are occasionally hanging while all the other servers are working fine. As an operator, you check the Runtime Environment and compare the server having problems to one of the properly functioning servers. You select the Installed Binary Check to verify whether the files on both servers are the same. You find that one of the files on Server D is not the same as the file on the server that is properly functioning. Install the proper file to correct the problem.

Trap and Alert Management

Use Trap and Alert Management to monitor server health and determine problems with applications. This feature prevents disruptions in service by receiving alerts before problems arise. It also gathers data that helps you to pinpoint the root cause of difficult-to-reproduce problems.

- Diagnosing GC

  In this scenario, GC on server J, ever so often, takes over five minutes. During these times, requests that typically complete in a few milliseconds take 10 seconds to complete. Because this problem does not occur frequently, you set a trap so that you can determine immediately when GC in server J is churning. In particular, you choose a Server Resource Trap for Garbage Collection Time with a Threshold of 120,000 ms (two minutes), choose the Alert Action to Send E-mail to your pager, and apply this trap to server J. When you receive the page, you have approximately three minutes to investigate server J (assuming that this is an example where the GC underway will take five minutes).

- Debugging complex applications

  You are monitoring application A, which has a J2EE component on server S and a mainframe Customer Relationship Management (CRM) back end. The Java component of application A frequently exhibits idle times of several seconds, even when there is not much load on server S. You do not want to run at L3, but you want to see in what methods the Java application is waiting. You set an Application Trap for Wait Time with a Threshold of 2,000 ms, by request for application A, Choose the Stack Trace Data Action, and apply this trap to server S. The next time a request for application A takes longer than two seconds, the system will take a stack trace of server S. Look in the Trap Action History to obtain the stack trace to determine where application A is waiting.

System Resources

System Resources helps you to tune your application servers.

- Eliminating bottlenecks

  The response time of application A becomes unacceptable when the server experiences modest throughput. You determine that much of the resident time is spent idle. To verify whether the cause is a bottleneck in the application server pools, use System Resources during these times to view the percentage of threads used in the Database Connection Pools, Thread Pools, and/or Java Connector Architecture (JCA) Connection Pools. If any pool is at or near 100%, it is likely that the demand for application A is saturating these resources. You may be able to fix the problem by creating more or larger pools.

- Diagnosing imbalanced performance

  You have several supposedly identical servers in server group G that host the same applications and have similar workloads. However, one of your servers in server group G is noticeably more sluggish than the others. To specifically investigate the differences in performance and resource usage among these

servers, you use System Resource Comparison to compare these servers, one resource at a time. You may find that they have different resources available, are configured differently, or serve different workloads.

Daily Statistics

Use Daily Statistics to view snapshots of the daily use of your WebSphere z/OS application server instances.

IBM Tivoli Composite Application Manager for WebSphere data collector downtime: You must restart the data collector that monitors your WebSphere application server on z/OS to reconfigure it, but you want to view the activity during this downtime. IBM Tivoli Composite Application Manager for WebSphere will not be able to collect the Performance Management Interface (PMI) statistics during the time that the data collector is down. However, you can get a view of the activity on the WebSphere z/OS application server using Daily Statistics, because this information comes from System Measurement Facility (SMF).

Note: This feature is not available for distributed (UNIX/Windows) versions of WebSphere Application Server. It is also not available for CICS or Information Management System (IMS).

System Resource Comparison

Use System Resource Comparison to compare a selected resource across all servers in a group.

- Verifying memory utilization

  You notice that memory usage for server Trade_01, in the Trade group, is very high and you want to check whether this is abnormal. You perform a comparison and view the JVM Memory Usage for all the servers in the server group Trade. You determine that other servers in this group are not utilizing memory at the same pace. You can now select Memory Analysis or Server Statistics Overview and begin to work out the problem.

- Confirming resources in preproduction

  You have two servers with the same applications installed. Before you place them both into production, you perform a System Resource Comparison to check the difference in their resources. You determine that server Quote_03 has 20 Database Connection Pools while server Quote_02 has only 10 pools. Increase the number of Database Connection Pools on server Quote_02.

Performance Analysis and Reporting

Use Performance Analysis and Reporting to analyze historical data. This helps you to understand the performance of your applications and the utilization of your servers.

- Investigating poor response time claims

  Customers complain about poor performance on Application A. As a performance analyst, you select IBM Tivoli Composite Application Manager for WebSphere and draw up a Response Trend Report for Application A for the last week to verify the customers' claims. When you verify that there are instances of poor response time, you decompose the problematic period to see how different requests impact the response time. Drill down to a method trace of an actual instance of a slow transaction, and e-mail this Trace Report to the developers so that they can determine why the transaction was slow.

- Predicting how servers will handle a new workload

  The marketing department is set to launch a new campaign to bring more visitors to your site. Your manager wants to make sure that there is sufficient capacity to handle the projected workload without degrading response times. As a capacity planner, you have to project how well your current servers will perform under the new workload. You create a Capacity Analysis report to compare throughput versus response time. You can use the trend line to estimate at what throughput the response time will be unacceptable.

Composite Requests

Use Composite Request features to monitor transactions that utilize resources on more than one server.

Discovering application architecture: Your manager asks you to provide an example of a complete transaction of an airline reservation application. This involves a Web-based Java application, a CICS credit card processing application, a CICS ticket reservation application, and a frequent-flyer account, which is also a CICS system. You check Performance Analysis and Reporting for examples of the airline reservation application, some of which will have the Composite Request indicator. Click the indicator to view the Composite Request View of the Method Trace. This allows you to navigate among these requests so that you can see which application calls which one, and by what mechanism. You can e-mail PDFs of each request that is involved in the composite transaction to your manager.

Note: Your IBM Tivoli Composite Application Manager for WebSphere administrator must enable composite request support for all data collectors that participate in composite requests.

Audit Trails

Audit Trails provide a means to trace user actions in the system. This helps in both accountability and troubleshooting.

- Verifying high server response time

  On returning from vacation, you see that the response time is high for most of the servers in the group ABC. You review the servers in the group and realize that two servers are missing. You enter the audit trail to check who took the servers offline. You contact the employee who took the servers offline and learn that the servers are being upgraded.

- Verifying report definition change

  In your role as a capacity planner you run a report and notice that its results are abnormal. When you review it, you see that the report's definition has changed. You ask the administrator to verify the audit trail to determine who changed the report's definition. You can now consult with your colleague about why the report's definition has changed.

Request Mapper

Use the Request Mapper to customize how requests are named within the application monitor. Also, use the Request Mapper to display the user names that are associated with requests.

- Aggregating across distinct Original Request String (ORS)

  The application you are monitoring uses a distinct Uniform Resource Identifier (URI) to represent each specific application function, such as login, checkout, or logout. You want to analyze all these requests as a single application. Use the Request Mapper to populate the Request Name field with a common application name.

- Differentiating a uniform ORS

  You are monitoring an application that uses session variables to represent the underlying function, while using the same request name throughout these different interactions. You want to compare the performance of different application functions, such as login, checkout, or logout. Therefore, you use the Request Mapper to assign each function a distinct request name.

Note: From the application server perspective, there are two major types of requests: JSP and Servlet. These calls come either from a Web server, or from an application server other than itself. We call such a request, generally expressed in the form of a string, ORS. It is composed of the URI plus the query string.

4.2 Solving application performance problems

This section demonstrates, in further detail, some specific techniques for troubleshooting and resolving performance problems using IBM Tivoli Composite Application Manager for WebSphere. The problems covered are:

- 4.2.1, “Response time problem” on page 104
- 4.2.2, “Locking problem” on page 107
- 4.2.3, “Memory leak investigation” on page 111
- 4.2.4, “SQL analysis” on page 120

4.2.1 Response time problem

You can perform the response time problem investigation directly from the IBM Tivoli Composite Application Manager for WebSphere Web console. Operators who monitor online response times from the summary monitoring views can identify response time problems immediately. See 4.1, “Monitoring WebSphere Application Server” on page 78.

One approach is to use the recent activity display. From the main menu, select **PROBLEM DETERMINATION → Server Activity → Activity Display**. You can quickly identify any recently executing transaction and its response time, as shown in Figure 4-18.

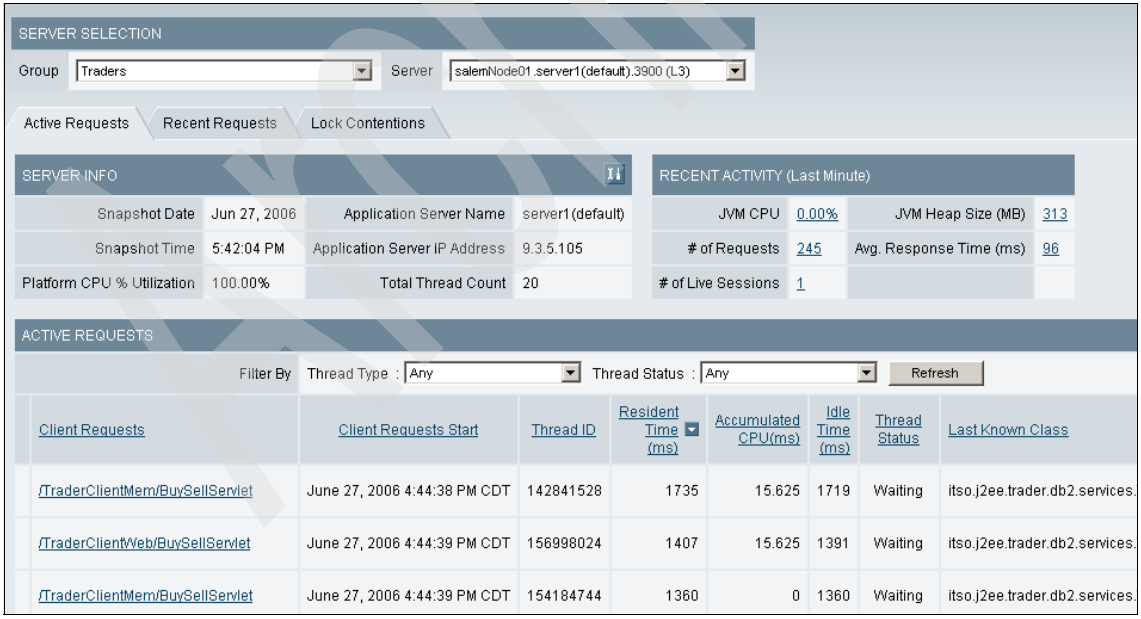


Figure 4-18 Recent activity display

The recent activity display only displays some of the most current transactions that are executing and observed by the data collector. This information is retrieved directly from the data collector.

Another approach is to show the request or transaction report for transaction problems that are no longer visible in the recent activity display. Select **PERFORMANCE ANALYSIS** → **Create Application Reports** → **Request/Transaction**. Select the appropriate time frame with Time Series in Hour axis option. The initial report shows response time breakdown by hour, as shown in Figure 4-19.

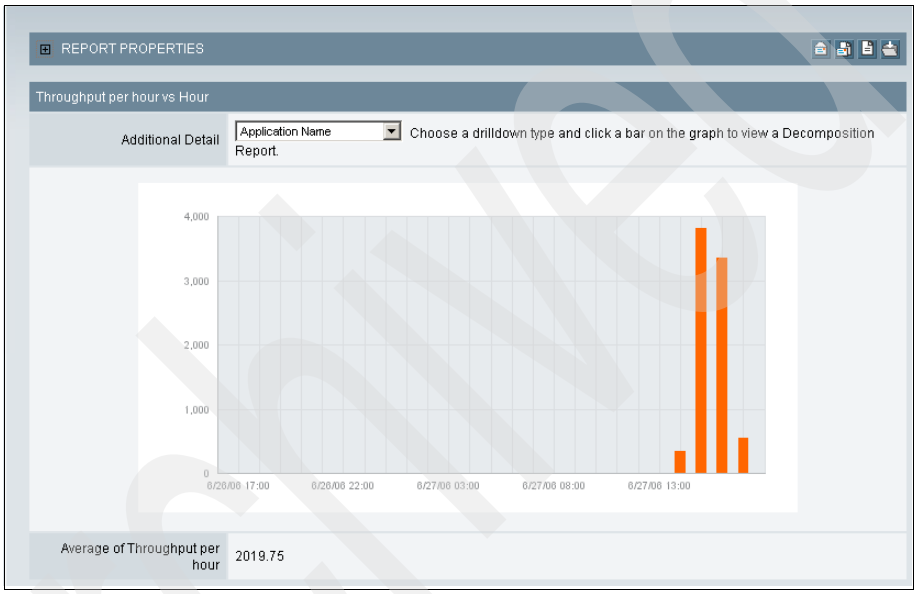


Figure 4-19 Response time distribution

Click the peak response time to drill into the data and see the decomposition chart, as shown in Figure 4-20. To further drill into the response time information, you can sort the application/transaction information by response time or percentage. Alternatively, you can double-click the section of the pie (showing percentage).

