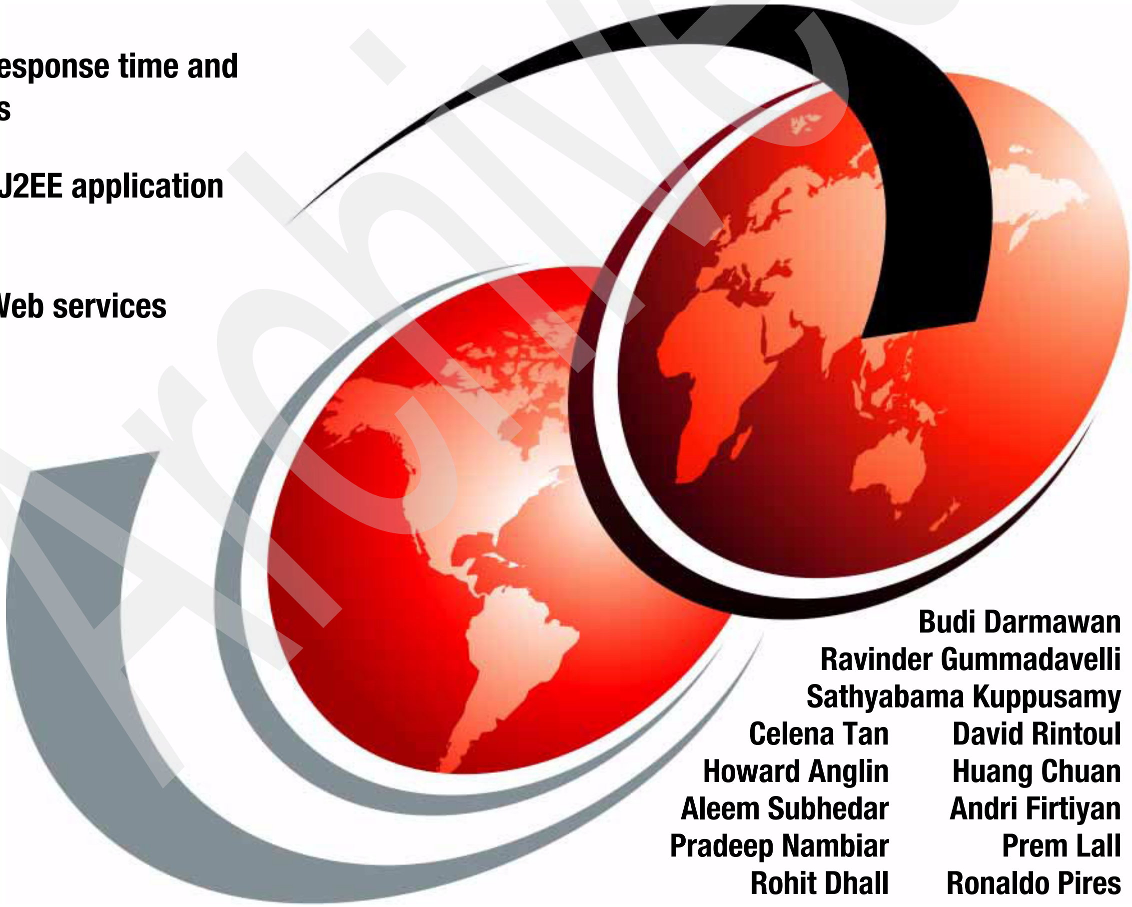
Tivoli software

IBM

IBM Tivoli Composite Application Manager Family

Installation, Configuration, and Basic Usage



- Managing response time and transactions
- Monitoring J2EE application servers
- Collecting Web services information

Budi Darmawan
Ravinder Gummadavelli
Sathyabama Kuppusamy
Celena Tan
David Rintoul
Howard Anglin
Huang Chuan
Aleem Subhedar
Andri Firtiyan
Pradeep Nambiar
Prem Lall
Rohit Dhall
Ronaldo Pires

Redbooks

ibm.com/redbooks

International Technical Support Organization

IBM Tivoli Composite Application Manager Family: Installation, Configuration, and Basic Usage

January 2008

SG24-7151-02

Note: Before using this information and the product it supports, read the information in "Notices" on page xi.

Third Edition (January 2008)

This edition applies to Version 6 of IBM Tivoli Composite Application Manager products:

- IBM Tivoli Composite Application Manager for CICS Transactions, 5698-A69
- IBM Tivoli Composite Application Manager for IMS Transactions, 5698-A70
- IBM Tivoli Composite Application Manager for WebSphere (z/OS), 5698-A71
- IBM Tivoli Composite Application Manager for WebSphere, 5724-L62
- IBM Tivoli Composite Application Manager for Web Resources, 5724-S32
- IBM Tivoli Composite Application Manager for J2EE, 5724-N95
- IBM Tivoli Composite Application Manager for Response Time Tracking (z/OS), 5698-A75
- IBM Tivoli Composite Application Manager for Response Time Tracking, 5724-L99
- IBM Tivoli Composite Application Manager for Response Time, 5724-C04
- IBM Tivoli Composite Application Manager for SOA (z/OS), 5698-A77
- IBM Tivoli Composite Application Manager for SOA (Distributed), 5724-M07
- IBM Tivoli Composite Application Manager for Internet Service Monitoring, 5724-Q22

Note: This book is based on a pre-GA version of a product and may not apply when the product becomes generally available. We recommend that you consult the product documentation or follow-on versions of this book for more current information.

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 illustrate 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.

Trademarks

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

Redbooks (logo) ®
pSeries®
z/OS®
zSeries®
AIX 5L™
AIX®
Candle®
CICS®
DataPower®
DB2 Universal Database™
DB2®
ETE™
ETEWatch®
IBM®
IMS™
Lotus Notes®
Lotus®
Micromuse®
Monitoring On Demand®
MVS™
Netcool/Internet Service Monitors®
Netcool®
NetView®
Notes®
OMEGAMON®
Rational®
Redbooks®
RACF®
Tivoli®
WebSphere®
Workplace™

The following terms are trademarks of other companies:

SAP NetWeaver, SAP, and SAP logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries.

Oracle, JD Edwards, PeopleSoft, Siebel, and TopLink are registered trademarks of Oracle Corporation and/or its affiliates.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

Enterprise JavaBeans, EJB, Java, Java Web Server, JavaBeans, JDBC, JDK, JMX, JNI, JRE, JSP, JVM, J2EE, J2SE, Solaris, Sun, Sun Java, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Internet Explorer, Microsoft, Outlook, SQL Server, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Pentium, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries 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 IBM® Redbooks® publication describes the IBM Tivoli® Composite Application Manager (ITCAM) family of products. The Composite Application Manager family currently consists of:

- IBM Tivoli Composite Application Manager for WebSphere®
- IBM Tivoli Composite Application Manager for J2EE™
- IBM Tivoli Composite Application Manager for Web Resources
- IBM Tivoli Composite Application Manager for Response Time Tracking
- IBM Tivoli Composite Application Manager for Response Time
- IBM Tivoli Composite Application Manager for SOA
- IBM Tivoli Composite Application Manager for Internet Service Monitoring
- IBM Tivoli Composite Application Manager for CICS® Transactions
- IBM Tivoli Composite Application Manager for IMS™ Transactions

The aim of IBM Tivoli Composite Application Manager is to simplify and enhance distributed application management. Application components can reside on multiple servers, across different platforms and J2EE environments, even through mainframes. The complexity of understanding and solving application-related problems, typically around performance issues, requires a cohesive set of tools to provide an end-to-end view of the application.

The discussion that we provide in this book mainly consists of the basic installation and configuration of the products. The discussion is aimed at augmenting the information provided in the product manuals. Consult the appropriate product manual before starting to implement these products.

We provide usage scenarios in this book as a demonstration of using these products in our sample environment. We use a single application environment that can be managed by all of the products.

The team that wrote this book

This book was produced by a team of specialists from around the world working at the International Technical Support Organization, Austin Center.

Figure 1 Back: Howard, Sathyabama, Rohit, Celena, David, Ravinder; front: Huang, Ronaldo, Budi, Aleem and Andri

Budi Darmawan is a Project Leader at the International Technical Support Organization, Austin Center. He writes extensively and teaches IBM classes worldwide on all areas of systems management, primarily application management, business service management, and workload scheduling. Before joining the ITSO in 1999, Budi worked in IBM Indonesia as Lead Implementor and Solution Architect. His current interests are J2EE and SOA application management, z/OS® integration, and business service management.

Aleem Subhedar is a staff Software Engineer with India Software Labs in Pune, India. He has seven years of experience in AIX® and Middleware System Administration. He holds a degree in chemistry from Pune University. His areas of expertise include AIX, pSeries®, and related system technologies. He is an IBM Certified System Expert. His areas of interest include pSeries Virtualization and High Availiability.

Andri Firtiyan is a Solution Architect at Software Lab Services based in IBM Indonesia. He has 11 years of experience in IT. He holds a master's degree in Computer Science from the University of Indonesia. He was involved in the first SOA Offering Roadmap project in the ASEAN region. He is an IBM Certified SOA Solution Designer. His current focus in Tivoli Best Practice, Tivoli Business Automation, and service-oriented architecture.

Celena Tan is a Managing Consultant with IBM Software Group Services in Australia. She has 14 years of experience in the IT field. She holds a master's of technology from the National University of Singapore and a bachelor's of electrical engineering (honors) from the University of Tasmania. Her areas of expertise include ITCAM family products and Rational® Testing, Change and Configuration Management products.

David Rintoul is a Senior IT Specialist who works as part of the TechWorks group in AP SWG. He has over 20 years of experience in the IT field. He holds a degree in mathematics from Newcastle University. His areas of expertise include IBM Service Management products, the ITCAM family of products, and Tivoli zSeries® products.

Howard Anglin is a Deployment Expert for ITCAM for WebSphere, Response Time Tracking, and IBM Tivoli Monitoring in the United States. He has worked with various large customers and in his role as an IT Specialist he has resolved deployment, integration, and performance issues. He has nine years of experience in the software test and development field with emphasis on the WebSphere Application Server. He holds a Bachelor of Science in Electrical Engineering from Manhattan College, Riverdale, New York. Howard began his career at IBM in the pSeries Hardware Group as a test engineer developing automation solutions for the production line. He then transferred to the software group.

Huang Chuan is a senior test lead of IBM China CSDL lab. He has five years of experience in software developing and over six years of experience in software product testing. He has led the ITCAM for Response Time Tracking test project for several releases. He holds a degree in computer science from the University of Electronic Science and Technology of China.

Prem Lall is a Software Engineer currently assigned to the ITCAM for SOA project where he specializes in the field of Web services management. He has had over 15 years of experience in the IT field. During his 11 years at IBM, he has helped design and implement a variety of software products. He has expertise in front-end, middleware, and back-end development with an emphasis on e-commerce. Among other things, he created end-to-end online banking solutions for IBM customers in the Integrion consortium, he has been part of the WebSphere Application Server development team, and he has helped create an extensive SOA-based e-File application for the IRS that is currently used by numerous businesses across the country. He holds a Master of Science degree in Pure and Applied Mathematics from California State University, Northridge. He also worked as an Actuary, and in the Atmospheric Physics Division of the NASA Goddard Space Flight Center.

Pradeep Nambiar works in the IBM Tivoli Business Automation Sales Enablement group. He has over 19 years of experience in the IT industry in various areas ranging from graphics systems, networked graphics, IBM Component Broker/WebSphere Application Server system management, business application architecture, design, and development. He is an IBM Certified SOA Solution Designer, IBM Certified WebSphere Enterprise Developer, and IBM Certified Solution Developer in XML and related technologies. His current focus is on application management and automation products including SOA management from IBM Tivoli. He is based in Austin, TX.

Ravinder Gummadavelli is a Software Engineer with IBM Systems Technology Group in the USA. He has over 10 years of experience in the IT systems design and development field. He holds a Masters of Technology degree in Electrical Engineering from REC, Warangal, India, and an Master of Science in Electrical Engineering from Auburn University, AL, USA. His areas of expertise include systems design, development and service-oriented architecture. His current interests include SOA and IBM Virtualization offerings.

Rohit Dhall is an IT Architect with GBS, IBM India. He has 10 years of IT experience in technologies like client-server computing, Web-based transactional systems, data warehousing, and data mining. His major expertise is in designing, implementating, and tuning large-scale Internet banking, eMortgage, and anti-money laundering solutions for the banking and financial sector. He is EXIN ITIL® certified and also holds certification in Java™ and EJB™ from Brainbench. His current interests include SOA and IBM Virtualization offerings.

Ronaldo Pires is an IBM IT Specialist. He joined IBM in 2004 and has been working on Global Technology Services Delivery in São Paulo, Brazil, supporting the systems management infrastructure for IBM outsourcing customers. His skills include IBM Tivoli Framework, IBM Tivoli Monitoring, IBM Tivoli Storage Manager, IBM Tivoli Identity Manager, Altiris Client Management Suite, BMC Control-M for z/OS, and BMC Control-D for z/OS. He has a Bachelor of Mathematics from Faculdade de Filosofia Ciências e Letras de Santo André. He is a Tivoli Certified Consultant for Tivoli Storage Manager and IBM Certified Deployment Professional for Tivoli Monitoring V5.1.2.

Sathyabama Kuppusamy is a Technical Lead with the Global Business Solution Center in India. Currently, she works at the SOA Solution Center for SOA-based products. She has six years of experience in SOA, testing, and the middleware system. She has a Bachelor’s of Engineering degree in Computer

Science from University of Madras, India, and also holds a degree in Management of Business Administration and Finance from University of Madras, India. Her areas of expertise include SOA, the insurance domain, middleware systems, and automation testing.

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

Bob Haimowitz, Richard M. Conway
International Technical Support Organization

Rebecca Poole, Adrian Mitu, Erik Klein, Michael Bacon, Greg Bowman
IBM Software Group

Danilo Rezende Araujo, Christian Minarovits, Kimberly Nguyen-Theunissen
Authors of the first edition of *IBM Tivoli Composite Application Manager V6.0 Family*, SG24-7151-00

Sören Eickhoff, Michael Stahl, Lee Hwa Yong
Authors of the second edition of *IBM Tivoli Composite Application Manager V6.1 Family Installation, Configuration, and Basic Usage*, SG24-7151

Become a published author

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

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

Learn 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 book form found at:

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. 0SJB Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Summary of changes

This section describes the technical changes made in this edition of the book and in previous editions. This edition may also include minor corrections and editorial changes that are not identified.

Summary of changes
for SG24-7151-02
for *IBM Tivoli Composite Application Manager Family: Installation, Configuration, and Basic Usage*

January 2008, Third Edition

This revision reflects the addition, deletion, or modification of new and changed information described here.

New information

- Included IBM Tivoli Composite Application Manager for Web Resources, IBM Tivoli Composite Application Manager for Response Time:
  - Chapter 4, "ITCAM for Web Resources concepts, installation, usage" on page 155
  - Chapter 7, "ITCAM for Response Time concepts, installation, usage" on page 333
- Added information for Rational Performance Tester usage with ITCAM for Response Time Tracking
- Added information regarding Tivoli Common Object Repository and Discovery Library Adapter for ITCAM for SOA

Changed information

- Various changes in figures and product capabilities related to new version
- Changes in Preface regarding authors
- Changes in Chapter 1, "IBM Tivoli Composite Application Manager overview" on page 1: added and changed topics:
  - 1.2.3, "ITCAM for Web Resources" on page 6
  - 1.2.5, "ITCAM for Response Time" on page 8
  - 1.3, "Project environment" on page 10
  - 1.4, "Document organization" on page 12

- Changes in Chapter 3, “ITCAM for WebSphere and ITCAM for J2EE usage” on page 65: new topics
  - 3.7, “Memory diagnosis” on page 96
  - 3.8, “Lock analysis” on page 108
  - 3.4, “Method profiling” on page 74
  - 3.5, “Portal performance reports” on page 79
- Changes in Chapter 6, “ITCAM for Response Time Tracking usage” on page 283: added topics
  - 6.9, “Rational Performance Tester” on page 320
- Changes in Chapter 8, “ITCAM for Internet Service Monitoring” on page 413: changed topics:
  - 8.3, “Installing ITCAM for Internet Service Monitoring” on page 416
  - 8.4, “Configuration and usage” on page 429
- Changes in Chapter 9, “ITCAM for SOA” on page 449: added topic
  - 9.4, “Discovery Library Adapters in SOA management” on page 506
- Appendix A, “Trader application description” on page 619, revamped with new information

April 2007, Second Edition

This revision reflects the addition, deletion, or modification of new and changed information described here.

New information

- Included IBM Tivoli Composite Application Manager for J2EE, IBM Tivoli Composite Application Manager for Internet Service Monitoring, and IBM Tivoli Composite Application Manager for J2EE Operations.
- Streamlined chapter structure. Instead of explaining each product in different parts, each chapter covered a different product.

Changed information

- The book covered Version 6.1 of the IBM Tivoli Composite Application Manager family.
- Various changes in figures and product capabilities related to new version.

1

IBM Tivoli Composite Application Manager overview

This chapter introduces the IBM Tivoli Composite Application Manager product family. These products are aimed at managing applications instead of the more traditional resource management approach. Application management requires management to understand the application behavior instead of looking for just the resources that the application uses. We divide this discussion into:

1.1 Application management background and issues

Nowadays, businesses require computer-based applications. Business processes are highly dependent on the applications they use. Without the applications, the business processes can no longer function. The traditional paper-based manual processes have been completely replaced by computer-based applications.

Because the *application's health* is now almost synonymous with the business process health, managing the application to optimize its health is critical for the overall business survival. Application health can be viewed as a multifaceted issues. The health can consist of:

- Availability: The application must be available to be used.
- Performance: The application must perform in a reasonable time.
- Integrity: The application must handle the information entered correctly.
- Reliability: The application must be able to recover the data that it has.

From this list, integrity and reliability are typically handled inside the application, which uses several redundant storage and commit mechanisms to achieve these. Sometimes, an offline backup process is added to manage data reliability.

The availability and performance of the application, however, depend on many of the components that support the application. This is critical for a major application that consists of multiple, different, interconnected components.

A typical e-business distributed application can have the components spread over several clustered application servers that are interconnected using several different mechanisms. These distributed interconnected applications are referred collectively as *composite applications*. Figure 1-1 shows a sample composite application.

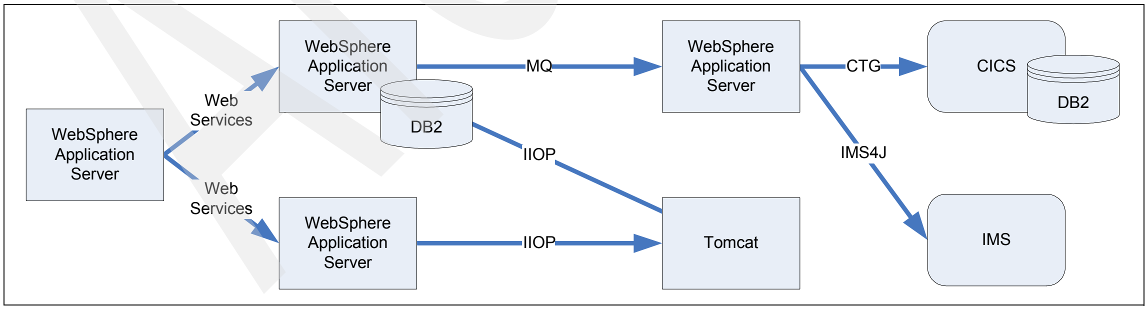


Figure 1-1 Composite application

Managing a composite application, as shown in Figure 1-1 on page 2, requires management of both the underlying resources and an understanding of how the components interact with each other. Understanding that the application is performing poorly from a user view does not necessarily means that the user-interaction application server has a problem; a back-end server might be suffering from a lack of resources.

Composite application management aims to be able to understand these relationships and present the *root cause* of the application problem. This includes decomposing the application and understanding the individual component resource needs to be able to pinpoint resource problems on an application context.

The IBM Tivoli Composite Application Manager family addresses the composite application management. These products address different components and decompose transactions to get to the root cause of the problem.

1.2 IBM Tivoli Composite Application Manager

IBM Tivoli Composite Application Manager is a family of products that is part of the application management building block of the IBM Tivoli portfolio. The system management portfolio of IBM Tivoli consists of a set of systems management suites to manage your entire *IT infrastructure*.

The IBM Tivoli product suite is in line with the Information Technology Infrastructure Library (ITIL) specification. With the launch of the IBM Tivoli Service Management suite of products, the overall IBM Tivoli portfolio is conceptually similar to the illustration in Figure 1-2.

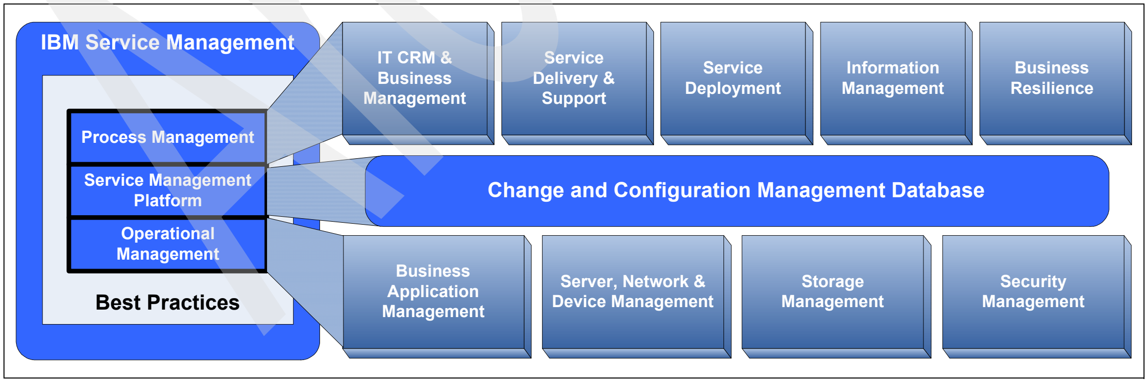


Figure 1-2 IBM Tivoli product portfolio

In Figure 1-2 on page 3, the application management product resides in the *availability management* discipline. For more information about other Tivoli product suites, visit the Tivoli home page at:

http://www.ibm.com/tivoli

The application management product suite consists of the following products:

- IBM Tivoli Composite Application Manager for WebSphere
- IBM Tivoli Composite Application Manager for J2EE
- IBM Tivoli Composite Application Manager for Web Resource
- IBM Tivoli Composite Application Manager for Response Time Tracking
- IBM Tivoli Composite Application Manager for Response Time
- IBM Tivoli Composite Application Manager for Internet Service Monitoring
- IBM Tivoli Composite Application Manager for SOA
- IBM Tivoli Composite Application Manager for CICS Transactions
- IBM Tivoli Composite Application Manager for IMS Transactions
- OMEGAMON® XE for Messaging

We discuss the individual products in the subsequent sections.

1.2.1 ITCAM for WebSphere

IBM Tivoli Composite Application Manager for WebSphere (ITCAM for WebSphere) is an evolution from WebSphere Studio Application Monitor and OMEGAMON XE for WebSphere Application Server.

ITCAM for WebSphere enables you to analyze the health of the WebSphere Application Server and the transactions that are invoked in it. It is able to trace the transaction execution to the detailed method-level information, and connects transactions that spawn from one application server and invokes services from other application servers, including mainframe applications in IMS or CICS.

ITCAM for WebSphere provides a flexible level of monitoring, from an non-intrusive production ready monitor, to a detailed deep-dive tracing for problems of locking or even memory leaks. ITCAM for WebSphere provides a separate interactive Web console and also allows monitoring data to be displayed on the Tivoli Enterprise Portal.

The current ITCAM for WebSphere Version 6.1 provides additional functions, such as:

- Integration with IBM Tivoli Service Manager by providing a Web services interface to get health status
- Improved memory leak and locking analysis pages
- Problem determination enhancements

- Advanced visualization, aggregation, persistence, and correlation of performance metrics in Tivoli Enterprise Portal
- Additional WebSphere server platform support, including WebSphere Portal Server and WebSphere Process Server
- Enhanced composite transaction tracing and decomposition
- Web session browser to help diagnose session-related problems

For more information, see the ITCAM for WebSphere page at:

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

1.2.2 ITCAM for J2EE

IBM Tivoli Composite Application Manager for J2EE (ITCAM for J2EE) is a complementary solution for monitoring and managing applications on a non-WebSphere Application Server-based Java 2 Platform, Enterprise Edition (J2EE) container. This includes support for the WebSphere Application Client J2EE container and non-IBM application servers, such as:

- SAP® NetWeaver
- Oracle® 9i/10g Application Server
- JBoss Application Server
- Apache Tomcat
- BEA WebLogic Server
- WebSphere Application Server CE
- J2SE™
- Weblogic Portal Server (from FP1)
- Sun™ Java™ System Application Server Enterprise Edition (from FP2 with IF001)

ITCAM for J2EE uses the same interface and technology as ITCAM for WebSphere. It handles application server-specific interfaces as additional metrics that are collected from the application servers.

Additional supports are available to monitor Web servers using Tivoli Enterprise Monitoring Agent for Web Servers, supporting:

- Apache Web Server
- Sun Java System Web Server
- Microsoft® Internet Information Server

ITCAM for J2EE enables you to:

- View all in-flight J2EE transactions, including composite transactions.
- Evaluate common performance bottlenecks and contributing factors with an automated problem finder to help detect, categorize, and analyze root causes easily.
- Analyze problematic transactions both historically and in real time, drill down into the details, and share the information with other stakeholders using built-in, interactive reporting tools that preserve some problem context.
- Correlate and profile transactions across multiple subsystems to determine the precise location and root causes of application failures.
- Set traps and alerts to detect and fix potentially troublesome situations before they affect users.
- Analyze resource consumption patterns, perform trends or historical analysis, and plan for future growth.

For more information about ITCAM for J2EE, see:

http://www.ibm.com/software/tivoli/products/composite-application-mgr-itcam-j2ee/

1.2.3 ITCAM for Web Resources

The IBM Tivoli Composite Application Manager for Web Resources is developed from ITCAM for WebSphere and ITCAM for J2EE to provide a quick-to-value solution for resource monitoring for a WebSphere and other J2EE-based Application Servers.

ITCAM for Web Resources uses data collectors from ITCAM for WebSphere and ITCAM for J2EE to collect information, and relays the monitoring result to an IBM Tivoli Monitoring infrastructure to be displayed on a Tivoli Enterprise Portal.

For more information, see the ITCAM for Web Resources page at:

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

1.2.4 ITCAM for Response Time Tracking

IBM Tivoli Composite Application Manager for Response Time Tracking (ITCAM for Response Time Tracking) is an evolution from IBM Tivoli Monitoring for Transaction Performance, Candle® End-to-end Watch, and Web Response Monitor.

ITCAM for Response Time Tracking allows monitoring and analysis of application transaction response time. It provides statistics of response times using instrumentation and robotic means. ITCAM for Response Time Tracking enables you to analyze and break down response time into individual components to quickly pinpoint a response time problem.

ITCAM for Response Time Tracking can decompose transactions from robotic means simulating users, tracking its execution in J2EE application servers all the way to the IMS or CICS back end. The response time information is presented on the Web management console or Tivoli Enterprise Portal.

ITCAM for Response Time Tracking:

- Proactively recognizes, isolates, and resolves transaction performance problems using robotic and real-time techniques
- Enables you to drill down each of the transaction's steps across multiple systems and measure each transaction component's contribution to overall response time
- Delivers application topology information to the CCMDB and monitoring status to the Availability Process Manager
- Integrates Web Response Monitor for real user response-time analysis
- Provides custom reporting using Tivoli Enterprise Portal or direct SQL queries of database views, and organizes reports by application, customer, and location
- Enables you to easily reuse Mercury LoadRunner scripts
- Offers new tracking support for business processes written in Business Process Execution Language (BPEL) running in WebSphere Process Server; portals in WebSphere Portal Server; J2EE applications on JBoss and Tomcat; and 64-bit applications, including 64-bit editions of WebSphere Application Server

For more information about ITCAM for Response Time Tracking, see:

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

1.2.5 ITCAM for Response Time

IBM Tivoli Composite Application Manager for Response Time is an evolution of the user response time monitoring part of ITCAM for Response Time Tracking. Its interfaces has been revamped to completely integrate with Tivoli Enterprise Portal. It allows easy implementation of user response time monitoring from the following interfaces:

- Web-based transaction monitors
- Customized client GUI monitors
- Robotic monitors

For more information about ITCAM for Response Time, see:

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

1.2.6 ITCAM for Internet Service Monitoring

IBM Tivoli Composite Application Manager for Internet Service Monitoring (ITCAM for Internet Service Monitoring) is based on Netcool® Internet Service Monitor. It enables monitoring of TCP/IP-based services and transactions using a robotic probe to check whether the service is available.

ITCAM for Internet Service Monitoring provides a Web-based administration console. It also plugs in seamlessly to Netcool System Service Monitors.

For more information about ITCAM for Internet Service Monitoring, see:

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

1.2.7 ITCAM for SOA

IBM Tivoli Composite Application Manager for SOA (ITCAM for SOA) is a product based on the IBM Tivoli Monitoring V6.1. It monitors, manages, and controls Web services and service-oriented architectures (SOAs) deployed using a wide range of IBM and third-party systems.

ITCAM for SOA:

- Recognizes and quickly isolates Web service performance problems, alerts you when Web service performance is degraded,and reports results against committed service levels
- Provides an integrated, easy-to-use console that helps you visualize the flows of Web services in their entirety

- Monitors your services where you want them with heterogeneous platform support
- Views by service requestor are now supported, reports on number of requests or response time by requestor

ITCAM for SOA includes the Web Services Navigator, a plug-in to IBM Rational and other Eclipse-based tools, which provides a deep understanding of service flows, patterns, and relationships to developers and architects using operational data from the Tivoli Data Warehouse or monitoring log files.

ITCAM for SOA is a core component of the IBM SOA Foundation Management Essentials, an integrated and open set of software, best practices, patterns, and skill resources to get you started with service-oriented architectures

ITCAM for SOA Version 6.1 with fix pack 1 supports:

- WebSphere Application Server
- WebSphere Enterprise Service Bus
- WebSphere Process Server
- WebSphere Message Broker
- WebSphere DataPower® appliance
- BEA WebLogic
- AXIS 1.2
- Microsoft Internet Information Server
- CICS Transaction Server
- WebSphere CE
- SAP NetWeaver®
- JBoss Application Server

For more information about ITCAM for SOA, see:

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

1.2.8 ITCAM for CICS Transactions

IBM Tivoli Composite Application Manager for CICS Transactions (ITCAM for CICS Transactions) is the data collector for ITCAM for Response Time Tracking and ITCAM for WebSphere for transactions that invoke CICS transactions in the back end. It provides CICS response time information and other performance metrics. This is not a stand-alone product. It has to run with either ITCAM for Response Time Tracking or ITCAM for WebSphere.

1.2.9 ITCAM for IMS Transactions

IBM Tivoli Composite Application Manager for IMS Transactions (ITCAM for IMS Transactions) is the data collector for ITCAM for Response Time Tracking and ITCAM for WebSphere for transactions that invoke IMS transactions in the back end. It provides IMS response time information and other performance metrics. This is not a stand-alone product. It has to run with either ITCAM for Response Time Tracking or ITCAM for WebSphere.

1.2.10 OMEGAMON XE for Messaging

OMEGAMON XE for Messaging, formerly OMEGAMON XE for WebSphere Business Integration, is a powerful messaging middleware performance and configuration management tool. It allows monitoring and management of WebSphere MQ-based middleware environments.

For more information about OMEGAMON XE for Messaging, refer to:

http://www.ibm.com/software/tivoli/products/omegamon-xe-messaging-dist-sys/
http://www.ibm.com/software/tivoli/products/omegamon-xe-messaging-zos/

We do not discuss this product in this book. Refer to *Implementing OMEGAMON XE for Messaging V6.0*, SG24-7357.

1.3 Project environment

We perform this ITCAM family project at the ITSO, Austin Center. We install the products and use them to manage a set of servers with a single distributed application called ITSOTrader.

1.3.1 Machine environment

Figure 1-3 illustrates the machines that we use in our environment.

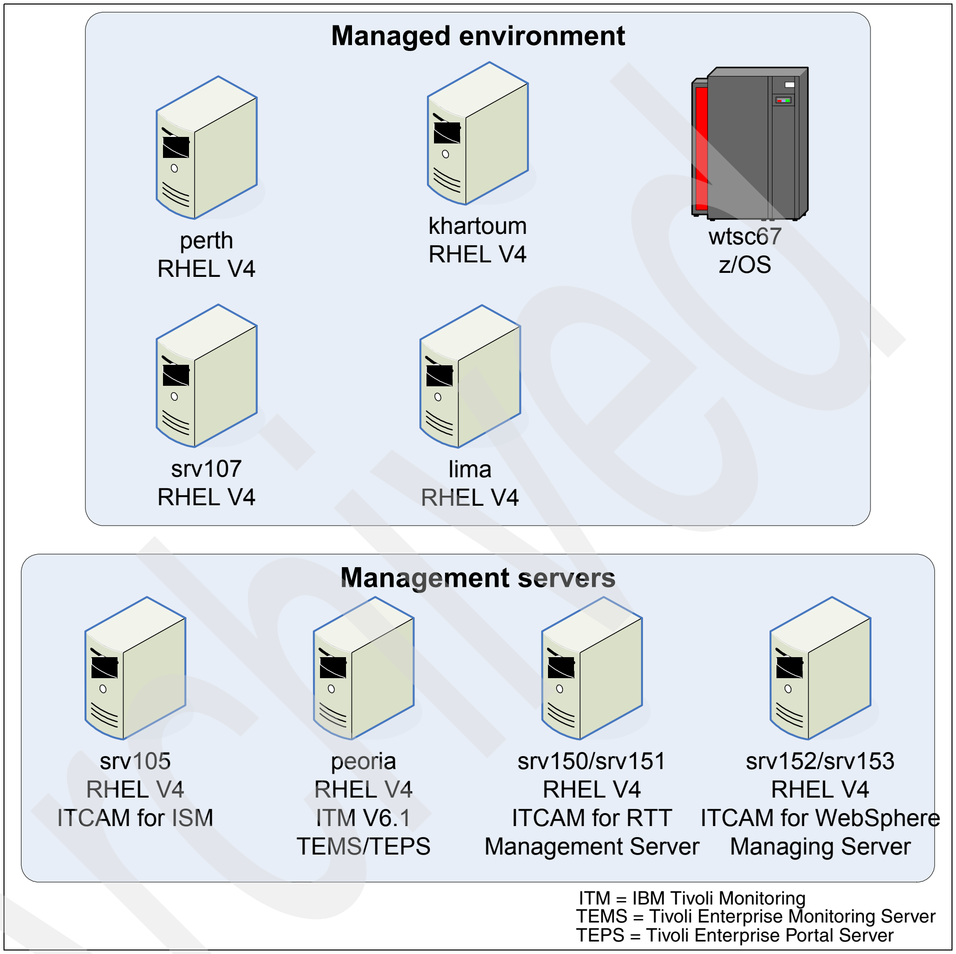


Figure 1-3 Project environment

The project mainly uses Linux®-based servers running Red Hat Enterprise Linux (RHEL) V4. The z/OS back end runs in another ITSO lab in Poughkeepsie. Because the usage is not necessarily related to the operating system platform that is used, we decided to limit our environment for the least number of operating systems to be maintained.

1.3.2 Application environment

The application, ITSO's trader, is a distributed application with the components illustrated in Figure 1-4.

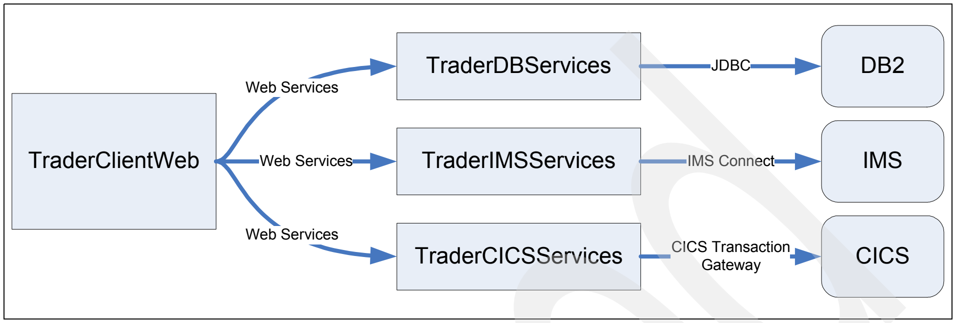


Figure 1-4 ITSO's trader application environment

The application consists of the following components:

- Front-end presentation called TraderWebClient that provides client-side access. It uses Web services calls to perform its functions.
- Service application that provides the business logic for the application, running in another WebSphere server. This application acts as a Web services provider to get the requests. There are three service applications that are available, TraderIMSServices, TraderCICSServices, and TraderDB2Services. Each have a different data store in the back end.
- IMS server on z/OS, accessed using an IMS Connect interface.
- CICS Transaction Server on z/OS, accessed using CICS Transaction Gateway.
- DB2® database, accessed using direct JDBC™ calls.

The trader application has been expanded to accommodate SOA mediation, WebSphere Portal Server, and Tomcat interface. Although these additions are not fully exploited in this project, the complete trader application environment is discussed in <Traderapp>.

1.4 Document organization

This book is organized by product. We discuss individual products in their own chapters. We specifically pull out the discussion for the z/OS-based implementation in a separate chapter because most readers who implement the z/OS platform are systems programmers who want this information in one place. The last chapter addresses the issue of integration of these different products.

The chapter structure is:

2

ITCAM for WebSphere and ITCAM for J2EE concepts and installation

This chapter describes IBM Tivoli Composite Application Manager (ITCAM) for WebSphere and IBM Tivoli Composite Application Manager for J2EE. The discussion includes:

2.1 Monitoring J2EE application servers

The IBM Tivoli application management solution for J2EE application servers comes in the form of ITCAM for WebSphere and ITCAM for J2EE. These two products share the same managing server. ITCAM for WebSphere and ITCAM for J2EE observe and report on the health of J2EE-based applications. They track the progress of applications as they traverse through J2EE application servers, middleware adapters and transports, and database calls, and on to back-end systems such as CICS or IMS to extract business data or to invoke mainframe business processes.

Tracking applications produces request traces, where the events in a request's life are recorded and stored in a monitoring repository database. ITCAM for WebSphere and ITCAM for J2EE capture the CPU and the elapsed internal times when events are called and when they are exited, measuring as far down as the CPU 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.

ITCAM for WebSphere manages and monitors WebSphere-based application servers, while ITCAM for J2EE manages and monitors the following J2EE containers:

- JBoss
- Tomcat
- SAP NetWeaver
- BEA WebLogic Server
- Oracle Application Server
- WebSphere Application Server CE

ITCAM for Web Servers Tivoli Enterprise Monitoring Agent monitors:

- Apache Web Server
- Sun Java System Web Server
- Microsoft IIS

ITCAM for WebSphere and ITCAM for J2EE do not need modification of any J2EE or mainframe application code. The data collectors use the following principal data sources: Java Virtual Machine Tool Interface (JVMTI) for JDK™ 1.5 and Java Virtual Machine Performance Interface (JVMPI) for JDK 1.3 and 1.4. ITCAM for WebSphere also uses WebSphere Performance Management Interface (PMI) and z/OS System Measurement Facility (SMF) 120 records. The monitoring data is collected and analyzed to offer a wealth of information about the health of J2EE applications and their servers.

These products collect and report many system-level performance metrics about J2EE application servers. The status of the servers and their resources (particularly at vital checkpoints such as CPU utilization), memory usage, and the status of internal components (such as database connection pools, JVM™ thread pools, EJB usage, and request processing statistics) can be very important in locating real-time problems with J2EE applications. ITCAM for WebSphere and ITCAM for J2EE bring attention to these critical indicators with real-time, graphical displays of their values and their trends over spans of time.

ITCAM for WebSphere and ITCAM for J2EE now have a new feature called the Problem Center. This new feature provides expert advice to quickly determine the root cause of a problem. Problems can come from a situation from IBM Tivoli Monitoring, and the wizard-like feature runs a set of rules on the problem and recommends actions, such as setting a time trap, to get to the root cause of the problem.

2.2 Architecture and interconnection

ITCAM for WebSphere and ITCAM for J2EE are distributed performance monitoring applications for application servers. The components are connected through TCP/IP communication. The central component of ITCAM for WebSphere and ITCAM for J2EE, the *managing server*, is its heart and brain. It collects and displays various performance information from application servers.

The application servers run the *data collector*, which is a collecting agent that runs in the application server and sends monitoring information to the managing server. These data collectors operate independently of each other. Figure 2-1 shows the overall architecture of ITCAM for WebSphere and ITCAM for J2EE.

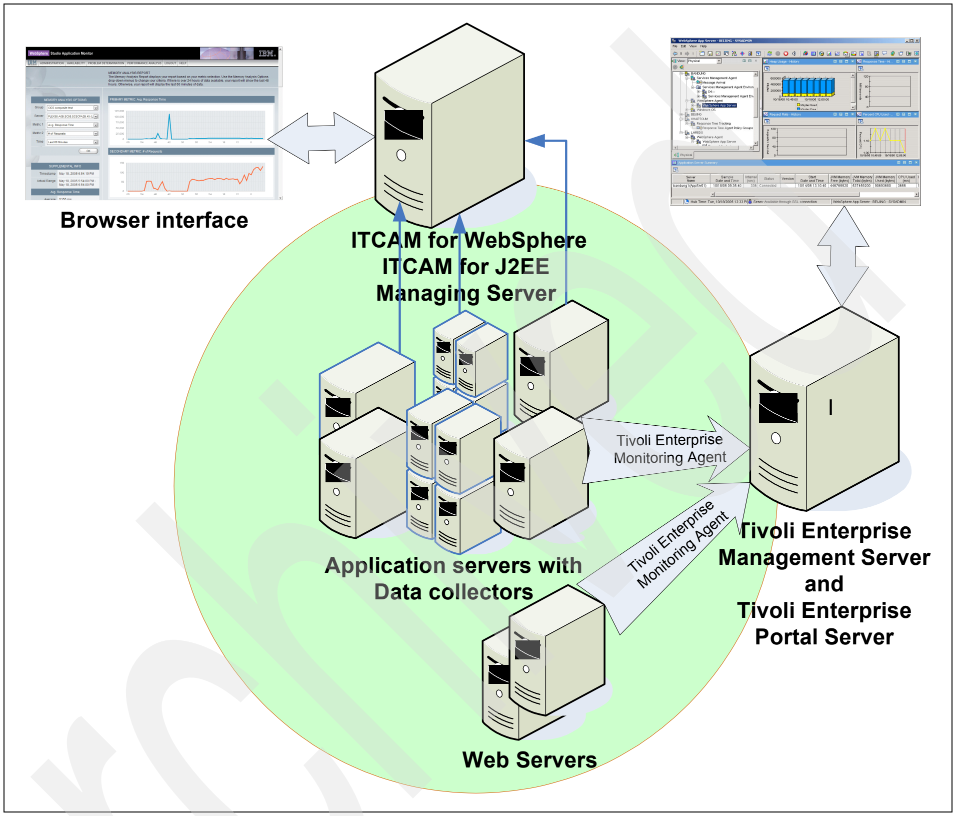


Figure 2-1 ITCAM for WebSphere and ITCAM for J2EE architecture

The application monitor consists of two main parts: the managing server and the data collectors. A data collector agent runs on each monitored application server, whether J2EE, CICS, or IMS, and communicates essential operational data to the managing server. Unique sampling algorithms maintain low CPU and network processing while providing application-specific performance information. The managing server consists of several Java-based components that provide the environment to collect and present management data.

2.2.1 The managing server

ITCAM for WebSphere and ITCAM for J2EE use one common managing server that controls and coordinates data collectors for J2EE, CICS, and IMS servers that run applications. The difference between ITCAM for J2EE and ITCAM for WebSphere is the platform support for the data collectors. These data collectors can run independently.

The managing server uses the following software:

- Managing server database (DB2 UDB or Oracle) for relational data repository
- WebSphere Application Server to run the visualization engine Web console application
- An optional Web server, such as IBM HTTP Server
- The managing server overseer components, which are a set of Java-based processes

The overseer components are the controlling logic for the managing server. For the overseer components:

- Kernels control the managing server. There are always two copies of the kernels running on an ITCAM for WebSphere and ITCAM for J2EE managing server for redundancy and failover. The kernels register components as they join the managing server, periodically renew connections and registrations with components and data collectors, and collect server and component availability information.
- Publishing servers receive application and system event data from the data collectors, gather and compute request-level information about performance metrics such as response times, and implement the trap monitoring and alerts features.
- Archive agents receive monitoring data from the publish servers, and store the monitoring data in the repositories of ITCAM for WebSphere and ITCAM for J2EE.
- The global publishing server collects information from the publish servers and correlates all parts and pieces of multiserver requests, such as requests from J2EE servers to execute CICS or IMS programs.
- The message dispatcher is a conduit for messages from ITCAM for WebSphere and ITCAM for J2EE using e-mail and SNMP facilities.

- The visualization engine is a Web-based GUI with access to graphics, ITCAM for WebSphere and ITCAM for J2EE performance reports, real-time views of different slices of monitoring data, and ITCAM for WebSphere and ITCAM for J2EE internal commands and event-driven functions. The visualization engine runs on a WebSphere Application Server.

Figure 2-2 shows the conceptual relationship between the components.

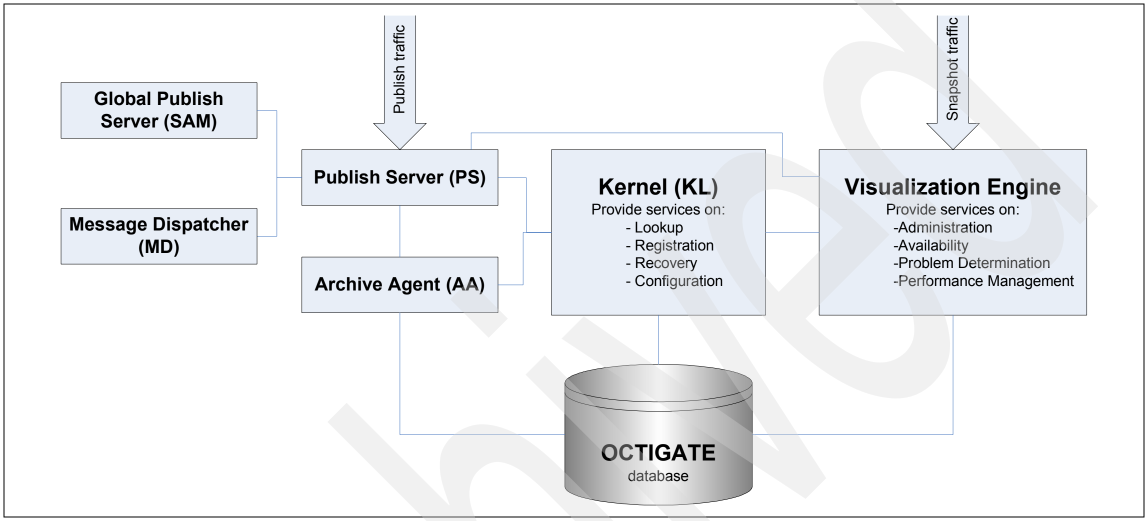


Figure 2-2 Kernel components

At the managing server, monitoring data is prepared for real-time displays within the monitoring console and is inserted into the OCTIGATE data repository. These are very resource-intensive operations. Moving them the managing server isolates them from other the application servers, thus reducing the footprints of ITCAM for WebSphere and ITCAM for J2EE in the monitored systems. This design also helps keep the data collectors's processing at levels low enough for 24x7 production system monitoring.

Data from the data collectors is collected by the publishing server then stored in the OCTIGATE database by the archive agent. The visualization engine reads the database to present data through the Web console, and snapshot information, such as lock analysis and in-flight transactions, is retrieved directly from the data collectors.

2.2.2 J2EE and WebSphere data collectors

The data collectors run inside the application servers. They 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 MVS™ Cross-Memory Services and address space fencing, which are not available on distributed systems.

Data collectors are configured as a multithreaded process. They consist of the following 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 publish servers according to polling frequencies. This data includes system initialization data, application request-level data, and application method-level data.

- Secondary collector

  The optional secondary collector provides support for displaying data in Tivoli Enterprise Portal for collecting WebSphere Application Server and other J2EE application server performance metrics. This component communicates with Tivoli Enterprise Monitoring Agent using a TCP/IP port.

Collectively, these agents and other data collector routines unleash the probes, package the monitoring data into Java formats for the managing server, and deliver the data to the managing server.

The data collectors send the probes into the application servers to analyze the applications' performance. The probes collect monitoring data and feed it to transport routines that in turn route the data to the managing server. The managing server processes it for display in the Web console and for storage in the OCTIGATE repository. This relieves the processing burden of ITCAM for WebSphere and ITCAM for J2EE 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 that are employed by ITCAM for WebSphere and ITCAM for J2EE are:

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

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

The data collector in the J2EE server runs as a custom service called am in distributed and “-” for z/OS. Figure 2-3 shows the conceptual data collector structure of the distributed WebSphere data collector.

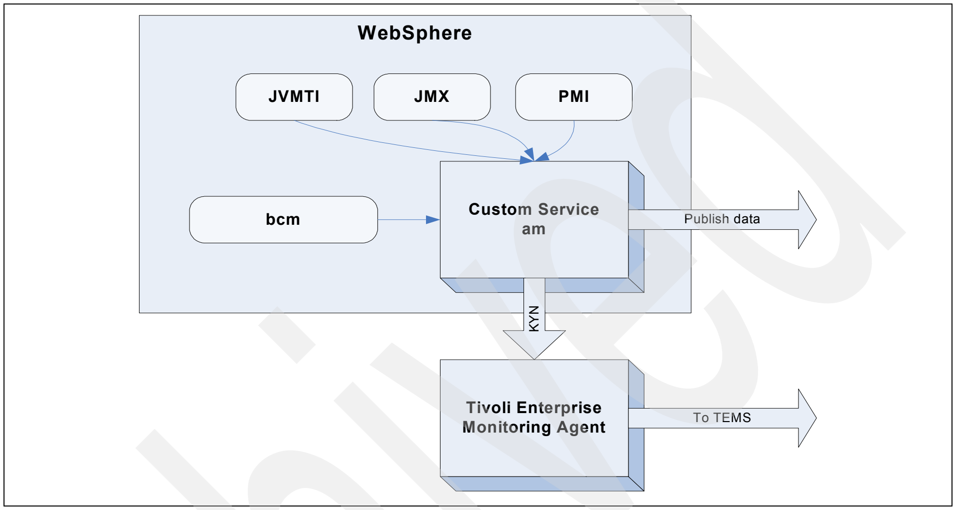


Figure 2-3 J2EE data collector structure

2.2.3 Tivoli Enterprise Monitoring Agent

The ITCAM for WebSphere and ITCAM for J2EE Tivoli Enterprise Monitoring Agent can forward monitoring information to Tivoli Enterprise Monitoring Server for monitoring using Tivoli Enterprise Portal. There is an additional component in Version 6.1, Tivoli Enterprise Monitoring Agent for Web Servers. Web server monitoring no longer uses the managing server’s polling agent, but uses Tivoli Enterprise Monitoring Agent for Web Servers instead.

The existing Tivoli Enterprise Monitoring Agent for WebSphere and J2EE provides application server performance information, while the new Tivoli Enterprise Monitoring Agent for Web Servers displays Web server performance information.

2.2.4 Monitoring resources

ITCAM for WebSphere and ITCAM for J2EE uses the following terminology:

Monitoring on demand Monitoring level for the amount of instrumentation performed by the data collector and the percentage of information that is stored.

Server configuration Assigning a data collector with a profile. A server profile determines the classes to include or exclude for monitoring.

Server group A logical grouping of servers that enables aggregation of status or is used for report generation grouping.

Report Interactive display that enables navigation of collected information in the database to analyze previous performance of application. Reports can also be generated into a PDF file.

Monitoring levels

The monitors for ITCAM for WebSphere and ITCAM for J2EE run on a predefined monitoring level. These levels can be changed dynamically without the need to restart the monitored systems. The levels are:

- Level 1: Production-level monitoring that provides basic response time and transaction information.
- Level 2: More detailed information that includes method call details and memory usage. In Version 6.1, there is a new method profiling option for this level to collect aggregate method calling statistics with a much lower processing requirement than level 3 monitoring.
- Level 3: Detailed tracing information that shows all method entries and exits with significant amounts of information collected.

Data collector configuration

The data collector configuration is stored in the $DC_HOME/runtime sub directory. It is governed by several configuration files, among which are:

- custom/toolkit_custom.properties

  This file stores custom settings for bytecode instrumentation properties.
- 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.

- jiti.properties

  Just-in-time instrumentation property for Java class instrumentation profile.

- <wasver>.<node>.<srv>.datacollector.policy

  Java security permission of the data collector.

- <wasver>.<node>.<srv>.datacollector.properties

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

- <wasver>.<node>.<srv>.kwjdc.properties

  Property file for Tivoli Enterprise Monitoring Agent connector.

- <wasver>.<node>.<srv>.toolkit.*

  Instrumentation properties with ITCAM toolkit.

- <node>.<srv>.cyaneaGpsCounter.txt

  Counter that identifies a sequence number for matching a composite transaction.

Managing server utilities

Several utilities for the managing server are useful for maintaining the operation of ITCAM for WebSphere/J2EE.

The following utilities are for the managing server components:

- klctl.sh: With this command, you can start, stop, and ping the kernel manually. You can also change the debugging level and check the database connection.
- am-check.sh: This verifies whether all settings are correct. This test can be run manually. am-start.sh executes this check implicitly.
- amctl.sh: This script can start, stop, ping, get status, and set logging levels for all components.
- am-start.sh: This script starts the managing server components.
- am-stop.sh: This script stops the managing server components.
- add-ps.sh: This script adds a separate publish server instance.
- add-aa.sh: This script adds a separate archive agent instance.

Some programs manage the OCTIGATE database in the management server, such as initializing or maintaining the database. Those are:

- db2configuration.sh: database creation tools
- db2createschema.sh: database schema creation, when the database is already created
- db2settings.sh: database setting definition for existing database
- migration*.sh: database migration script
- migration*.sh: updates the database statistics
- aa_deletedata.sh and datatrim.sh: trims data

Other programs that are provided in the managing server include:

- authenticate: verifies the user's account and password to the operating system for the Web console
- rotate-apache-logs.sh: rotates Apache logs by issuing the **apachectl** command
- setenv.sh: sets configuration parameters for the managing server components, including memory sizes and database access
- dcctl.sh: controls the logging level in data collectors

2.2.5 Logging and troubleshooting

The ITCAM 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. In general, the logging level can be modified, either from the properties file to include am.debug=yes or 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 in $commondir\CYN\logs. Some logs might be in <admin>.<server> path, such as:

- msg-dc.log
- trace-dc.log
- msg-dc-native.log
- trace-dc-native.log

The z/OS data collector logs are in /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 is to modify part of the logging for the CICS instrumentation to DEBUG MAX, as shown in Example 2-1.

Example 2-1 Excerpt of cynlogging.properties file

```
   . . .

#------------------------------------------------
# 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

   . . .
```

2.3 Implementation overview

The following procedure describes the overall implementation process for ITCAM for WebSphere and ITCAM for J2EE:

1. Plan for the configuration. The primary concerns for this planning process are the expected load and sizing of the managing server. The managing server is the centerpiece of ITCAM for WebSphere and ITCAM for J2EE, so the implementation must consider its performance and capacity implications. Another important planning factor is the security and communication structure between the managing server and data collectors. We discuss implementation planning in this section.
2. Install the managing server. We discuss this in 2.4, “Installation process” on page 33. This includes database setup and tuning.
3. Install data collectors. Different data collectors require different installation mechanisms. The available data collectors are for ITCAM for WebSphere, ITCAM for J2EE, ITCAM for IMS Transactions, and ITCAM for CICS Transactions. We discuss the distributed-based data collectors in this chapter, and discuss the z/OS-based data collector in Chapter 10, “Implementation of ITCAM products on z/OS” on page 521.
4. (Optional) For feeding information into Tivoli Enterprise Monitoring Server with Tivoli Enterprise Portal, you must install the Tivoli Enterprise Monitoring Agent for ITCAM for WebSphere or ITCAM for J2EE on each data collector system. See 2.5, “Integration with Tivoli Enterprise Portal” on page 49.
5. Certain administrative tasks must be performed before monitoring can take place. We discuss these in 2.6, “Initial configuration and operation” on page 57:
   - Assign the data collector configuration.
   - Modify the monitoring level.
   - Define server groups.
   - Create operators.

The planning considerations include:

- 2.3.1, “Sizing consideration” on page 27
- 2.3.2, “Security considerations” on page 29
- 2.3.3, “Implementation configuration” on page 31
- 2.3.4, “Managing server installation options” on page 31

2.3.1 Sizing consideration

As discussed in 2.2.1, “The managing server” on page 18, the managing server consists of multiple processes that each run in its own Java Virtual Machine. This architecture, while providing redundancy and scalability, also requires a large amount of memory. The ITCAM for WebSphere and ITCAM for J2EE managing server needs at least 4 GB of memory. The memory and processing requirements largely depend on:

- The number of data collectors connected to the publish server and kernel
- The amount of transaction information sent to the publish server
- Monitoring levels of transactions by the data collectors, which translate to the data size sent to the publish server
- The monitoring sampling rate, which determines the numbers of records that are passed from the publish server to the archive agent
- The amount of correlation for composite transactions that the Global Publish Server must process
- The number of users connected to the visualization engine
- The number of users who perform problem determination, which translates to direct connection to the data collectors from the visualization engine

Database sizing for the managing server can be estimated using the worksheet on this Web page:

http://www.ibm.com/support/docview.wss?rs=2344&context=SS3PGL&dc=DB520&q1=database&uid=swg21250907&loc=en_US&cs=utf-8&lang=en

JVM sizes are determined from the setenv.sh script in the bin directory of the managing server. Example 2-2 shows the default memory sizes.

Example 2-2 Memory size

```
HEAP_MIN_SIZE_PS=256
HEAP_MAX_SIZE_PS=512
HEAP_MIN_SIZE_ARCHIVE_AGENT=256
HEAP_MAX_SIZE_ARCHIVE_AGENT=512
HEAP_MIN_SIZE_EMAIL=32
HEAP_MAX_SIZE_EMAIL=64
HEAP_MIN_SIZE_MESSAGE_DISPATCHER=32
HEAP_MAX_SIZE_MESSAGE_DISPATCHER=64
HEAP_MIN_SIZE_KERNEL=128
HEAP_MAX_SIZE_KERNEL=256
HEAP_MIN_SIZE_AVM=32
HEAP_MAX_SIZE_AVM=64
HEAP_MIN_SIZE_SAM=64
HEAP_MAX_SIZE_SAM=128
HEAP_MIN_SIZE_POLLING_AGENT=64
HEAP_MAX_SIZE_POLLING_AGENT=64
```

Note: For proof of concepts and demonstration systems without any production load, you can decrease the memory footprint for publish servers and archive agents. You might also want to suppress starting and stopping processes that you do not need in the am-start.sh and am-stop.sh scripts.

As shown in Figure 2-4, the publish server and archive agent use the majority of the memory. Publish servers and archive agents process transaction information from the data collectors. Visualization engine memory can be enforced from the WebSphere Application Server administrative console.

To optimize performance and provide additional reliability, the managing server can be divided among several machines. Although the provided installation wizard allows only a single-machine installation for the managing server components, customization can be performed to split the managing server.

Note: The wizard allows you to only install the database on a remote DB2 database. We do not cover split-server installation in this book.

Figure 2-4 shows a sample split-server installation.