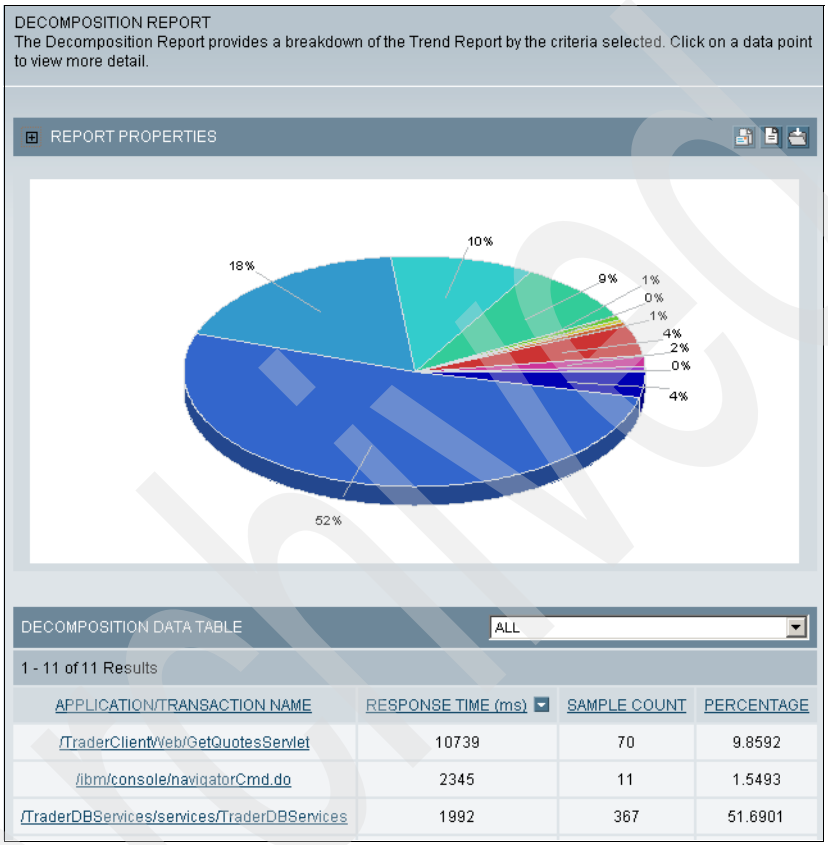


Figure 4-20 Decomposition chart

This opens a further detailed reports data table listing the instances of that named transaction with more detailed information specific to each of these transactions, as shown in Figure 4-21.

REQUEST REPORT DETAIL
The Request/Transaction Report Detail displays a breakdown of the data for the portion of the Decomposition Report.

Detail | Summary | Worst Performers | Locks

REPORT PROPERTIES

DETAIL REPORTS DATA TABLE — 20 per Page

1 - 20 of 70 Results — 1 2 3 4 Next > Last > >

REQUEST/TRANSACTION NAME	REQUEST/TRANSACTION TYPE	RESPONSE TIME (ms)	CPU TIME (ms)	SERVER NAME	TIMESTAMP	METHOD/COMPONENT RECORDS
/TraderClientWeb/GetQuotesServlet	Servlet	49843	203.125	salemNode01.server1 (default)(L3)	Jun 27, 2006 2:31:18 PM	30
/TraderClientWeb/GetQuotesServlet	Servlet	46969	46.875	salemNode01.server1 (default)(L3)	Jun 27, 2006 2:31:17 PM	32
/TraderClientWeb/GetQuotesServlet	Servlet	46859	0.000	salemNode01.server1 (default)(L3)	Jun 27, 2006 2:31:18 PM	26
/TraderClientWeb/GetQuotesServlet	Servlet	46828	31.250	salemNode01.server1 (default)(L3)	Jun 27, 2006 2:31:19 PM	26

Figure 4-21 Transaction list response time

You can identify the response time problem, whether it occurs erratically or consistently, for every transaction. Erratic problems may indicate locking issues, while consistently bad response time is usually a symptom of inefficient transaction design. Transaction design problems may require Level 2 monitoring analysis to identify the problematic methods. Locking problem analysis is discussed in the following section.

4.2.2 Locking problem

Problems with object locking in application servers are often hard to identify. Typical symptoms of a locking problem are erratic response time and occasional timeouts with an HTTP return code of 500. Although these symptoms can also be caused by bad data and programming, these symptoms are typical of a deadlock issue.

This section discusses lock analysis on the assumption that we have already identified a locking problem in an application. You must prepare and activate the

lock analysis facility within IBM Tivoli Composite Application Manager for WebSphere before you can use it.

To prepare for lock analysis, perform the following steps:

1. Increase the monitoring level to L2 or L3. After resolving the problem, ensure that you set the monitoring level back to L1. To temporarily override the monitoring level:
   a. From the menu, select **ADMINISTRATION** → **Monitoring On Demand**.
   b. Select the link under **Schedule Change/Override**, as shown in Figure 4-22.

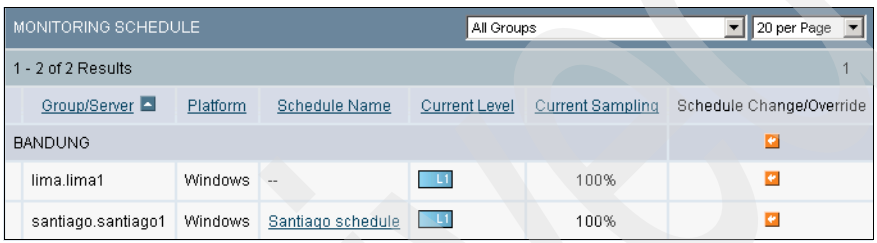


Figure 4-22 Selecting server for monitoring level change

   c. Set the monitoring level to at least **(L2) Problem Determination Mode** in the Server Setting Options window, as shown in Figure 4-23.

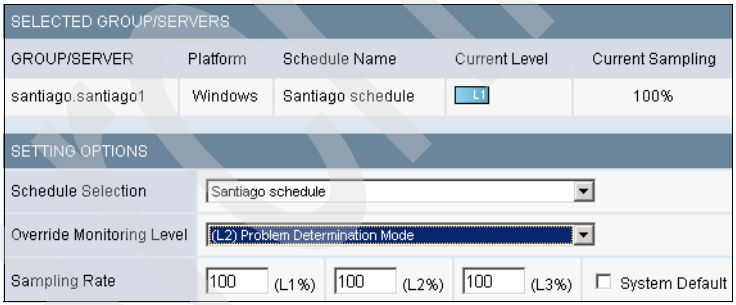


Figure 4-23 Changing monitoring level to L2

2. Before you perform lock analysis, enable the data collector for lock analysis. You must modify the data collector's property file cyn.user.classes.xml, located in the $DC_HOME/etc/ directory. The cyn.user.classes.xml file is referenced from the bcm.properties file in the userbcm.xmlfilename parameter.

The `lockAnalysis` tag determines whether the class should be instrumented for the lock analysis feature. Example 4-1 shows that all classes matching insuNetWeb.* are instrumented for lock analysis because the value `true` is specified for `lockAnalysis`.

Example 4-1 Enabling lock analysis

```
<defineInstrumentation>
  <enableSignature>true</enableSignature>
  <userClasses>
       <selectClass>
           <mask>insuNetWeb.*</mask>
           <methods>
               <methodName>*</methodName>
           </methods>
           <ignoreTrivial>true</ignoreTrivial>
           <lockAnalysis>true</lockAnalysis>
           <objectAllocations>
           <allocateClass>*</allocateClass>
           </objectAllocations>
           <objectAllocationTimes>true</objectAllocationTimes>
       </selectClass>
   </userClasses>
</defineInstrumentation>
```

If the data collector does not have lock analysis instrumentation activated, lock analysis will not work and you will see the following message:

```
CYNVE0851E Lock analysis data is not being collected.
```

After you complete the two-step enablement process, perform the lock analysis using the Web console.

1. From the menu bar, select **PROBLEM DETERMINATION → Server Activity Display**.
2. On the Server Activity Display page (Figure 4-24), you can determine and verify that:
   - InsuNetWeb application is still hanging at the quoteAction component.
   - Resident Time and Idle Time show hours.
   - The Thread Status is `Waiting`. This indicates that the thread has probably entered a deadlock state.
   - The concrete Java class that is probably causing this deadlock is insunet.web.actions.QuoteAction.

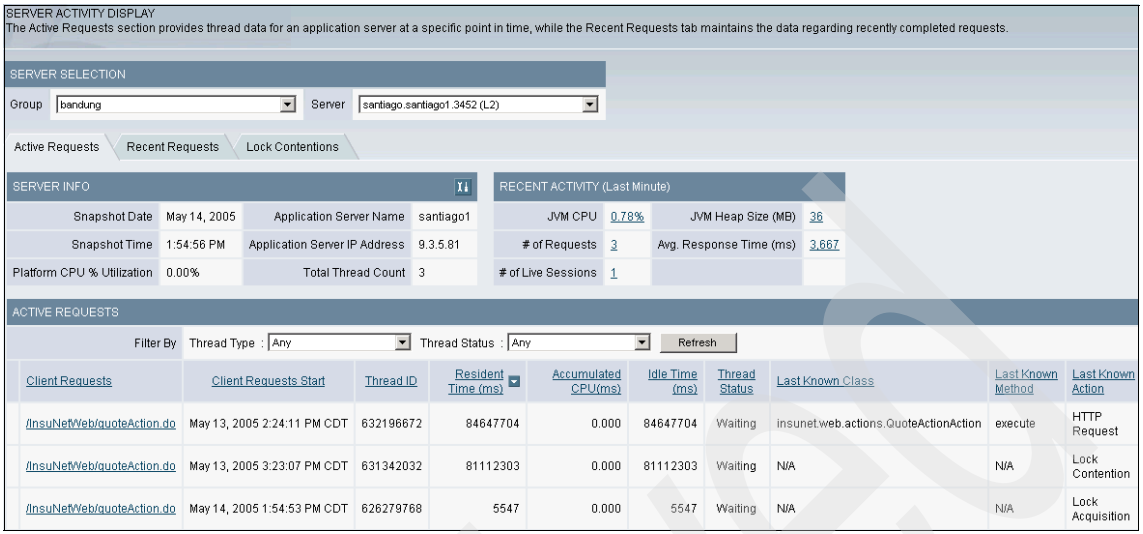


Figure 4-24 Server activity display

You can confirm the same information by selecting the **Lock Contentions** tab and viewing the list of locks in this same component.

3. Select **Cancel Requests**, causing the lock contention to check the request detail page shown in Figure 4-25. Note the available details on this page. Click **OK** to cancel the request.

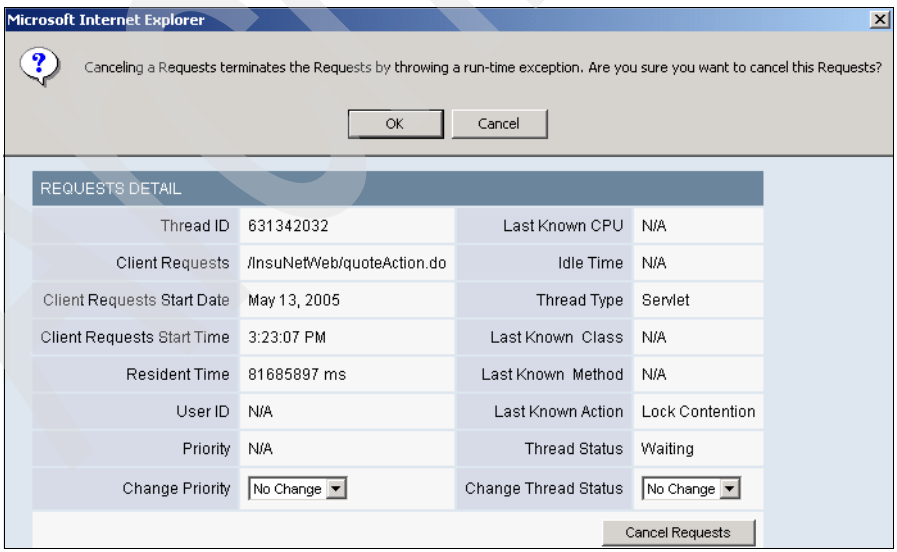


Figure 4-25 Cancelling request for Lock contention

4. On the left-side navigator menu, select **Link Method/Component Trace** to generate a detailed Complete Flow View, as shown in Figure 4-26. From this report perspective, you can identify what method in what class caused the lock contention. In this example, the class is `insunet.web.action.QuoteTransaction` and the method is `synchronize_update`.

COMPLETE FLOW VIEW — 1000 per Page

Queue Names Match Queue Names Don't Match ** indicates values that cross thresholds

1 - 7 of 7 Results

Depth	Event Type	Event Data	Elapsed Time (ms)	CPU Time (ms)	Δ Elapsed Time (ms)	Δ CPU Time (ms)
0	Servlet Entry	/InsuNetWeb/quoteAction.do	0	0	0	0
1	JNDI Entry	**Provider URL:**corbaloc:rir:/NameServiceServerRoot **Lookup Name:**ejb/insunet/ejb/QuoteMediatorHome	0	0	0	0
1	JNDI Exit	**Provider URL:**corbaloc:rir:/NameServiceServerRoot **Lookup Name:**ejb/insunet/ejb/QuoteMediatorHome	0	0	0	0
1	EJB Entry	**EJB Name:**insunet.ejb.QuoteMediatorBean **Method:**create	0	0	0	0
1	EJB Exit	**EJB Name:**insunet.ejb.QuoteMediatorBean **Method:**create	0	0	0	0
1	Lock Acquisition Entry	**Lock Object Class:**java.lang.Class **Class Acquiring Lock:**insunet.web.actions.QuoteTransaction **Method Acquiring Lock:**synchronizeUpdate	0	0	0	0
2	Lock Contention Entry	**Lock Object Class:**java.lang.Class **Class Owning Lock:**insunet.web.actions.QuoteTransaction **Method Owning Lock:**synchronizeUpdate	0	0	0	0

Figure 4-26 Flow view

The lock analysis investigation process has provided sufficient information to enable a development team to fix the malfunctioning code.

4.2.3 Memory leak investigation

Memory is a key resource that is allocated to an application server for use by the application components running on it. Complications arise when memory is used by components and erroneously not returned back to the system for reuse by the application. This memory leak condition is critical for long-running, mission-critical application servers that do not get restarted for long periods of time.

IBM Tivoli Composite Application Manager for WebSphere supports detailed analysis of memory that is allocated to application servers. You can perform memory leak analysis using the following three steps:

- “Investigating a potential memory leak” on page 112
- “Narrowing the cause of a memory leak” on page 115
- “Determining the memory leak offender” on page 117

Investigating a potential memory leak

When a memory leak situation is suspected, investigation begins. The investigation comprises the following:

- Memory leak confirmation reports
- Trend reports for slower leaks
- Using traps to alert for low memory

In this section, we show only the memory leak confirmation report. IBM Tivoli Composite Application Manager for WebSphere has pre-built reports that are useful for determining whether memory growth is related to memory leak activity. To access these reports, select **PROBLEM DETERMINATION** → **Memory Diagnosis** → **Memory Leak**.

The Step 1: Memory Leak Confirmation Report window (Figure 4-27) shows three types of reports. Each report is labeled according to the question it answers and the data it presents.

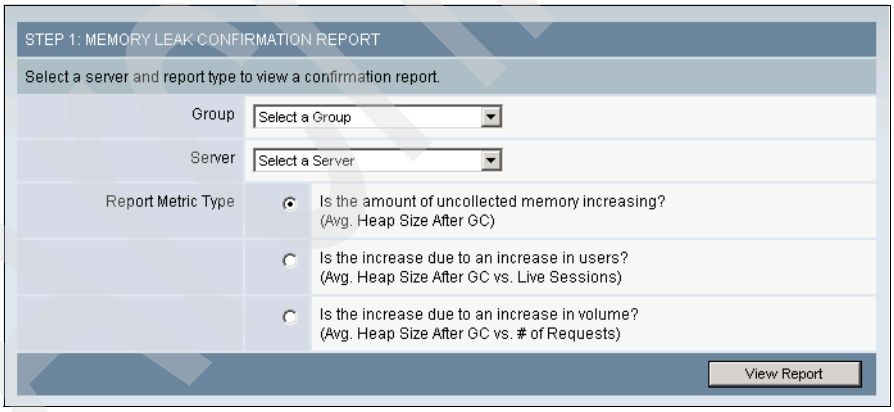


Figure 4-27 Three pre-built memory leak reports for memory leak confirmation

The first report option graphs memory growth over time. Memory growth is only shown after GC as this represents the real amount of used memory. However, memory growth can also occur because there are more users or more activities causing more Java objects to be allocated. The second report option presents

memory growth as a function of users. The third report option plots memory growth as a function of transaction volume.

Figure 4-28 shows an upward memory growth trend even though the number of live sessions remains relatively steady in the range of 50 sessions. Such an upward memory growth trend coupled with steady session metrics indicates that a memory leak is likely.

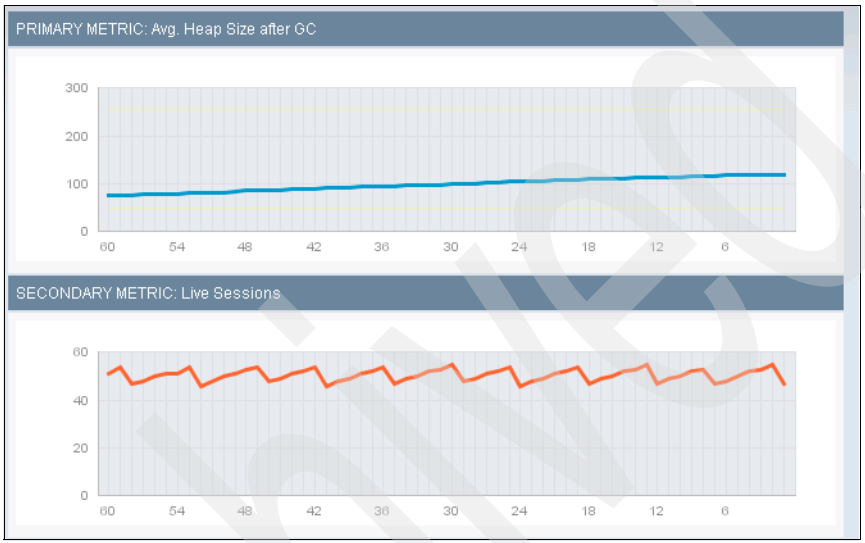


Figure 4-28 Memory growth with a constant average number of live sessions

Similarly, the report for heap size after GC compared to the number of requests helps to indicate whether server load is affecting memory growth. Actively executing Java methods adds objects to memory heap load as they complete their tasks. As soon as these requests are finished, memory usage should be freed accordingly.

Figure 4-29 shows a slow upward growth in memory usage for the first 45 minutes with a corresponding large number of requests per minute. When these requests drop to zero in the last 15 minutes of the graph, the heap size flattens but does not drop as is expected. This graph indicates that there can be a memory leak because the rise in memory occurs with a consistent average number of requests but does not fall when the number of requests falls.

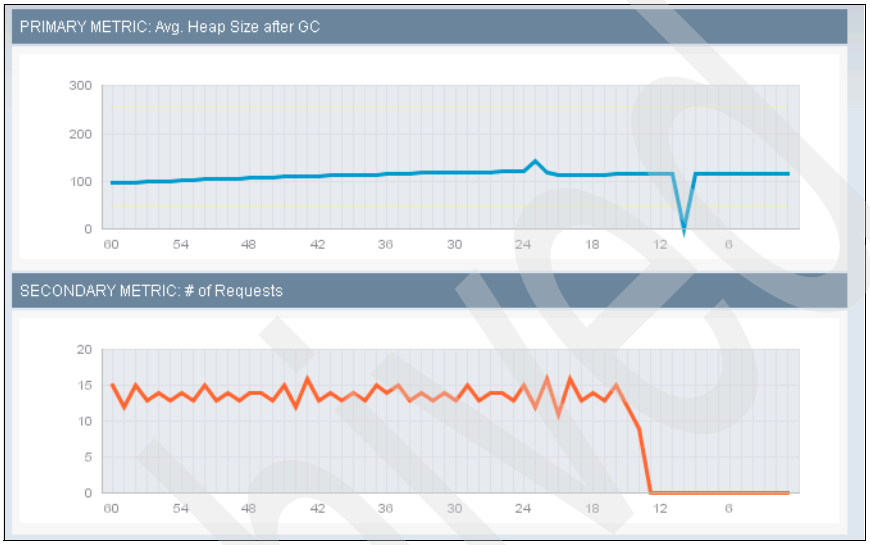


Figure 4-29 Heap size does not decrease when the number of requests drops

Note: The three canned reports in the memory leak report are more useful than the three offered in the memory analysis report's JVM heap size section. The memory leak reports use the average heap size after GC as their baseline. However, the memory analysis canned reports use JVM heap size. When you investigate memory leaks, we recommend that you perform a GC so that no allocated but unused objects reside in memory when determining the memory size.

By default, the three memory leak graphs show the past 60 minutes of data. This time range may be too narrow. There is an option to change the graph range to the past 48 hours. You can change any of the mapped metrics as well as the server and group selected to quickly compare a different server's memory growth. To detect a slower memory leak, use the trend report from system reports to generate a report of memory growth over a period of several weeks.

Narrowing the cause of a memory leak

When a leak is suspected, IBM Tivoli Composite Application Manager for WebSphere provides a Step 2: Memory Leak Candidate Finder Report that helps to identify potential leaking classes. The leak candidate report takes two snapshots of the suspected memory leaking JVM. It compares the older snapshot to the newer one to determine which objects have increased their size between the snapshots.

Specify the server group, server, and the wait time, as shown in Figure 4-30. The wait time specifies how long to wait before taking the second snapshot of the JVM heap. This wait time depends on the rate of growth of the leak. If an initial memory leak investigation indicates a noticeable growth in just an hour, then a wait time of as little as 5 minutes to 10 minutes may be sufficient. If the noticeable growth is in the order of a day or more, then you must wait for a few hours.

Note: Ideally you must take the snapshot when there is little or no load to ensure that all the objects are in use.

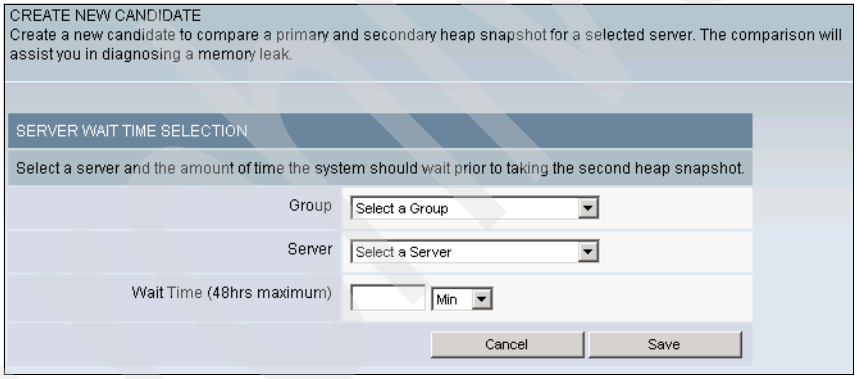


Figure 4-30 Configuring memory leak candidate finder

Note: You can take the JVM heap snapshots with the monitoring level at L1, L2, or L3.

When the report has finished collecting both snapshots, click the server name to show the analysis report for the first snapshot. The same report is available for the second snapshot. This report shows the objects that were in memory when the snapshot was made and the associated metrics such as the percentage of memory they consumed.

The comparison data report is more useful than individual JVM heap snapshot reports because it shows only classes that have changed between the two snapshots. Figure 4-31 shows the comparison report sorted by the change in the number of instances of live objects in memory. The top entries in this heap comparison results table are the most likely memory leak candidates because these are the classes that have grown the most. We use these classes to determine the cause of the memory leak.

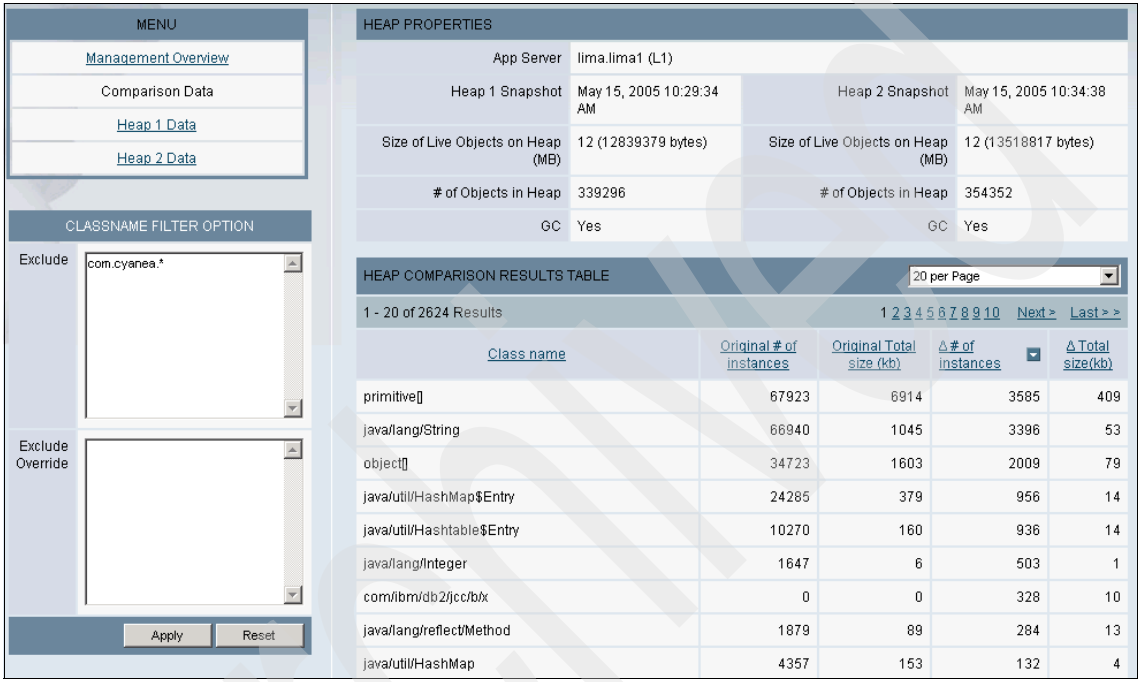


Figure 4-31 Memory Leak candidate comparison report

Tip: Classes that are filtered out by the monitoring configuration installation defaults are excluded from this analysis. Although you may not want to monitor basic classes such as strings and lists, these objects can often be a source of leaks and should be examined. To do this, clear the exclude section of the class name filter option and click **Apply** to see everything in the heap. (This is done dynamically without re-executing the report.) Selectively exclude classes and packages that you determine are not a concern. Sort the columns from most to least that show changes in the number of instances. This will result in a richer view of the objects that are growing in memory.