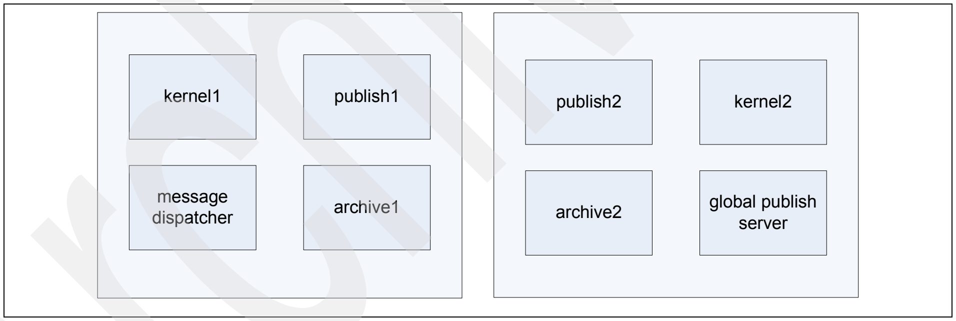


Figure 2-4 Split-server installation

2.3.2 Security considerations

Communication security between the ITCAM for WebSphere managing server and the data collectors requires the use of certificates and SSL encryption. Because transaction information carries a large amount of information, including potentially sensitive information, this is a must for an enterprise. We do not discuss SSL implementation in this book, but we recommend performing this with your own certificates, instead of using the presupplied certificate that comes with the product. This ensures the confidentiality of communication.

Another important security consideration is the use of the TCP/IP port. ITCAM for WebSphere and ITCAM for J2EE use a range of ports in the data collector to communicate with the publish server and kernel in the managing server. When communicating through a firewall, you might want to use a port consolidator proxy. Example 2-3 shows default port usage.

Example 2-3 Port usage

```
PORT_KERNEL_CODEBASE01=9122
PORT_KERNEL_CODEBASE02=9123
PORT_KERNEL_RFS01=9120
PORT_KERNEL_RMI01=9118
PORT_ARCHIVE_AGENT1=9129
PORT_ARCHIVE_AGENT2=9130
PORT_MESSAGE_DISPATCHER=9106
PORT_PS=9103
PORT_PS2=9104
PORT_PA=9111
PORT_SAM=9126
```

2.3.3 Implementation configuration

Figure 2-5 shows the overall environment of the implementation that we performed on our system to manage the ITSOTrader application. The managing server is installed in a single machine in peoria, and the data collectors are connected directly to the managing server.

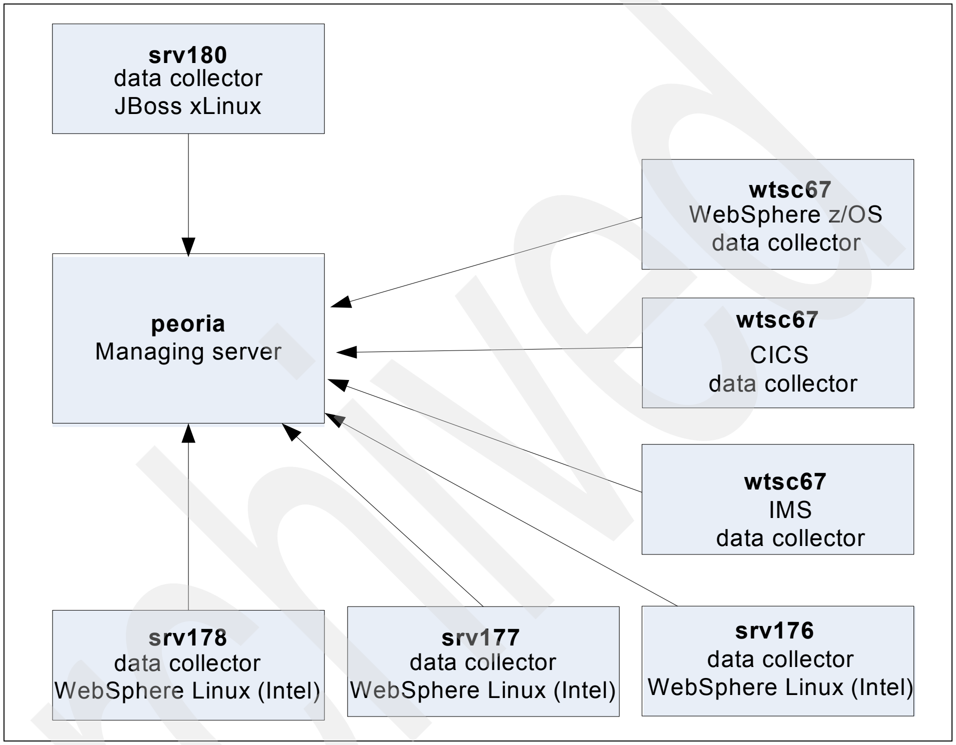


Figure 2-5 ITCAM for WebSphere and ITCAM for J2EE implementation

2.3.4 Managing server installation options

There are several installation options for the managing servers. These decision points are:

- Whether to perform a typical installation or a custom installation. The typical installation is also known as the embedded installation. This allows installation of the WebSphere Application Server and the DB2 Universal database to be performed with a single wizard, thus simplifying the installation process.
- Whether to perform a graphical (attended) or silent (unattended) installation. Both typical or custom installation can be performed with either methods.

The embedded installation installs WebSphere Application Server and DB2 with the ITCAM for WebSphere managing server. This embedded installation simplifies the installation process and does not require separate WebSphere and DB2 installation. The embedded installation is divided into two options: the typical installation and the custom installation.

Using the typical installation, the wizard selects the following default values for the installation:

- The installation paths for the DB2, WebSphere, and managing server installations:
  - Product directory for DB2 /opt/IBM/db2 or C:\Program Files\IBM\itcam\WebSphere\MS\DB2
  - Product directory for WebSphere /opt/IBM/itcam/WebSphere or C:\Program Files\IBM\itcam\WebSphere\MS\WAS61
  - Product director for managing server /opt/IBM/itcam/WebSphere/MS or C:\Program Files\IBM\itcam\WebSphere\MS
- The DB2 users needed for the installation:

  db2admin The DB2 administration user
  db2inst1 The DB2 instance user
  db2fenc1 The DB2 fence user
- The installation procedure also creates a database called OCTIGATE to store the performance information collected by the managing server. The default port number for DB2 is 50000.

To use any values other than the default ones for the installation, use the custom installation. This option enables you to customize the following values:

- For DB2, you can choose the following settings:
  - The user names
  - The port for the database
- For WebSphere, you can choose the following settings:
  - The installation path
  - The user you want to use for the installation and the user you want to use to access the user interface
- For the managing server, you can define the ports for the different components of the managing server.

The second way to install the managing server is separate installations of WebSphere, DB2, and the managing server. You can also use an existing database on the local system or a remote DB2 or Oracle database. When using an existing database, we recommend using a supported version. In our installation, we used DB2 Version 8.2 Fix Pack 1. You also have to use WebSphere Version 6.1. When you do a separate installation of the database or use an existing database, the installation wizard can still configure the database tables. You can also use the scrips that are supplied on the installation CD for the database configuration.

Another way to install the managing server is the silent script, which also comes with the installation CD in the folder /silent. When using the silent installation, you have to define all installation options and values in the script before starting the installation.

In this book, we install the managing server using the embedded installation with the typical settings.

2.4 Installation process

In this section, we discuss the ITCAM for WebSphere and ITCAM for J2EE installation. The subsections are:

- 2.4.1, “ITCAM for WebSphere and ITCAM for J2EE managing server” on page 33
- 2.4.2, “ITCAM for WebSphere data collector” on page 38
- 2.4.3, “J2EE data collector” on page 45
- 2.4.4, “Setting up Apache HTTP Server” on page 48

2.4.1 ITCAM for WebSphere and ITCAM for J2EE managing server

We chose Red Hat Enterprise Linux 3.0 as our managing server platform. Typically, using a Linux-based system decreases the memory requirement for the base operating systems compared to Microsoft Windows®-based systems.

The following steps describe the installation of the ITCAM for WebSphere managing server:

1. Create the necessary user ID for the managing server installation. This user is used as an administrative user for the WebSphere and ITCAM managing server. The user will be able to access the WebSphere administrator’s console and the user interface of the managing server. For a UNIX® environment, this user needs root privileges. For a Windows environment, the user must be member of the administrator group.

Note: For a Windows-based managing server, you need to install the Microsoft Services for UNIX (SFU) utility. Download this from:

`http://www.microsoft.com/technet/interopmigration/unix/sfu/default.mspx`

2. Copy the installation images of DB2 and WebSphere to a local directory before starting the installation. The embedded installation method requires access to these installation images. The DB2 and WebSphere software level must match the requirement for the embedded installation process exactly.
3. Use the *ITCAM for WebSphere managing server for Linux* CD-ROM to start the installation wizard. The installation wizard in a UNIX or Linux-based system requires an X Window System server application, such as the native desktops or Windows-based X Window System server application.

4. Start the installation by using the `launchpad` command from the installation media. Use the appropriate launchpad executable for your platform. We use launchpad.sh for the Linux system. Select **Install ITCAM** and click the **Start the installation process for ITCAM for WebSphere and J2EE Managing Server Version 6.1** link, as shown in Figure 2-6.

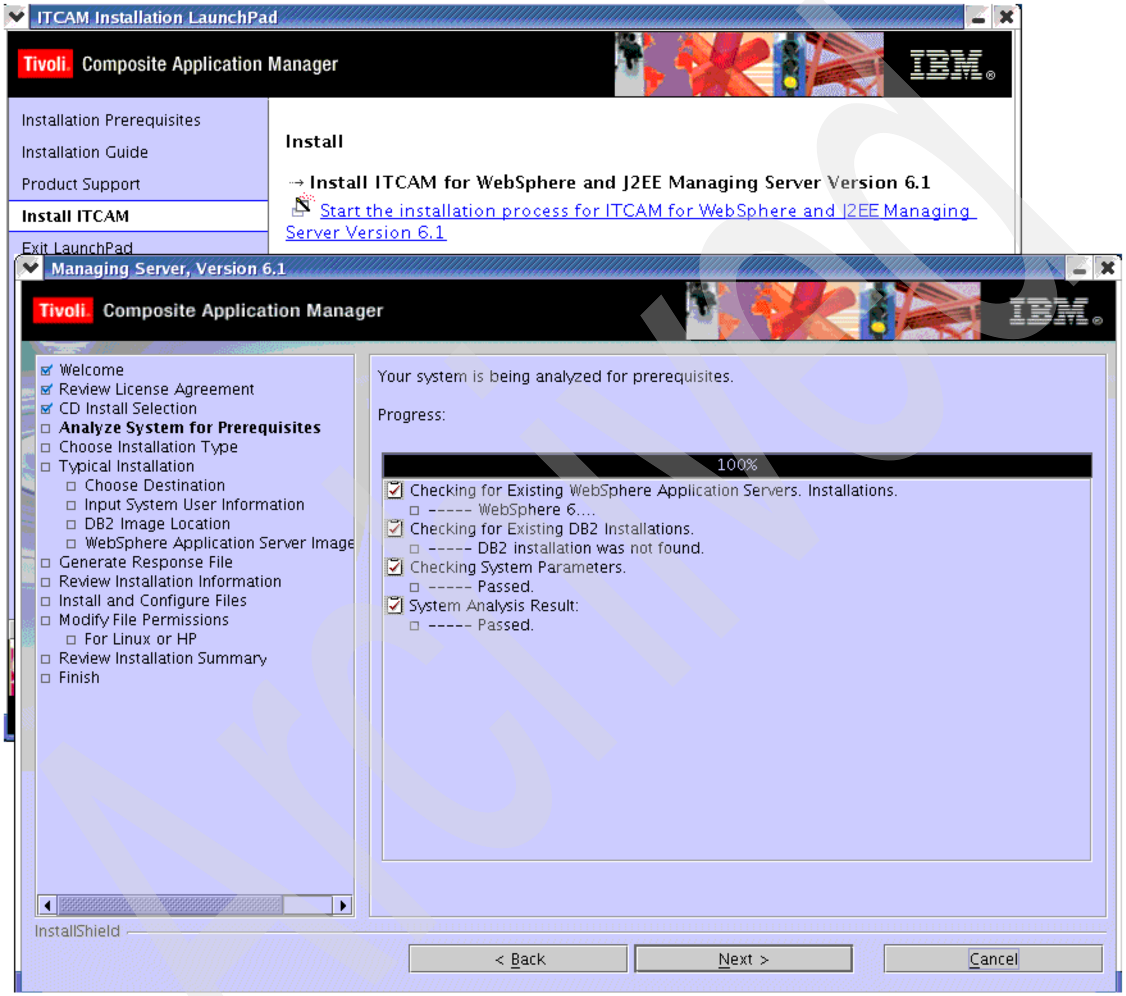


Figure 2-6 Installation startup window

5. If you are installing from CD-ROM, the wizard copies the installation files to the temp directory. If you are installing from an installation image on the hard drive, the wizard uses the existing installation media.

   The wizard checks the prerequisite for the installation and shows you the results, as shown in Figure 2-6.

6. The next step enables us to select the type of installation: typical or custom. We decide to keep the default settings for the installation. Therefore, we select **Typical**.
7. In the next step, we define the installation path of the managing server, as shown in Figure 2-7.

Figure 2-7 Defining the installation path

We typed in the managing server administrative user that we defined at the beginning of the installation, as shown in Figure 2-7.

8. Because we copied the DB2 and WebSphere installation images to a local directory before we started the installation, we select those images, as shown in Figure 2-8.

Figure 2-8 Selecting the installation images

9. As you can see in Figure 2-9, you can install the managing server, and you can also create a response file. The response file saves your settings so that you can use them again for a silent installation of the managing server. It is also possible to just create the response file if you do not want to install the managing server at this time, but you want to keep your settings.

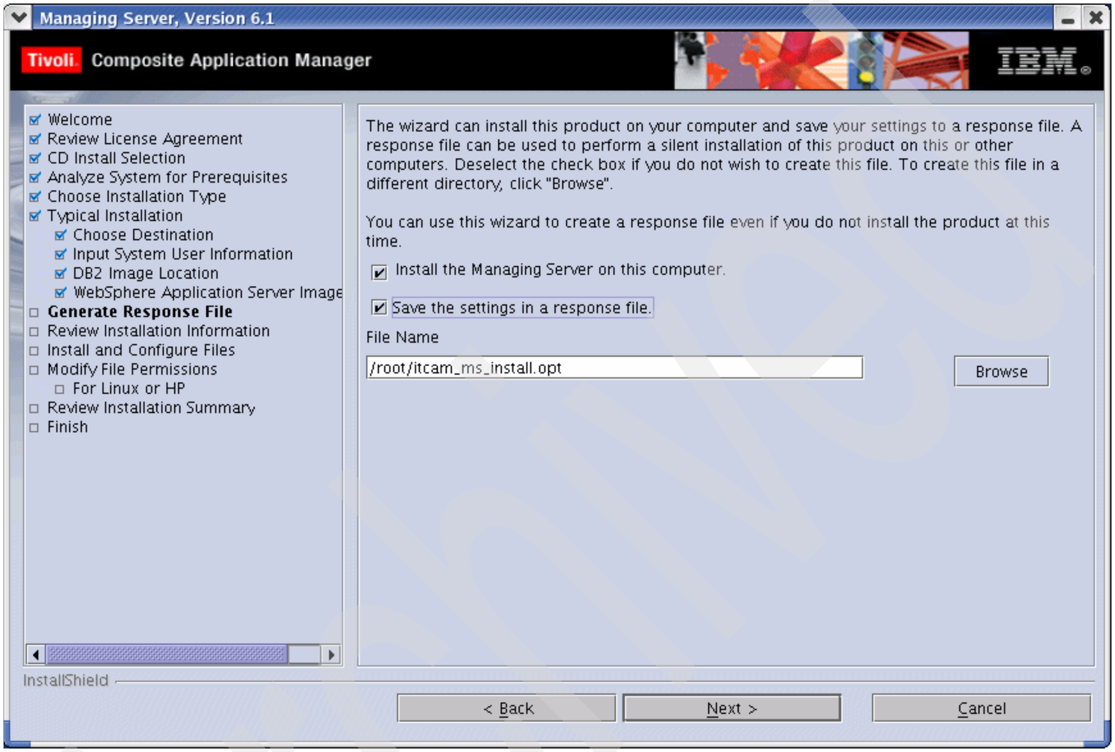


Figure 2-9 Creating a response file

2.4.2 ITCAM for WebSphere data collector

The distributed WebSphere Application Server data collector is installed using the installation wizard in a two-step process. The first phase transfers the necessary files, and the second step configures a specific data collector.

Requirements for installing a distributed WebSphere Application Server data collector include:

- Because the data collector will be installed in the application server, the installer requires access to the administrative console for the appropriate application server. This means that, for a network deployment environment, it accesses the network deployment server using SOAP access.

- To use the wizard, you need an X Window System environment for UNIX or Linux-based systems, either natively or using an X Server application, such as Exceed Hummingbird.
- You need a file system for the installed product. You also need to allocate the Tivoli common directory path that will contain log files (/var/ibm/tivoli/common or C:\Program Files\ibm\tivoli\common). This file system must be large enough to contain the various message files and trace log files.
- To install the data collector, make sure that the WebSphere Application Server into which you want to install the data collector is running.

The following sample installation flow is from a Linux system with a WebSphere Application Server V6.0.2 Network Deployment environment. We highlight some of the important windows.

1. Start the installation by using the `launchpad` command from the data collector installation media. Use the appropriate launchpad executable for your platform. We use launchpad.sh for the Linux system. We select **Install ITCAM → Start the installation process for ITCAM for WebSphere Data Collector**, as shown in the Figure 2-10.

Figure 2-10 WebSphere Data Collector Installation first steps

2. Fill in the path where you want the data collector to be installed, as shown in Figure 2-10. You can also choose whether you want to create a response file

for a silent installation, as discussed in 2.4.1, “ITCAM for WebSphere and ITCAM for J2EE managing server” on page 33.

3. After the wizard copies the data collector files into the specified directory, we configure the data collector. This starts the configuration of the WebSphere data collector in a another window.

4. You can see in Figure 2-11 that we do both the configuration for the integration to Tivoli Enterprise Portal and the ITCAM for WebSphere Application Server interface. Be aware that you need to install another agent called Tivoli Enterprise Monitoring Agent to integrate the WebSphere data collector into Tivoli Enterprise Portal. We describe the installation of Tivoli Enterprise Monitoring Agent in 2.5, "Integration with Tivoli Enterprise Portal" on page 49.To configure the data collector for the ITCAM for WebSphere and ITCAM for J2EE application monitoring interface, you need to install the managing server, as described in 2.4, "Installation process" on page 33.
5. As you can also see in Figure 2-11, we insert the installation directory of our managing server.

Figure 2-11 Configuring the data collector for the managing server

6. In the next window, we configure the data collector to find the managing server by inserting the host name and the codebase port that can be defined at the managing server installation. The default port is 9122.
7. We enter the host name of the Tivoli Monitor Enterprise Agent and the port on which the agent is listening. In our case, we type in the local IP address and the default port, `63335`.
8. If there is a firewall between the data collector and the managing server, you need to enable the firewall option and specify the RMI port numbers.
9. Select the type of WebSphere server that you want to monitor. We select **WebSphere Application Server** in our installation.
10. The wizard detects the running WebSphere Application Server and asks you to select the server that you want to monitor.

11. As shown in Figure 2-12, we specify the home directory of the WebSphere Application Server. We also insert the WebSphere version and the Java home directory.

Figure 2-12 WebSphere configuration

12. In the second window in Figure 2-12, we specify the host name of the network deployment server we are using. If you do not use a deployment server, use the host name of the local WebSphere server.

13. In the last window shown in Figure 2-12 on page 44, we select the server that we want to monitor.

14. You can generate a response file for silent installations.

> **Note:** The data collector configuration configures WebSphere on the application server level. Changes are performed in the following server level xml files:
>
> - pmi-config.xml: configuring Performance Management Interface settings
> - variables.xml: adding several ITCAM for WebSphere variables:
>   **ITCAM61HOME** Data collector installation directory
>   **MS_AM_HOME** Management server installation directory
> - server.xml: adding the custom services am and generic JVM argument

After the configuration, you see the data collector registered in the managing server. You can see the data collector from the following menu from the visualization engine:

- Unconfigured data collectors from **Administration** → **Server Management** → **Data Collector Configuration** → **Unconfigured Data Collectors**
- Data collector controller process (PPEController) from **Administration** → **Managing Server** → **Self Diagnostic**

For further configuration, see 2.6, "Initial configuration and operation" on page 57.

2.4.3 J2EE data collector

In general, the installation of the ITCAM for J2EE data collector is similar to the installation of the ITCAM for WebSphere data collector. There are some differences depending on the J2EE platform on which you install the data collector. We demonstrate the installation of the ITCAM for J2EE data collector on a JBoss Application Server in this section. We also describe some differences between the WebSphere data collector and JBoss data collector installation. See 2.3, "Implementation overview" on page 27, for the list of all Web and application servers supported by the ITCAM for J2EE data collector.

Similar to the WebSphere data collector, the J2EE data collector implementation is performed in two stages: the file copying stage and the application server configuration stage. The file copying window is very similar to the ITCAM for WebSphere data collector that we describe in 2.4.2, “ITCAM for WebSphere data collector” on page 38, in step 1 on page 40 to step 8 on page 43.

1. After you finish the first stage, start configuring the J2EE application server. The following window shows the configuration values specific for the JBoss Application Server (Figure 2-13).

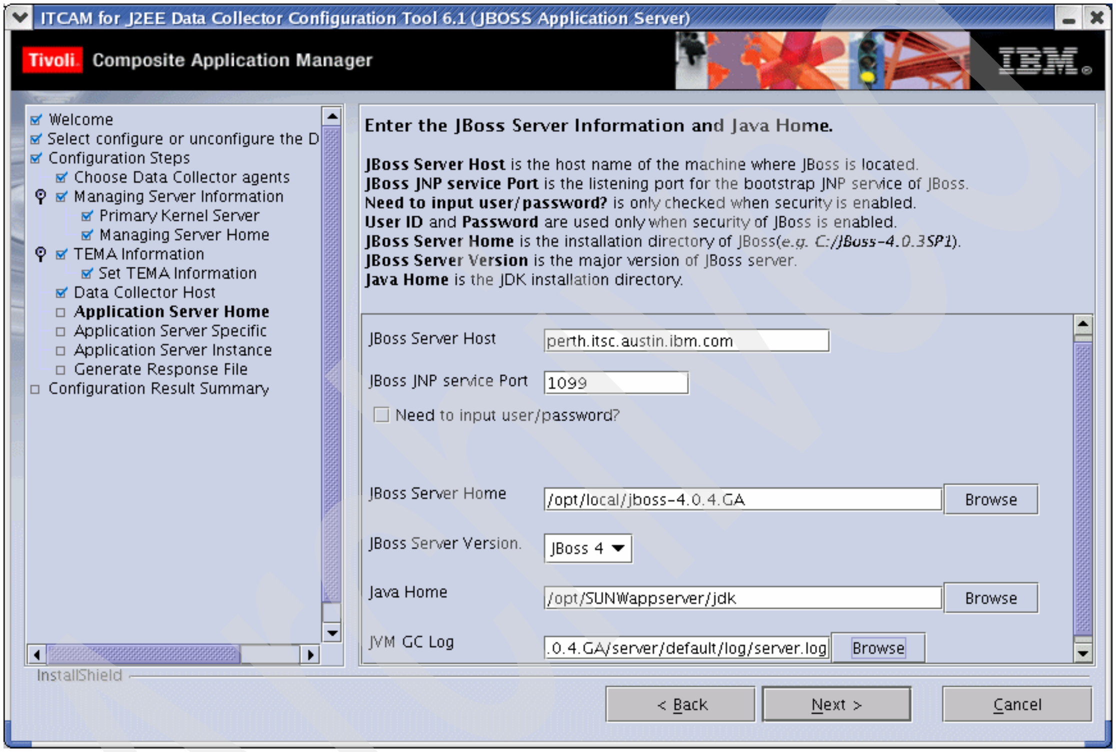


Figure 2-13 JBoss J2EE Data Collector Configuration

2. Select the JBoss Server Startup Script, as shown in Figure 2-14.

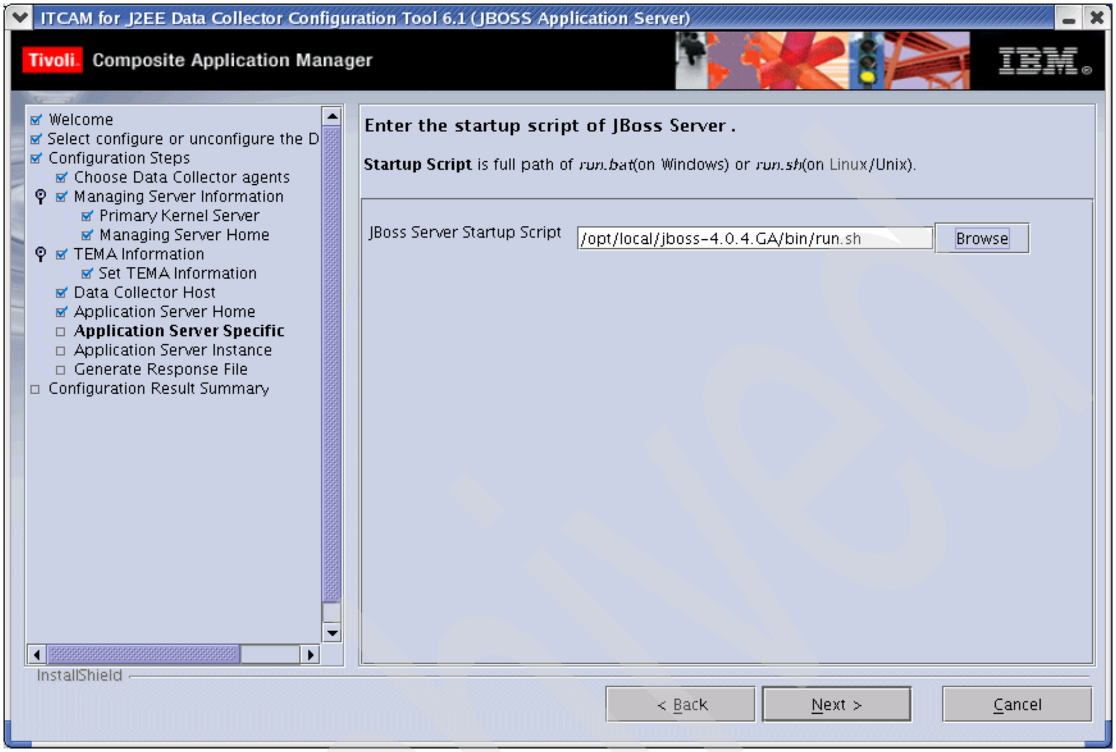


Figure 2-14 JBoss Server Startup Script

3. We select the server that we want to monitor. In our case, it is the default server.

4. In the last step, you can generate a response file for a silent installation.

2.4.4 Setting up Apache HTTP Server

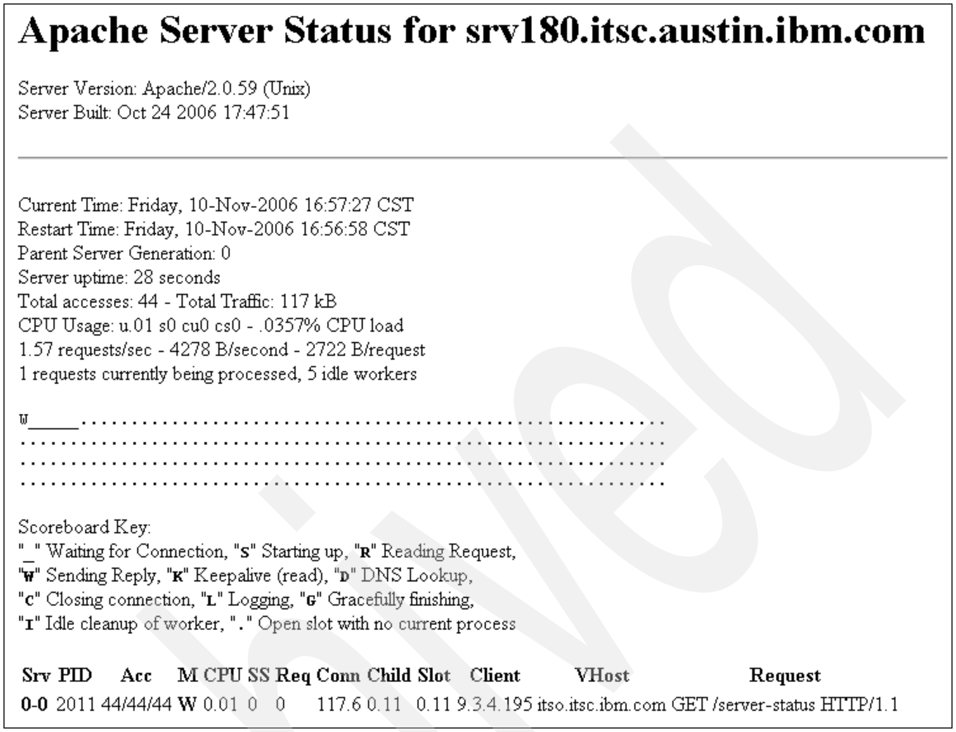
To be able to monitor the Apache Web Server using the ITCAM for Web Servers feature on ITCAM for WebSphere, you must activate the status monitor. To activate the status monitor, modify the httpd.conf file with the ExtendedStatus option. You need to add the server-status page in the location directive. See Example 2-4.

Example 2-4 Modification to httpd.conf file

```
   . . .
ExtendedStatus On
   . . .
<Location /server-status>
SetHandler server-status
<Location>
```

> **Note:** The runtime environment in our sample configuration uses Apache HTTP Server 2.0.59 on Red Hat Enterprise Linux AS Version 3 (Taroon Update 7). We use the default configuration.

As a test, we displayed the server status page, as shown in Figure 2-15.



Apache Server Status for srv180.itsc.austin.ibm.com

Server Version: Apache/2.0.59 (Unix)
Server Built: Oct 24 2006 17:47:51

Current Time: Friday, 10-Nov-2006 16:57:27 CST
Restart Time: Friday, 10-Nov-2006 16:56:58 CST
Parent Server Generation: 0
Server uptime: 28 seconds
Total accesses: 44 - Total Traffic: 117 kB
CPU Usage: u.01 s0 cu0 cs0 - .0357% CPU load
1.57 requests/sec - 4278 B/second - 2722 B/request
1 requests currently being processed, 5 idle workers

```
W_____..........................................................
................................................................
................................................................
................................................................
```

Scoreboard Key:
"_" Waiting for Connection, "S" Starting up, "R" Reading Request,
"W" Sending Reply, "K" Keepalive (read), "D" DNS Lookup,
"C" Closing connection, "L" Logging, "G" Gracefully finishing,
"I" Idle cleanup of worker, "." Open slot with no current process

Srv	PID	Acc	M	CPU	SS	Req	Conn	Child	Slot	Client	VHost	Request
0-0	2011	44/44/44	**W**	0.01	0	0	117.6	0.11	0.11	9.3.4.195	itso.itsc.ibm.com	GET /server-status HTTP/1.1

Figure 2-15 Apache Server Status

2.5 Integration with Tivoli Enterprise Portal

ITCAM for WebSphere and ITCAM for J2EE can be integrated into the IBM Tivoli Monitoring V6.1 infrastructure. This enables you to exploit IBM Tivoli Monitoring V6.1 functionality such as Tivoli Data Warehouse V2.1, correlation of WebSphere and other J2EE application server information with data from other IBM Tivoli Monitoring V6.1 agents, and presentation with Tivoli Enterprise Portal.

Tivoli Enterprise Monitoring Agent retrieves performance data about the WebSphere and other J2EE application servers for the following information:

- Response time data for application server requests from the ITCAM for WebSphere and J2EE data collector
- Resource data from the WebSphere and J2EE performance monitoring infrastructure
- WebSphere Application Server and other J2EE server log messages

- Garbage collector activity recorded in the Java Virtual Machine's verbose garbage collector trace

The Tivoli Enterprise Monitoring Agent retrieves performance data from the secondary collector of the data collector and ships it to the Tivoli Enterprise Monitoring Server, which processes the data for storage and display in the IBM Tivoli Monitoring infrastructure. Figure 2-16 shows the interconnectivity structure.

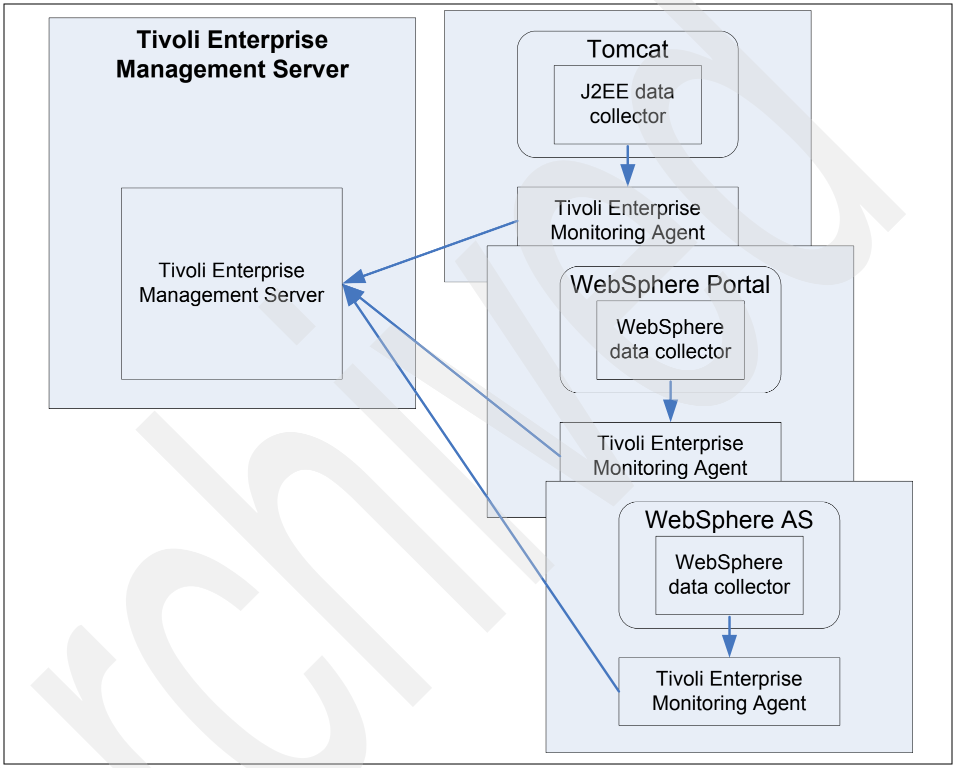


Figure 2-16 ITCAM for WebSphere integration with Tivoli Enterprise Monitoring Agent

In this book, we discuss the integration of data collectors into the IBM Tivoli Monitoring V6.1 infrastructure. For ITCAM for WebSphere and ITCAM for J2EE agents:

- There are separate installation images for the Tivoli Enterprise Monitoring Agent and the application support file.
- The agent code for ITCAM for WebSphere is YN, while ITCAM for J2EE is YJ. The agent code uses the command line to configure the agent. To configure ITCAM for WebSphere, use the command **`itmcmd.sh config -A yj`**. To configure ITCAM for J2EE, use the command **`itmcmd.sh config -A yn`**.

This section assumes that you have already installed the IBM Tivoli Monitoring V6.1 infrastructure. The implementation has three steps:

1. Preparing the IBM Tivoli Monitoring V6.1 infrastructure
2. Installing Tivoli Enterprise Monitoring Agent
3. Configuring Tivoli Enterprise Monitoring Agent

2.5.1 Preparing the IBM Tivoli Monitoring V6.1 infrastructure

To integrate ITCAM for WebSphere information into 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 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 Tivoli Enterprise Portal clients with product-specific information such as help files

This section covers these steps from a process point of view and guides you through the installation process. For the 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:** Be sure that you know on what machine these activities have to run. Check with your IBM Tivoli Monitoring V6.1 administrator and consult the appropriate IBM Tivoli Monitoring V6.1 documentation.

Perform the following steps:

1. Start the installation by clicking **Setup** in the Windows directory of the *ITCAM for WebSphere Tivoli Enterprise Monitoring Agent* CD-ROM or by running install.sh in your Linux or UNIX environment.

   Figure 2-17 shows the Welcome window for the installation of ITCAM for WebSphere. Click **Next**.

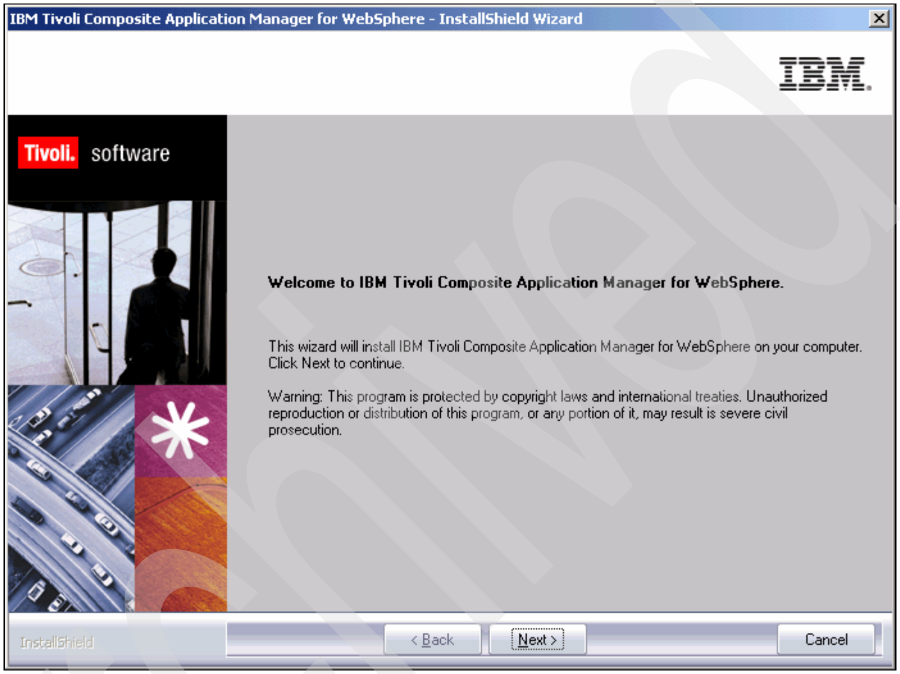


Figure 2-17 Tivoli Enterprise Monitoring Agent Welcome window

2. Review the Prerequisites window.
3. Accept the license agreement.
4. Choose the installation path.
5. Check the user data encryption key.

6. Select the features to install. Depending on where you are running this installation wizard, you may install different components on different machines. In our environment, we have both Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server on a single machine, so we selected (Figure 2-18):

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

   Note: You might have to install this on your Tivoli Enterprise Monitoring Server machine, Tivoli Enterprise Portal Server machine, and individual Tivoli Enterprise Portal desktop clients. You only have to install the component that is available in each machine.

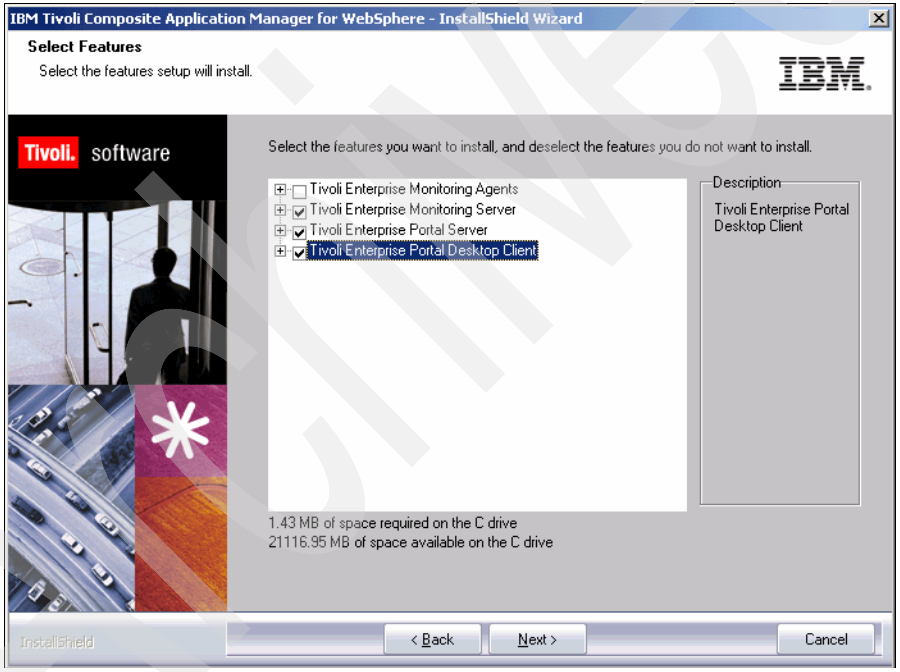


Figure 2-18 Tivoli Enterprise Monitoring Agent components

7. Follow the installation wizard and configure the components. Most of the components will already be preconfigured because 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 is a TCP connection.

2.5.2 Installing Tivoli Enterprise Monitoring Agent

We install Tivoli Enterprise Monitoring Agent on the same machine on which we installed the ITCAM for WebSphere or ITCAM for J2EE data collector. We recommend this because Tivoli Enterprise Portal displays the agent by machine, and having Tivoli Enterprise Monitoring Agent on the same machine as the application server reduces any confusion that might arise.

The installation of Tivoli Enterprise Monitoring Agent on the Linux platform is command-line based.

Start the installation using the install.sh script from the installation CD-ROM directory. Enter the name of the IBM Tivoli Monitoring directory. Example 2-5 lists the responses in our installation.

> **Note:** We recommend starting the agent with the same user ID used to start WebSphere, because the agent must be able to start and stop WebSphere with the default startServer and stopServer scripts.

Example 2-5 Tivoli Enterprise Monitoring Agent installation values

```
Enter the name of the IBM Tivoli Monitoring directory
[ default = /opt/IBM/ITM ]:/opt/IBM/ITM
   "/opt/IBM/ITM" does not exist
   Try to create it [ y or n; "y" is default ]?  y

Select one of the following:
   1) Install products to the local host.
   2) Install products to depot for remote deployment (requires TEMS).
   3) Exit install.
Please enter a valid number:  1

Initializing ...
   Software Licensing Agreement
         1. Czech
         2. English
         ...
   Please enter the number that corresponds to the language
   you prefer. 2
Software Licensing Agreement
...
1
Preparing to install the IBM Global Security Kit (GSkit)
   Preparing packages for installation...
   gsk7bas-7.0-3.18
```

```
   Will enable automatic agent initiation after reboot.
  Enter a 32-character encryption key, or just press Enter to use the
  default
          Default = IBMTivoliMonitoringEncryptionKey
  ....+....1....+....2....+....3..
  GSkit encryption key has been set.
  Key File directory: /opt/IBM/ITM/keyfiles

Product packages are available in /code/TEMA-YN/unix
Product packages are available for the following operating systems and
component support categories:
    1) Linux AMD64 R2.6 (64 bit)
    2) Linux Intel R2.4 (32 bit)
...
Type the number for the OS or component support category you want, or
type "q" to quit selection
[ number "2" or "Linux Intel R2.4 (32 bit)" is default ]:  2
Is the operating system or component support correct [ y or n; "y" is
default ]? y

The following products are available for installation:
    1) IBM Tivoli Composite Application Manager for WebSphere
  V06.10.00.00
    2) Tivoli Enterprise Services User Interface  V06.10.03.00
    3) all of the above
Type the numbers for the products you want to install, or type "q" to
quit selection. If you enter more than one number, separate the numbers
by a comma or a space.
Type your selections here:  1

The following products will be installed:
  IBM Tivoli Composite Application Manager for WebSphere  V06.10.00.00
Are your selections correct [ y or n; "y" is default ]? y
 ... installing "IBM Tivoli Composite Application Manager for WebSphere
V06.10.00.00 for Linux Intel R2.4 (32 bit)"; please wait.

 => installed "IBM Tivoli Composite Application Manager for WebSphere
V06.10.00.00 for Linux Intel R2.4 (32 bit)."
 ... Initializing database for IBM Tivoli Composite Application Manager
for WebSphere  V06.10.00.00 for Linux Intel R2.4 (32 bit).
 ... IBM Tivoli Composite Application Manager for WebSphere
V06.10.00.00 for Linux Intel R2.4 (32 bit) initialized.

Do you want to install additional products or product support packages
[ y or n; "n" is default ]? n
```

```
    ... postprocessing; please wait.
    ... finished postprocessing.
    Installation step complete.

 As a reminder, you should install product support on each of your TEM
servers for any agents you have just installed. This is done via the
"[ITM home]/bin/itmcmd support" command on your TEM servers.

 You may now configure any locally installed IBM Tivoli Monitoring
product via the "/opt/IBM/ITM/bin/itmcmd config" command.
```

2.5.3 Configuring Tivoli Enterprise Monitoring Agent

After finishing the installation, we configure the agent by running the **./itmcmd.sh config -A *yn*** command from the bin directory of the installation path, where *yn* is the two-character product code for ITCAM for WebSphere. Example 2-6 shows the configuration responses.

Example 2-6 Configuring Tivoli Enterprise Monitoring Agent

```
# ./itmcmd config -A yn
Agent configuration started...
Edit 'Basic' settings? (default is: Yes):Yes
Request Data Monitoring
        Type number of item from the below list
        1. Disable
        2. Level1
        3. Level2
         (default is: LEVEL1): 3
Request Data Monitoring Method
        Type number of item from the below list
        1. Fixed Interval
        2. On Demand
         (default is: FIXEDINTERVAL):1
Resource Data Monitoring
        Type number of item from the below list
        1. Disable
        2. Enable
         (default is: ENABLE):2
Resource Data Monitoring Method
        Type number of item from the below list
        1. Fixed Interval
        2. On Demand
         (default is: ONDEMAND):2
Garbage Collection Monitoring
```

```
        Type number of item from the below list
        1. Disable
        2. Enable
         (default is: ENABLE):2
Edit 'Agent (Advanced)' settings? (default is: Yes):n

Edit 'Collection (Advanced)' settings? (default is: Yes):n

Will this agent connect to a TEMS? [YES or NO] (Default is: YES):y
TEMS Host Name (Default is: srv178): lima
Network Protocol [ip, sna, ip.pipe or ip.spipe] (Default is:
ip.pipe):ip.pipe
     Now choose the next protocol from one of these:
     - ip
     - sna
     - ip.spipe
     - none
Network Protocol 2 (Default is: none):none
IP.PIPE Port Number (Default is: 1918):1918
Enter name of KDC_PARTITION (Default is: null):

Configure connection for a secondary TEMS
? [YES or NO] (Default is: NO):NO
Enter Optional Primary Network Name or "none" (Default is: none): none
Agent configuration completed...
```

After finishing the configuration, we start the Tivoli Enterprise Monitoring Agent using the **./itmcmd agent start yn** command.

> **Note:** Be sure that you configured the data collector to work with the Tivoli Enterprise Monitoring Agent. You can also check the kwjdc.properties file in the data collector runtime directory.

2.6 Initial configuration and operation

In this section we discuss the following topics:

- 2.6.1, “Configuring the data collector” on page 58
- 2.6.2, “Defining operators” on page 63

2.6.1 Configuring the data collector

When a data collector connects to the managing server, it does not start monitoring immediately. It only starts the controller communication to the managing server. The managing server then starts the probe based on the configuration that it knows. You must perform these tasks:

1. Assign a configuration for the data collector. See “Configuring the data collector” on page 58.
2. Define the monitoring level and sampling rate for the data collector. See “Monitoring level and sampling rate” on page 59.
3. Assign the data collector to a server group. See “Server group considerations” on page 60.

Configuring the data collector

A configuration is a profile that provides a specific monitoring definition for the data collector. Select **ADMINISTRATION** → **Server Management** → **Data Collector Configuration** → **Unconfigured Data Collector**. Assign the appropriate default profile, as shown in Figure 2-19. Usually these default profiles are adequate for your monitoring needs. For additional customization, use the configuration files on the data collector.

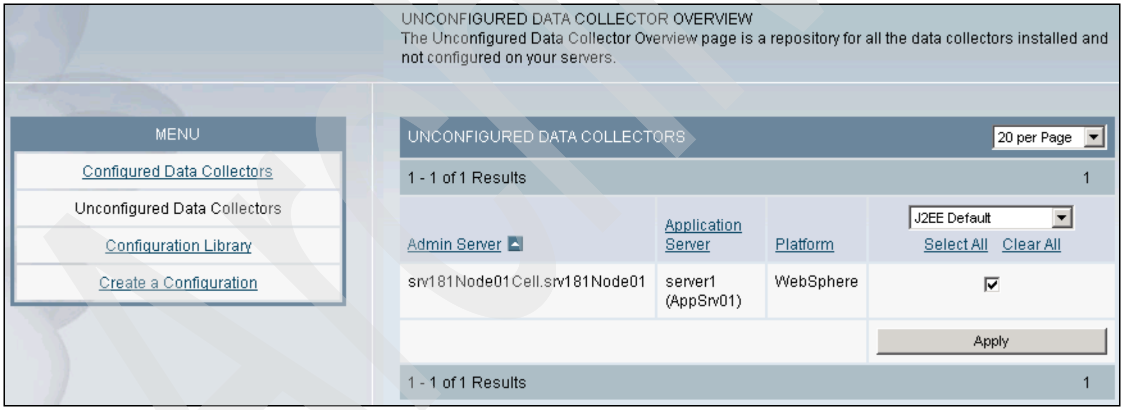


Figure 2-19 Data collector configuration

Monitoring level and sampling rate

For the default monitoring level and sampling rate, select **ADMINISTRATION → Managing Server → System Properties** (Figure 2-20).

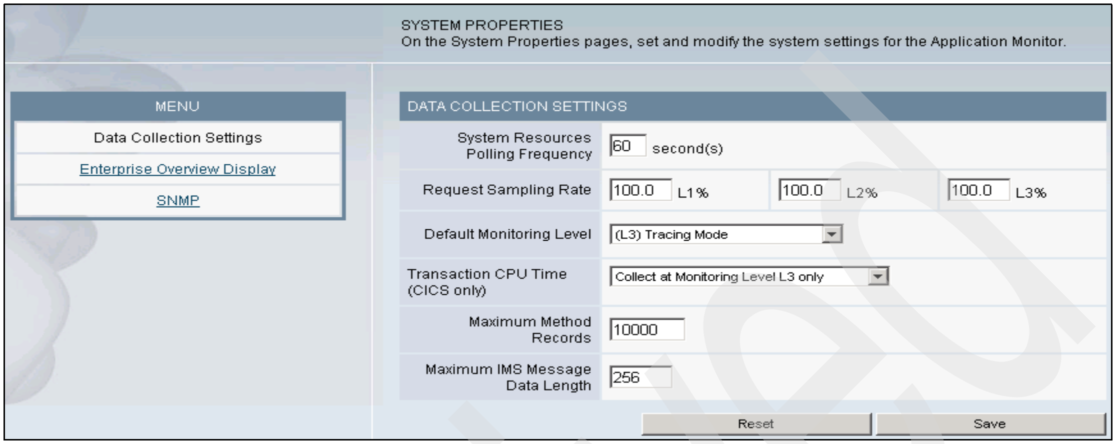


Figure 2-20 Default monitoring

Select **ADMINISTRATION → Monitoring On Demand®** to specify the overall defaults and provide a specific level for each server (Figure 2-21). For further information about those settings, see 2.3.1, "Sizing consideration" on page 27.

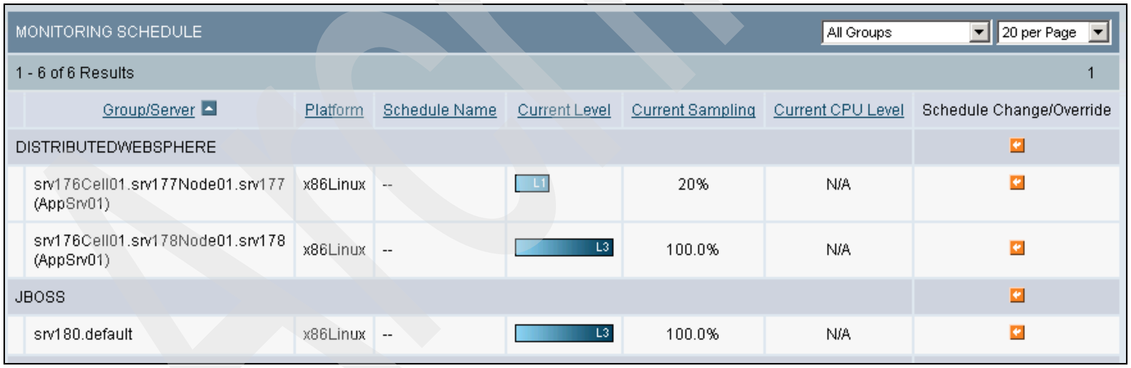


Figure 2-21 Monitoring on demand

Server group considerations

Server groups consolidate information from several servers and increase efficiency in handling ITCAM for WebSphere and ITCAM for J2EE. Examples where server groups help are the enterprise overview page and report generations are:

- The enterprise overview enables users to understand the behavior of their application servers at a glance, by graphically displaying the throughput and response time of server groups. In this case, grouping servers helps maintain the overview character of this workspace in multiserver environments.
- Using server groups for report generation enables you to run reports against a combined group of servers, in contrast to having the report run against every single server.
- Another option server grouping provides is more granular authorization, because you can grant access to servers at the group level.
- Server grouping is also helpful if you want to change the monitoring level of several servers manually at once.

ITCAM for WebSphere and ITCAM for J2EE can group the servers and aggregate information from all servers in various ways. Examples of server grouping are grouping by cell or node, operating system, physical location, or responsibility. For instance, you group the servers according to the group of people responsible for managing those servers and grant access to the appropriate group of servers only to the various teams.

Figure 2-22 shows an example of user distrTr with limited access to servers, in this case authorized only for servers in the TraderIMSdistr group.

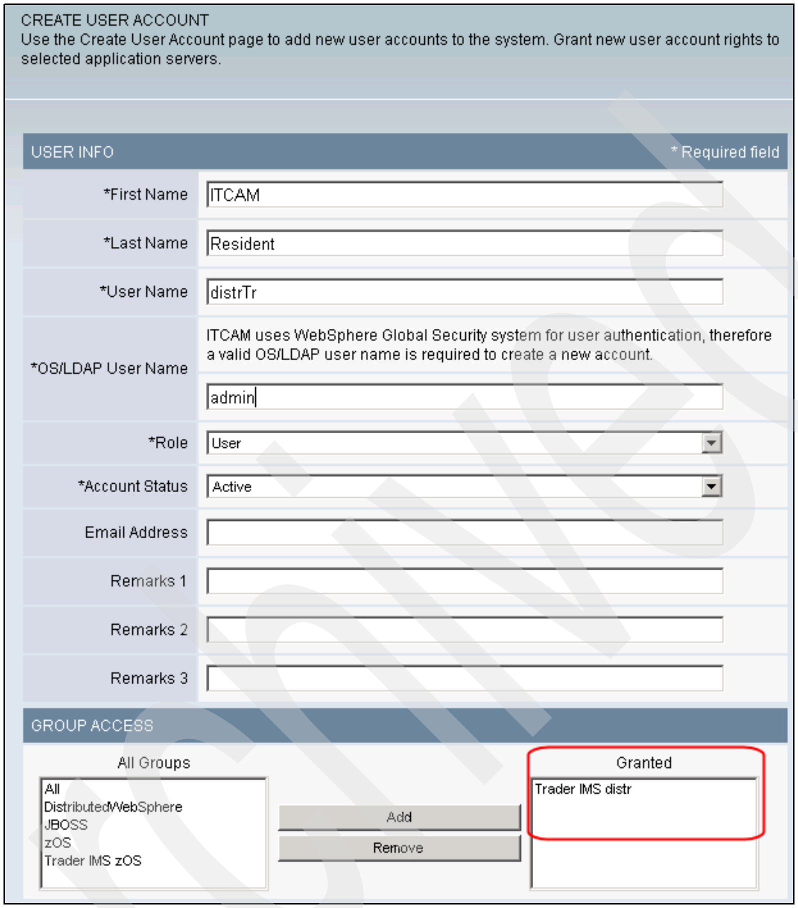


Figure 2-22 Create user account

TraderIMSdistr in Figure 2-22 is a server group in which we grouped servers by application. ITCAM for WebSphere and ITCAM for J2EE allow a server to be a member of several groups. This enables you to create server groups from an application point of view and for transactions that share servers with other applications. In our lab environment, we have several WebSphere Application Servers access the same CICS and IMS systems, and those servers are members of several groups. For example, the IMS server is part of group IMS distributed and group IMS zOS.

Note: Using *shared* servers in server groups might cause misinterpretations in the enterprise overview. Transactions are not distinguished at the component level. For example, every server group containing the same shared CICS server sees all transactions from this CICS server. The number of requests and response time aggregated in the enterprise view also include transactions triggered by servers that are not part of the group.

Figure 2-23 shows an example with servers grouped from an application perspective. Note the patterns for the Trader_IMS_distr and Trader_IMS_distr groups although there have not been any IMS activities during this period of time. The reason for this is that we run both CICS and IMS transactions on the same WebSphere Application Server.

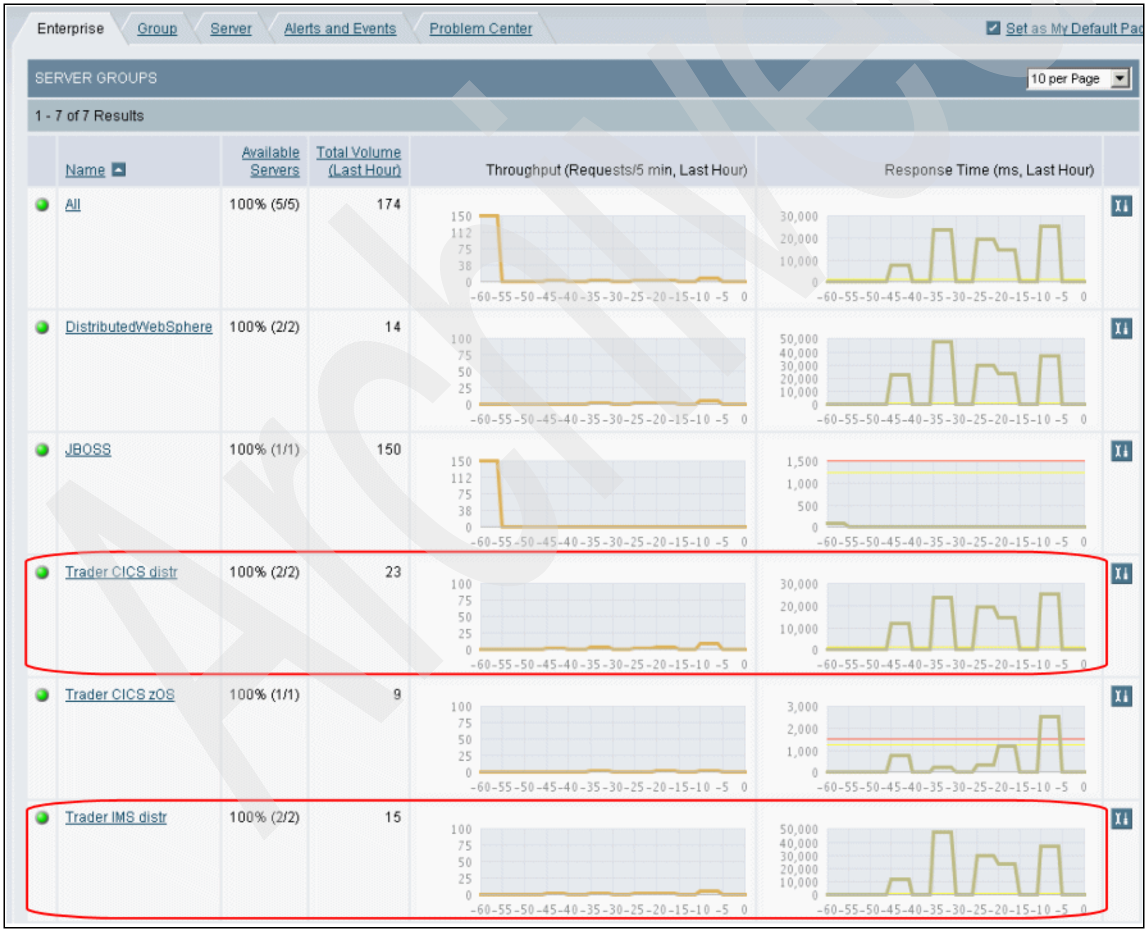


Figure 2-23 Server groups

To find the most helpful server grouping, you might consider:

- The number of servers that you want to monitor: Does grouping make sense?
- Organization and processes in your environment: Who is responsible for what servers and who needs to have access to what server information?
- The way you use reporting: What reports do you use, how many of your reports are manual reports, and how many scheduled reports are there?

If you have a shared server environment as described in this section (depending on your requirement of accuracy in the enterprise overview), grouping servers from an application point of view might be an option.

2.6.2 Defining operators

You must define the operators to use ITCAM for WebSphere and ITCAM for J2EE. To define the operators:

1. Define the user ID to the operating system where you install the ITCAM for WebSphere and ITCAM for J2EE managing server, using one of the following methods:
   - Use Computer Management on a Windows system.
   - Use **`smit`** for an IBM AIX 5L™ system.
   - Use the **`adduser`** command on a Linux system.
   - Otherwise, use the appropriate tools for your server platform.

   Our managing server is a Linux box, so we define the additional user oper1 using the command **`adduser oper1`**, and then we assign a password for oper1 using the **`passwd oper1`** command. You need to log on to the system to change the password, because the password from the **`passwd`** command expires immediately.

2. Define the operator in ITCAM for WebSphere and ITCAM for J2EE using the Web console. Select **ADMINISTRATION** → **Account Management** → **User Profiles** and create the oper1 user, as shown in Figure 2-24.

Figure 2-24 Creating the operator oper1

In Figure 2-24, the operator name in ITCAM for WebSphere and ITCAM for J2EE and the operating system logon do not have to be the same. You can have many ITCAM for WebSphere and ITCAM for J2EE operators defined to the same operating system user. This might be confusing for maintenance purposes, so we recommend using a unique operating system user for each ITCAM for WebSphere and ITCAM for J2EE operator. However, the ITCAM for WebSphere and ITCAM for J2EE operators need to be the same as the operating system logon to avoid confusion.

3

ITCAM for WebSphere and ITCAM for J2EE usage

This chapter demonstrates the use of IBM Tivoli Composite Application Manager for WebSphere in our environment. The discussion includes:

- 3.1, “Usage scenarios” on page 66
- 3.2, “Enterprise monitoring” on page 67
- 3.3, “Server activity analysis” on page 72
- 3.4, “Method profiling” on page 74
- 3.5, “Portal performance reports” on page 79
- 3.6, “Transaction reporting” on page 87
- 3.7, “Memory diagnosis” on page 96
- 3.8, “Lock analysis” on page 108
- 3.9, “JBoss Application Server data collector” on page 124
- 3.10, “Composite transaction analysis” on page 135
- 3.11, “Using Tivoli Enterprise Portal” on page 143
- 3.12, “Historical information and reporting” on page 148

3.1 Usage scenarios

ITCAM for WebSphere and ITCAM for J2EE are solutions that primarily aim at second-line support to perform diagnosis of J2EE-based applications and services. This chapter discusses some usage scenarios for ITCAM for WebSphere and ITCAM for J2EE within our environment and explores the new functions of ITCAM for WebSphere and ITCAM for J2EE. The scenarios here consist of:

- Working with the ITCAM for WebSphere and J2EE Web console to monitor the environment. We discuss this in 3.2, "Enterprise monitoring" on page 67. Server groups influence the presentation on the enterprise overview and operator assignment, affecting the display.
- Monitoring J2EE application server activity, discussed in 3.3, "Server activity analysis" on page 72. Here we explore some examples of J2EE application server activity.
- Transaction analysis and reporting, discussed in 3.6, "Transaction reporting" on page 87. Trend reports and top reports are the two most common reports used for analysis. Trend reports show how transactions behave, and top reports show potential bottlenecks in the system.
- Analyzing composite applications, discussed in 3.10, "Composite transaction analysis" on page 135, shows the result of having IMS and CICS transactions as part of the J2EE-based application.
- Working with Tivoli Enterprise Portal information, discussed in 3.11, "Using Tivoli Enterprise Portal" on page 143.
- Historical data collection with Tivoli Data Warehouse V2.1, discussed in 3.12, "Historical information and reporting" on page 148.

For additional scenarios and usage, refer to *IBM Tivoli Composite Application Manager for WebSphere Usage Guide*, GC32-1934, which includes in-depth discussions about memory analysis, locking analysis, and other advanced analysis scenarios.

It is important to consider how the product will be used in your environment and who will use it. The solution produces reports that can be relevant to operational managers and service or application managers. Identifying the roles and responsibilities is important because this affects the customization of ITCAM for WebSphere and J2EE, especially for such things as server groups.

3.2 Enterprise monitoring

When you initially log on the ITCAM for WebSphere and J2EE managing server, you see the availability display. This display shows all data collectors that you have access to with the following level of detail:

Enterprise	Shows the overview of all server groups to which you have access
Group	Shows all servers within a particular server group
Server	Shows the overview page of an application server
Alerts and Events	Shows all alerts and events in your environment
Problem Center	Views problems in your environment

Figure 3-1 shows our enterprise overview.

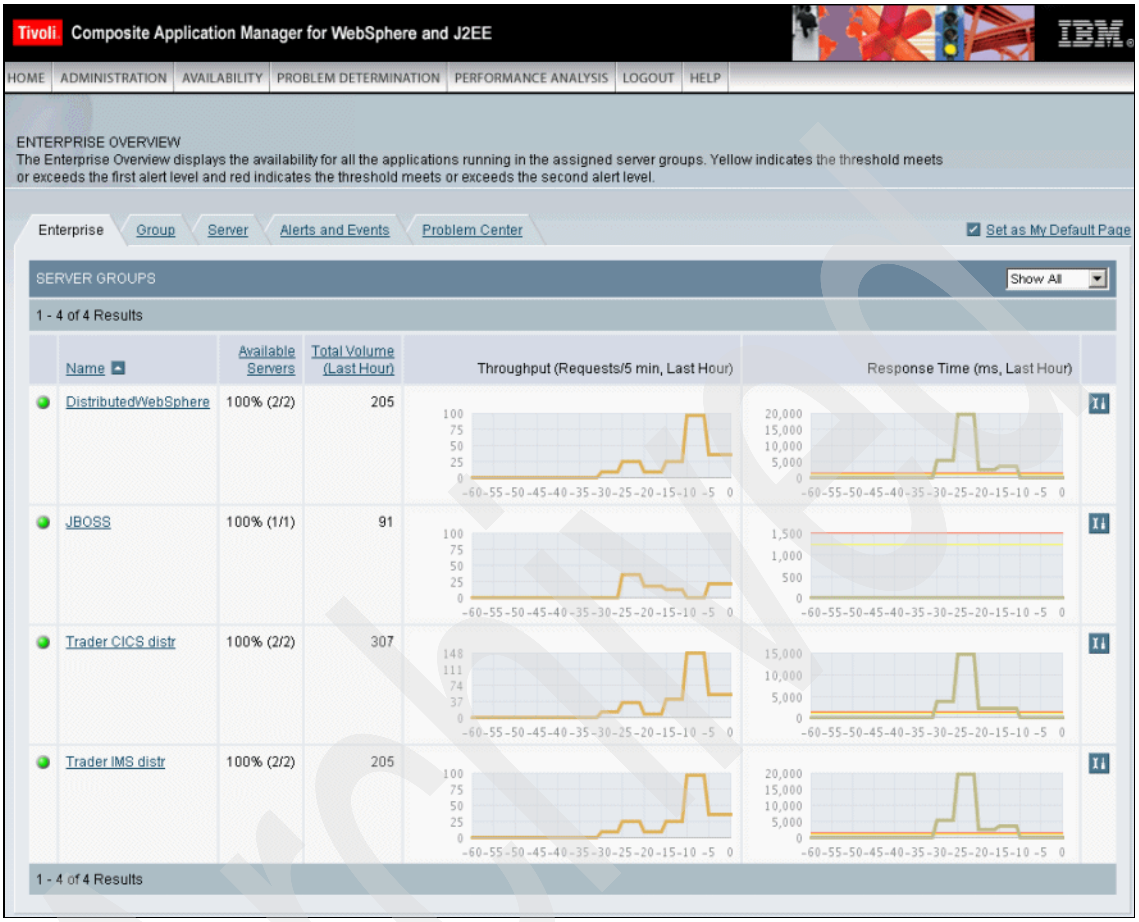


Figure 3-1 Enterprise overview

The enterprise overview shows the transaction volume and response time for each server group. The transaction volume is the total of all transaction volumes for each server in the server group, and the response time shows the average response time across all servers in the group. This level of display hides the fact that each application server can have a different workload profile. This group display is beneficial only if you have an application server cluster with a load balancer where each application server is identical with a similar load profile.

Our enterprise overview in Figure 3-1 shows that we have four server groups. The second column, available servers, shows how many of the servers in this group are available out of a maximum. In the server group JBOSS, there is a maximum of one server defined, and it is available.

Figure 3-1 on page 68 shows the yellow and red horizontal lines that indicate warning and critical response time thresholds. These settings are defined in the server group level that can be modified from **ADMINISTRATION** → **Server Management** → **Server Groups**. Figure 3-2 shows an example for DistributedWebSphere.

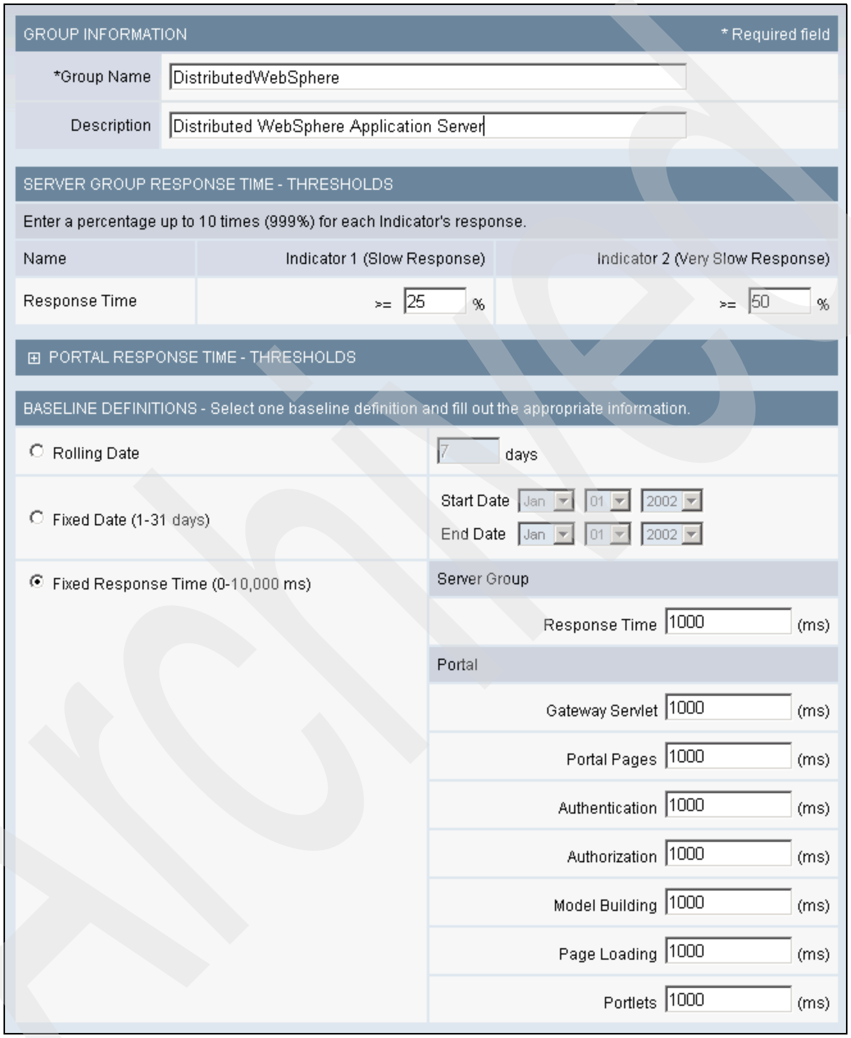


Figure 3-2 Modify server group

The threshold is defined as the deviation from a baseline performance. You can use an average from a rolling period of the past few days, a reference period, or a hard-coded response time threshold. In Figure 3-2, we use a fixed baseline of 1000 ms.

Be aware that neither the yellow nor the red markers drive the issue of alerts. They are only a visual indicator on this display. If alerts are needed, you can use the ITCAM for WebSphere trap-and-alert facility.

The Enterprise Overview shown in Figure 3-1 on page 68 reveals that the response time graph for the TraderCICSdistr group is consistently higher than both markers and suggests that either the marker needs adjusting or a genuine performance problem exists. Because we intend to investigate using the Server Group view, click the relevant server group, such as **DistributedWebSphere**, to display the group, as shown in Figure 3-3.

Figure 3-3 Group overview

This page shows the overview of each server in the server group, including the transaction volume and average response time for each application server. This page also shows whether the data collector in the application server is online.

The DistributedWebSphere server group is very simple. It consists of two different types of servers. Consider it a line-of-business type view. Server groups defined as containing different server types tend to be useful for reporting and diagnosing bottlenecks in one of the servers.

In our example, the throughputs through both srv176 and SCSCBUD1 are quite low. Therefore, the high response time is not caused by the high volume of transactions.

A finer level of granularity is available by selecting a server from this group. The server summary display is similar to that shown in Figure 3-4.

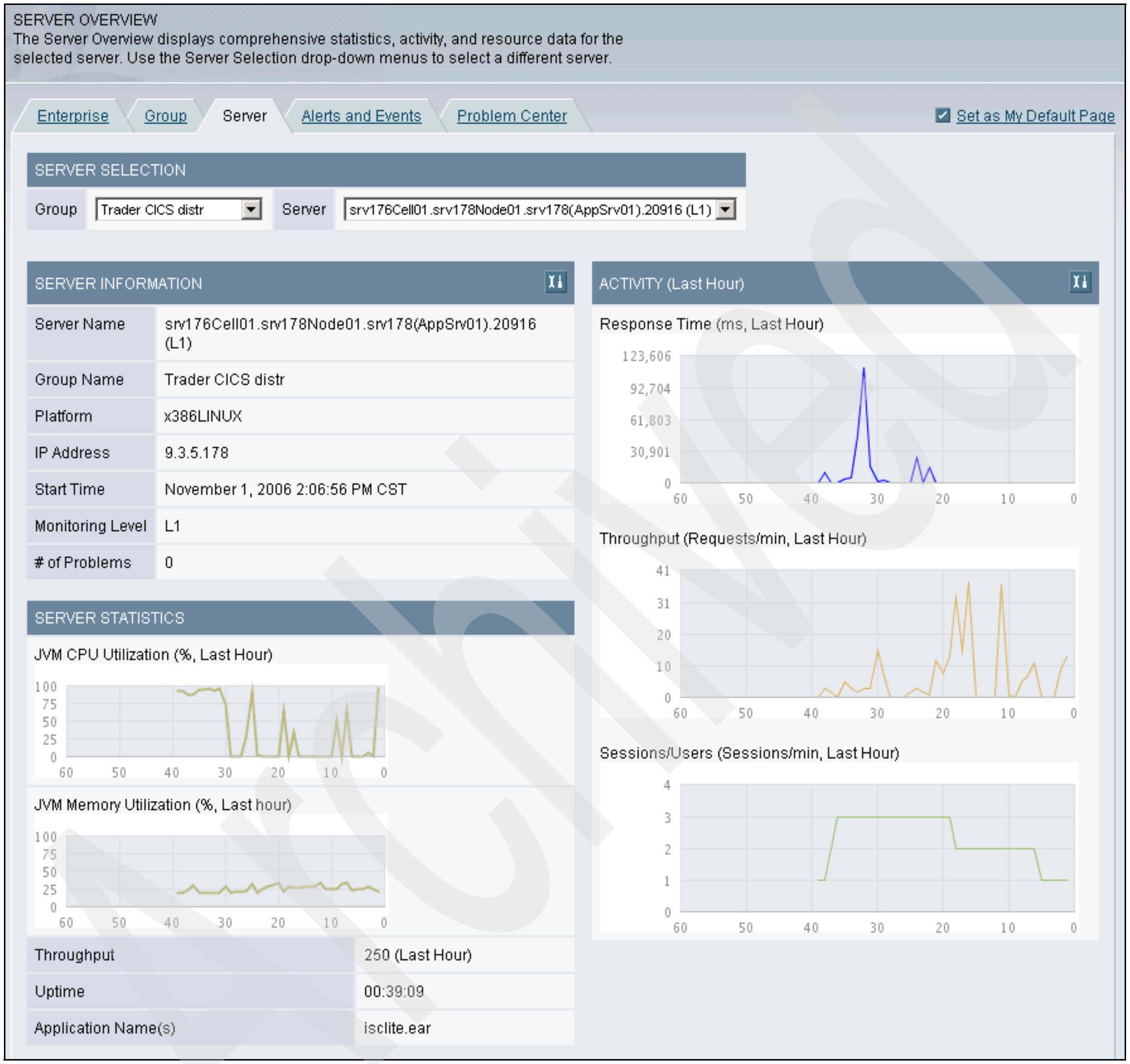


Figure 3-4 Server overview

The detailed server view is most likely to be used by second-line support teams to diagnose application issues and narrow down possibilities to the root cause.

Note: These availability pages automatically refresh every 60 seconds. They do not time-out the user. This can be an issue in environments that are particularly security conscious.

3.3 Server activity analysis

When we start work or zoom in to a server for more detailed analysis, typically we start with the server overview, as shown in Figure 3-4 on page 71. You might also want a snapshot of the activity of the server using the server activity display by selecting **PROBLEM DETERMINATION** → **Server Activity Display**.

In loading the page in Figure 3-5, the visualization engine requests a snapshot of information directly from the data collector. This can be a costly operation.

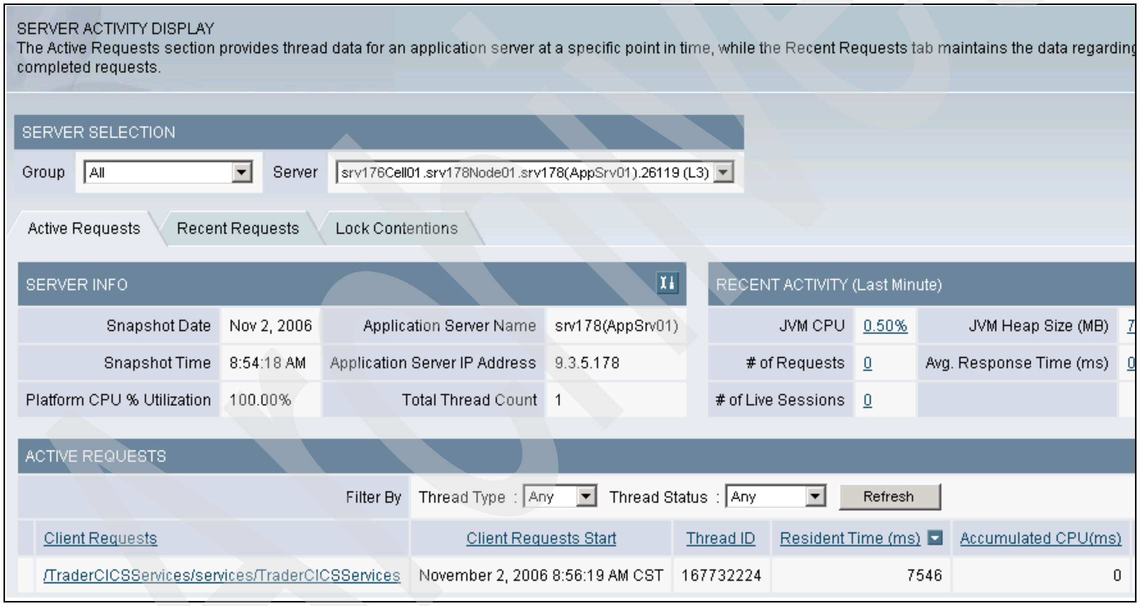


Figure 3-5 Server activity display

In our sample display, we capture an active transaction under the Active Requests tab. This panel is useful for support personnel to identify long-running requests, usually indicated by a high resident and idle time, or perhaps indicating looping code if the accumulated CPU is high.

Select the **Recent Requests** tab to view a snapshot of requests from the data collector, as shown in Figure 3-6. By default, this is limited to 100 entries. In some live environments, this is very recent data and might not give a long enough time frame to diagnose properly.

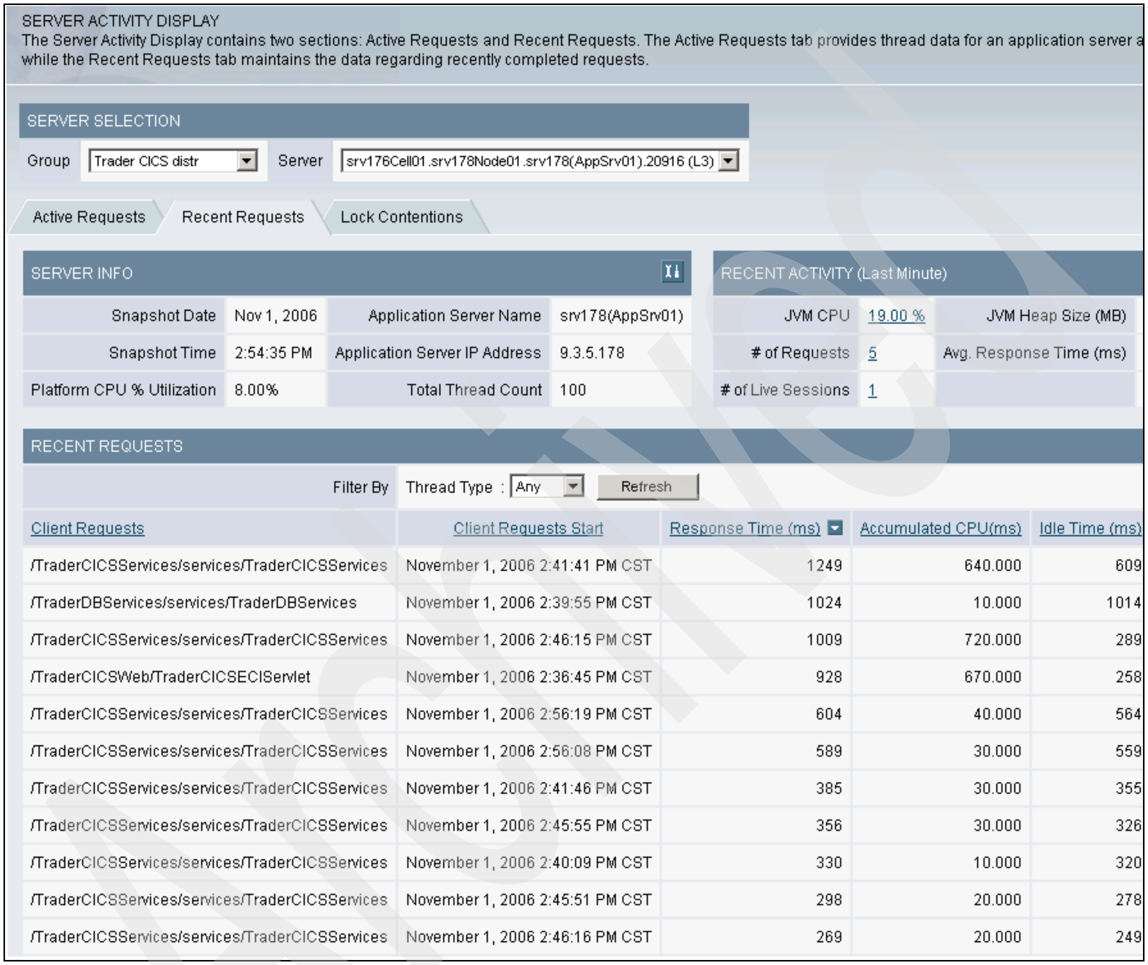


Figure 3-6 Recent Requests tab

If you need more data, retrieve the transaction history using the request/transaction report by selecting **PERFORMANCE ANALYSIS** → **Create Application Reports** → **Request/Transaction**. See some reports in 3.6, “Transaction reporting” on page 87.

You can also invoke the Server Activity Display shown in Figure 3-5 on page 72 by clicking the tool icon shown in Figure 3-7. Several other functions are also available for diagnosing problems.

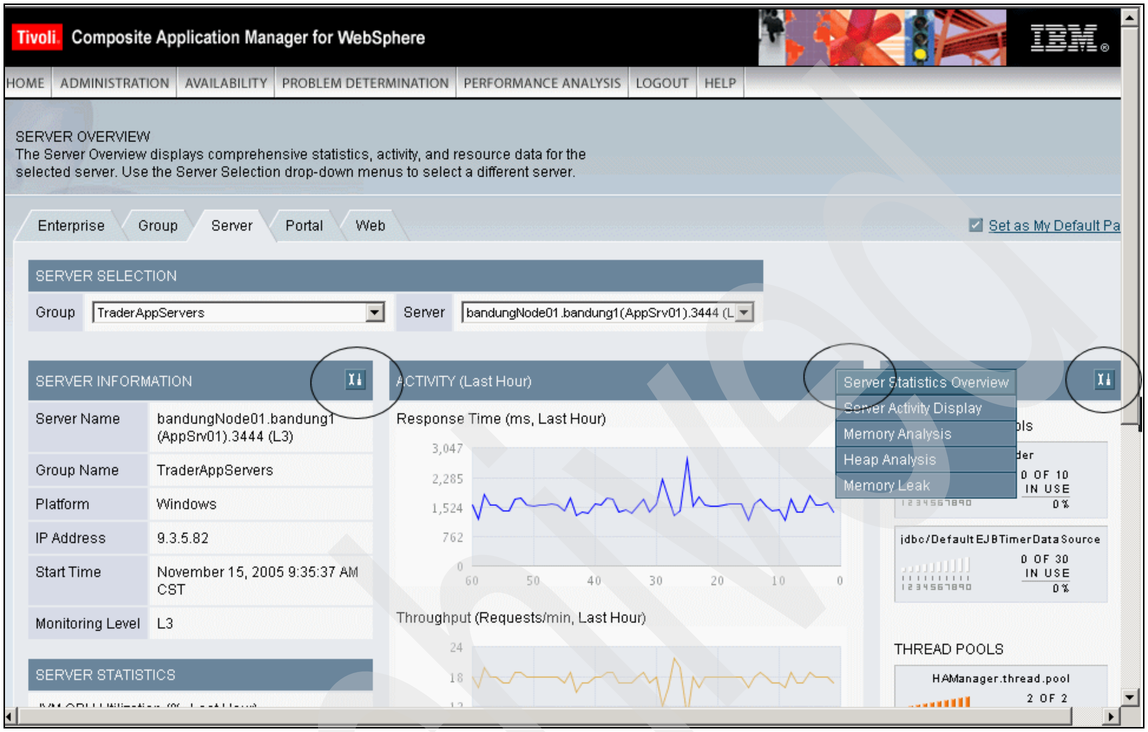


Figure 3-7 Tool icons to launch server activity display

3.4 Method profiling

Method profiling offers a means of gathering aggregate data of an application's method profile at monitoring level 2 rather than level 3. These metrics include total CPU time, total elapsed time, total hits, average CPU time, and average elapsed time spent in each method. These metrics may be used as a first step to understanding the behavior of the application.

To enable method profiling, perform the following steps:

1. On the data collector, modify <DC_HOME>/runtime/app_server_version.node_name.server_name/custom/toolkit_custom.properties to enable method entry exit trace, as shown in Example 3-1. Then restart the application server.

Example 3-1 Modifications required in toolkit_custom.properties

```
am.camtoolkit.gpe.customxml.L3=/opt/IBM/itcam/WebSphere/DC/itcamdc/e
tc/method_entry_exit.xml
com.ibm.tivoli.itcam.toolkit.ai.methodentryexittrace=true
```

> **Note:** The default behavior instrumenting the method entry and exit trace is to instrument all classes and methods. Once you have identified which classes or methods you need to drill down on, modify the <DC_HOME>/itcamdc/etc/method_entry_exit.xml to include the class and enable L3 tracing.
>
> The <DC_HOME>/itcamdc/etc/method_entry_exit.xml is used for all the application servers that you have instrumented and enabled. To set a specific filter for a specific set of classes or methods for a particular application server, copy the <DC_HOME>/runtime/app_server_version.node_name.server_name/custom file, modify the filters, and modify the path in toolkit_custom.properties to refer to it.

2. From the Menu option select **ADMINISTRATION** → **Monitoring On Demand**.

3. Figure 3-8 shows all the available servers and their corresponding trace levels. In order to modify the trace level, select **Schedule Change/Override**.

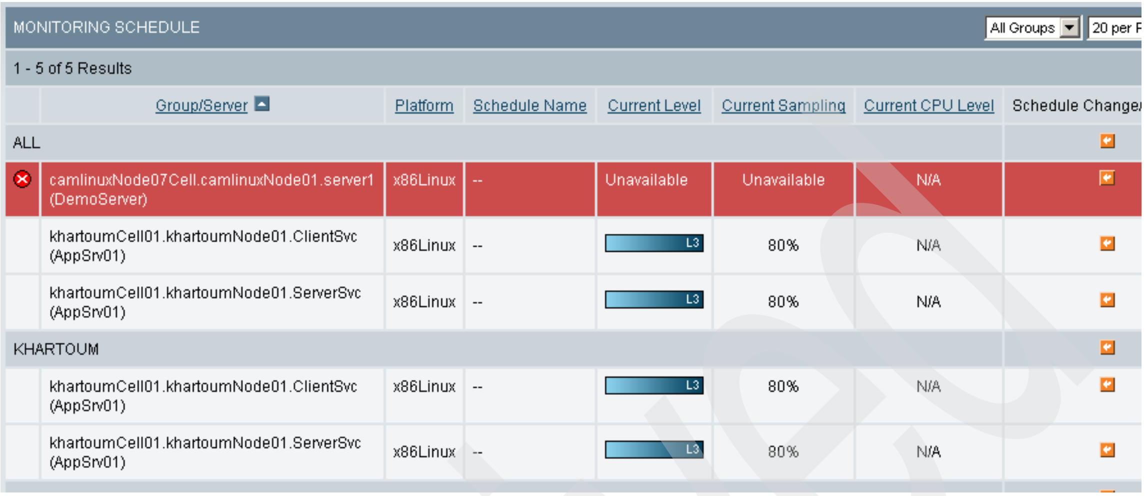


Figure 3-8 Monitoring on demand console

4. Override the monitoring level by selecting **(L2) Problem Determination Mode**, tick the Enable Method Profiling check box, and set the interval to reset and publish the statistics. Figure 3-9 depicts an example of enabling method profiling at a frequency of 5 minutes and a sampling rate of 80%.

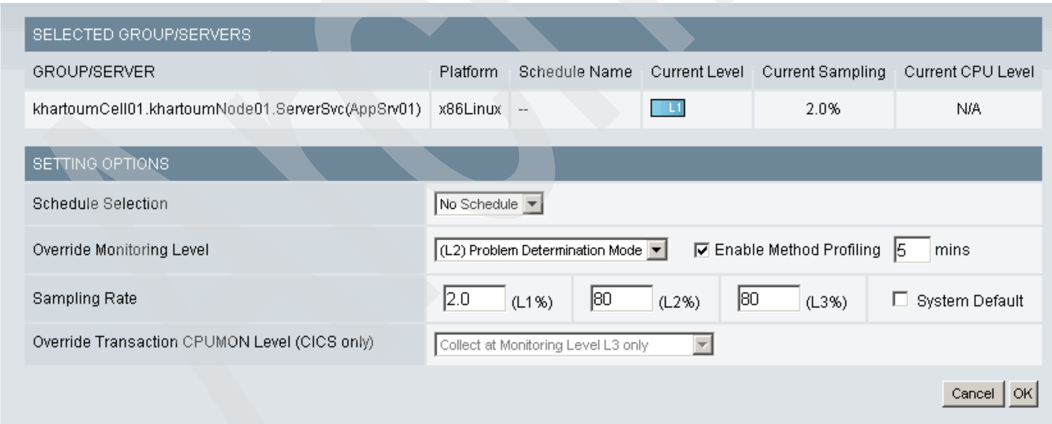


Figure 3-9 Modify server settings

5. Finally select **OK** to apply the changes.

Note: The update of the level change to the monitoring on demand console is not instantaneous.

6. To access profile method management select the menu option **Performance Analysis** → **Method Profiling**. Figure 3-10 shows the available method profiles that has been captured during a 5-minute interval. Click the highlighted link to select the required profile report.

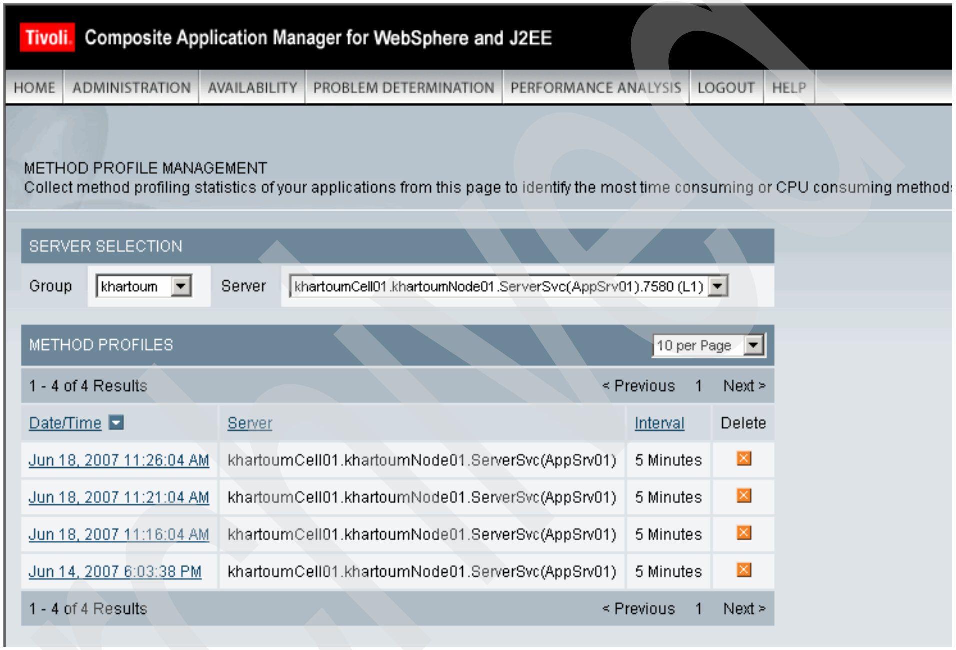
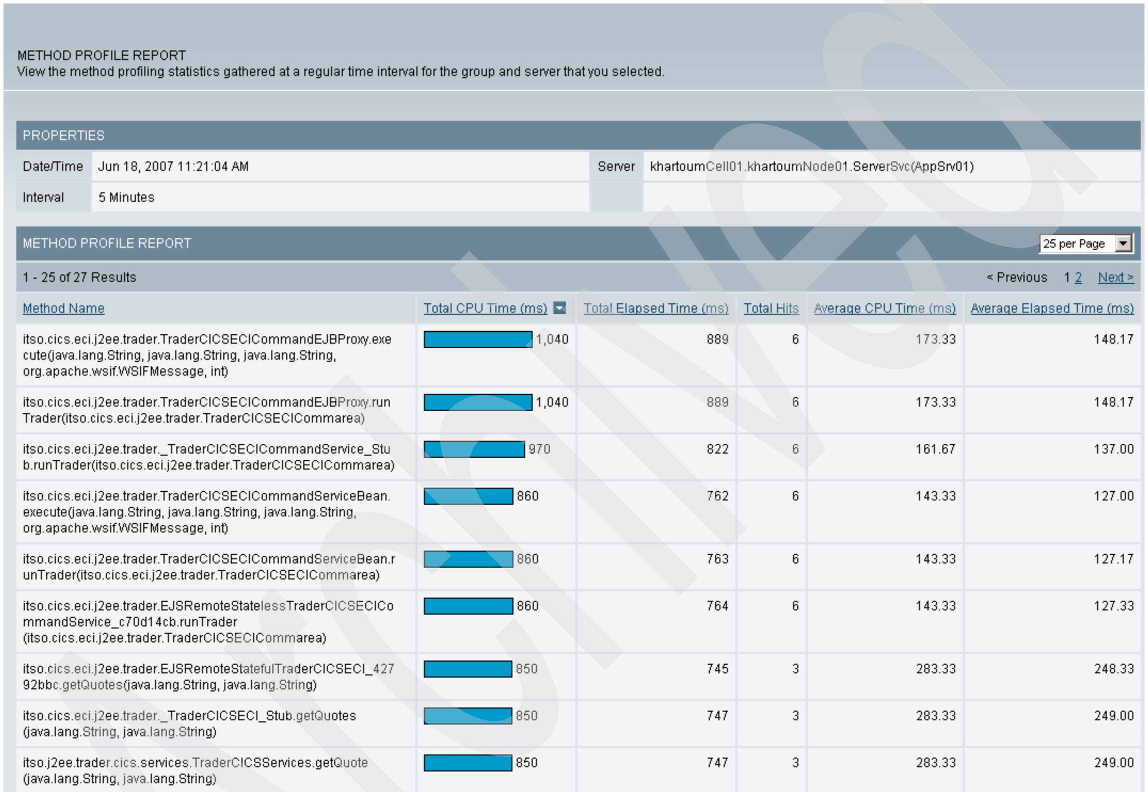


Figure 3-10 Method profile management

7. Figure 3-11 shows the available statistics for the method profiling report. These include the total CPU time, total elapsed time, total hits, average CPU time, and average elapsed time per method. In this example we see that the method that utilizes the most CPU time is itso.cics.eci.j2ee.TraderCICSECICommandEJBProxy.execute and itso.cics.eci.j2ee.TraderCICSECICommandEJBProxy.runTrader. It is possible to re-organize the results by any of the metrics being captured (for example, by total elapsed time, total hits, average cpu time, or average elapsed time).



METHOD PROFILE REPORT
View the method profiling statistics gathered at a regular time interval for the group and server that you selected.

PROPERTIES

Date/Time	Jun 18, 2007 11:21:04 AM	Server	khartoumCell01.khartoumNode01.ServerSvc(AppSrv01)
Interval	5 Minutes		

METHOD PROFILE REPORT — 25 per Page

1 - 25 of 27 Results — < Previous 1 2 Next >

Method Name	Total CPU Time (ms)	Total Elapsed Time (ms)	Total Hits	Average CPU Time (ms)	Average Elapsed Time (ms)
itso.cics.eci.j2ee.trader.TraderCICSECICommandEJBProxy.execute(java.lang.String, java.lang.String, java.lang.String, org.apache.wsif.WSIFMessage, int)	1,040	889	6	173.33	148.17
itso.cics.eci.j2ee.trader.TraderCICSECICommandEJBProxy.runTrader(itso.cics.eci.j2ee.trader.TraderCICSECICommarea)	1,040	889	6	173.33	148.17
itso.cics.eci.j2ee.trader._TraderCICSECICommandService_Stub.runTrader(itso.cics.eci.j2ee.trader.TraderCICSECICommarea)	970	822	6	161.67	137.00
itso.cics.eci.j2ee.trader.TraderCICSECICommandServiceBean.execute(java.lang.String, java.lang.String, java.lang.String, org.apache.wsif.WSIFMessage, int)	860	762	6	143.33	127.00
itso.cics.eci.j2ee.trader.TraderCICSECICommandServiceBean.runTrader(itso.cics.eci.j2ee.trader.TraderCICSECICommarea)	860	763	6	143.33	127.17
itso.cics.eci.j2ee.trader.EJSRemoteStatelessTraderCICSECICommandService_c70d14cb.runTrader(itso.cics.eci.j2ee.trader.TraderCICSECICommarea)	860	764	6	143.33	127.33
itso.cics.eci.j2ee.trader.EJSRemoteStatefulTraderCICSECI_42792bbc.getQuotes(java.lang.String, java.lang.String)	850	745	3	283.33	248.33
itso.cics.eci.j2ee.trader._TraderCICSECI_Stub.getQuotes(java.lang.String, java.lang.String)	850	747	3	283.33	249.00
itso.j2ee.trader.cics.services.TraderCICSServices.getQuote(java.lang.String, java.lang.String)	850	747	3	283.33	249.00

Figure 3-11 Method profiling report

3.5 Portal performance reports

The portal performance report provides a picture of the behavior within a WebSphere Portal Server. This report can either be by portlets or through a portal pages request.

1. Ensure that:
   - In the data collector configuration, WebSphere Portal Server is selected as the application server type, as shown in Figure 3-12.

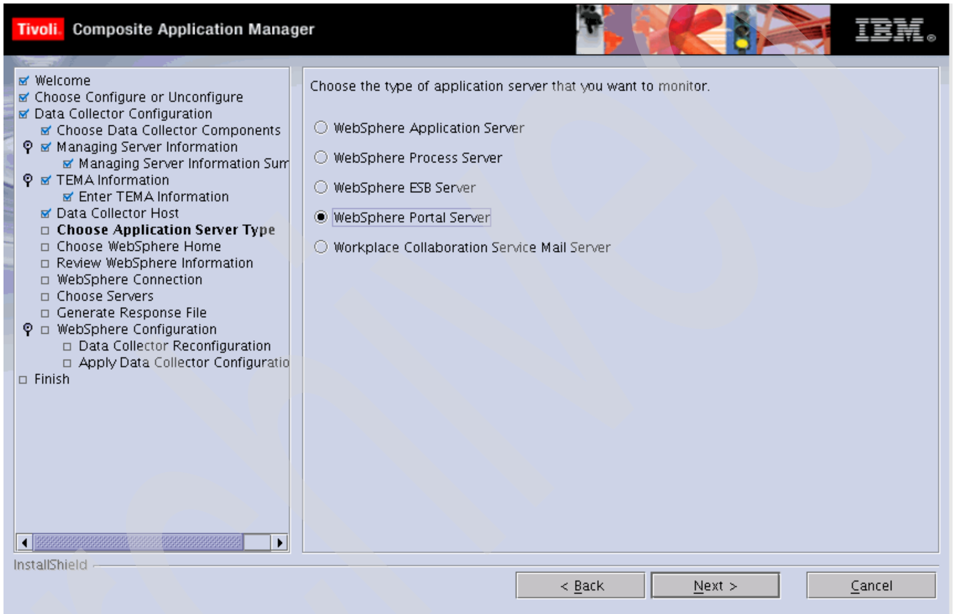
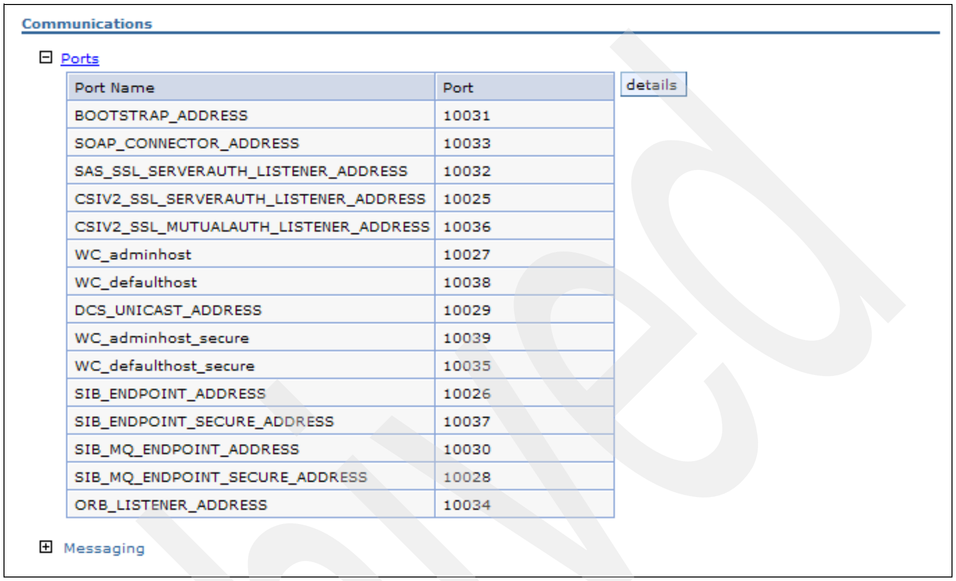


Figure 3-12 Data collector configuration for a portal server

– Ensure that you have the correct SOAP port for the portal server by checking in the WebSphere administration console: **Servers** → **Application Server** → **<WebSpherePortal_Server>** → **Ports**, as shown in Figure 3-13, for the value of SOAP CONNECTOR ADDRESS.



Communications

Ports

Port Name	Port
BOOTSTRAP_ADDRESS	10031
SOAP_CONNECTOR_ADDRESS	10033
SAS_SSL_SERVERAUTH_LISTENER_ADDRESS	10032
CSIV2_SSL_SERVERAUTH_LISTENER_ADDRESS	10025
CSIV2_SSL_MUTUALAUTH_LISTENER_ADDRESS	10036
WC_adminhost	10027
WC_defaulthost	10038
DCS_UNICAST_ADDRESS	10029
WC_adminhost_secure	10039
WC_defaulthost_secure	10035
SIB_ENDPOINT_ADDRESS	10026
SIB_ENDPOINT_SECURE_ADDRESS	10037
SIB_MQ_ENDPOINT_ADDRESS	10030
SIB_MQ_ENDPOINT_SECURE_ADDRESS	10028
ORB_LISTENER_ADDRESS	10034

details

Messaging

Figure 3-13 Ports used by WebSphere Application Server

Use this value when giving the parameters for the WebSphere connection, as shown in Figure 3-14.

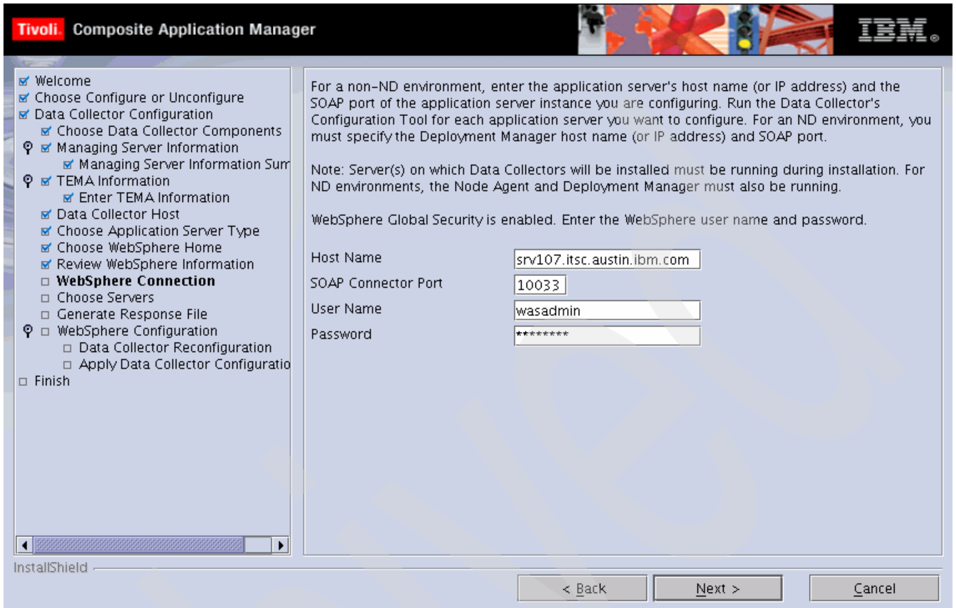


Figure 3-14 Data collector configuration

– <DC_HOME>/runtime/app_server_version.node_name.server_name/custom/toolkit_custom.properties and <DC_HOME>/itcamdc/etc/method_entry_exit.xml and have been modified appropriately to enable method entry exit trace, and the application server is re-started.

– The trace levels have been modified to L3 through the Monitoring on Demand Console for the appropriate server.

2. To invoke the performance report select **PERFORMANCE** → **Portal**. First set the recurrence option.

3. Select the appropriate portal server, as depicted in Figure 3-15. Then select **Next** to proceed to the next option.

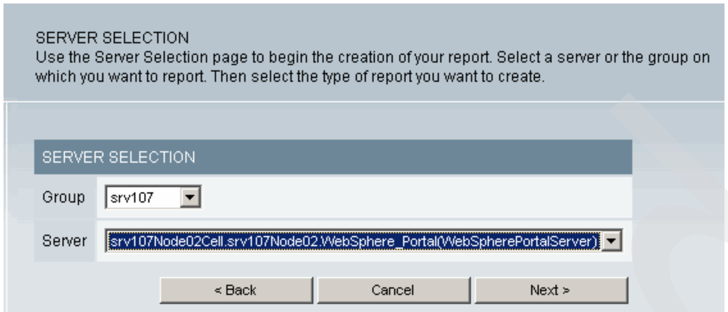


Figure 3-15 Portal server selection

4. Select the report type as depicted in Figure 3-16. Metrics that may be selected include:

   – Throughput per Second
   – Throughput per Minute
   – Throughput per Hour
   – Response Time(ms)
   – CPU Time(ms)

   There are two types of nested request type that may be selected: portlet or portal pages. You can also limit the requests that are being reported by either entering the portal page name or the portlet name.

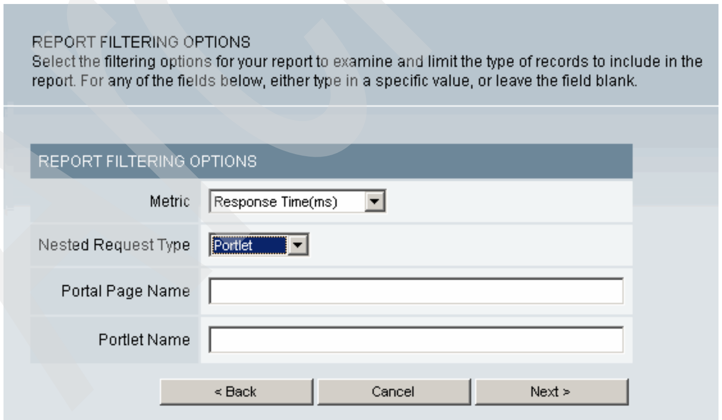


Figure 3-16 Select report type

5. Filters may then be set as depicted in Figure 3-17. The date range maybe selected by selecting the preset options, selecting the start and end date, or setting the advanced filters.
6. Set the graphing options and select **View Report**. The following values may be selected to set for the X-axis:
   - Time Series in Seconds
   - Time Series in Minutes
   - 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

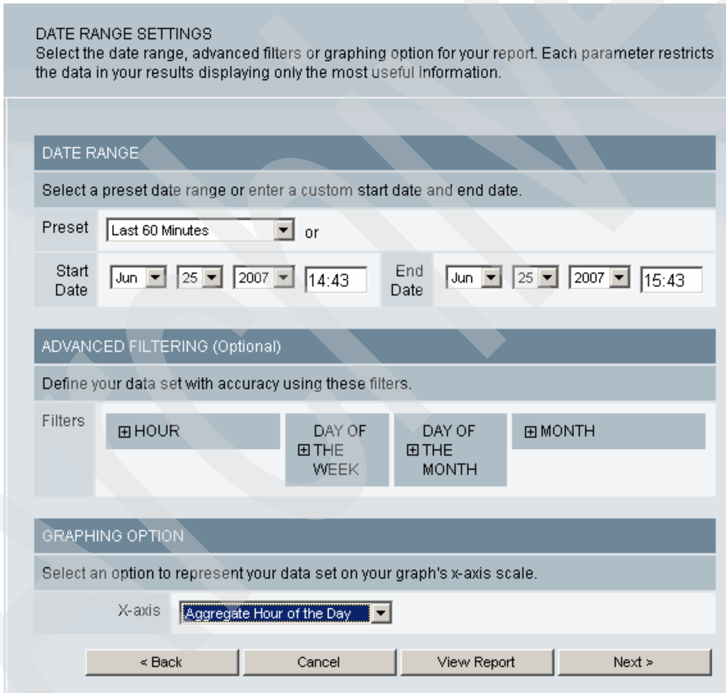


Figure 3-17 Select filters

7. Figure 3-18 depicts the decomposition report. For our scenario we have three portal pages: login, trader, and welcome. For each of the pages we see the response time, the sample count, and the percentage.

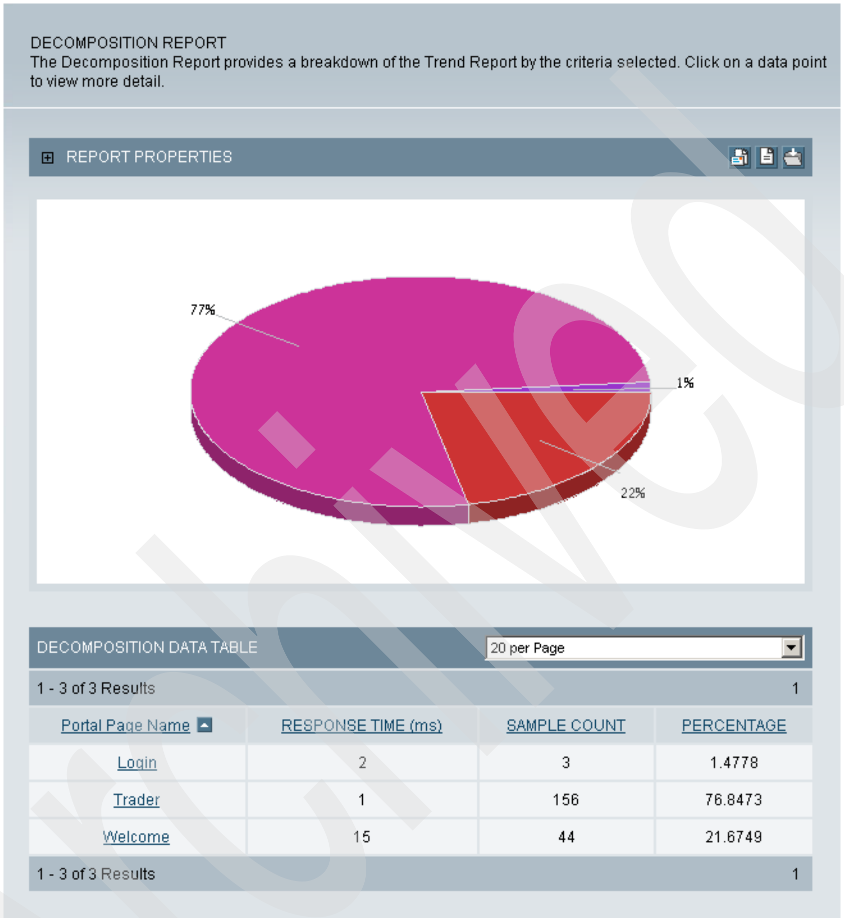


Figure 3-18 Portal decomposition report

8. Figure 3-19 shows a decomposition of a portlet to the method and jdbc calls. Here we can see the data source name and sql statement being used. At the top of table is the nesting summary detailing the total, average response time (ms), and average CPU time (ms) for the servlet, JDBC, portlet, lock acquisition, lock release, portal request, and page rendering.

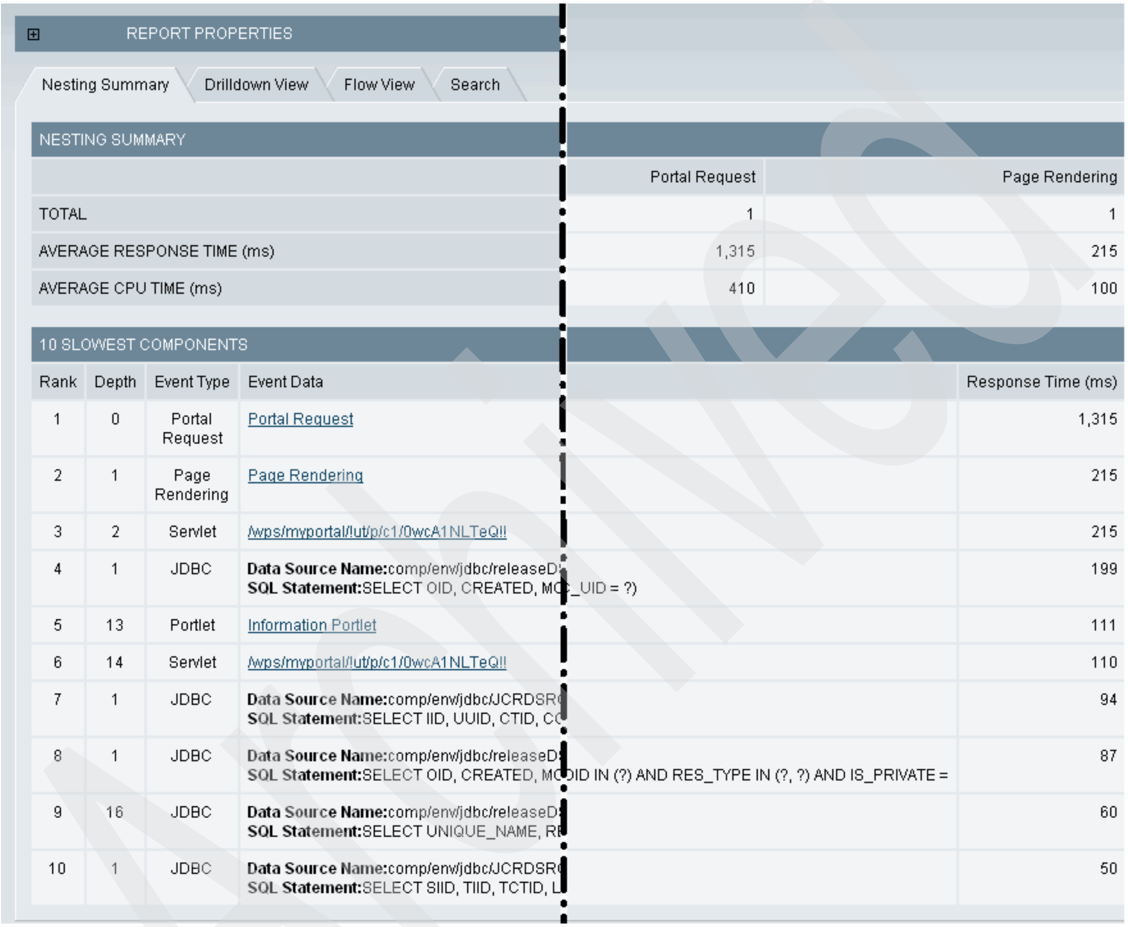


Figure 3-19 Decomposition to the request level

9. Drilling down further, we can see the details of the methods being called, the duration, and the CPU time. Figure 3-20 shows the decomposition to the method entry. The Search tab offers the user the ability to set a filter and only view specific methods.