Determining the memory leak offender

After you identify the potential leaking classes, the next step is to determine which classes are actually causing the leak. The Step 3: Memory Leak Diagnosis Report finds the most-leaked objects and shows the class, method, and line number of the allocating class of leaked objects. This detailed information, down to the code line number, helps a programmer to determine the trouble spots of code.

To set up this report, reconfigure the data collector to monitor the suspected growing classes that you identified using the technique described in “Narrowing the cause of a memory leak” on page 115. The memory leak diagnosis report is set up using an eXtensible Markup Language (XML) file on the data collector installed machine in the $DC_HOME/etc/ directory. The bcm.properties file has a key/value pair called *userbcm.xmlfilename* that identifies which XML configuration file is used. Adjusting the bcm.properties files allows for multiple XML files to be swapped in and out.

Note: IBM Tivoli Composite Application Manager for WebSphere generates a server-specific bcm.properties file called <*host name*>.<*app server name*>.bcm.properties. When you change the original bcm.properties, you must delete the generated version so that IBM Tivoli Composite Application Manager for WebSphere can regenerate the bcm.properties file.

The XML file is composed of multiple sections of selectClass tags that identify how to treat classes matching the mask pattern. The XML snippet in Example 4-2 shows the modification required to monitor memory leaks.

Example 4-2 XML for monitoring memory leak

```
<selectClass>
    <mask>*</mask>
    <methods>
         <methodName>*</methodName>
    </methods>
    <ignoreTrivial>true</ignoreTrivial>
    <lockAnalysis>true</lockAnalysis>
    <objectAllocations>
         <allocateClass>*</allocateClass>
    </objectAllocations>
    <objectAllocationTimes>true</objectAllocationTimes>
</selectClass>
```

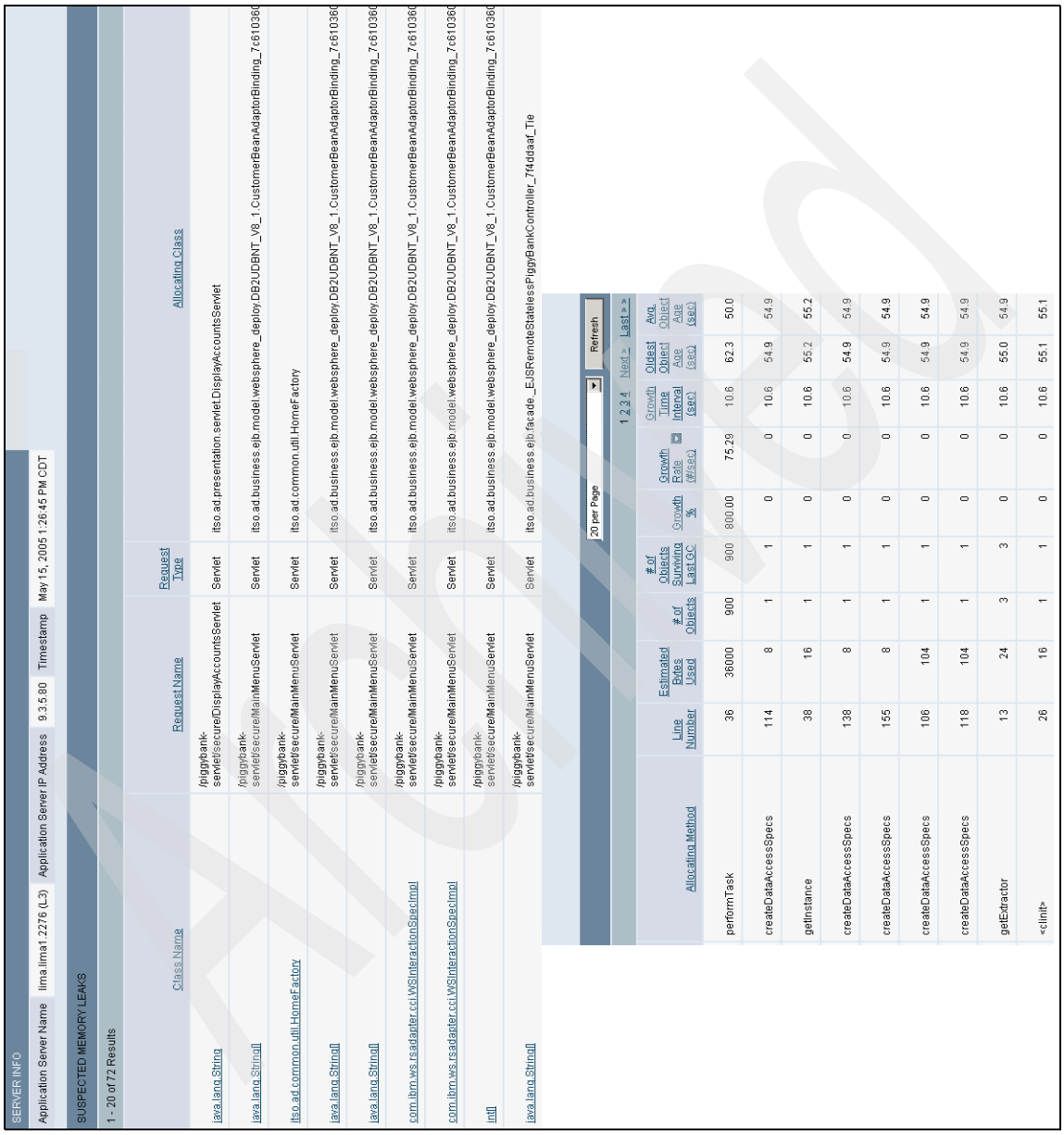
Only classes found inside the allocateClass tag are monitored as potential leaks. An asterisk in the tag monitors all classes for leaks. A better strategy is to replace

the asterisk with the top class names identified from the Step 2: Memory Leak Candidate Finder Report (Figure 4-31 on page 116) for further analysis. If Step 2 does not offer any good candidates, using the asterisk means that all the objects are monitored. This will have an adverse impact on performance.

If you set the objectAllocationTimes tag to true, this activates and populates the growth percentage and growth rate columns. This can help to show the rate of growth in the objects.

Important: The diagnosis report is functional only in L3 mode. It can slow down a production system severely and should be used only in controlled test situations.

The Memory Leak Diagnosis Report is Step 3 of the memory leak report. You select the group and server to get to the report's table. The report result is a very wide table. Figure 4-32 shows the overall result.


SERVER INFO

Application Server Name	lima.lima1:2276 (L3)	Application Server IP Address	9.3.5.80	Timestamp	May 15, 2005 1:26:45 PM CDT

SUSPECTED MEMORY LEAKS

1 - 20 of 72 Results

Class Name	Request Name	Request Type	Allocating Class
java.lang.String	/piggybank-servlet/secure/DisplayAccountsServlet	Servlet	itso.ad.presentation.servlet.DisplayAccountsServlet
java.lang.String[]	/piggybank-servlet/secure/MainMenuServlet	Servlet	itso.ad.business.ejb.model.websphere_deploy.DB2UDBNT_V8_1.CustomerBeanAdaptorBinding_7c61036(
itso.ad.common.util.HomeFactory	/piggybank-servlet/secure/MainMenuServlet	Servlet	itso.ad.common.util.HomeFactory
java.lang.String[]	/piggybank-servlet/secure/MainMenuServlet	Servlet	itso.ad.business.ejb.model.websphere_deploy.DB2UDBNT_V8_1.CustomerBeanAdaptorBinding_7c61036(
java.lang.String[]	/piggybank-servlet/secure/MainMenuServlet	Servlet	itso.ad.business.ejb.model.websphere_deploy.DB2UDBNT_V8_1.CustomerBeanAdaptorBinding_7c61036(
com.ibm.ws.rsadapter.cci.WSInteractionSpecImpl	/piggybank-servlet/secure/MainMenuServlet	Servlet	itso.ad.business.ejb.model.websphere_deploy.DB2UDBNT_V8_1.CustomerBeanAdaptorBinding_7c61036(
com.ibm.ws.rsadapter.cci.WSInteractionSpecImpl	/piggybank-servlet/secure/MainMenuServlet	Servlet	itso.ad.business.ejb.model.websphere_deploy.DB2UDBNT_V8_1.CustomerBeanAdaptorBinding_7c61036(
int[]	/piggybank-servlet/secure/MainMenuServlet	Servlet	itso.ad.business.ejb.model.websphere_deploy.DB2UDBNT_V8_1.CustomerBeanAdaptorBinding_7c61036(
java.lang.String[]	/piggybank-servlet/secure/MainMenuServlet	Servlet	itso.ad.business.ejb.facade._EJSRemoteStatelessPiggyBankController_7f4ddaaf_Tie

20 per Page | Refresh

1 2 3 4 Next > Last > >

Allocating Method	Line Number	Estimated Bytes Used	# of Objects	# of Objects Surviving Last GC	Growth %	Growth Rate (#/sec)	Growth Time Interval (sec)	Oldest Object Age (sec)	Avg. Object Age (sec)
performTask	36	36000	900	900	800.00	75.29	10.6	62.3	50.0
createDataAccessSpecs	114	8	1	1	0	0	10.6	54.9	54.9
getInstance	38	16	1	1	0	0	10.6	55.2	55.2
createDataAccessSpecs	138	8	1	1	0	0	10.6	54.9	54.9
createDataAccessSpecs	155	8	1	1	0	0	10.6	54.9	54.9
createDataAccessSpecs	106	104	1	1	0	0	10.6	54.9	54.9
createDataAccessSpecs	118	104	1	1	0	0	10.6	54.9	54.9
getExtractor	13	24	3	3	0	0	10.6	55.0	54.9
<clinit>	26	16	1	1	0	0	10.6	55.1	55.1

Figure 4-32 Memory leak diagnosis report

By default, the Growth Rate column is sorted so that the fastest-growing classes are shown at the top of the table. The columns that indicate the allocating class of the leaking object are very useful and important. This information directs the programmer where to look and suggest why the classes may be leaking.

Note: If you have not set objectAllocationTimes, there will be no growth rate data. Thus, the default sort will not be useful and we suggest another sorting of this information. Sorting by the Number of Objects Surviving Last GC column may be effective.

4.2.4 SQL analysis

For SQL-based transactions that are invoked directly using a Java Database Connectivity (JDBC) interface instead of a Java 2 Connectivity (J2C) resource, SQL reports are available for generation. The IBM Tivoli Composite Application Manager for WebSphere SQL report reveals SQL commands issued by the application along with their response times. This report is useful for understanding whether a response time problem is caused by SQL calls.

To generate the SQL report, select **PERFORMANCE ANALYSIS** → **Create Application Reports** → **SQL**. Follow the prompts to generate a report. A sample SQL report is shown in Figure 4-33.

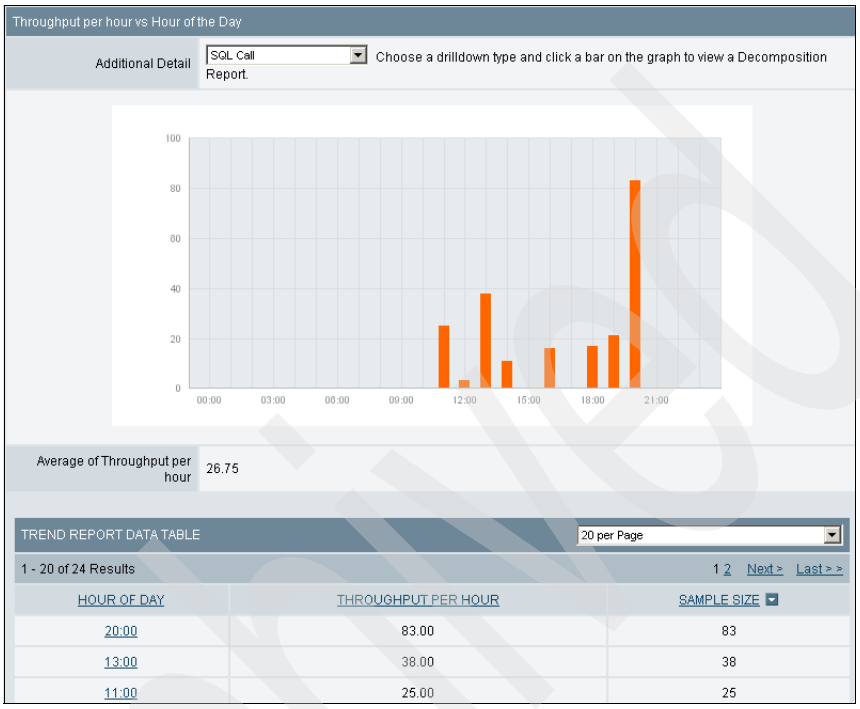


Figure 4-33 Sample SQL report

The SQL Analysis Report provides information about SQL calls that have been processed by the application server. You can generate a Trend Report, Decomposition Report, and detailed SQL Report (including Detail, Summary, and Worst Performers tabs) from the SQL Analysis Report after it is created.