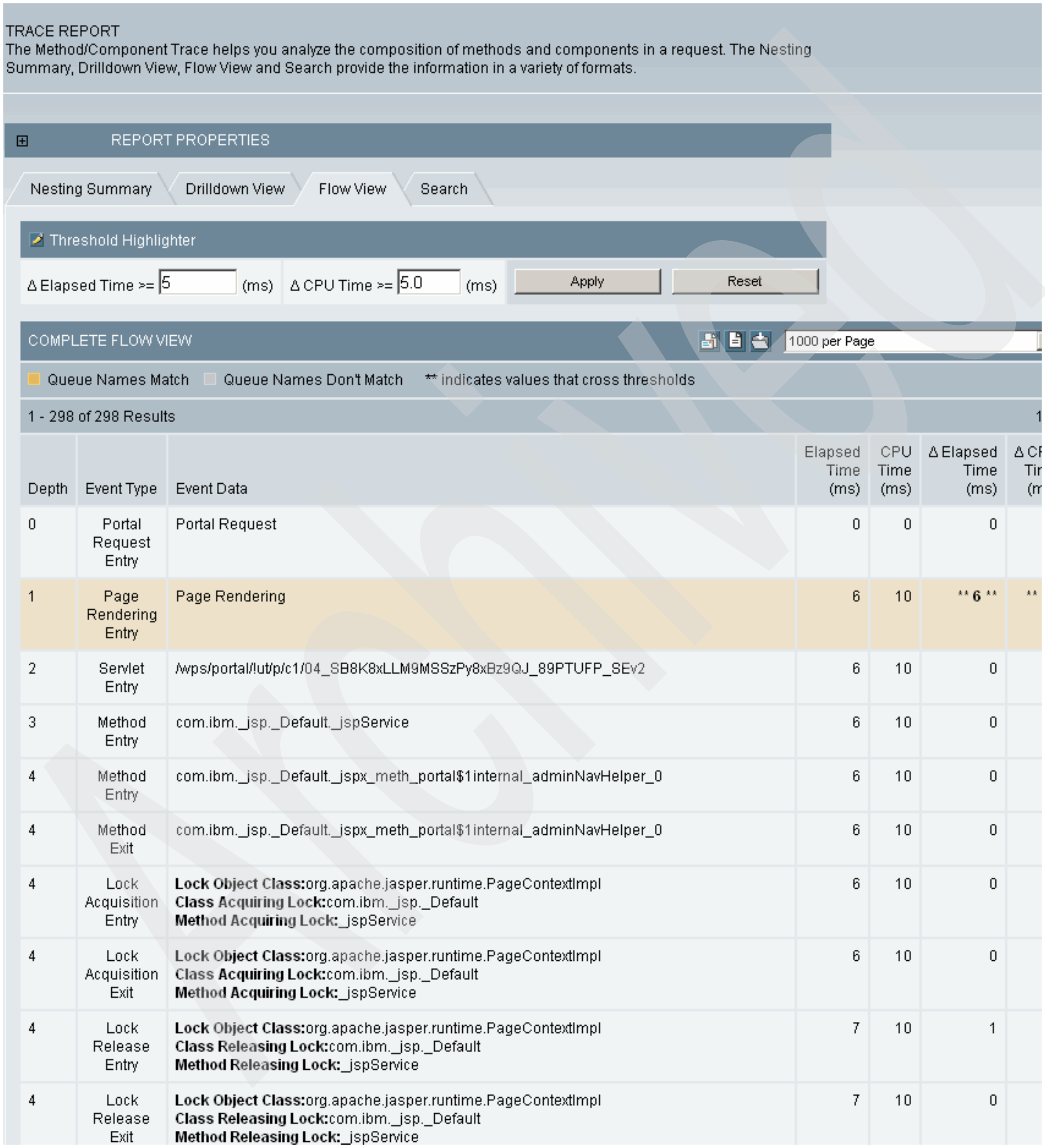


Figure 3-20 Decomposition to the method level

3.6 Transaction reporting

In 3.3, "Server activity analysis" on page 72, we analyze views in real time from snapshot data directly from the data collector. This section discusses reporting from stored data. This stored data is transaction information that is passed from the publish server to the archive agent. The transaction shown here is subject to filtering based on the sampling rate specified on the application server data collector setting. ITCAM for WebSphere and ITCAM for J2EE reporting is based on data stored in the OCTIGATE database.

3.6.1 Top reports

We looked at the reports provided by ITCAM for WebSphere and ITCAM for J2EE from the perspective of how to identify application bottlenecks quickly and to get a better understanding of the application flow. Begin with top reports to identify the top players in your environment. In the development phase, these reports might be run at regular intervals to understand application behavior and pinpoint possible bottlenecks.

The top reports delivered with ITCAM for WebSphere and ITCAM for J2EE and the required monitor levels are:

Requests Used	Most-called requests with the number of calls (L1)
Methods Used	Most-called methods including the number of calls (L3)
Slowest Requests	Slowest requests and their average response time (L1)
Slowest Methods	Slowest methods and their average response time (L3)
CPU-Intensive Requests	Requests sorted by total CPU time (L1)
CPU-Intensive Methods	Methods sorted by total CPU time (L3)
SQL-Intensive Requests	Requests sorted by number of SQL calls (L2)
SQL-Intensive Methods	Methods sorted by number of SQL calls (L3)
Tables Used	The database tables called most often and the number of calls (L1)
SQL Used	Top five SQL call types and number of calls (L2)

A customer example of how top request reports can help is the appearance of the authentication method with an unexpectedly high number of requests. Based on this report, the customer saw redundant calls in the authentication process and lowered resource usage by redesigning the application.

Figure 3-21 shows an example of a *top requests used* report from our lab environment. This report looked as we expected it to, so we did not look for more details in this case.

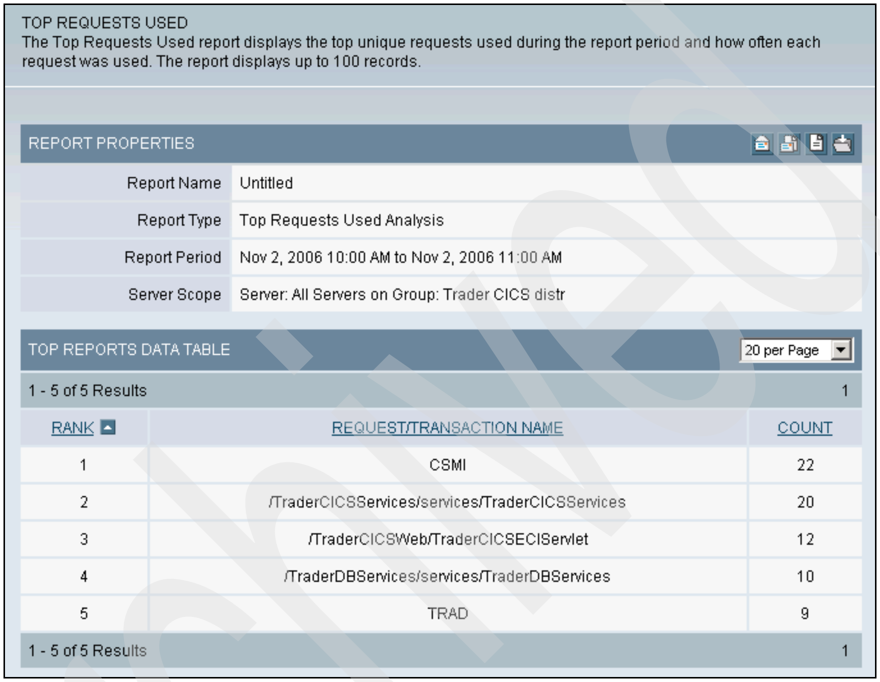


Figure 3-21 Top reports: top requests used

3.6.2 Problem investigation using reports

Here is an example of how we used application reports to check the behavior of our test application. We started with top reports for an overview and followed suspicious data down to the method level by using request/transaction reports.

> **Note:** The vast majority of the reports we used to narrow down this problem are based on L1 information. The only time L3 information is required is in the flow view used to identify the method.

In Figure 3-22, the report helps you figure out the requests with the slowest response time and helps you start further diagnosis in case of unexpected results.

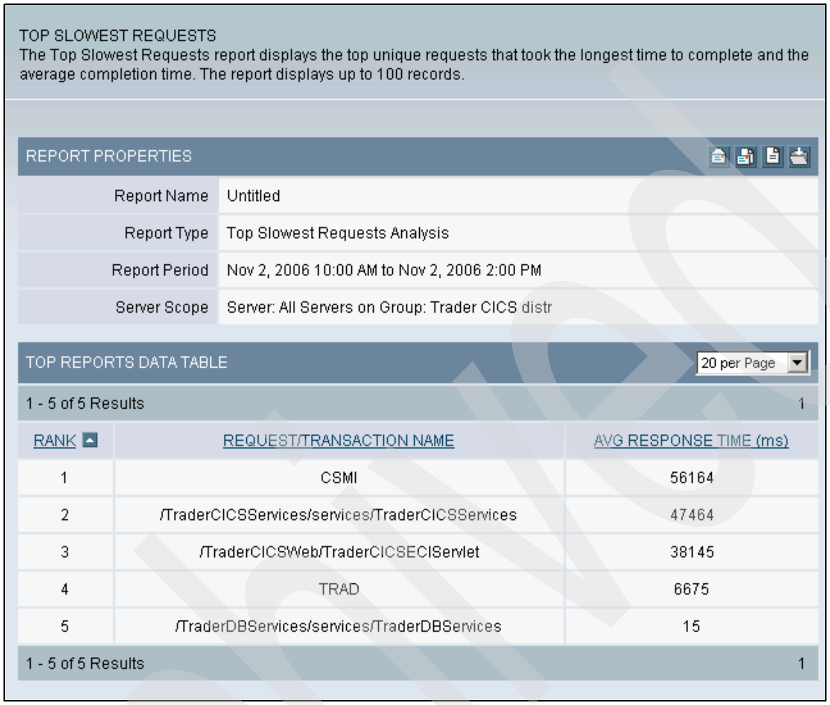


Figure 3-22 Top reports: top slowest requests

Figure 3-22 shows CSMI as the top request in terms of response time. We start investigating the reason for the high response time by drilling down deeper into the details of the report.

We select **PERFORMANCE ANALYSIS** → **Create Application Reports** → **Request/Transaction** and select the **TraderCICSdistr** group to get more details of the CSMI request. We are able to filter our search by a certain request. In our case, we look at all requests in the specified server group. Therefore, we leave this field blank. We also select the metric **Response Time(ms)** to see the request with the highest response times (Figure 3-23).

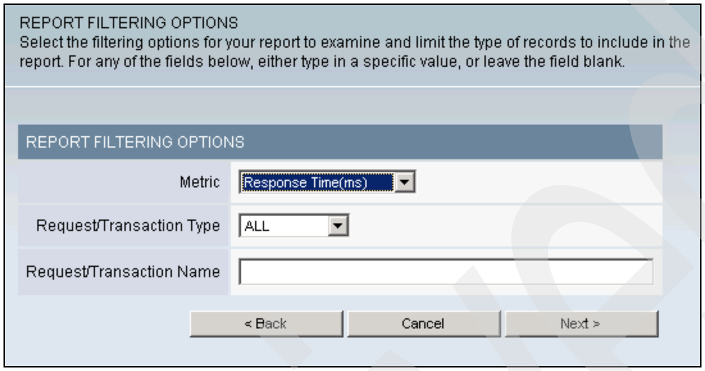


Figure 3-23 Report filtering options

Figure 3-24 shows the resulting report. The graphic in the report immediately informs you that the requests with the worst responsiveness appeared between 10:00 and 14:00 of that day. In this time frame, we had requests with responsiveness longer than 60 seconds. We click the bar showing the bad performer to investigate that specific time frame.

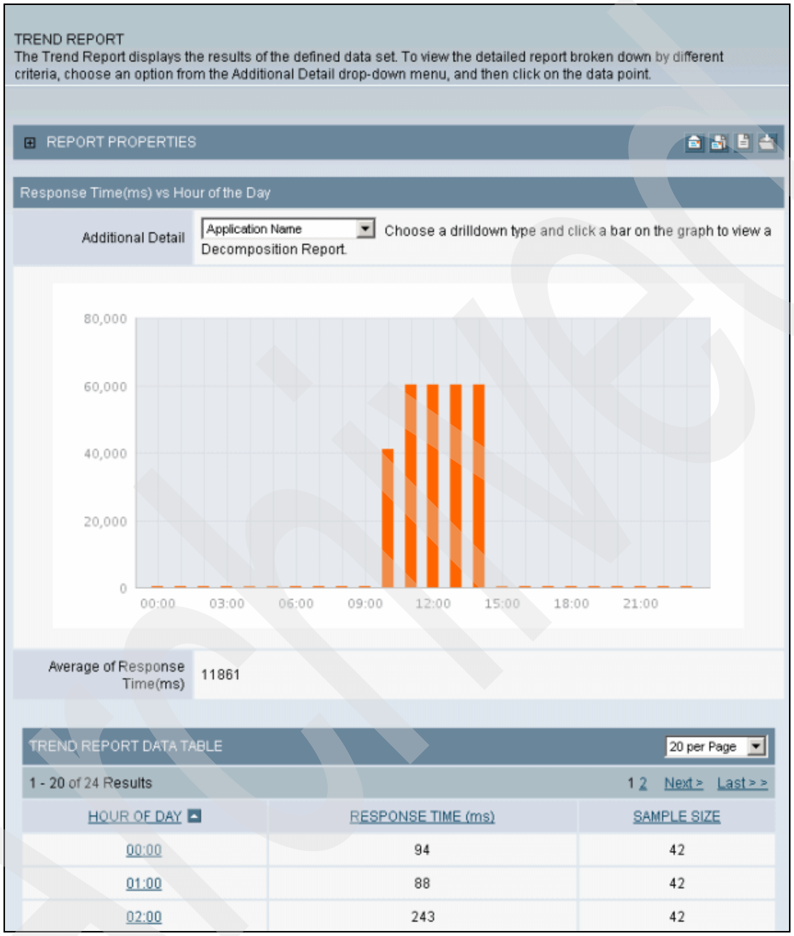


Figure 3-24 Response time of requests by hour

Selecting the request between 10:00 and 11:00, the report shows a pie chart with all the requests in this time frame on the selected server group. We can see the CSMI CICS application with responsiveness as long as 60462 ms. The TraderCICSServices application has even longer response times. We look at the CSMI application now. We investigate the TraderCICSServices application in 3.10, “Composite transaction analysis” on page 135.

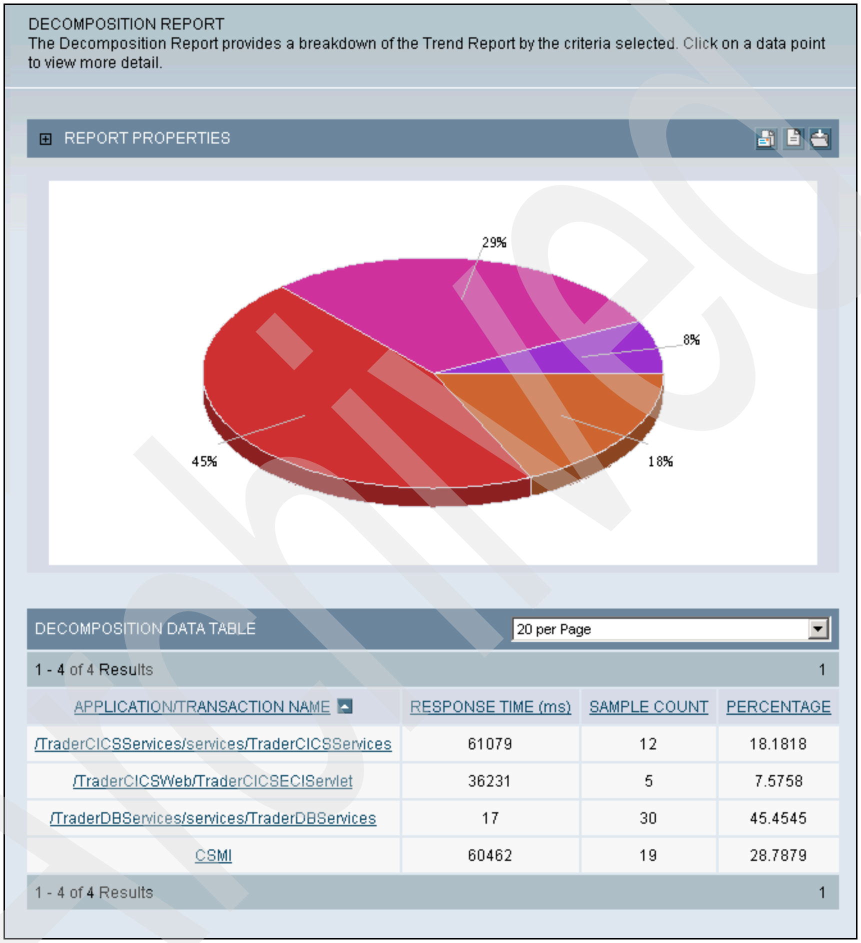


Figure 3-25 Decomposition report

Click an application for a detail report. We select the CSMI application. The report shows all the CSMI transactions in the selected hour. We can also see a symbol on the left of the requests that tells us that the request is a composite request (as discussed in 3.10, “Composite transaction analysis” on page 135). The report also provides the exact time and date that the request was executed and the response time of the request. We now have a closer look at one of those requests by selecting the name next to the composite request symbol.

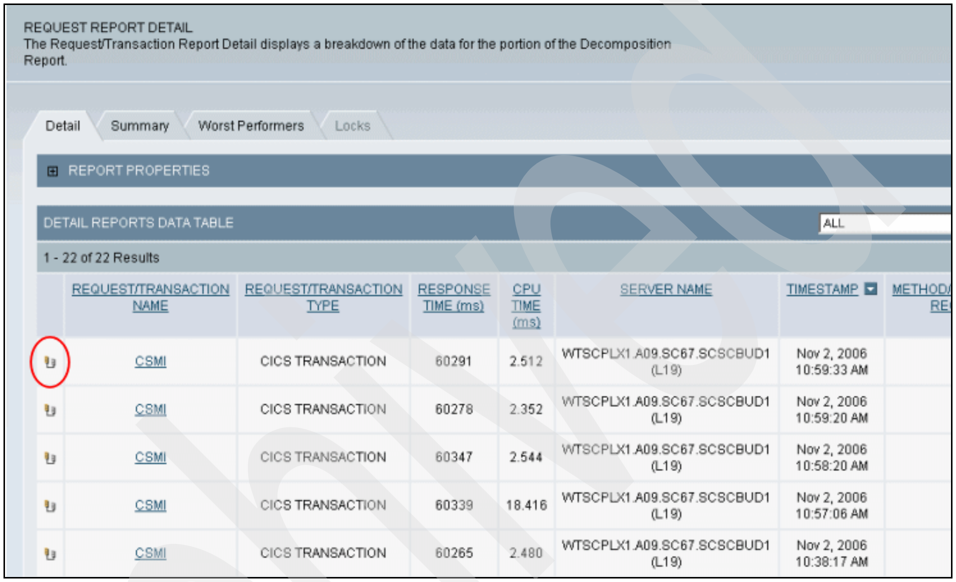


Figure 3-26   Sorted detail CSMI report

As shown in Figure 3-27, the report shows the flow view of the selected transaction. We can see the highlighted CICS EXEC command DELAY that has an elapsed time of 60 seconds. That is the reason for the overall high response time of the CICS transaction. In this case, we programed the CICS transaction to sleep for 60 seconds to show you the possibility of a deep dive analysis of a specified transaction.

> **Note:** For detailed information about method calls in the flow view, you must be at L3 monitoring level.

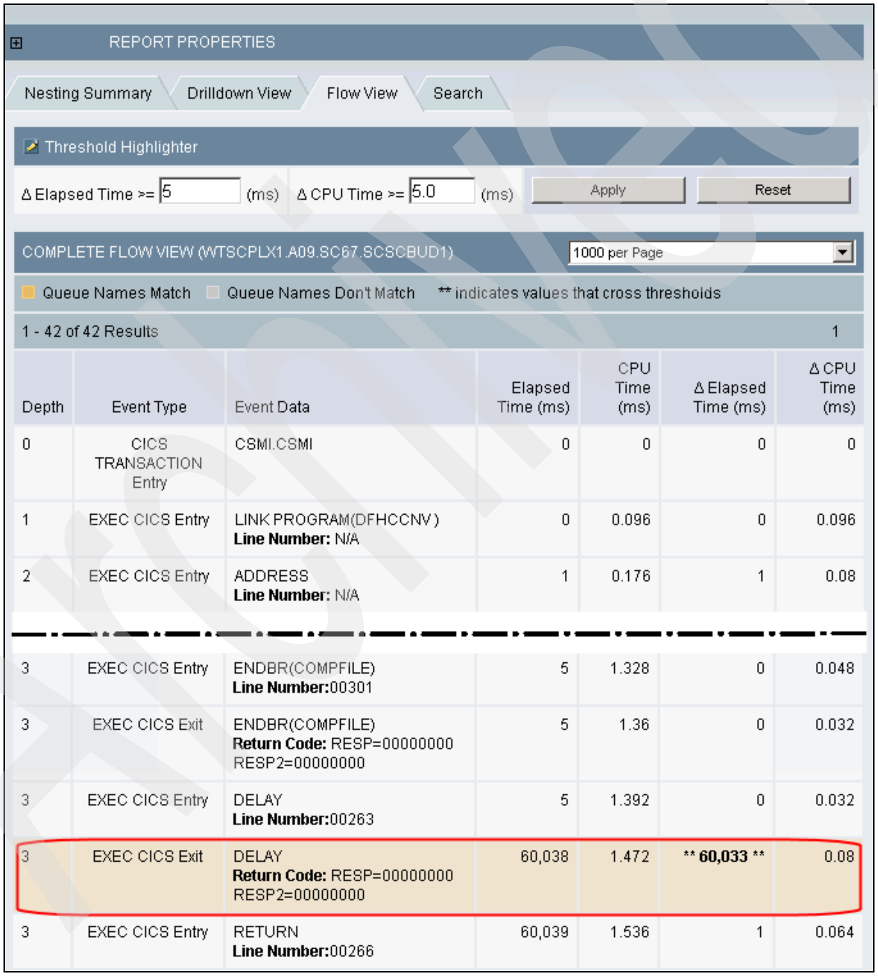


Figure 3-27 Flow view for the transaction in question

3.6.3 Method-based reporting

You can dig even deeper into transactions by displaying every method called by the transaction. To do this, we configure this option on the data collector side by editing the toolkit_custom.properties file in the <DC_HOME>\runtime\<Server>\ directory. Figure 3-28 shows a report with method details.

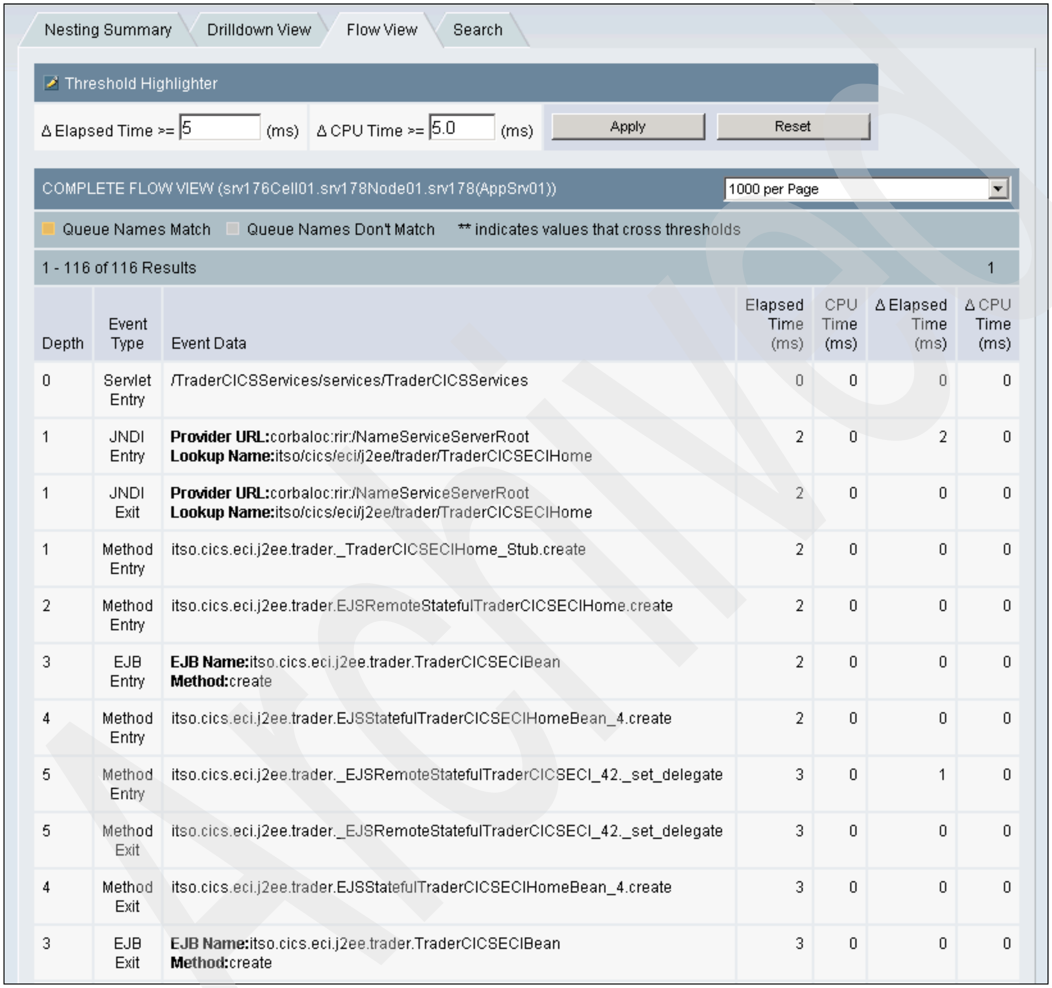


Figure 3-28 Method detail report

3.7 Memory diagnosis

ITCAM for WebSphere has several features that can aid you in finding out the different aspects of heap/memory usage by your application. You can view how your heap is performing under different load scenarios and how garbage collection is effecting your application. Using this information, you can decide whether you need to make any adjustments in heap size or garbage collection.

ITCAM for WebSphere also provides features related to heap dump management. Using these features, you can create heap dumps and analyze and compare these dumps. You can also use ITCAM for WebSphere to find any possible memory leaks in your application.

This section explains all the features mentioned above. We describe how to use ITCAM for WebSphere to find any memory leaks in your Java applications, how to analyze the performance of heap, and the effect of GCs on your application. We also show you how to use ITCAM for WebSphere to create heap dumps and compare these two heap dumps.

3.7.1 Memory analysis

This feature can be used to figure out performance-related problems due to garbage collection or JVM heap size. To create a memory analysis report:

1. Click **Problem Determination** → **Memory Diagnosis** → **Memory Analysis**.
2. From the Memory Analysis page, select the group and server, as shown in Figure 3-29.
3. Select the analysis type based on whether you want to analyze garbage collection or heap size, as shown in Figure 3-29.

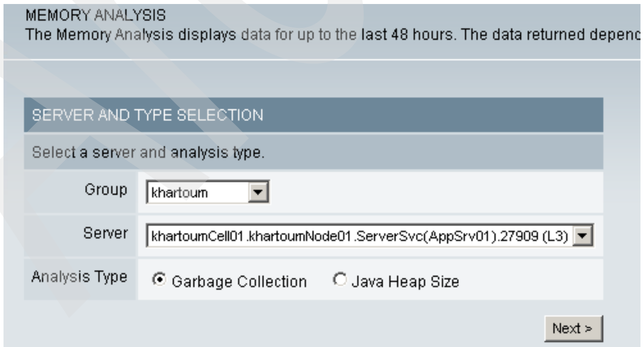


Figure 3-29 Memory analysis

4. Click **Next** and select the appropriate metric that you want to analyze, as shown in Figure 3-30.

> **Note:** The options in metric selection vary depending on which type of analysis type is chosen.

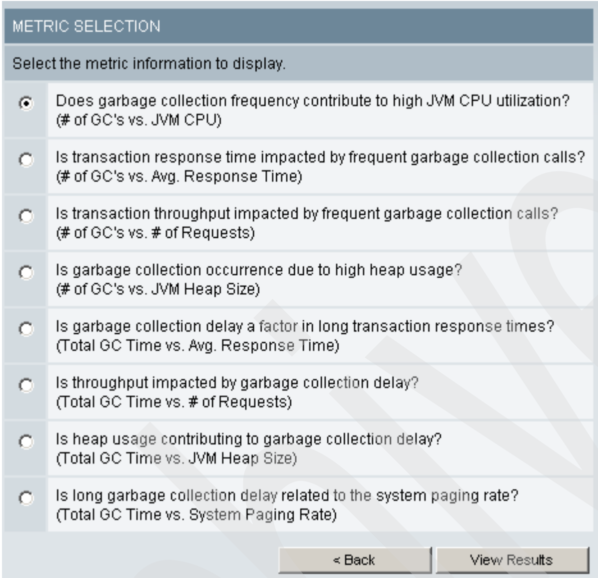


Figure 3-30 Metric selection

5. Click **View Results**. You will be able to see the memory analysis report on the basis of the metric selected, as shown in Figure 3-31.

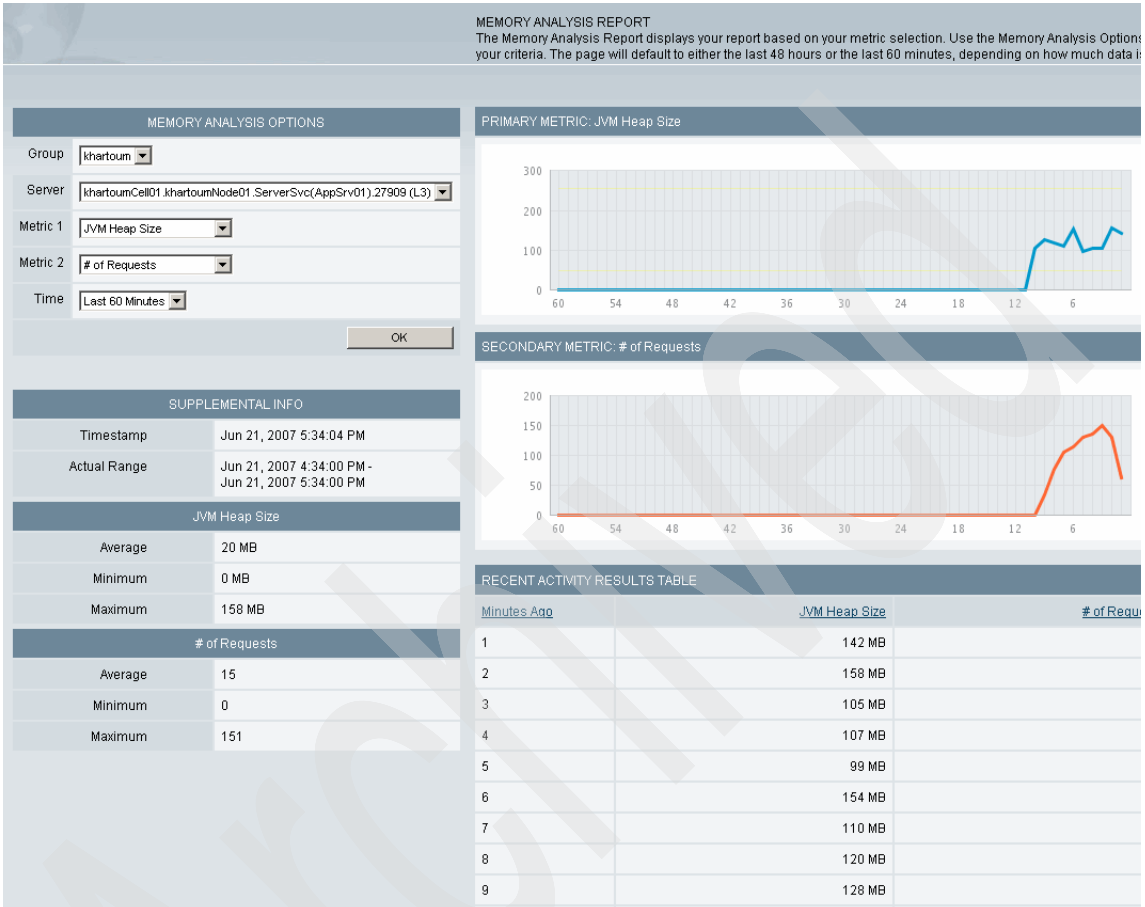


Figure 3-31 Memory analysis report

6. Using the memory analysis options in the memory analysis report, as shown in Figure 3-31, you can analyze different factors, like the effect of a higher number of sessions on JVM size, on number of times garbage collection is invoked, and so on. Similarly, you can also figure out whether your JVM heap size is continuously peaking with respect to the number of live sessions. This kind of analysis can be used to perform trend analysis and capacity planning.

3.7.2 Heap dump management

The heap dump management option in ITCAM for WebSphere lets you schedule a heap dump, analyze it, and delete it when you are done. The dumps created using the heap dump can be compared using the Memory Dump Diagnostics for Java (MDD4J) tool. This tool is available from:

http://www.ibm.com/developerworks/websphere/downloads/memory_dump.html

1. Click **Problem Determination** → **Memory Diagnosis** → **Heap Dump Management**.
2. The heap dump management window shows you all the dumps that have been scheduled in the last 48 hours, as shown in Figure 3-32.

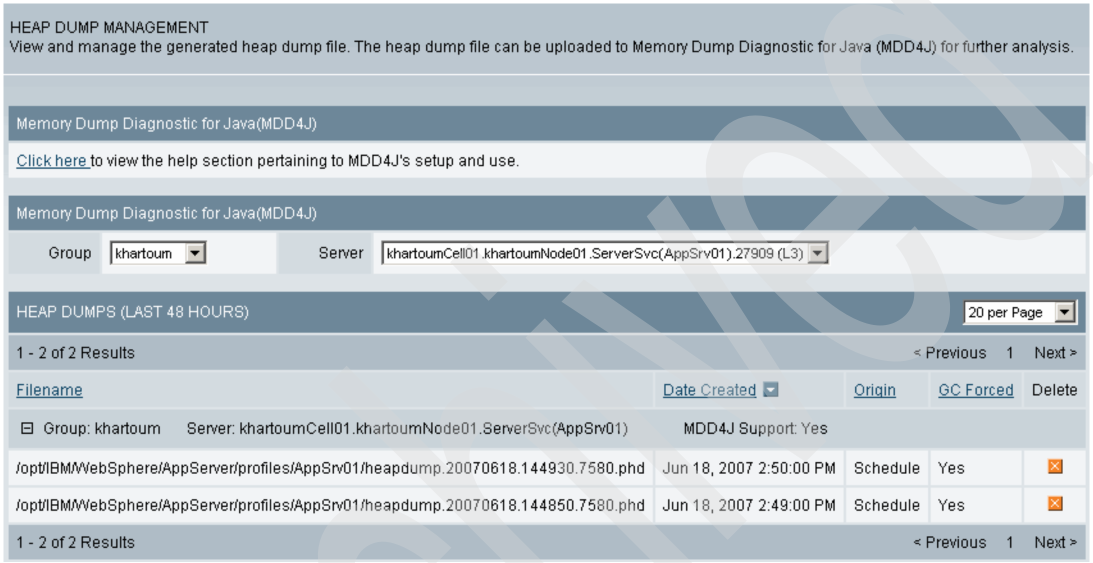


Figure 3-32 Heap dump management report

3. You can also schedule a new heap dump by selecting **Schedule a Heap Dump.** You can also select whether garbage collection needs to be run before generating a heap dump, as shown in Figure 3-33.

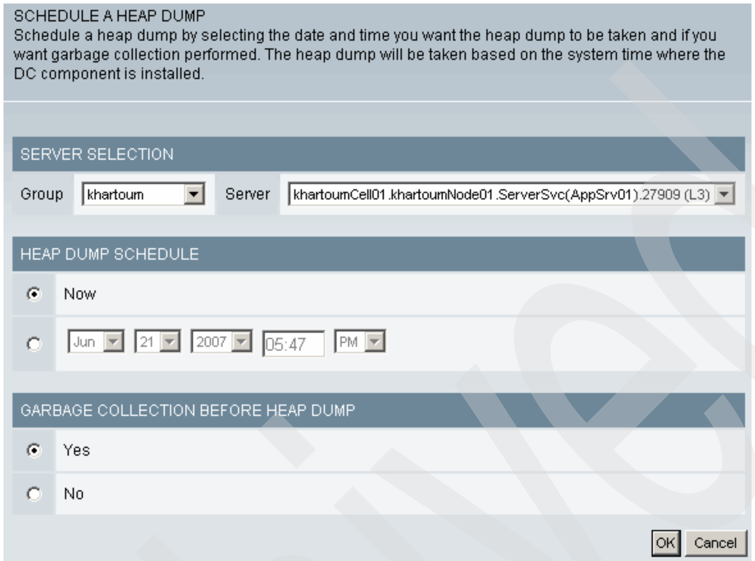


Figure 3-33 Scheduling a heap dump

3.7.3 Heap analysis

Using the heap analysis option, you can view how heap is being used by objects of different classes. When you run the heap analysis report, ITCAM for WebSphere takes a snapshot of the heap and provides the usage of heap by individual class name:

1. Click **Problem Determination** → **Memory Diagnosis** → **Heap Analysis**.
2. Select the appropriate group and server that you want to analyze.
3. Select whether you want to run garbage collection. Click **OK**.

4. The heap analysis results report is displayed, as shown in Figure 3-34.

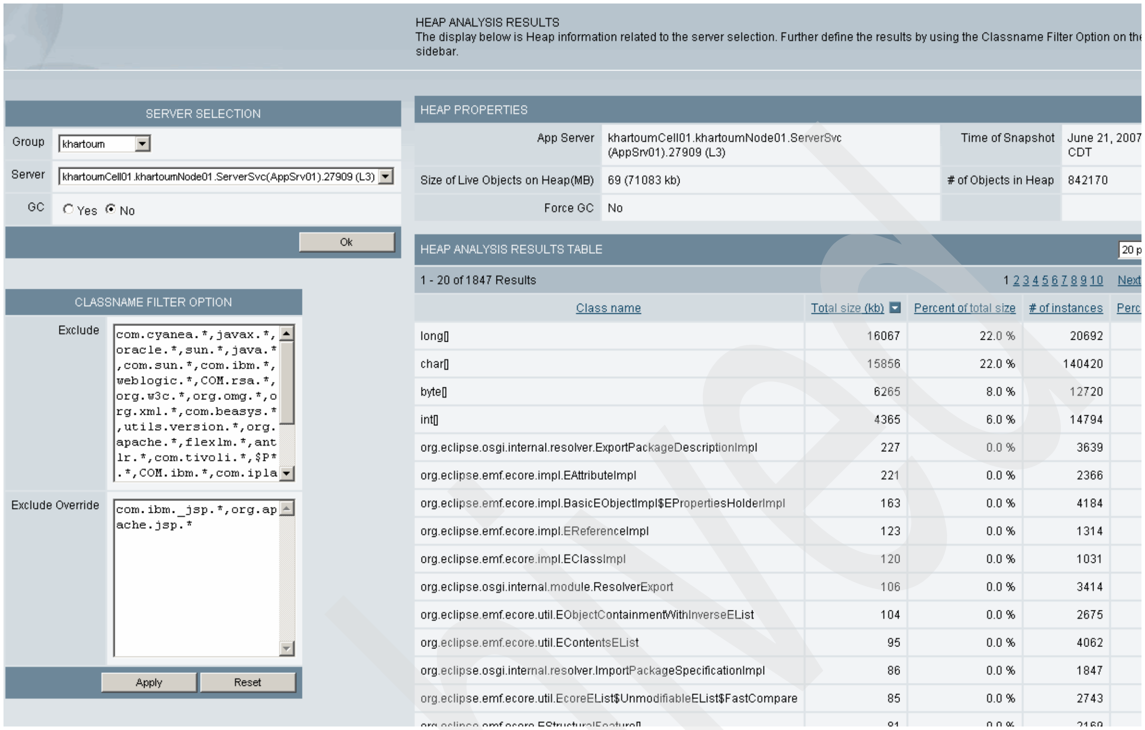


Figure 3-34 Heap analysis results

5. If you want to view only the usage by certain classes and ignore other classes, you can use the Exclude and Exclude Override option on the left side of the report, as shown in Figure 3-34.

> **Tip:** It might happen that some of your classes are hogging the memory and that the system is unable to release memory for these classes. This might lead to accumulation of these objects in JVM. These might be a source of memory leaks. Use the memory leak option, as explained in 3.7.4, “Memory leak analysis” on page 102, to learn about potential memory leaks.

3.7.4 Memory leak analysis

ITCAM for WebSphere provides different types of reports that can be analyzed to learn about the potential memory leaks in your application. These reports are:

1. Memory leak confirmation report
2. Memory leak candidate finder report
3. Memory leak diagnosis report

This section explains what these reports are and how you can use these features of ITCAM for WebSphere to locate possible memory leaks.

Memory leak confirmation report

This report can be used to learn about how your heap is performing under different load scenarios and to learn about any trends in memory leaks. You can compare JVM heap size to various metrics like live sessions and number of requests to learn about the behavior of heap under various load conditions. This helps you confirm any leaks, such as your heap size usage suddenly spiking up during high load. We expect that once the load is low, heap usage falls down to a certain level. You look at the memory leak confirmation report, but it shows that heap usage is still high even when load is very low. This shows that there might be memory leaks in the application, as your objects are not being freed up even after garbage collection. To run a memory leak confirmation report:

1. Click **Problem Determination** → **Memory Diagnosis** → **Memory Leak**.
2. Select the group and server from the Server Selection page.

3. The memory leak confirmation report opens, as shown in Figure 3-35. You can change different metrics and analyze the behavior of the heap.

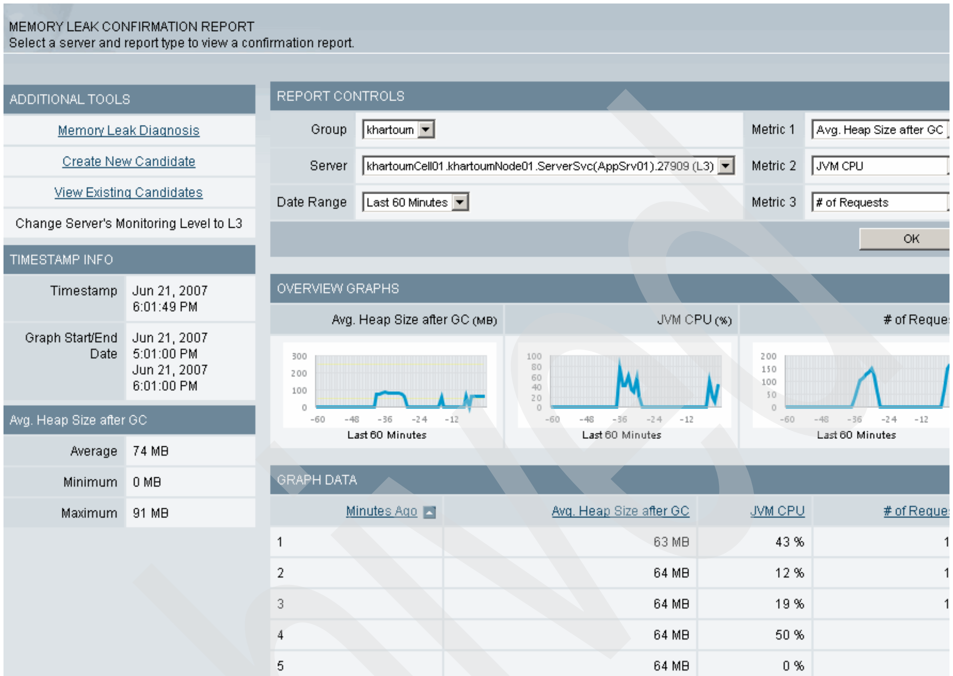


Figure 3-35 Memory leak confirmation report

Memory leak candidate finder report

Using this report, you can compare two heap snapshots. This helps you determine whether the number of instances of particular classes is increasing over time. If this is the case, then this class may be a candidate for a memory leak. To run a memory leak candidate finder report:

1. Click **Problem Determination** → **Memory Diagnosis** → **Memory Leak**.
2. Select the group and server from the Server Selection page.
3. The memory leak confirmation report opens, as shown in Figure 3-35.
4. In Additional Tools, click **Create New Candidate**.
5. The Create New Candidate page opens. Select the group and server that you want to analyze.

6. Enter the wait time. The wait time is the time the system waits before taking the second heap snapshot, as shown in Figure 3-36.

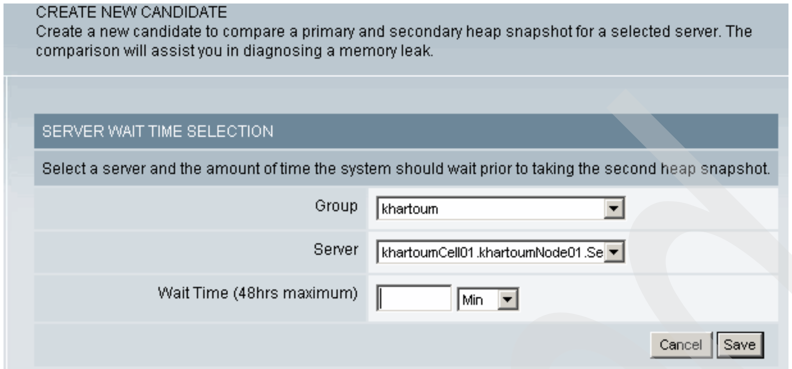


Figure 3-36 Create new candidate page

The above steps create a new candidate finder report. Once the wait time that you specified has elapsed, you can check the report:

1. Click **Problem Determination** → **Memory Diagnosis** → **Memory Leak**.
2. Select the group and server on the Server Selection page.
3. The memory leak confirmation report opens. Click **Additional Tools** → **View Existing candidates**.
4. The Memory Leak Candidate Finder Management page opens, showing all the candidates. The report you created earlier should be completed by now. If the wait time has not elapsed, it should be in the waiting state. Once it is complete, click the link under Server Name, as shown in Figure 3-37.

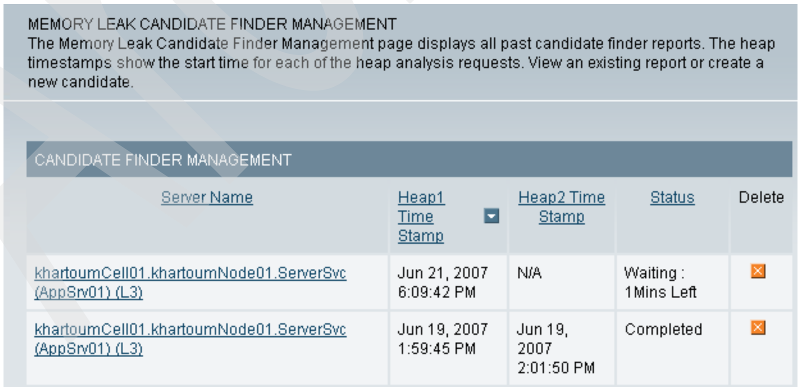


Figure 3-37 Memory leak candidate finder management page

5. The memory leak candidate finder report opens, as shown in Figure 3-38.

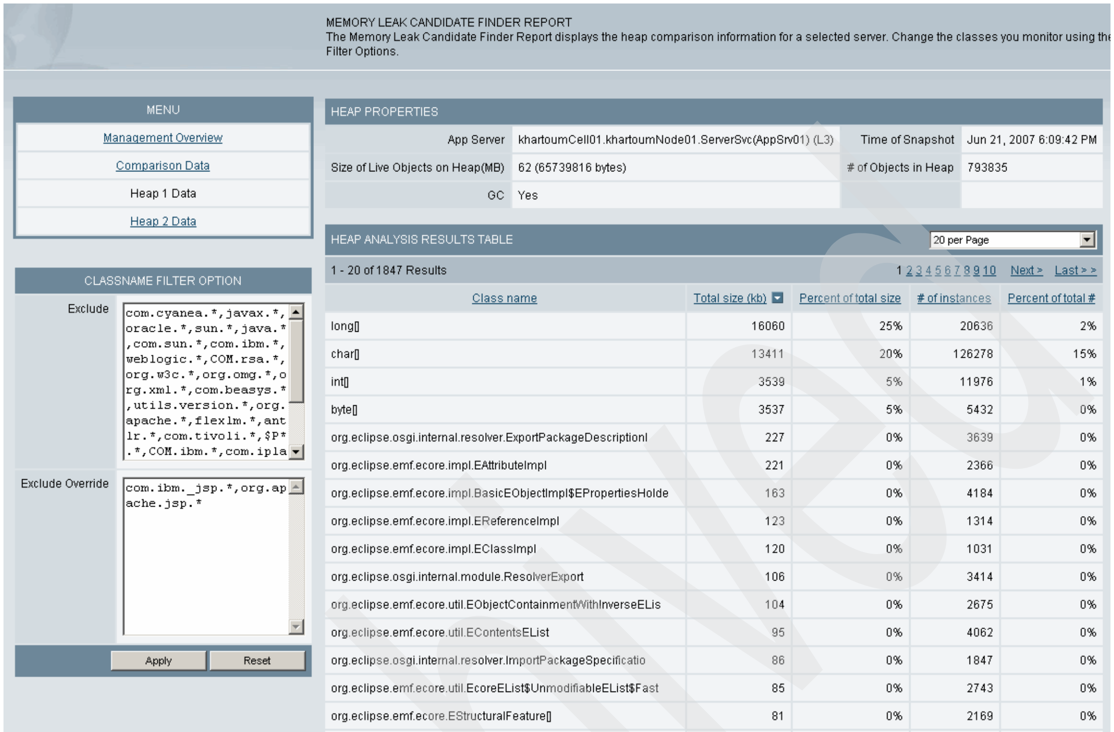
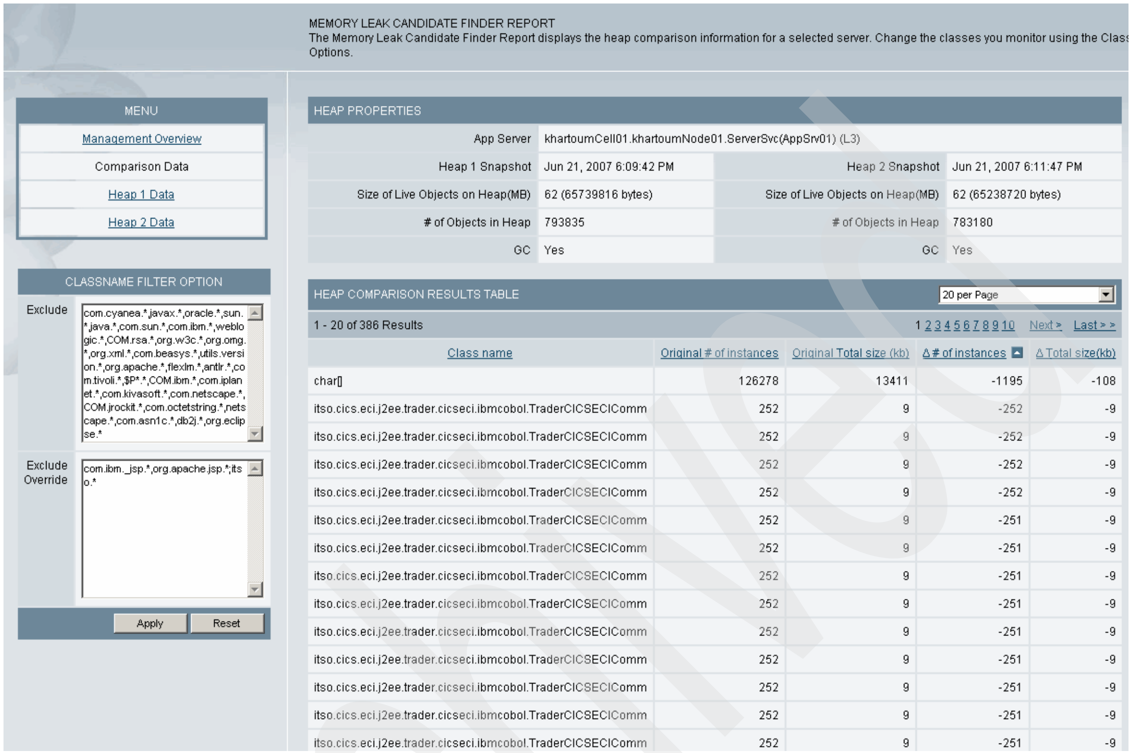


Figure 3-38 Memory leak candidate finder report

6. Click the **Comparison Data** link in the memory leak candidate finder report.

7. Comparison data shows the class name, the change in number of instances, and the change in total size, as shown in Figure 3-39.



Class name	Original # of instances	Original Total size (kb)	Δ # of instances	Δ Total size(kb)
char[]	126278	13411	-1195	-108
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-252	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-252	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-252	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-252	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9
itso.cics.eci.j2ee.trader.cicseci.ibmcobol.TraderCICSECIComm	252	9	-251	-9

Figure 3-39 Heap comparison data

8. This difference in number of instances/size might be an memory leak candidate. You can also view individual heap data using the heap 1 data or heap 2 data link.

Memory leak diagnosis report

The memory leak diagnosis report lets you drill down to the exact cause of the memory leak. This report gives you detailed information, such as for a given class, the growth rate of objects in memory, how many objects survived garbage collection, the average age of objects in heap, which particular method of a class has created this given object, and so on.

> **Note:** To run the memory leak diagnosis report, your server must be configured to run at the L3 monitoring level.

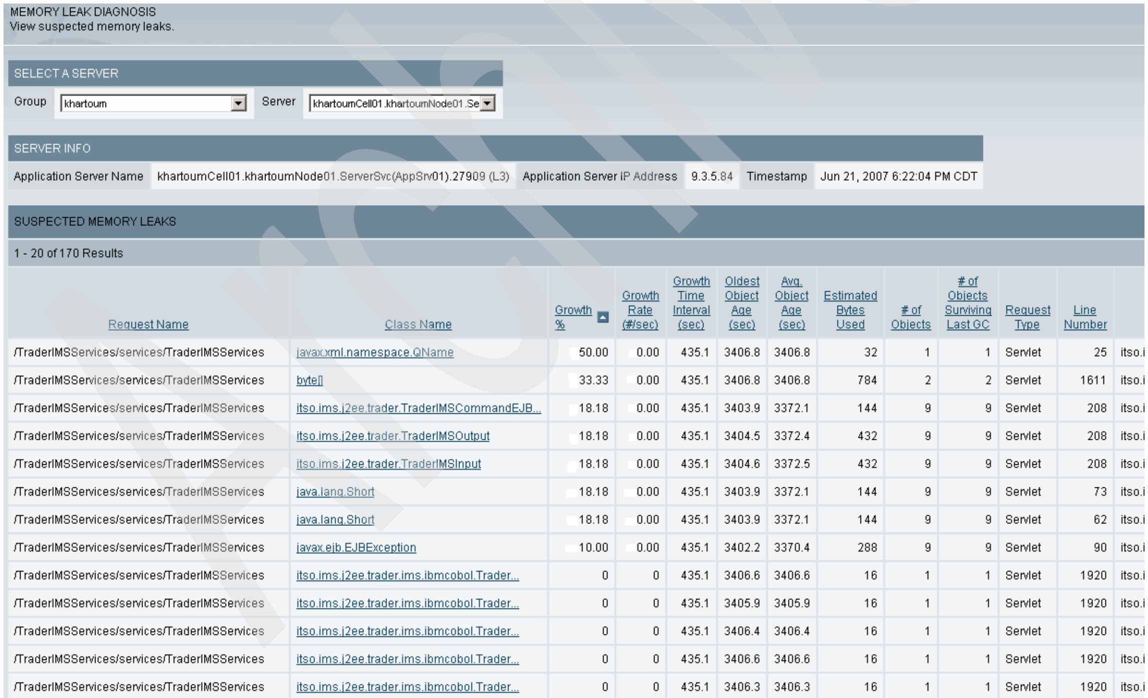
Also, make sure that you have enabled the memory leak diagnosis in you DC configuration, by making the changes shown in Figure 3-2 in $DC_HOME/runtime/<was -node>.<was-server>/custom/toolkit_custom.properties.

Example 3-2 Sample contents of toolkit_custom.propeties

```
am.camtoolkit.gpe.customxml.leak=/opt/IBM/itcam/WebSphere/DC/itcamdc/et
c/memory_leak_diagnosis.xml
com.ibm.tivoli.itcam.toolkit.ai.enablememoryleakdiagnosis=true
```

To run a memory leak diagnosis report:

1. Click **Problem Determination** → **Memory Diagnosis** → **Memory Leak**.
2. Select the group and server on the Server Selection page.
3. From the memory leak confirmation report, click **Additional Tools** → **Memory Leak Diagnosis**.
4. The memory leak diagnosis report opens, displaying detailed information, as shown in Figure 3-40.



Request Name	Class Name	Growth %	Growth Rate (#/sec)	Growth Time Interval (sec)	Oldest Object Age (sec)	Avg. Object Age (sec)	Estimated Bytes Used	# of Objects	# of Objects Surviving Last GC	Request Type	Line Number	
/TraderIMSServices/services/TraderIMSServices	javax.xml.namespace.QName	50.00	0.00	435.1	3406.8	3406.8	32	1	1	Servlet	25	itso.i
/TraderIMSServices/services/TraderIMSServices	byte[]	33.33	0.00	435.1	3406.8	3406.8	784	2	2	Servlet	1611	itso.i
/TraderIMSServices/services/TraderIMSServices	itso.ims.j2ee.trader.TraderIMSCommandEJB...	18.18	0.00	435.1	3403.9	3372.1	144	9	9	Servlet	208	itso.i
/TraderIMSServices/services/TraderIMSServices	itso.ims.j2ee.trader.TraderIMSOutput	18.18	0.00	435.1	3404.5	3372.4	432	9	9	Servlet	208	itso.i
/TraderIMSServices/services/TraderIMSServices	itso.ims.j2ee.trader.TraderIMSInput	18.18	0.00	435.1	3404.6	3372.5	432	9	9	Servlet	208	itso.i
/TraderIMSServices/services/TraderIMSServices	java.lang.Short	18.18	0.00	435.1	3403.9	3372.1	144	9	9	Servlet	73	itso.i
/TraderIMSServices/services/TraderIMSServices	java.lang.Short	18.18	0.00	435.1	3403.9	3372.1	144	9	9	Servlet	62	itso.i
/TraderIMSServices/services/TraderIMSServices	javax.ejb.EJBException	10.00	0.00	435.1	3402.2	3370.4	288	9	9	Servlet	90	itso.i
/TraderIMSServices/services/TraderIMSServices	itso.ims.j2ee.trader.ims.ibmcobol.Trader...	0	0	435.1	3406.6	3406.6	16	1	1	Servlet	1920	itso.i
/TraderIMSServices/services/TraderIMSServices	itso.ims.j2ee.trader.ims.ibmcobol.Trader...	0	0	435.1	3405.9	3405.9	16	1	1	Servlet	1920	itso.i
/TraderIMSServices/services/TraderIMSServices	itso.ims.j2ee.trader.ims.ibmcobol.Trader...	0	0	435.1	3406.4	3406.4	16	1	1	Servlet	1920	itso.i
/TraderIMSServices/services/TraderIMSServices	itso.ims.j2ee.trader.ims.ibmcobol.Trader...	0	0	435.1	3406.6	3406.6	16	1	1	Servlet	1920	itso.i
/TraderIMSServices/services/TraderIMSServices	itso.ims.j2ee.trader.ims.ibmcobol.Trader...	0	0	435.1	3406.3	3406.3	16	1	1	Servlet	1920	itso.i

Figure 3-40 Memory leak diagnosis

5. You can also click an individual class name in the memory leak diagnosis report to drill down to the reference to the live objects report.
6. The reference to the live objects report shows the references that have been made to the live objects in the heap by this class, as shown in Figure 3-41.

Figure 3-41 Reference to live objects report

3.8 Lock analysis

The lock analysis feature provides the user with the ability to track synchronization calls, lock acquired and released. Lock data is considered nested request data and appears at monitoring level 2. A user can view both in-flight and historic lock information.

Enabling lock analysis increases both application startup time and memory requirements. Activating the lock analysis feature modestly increases the time necessary to start ITCAM for WebSphere. It also increases the memory requirements of both the WebSphere application server and the ITCAM for WebSphere data collector. This increased consumption of system resources is caused by the lock analysis feature's requirement to implement bytecode instrumentation for application's Java classes.

This section discusses the following topics:

- 3.8.1, “Configuring data collector for lock analysis” on page 109
- 3.8.2, “Lock contention on server activity display” on page 111
- 3.8.3, “Lock analysis events” on page 113
- 3.8.4, “Creating lock analysis report” on page 113

3.8.1 Configuring data collector for lock analysis

This section reviews the procedures necessary to enable lock analysis and to customize lock analysis settings.

1. Make a copy of the file <DC_HOME>/itcamdc/etc/lock_analysis.xml and open it in a text editor.
2. Modify the lockingClasses parameter in the lock_analysis.xml file.

   The parameter defines the classes for which lock requests will be Byte-Code-Instrumented. By default, all lock requests in all application classes are selected, <lockingClass>*</lockingClass>. By modifying this tag, you can implement a more granular selection, although within a class all lock requests are Byte-Code-Instrumented. Multiple lockingClasses tags can be specified.

   The lockingClasses tag can include wildcard characters. The following is a summary of how the wildcard characters work:

   – An asterisk (*) stands for zero or more occurrences of any character when used by itself. When embedded within a sequence of characters (for example, java.*.String), it matches zero or more occurrences of any character except the package separator (.).

   – Two periods (..) can be used to specify all sub-packages (for example, java..String matches java.lang.String). It matches any sequence of characters that starts and ends with the package separator (.).

   – If the locking class name begins with an exclamation point (!), any classes matching the classes identified in the tag are specifically excluded from BCI for lock analysis. This is useful for indicating that all classes are to be Byte-Code-Instrumented except for those classes that are specifically excluded.

   For example, an application with a package name of com.ibm.myapp has the following requirements:

   – Only classes that begin with Cus or Sup should be Byte-Code-Instrumented for lock analysis.

   – The supplier class should not be Byte-Code-Instrumented for lock analysis.

The contents of the Example 3-3 customized lock_analysis.xml file illustrate the options described in 3.8.1, “Configuring data collector for lock analysis” on page 109.

Example 3-3 Sample lock_analysis.xml file ITCAM V6.1

```
<aspect>
     <type>application</type>
<name>com.ibm.tivoli.itcam.toolkit.ai.aspectj.apptrace.CaptureLock</nam
e>
     <enabledProperty>
com.ibm.tivoli.itcam.toolkit.ai.enablelockanalysis</enabledProperty>
     <defaultEnabled>true</defaultEnabled>
     <lockingClass>com.ibm.myapp.Cus*</lockingClass>
     <lockingClass>com.ibm.myapp.Sup*</lockingClass>
     <lockingClass>!com.ibm.myapp.Supplier</lockingClass>
  </aspect>
```

3. Save the modified lock_analysis.xmlfile in the <DC_HOME>/runtime/app_server_version.node_name.server_name/custom directory.
4. In /runtime/app_server_version.node_name.server_name/custom/ modify the toolkit_custom.properties file to complete the lock analysis configuration. Set the property am.camtoolkit.gpe.customxml.lock to /opt/IBM/itcam/WebSphere/DC/itcamdc/etc/lock_analysis.xml and change the property com.ibm.tivoli.itcam.toolkit.ai.enablelockanalysis to true.

> **Note:** You need to restart the server that you have instrumented for lock analysis.

3.8.2 Lock contention on server activity display

A user can view in-flight lock contention information from the Server Activity Display (SAD) page using a tab called Lock Contentions. The ITCAM for WebSphere application monitor gets in-flight lock contention information directly from the data collector. To view in-flight lock contention information:

1. From the top navigation, click **Problem Determination** → **Server Activity Display** (Figure 3-42).

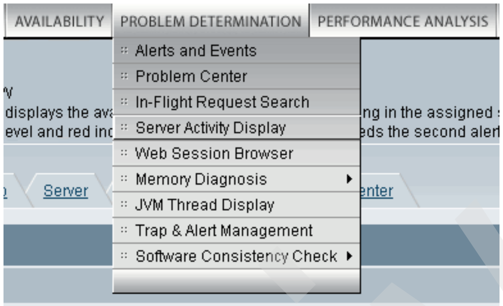


Figure 3-42 Getting to the Server Activity Display page

This brings you to the Active Requests tab for the server activity display (Figure 3-43). The Server Activity Display window provides thread data for an application server at a specific point in time.

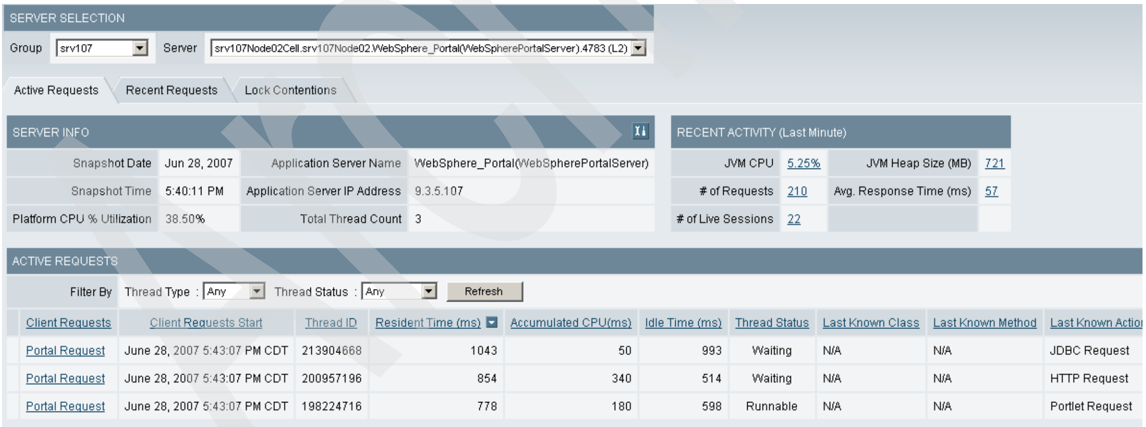


Figure 3-43 Lock analysis - Active Request tab

2. Clicking the **Lock Contentions** tab displays a table with the current active locks. Figure 3-44 shows three active locks. By default, the lock contention result table is sorted by locked object class column.


Active Requests | Recent Requests | Lock Contentions

ACTIVE LOCKS

Locked Object Class	Owner Request/Transaction Name	Owner Request/Transaction Type	Owner Class	Owner Method	Waiting Time (ms)	
java.lang.Class	/AIMLockTestappWeb/LockTestServlet?componentLA=Servlet&opera	Servlet	com.aim.testapp.lockejbapp.CoarseLock	getNextSerialNum	728	com.aim.te
java.lang.Class	/AIMLockTestappWeb/LockTestServlet?componentLA=Servlet&opera	Servlet	com.aim.testapp.lockejbapp.CoarseLock	getNextSerialNum	727	com.aim.te
java.lang.Class	/AIMLockTestappWeb/LockTestServlet?componentLA=Servlet&opera	Servlet	com.aim.testapp.lockejbapp.CoarseLock	getNextSerialNum	727	com.aim.te

Figure 3-44 Lock analysis - Lock Contentions

Table 3-1 lists the information for each column on the lock contention result data table.

Table 3-1 Columns for the lock contention table

Column name	Description
Locked Object Class	The class name of the locked object
Owner Request/Transaction	The name of the request that is locked
Name Owner Request/Transaction Type	The type of owner request/transaction (for example, servlet, EJB, JSP™, and so on)
Owner Class	The class that is holding the lock
Owner Method	The method that is holding the lock
Waiting Time (ms)	The amount of time in milliseconds spent by the waiting request
Waiting Class	The class that is acquiring the lock
Waiting Method	The method that is acquiring the lock
Waiting Request/Transaction Name	The name of the waiting request/transaction
Waiting Workload Type	The type of waiting request/transaction (for example, servlet, EJB, JSP, and so on)

3.8.3 Lock analysis events

To support the lock analysis feature, level 2 event types are passed from a data collector to the publish server. Those events are:

- Lock acquisition start: The application is trying to acquire a lock.
- Lock acquisition end: The application has acquired a lock.
- Lock release: The start application is starting to release a lock.
- Lock release: The end application has released a lock.
- Lock contention start: The application is trying to acquire a lock, but has to wait because the lock is owned by another thread. Normally, this event follows a lock acquisition start event.
- Lock contention end: The thread that owned the lock has released it. Normally, a lock acquisition end event follows this event.

In-flight lock analysis events and requests can be viewed from Server Activity Display pages. Historic lock analysis events and requests can be viewed from Performance Analysis Report pages. See 3.8.4, “Creating lock analysis report” on page 113.

3.8.4 Creating lock analysis report

The lock analysis report allows you to examine lock history data for your in-flight transactions. As with all performance analysis and reporting data, all lock analysis data are historical. Lock data are not available for CICS and IMS transactions. In addition, the include and exclude filters do not affect the collection or reporting of lock data.

To define a lock analysis report:

1. From the top navigation, click **Performance Analysis** → **Create Reports** → **Lock Analysis** (Figure 3-45).

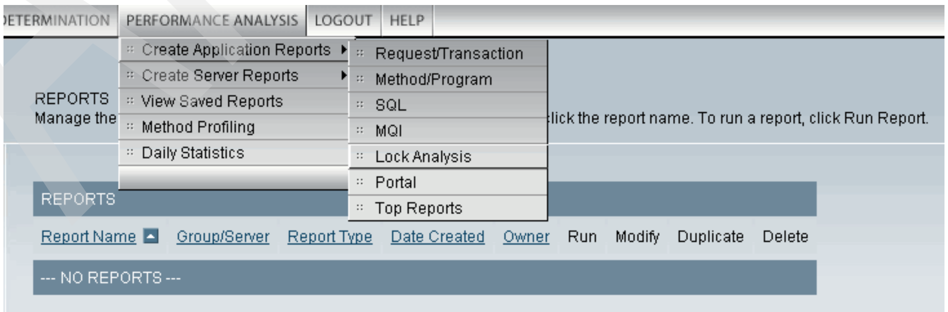


Figure 3-45 Creating a lock analysis report

2. The recurrence page opens (Figure 3-46). As we just want to view the report once, we select **No** and click **Next**.

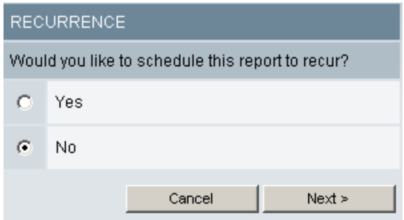


Figure 3-46 Scheduling report for recurrence

3. Select the group and the server on which you want to report from the drop-down menus, and click **Next**. See Figure 3-47.

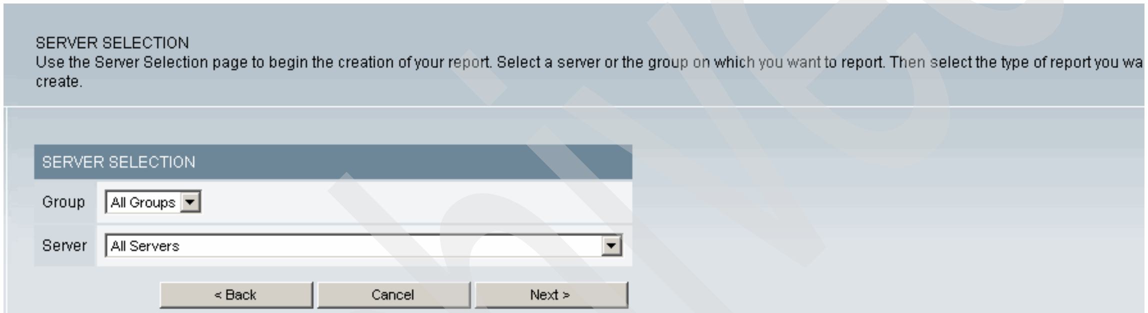


Figure 3-47 Group and server filtering options

4. The Report Filtering Options page opens (Figure 3-48). For the metric option, select one of the following from the drop-down list:

   – Number of Lock Acquisitions: the total number of locks acquired, per request

   – Number of Lock Contentions: the total number of locks that a request had to wait for

   – Total Acquisition Time: the total time a request held a lock

   Set the request/transaction type: EJB, JSP, servlet, portal, or all. Optionally, set the request/transaction name. Optionally, define the method/program. Click **Next**.

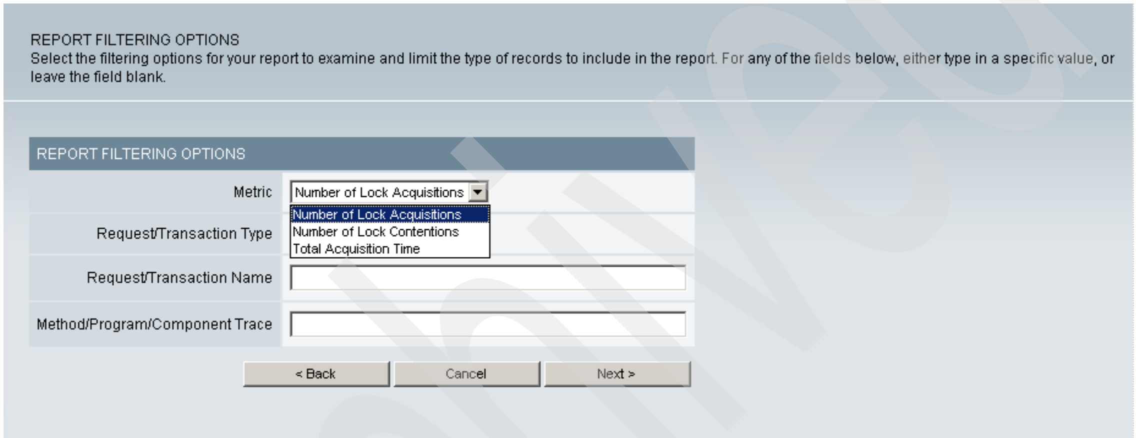


Figure 3-48 Lock report filtering options

5. The Date Range Settings page opens. Set the start date, end date, start time, and end time. If applicable, set the advanced filtering to extract the data of a specific time period. Click **Next**.

6. Set the graphing option for your report's X-axis (Figure 3-49):
   - Time Series in Month
   - Time Series in Week
   - Time Series in Day
   - Time Series in Hour
   - Aggregate Minute of the Hour
   - Aggregate Hour of the Day
   - Aggregate Day of the Week
   - Aggregate Month of the Year

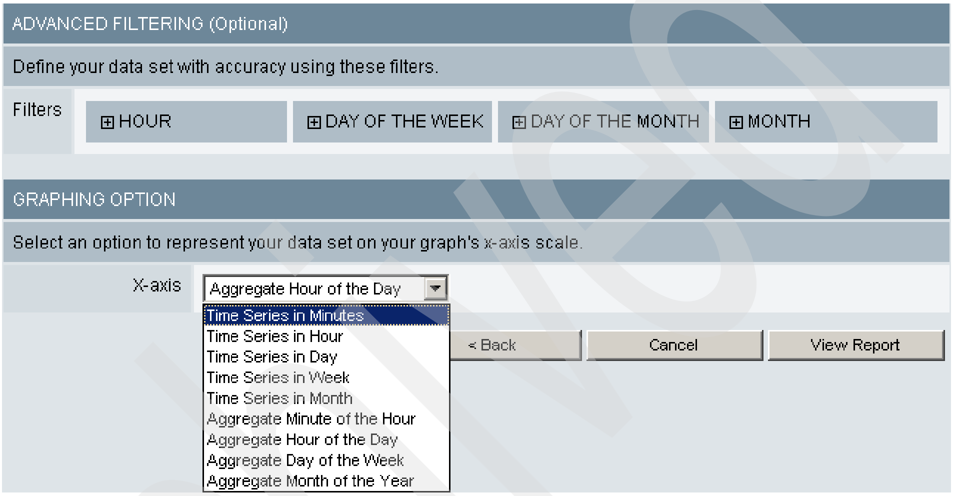


Figure 3-49 Graphing options for X-x-axis

7. Click **View Report**. The Lock Trend Report opens.

   From the Lock Trend Report, (Figure 3-50) you can request more detailed information about a particular group of locks, decomposed by either the type of application running (JSP, EJB, servlet, or portlet), the application name, or the server on which the locks occurred.

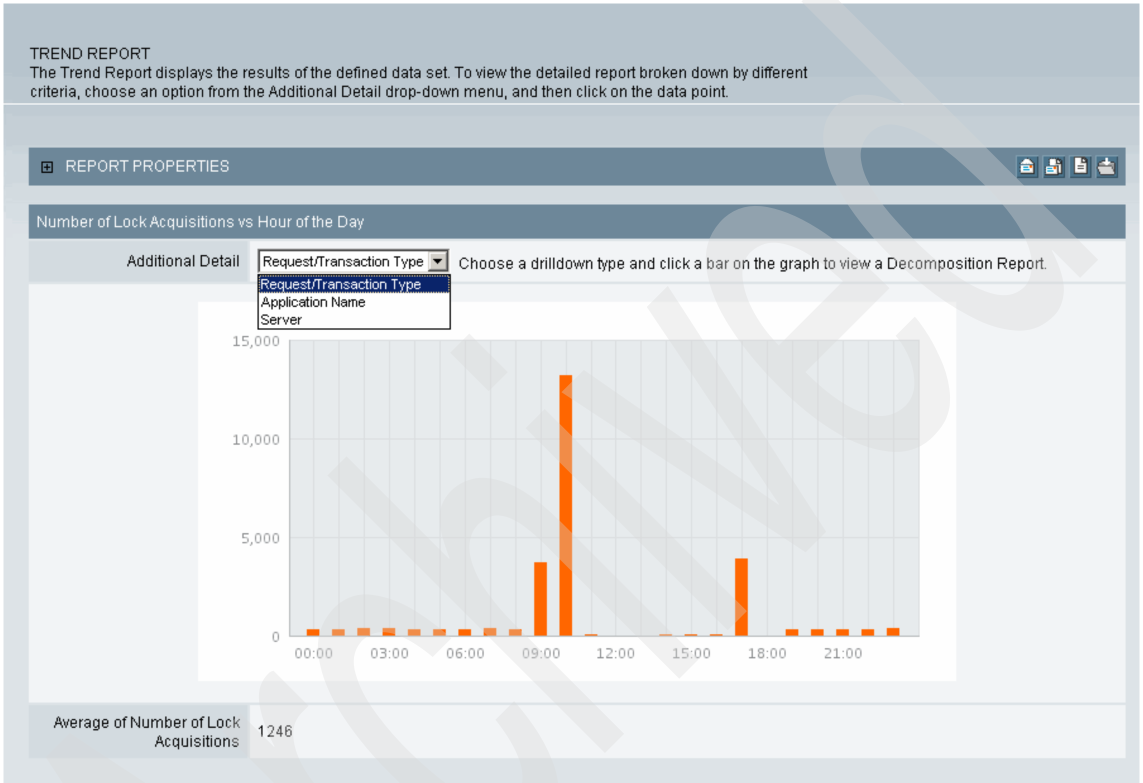


Figure 3-50 Lock trend report

8. You may choose a drilldown type, request/transaction type, application name, or server and click a bar on the graph to view a Decomposition Report From the Additional Detail drop-down. Select one of the following options:

   – Request/Transaction Type
   – Application Name
   – Server

9. Select either a bar from the number of lock acquisitions versus hour of day bar graph (Figure 3-50 on page 117) or a time of day from the trend report data table. See Figure 3-51.

TREND REPORT DATA TABLE 20 per Page

1 - 20 of 24 Results 1 2 Next > Last > >

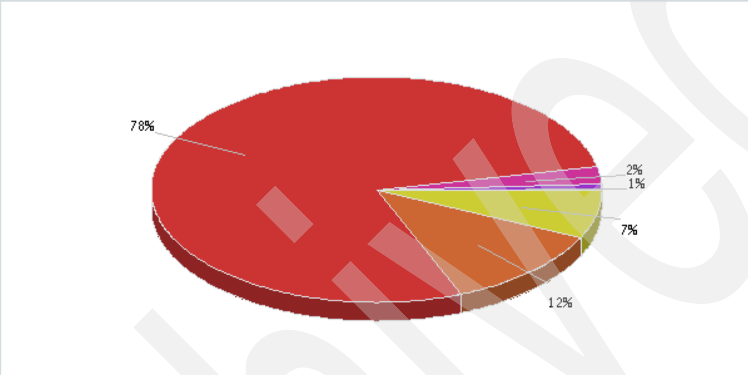
HOUR OF DAY	NUMBER OF LOCK ACQUISITIONS	SAMPLE SIZE
00:00	373	298
01:00	373	298
02:00	386	309
03:00	385	308
04:00	348	278
05:00	356	285
06:00	351	281
07:00	403	322

Figure 3-51 Trend report data table

10. The lock decomposition report then displays, showing the locks recorded at the selected time of day broken down by either application type or application name, as you selected. Application name was chosen as the option for the next illustration of a lock analysis decomposition report. See Figure 3-52.

DECOMPOSITION REPORT
The Decomposition Report provides a breakdown of the Trend Report by the criteria selected. Click on a data point to view more detail.

⊞ REPORT PROPERTIES



DECOMPOSITION DATA TABLE — 20 per Page

1 - 5 of 5 Results — 1

APPLICATION/TRANSACTION NAME	TOTAL LOCK ACQUISITION TIME (ms)	SAMPLE COUNT	PROJECTED COUNT	PERCENTAGE
Getting Started	13	729	911	6.8943
Login	4	259	323	2.4494
Trader	135	8220	10275	77.7378
Welcome	21	1274	1592	12.0484
com.ibm.workplace.wcm.util.scheduler.ejb.EJBSchedulerBean	2	92	115	0.8701

1 - 5 of 5 Results — 1

Figure 3-52 Application name lock decomposition report

11. The request/transaction type was chosen for the illustration of a lock analysis decomposition report. See Figure 3-53.

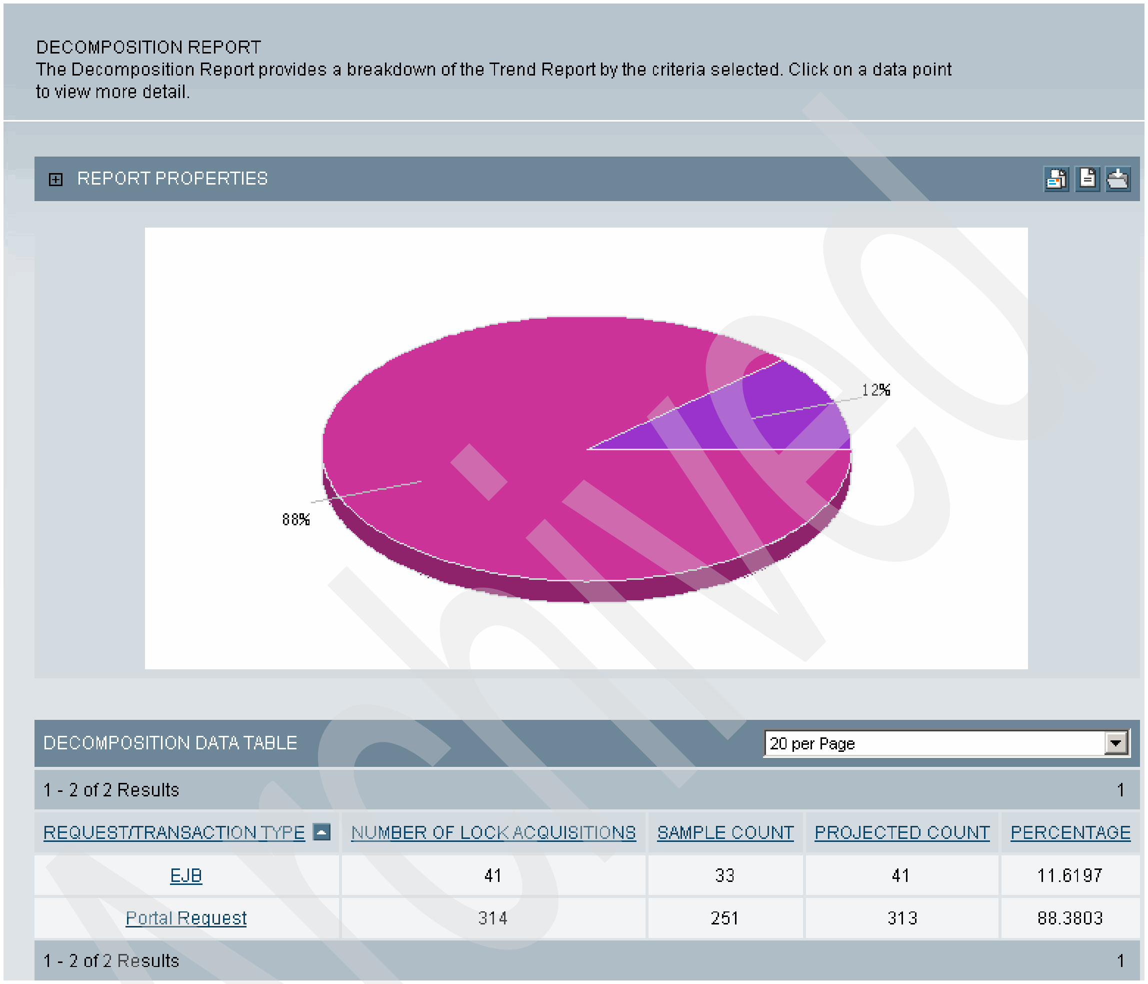


Figure 3-53 Request/transaction type lock decomposition report

12. From the lock decomposition report, you can request more detailed lock information about all applications of a particular type (JSP, EJB, servlet, or portlet) or all transactions for a particular application or server (depending on the decomposition option you chose when generating the lock decomposition report).

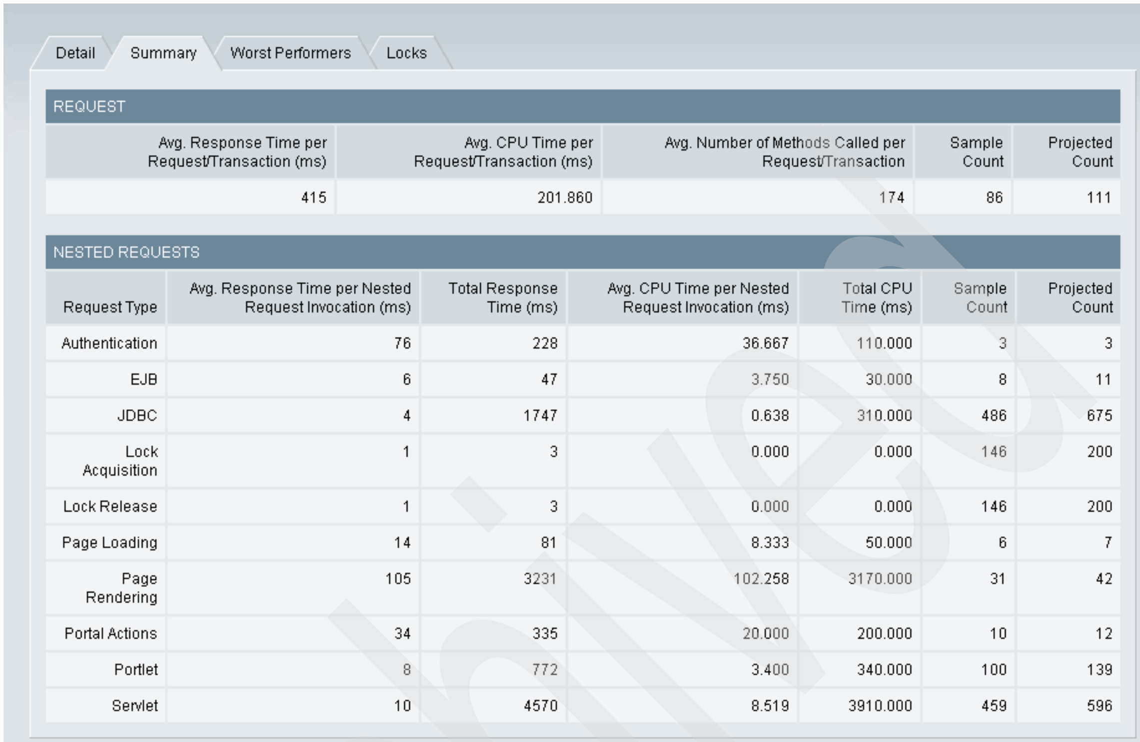
 Within the Report Properties pane, select a pie segment (for example, see Figure 3-52 on page 119), or within the decomposition data table select either a request/transaction type, an application/transaction name, or a server name.

13. The Lock Detail Report's Detail page (Figure 3-54) shows the locks recorded for all transactions of the type selected or all transactions for the application or server selected.

    The request/transaction report detail displays a breakdown of the data for the portion of the decomposition.

Figure 3-54 Lock detail report

14. To view summary information for all locks, select the **Summary** tab.



REQUEST

Avg. Response Time per Request/Transaction (ms)	Avg. CPU Time per Request/Transaction (ms)	Avg. Number of Methods Called per Request/Transaction	Sample Count	Projected Count
415	201.860	174	86	111

NESTED REQUESTS

Request Type	Avg. Response Time per Nested Request Invocation (ms)	Total Response Time (ms)	Avg. CPU Time per Nested Request Invocation (ms)	Total CPU Time (ms)	Sample Count	Projected Count
Authentication	76	228	36.667	110.000	3	3
EJB	6	47	3.750	30.000	8	11
JDBC	4	1747	0.638	310.000	486	675
Lock Acquisition	1	3	0.000	0.000	146	200
Lock Release	1	3	0.000	0.000	146	200
Page Loading	14	81	8.333	50.000	6	7
Page Rendering	105	3231	102.258	3170.000	31	42
Portal Actions	34	335	20.000	200.000	10	12
Portlet	8	772	3.400	340.000	100	139
Servlet	10	4570	8.519	3910.000	459	596

Figure 3-55 Lock detail report - Summary tab

15. To sort the list so that the applications with the most locks appear at the top, select the **Worst Performers** tab (Figure 3-56).

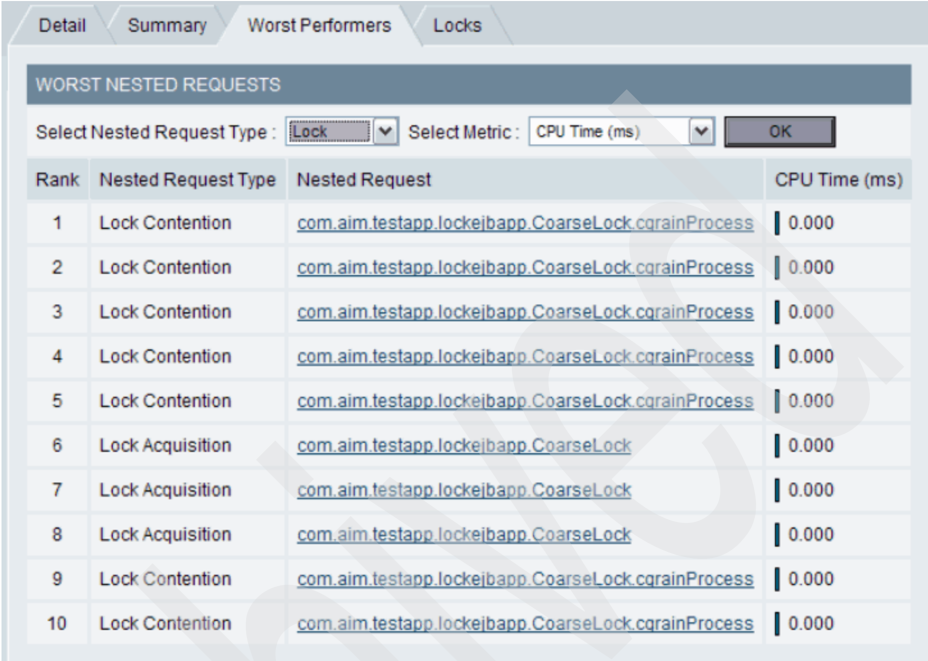
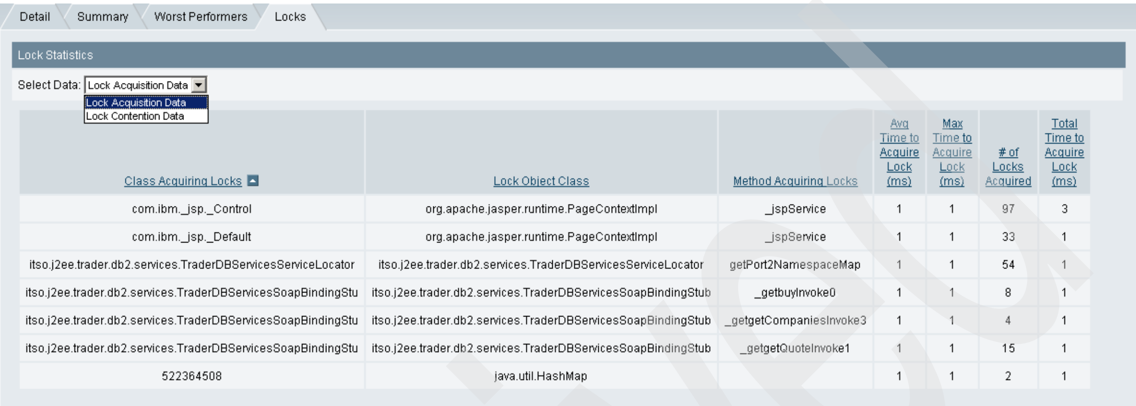


Figure 3-56 Lock detail report - Worst Performers tab

16. To summarize lock-acquisition versus lock-contention information, select the **Lock** tab (Figure 3-57). A lock acquisition is logged whenever an application attempts to lock an object and the object is free, whereas a lock contention is logged whenever an application attempts to lock an object and the object is already owned.



Class Acquiring Locks	Lock Object Class	Method Acquiring Locks	Avg Time to Acquire Lock (ms)	Max Time to Acquire Lock (ms)	# of Locks Acquired	Total Time to Acquire Lock (ms)
com.ibm._jsp._Control	org.apache.jasper.runtime.PageContextImpl	_jspService	1	1	97	3
com.ibm._jsp._Default	org.apache.jasper.runtime.PageContextImpl	_jspService	1	1	33	1
itso.j2ee.trader.db2.services.TraderDBServicesServiceLocator	itso.j2ee.trader.db2.services.TraderDBServicesServiceLocator	getPort2NamespaceMap	1	1	54	1
itso.j2ee.trader.db2.services.TraderDBServicesSoapBindingStu	itso.j2ee.trader.db2.services.TraderDBServicesSoapBindingStub	_getbuyInvoke0	1	1	8	1
itso.j2ee.trader.db2.services.TraderDBServicesSoapBindingStu	itso.j2ee.trader.db2.services.TraderDBServicesSoapBindingStub	_getgetCompaniesInvoke3	1	1	4	1
itso.j2ee.trader.db2.services.TraderDBServicesSoapBindingStu	itso.j2ee.trader.db2.services.TraderDBServicesSoapBindingStub	_getgetQuoteInvoke1	1	1	15	1
522364508	java.util.HashMap		1	1	2	1

Figure 3-57 Lock detail report - Lock tab

Two filter metrics are available in the lock analysis report through the Detail Report Lock tab option:

– Lock Acquisition Data
– Lock Contention Data

Note: The Lock tab is clickable in the lock report type only.

3.9 JBoss Application Server data collector

The ITCAM for J2EE data collector supports several J2EE application servers to perform diagnosis of Web applications and services. The functionality and management of a J2EE data collector is similar to the WebSphere Application Server data collector in ITCAM for WebSphere.

This section discusses usage scenarios for the JBoss Application Server data collector within our environment and explores the new functions of ITCAM for J2EE. We manage the software traps set on your system on the Trap and Alert Management page. Trap and Alert Management influences the presentation on the enterprise overview and operator assignment, affecting the display. Traps and alerts show how transactions behave, and the method trace is helpful for further analysis.

The steps documented in this section take place after the installation and configuration of the JBoss Application Server. We assume that you are familiar with the functionality of the JBoss Application Server and that you have finished installation processing for those components successfully.

> **Note:** The runtime environment in our sample configuration uses JBoss Application Server Version 4.0.4 GA on Red Hat Enterprise Linux AS Version 3 (Taroon Update 7). We use the default configuration, including a default profile of the application server.

We configured and customized the JBoss Application Server and the ITCAM for J2EE data collector for JBoss based on instructions in 2.4.3, “J2EE data collector” on page 45.

This scenario shows how to create, modify, and activate a trap for server resources and applications in order to get a component method trace for further analysis. We describe managing the software traps in trap and alert management in ITCAM for J2EE on the basis of a sample Web application.

Ensure that the J2EE data collector for JBoss is running and appears on the Enterprise Overview page in ITCAM for J2EE, as shown in Figure 3-58.

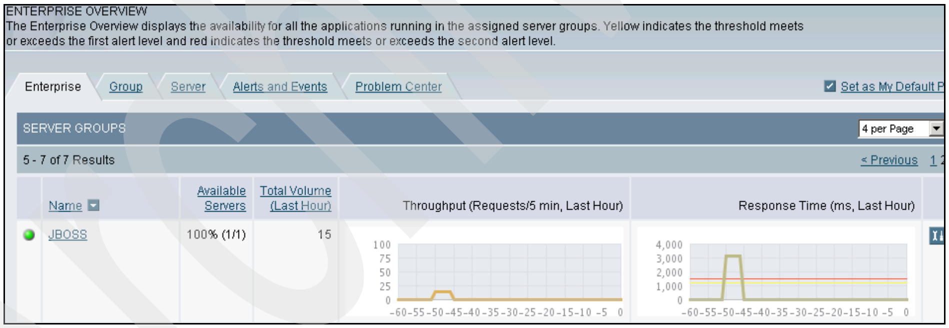


Figure 3-58 Enterprise overview showing active JBoss

The ITCAM for J2EE settings in the JBoss Web Console (typically at http://<machine>:8080/web-console) reside under **System** → **JMX MBeans** → **itcam listener**. The listener MBean name is ServerListenerService.

For the simulation of the scenario:

- One Web application runs well with an average CPU time of less than 1 ms. The situation changes, and randomly a high CPU time consumption attracts attention in the reports. To analyze the reason, ITCAM for J2EE provides the trap and alert management to force a snapshot for this Web application at run time.

- In our environment, we deployed a long-running sample application so that we could make a snapshot of active threads. Figure 3-59 demonstrates the view of this long-running thread in the ITCAM for J2EE console in **Server Activity Display** → **Recent Requests**.

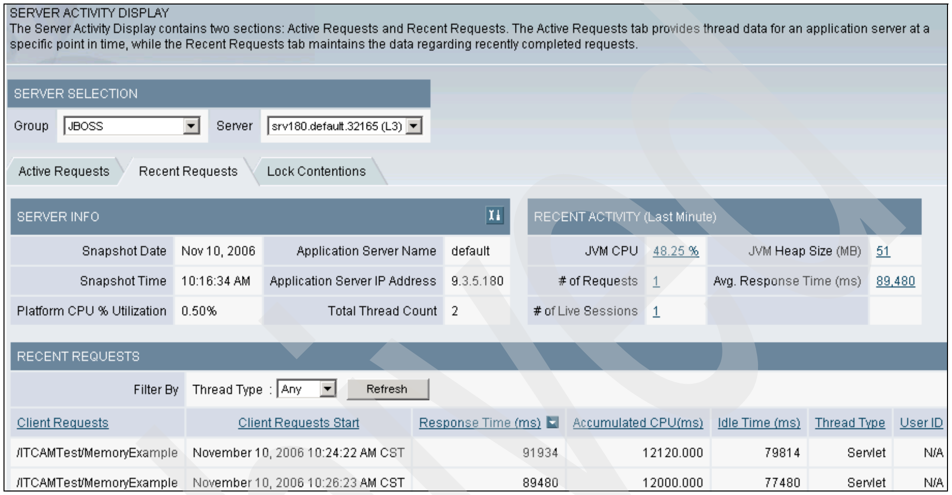


Figure 3-59 Request detail of recent requests of sample application

To create a trap event for the appropriate Web application:

1. From the main menu in ITCAM for J2EE, select the function for trap and alert management, using **PROBLEM DETERMINATION** → **Trap and Alert Management**. The Trap and Alert Management page opens, from which you manage the software traps set on your system. The overview page shows the current active traps and trap profiles, which can be activated, modified, duplicated, or deleted.

2. To create a new trap profile, select **Create Trap**. Figure 3-60 shows the first step page for this setting. On this page, select the trap type and the target type of resource. In our case, referring to Figure 3-60, we select the following values:

   – Trap Type: **Application Trap**
   – Target Type: **CPU Time**

   Click **Next** for further definition steps.

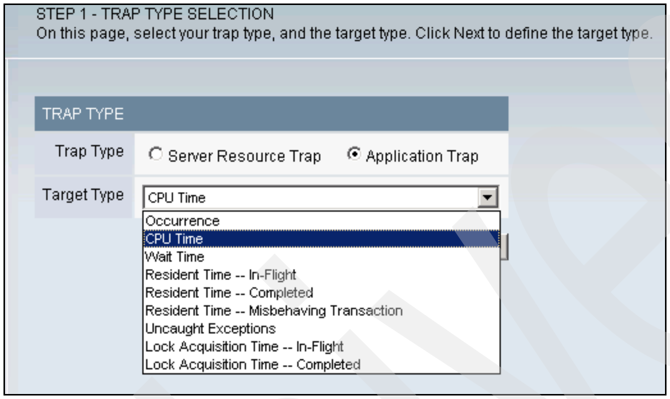


Figure 3-60 Create application trap: step 1

3. To define the trap, set the threshold value and provide the content. The content is divided by request or by method. The setting matches the request and triggers the trap. The threshold is a level that causes an alert when the system count exceeds the defined threshold. As shown in Figure 3-61, for the values of the sample Web application, we select the following values:
   - Threshold(ms): **2**
   - By Request: **ITCAMTest**

   Click **Next** to set trap alerts.

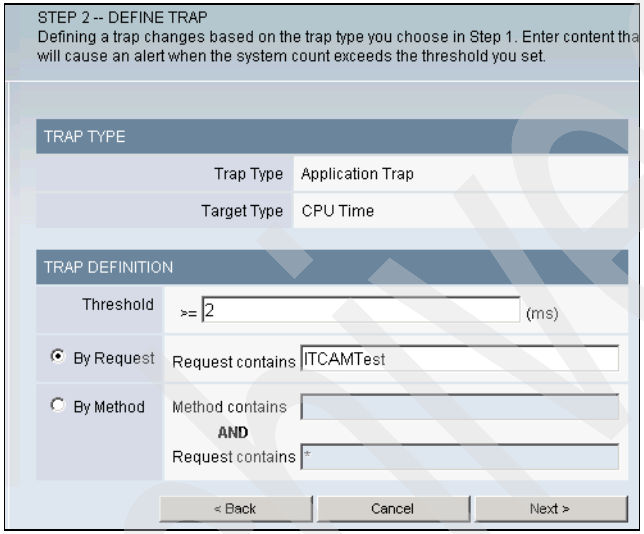


Figure 3-61 Create application trap: step 2

4. In step 3, we set the trap alerts settings. Several conditions can be added to the condition list. Set the number of times the trap occurs and the severity for this condition. Define alert actions for notifications by e-mail and SNMP messages. Further analysis of the request is possible by activating the component method trace if the trap is triggered. Optionally, the trap can be suppressed after an occurrence by setting a waiting period. In Figure 3-62, we add one condition with the following settings:

   – Number of time(s): **1**
   – Severity: **High**
   – Optional suppress: **5** (min)

Figure 3-62   Create application trap: step 3

Determine the condition settings and click **Add** to enlarge the condition list with a minimum of one condition. Optionally, select a value for the default suppression setting. Click **Next** to continue.

5. Finally, enter a trap profile name and description. Click **Save & Activate** to finish the configuration process and activate the trap immediately. Otherwise, click **Save** to store the trap profile in the trap profile list. Figure 3-63 shows the last configuration step. In our case, we activate the trap immediately with the Save & Activate option. Therefore, the activation process and configuration page opens.

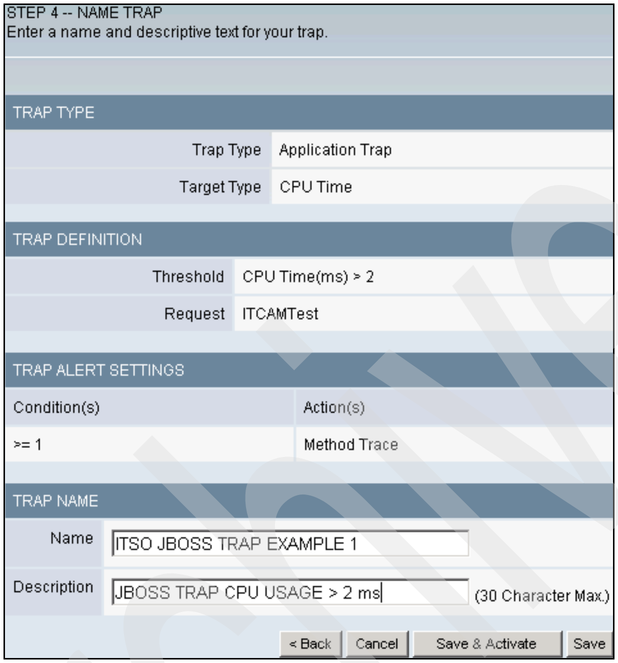


Figure 3-63 Create application trap: step 4

6. To activate the trap profile, we need to complete a few settings, such as server selection, alert suppression, and deactivation. Select an entire group of servers or select only a specific server where the trap needs to be activated. Select the time period of alert suppression. Specify the deactivation of the trap by a time period or number of occurrences. We set up following settings, as shown in Figure 3-64:

   – Server filter: **JBOSS**
   – Server: **srv180.default(L3)**
   – Alert suppression: **15min** (trap default)
   – Deactivation: **5** (number of occurrences)

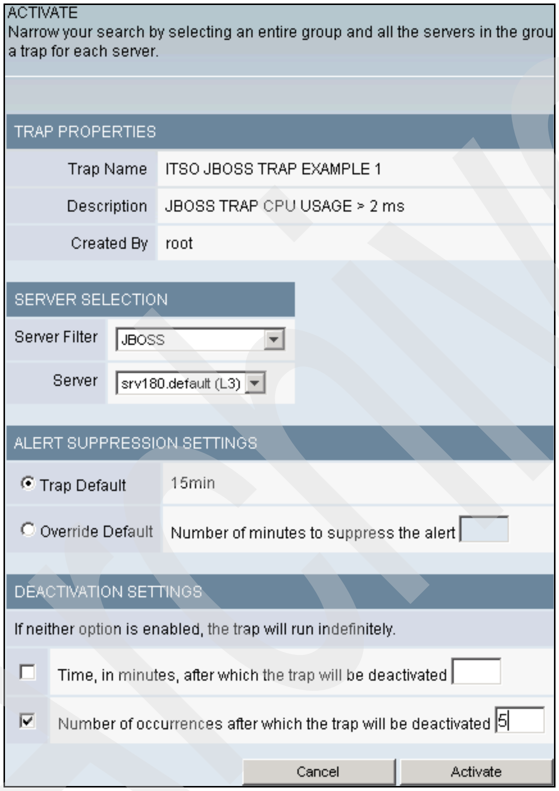


Figure 3-64 Activate application trap

After finishing the trap activation process, you see the new trap profile named ITSO JBOSS TRAP EXAMPLE1 in the active traps list on the Trap and Alert Management page, as illustrated in Figure 3-65.

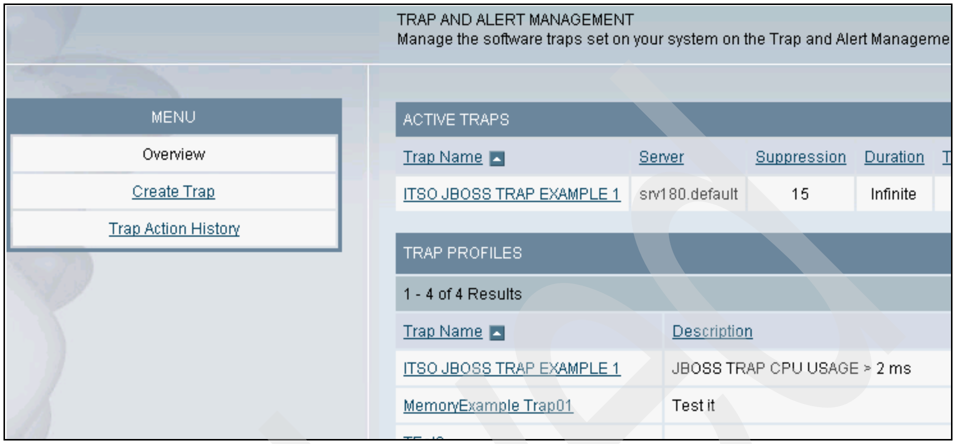


Figure 3-65 Trap and alert management: active traps list

Now trigger the active trap by requesting the designated Web application. During the request, the trap automatically executes a component method trace and stores the trace in the database. The Alerts and Events Management page lists the processed trap with its profile name, server occurrence, and time stamp, as shown in Figure 3-66. This figure shows the new predefined trap profile in our environment, which occurred and executed in the JBoss Application Server.

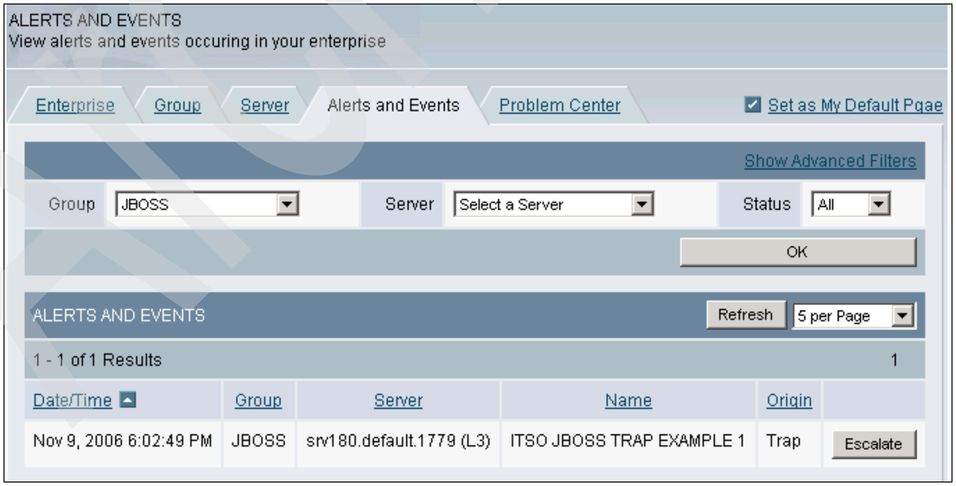


Figure 3-66 Alerts and events management

The analyzing process for the component method trace starts on the Trap and Alert Management page. Select **PROBLEM DETERMINATION** → **Trap & Alert Management** → **Trap Action History** to retrieve the data using the trap action history. The trap action history provides a record of traps that meet the set conditions. Choose the appropriate action and click **Method Trace**. The Trap Method Trace page opens, as shown in Figure 3-67.

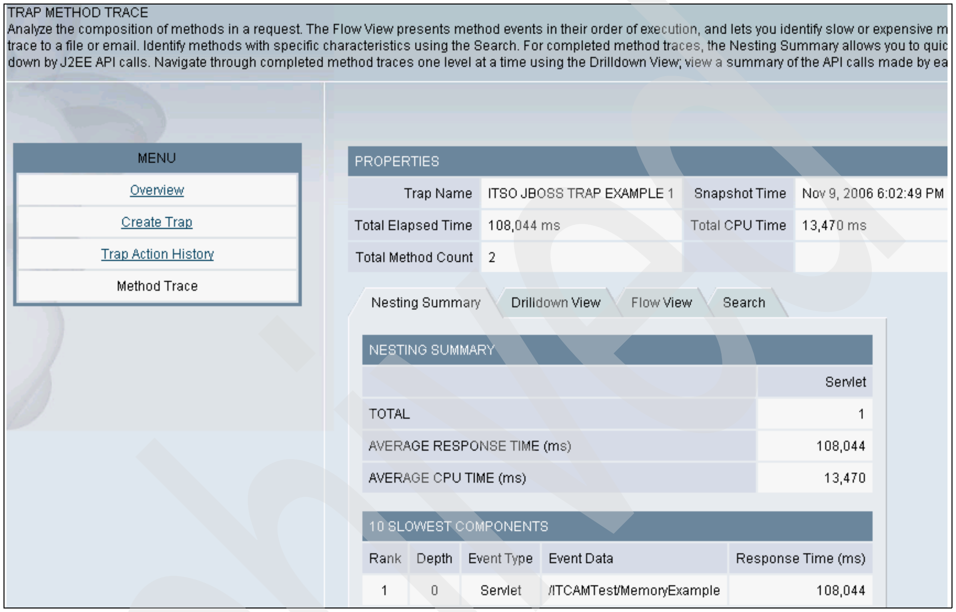


Figure 3-67 Trap method trace: Summary view

By analyzing the composition of methods in the request, the nested summary enables you to quickly access the likely causes of problems within a request, separated by J2EE API calls. The trap method trace provides the following problem determination views:

- Drilldown view
- Flow view
- Search-specific characteristic view

The Drilldown view supports navigating through completed method traces one level at a time, as shown in Figure 3-68. Each entry shows the data as execution order, event type, event data, resident time, and CPU time.

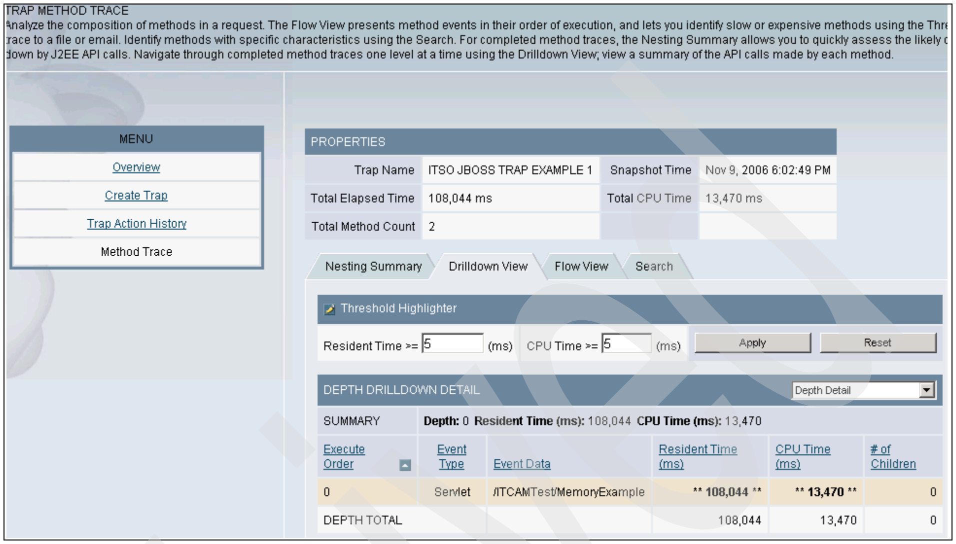


Figure 3-68 Trap method trace: Drilldown view

The Flow view presents method events in their order of execution and lets you identify slow or expensive methods using the threshold highlighter and export the method trace to a file or e-mail. Depth is ordered by servlet entry and servlet exit and indicates the elapsed time and CPU time. Figure 3-69 displays the Flow view of our method trace.

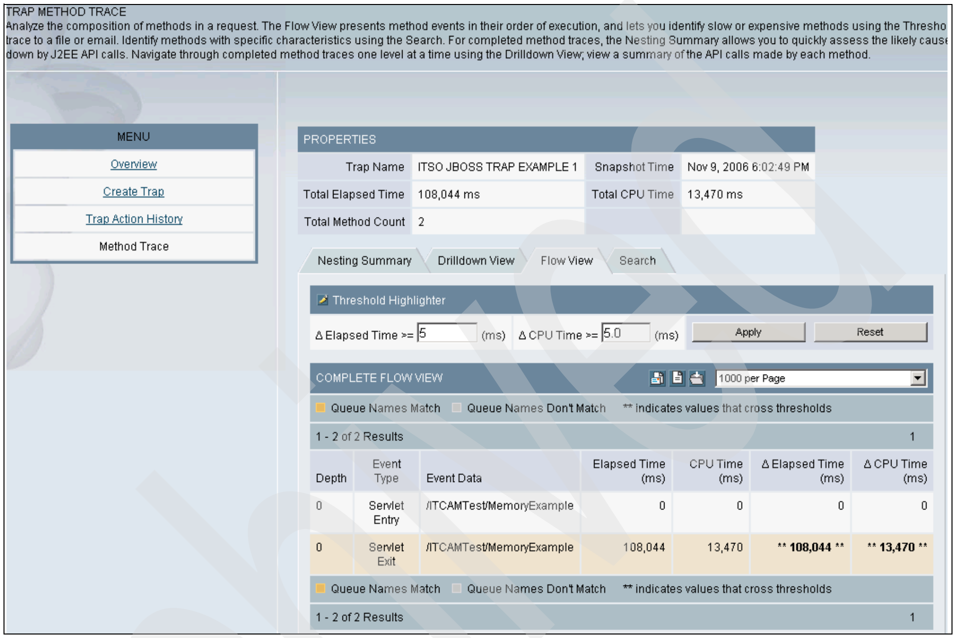


Figure 3-69 Trap method trace: Flow view

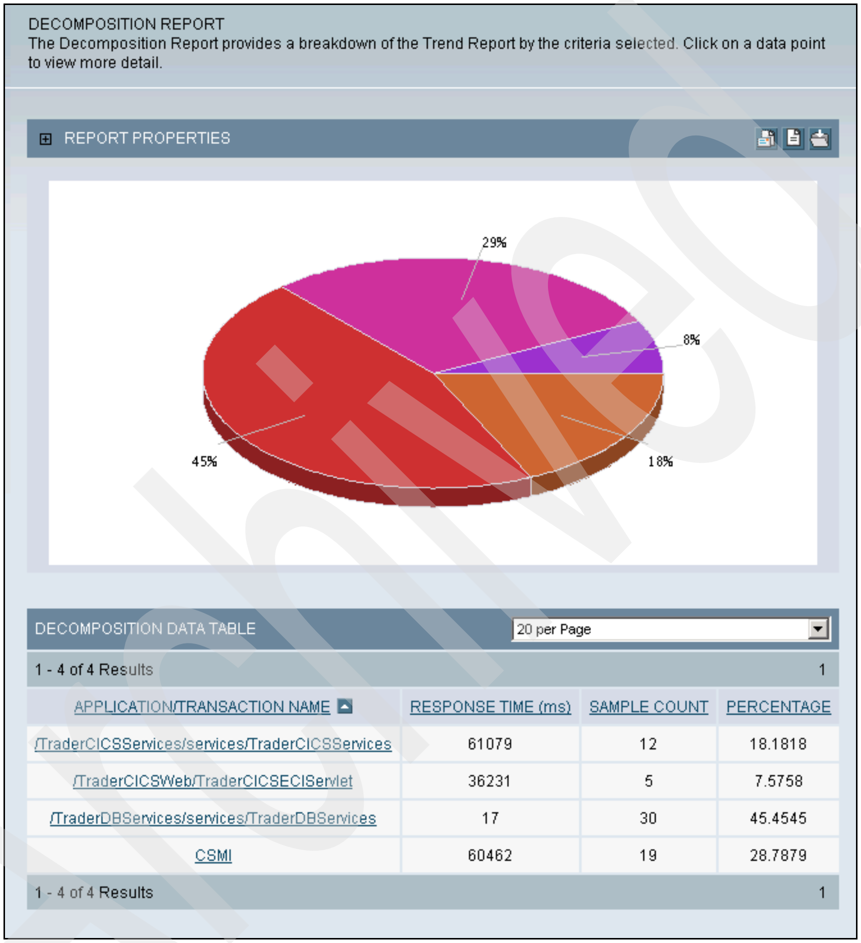
3.10 Composite transaction analysis

One of the key benefits of ITCAM for WebSphere and J2EE is the ability to track a transaction across the J2EE application server to the back-end system, CICS or IMS. The composite transaction can be seen from either the server activity display or the transaction report. The topics in this section are:

- 3.10.1, “Transaction report” on page 135
- 3.10.2, “In-flight request search” on page 140

3.10.1 Transaction report

The request/transaction report (invoked from **PERFORMANCE ANALYSIS** → **Create Application Reports** → **Request/Transaction**) enables you to collect transaction requests on specific servers. We requested the transaction from the server group TraderCICSdistr presented by WebSphere Application Server and the CICS transaction server, as discussed in 3.6, “Transaction reporting” on page 87. This time, we have a closer look at the TraderCICSServices application, which is hosted from the WebSphere application server, by clicking the application.