A detail SQL report is shown in Figure 4-34.

Detail | Summary | Worst Performers | Locks

REPORT PROPERTIES

DETAIL REPORTS DATA TABLE

1 - 8 of 8 Results

SQL CALL	TABLE/CURSOR NAME	SQL	RESPONSE TIME (ms)	SERVER NAME	TIMESTAMP	REQUEST/TRANSACTION NAME
SELECT	TRADER.COMPANY	SELECT TRADER.COMPANY.COMPANY FROM TRADER.COMPANY	266	salemNode02.server1 (clnt001)	Jul 5, 2006 10:38:19 PM	/TraderDBServices/VBDTradeGetCompanies
SELECT	TRADER.COMPANY	SELECT TRADER.COMPANY.COMPANY FROM TRADER.COMPANY	250	salemNode02.server1 (clnt001)	Jul 5, 2006 10:38:19 PM	/TraderDBServices/VBDTradeGetCompanies
SELECT	TRADER.COMPANY	SELECT TRADER.COMPANY.COMPANY FROM TRADER.COMPANY	250	salemNode02.server1 (clnt001)	Jul 5, 2006 10:38:19 PM	/TraderDBServices/VBDTradeGetCompanies
SELECT	TRADER.COMPANY	SELECT TRADER.COMPANY.COMPANY FROM TRADER.COMPANY	250	salemNode02.server1 (clnt001)	Jul 5, 2006 10:38:19 PM	/TraderDBServices/VBDTradeGetCompanies
SELECT	TRADER.COMPANY	SELECT TRADER.COMPANY.COMPANY FROM TRADER.COMPANY	250	salemNode02.server1 (clnt001)	Jul 5, 2006 10:38:19 PM	/TraderDBServices/VBDTradeGetCompanies
SELECT	TRADER.COMPANY	SELECT TRADER.COMPANY.COMPANY FROM TRADER.COMPANY	234	salemNode02.server1 (clnt001)	Jul 5, 2006 10:38:19 PM	/TraderDBServices/VBDTradeGetCompanies
SELECT	TRADER.COMPANY	SELECT TRADER.COMPANY.COMPANY FROM TRADER.COMPANY	234	salemNode02.server1 (clnt001)	Jul 5, 2006 10:38:19 PM	/TraderDBServices/VBDTradeGetCompanies
SELECT	TRADER.COMPANY	SELECT TRADER.COMPANY.COMPANY FROM TRADER.COMPANY	0	salemNode02.server1 (clnt001)	Jul 5, 2006 10:38:19 PM	/TraderDBServices/VBDTradeGetCompanies

Figure 4-34 Detail SQL report

This SQL information is useful for investigating database performance or tuning issues, application design issues, as well as other SQL-related problems.

This section has demonstrated some specific techniques for troubleshooting and resolving performance problems using IBM Tivoli Composite Application Manager for WebSphere. Problem investigation strategies and examples using Tivoli Enterprise Monitoring are presented in Appendix A, “Tivoli Enterprise Monitoring Agent component” on page 127.

4.3 IBM Tivoli Composite Application Manager for WebSphere problem determination

Now that we have presented the usage of IBM Tivoli Composite Application Manager for WebSphere in day-to-day operations, it may be necessary to also find and repair problems with IBM Tivoli Composite Application Manager for WebSphere itself. This section provides information about the location of various useful files and how to manipulate logging and tracing to help resolve problems with IBM Tivoli Composite Application Manager for WebSphere. The discussion is divided into:

4.3.1 Configuration files

The configuration files consist of the following files:

- Managing server configuration files

  The managing server configuration information is contained in these files:

  bin/setenv.sh Setting environment variables for all kernel components

  etc/*.properties Individual component-specific property files

- Data collector configuration files

  The data collector is governed by several configuration files:

  cynlogging.properties
  Message logging and tracing level for the data collector components. The level is typically set to INFO. Other possible levels are DEBUG_MIN, DEBUG_MID, and DEBUG_MAX.

  datacollector.policy Java security permission of the data collector

  datacollector.properties
  Monitoring properties, levels, and timeout parameters. This file is used to generate an instance-based configuration file called <*node*>.<*server*>(<*profile*>).datacollector.properties.

  bcm.properties Lists the bcm XML files that are used by the data collector to instrument classes for providing additional monitoring information. The original Java classes do *not* have to be modified. This file is also used to

	generate an instance-based configuration file called <*node*>.<*server*>.bcm.properties.
gpsCounter.txt	Counter that identifies a sequence number for correlating and matching a composite transaction
***.xml**	A set of XML files that is referenced by the bcm.properties file to instruct the data collector as to how to modify the instrumented classes. It is necessary to de-reference the appropriate XML file to *not* instrument a specific function.

4.3.2 Logs and traces

The IBM Tivoli Composite Application Manager for WebSphere logs are located either in the Tivoli common log directory with the identifier of CYN or in the logs subdirectory of the installation path. For Windows-based systems, the default installation path is C:\Program Files\ibm\tivoli\common. For UNIX-based systems, the default installation path is /var/ibm/tivoli/common. You can use a custom installation path. In general, you can modify the logging level, either from the properties file to include `am.debug=yes` or by using the control commands: **`dcctl.sh`** or **`amctl.sh`**.

The managing server logs are:

- $AM_HOME/logs
  - am_stderr.log
  - am_stdout.log
- /var/ibm/tivoli/common/CYN/logs
  - msg-<*component*>.log
  - trace-<*component*>.log
  - audit-ms.log

The distributed data collector logs are:

- $commondir\CYN\logs: some logs may reside in <*admin*>.<*server*> path
  - msg-dc.log
  - trace-dc.log
  - msg-dc-native.log
  - trace-dc-native.log
- $DC_HOME\logs: <*admin*>.<*server*>.datacollector.log

The z/OS data collector logs are:

/var/ibm/tivoli/common/CYN/logs

- trace-zprobe.log
- trace-imsprobe-native.log
- msg-zprobe-native.log
- msg-zprobe.log

The cynlogging.properties file controls the logging level of the components. We recommend that you activate only the appropriate detailed logging level for a specific component. An example of this is to modify part of the logging for the CICS instrumentation to DEBUG MAX, as shown in Example 4-3.

Example 4-3 Excerpt of cynlogging.properties

```
   .  .  .

#-------------------------------------------
# COMPONENT SPECIFIC LOGGERS
#-------------------------------------------

   .  .  .
#-------------------------------------------
# CICS Data Collector
#-------------------------------------------

# MESSAGE LOGGER
CYN.msg.cicsdc.level=INFO
CYN.msg.cicsdc.logging=true

# TRACE LOGGER
CYN.trc.cicsdc.level=DEBUG_MAX
CYN.trc.cicsdc.logging=true

   .  .  .
```

A

Tivoli Enterprise Monitoring Agent component

This appendix describes the IBM Tivoli Composite Application Manager for WebSphere interface to IBM Tivoli Monitoring. The IBM Tivoli Composite Application Manager for WebSphere Tivoli Enterprise Monitoring Agent is based on the IBM Tivoli Monitoring Services V6.1. This section assumes that you have already built your IBM Tivoli Monitoring V6.1 environment. If not, you have to install your Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server before installing the monitoring agent.

The Tivoli Enterprise Monitoring Agent installation is divided into:

Installing Tivoli Enterprise Monitoring Agent

For feeding information into the Tivoli Enterprise Monitoring Server with Tivoli Enterprise Portal, you must install the Tivoli Enterprise Management Agent for IBM Tivoli Composite Application Manager for WebSphere on each data collector system.

Start the setup from the IBM Tivoli Composite Application Manager for WebSphere Tivoli Enterprise Monitoring Agent on the same machine where you have installed the WebSphere Application Server data collector that you want to expose with IBM Tivoli Monitoring V6.1.

1. For the feature to install, select only the Tivoli Enterprise Monitoring Agent on the data collector machine, as shown in Figure A-1.

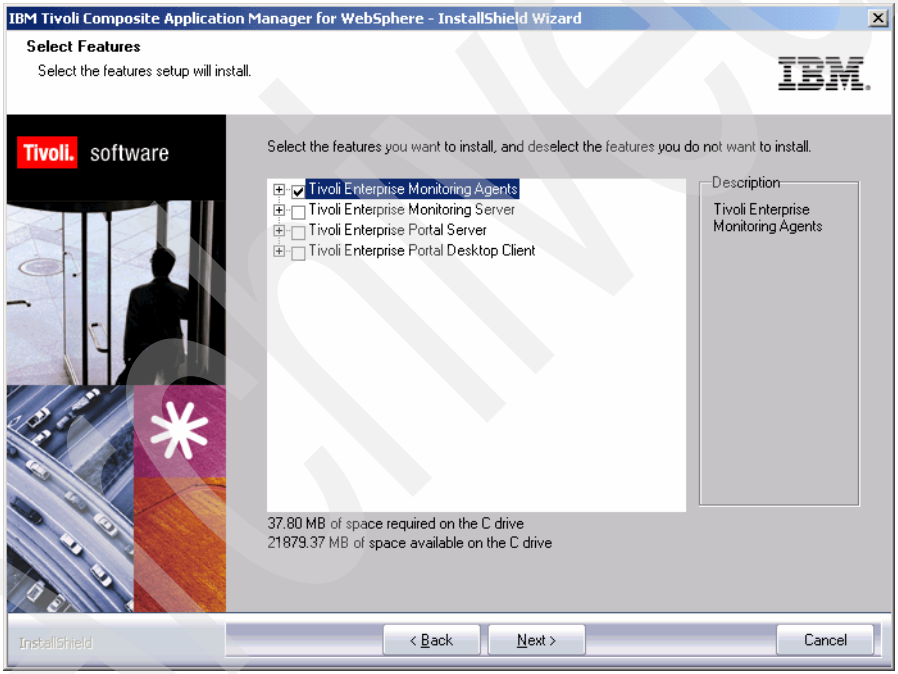


Figure A-1 Tivoli Enterprise Monitoring Agent installation

2. When the installation is finished, you have to configure the agent communication for Tivoli Enterprise Monitoring Server. The communication uses IP:PIPE. You must specify the Tivoli Enterprise Monitoring Server server name, as shown in Figure A-2.

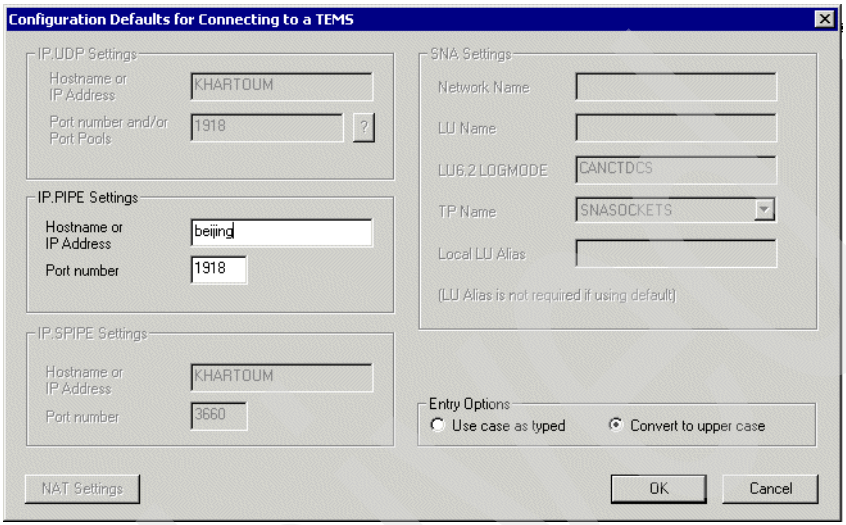


Figure A-2 Tivoli Enterprise Monitoring Agent communication

3. Configure the IBM Tivoli Composite Application Manager for WebSphere Tivoli Enterprise Monitoring Agent. The general page is shown in Figure A-3.