APPLICATION/TRANSACTION NAME	RESPONSE TIME (ms)	SAMPLE COUNT	PERCENTAGE
/TraderCICSServices/services/TraderCICSServices	61079	12	18.1818
/TraderCICSWeb/TraderCICSECIServlet	36231	5	7.5758
/TraderDBServices/services/TraderDBServices	17	30	45.4545
CSMI	60462	19	28.7879

1 - 4 of 4 Results 1

Figure 3-70 Decomposition report

Going into the details for TraderCICSServices, you can see the composite application icon ( ). Clicking the composite icon displays the Flow view, as shown in Figure 3-71. (Note that for this figure, we had to cut out part of these calls because the nesting level is quite deep.)

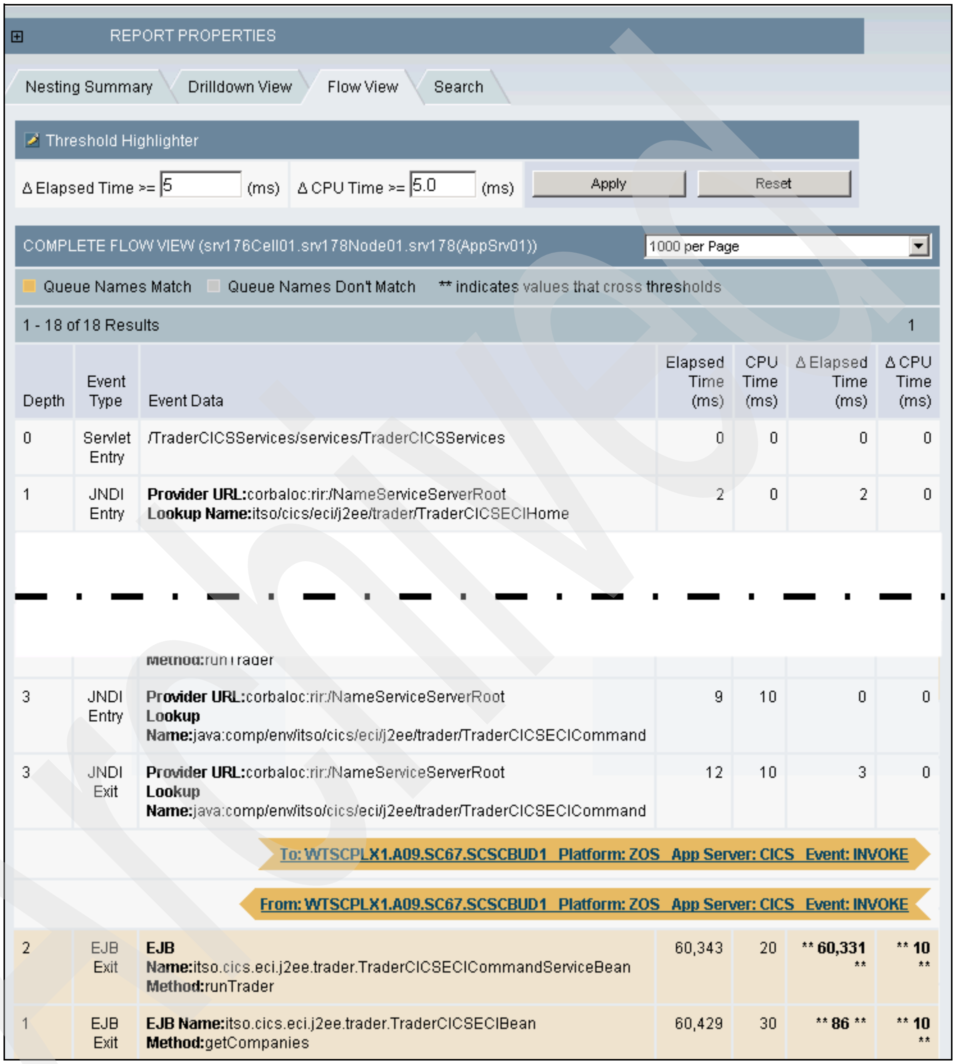
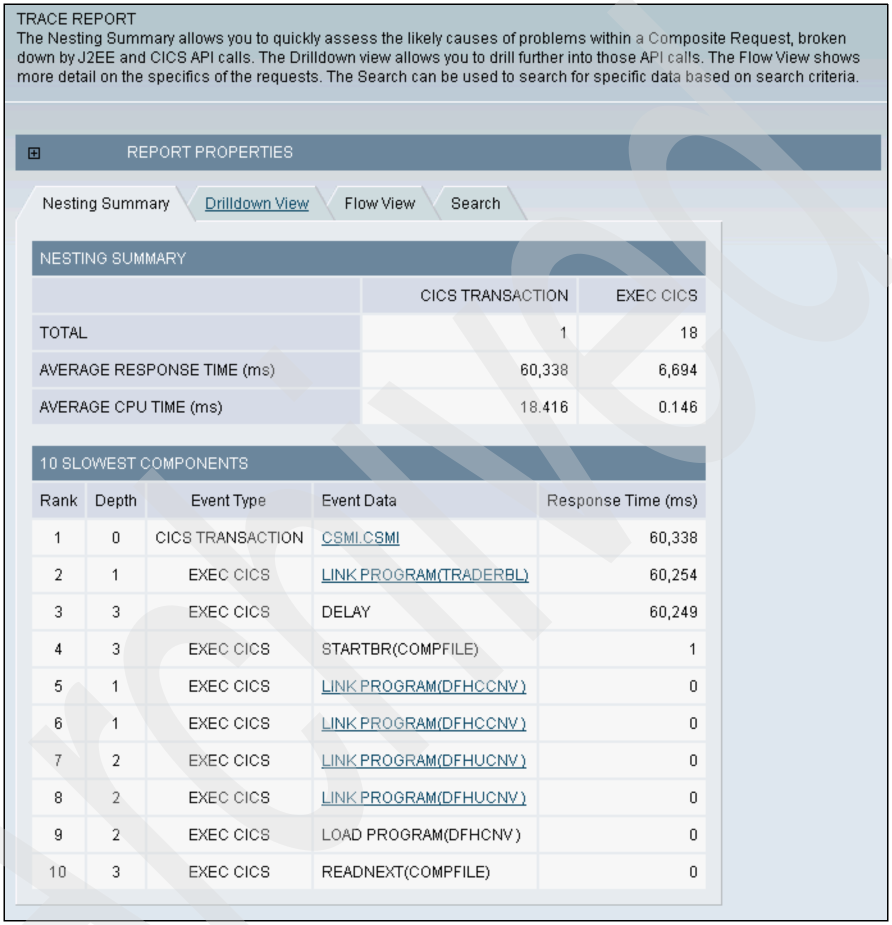


Figure 3-71 Flow view for TraderCICSServices

From the Flow view, you can click the composite link (arrow) to go to its partner transaction in CICS. We are now in the CICS CSMI transaction. This explains why TraderCICSServices has an even higher response time than the CICS CSMI transaction. TraderCICSServices includes the response time of the CSMI transaction. Figure 3-72 shows the nesting summary of the CSMI transaction.



TRACE REPORT

The Nesting Summary allows you to quickly assess the likely causes of problems within a Composite Request, broken down by J2EE and CICS API calls. The Drilldown view allows you to drill further into those API calls. The Flow View shows more detail on the specifics of the requests. The Search can be used to search for specific data based on search criteria.

REPORT PROPERTIES

Nesting Summary | Drilldown View | Flow View | Search

NESTING SUMMARY

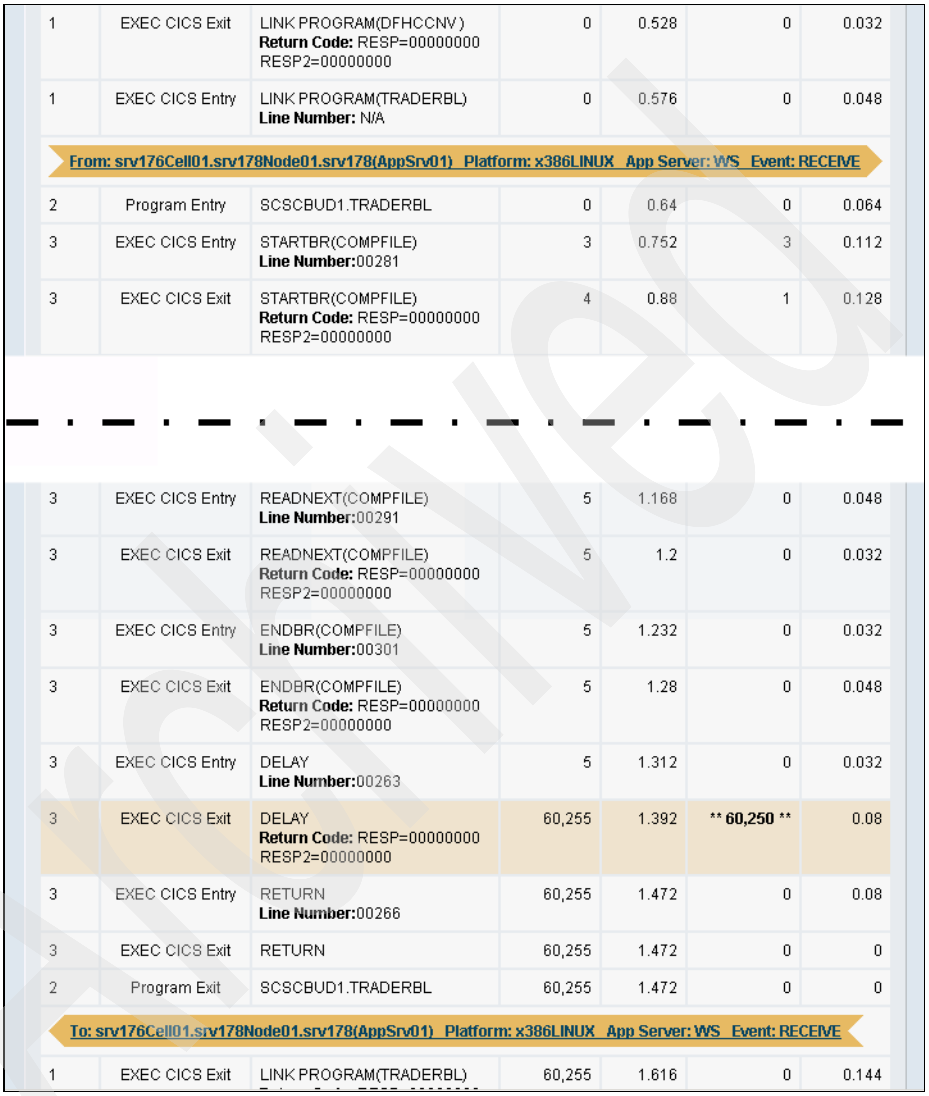
	CICS TRANSACTION	EXEC CICS
TOTAL	1	18
AVERAGE RESPONSE TIME (ms)	60,338	6,694
AVERAGE CPU TIME (ms)	18.416	0.146

10 SLOWEST COMPONENTS

Rank	Depth	Event Type	Event Data	Response Time (ms)
1	0	CICS TRANSACTION	CSMI.CSMI	60,338
2	1	EXEC CICS	LINK PROGRAM(TRADERBL)	60,254
3	3	EXEC CICS	DELAY	60,249
4	3	EXEC CICS	STARTBR(COMPFILE)	1
5	1	EXEC CICS	LINK PROGRAM(DFHCCNV)	0
6	1	EXEC CICS	LINK PROGRAM(DFHCCNV)	0
7	2	EXEC CICS	LINK PROGRAM(DFHUCNV)	0
8	2	EXEC CICS	LINK PROGRAM(DFHUCNV)	0
9	2	EXEC CICS	LOAD PROGRAM(DFHCNV)	0
10	3	EXEC CICS	READNEXT(COMPFILE)	0

Figure 3-72 CSMI transaction

Figure 3-73 shows the CICS transaction in the flow view. You can see the incoming WebSphere-based transaction shown as the first arrow. The second arrow shows the exit from CICS back to WebSphere.



1	EXEC CICS Exit	LINK PROGRAM(DFHCCNV) **Return Code:** RESP=00000000 RESP2=00000000	0	0.528	0	0.032
1	EXEC CICS Entry	LINK PROGRAM(TRADERBL) **Line Number:** N/A	0	0.576	0	0.048
From: srv176Cell01.srv178Node01.srv178(AppSrv01) Platform: x386LINUX App Server: WS Event: RECEIVE						
2	Program Entry	SCSCBUD1.TRADERBL	0	0.64	0	0.064
3	EXEC CICS Entry	STARTBR(COMPFILE) **Line Number:**00281	3	0.752	3	0.112
3	EXEC CICS Exit	STARTBR(COMPFILE) **Return Code:** RESP=00000000 RESP2=00000000	4	0.88	1	0.128
3	EXEC CICS Entry	READNEXT(COMPFILE) **Line Number:**00291	5	1.168	0	0.048
3	EXEC CICS Exit	READNEXT(COMPFILE) **Return Code:** RESP=00000000 RESP2=00000000	5	1.2	0	0.032
3	EXEC CICS Entry	ENDBR(COMPFILE) **Line Number:**00301	5	1.232	0	0.032
3	EXEC CICS Exit	ENDBR(COMPFILE) **Return Code:** RESP=00000000 RESP2=00000000	5	1.28	0	0.048
3	EXEC CICS Entry	DELAY **Line Number:**00263	5	1.312	0	0.032
3	EXEC CICS Exit	DELAY **Return Code:** RESP=00000000 RESP2=00000000	60,255	1.392	** 60,250 **	0.08
3	EXEC CICS Entry	RETURN **Line Number:**00266	60,255	1.472	0	0.08
3	EXEC CICS Exit	RETURN	60,255	1.472	0	0
2	Program Exit	SCSCBUD1.TRADERBL	60,255	1.472	0	0
To: srv176Cell01.srv178Node01.srv178(AppSrv01) Platform: x386LINUX App Server: WS Event: RECEIVE						
1	EXEC CICS Exit	LINK PROGRAM(TRADERBL)	60,255	1.616	0	0.144

Figure 3-73 Flow view CICS transaction

3.10.2 In-flight request search

From the in-flight request search, the current request's view shows incomplete and running requests on the application server. Most of the time, these transactions are quick, and we cannot capture what is happening inside the transaction.

However, for some slow transactions we can collect additional information from the in-flight request search, as shown in Figure 3-74.

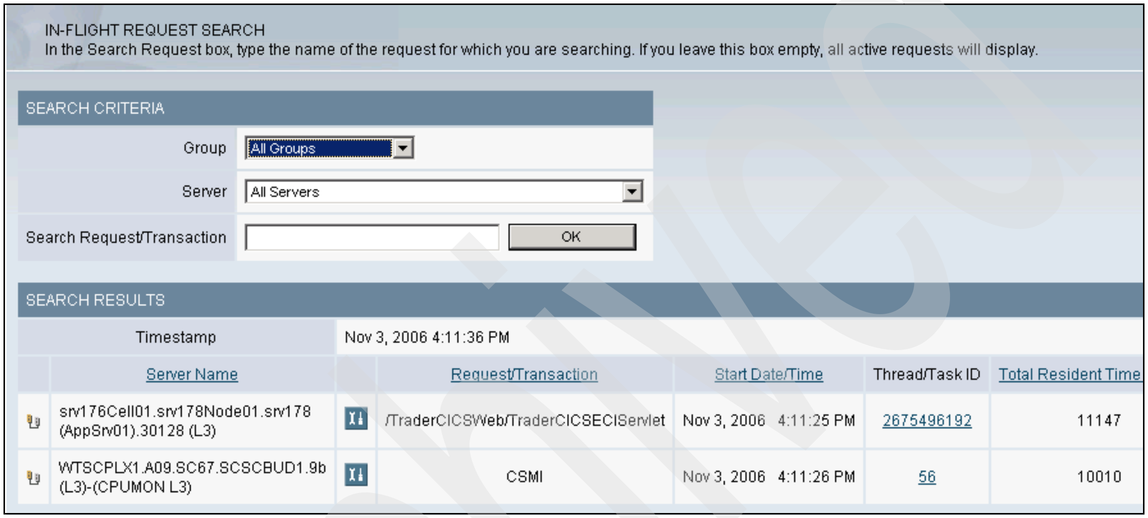


Figure 3-74 In-flight request list

Figure 3-75 shows the incomplete TraderCICSService in the srv178 server.

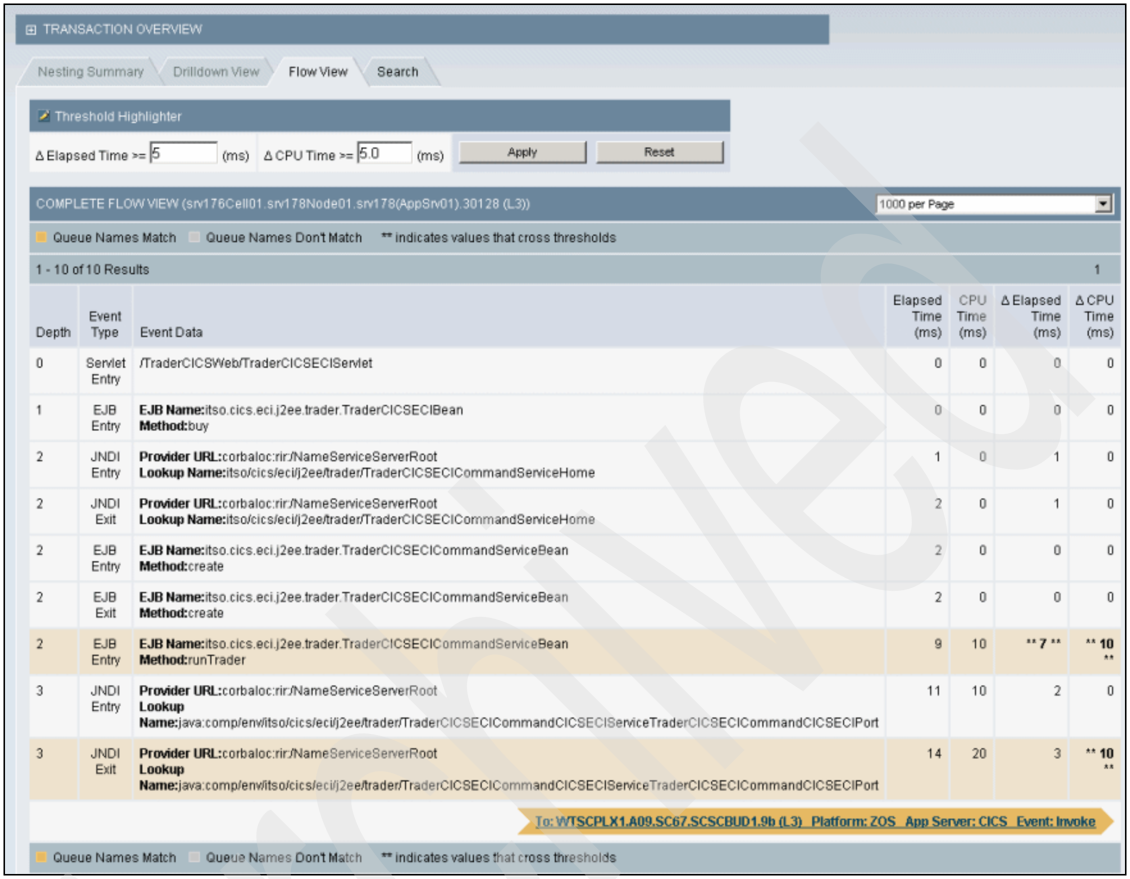


Figure 3-75 Incomplete TraderCICSService

In Figure 3-75 on page 141, you can see that the transaction is calling the CSMI transaction in CICS. Following the link takes you to the CICS transaction, as shown in Figure 3-76.

1 - 21 of 21 Results 1

Depth	Event Type	Event Data	Elapsed Time (ms)	CPU Time (ms)	Δ Elapsed Time (ms)	Δ CPU Time (ms)
0	CICS TRANSACTION Entry	CSMI.CSMI	0	0	0	0
1	EXEC CICS Entry	LINK PROGRAM(DFHCCNV) **Line Number:** N/A	0	0	0	0
2	EXEC CICS Entry		0	0	0	0
2	EXEC CICS Exit	**Return Code:** RESP=00000000 RESP2=00000000	0	0	0	0
2	EXEC CICS Entry		0	0	0	0
2	EXEC CICS Exit	**Return Code:** RESP=00000000 RESP2=00000000	0	0	0	0
2	EXEC CICS Entry	LINK PROGRAM(DFHUCNV) **Line Number:** N/A	0	0	0	0
3	EXEC CICS Entry		0	0	0	0
3	EXEC CICS Exit		0	0	0	0
2	EXEC CICS Exit	**Return Code:** RESP=00000000 RESP2=00000000	0	0	0	0
1	EXEC CICS Exit	**Return Code:** RESP=00000000 RESP2=00000000	0	0	0	0
1	EXEC CICS Entry	LINK PROGRAM(TRADERBL) **Line Number:** N/A	0	0	0	0
From: srv176Cell01.srv178Node01.srv178(AppSrv01).30128 (L3) Platform: x386LINUX App Server: WS Event: Receive						
2	Program/Component Trace Entry	SCSCBUD1.TRADERBL	0	0	0	0
3	EXEC CICS Entry		3	0	3	0
3	EXEC CICS Exit	**Return Code:** RESP=00000000 RESP2=00000000	4	0	1	0
3	EXEC CICS Entry		4	0	0	0
3	EXEC CICS Exit	**Return Code:** RESP=00000013 RESP2=00000080	6	0	2	0
3	EXEC CICS Entry		6	0	0	0
3	EXEC CICS Exit	**Return Code:** RESP=00000018 RESP2=00000100	7	0	1	0
3	EXEC CICS Entry		8	0	1	0

Figure 3-76 Incomplete CICS transaction

Because we were running in a test environment, we injected a delay timer in the program so that we could capture the thread.

3.11 Using Tivoli Enterprise Portal

Tivoli Enterprise Portal provides a portal-based consolidated view of data collected by ITCAM for WebSphere and ITCAM for J2EE and data collected by other IBM Tivoli agents. It also provides mechanisms to tailor views of the data according to user needs.

Tivoli Enterprise Portal can be used by anyone who needs to monitor and analyze many servers or technology types from a common user interface. Usually, this is most beneficial to operational areas supporting many systems or application support for complex applications.

The default application monitoring from ITCAM for WebSphere and J2EE is shown in the physical navigator tree. This section outlines the features of the physical views of ITCAM for WebSphere information. It is likely that operations are more likely to use these views because it is easier to associate aspects of the application with physical hardware.

Note that most of the information that you can view in Tivoli Enterprise Portal is similar to the information that you collect from the ITCAM for WebSphere and J2EE Web console under **AVAILABILITY** → **System Resources**.

In the following example, we show the integration into Tivoli Enterprise Portal from the ITCAM for WebSphere perspective.

3.11.1 WebSphere summary workspace

Immediately under the WebSphere Agent in the physical view hierarchy is the WebSphere App Server workspace. Figure 3-77 shows this workspace for our sample environment.

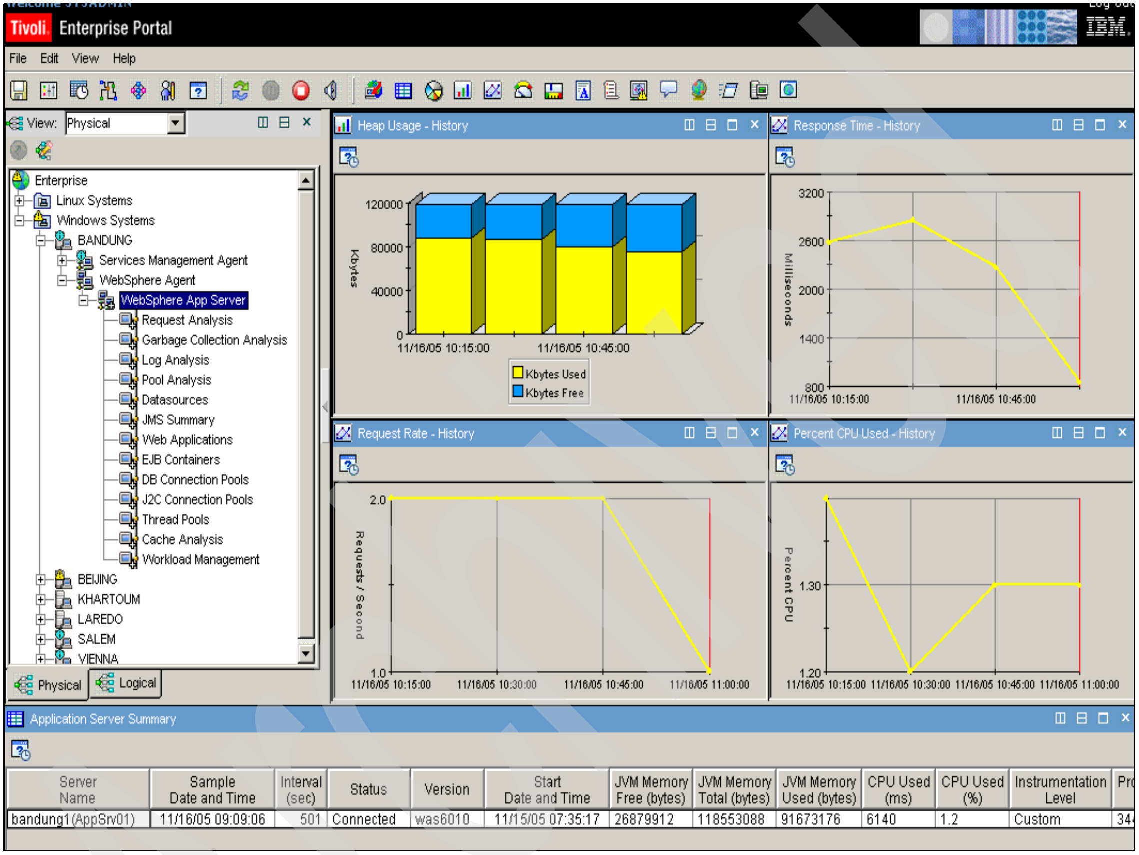


Figure 3-77 WebSphere App Server workspace

The default workspace shows high-level metrics to establish the health of this WebSphere Application Server 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, you can change the time frame so that, in our case, we can display the heap usage trend over the past month. This can provide support staff or capacity planners with much better information on which to base their decisions. This section takes you through some of the interesting areas of our views.

3.11.2 Garbage collection workspace

Figure 3-78 shows our WebSphere Application Server garbage collection performing adequately. The heap usage shows a healthy amount of free memory, and the garbage collection is being initiated regularly but not adversely affecting performance.

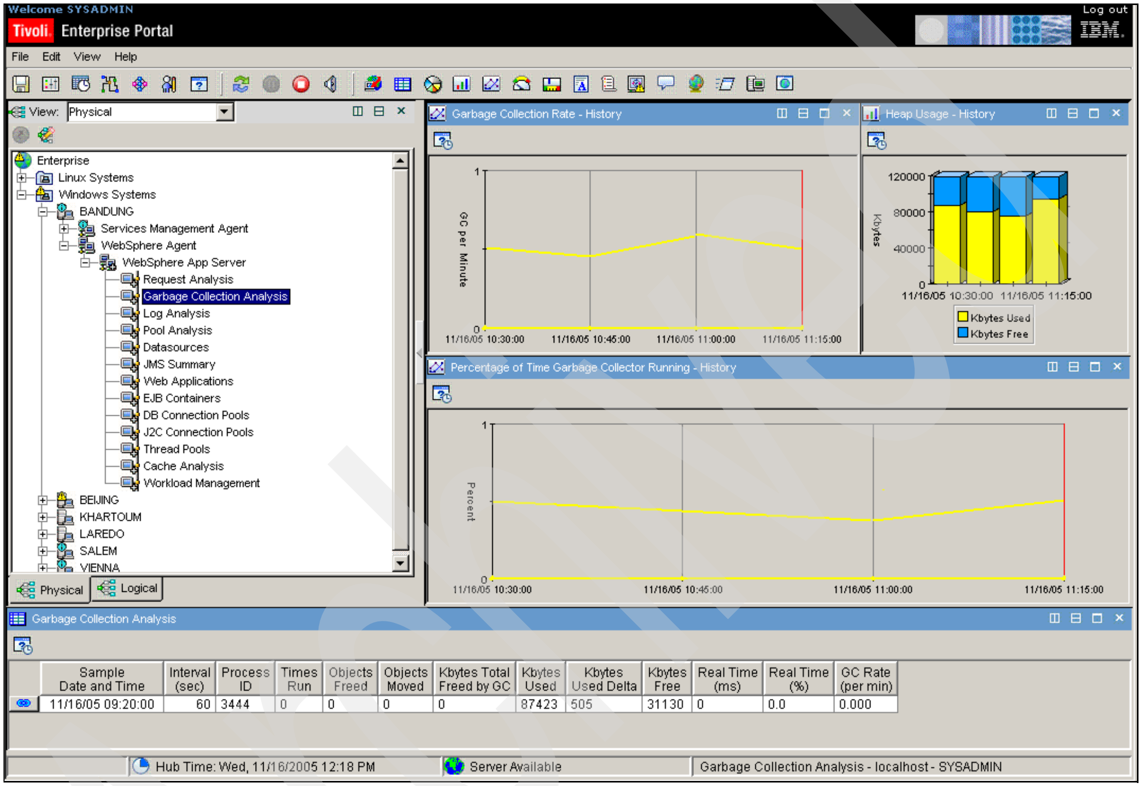


Figure 3-78 Garbage collection normal view

Garbage collection parameters must be tuned to obtain the optimum balance. Figure 3-79 shows an example where it is being forced into action periodically due to poorly scheduled collection or a badly performing application that is forcing the heap too high.

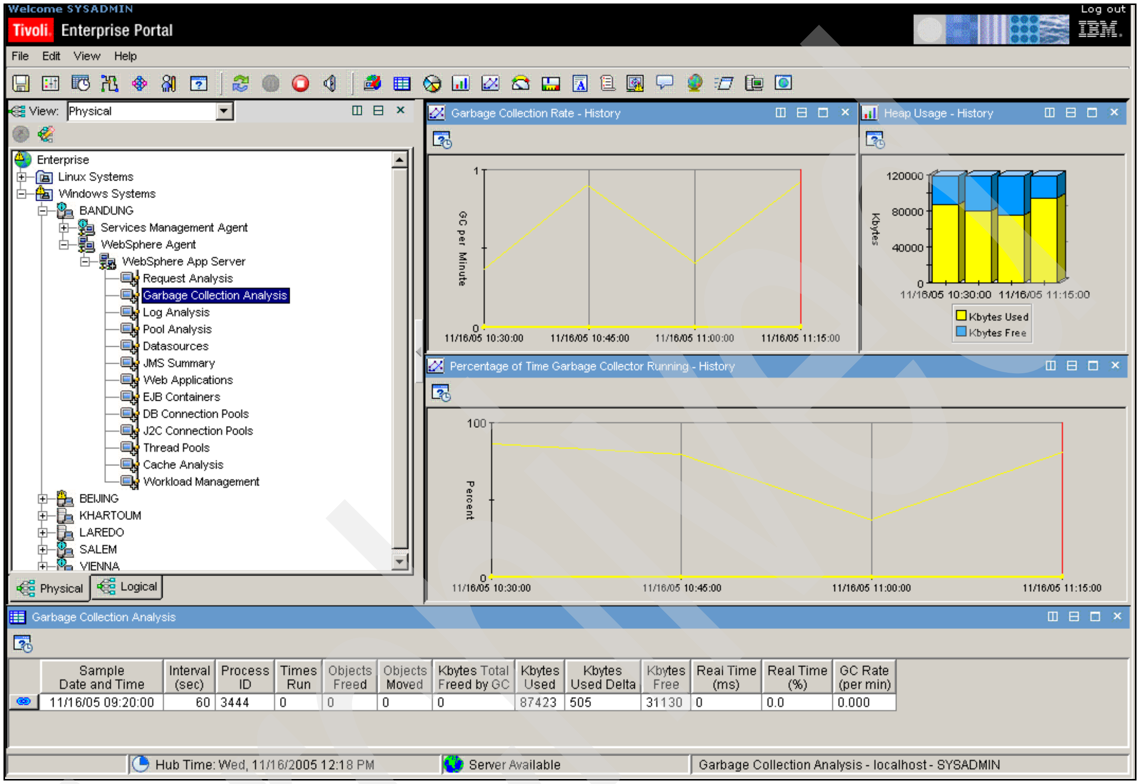


Figure 3-79 Garbage collection poor performance

This example shows the other extreme, where there is no forced collection being initiated. The scheduled collection is almost non-existent, yet the heap is consistently high. The panel shown in Figure 3-80 highlights another negative impact to the business service if this situation occurs because the application is always waiting for free memory.

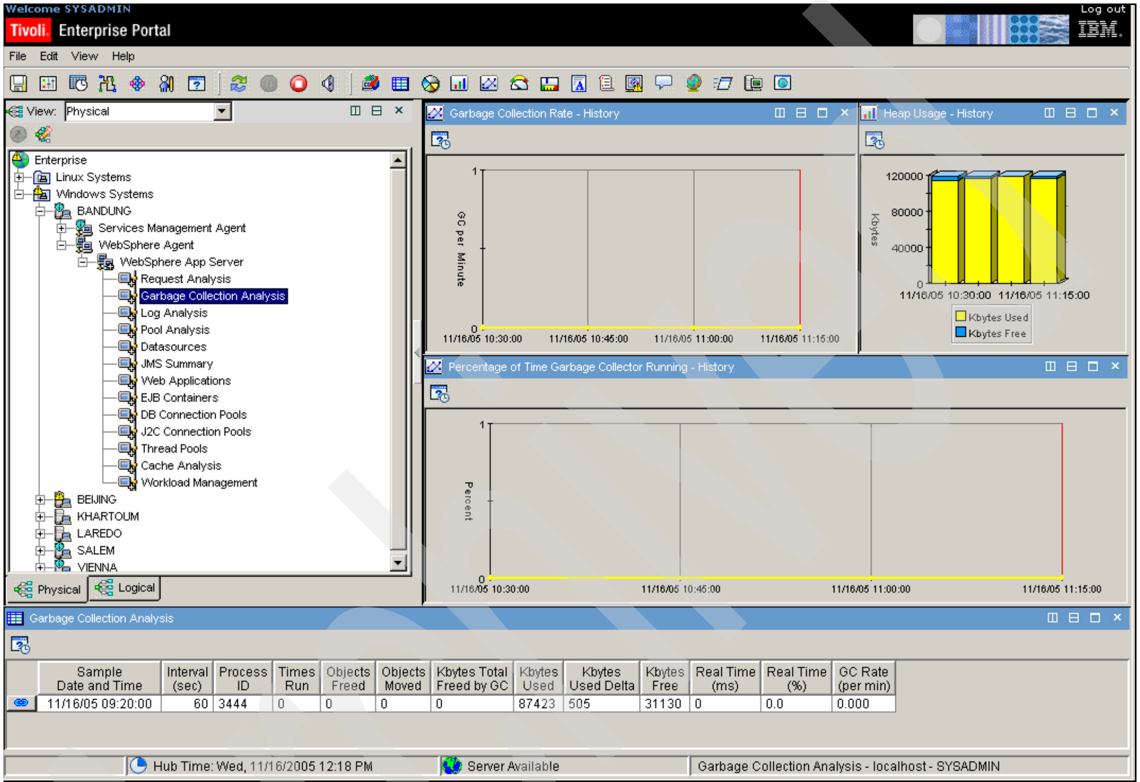


Figure 3-80 Garbage collection high heap

3.11.3 Web applications workspace

The Web applications view provides high-level usage and performance data, such as that shown for our trader application in Figure 3-81.

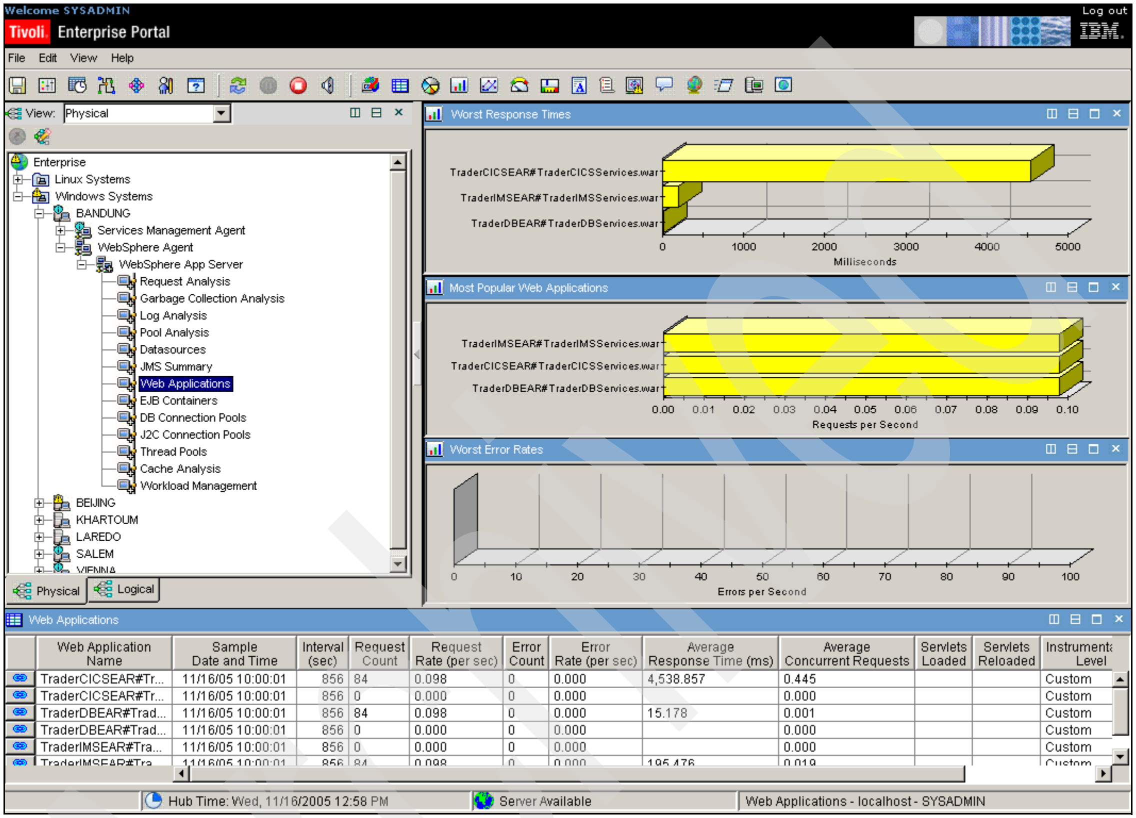


Figure 3-81 Tivoli Enterprise Portal Web Applications view

Our example clearly shows that each of the IMS, DB2, and CICS Web services calls has roughly the same load. However, note that the Worst Response Times pane indicates that the CICS calls are substantially slower to complete.

3.12 Historical information and reporting

This section discusses the capabilities that ITCAM for WebSphere offers in terms of historical data collection and reporting.

Integration with the Tivoli Enterprise Monitoring Server enables you to exploit all the features of Tivoli Enterprise Monitoring Server, including short-term history processing and the use of Tivoli Data Warehouse VS2.1. Integrating ITCAM for WebSphere with Tivoli Data Warehouse V2.1 means that you make the information collected about your J2EE environment available for other management tools such as IBM Tivoli Service Level Advisor or your standard capacity planning software.

The Tivoli Data Warehouse option is a complement of the available historical processing with the standard reporting function of ITCAM for WebSphere.

Tivoli Enterprise Monitoring Server history is independent of ITCAM for WebSphere data storage. Although you use Tivoli Enterprise Monitoring Server history based on your requirements, there is always a minimum level of history collection recommended in ITCAM for WebSphere:

- ITCAM for WebSphere historical data is used for shorter periods of time to show transaction behavior. We do not recommend keeping the data for a long period of time without pruning for performance reasons.
- The Tivoli Data Warehouse solution with Tivoli Enterprise Monitoring Server is an offline reporting database. Some support of summarization exists there, thus allowing data to be kept for longer periods of time to show trends. The information might not be as detailed as the transaction information in ITCAM for WebSphere database.

In Tivoli Enterprise Monitoring Server, we distinguish between to two types of historical information:

Short-term history This refers to historical information stored in flat files for the specific agent in a 24-hour interval. Short-term history information is generated by accumulating real-time data and writing it off to history files at predefined intervals.

Long-term history Long-term history data is stored in Tivoli Data Warehouse V2.1. Tivoli Enterprise Monitoring Server allows offloading short-term history information to Tivoli Data Warehouse V2.1 at definable intervals. The offload is handled by the data warehouse proxy.

Both types of historical information can be displayed in Tivoli Enterprise Portal, enabling you to combine historical information with real-time data, as shown in Figure 3-82. Report against Tivoli Data Warehouse V2.1 using the standard reporting tools in your installation.

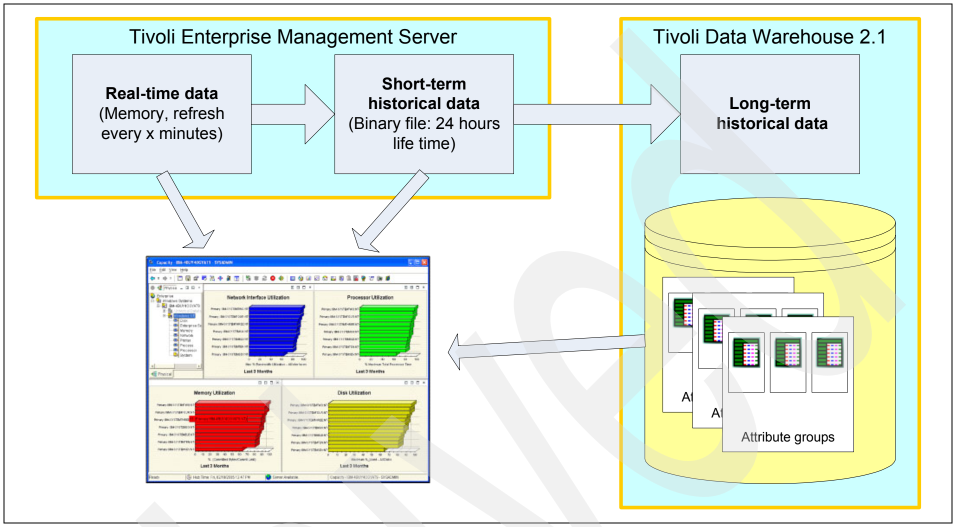


Figure 3-82   Tivoli Enterprise Monitoring Server historical processing

Tivoli Data Warehouse V2.1 supports the DB2 UDB, Microsoft SQL Server™, and Oracle databases. The warehouse proxy agent is the interface between the Tivoli Enterprise Monitoring Server and Tivoli Data Warehouse V2.1, and it transfers short-term history data into the database system used by Tivoli Data Warehouse V2.1. Note that granular data that is available in short-term history files is transferred. All data management within the Tivoli Data Warehouse is done by the summarization and pruning agent. Data can be aggregated at various levels (yearly, quarterly, monthly, weekly, or hourly), and obsolete data can be pruned at regular intervals. An example is summarizing and averaging hourly CPU usage data at a daily or weekly level. For more information about Tivoli Data Warehouse V2.1 and Tivoli Enterprise Monitoring Server historical processing capabilities, see *Getting Started with IBM Tivoli Monitoring 6.1 on Distributed Environments*, SG24-7143.

Tivoli Enterprise Monitoring Server history information is defined and activated at the agent level. Figure 3-83 shows the standard historical collection configuration for the Tivoli Enterprise Monitoring Agent for ITCAM for WebSphere. History is activated out of the box for the following attribute groups:

- Application_Server
- Garbage_Collection_Analysis
- Request_Times_and_Rates
- Thread_Pools

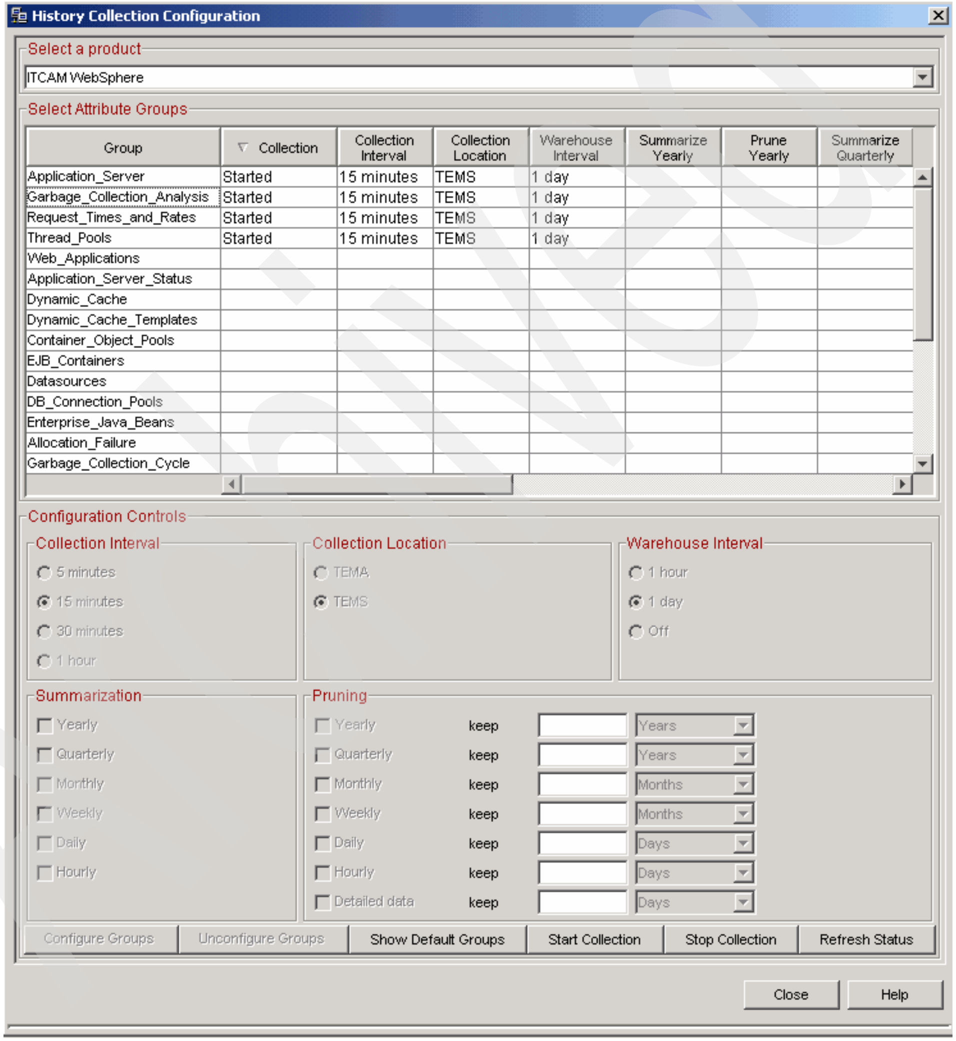


Figure 3-83 ITCAM for WebSphere: history collection configuration

> **Note:** Historical data collection can be defined to happen at the Tivoli Enterprise Monitoring Server of the Tivoli Enterprise Monitoring Agent level. You can change this setting depending on your environment.

Figure 3-84 shows an example of how to use historical data in the Tivoli Enterprise Portal. This shows the WebSphere App Server workspace provided by the Tivoli Enterprise Monitoring Agent for ITCAM for WebSphere. It uses historical information for an overview of the activities in a particular WebSphere Application Server. In this case, we use short-term history information to display heap usage, average response time, request rate, and CPU usage over the past hour.

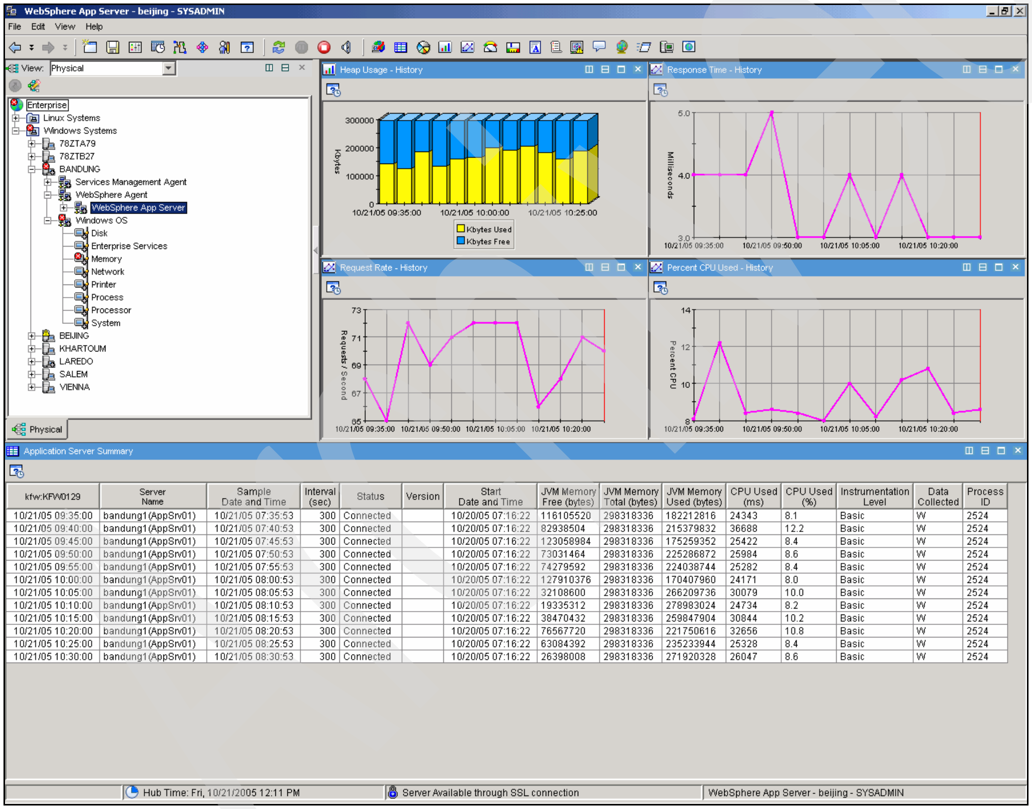


Figure 3-84 Historical information with Tivoli Enterprise Server

The graph also illustrates the advantages of combining ITCAM for WebSphere with other Tivoli Enterprise Monitoring Server agents. From looking at this workspace, we learn that the WebSphere Application Server heap size does not appear to be optimal for this Microsoft Windows machine. The Tivoli Enterprise

Monitoring Server Windows agent, which is also installed on this machine, generates an alert (note the small red icon with a cross in the Windows Memory field in the navigation tree) indicating that Windows is running out of memory. In this special case, Windows is already complaining about memory constraints, but WebSphere Application Server still has not reached it defined memory limits (see the heap usage graph in Figure 3-84 on page 152).

4

ITCAM for Web Resources concepts, installation, usage

The discussion on ITCAM for Web Resources in this chapter is divided into the following sections:

4.1 Product overview

ITCAM for Web Resources provides IT operations with resource and application monitoring to quickly identify, isolate, and route issues to the appropriate support personnel using the Tivoli Enterprise Portal. ITCAM for Web Resources does not provide deep dive diagnostic functionality, such as method tracing, and does not require a separate managing server. This allows ITCAM for Web Resources to be installed and configured quickly.

ITCAM for Web Resources provides support for the following Web server and J2EE application servers:

- Web servers:
  - IIS
  - Apache
  - IBM HTTP Server
  - Sun ONE
- Application servers:
  - WebSphere
  - Netweaver
  - Oracle
  - Tomcat
  - JBOSS
  - WebLogic
  - WebSphere ESB
  - WebSphere Portal Server
  - WebSphere Process Server
  - Lotus® Workplace™ Server
  - J2SE Applications
  - WebLogic Portal

The following are some of the important features of ITCAM for Web Resources:

- Enhanced ease of use and time to value

  A single, consolidated user interface built on Tivoli Enterprise Portal provides for the ability to combine multiple sources of information into a single workspace, showing operating system, database application server metrics from a single user interface.
- Simple installation and configuration

  Best practices for creation of logical views and creation of situation correlation. See:

  http://catalog.lotus.com/wps/portal/tcam

- Application dashboard

  Summary view of applications to give operators and administrators, at a glance, the ability to see where the problem is located, such as client tier, application tier, or backend tier. Drill down by tier to view, at a glance, whether the problem is related to a memory leak, resource pool issue, or other causes.

- Auto learning of thresholds

  Benchmark performance for a new application to understand where thresholds should be established. This feature allows customers to baseline performance of an application and to set thresholds according to the application baseline.

4.2 Architecture and interconnection

ITCAM for Web Resources is performance monitoring applications for J2EE application servers. It collects J2EE application servers' performance metric using a data collector and forwards its information using the IBM Tivoli Monitoring V6.1 infrastructure.

The application servers run the *data collector*, which is a collecting agent that runs in the application server and sends monitoring information using Tivoli Enterprise Monitoring Agent to the Tivoli Enterprise Monitoring Server. These data collectors operate independently of each other. Figure 4-1 shows the overall architecture of ITCAM for Web Resources.

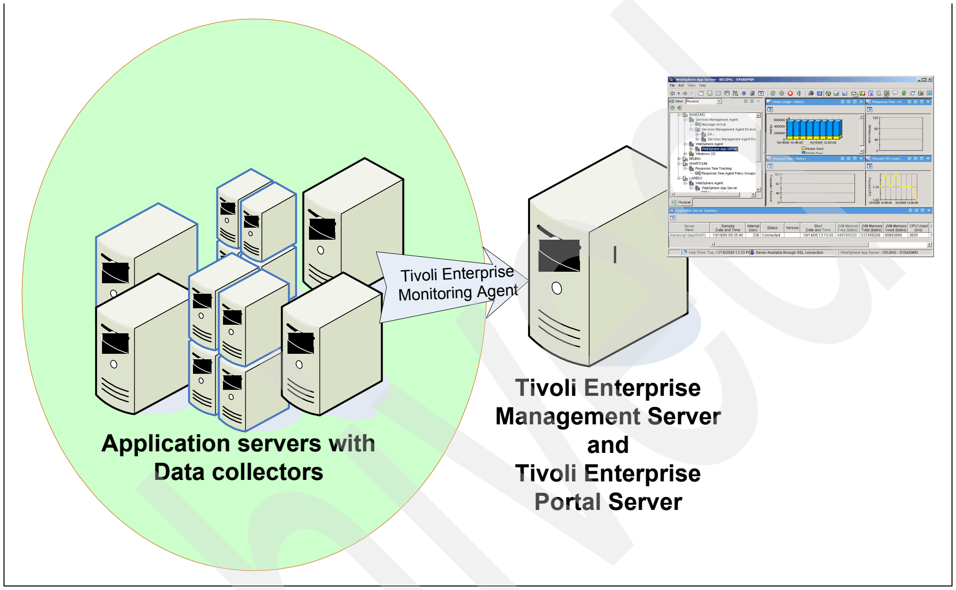


Figure 4-1 ITCAM for Web Resources architecture

The product consists of two main parts: the data collectors and the Tivoli Enterprise Monitoring Agents. A data collector agent runs on each monitored J2EE application server and communicates with the Tivoli Enterprise Monitoring Agent. The Tivoli Enterprise Monitoring Agent sends the performance information to the Tivoli Enterprise Monitoring Server for displays on Tivoli Enterprise Portal.

4.2.1 J2EE and WebSphere data collectors

The data collectors run inside the application servers. They use native system services, and they are tailored for the particular environments where they execute.

Data collectors are configured as a multithreaded process. They consist of the following 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 publish servers according to polling frequencies. This data includes system initialization data, application request-level data, and application method-level data.

- Collector process

  The collector provides the monitoring data for the Tivoli Enterprise Portal. It collects WebSphere Application Server and other J2EE application server performance metrics. This component communicates with the Tivoli Enterprise Monitoring Agent using a TCP/IP port.

The data collectors send the probes into the application servers to analyze the applications' performance. The probes collect monitoring data and feed it to transport routines for the Tivoli Enterprise Monitoring Agent. This relieves the processing burden of ITCAM for Web Resources 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 to the Tivoli Enterprise Monitoring Agent.

The data sources that are employed by the ITCAM for Web Resources data collector are:

- JVMTI garbage collection data, method trace, stack trace, CPU time, and heap dump
- JMX system resources
- PMI system resources (WebSphere only)
- Byte Code Instrumentation (BCI) for some classes

Figure 4-2 shows the conceptual data collector structure of the distributed WebSphere data collector.

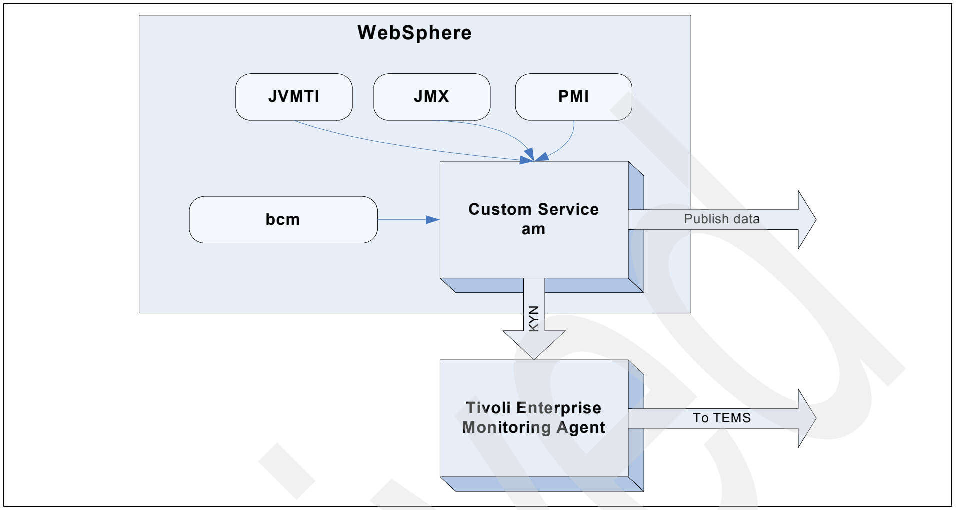


Figure 4-2 J2EE data collector structure

4.2.2 Tivoli Enterprise Monitoring Agent

The ITCAM for Web Resources Tivoli Enterprise Monitoring Agent can forward monitoring information to the Tivoli Enterprise Monitoring Server for monitoring using Tivoli Enterprise Portal. For monitoring Web Servers, you can also use Tivoli Enterprise Monitoring Agent for Web Servers. The Web servers monitoring does not need a data collector on the Web servers.

4.2.3 Data collector and Tivoli Enterprise Monitoring Agent migration

The data collector from ITCAM for WebSphere or ITCAM for J2EE must be patches to be used with ITCAM for Web Resources V6.2. They require fix pack 1 and interim fix 6, as explained in the launchpad of the data collector installation.

The Tivoli Enterprise Monitoring Agent for ITCAM for WebSphere and ITCAM for J2EE are not compatible with ITCAM for Web Resources. They use the same agent codes (YN and YJ) but they have slightly different monitoring capabilities. There are new attributes and facilities with ITCAM for Web Resources V6.2.