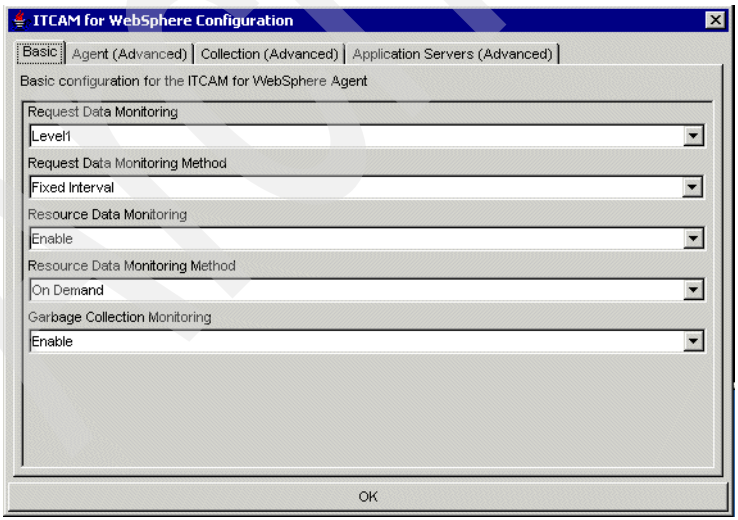


Figure A-3 Tivoli Enterprise Monitoring Agent basic configuration

4. Define the WebSphere Application Server environment for the agent to monitor. This is specified in the Application Server (Advanced) tab. Click **New** to define the server. The definition screen is shown in Figure A-4.

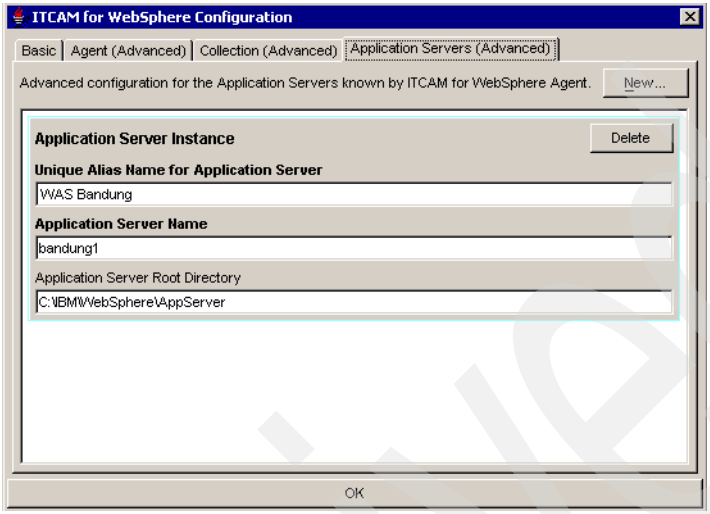


Figure A-4 Tivoli Enterprise Monitoring Agent server information

Seeding Tivoli Enterprise Monitoring Server

To integrate IBM Tivoli Composite Application Manager for WebSphere information with the IBM Tivoli Monitoring V6.1 infrastructure, you must set the infrastructure to support this type of agent.

- Tivoli Enterprise Monitoring Server

  Generates agent-specific information in the Tivoli Enterprise Monitoring Server such as product situations and agent tables.

- Tivoli Enterprise Portal Server

  Adds workspaces and presentation files for the agents to the Tivoli Enterprise Portal Server.

- Tivoli Enterprise Portal

  Updates the Tivoli Enterprise Portal clients with product-specific information such as help files.

This section covers the following steps from a process point of view and guides you through the installation process. For installation and customization, we follow the instructions in *IBM Tivoli Composite Application Manager for WebSphere Installing and Configuring the Tivoli Enterprise Monitoring Agent*, SC32-1801.

Important: It is important to understand what machine these activities have to run on. Check with your IBM Tivoli Monitoring V6.1 administrator and consult the appropriate IBM Tivoli Monitoring V6.1 documentation.

1. Start the installation by clicking **setup** in the Windows directory of the IBM Tivoli Composite Application Manager for WebSphere Tivoli Enterprise Monitoring Agent CD-ROM. Alternatively, you can run the **`install.sh`** command in your Linux/UNIX environment.
2. Figure A-5 shows the welcome screen for the installation of IBM Tivoli Composite Application Manager for WebSphere. Click **Next** on the welcome screen and follow the installation wizard dialogs.

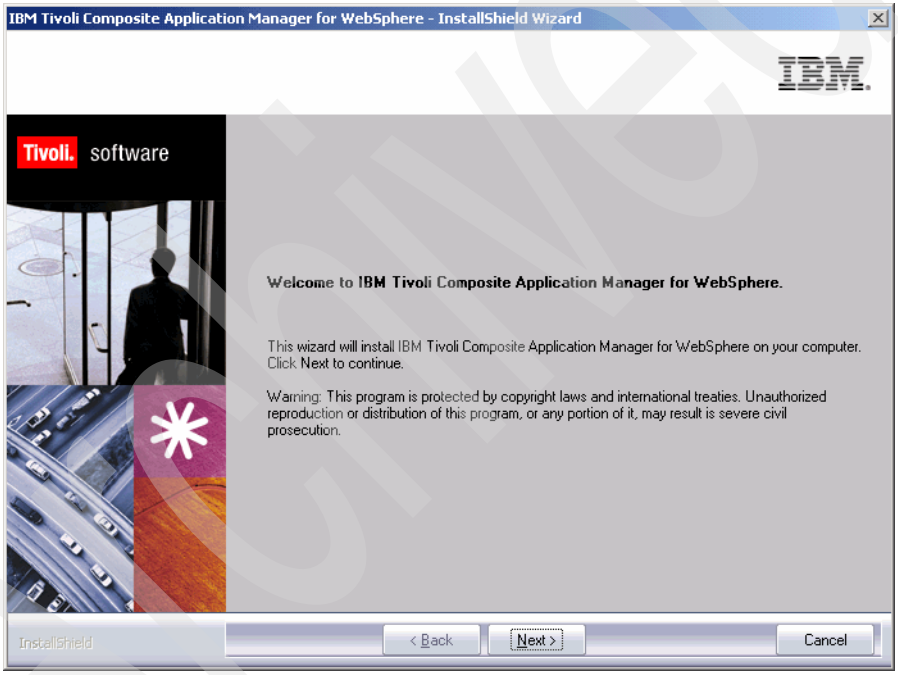


Figure A-5 Tivoli Enterprise Monitoring Agent welcome screen

3. Select the features to install. Depending on where you are running this installation wizard, you can install different components on different machines. In our environment, we have both the Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server in a single machine. Therefore, we select the following, as shown in Figure A-6:

   – Tivoli Enterprise Monitoring Server
   – Tivoli Enterprise Portal Server
   – Tivoli Enterprise Portal Server

   Click **Next**.

   Note: You may have to install the agent support files on Tivoli Enterprise Monitoring Server machine, Tivoli Enterprise Portal Server machine, and the individual Tivoli Enterprise Portal desktop client. You have to install only the component that is available in each machine.

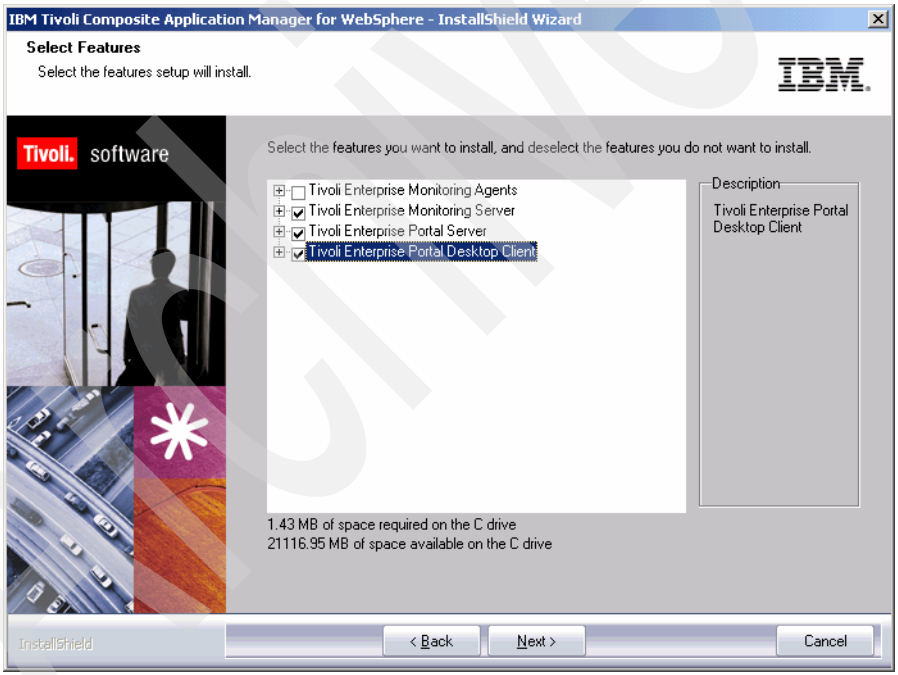


Figure A-6 Tivoli Enterprise Monitoring Agent components

4. Follow the installation wizard and configure the components. Most of the components will already be pre-configured as they are installed on an existing IBM Tivoli Monitoring server.

The communication to Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server in our environment uses IP:PIPE, which represents Transmission Control Protocol (TCP) connection.

Working with Tivoli Enterprise Portal

The information that is presented in the IBM Tivoli Composite Application Manager for WebSphere Web console can also be seen from Tivoli Enterprise Portal. Tivoli Enterprise Portal uses workspaces to show the health of systems. It also has background monitors that are called *situations*. These situations check system attributes and compare them against specified thresholds. When the thresholds are exceeded, the situations fire alerts.

The following are sample workspaces in Tivoli Enterprise Portal:

- “WebSphere summary workspace” on page 134
- “Garbage collection workspace” on page 135
- “Web applications workspace” on page 138

WebSphere summary workspace

When you expand WebSphere Agent in the physical view hierarchy, you can see the WebSphere Application Server workspace. Figure A-7 shows this workspace for our sample environment.

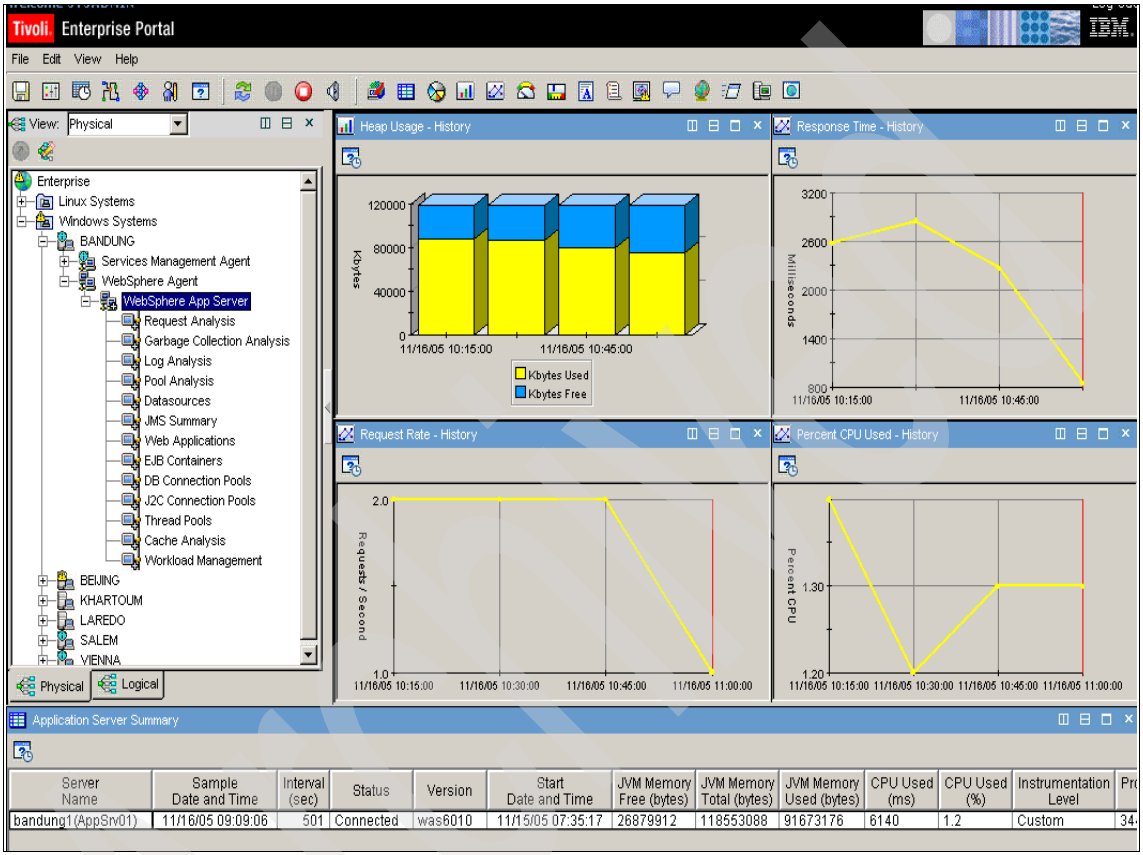


Figure A-7 WebSphere application server workspace

The default workspace shows high-level metrics to establish the health of this WebSphere instance. For example, the Heap Usage - History pane clearly suggests that during the time frame shown, the amount of free space in the heap is not diminishing.

By clicking the icon in the top left corner of the pane, it is possible to change the time frame. In our case, we display the heap usage trend over the last month. This provides support staff or capacity planners with much better information to base their decisions on. The following sections present some of the Tivoli Enterprise views that are available.