ITCAM for Web Resources has two data monitoring switches:

- Request monitoring level, which can be disabled, level 1 or level 2
- Resource monitoring status, which can be enabled or disabled

These settings can be changed using standard IBM Tivoli Monitoring actions. The ITCAM for Web Resources Tivoli Enterprise Monitoring Agent has two monitoring modes, on demand and fix interval.

4.3 Monitoring resources

The discussion for the monitoring resources is divided into:

- 4.3.1, “Monitoring levels” on page 161
- 4.3.2, “Data collector configuration” on page 161
- 4.3.3, “Logging and troubleshooting” on page 162

4.3.1 Monitoring levels

The monitors for ITCAM for WebSphere and ITCAM for J2EE run on a predefined monitoring level. These levels can be changed dynamically without the need to restart the monitored systems. The levels are:

- Level 1: production-level monitoring that provides basic response time and transaction information.
- Level 2: more detailed information that includes method call details and memory usage. In Version 6.1, there is a new method profiling option for this level to collect aggregate method calling statistics with a much lower processing requirement than level 3 monitoring.

4.3.2 Data collector configuration

The data collector configuration is stored in the $DC_HOME/runtime sub directory. It is governed by several configuration files. Among them are:

- custom/toolkit_custom.properties

  This file stores custom settings for bytecode instrumentation properties.
- 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.
- jiti.properties

  Just-in-time instrumentation property for Java class instrumentation profile.
- <wasver>.<node>.<srv>.datacollector.policy

  Java security permission of the data collector.

- <wasver>.<node>.<srv>.datacollector.properties

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

- <wasver>.<node>.<srv>.kwjdc.properties

  Property file for Tivoli Enterprise Monitoring Agent connector.

- <wasver>.<node>.<srv>.toolkit.*

  Instrumentation properties with ITCAM toolkit.

- <node>.<srv>.cyaneaGpsCounter.txt

  Counter that identifies a sequence number for matching a composite transaction.

4.3.3 Logging and troubleshooting

The ITCAM 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. In general, the logging level can be modified, either from the properties file to include am.debug=yes or using the control commands, **`dcctl.sh`** or **`amctl.sh`**.

The distributed data collector logs are in $commondir\CYN\logs. Some logs might be in <admin>.<server> path, such as:

- msg-dc.log
- trace-dc.log
- msg-dc-native.log
- trace-dc-native.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.

4.4 Installation overview

ITCAM for Web Resources is a solution that is based on the IBM Tivoli Monitoring V6.1 platform. It monitors and manages applications running on a commonly available application server and Web servers. The application server platforms include IBM WebSphere, WebLogic, SAP, Oracle, JBoss, Tomcat, J2SE, and IBM WebSphere Application Server Community Edition. The Web servers supported are Microsoft IIS, SUN, and Apache Web Servers.

As discussed in 4.2, “Architecture and interconnection” on page 157, the product consists of the data collector and the Tivoli Enterprise Monitoring Agent that connects to the Tivoli Enterprise Monitoring Server. As such, the installation process consists of:

- 4.4.1, “Installing application support files” on page 163, is a prerequisite for using the agents. The Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server need the information on the agent to be pre-populated.
- 4.4.2, “Installing Tivoli Enterprise Monitoring Agent” on page 168, discusses the installation of Tivoli Enterprise Monitoring Agent for WebSphere.
- 4.4.3, “Installing the data collector” on page 172, describes the necessary installation process for the data collector. This data collector is the same code for both ITCAM for WebSphere and ITCAM for Web Resources.

> **Note:** The installation media contains the Netcool ASM agent. This is an optional component and can integrate to the Micromuse® Netcool solution. Our installation process does not cover this component.

For more information about IBM Tivoli Monitoring consult *Deployment Guide Series: IBM Deployment Guide Series: IBM Tivoli Monitoring 6.1*, SG24-7188. That is a step-by-step deployment guide for IBM Tivoli Monitoring V6.1 and covers small to large environments and discusses best practices for a deployment plan.

Figure 4-3 shows our sample deployment environment. We have the application server node on srv177 and the IBM Tivoli Monitoring server in srv178. This configuration where all IBM Tivoli Monitoring components are running on the same machine is only fitting for a demonstration or proof-of-concept environment.

Figure 4-3 Sample deployment environment

4.4.1 Installing application support files

This section describes the installation of the application support files for the Tivoli Enterprise Monitoring Agent for WebSphere in Windows platform. Before you can view data collected by monitoring agents, you must install and enable application support for the agents. The application support files provide agent-specific information for workspaces, helps, situations, template, and other data.

All monitoring agents require that application support be configured on all instances of the following infrastructure components:

- Tivoli Enterprise Monitoring Server (both hub and remote monitoring servers)
- Tivoli Enterprise Portal Server
- Tivoli Enterprise Portal Desktop Client

You must first install application support files on the Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, and Tivoli Enterprise Portal desktop client. You must acquire the appropriate IBM Tivoli Monitoring environment information, including the host names and communication protocols for the appropriate components listed above.

In our sample deployment environment, we have a single server running Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal, and Tivoli Enterprise Portal Server. This is indicated in the feature selection in Figure 4-4.

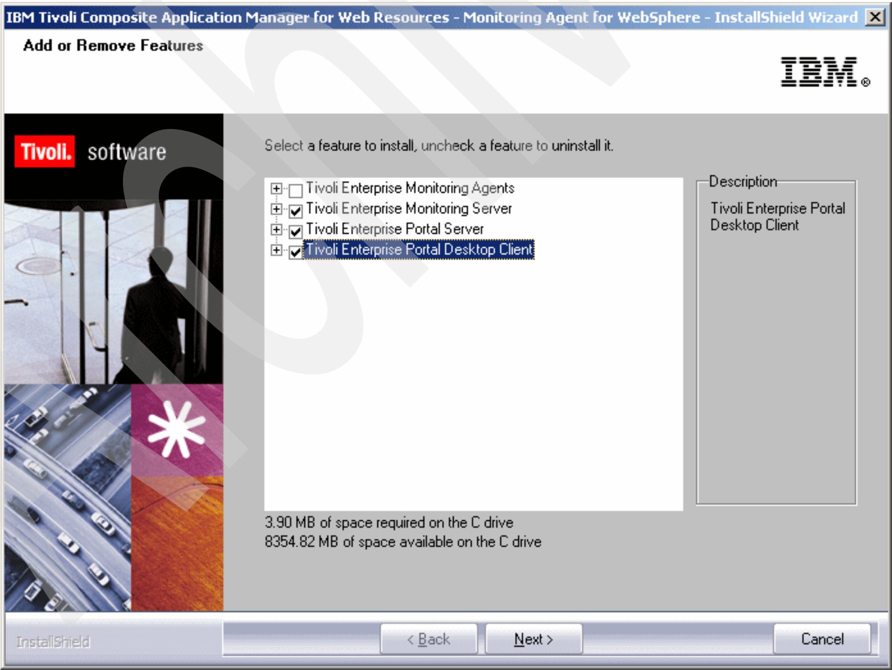


Figure 4-4 Select features

You can use default installation options. When the installation is complete, you are presented with the configuration options shown in Figure 4-5.

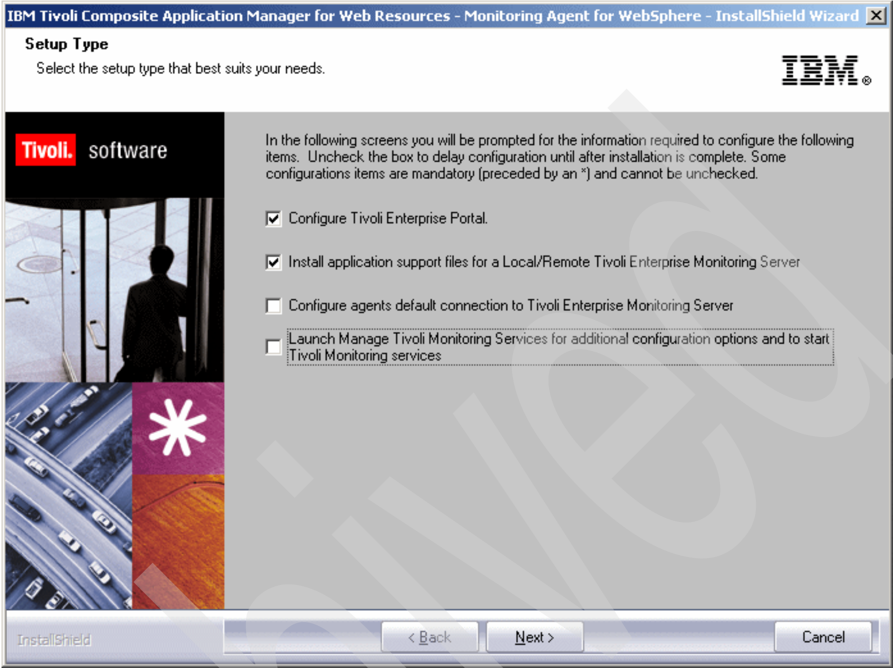


Figure 4-5 Configuration options

You can configure the following:

- The Tivoli Enterprise Portal and Tivoli Enterprise Portal Server configuration. This requests the host name of the place that the Tivoli Enterprise Portal Server resides and rebuilds presentation files.

- The Tivoli Enterprise Monitoring Server definition
  - This defines the communication protocol and communication parameters to connect to the Tivoli Enterprise Monitoring Server. See Figure 4-6.

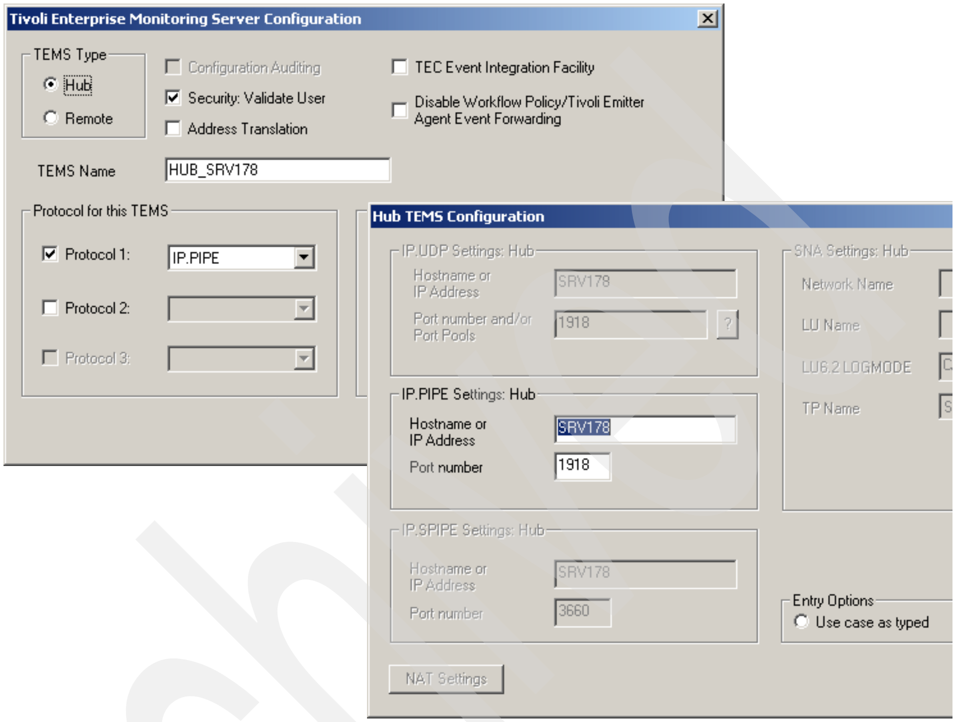


Figure 4-6 Tivoli Enterprise Monitoring Server communication protocol

– This populates the Tivoli Enterprise Monitoring Server with the data for ITCAM for Web Resources. See Figure 4-7. Make sure that the execution result is 0.

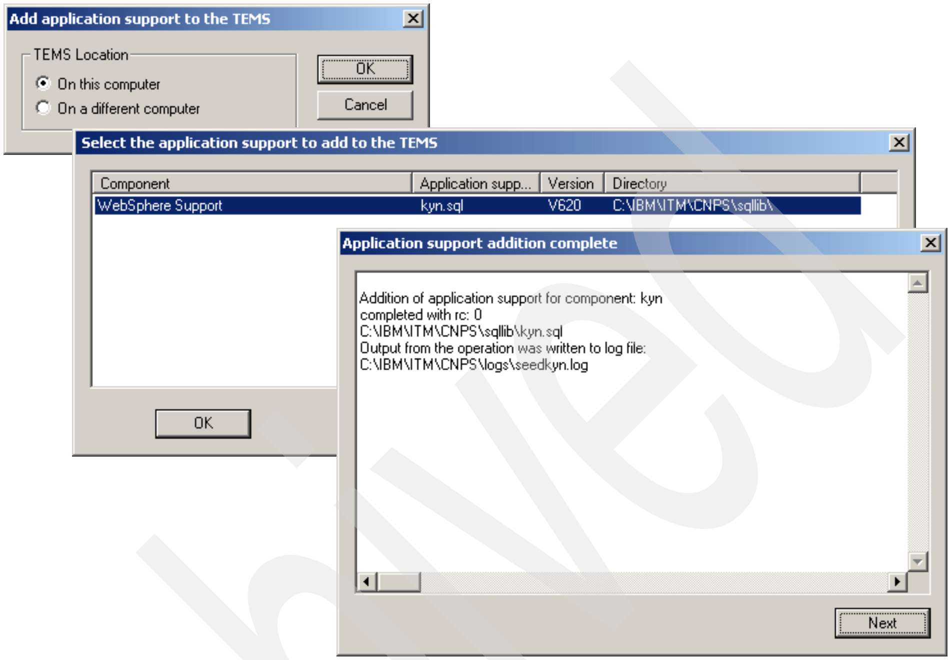


Figure 4-7 Selecting Tivoli Enterprise Monitoring Server

- The next steps configure the agent default connection to the Tivoli Enterprise Monitoring Server. This will be used for remote deployment. Specify the default communication protocol and its parameters for the agent to communicate with the Tivoli Enterprise Monitoring Server. See Figure 4-8.

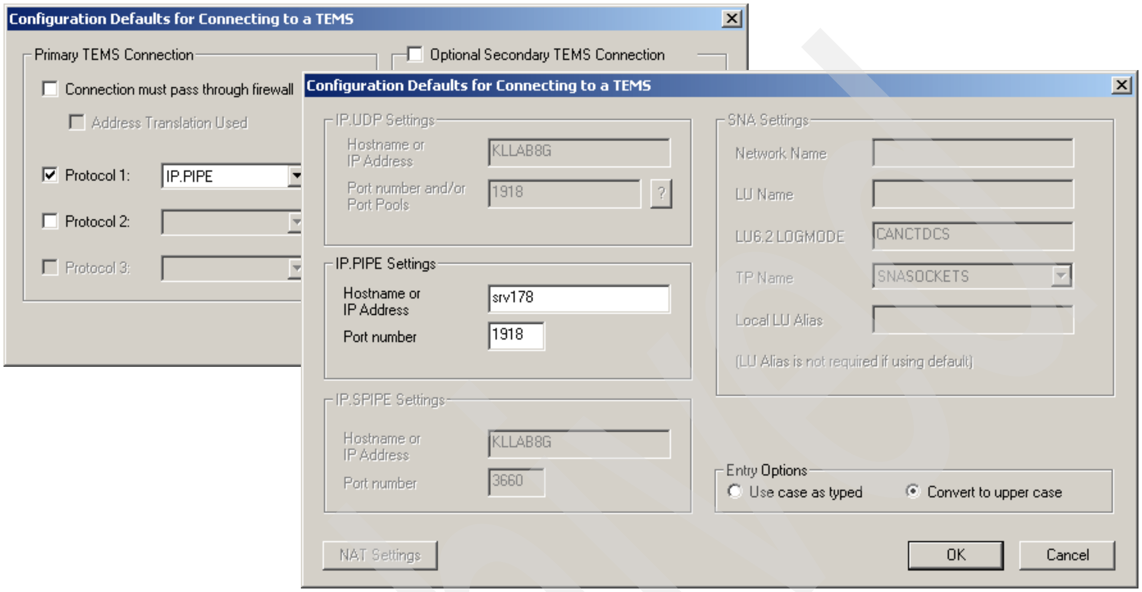


Figure 4-8 Communication protocol defaults

4.4.2 Installing Tivoli Enterprise Monitoring Agent

The ITCAM for Web Resources Tivoli Enterprise Monitoring Agent must be installed to facilitate communication from the data collector to the Tivoli Enterprise Monitoring Server. The Tivoli Enterprise Monitoring Agent installation is not required to be performed on the data collector machine, but for the sake of clarity for the display in Tivoli Enterprise Portal, we highly recommend that you have the Tivoli Enterprise Monitoring Agent installed on the same machine as the data collector.

The installation for the Tivoli Enterprise Monitoring Agent uses the same wizard as the application support files in 4.4.1, "Installing application support files" on page 163.

The Tivoli Enterprise Monitoring Agent for ITCAM for Web Resources uses the same product code as ITCAM for WebSphere or ITCAM for J2EE, therefore an installation of ITCAM for Web Resources removes previous ITCAM for WebSphere Tivoli Enterprise Monitoring Agent or ITCAM for J2EE Tivoli Enterprise Monitoring Agent.

The installation is started by running setup.exe from the WINDOWS subdirectory of the installation media. You can use the default options for the installation. Some of the installation dialog is shown in Figure 4-9. The installation would install IBM GSKit and Java if needed.

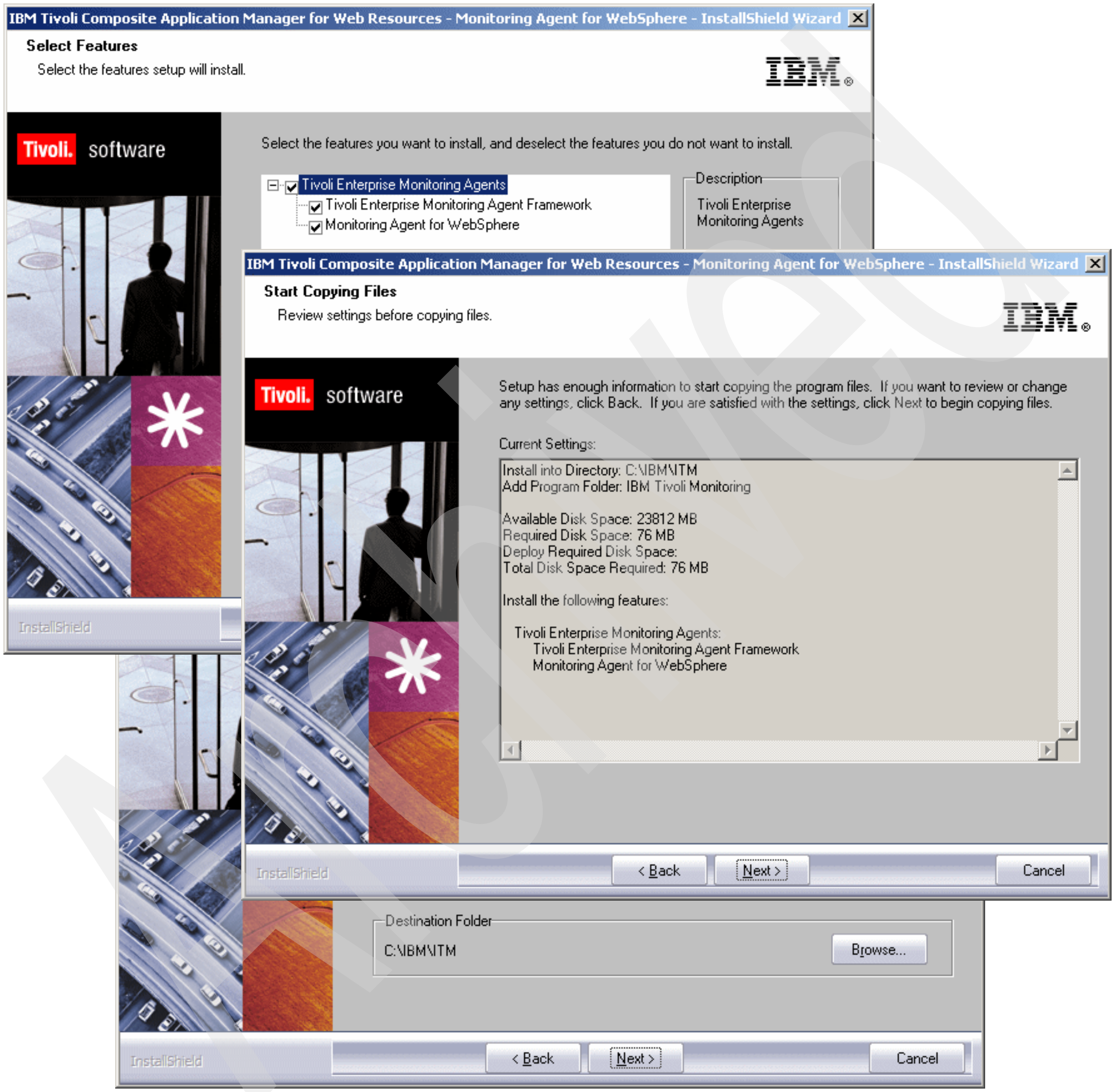


Figure 4-9 Installation dialog

After the installation is complete, the configuration option is shown (Figure 4-10).

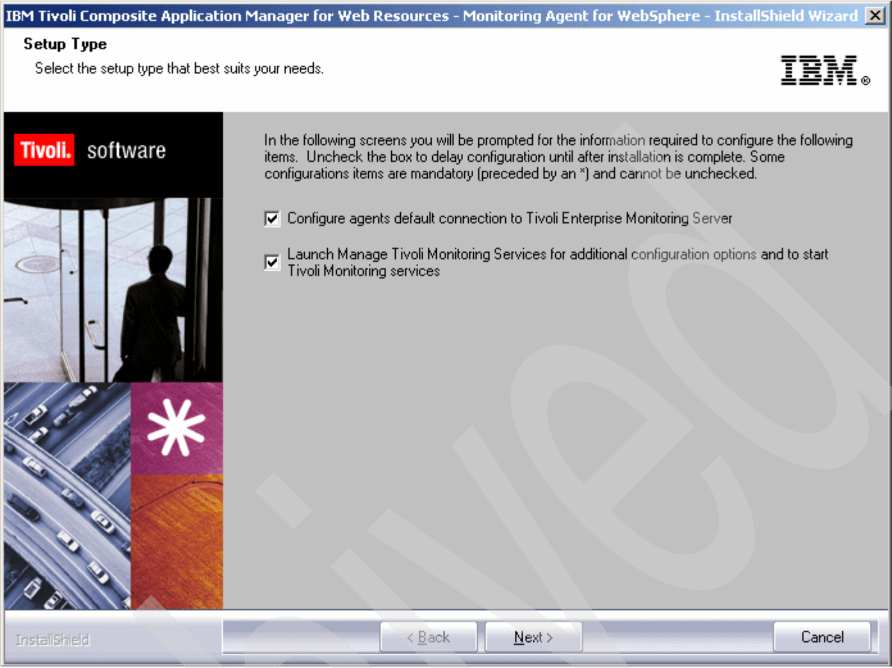


Figure 4-10 Configuration options

From the configuration option, we configure the communication protocol and its parameter to connect to the Tivoli Enterprise Monitoring Server, as shown in Figure 4-11.

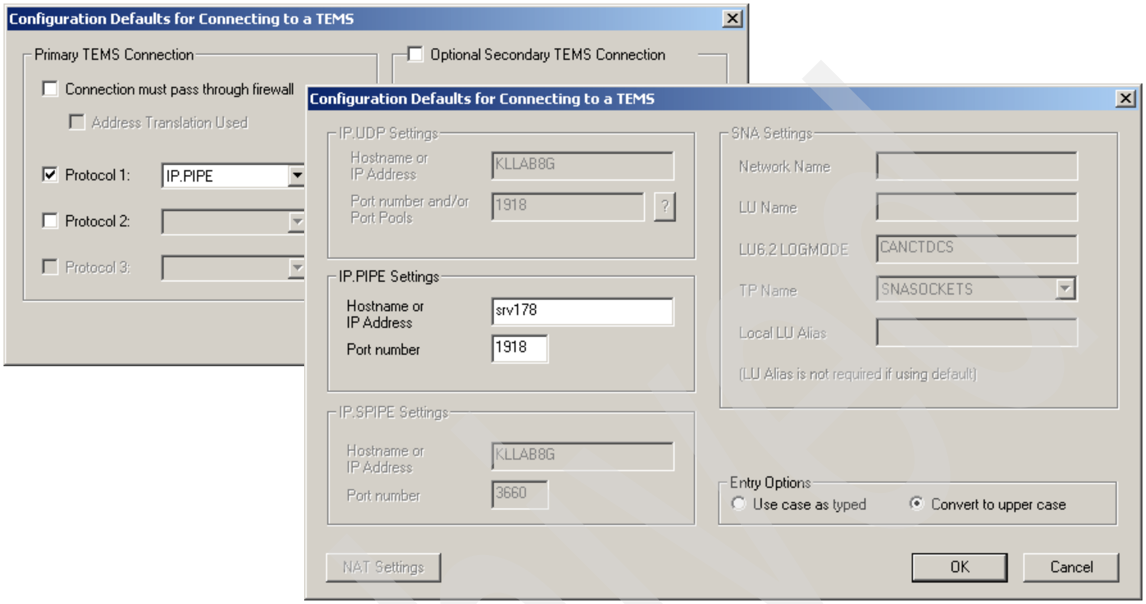


Figure 4-11 Communication to Tivoli Enterprise Monitoring Server

The WebSphere agent configuration dialogs is a tabbed window, shown in Figure 4-12. You can typically accept all the defaults. Click **OK**.

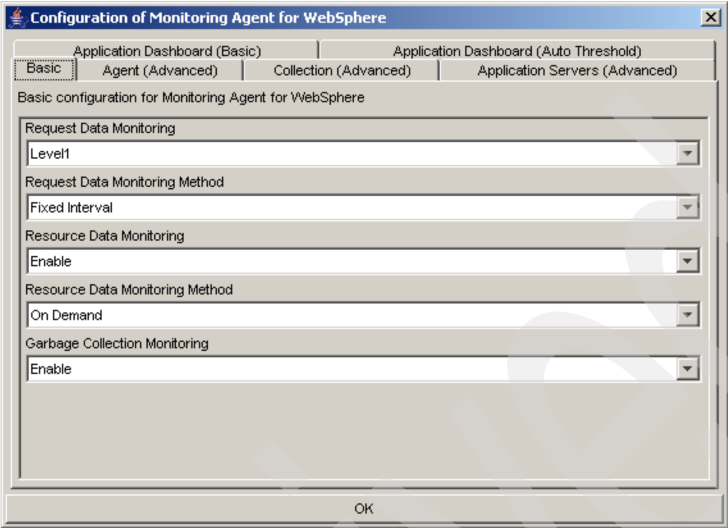


Figure 4-12 Configuration for WebSphere agent

4.4.3 Installing the data collector

The data collector runs on each monitored application server, and relays monitoring information to IBM Tivoli Monitoring agent. The data collector components from ITCAM for WebSphere must be installed on a specific level. This includes the data collector Version 6.1 using fix pack 1 and interim fix 4. The installation is performed from both the data collector installation image and the launch pad CD, on which the required fixes are distributed.

The installation is performed in the following order:

1. "Setting up the application server" on page 173 lists some prerequisite steps for the data collector installation.
2. "Installing the base data collector Version 6.1" on page 173 includes the installation of the ITCAM for J2EE V6.1 or ITCAM for WebSphere V6.1 data collector.
3. "Applying fix pack 1 and interim fix 4" on page 177 discusses the patches required for ITCAM for Web Resources V6.2.
4. In "Configuring data collectors in the application servers" on page 178 we recommend configuring the data collector after all the required patches are installed.

Setting up the application server

Before you install the data collector, you must perform these actions:

- Set up permission for accessing the application server configuration.
- Add 128 MB of heap size for the data collector. If your application server does not have the heap size defined, the default heap size is 256 MB.
- If your Windows machine runs terminal server, you must issue the command **`change user /install`** to enable installation mode.

Installing the base data collector Version 6.1

To install the data collector:

1. The data collector can be installed using the launch pad for easy access or by directly invoking the installation wizard. Launch the launchpad.cmd file, as shown in Figure 4-13.

Figure 4-13 LaunchPad Window

2. From the Install Product link, select the Install Data Collector for ITCAM for WebSphere V6.1. Clicking the **Quick Launch** column invokes the setup_DC_w32.exe from the path specified in the list. Make sure that you have the correct directory. This launches the GUI installation program for the data collector. The flow is the same as the data collector installation for ITCAM for WebSphere (see also 2.4.2, “ITCAM for WebSphere data collector” on page 38). Skip the welcome panel and accept the license information.

3. Figure 4-14 shows the installation dialogs: installation path, response file specification, and installation summary. We use the path C:\IBM\itcam\WebSphere\DC.

Figure 4-14 Installation directory

4. In the dialog for launching the configuration tools, select the option to defer the launching of the configuration tool, as shown in Figure 4-15.

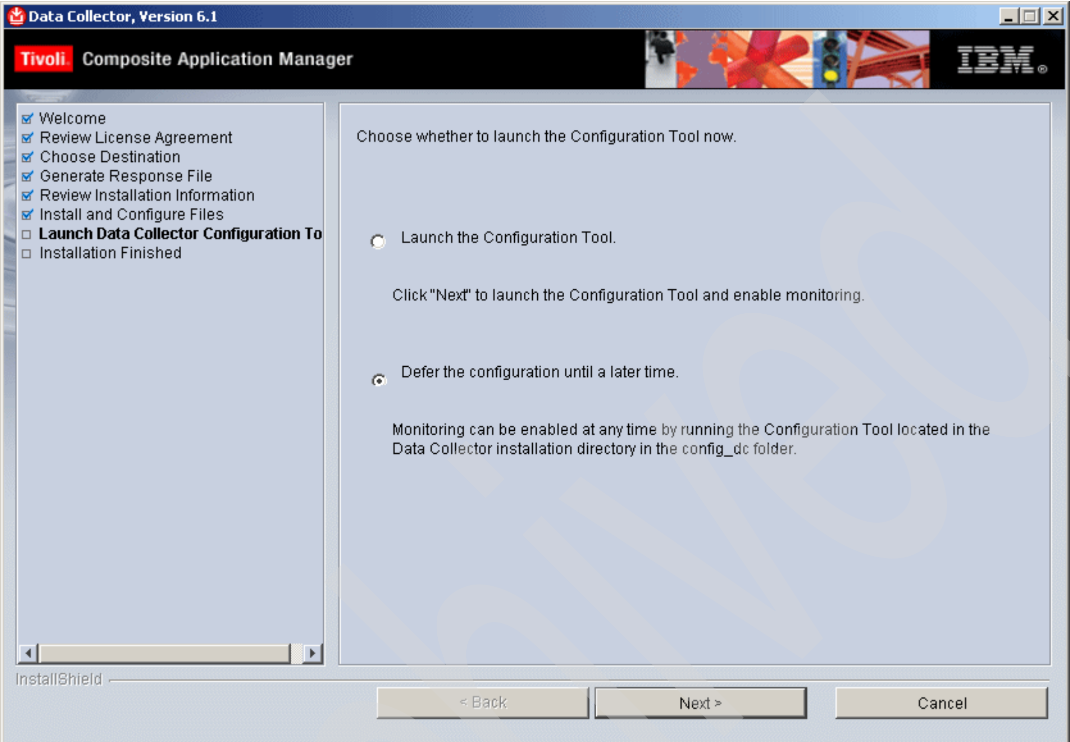


Figure 4-15 Launch data collector configuration window

5. Click **Next** to continue. A successful completion of the installation is displayed. Click **Finish**. See "Configuring data collectors in the application servers" on page 178 for the data collector configuration.

Applying fix pack 1 and interim fix 4

Perform the following steps:

1. Install the data collector fix pack and interim pack using the launch pad. The required patches are supplied in the same CD image as the launch pad. On the install product page, ensure that the correct image location is supplied for the patches. The defaults are 6.1.0-TIV-ITCAMfWAS_MP-FP0001 and 6.1.0.1-TIV-ITCAMfWAS_MP-IF0004. The directories contain the silentUpdate.bat file. If you need to change the directory, click the link and browse to the correct directory, as shown in Figure 4-16.

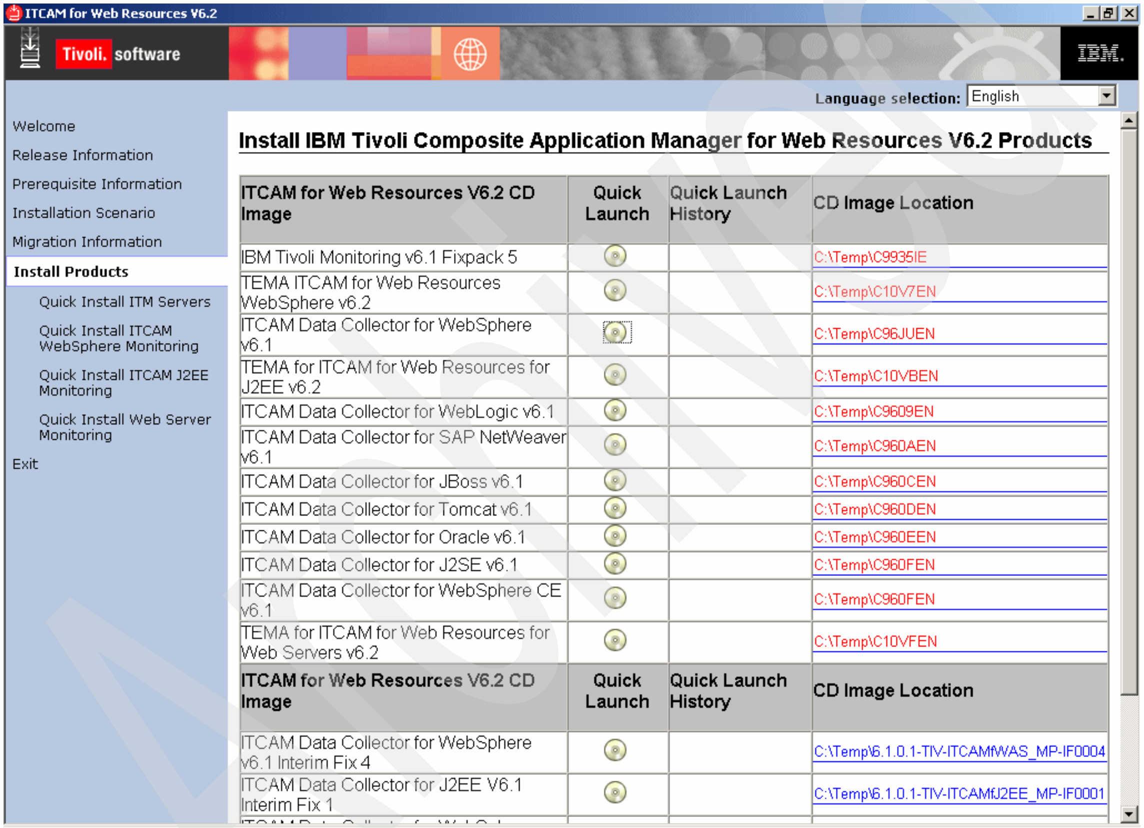


Figure 4-16 Product installation window

2. Click the icon under the column labeled Quick Launch. This launches the installation program for the fix pack for the data collector. First you have to launch the 6.1.0-TIV-ITCAMfWAS_MP-FP0001 installation.

3. Run the installation for 6.1.0.1-TIV-ITCAMfWAS_MP-IF0004.

4. If terminal services is enabled on the Windows 2000 or Windows 2003 Server, run the following command **`change user /execute`**.

Configuring data collectors in the application servers

Using the configuration tool is necessary to instrument the application server for monitoring by the data collector. It can be launched either from the installer program during the end of the installation process or by running the tool from the installation directory. Run the data collector's configuration tool for each application server instance that you want to configure.

1. Browse the installation directory for the data collector and change to the config_dc sub directory.
2. Run config_dc.bat. Click **Next** in the Welcome window for the configuration tool. Select to configure the data collector and click **Next**.
3. For ITCAM for Web Resources, only configure the data collector for displaying data on the Tivoli Enterprise Portal interface. The application management interface is provided if you use either ITCAM for WebSphere or ITCAM for J2EE. See Figure 4-17. Click **Next**.

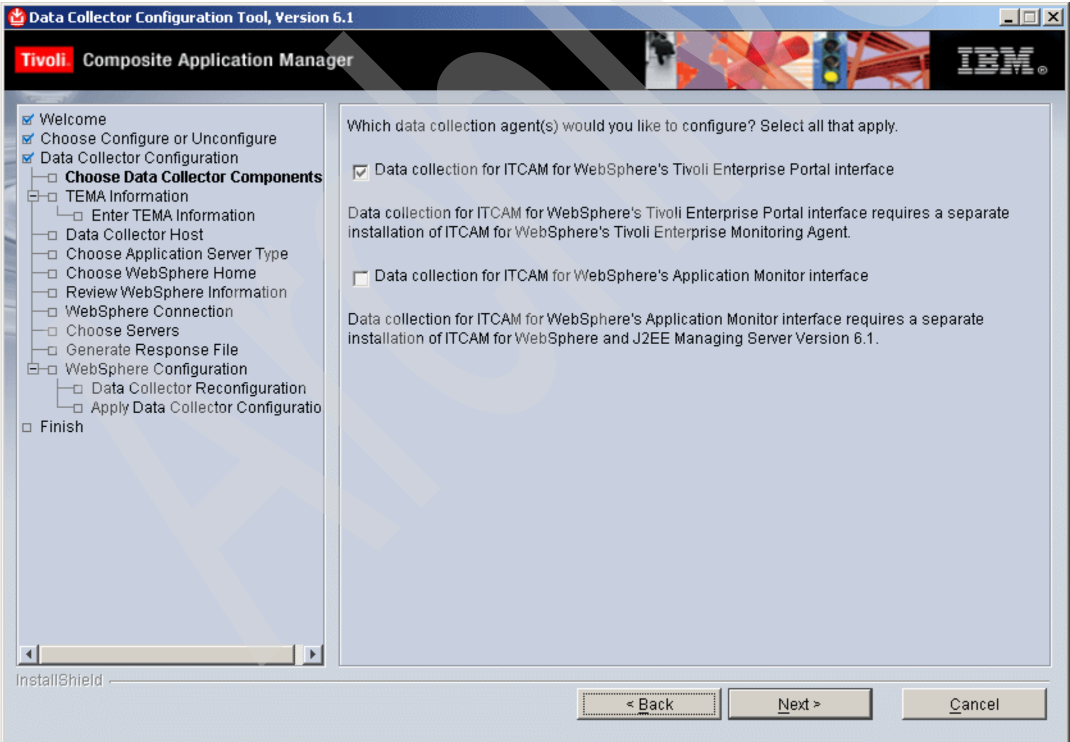


Figure 4-17 Choose data collector server type

4. In Figure 4-18, specify the host name and port for the Tivoli Enterprise Monitoring Agent. This dialog checks whether the Tivoli Enterprise Monitoring Agent is available. You can still continue installing the data collector, even if the Tivoli Enterprise Monitoring Agent is not available at this time. We use all the default values. Click **Next**.

> **Note:** If you need to change this values, the configuration is recorded in the file $DC_home/runtimes/<ver>.<node>.<server>/*.kwjdc.properties.

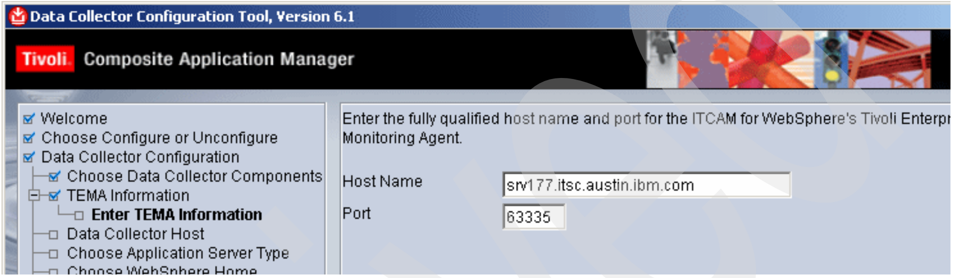


Figure 4-18 Tivoli Enterprise Monitoring Agent window

5. A window for indicating a selected IP address for the data collector when you have multiple network cards and for adjusting the ports for an enabled firewall displays, as shown in Figure 4-19. This dialog is *not* used for ITCAM for Web Resources. Just keep default information here. The communication parameter is performed by the Tivoli Enterprise Monitoring Agent to the Tivoli Enterprise Monitoring Server. Click **Next**.

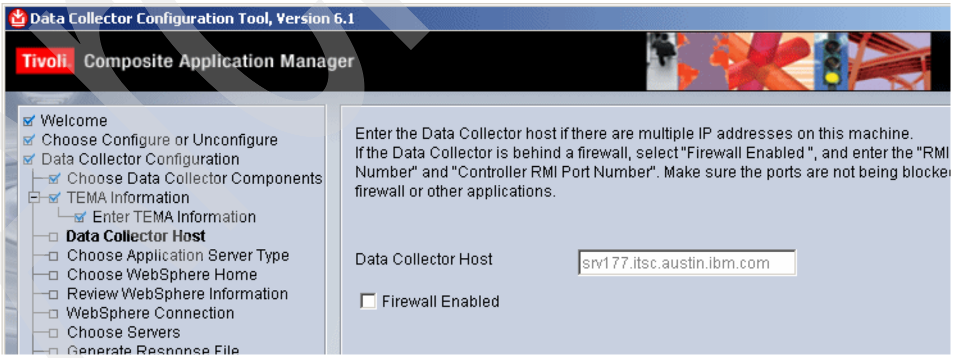


Figure 4-19 Data Collector host window

6. Figure 4-20 shows the application server selection. In our case, we install this to a standard WebSphere Application Server. Click **Next**.

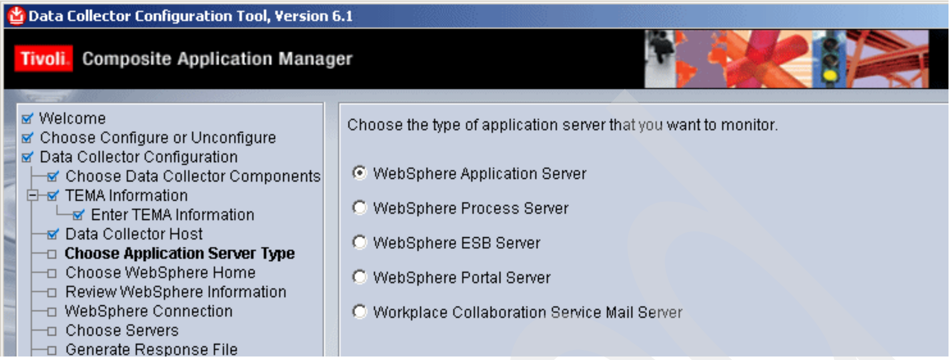


Figure 4-20 Application server type window

7. The installation wizard searches for installations of that type of application server and then displays a list, as shown in Figure 4-21. Select the application server that the data collector will monitor and click **Next**.

Note: For the WebSphere Application Server Version 6, if there are several profiles that exist for the installed application server, make sure that the profile that is selected is the profile that contains the application servers that you are instrumenting.

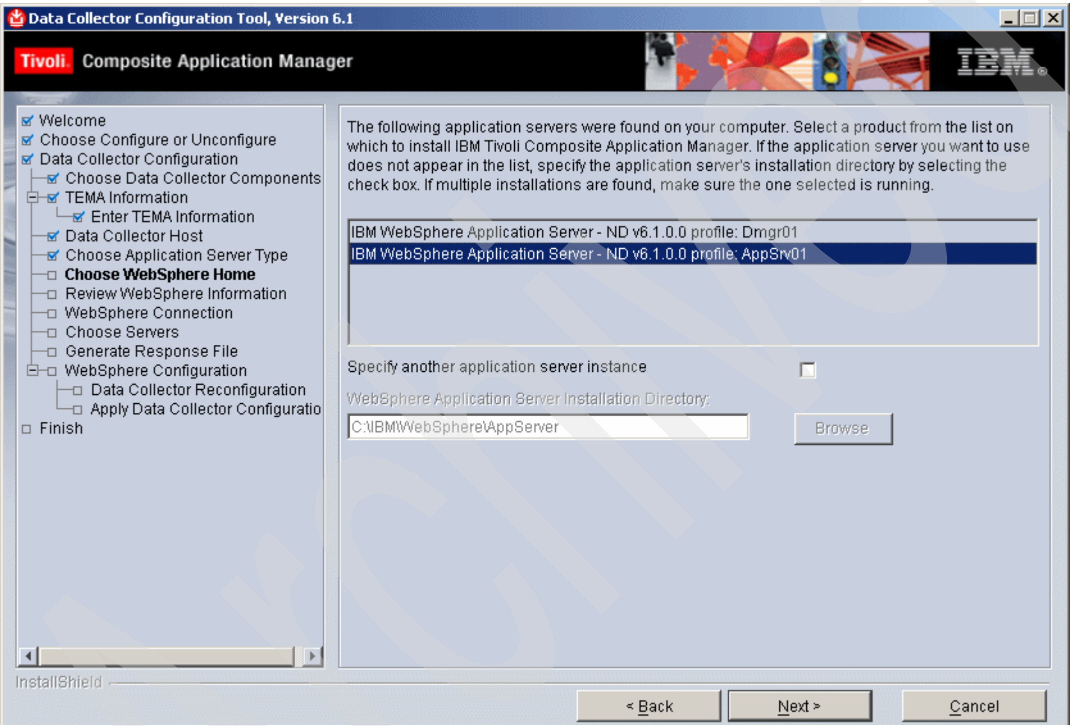


Figure 4-21 WebSphere option dialog

8. Figure 4-22 shows information about the selected application server. Confirm the information for the following by clicking **Next**.

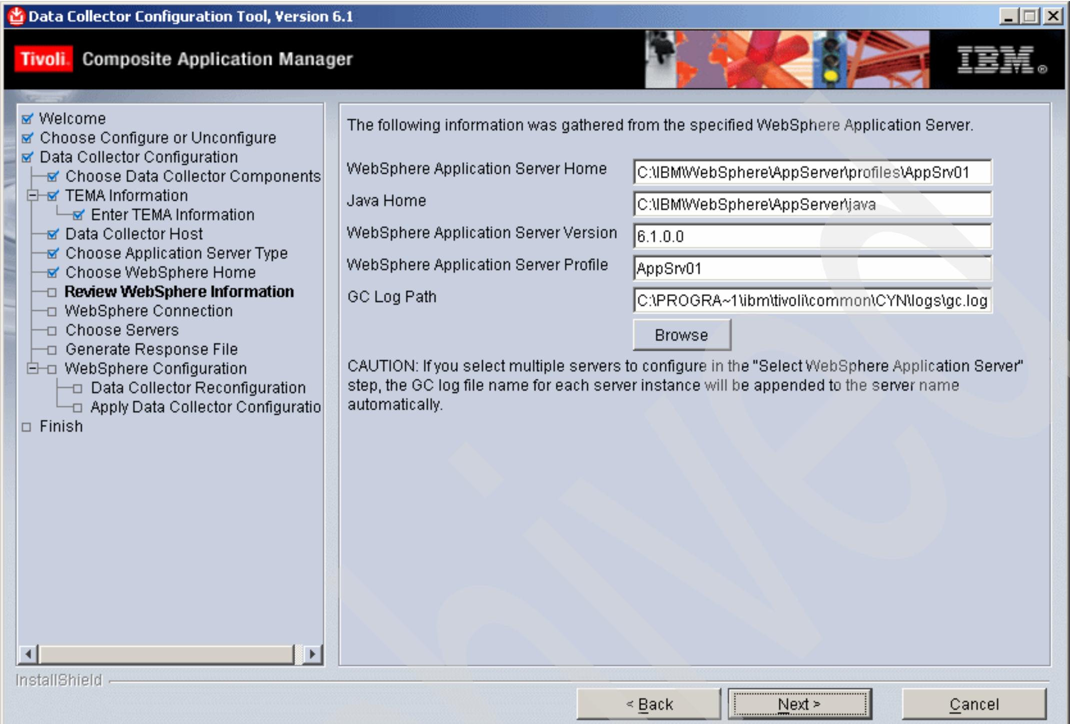


Figure 4-22 WebSphere information review window

9. The installation wizard needs to communicate to the administration application of the application server. For the network deployment installation, this connects to the deployment manager instance. In Figure 4-23, specify the correct host name and port that is used for the administration application. Click **Next**.

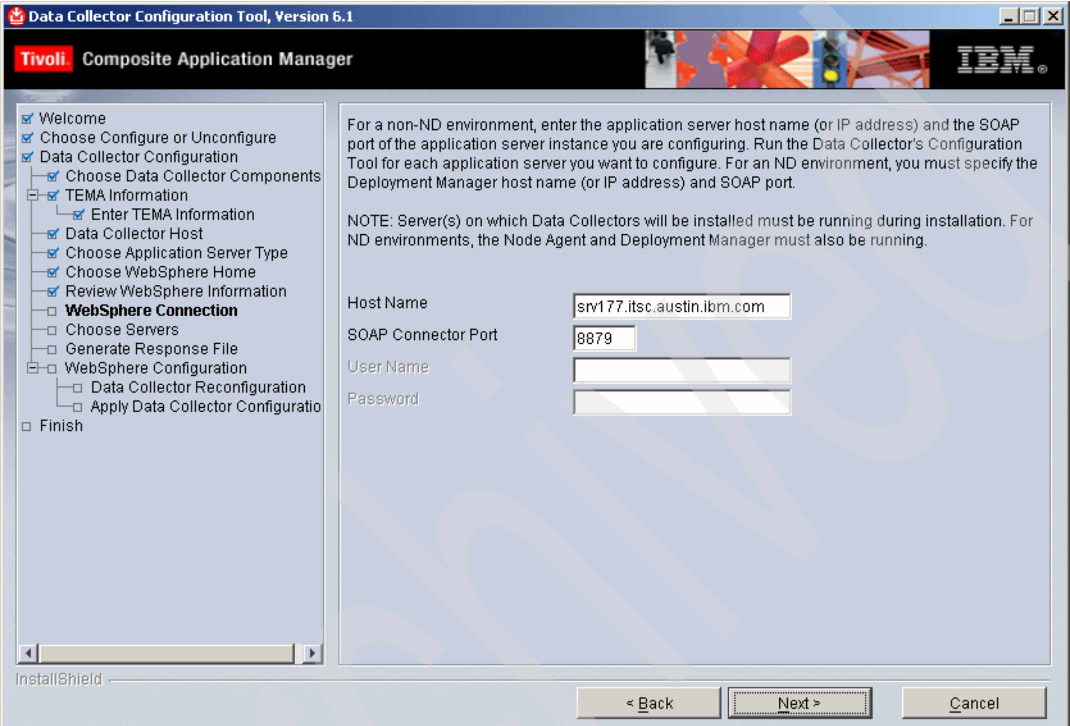


Figure 4-23 WebSphere administration application connection window

10. The installation wizard connects to the administration application and displays a tree of the application server, as shown in Figure 4-24. Select the servers and click **Next**.

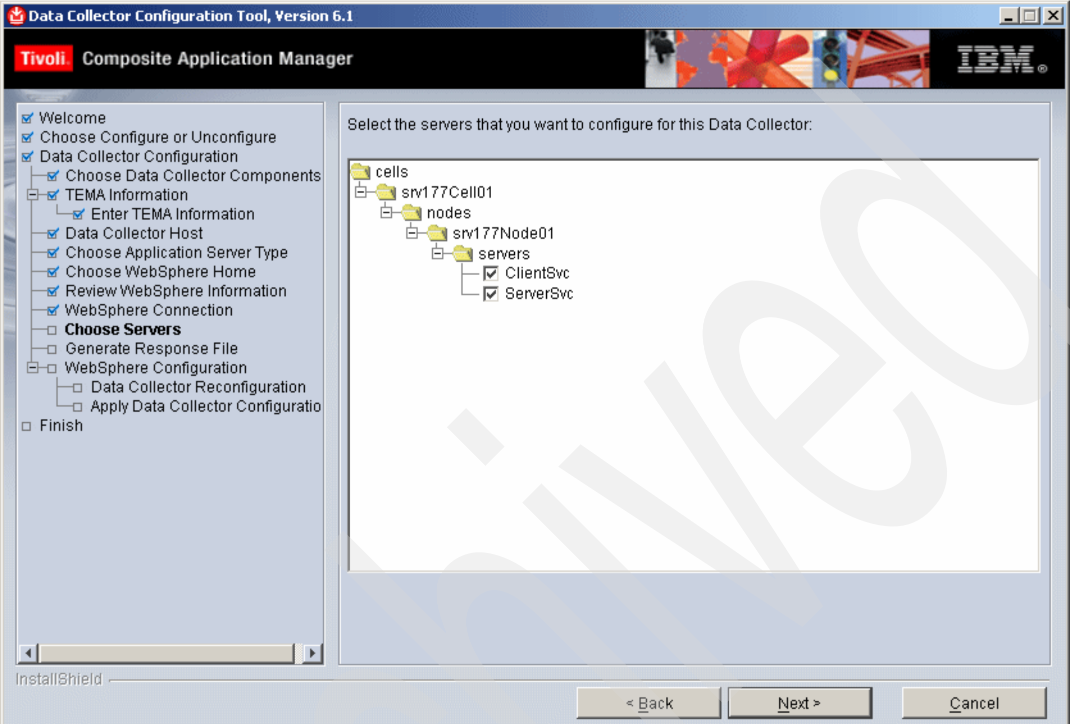


Figure 4-24 Choose server window

11. Again, you can save a response file for performing silent install for other data collectors. This is useful for deploying a large number of data collectors with minimal interaction with the machines. Click **Next**.

12. The configuration tool applies the configuration to the data collector, then a panel indicating the results of the configuration displays. Click **Finish**.

13. Restart all affected application servers.

4.4.4 Verifying the installation

Installation verification can be performed from several different aspects. Those includes:

- Verifying that the communication between the configured data collector to the Tivoli Enterprise Monitoring Agent on port 63335. Using the **netstat -all** command, you can see all active communication. See Figure 4-25. If you have Microsoft Services for UNIX (SFU) or Cygwin then you can also use the **grep** command.

```
C:\> netstat -all | grep 63335
```

Figure 4-25 Communication with Tivoli Enterprise Monitoring Agent

- Verifying that the appropriate agent and workspaces are shown in Tivoli Enterprise Portal. The sample workspace that we have for srv177 is shown in Figure 4-26.

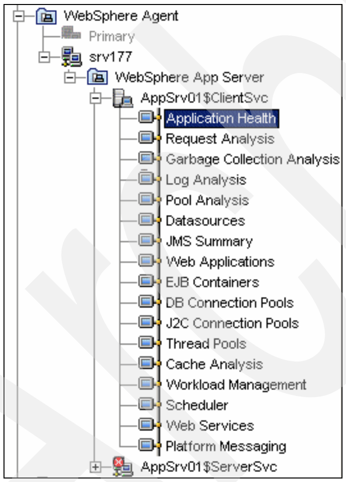


Figure 4-26 Tivoli Enterprise Portal workspace structure

4.5 Usage scenario overview

This usage of ITCAM for Web Resources is based on monitoring our trader application environment. The application runs on a set of two WebSphere Application Servers. See <Trader app> for more details on this application.

The monitoring environment is shown in Figure 4-27.

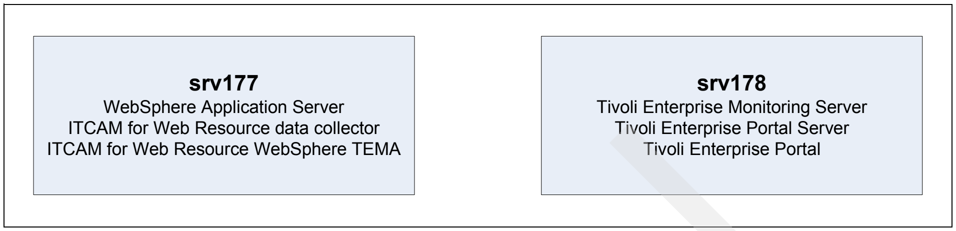


Figure 4-27 Application environment

We run the scenario on two servers:

- The monitored application server in srv177
- IBM Tivoli Monitoring server in srv178

The discussion in this section is divided into:

- Section 4.5.1, “Using the workspaces” on page 186, describes the available workspaces that come with ITCAM for Web Resources.
- Section 4.5.2, “Defining application health and baseline” on page 193, discusses defining generic application health and defining a baseline for this health indicator.

4.5.1 Using the workspaces

The workspace for ITCAM for Web Resources is created under the Tivoli Enterprise Monitoring Agent machine that the data collector is connected to, not the machine on which the data collector resides. It is a good practice to always have the Tivoli Enterprise Monitoring Agent running on the same machine as the data collector.

The workspace is structured by agent host and then each application server node. Individual workspaces aimed to collect certain performance information are available under the application server node. See Figure 4-28.

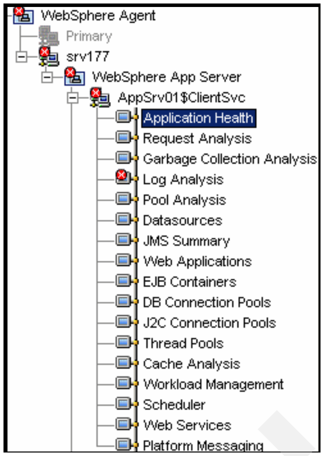


Figure 4-28 Workspace structure

Let us evaluate some of these workspaces:

- The primary agent workspace

  This is the workspace that contains all situation events from the agent and the health summary of the application servers that it monitors. Figure 4-29 shows an example of this workspace.

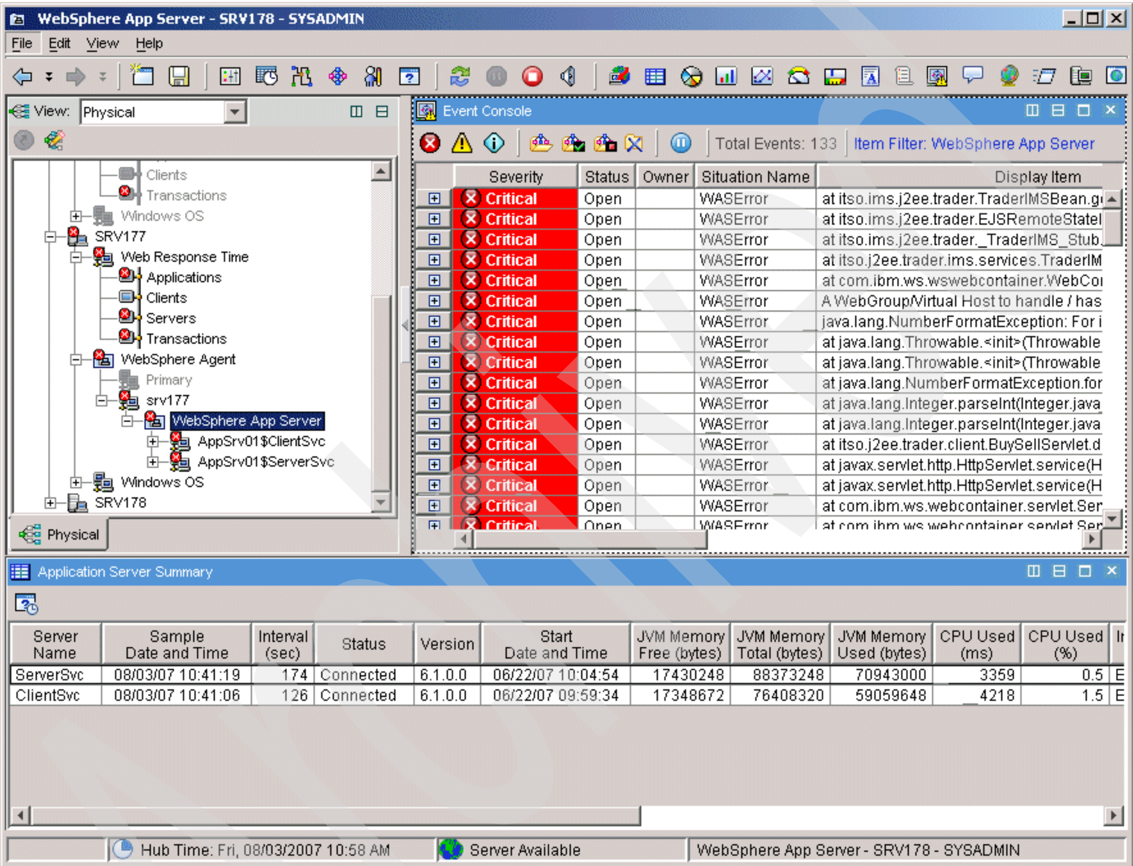


Figure 4-29 Agent summary workspace

- The application server summary workspace provides an at-a-glance summary of the important metrics of the application server. This workspace is shown in Figure 4-30.