Garbage collection workspace

Figure A-8 shows that the WebSphere Application Server garbage collection is performing adequately. The heap usage shows a healthy amount of free memory. The garbage collection is being initiated regularly but is not adversely affecting performance.

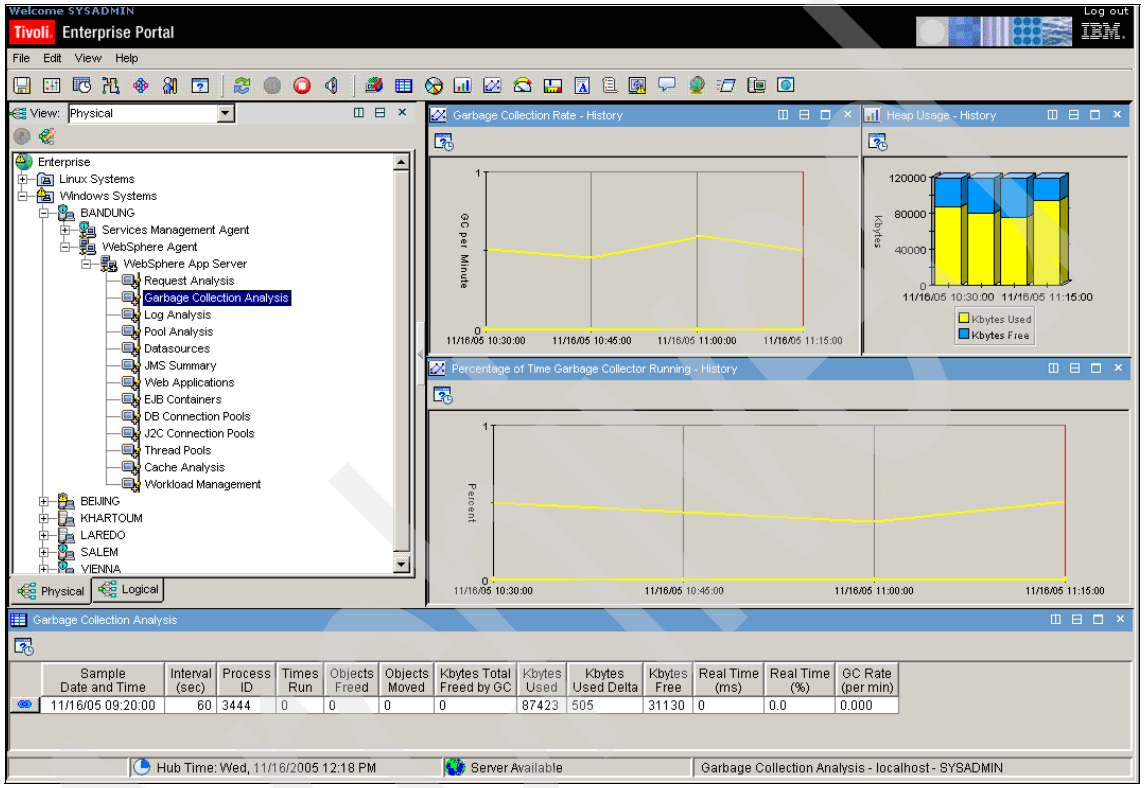


Figure A-8 Garbage collection: Normal view

You have to tune the garbage collection parameters to obtain the optimum balance. Figure A-9 shows an example screen capture where it is forced into action periodically due to poorly scheduled collection or a badly performing application that is forcing the heap too high.

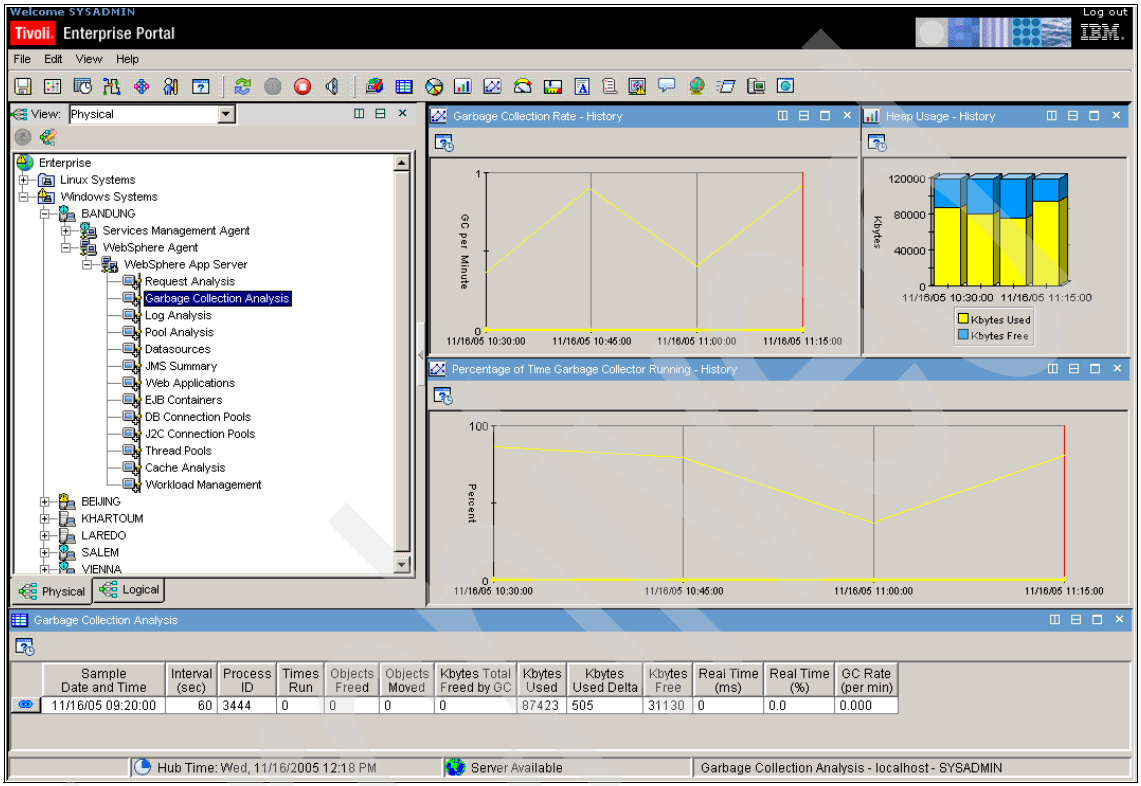


Figure A-9 Garbage collection: Poor performance

The third example shows the other extreme where no force collection is initiated, the schedule collection is almost non-existent, yet the heap is consistently high. Figure A-10 highlights the negative impact to the business service if this situation occurs because the application is always waiting for free memory.

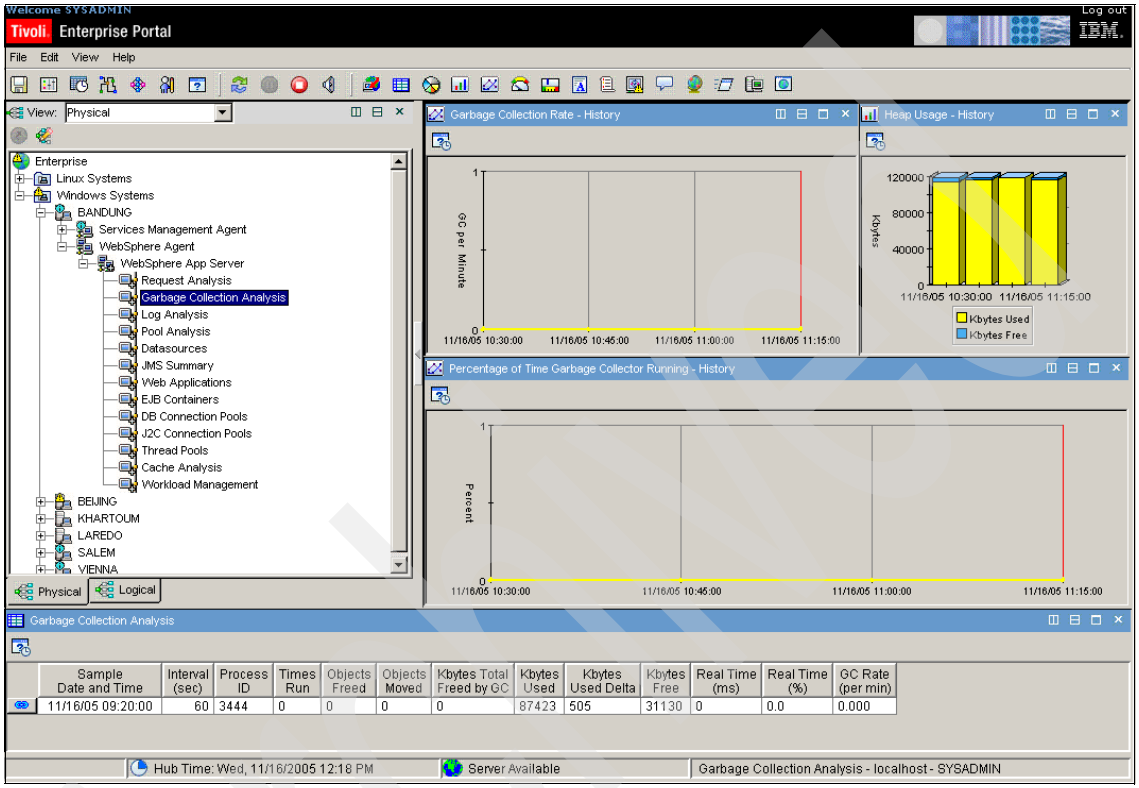


Figure A-10 Garbage collection: High heap

Web applications workspace

The Web applications view provides high-level usage and performance data such as that shown for our Trader application. See Figure A-11.

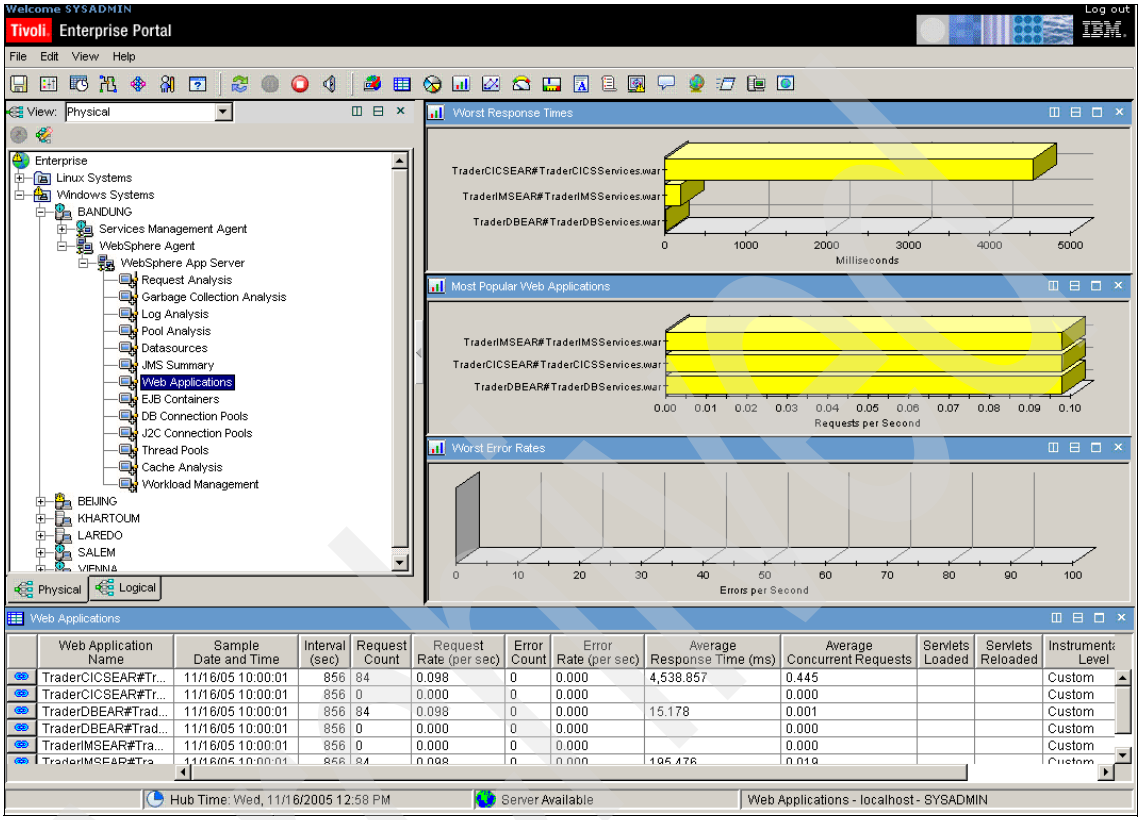


Figure A-11 Tivoli Enterprise Portal: Web application view

Figure A-11 clearly shows that each of the Information Management System (IMS), DB2 and Customer Information Control System (CICS) Web Services calls has roughly the same load. But importantly, the Worst Response Times pane indicates that the CICS calls are substantially slower to complete.

B

Trader application usage

This appendix discusses the sample Loader Java client and Java 2 Platform, Enterprise Edition (J2EE) Trader applications that are used to provide monitoring information for IBM Tivoli Composite Application Manager for WebSphere in this guide. The discussion in this appendix includes:

Trader application

The Trader application runs as an enterprise application on a WebSphere Application Server instance. We install the IBM Tivoli Composite Application Manager for WebSphere data collector to monitor it. The Trader application is a lightweight J2EE application composed of basic Java components. It uses DB2 database tables as its repository. The Trader application consists of two components:

- TraderClient.ear, which acts as the Web-based user interface
- TraderDBSvc.ear, which accesses the database and provides the business logic

These two components can reside either in the same application server instance or on a different one. The communication between these applications is performed using Web Services calls.

Loader application

The Loader application is a small Java client application that provides automated client calls to the Trader application. It is used to generate modifiable transaction request loads on the Trader application. This transaction information is monitored and analyzed using the IBM Tivoli Composite Application Manager for WebSphere Web console.

You can start the loader application by using the java command with sourcing the classpath for loader.jar and loaderaux.jar. You must also have the load.properties file in the current directory. A typical command to run this is:

```
java -cp loader.jar;loaderaux.jar com.ibm.vbd.loader.LauncherMain
```

Using the Loader client

To use the Loader client application, follow this procedure:

1. Verify that the loader Java client has started. A window should have opened when you started the loader client previously. See Figure B-1. Select the **com.ibm.vbd.loader.BasicLoader** item and click **Launch**.

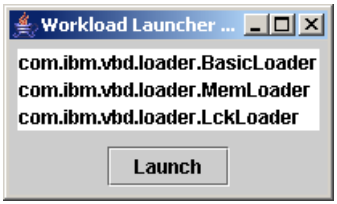


Figure B-1 Loader client launch interface

2. A new window opens, as shown in Figure B-2. Enter the appropriate values in each field:
   a. Target Host: Enter the URL and HTTP port number for the application server instance where the TraderClient application is running. In the lab we used: `salem.itsc.austin.ibm.com:9080`.
   b. Argument: Enter the URL and the HTTP port number for the application server instance where the TraderDBSvc application is running. In the lab we used: `salem.itsc.austin.ibm.com:9081`.
   c. Maximum users: Initially enter the number `10`.
   d. Average think time: Initially enter the number `5`.
   e. Click **Run** to start the Loader client.

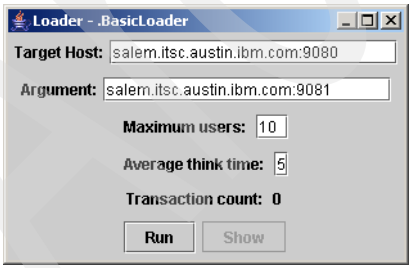


Figure B-2 Loader client configuration interface

The Loader client now makes call requests to the Trader application that is running in the WebSphere Application Server.

Generating SQL calls

The Trader application makes database calls using a container managed persistent (CMP). IBM Tivoli Composite Application Manager for WebSphere monitors and analyzes Structured Query Language (SQL) information that is generated using plain Java Database Connectivity (JDBC) calls.

To generate SQL data for monitoring, open a browser on the system that has the TraderDBSvc application installed on it. Enter a URL similar to the following:

```
http://salem.itsc.austin.ibm.com:9081/TraderDBServices/VBDTradeGetCompanies
```

Replace salem.itsc.austin.ibm.com:9081 with the information that is appropriate for your system.

C

Additional material

This appendix refers to additional material that can be downloaded from the Internet. The discussion in this appendix includes:

- Locating the Web material
- Web material description
- System requirements
- Installing and configuring the Web material

Locating the Web material

The Web material associated with this book is available in softcopy on the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG247252

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG247252.

Web material description

The additional Web material that accompanies this book includes the following files:

File name	*Description*
SG247252.zip	Zipped additional material components
SG247252.readme	Short information regarding the additional material

The zip file contains the following files:

File name	*Description*
TraderClient.ear	Sample client Java 2 Platform, Enterprise Edition (J2EE) application
TraderClientMem.ear	Sample client J2EE application with memory leak
TraderClientLck.ear	Sample client J2EE application with a locking problem
TraderDBSvc.ear	Sample database access J2EE application that accepts Web Services call
loader.jar	Java archive for the loader application
loaderaux.jar	Additional libraries that are required to run the loader application
load.properties	Property file for the loader application
trader.zip	Zipped database extract to build TRADER database

System requirements

The system requirements for both the Trader and Loader applications are extremely small. Because they are hosted by an application server instance of WebSphere Application Server, the system requirements for WebSphere Application Server are sufficient.

For a complete and up-to-date list of the WebSphere Application Server system requirements, refer to the following Web site:

http://www-1.ibm.com/support/docview.wss?rs=180&uid=swg27006921

Installing and configuring the Web material

This section discusses how to install and configure the Web material.

Stage material

Download the SG247252.zip file from the location specified in the previous section. Extract the files into a local directory.

Trader application

To install the Trader application, follow this procedure:

1. Have a machine installed with WebSphere Application Server and DB2 Universal Database. We tested the application on WebSphere Application Server V6.0.1 and DB2 V8.2 Fix Pack 3.
2. Create the TRADER database in DB2. From the DB2 command window, run the following command:

   ```
   db2 create database TRADER
   ```

3. Extract the content of the trader.zip file into a writable directory. Run the following from that directory:

   ```
   db2move TRADER import
   ```

4. Create a Java Database Connectivity (JDBC) connection factory for DB2 Universal JDBC driver in WebSphere. Also create a data source called Trader with Java Naming and Directory Interface (JNDI) name jdbc/Trader.
5. Install the TraderClient.ear and TraderDBSvc.ear into the application server instance to be monitored using IBM Tivoli Composite Application Manager for WebSphere. Use the WebSphere Application Server administrative console to install these applications.

Note: You may want to install TraderClientMem.ear or TraderClientLck.ear to perform memory leak or locking demonstration.

6. Restart the WebSphere Application Server.
7. Verify that the Trader application is installed and running using the WebSphere Application Server administrative console.

Loader client application

To install the Loader client application, follow this procedure:

1. Store the files loader.jar, loaderaux.jar, and load.properties in the same directory.
2. Find a suitable Java Runtime Environment (JRE™) of Version 1.3 or later versions. You can use the JRE provided by WebSphere Application Server.
3. From the directory where the files are located, create a batch file or shell script that runs the command similar to the following:

```
java -cp loader.jar;loaderaux.jar com.ibm.vbd.loader.LauncherMain
```

Note: The path separator for UNIX-based platforms is a colon (:). On Windows it is a semi-colon (;).

4. Save the batch file or shell script.

Abbreviations and acronyms

AIX	Advanced Interactive Executive
APPC	Advanced Program to Program Communication
BCM	Byte Code Modification
BTAM	Basic Telecommunication Access Method
CD-ROM	compact disc read-only memory
CICS	Customer Information Control System
CMDB	Configuration Management Database
CPU	central processing unit
CRM	Customer Relationship Management
CTG	CICS Transaction Gateway
DB2	Database 2™
EJB	Enterprise JavaBeans
ESE	Enterprise Server Edition
GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
IBM	International Business Machines Corp.
IMS	Information Management System
ITIL	Information Technology Infrastructure Library
ITSO	International Technical Support Organization
J2C	Java 2 Connectivity
J2EE	Java 2 Platform, Enterprise Edition
JCA	Java Connector Architecture
JDBC	Java Database Connectivity
JMX	Java Management Extension
JSP	JavaServer Pages
JVM	Java Virtual Machine
JVMTI	Java Virtual Machine Tool Interface
MVS	Multiple Virtual Storage
ORS	Original Request String
OTMA	Open Transaction Manager Access
PDF	Portable Document Format
PMI	Performance Management Interface
RHEL	Red Hat Enterprise Linux
SLES	SUSE Linux Enterprise Server
SMF	System Measurement Facility
SNMP	Simple Network Management Protocol
SOA	service-oriented architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SSL	Secure Socket Layer
TCP/IP	Transmission Control Protocol Internet Protocol
TEP	Tivoli Enterprise Portal
UDB	Universal Database
URI	Uniform Resource Identifier
URL	Universal Resource Locator
VTAM	Virtual Telecommunication Access Method
WBI	WebSphere Business Integration

WSAM	WebSphere Studio Application Monitor
XML	eXtensible Markup Language

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 151. Note that some of the documents referenced here may be available in softcopy only.

- *Installing WebSphere Studio Application Monitor V3.1*, SG24-6491
- *WebSphere Studio Application Monitor V3.2 Advanced Usage Guide*, SG24-6764
- *Implementing IBM Tivoli OMEGAMON XE for WebSphere Business Integration V1.1*, SG24-6768
- *Getting Started with IBM Tivoli Monitoring 6.1 on Distributed Environments*, SG24-7143
- *IBM Tivoli Composite Application Manager V6.0 Family: Installation, Configuration, and Basic Usage*, SG24-7151
- *Deployment Guide Series: IBM Tivoli Monitoring Express Version 6.1*, SG24-7217
- *Large Scale Implementation for IBM Tivoli Composite Application Manager for WebSphere*, REDP-4162

Other publications

These publications are also relevant as further information sources:

- IBM Tivoli Composite Application Manager for WebSphere publications
  - *IBM Tivoli Composite Application Manager for WebSphere Installation and Customization Guide*, GC32-9506
  - *IBM Tivoli Composite Application Manager for WebSphere: Tivoli Enterprise Monitoring Agent Problem Determination Guide*, SC32-1800
  - *IBM Tivoli Composite Application Manager for WebSphere: Installing and Configuring the Tivoli Enterprise Monitoring Agent*, SC32-1801

- *IBM Tivoli Composite Application Manager for WebSphere User's Guide*, SC32-9507
- *IBM Tivoli Composite Application Manager for WebSphere Operator's Guide*, SC32-9508
- *IBM Tivoli Composite Application Manager for WebSphere Problem Determination Guide*, SC32-9509

- IBM Tivoli Monitoring V6.1 manuals
  - *IBM Tivoli Monitoring: Upgrading from Tivoli Distributed Monitoring*, GC32-9462
  - *Introducing IBM Tivoli Monitoring*, GI11-4071
  - *Exploring IBM Tivoli Monitoring*, SC32-1803
  - *IBM Tivoli Monitoring Installation and Setup Guide*, GC32-9407
  - *IBM Tivoli Monitoring Administrator's Guide*, SC32-9408
  - *IBM Tivoli Monitoring User's Guide*, SC32-9409
  - *IBM Tivoli Monitoring Problem Determination Guide*, GC32-9458
  - *IBM Tivoli Universal Agent User's Guide*, SC32-9459
  - *IBM Tivoli Universal Agent API and Command Programming Reference Guide*, SC32-9461
  - *Configuring IBM Tivoli Enterprise Monitoring Server on z/OS*, SC32-9463

Online resources

These Web sites and URLs are also relevant as further information sources:

- Product online documentation

  http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?toc=/com.ibm.itcamwas.doc/toc.xml

- Product prerequisites information

  http://publib.boulder.ibm.com/tividd/td/ITCAMWAS/prereq60/en_US/HTML/itcam6.html

- Product Web page

  http://www-306.ibm.com/software/tivoli/products/composite-application-mgr-websphere/

- IBM Tivoli Composite Application Manager for WebSphere product support link

  http://www-306.ibm.com/software/sysmgmt/products/support/IBMTivoliCompositeApplicationManagerforWebSphere.html

- WebSphere fix packs links
  - ftp://ftp.software.ibm.com/software/websphere/appserv/support/fixpacks/was60/refreshpack1/Windows
  - ftp://ftp.software.ibm.com/software/websphere/appserv/support/fixpacks/was60/refreshpack1/Linux
- Microsoft Help and Support

  http://support.microsoft.com/default.aspx?scid=kb;en-us;899522

- Microsoft Windows Services for UNIX

  http://www.microsoft.com/windows/sfu/default.asp

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

E

F

G

H

I

J

K

L

T

U

V

W

X

IBM Redbooks

Deployment Guide Series: IBM Tivoli Composite Application Manager

(0.2”spine)
0.17”<->0.473”
90<->249 pages

Deployment Guide Series
IBM Tivoli Composite Application Manager for WebSphere V6.0

Analyzing performance of WebSphere-based application

Providing debugging and problem determination information

Dynamically adjusting monitoring level

This deployment guide helps you to plan the implementation, installation, and initial usage of IBM Tivoli Composite Application Manager for WebSphere V6.0. The objective of this IBM Redbook is to use a basic configuration of the product for an initial user to successfully install the product, become familiar with the various working components and how they interoperate. The user can understand some of the product capabilities, and begin to use the product to monitor and manage applications on the IBM WebSphere Application Server.

In addition to the installation steps, sample scenarios are included to highlight some of the ways in which you can use this product to increase the performance and availability of applications running on WebSphere Application Server. As a result, you can use this deployment guide to support an effective proof of concept demonstration of IBM Tivoli Composite Application Manager for WebSphere V6.0.

The instructions contained in this deployment guide target a Windows platform implementation. We do not cover other operating systems such as z/OS-based data collectors.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-7252-00 ISBN 0738495921