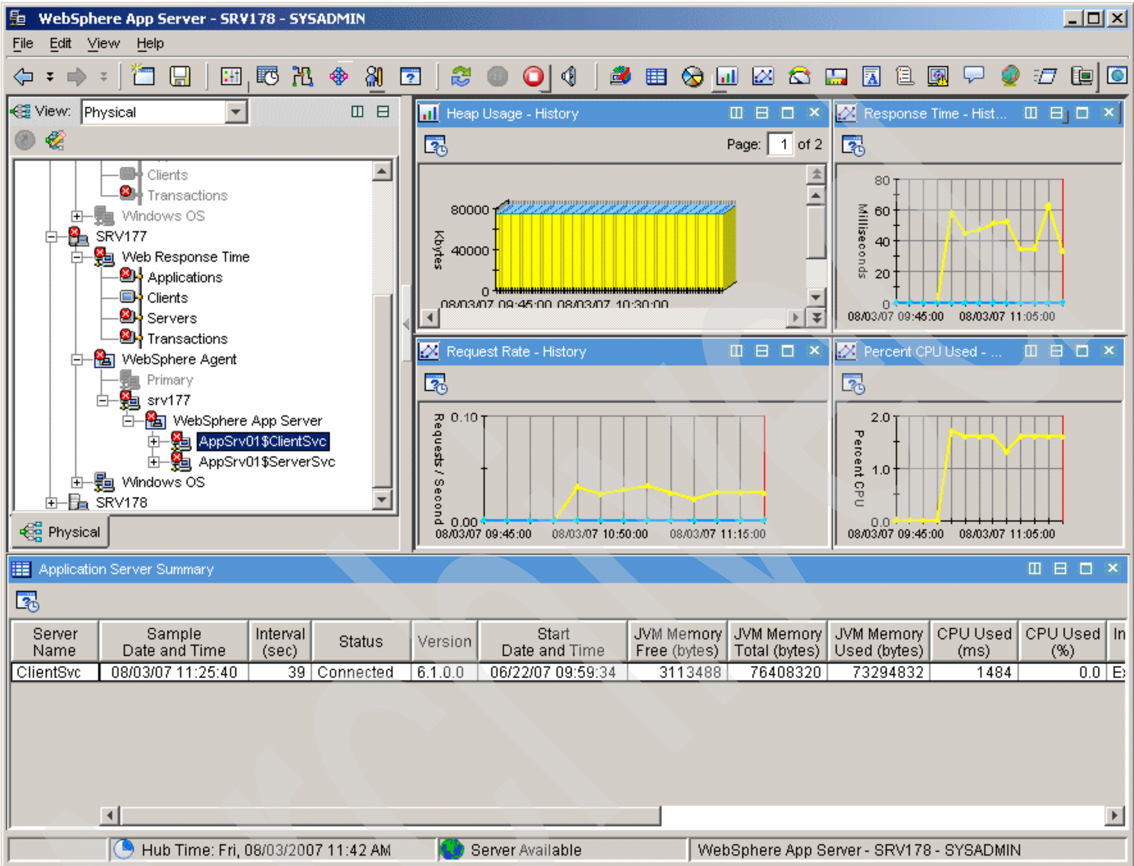


Figure 4-30 Application server summary workspace

- Under the application servers, there is a set of workspaces. The workspaces are:
  - Application health
  - Request analysis
  - Garbage collection analysis
  - Log analysis
  - Pool analysis
  - Datasource
  - JMS summary
  - Web application
  - EJB containers

– DB connection pool
– J2C connection pool
– Thread pool
– Cache analysis
– Workload management
– Scheduler
– Web services
– Platform messaging

We do not cover all of those workspaces. The following are some examples of the workspaces:

– The request analysis workspace is shown in Figure 4-31.

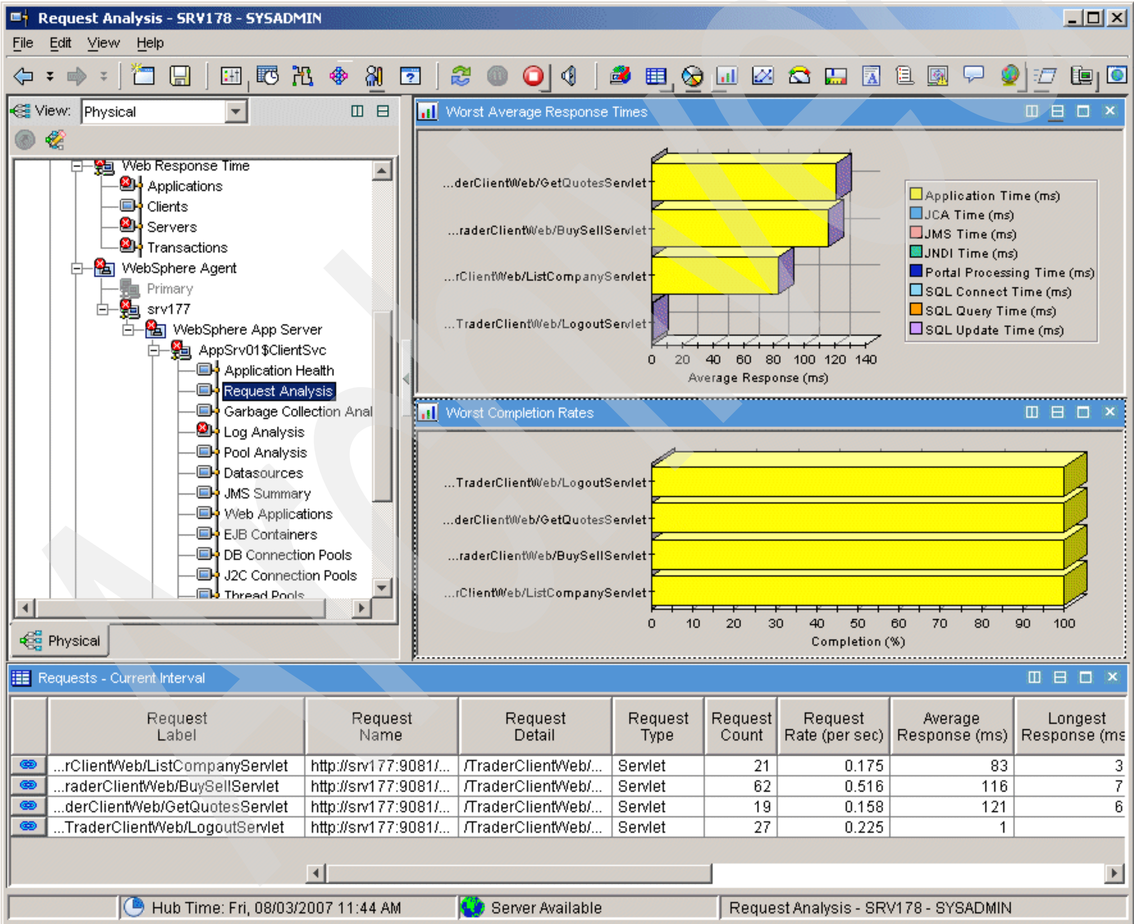


Figure 4-31 Request analysis workspace

– The log analysis summarizes WebSphere errors and exceptions, and ITCAM for Web Resources data collector messages in the WebSphere log. Figure 4-32 shows the workspace.

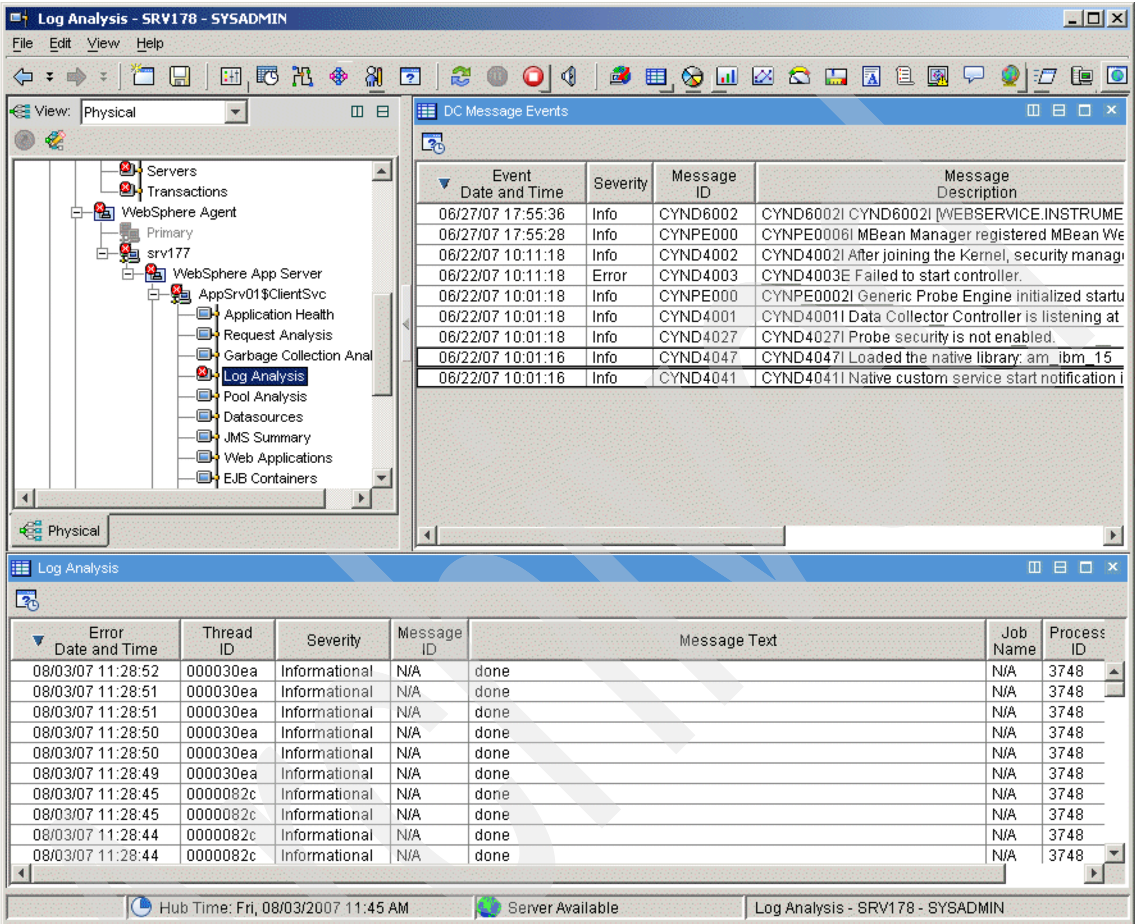


Figure 4-32 Log analysis workspace

– The Web application workspace that lists the available Web application and its performance is shown in Figure 4-33.

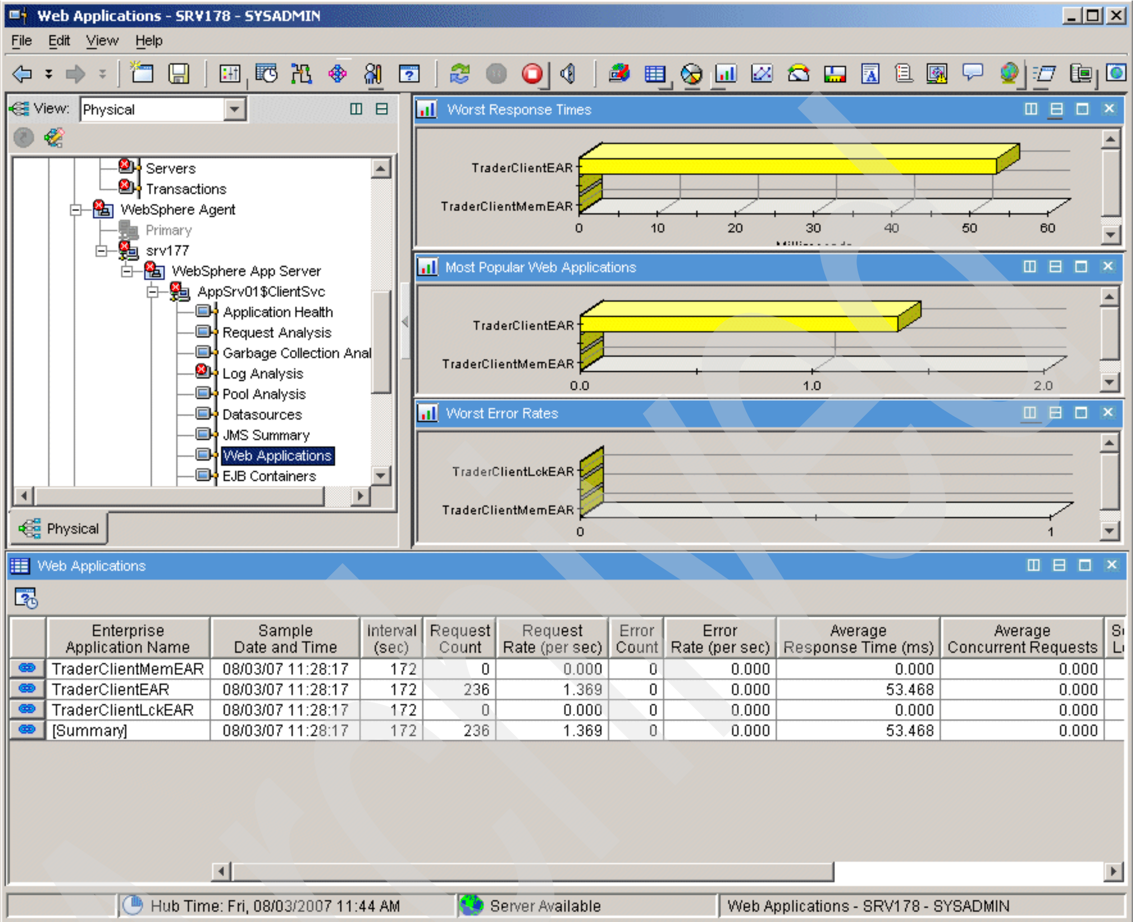
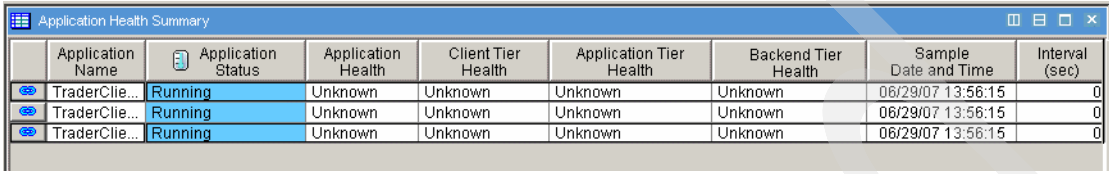


Figure 4-33 Web application workspace

4.5.2 Defining application health and baseline

ITCAM for Web Resources has a unique feature that provides an application health workspace. The application is represented by the installed enterprise application (EAR) file. Initially, all application health in the application health workspace is unknown, as shown in Figure 4-34.


Application Health Summary

	Application Name	Application Status	Application Health	Client Tier Health	Application Tier Health	Backend Tier Health	Sample Date and Time	Interval (sec)
	TraderClie...	Running	Unknown	Unknown	Unknown	Unknown	06/29/07 13:56:15	0
	TraderClie...	Running	Unknown	Unknown	Unknown	Unknown	06/29/07 13:56:15	0
	TraderClie...	Running	Unknown	Unknown	Unknown	Unknown	06/29/07 13:56:15	0

Figure 4-34 Tivoli Enterprise Portal window

The health is calculated against internally stored baseline thresholds. This baseline process uses mathematical and statistical methods to automatically calculate thresholds based on response time information that is collected over a period of time. The baseline data collection can be a long-running activity that can last for hours or even days.

> **Note:** The recommended period to create a baseline is when the application is running under a typical load. The performance of the application under this load should be considered as good. Deviation from this performance would indicate degrading health of the application.

To create an application baseline:

1. Open Tivoli Enterprise Portal and select the application server that you want to work on. Open the application health workspace.

2. From the application health summary table, select one application name and right-click the choose link, as shown Figure 4-35.

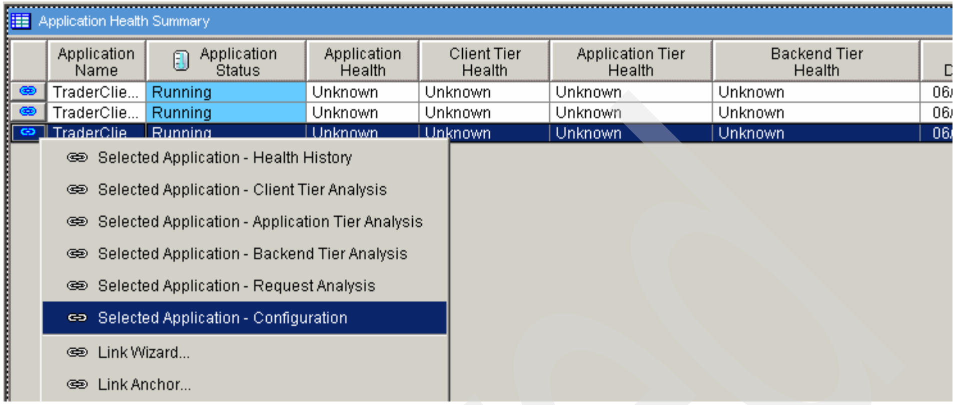


Figure 4-35 Application Health Summary

3. Select **Selected Application - Configuration** and the Application Request Configuration window opens, as shown in Figure 4-36.

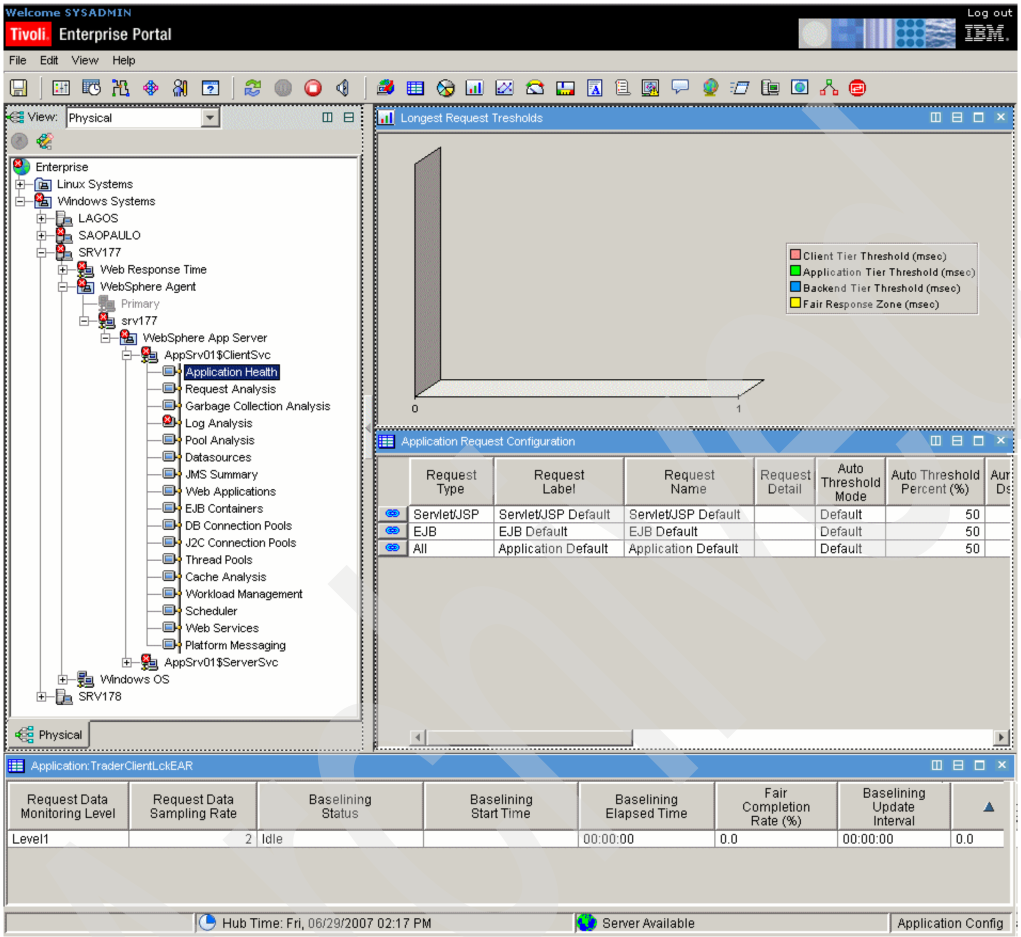


Figure 4-36 Application request configuration

4. In the application request configuration workspace, select the request type that you want to define the baseline on and right-click and select **Take Action** → **Select**, as shown in Figure 4-37.

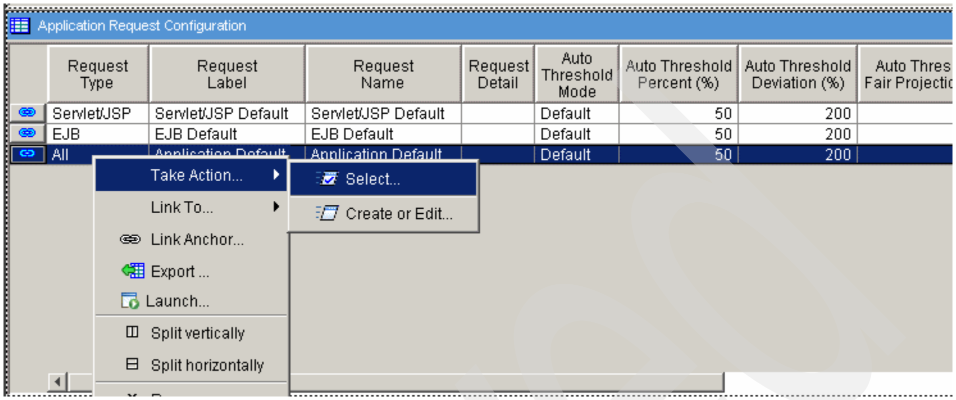


Figure 4-37 Invoking an action

5. In the Take Action dialog (Figure 4-38), select the **Start_Baselining** action.

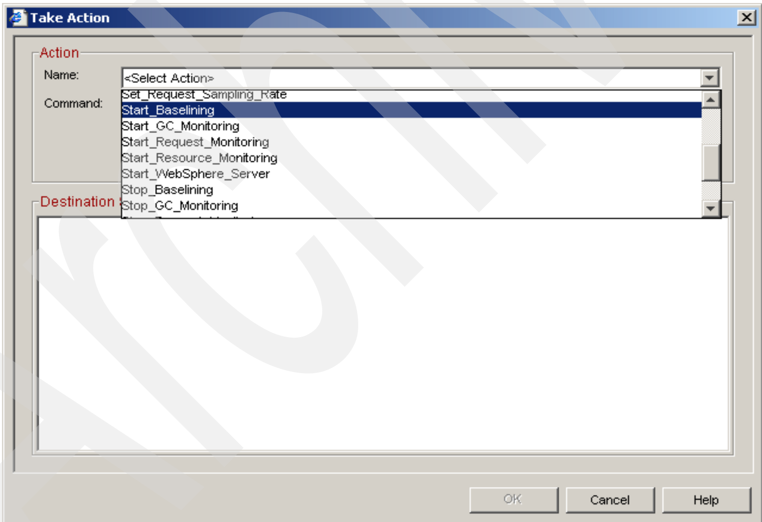


Figure 4-38 Take Action selection

6. Enter the arguments, as shown in Figure 4-39. Some of the values have been prefilled from the application that you select. Click **OK**.

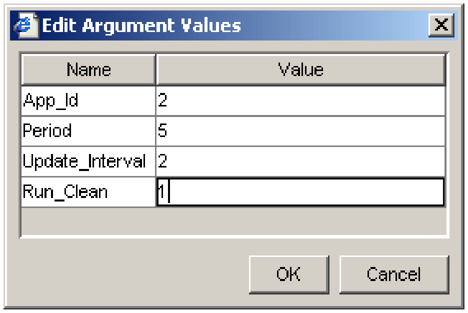


Figure 4-39 Edit argument values

7. Back in the Take Action window, click **OK** to invoke the action. The arguments and destination system have been preselected based on your invocation context.

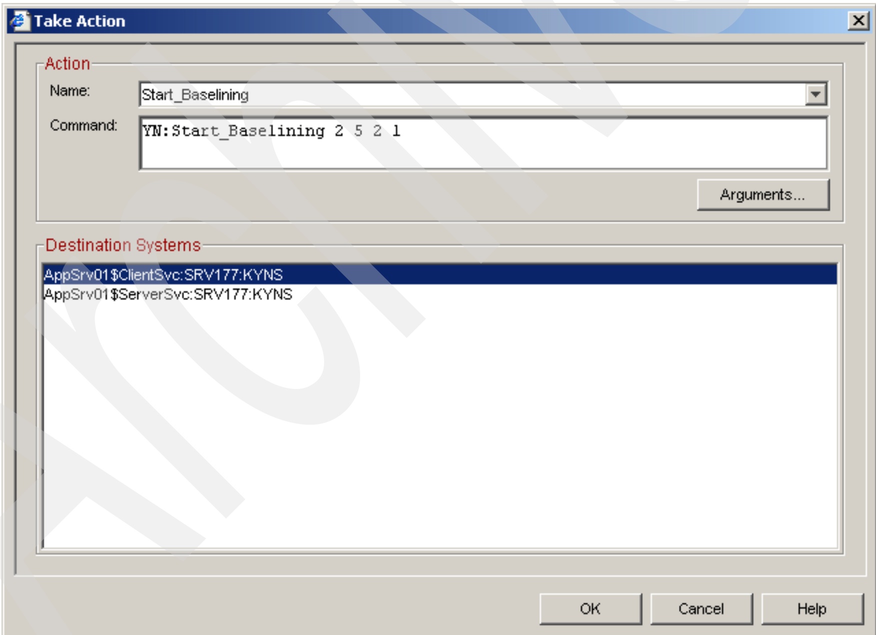


Figure 4-40 Take Action

8. When the command has been invoked successfully, the message in Figure 4-41 is shown. Click **OK**.

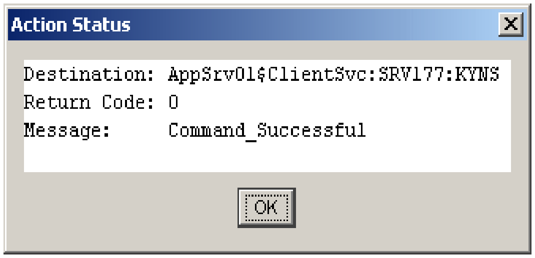


Figure 4-41 Action status command

9. Perform some transactions on the application and then you can invoke the Stop_Baselining action using the same mechanism.
10. You can start getting the result on the application health workspace.

5

ITCAM for Response Time Tracking concepts, installation, and implementation

This chapter describes the implementation of IBM Tivoli Composite Application Manager for Response Time Tracking in our environment. It includes:

5.1 Product architecture

ITCAM for Response Time Tracking Version 6.1 is an evolution from IBM Tivoli Monitoring for Transaction Performance V5.3. It inherits the major components and functions of IBM Tivoli Monitoring for Transaction Performance V5.3.

Figure 5-1 shows the ITCAM for Response Time Tracking components.

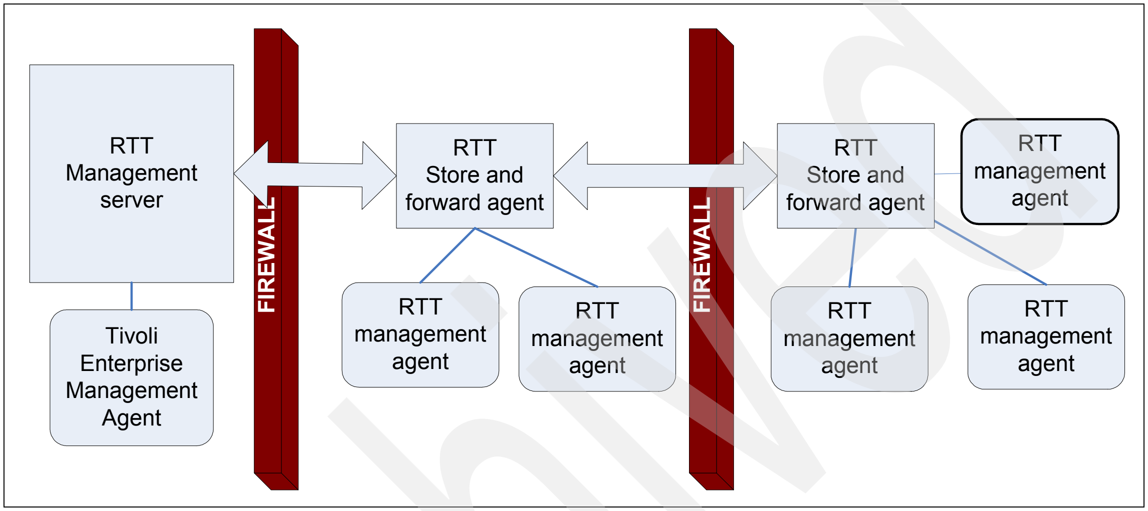


Figure 5-1 ITCAM for Response Time Tracking components

Basically, ITCAM for Response Time Tracking is controlled from the management server. The management server provides a centralized repository of monitors, configuration, and data for the ITCAM for Response Time Tracking environment.

The rest of ITCAM for Response Time Tracking consists of the management agents. The management agents perform performance and response-time data collection on behalf of the management server. The agent can perform response time collection from an application server or perform robotic simulation of a transaction for measuring response time. The management agent functions as a single agent that can have different monitoring components deployed on them to perform different functions.

The management server and management agent typically operate in an unrestricted port environment. When there is a firewall between them, they restrict the port usage to communicate. The firewall requirement typically requests that they use a single communication port to talk back and forth. This is where the store and forward agent comes in. It bundles the management communication between the management server and management agent to use a single port to pass through the firewall. The store and forward agent can be cascaded, so in this sense, there can be a chain of store and forward agents to pass through multiple layers of firewalls.

A special management agent resides on z/OS machines. The management agent on z/OS machines has the transaction server component activated to receive performance information from the CICS and IMS data collector.

In ITCAM for Response Time Tracking V6.1, information from the management server can be forwarded to the Tivoli Enterprise Monitoring Server for display in the Tivoli Enterprise Portal. To do this, use the Tivoli Enterprise Monitoring Agent for ITCAM for Response Time Tracking.

We discuss the components of ITCAM for Response Time Tracking in the following sections:

- 5.1.1, “The management server” on page 201
- 5.1.2, “Store and forward agent” on page 203
- 5.1.3, “Management agents” on page 204
- 5.1.4, “Tivoli Enterprise Monitoring Agent” on page 207
- 5.1.5, “ARM concepts” on page 208

5.1.1 The management server

The ITCAM for Response Time Tracking management server consists of a J2EE enterprise application that accesses a DB2 repository using JDBC. The management server runs on a WebSphere Application Server. The application server can be installed on a stand-alone WebSphere Application Server or on a clustered environment. Figure 5-2 shows a stand-alone management server.

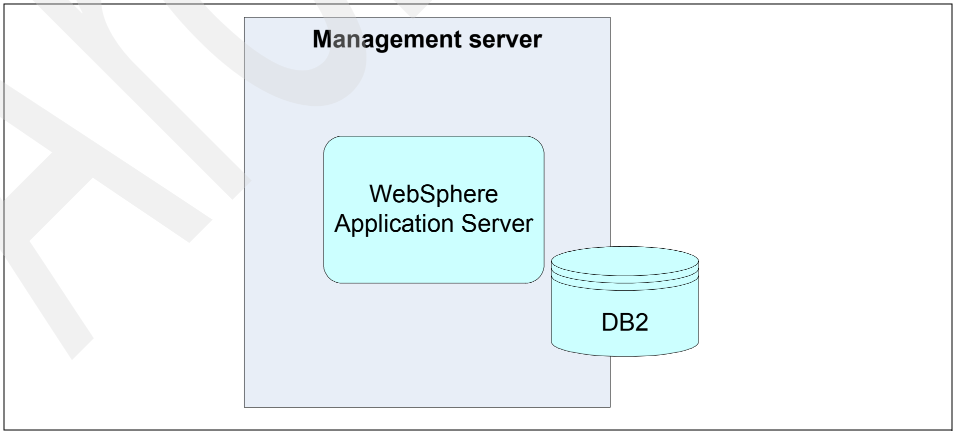


Figure 5-2 Stand-alone management server

Figure 5-3 shows the clustered management server. It consists of the WebSphere Edge Server for load balancing, the management server on several clustered WebSphere Application Server Node Deployment systems, and the database installed on a separate database server.

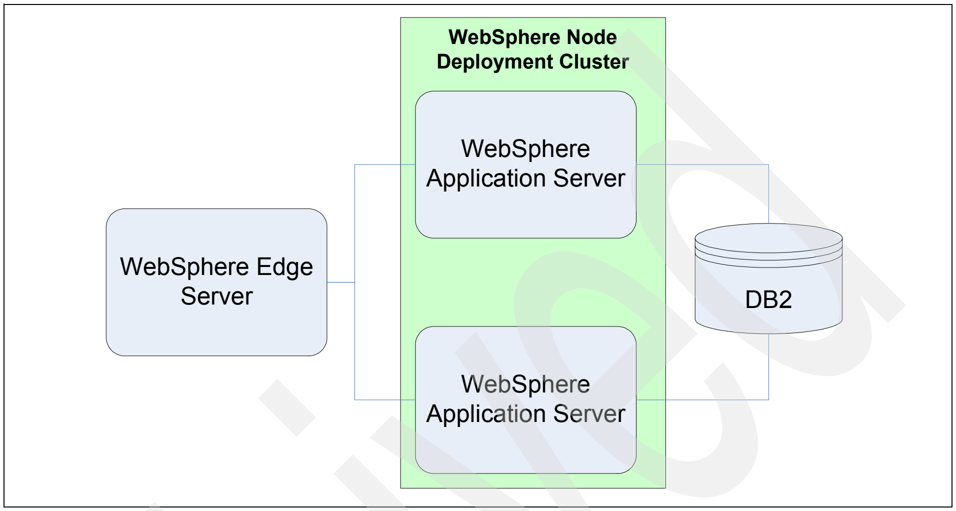


Figure 5-3 Clustered management server

Clustered management server benefits include:

- Separating the servers reduces the processing burden of a single machine.
- It allows failover for failure in WebSphere Application Server, so the other management server in the cluster can take over the work.

Some disadvantages are:

- Additional communication traffic between machines.
- More difficult setup. Refer to *IBM Tivoli Composite Application Manager for Response Time Tracking V6.0: Installing a Management Server in a WebSphere Cluster Environment*, SC32-1804, for installation instructions.

Overall, regardless of the management server types, the management server provides the following functions:

- Managing management agents and their deployed components. Management agents must sign in to the management server and retrieve all required monitors when it is started initially.
- Storing monitors for management agent operation, including discovery, listening monitors, and robotic monitors. These monitors are maintained using the Web interface or the new command-line interface.

- Managing a schedule repository. Note that the management agent performs the schedule application.
- Performing data collection from various management agents and storing them in its repository.
- Maintaining users and roles for accessing the Web interface.
- Serving the Web interface.

5.1.2 Store and forward agent

The store and forward agent acts as an intermediary between the management server and the management agent. Figure 5-4 shows its overall processing.

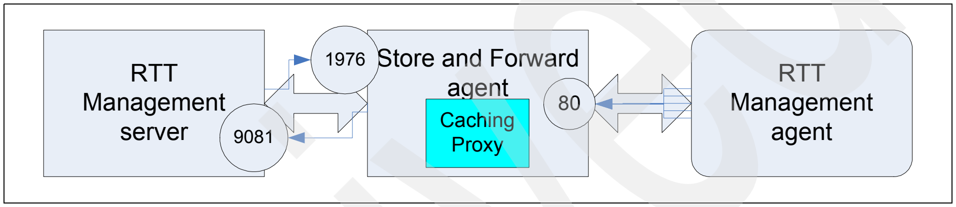


Figure 5-4 Store and forward agent

This agent consolidates communication from and to management agents and uses a single port to communicate to the upstream component. The store and forward agent can be cascaded. It uses IBM WebSphere Caching Proxy, which is part of WebSphere Edge Server V2.0. The caching proxy optimizes the connection with the management server.

The default port, to which the management agent must connect, is 9446 for SSL or 80 for non-SSL.

You can have multiple store and forward agents chained to get to the management server through multiple layers of firewalls. Figure 5-5 shows this concept.

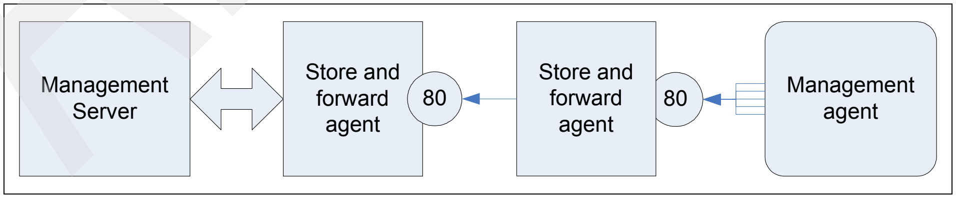


Figure 5-5 Multiple store and forward agents

5.1.3 Management agents

The management agent runs in a Java virtual machine on the managed server. It typically performs the following functions:

- Starting and stopping the management components
- Collecting monitoring and schedule information from the management server
- Informing the management components about what to perform
- Caching response time data in the temporary directory
- Uploading response time data as requested by the management server, either at regular collection time or on demand

The management agent behavior is based on the Application Response Measurement (ARM) specification. For discussion about ARM, refer to 5.1.5, "ARM concepts" on page 208. The management component monitors measure response time and report the times using the ARM specification to the ARM agent process (tapmagent executable). The ARM agent process stores response time information on physical disk. The management agent uploads the response time information to the management server at a regular interval.

There is a slight difference between the distributed management agent architecture and the z/OS-based management agent.

Distributed management agent

Conceptually, the processing of the management agent can be illustrated as shown in Figure 5-6.

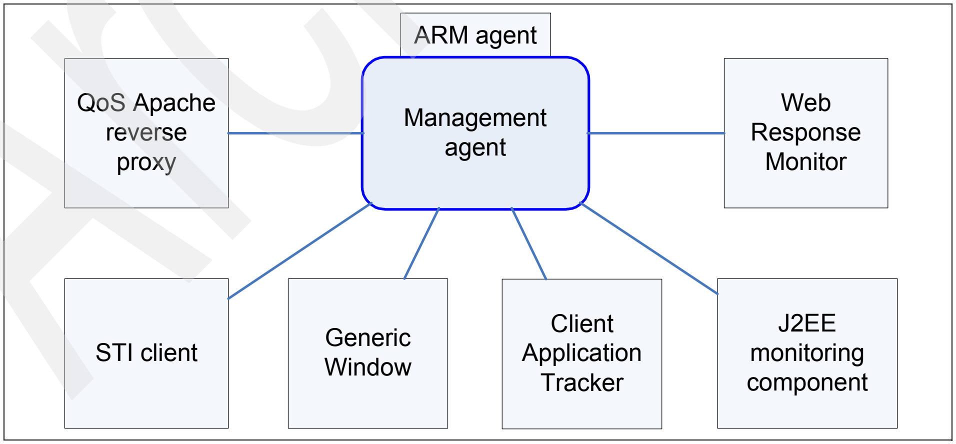


Figure 5-6 Management agent

The existing management components are:

- The Generic Window component allows investigation of a Windows-based application to use with the Rational Robot application. This enables the Robot to interact with a native Windows application, a Java application, or a browser-based application. This can be deployed only on a Windows system.
- The Synthetic Transaction Investigator (STI) simulates user interaction with a Web browser. STI transactions are recorded in advance using the STI recorder application. The STI component can be deployed only on a Windows system.
- Quality of Service (QoS) runs in an Apache Web server proxy that tracks the response time for a user. It inserts a small Java script for HTML code to reply back to QoS and indicates the overall user response time.
- The J2EE instrumentation component runs as JVMPI instrumentation for a J2EE-compliant application server such as WebSphere Application Server or BEA WebLogic. It monitors certain WebSphere classes to collect information about servlets and Web services calls. It also collects response-time information for JDBC connections and J2C accesses.
- ARM agent: The ARM agent, historically called Tivoli Application Performance Manager, is implemented as the executable tapmagent.
- The Web Response Monitor (WRM) measures the performance of Web-based applications, provides response time and other performance data, and tracks navigation paths and usage behavior.
- Rational Performance Tester: This uses the Rational Performance Tester Web test interface to collect response time information.
- Generic Playback calculates the response time that a command runs by the management agent.
- JBoss and Tomcat components measure J2EE performance for JBoss and Tomcat servers.
- The Mercury LoadRunner runs Mercury LoadRunner scripts and collects their response time.

Mainframe management agent

For z/OS-based systems, ITCAM for CICS Transactions and ITCAM for IMS Transactions data collectors can send transaction start and transaction end events to the management agent running on the same z/OS system as the CICS and IMS started task. These start and stop events for CICS and IMS transactions are translated into ARM start and ARM stop calls by a component called the transaction server. This allows IMS and CICS transactions to be shown as part of the distributed transaction or as a stand-alone transaction running on z/OS. Figure 5-7 shows the z/OS-based management agent structure.

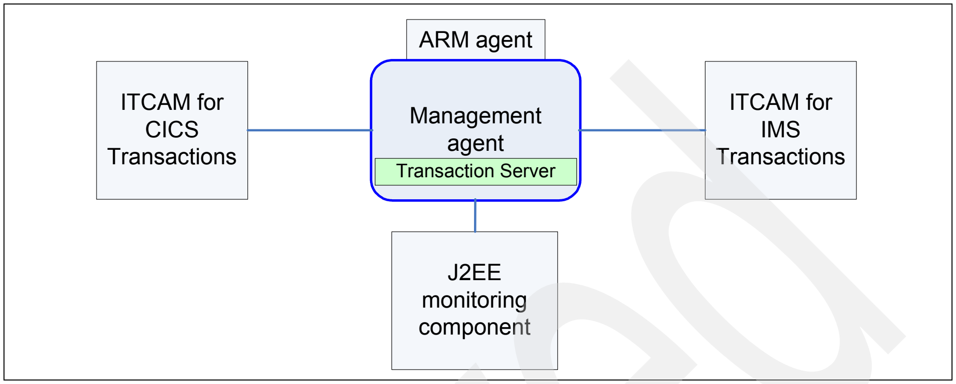


Figure 5-7 Mainframe management agent

> **Note:** The transaction server component is activated by default in all platforms that we installed. Only the z/OS management agent can use this feature. You can see which components are started from the configuration file tmtp_sc.xml.

The components of the management agent in z/OS are:

Management agent The management agent is responsible for communication with the ITCAM for Response Time Tracking management server and collecting ARM-related activities on behalf of CICS and IMS. The ARM engine or tapmagent is part of the management agent — it enables you to integrate any other ARM-instrumented application with ITCAM for Response Time Tracking.

CICS data collector The CICS data collector monitors transaction response times within CICS regions. It gives you information about how long it took to run the transaction in the monitored CICS region. If your CICS transaction flows through several CICS regions, installing CICS data collectors in every region involved enables you to track the complete transaction flow.

IMS data collector The IMS data collector monitors transaction response times within IMS regions. This means that you get information about how long it took to execute the single transaction in the monitored IMS region. If your IMS transaction spans multiple IMS regions, you can monitor the complete transaction flow by installing the IMS data collectors in the regions that are involved.

J2EE component Monitors the J2EE transaction in the WebSphere Application Server on z/OS.

The IMS data collectors and CICS data collectors report to the transaction server portion of the ITCAM for Response Time Tracking management agent on the z/OS machine on which they execute.

5.1.4 Tivoli Enterprise Monitoring Agent

The agent for the Tivoli Enterprise Monitoring Server for ITCAM for Response Time Tracking is provided as a separate installable feature. Figure 5-8 shows the connectivity of the Tivoli Enterprise Monitoring Agent for ITCAM for Response Time Tracking.

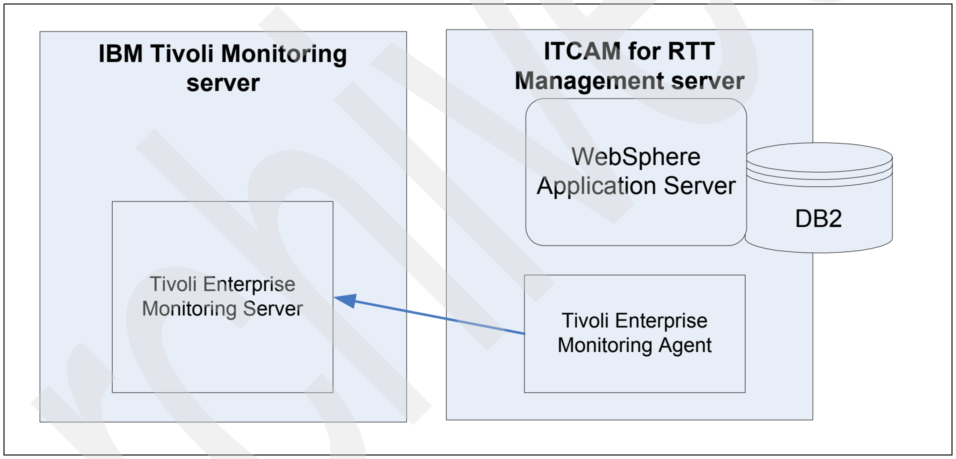


Figure 5-8 Tivoli Enterprise Monitoring Agent

The ITCAM for Response Time Tracking agent is assigned the code KT2. The Tivoli Enterprise Monitoring Agent components must be installed in the Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, and Tivoli Enterprise Portal desktop client.

ITCAM for Response Time Tracking provides a workspace that is dynamically linked based on the reporting groups that are available on the management server. See 5.4.5, “Integrating with Tivoli Enterprise Monitoring Server” on page 241, for the implementation of this Tivoli Enterprise Monitoring Agent and 6.10, “Working with Tivoli Enterprise Portal” on page 326, for use of the Tivoli Enterprise Portal with ITCAM for Response Time Tracking. Data can also be stored in the Tivoli Data Warehouse for historical analysis.

5.1.5 ARM concepts

Application Response Measurement (ARM) is an Open Group standard for measuring the response time of a custom transaction. For the current specification for ARM, see:

http://www.opengroup.org/arm

The ARM API provides a means to indicate the start and stop of a transaction that can then be measured to show response time in a distributed system. The current specification allows correlation between multiple components of the application to show response time components.

ARM can be used to instrument a simple application, as shown in Figure 5-10 on page 209. This includes using the ARM API as a stopwatch to measure response time.

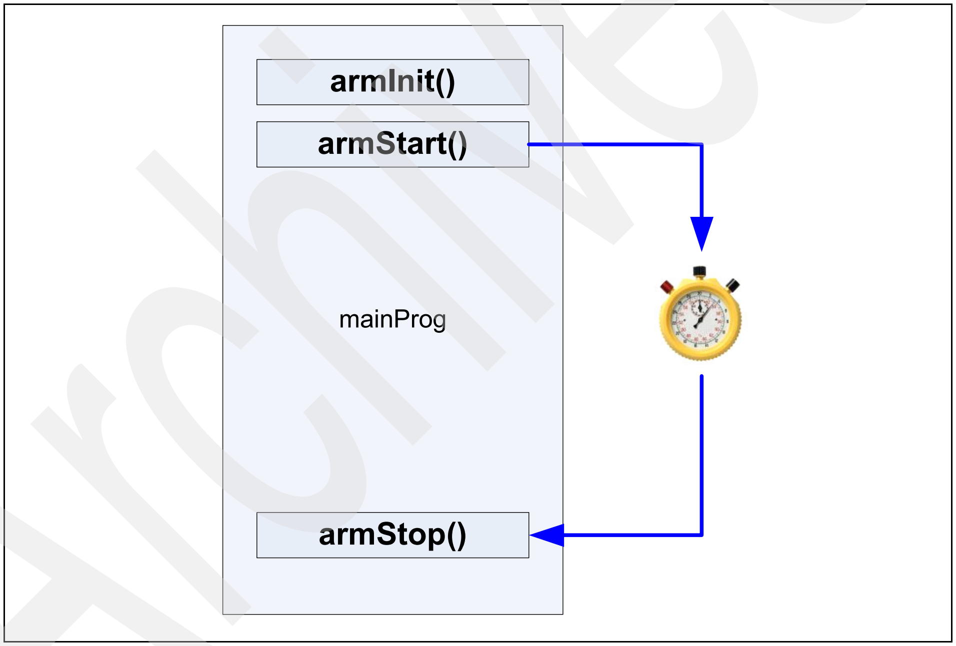


Figure 5-9 ARM concepts: simple application

Figure 5-10 shows a more complex ARM calling sequence in which a distributed application calls across several modules. Each module provides its own instrumentation to call ARM APIs.

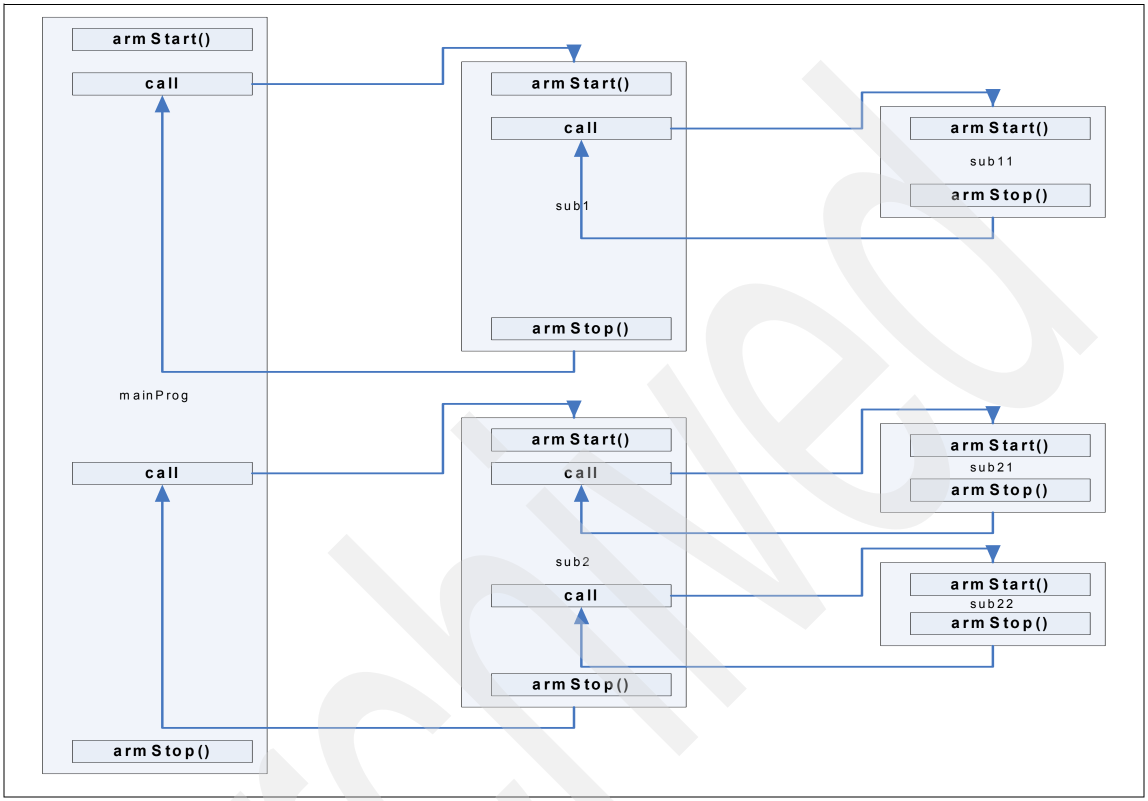


Figure 5-10 ARM concepts: complex application

Distributed application programs can be viewed as multiple components of programs that are called independently. The overall response time of the application is broken down into response time components that might not reside on a single machine, thus complicating performance analysis. The objective of ARM is to enable response time tracking and analysis for these components and identify them in a big picture.

In Figure 5-10, ARM acts in its basic function as a digital stopwatch. It records the start and end of transactions using the armStart() and armStop() calls. As the program makes a call to its subprogram, which can use Web services, Remote Procedure Call, or any other means, it also passes a correlator, which enables the ARM server to identify the relationship of the response time components. The same correlator is passed from the calling program to all of its subprograms. The calling sequence is of no importance to ARM.

To use ARM, the application program must:

1. Initialize the ARM environment.
2. (Optional) Collect the ARM correlator.
3. Indicate the start of the transaction.
4. (Optional) Pass the ARM correlator to the subtransaction.
5. Indicate the end of the transaction.
6. Clean up ARM resources.

The ARM start and stop API calls send a small packet of data to the ARM agent that is specified in the initialization logic. This data is then interpreted and response time is calculated. The correlator shows transaction hierarchy so that response time components can be analyzed.

The ARM API provides an implementation specification for both C language and Java-based programs. In this book, we discuss only the Java implementation of the ARM API using ITCAM for Response Time Tracking. Instrumenting applications with ARM calls enables ITCAM for Response Time Tracking to display custom application topology other than the default instrumentation structure. This enables the application programmer to show better structuring of the application response time.

The Java implementation of the ARM API is defined in a set of interfaces. These Java interfaces are from the software Open Group. Specific vendor implementations must only implement these interfaces to provide the function needed for response time measurement.

The ITCAM for Response Time Tracking implementation or ARM for Java implementation uses Java Native Interface (JNI™) calls that are made on the specific platform. For ARM Version 4, the required JAR file to load is called armjni4.jar, and it needs the corresponding shared native library to be available.

5.2 Management methodology

ITCAM for Response Time Tracking uses several management resources and certain terminology, which we discuss in this section. The discussion includes:

- 5.2.1, “Management resources” on page 211
- 5.2.2, “Reporting group and agent group” on page 211
- 5.2.3, “Management implementation overview” on page 213

5.2.1 Management resources

The management resources for ITCAM for Response Time Tracking are:

Reporting group A grouping for monitors that share certain properties

Agent group A grouping for management agents that enables monitors to be applied to all its members

Schedule Time rules about which monitors must be active

Discovery Activity to find transaction or other server processing

Listening monitor Activity to collect server-based response time information from applications

Robotic monitor Activity to send automated transactions, either using Client Application Tracker, STI, or Rational Robot

Out of these resources, we discuss the important concepts: reporting group and agent group.

5.2.2 Reporting group and agent group

This section discusses the groups used in ITCAM for Response Time Tracking: the reporting group and agent group.

Agent group

An *agent group* assists monitor deployment. The agent group determines on which machines the monitor executes. Figure 5-11 shows an example of agent group consideration.

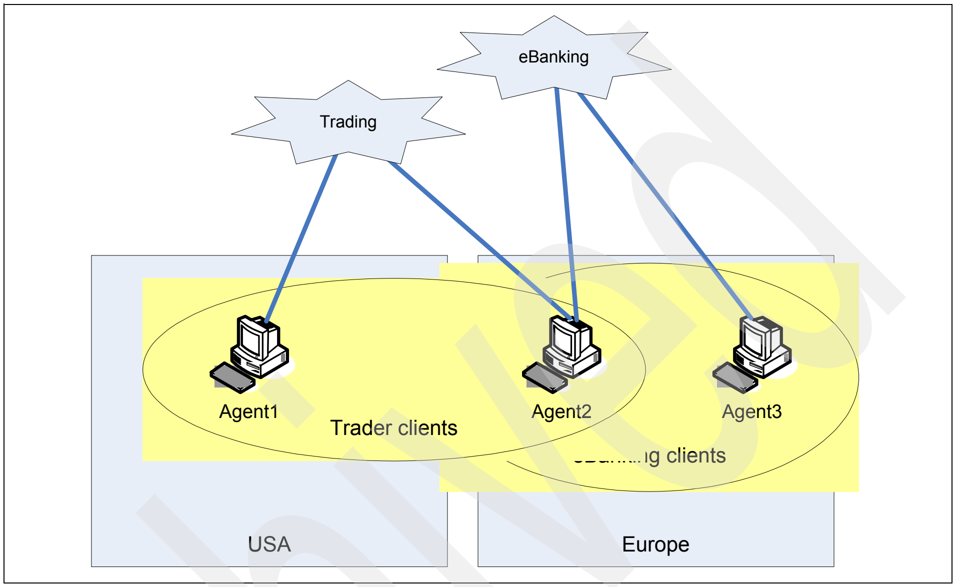


Figure 5-11 STI agent groups for multiple applications

Figure 5-11 shows three management agents for monitoring two applications from geographically separate sites. In this scenario, we want to monitor the Trading application on both Agent1 and Agent2. However, the eBanking application is available only in Europe, and the agent monitor for eBanking is deployed to Agent2 and Agent3. This case requires the trader clients and eBanking clients agent group. You might also need to have an agent group based on the location, so that U.S. clients and European clients can deploy local monitors.

You can create the agent group before creating any monitors using **Configuration** → **Agent Groups**. Alternatively, you can create the agent group when you create the monitor. However, some implementations might want to restrict this function to allow only the system administrator to add agent groups.

Reporting group

The *reporting group* differs from the agent group in that it does not directly affect the execution of monitors. A reporting group can control access to certain functions, using new role definition. Roles can be defined to allow read or update access for monitors *inside* the reporting group.

You can use the reporting group for:

- Defining who can access what. This relates to access levels of update, read, and none for the function.
- Simplifying access to the machine by operators.
- Providing aggregation in the Tivoli Enterprise Portal workspace display.
- Providing a way to group monitoring data by business function.

To anticipate additional administration work, carefully consider and document complex environments before deployment. Although you can create reporting groups at a later date, it is best to understand their importance up front to avoid future re-engineering. This can be done by examining the business requirements of the applications to be monitored and understanding the roles and responsibilities of those who will be using ITCAM for Response Time Tracking.

5.2.3 Management implementation overview

A typical management environment using ITCAM for Response Time Tracking must perform these actions:

1. Define reporting groups as a container to manage their monitors. Reporting groups also provide the ability to assign different reporting groups for different operators. This is assigned based on the role assignment.
2. Define agent groups to group the management agents so that monitor creation can use these groups to apply monitors to only the agents that are needed for the monitor.
3. Define monitoring schedules to determine the time schedule for monitoring transactions.
4. Discover either generic ARM-based transactions, J2EE instrumented transactions (servlets or Web services), or QoS instrumentations.
5. Create listening monitors for discovered transactions.
6. Define robotic transactions, either using Client Application Tracker, STI recorder, or Rational Robot.
7. Create a robotic monitor to simulate user response time using Client Application Tracker, STI recorder, or Rational Robot.

5.3 Implementation process

This is the overall implementation procedure for ITCAM for Response Time Tracking V6.1.

> **Note:** This chapter provides an overview of the ITCAM for Response Time Tracking main components and the installation process. Refer to *ITCAM for Response Time Tracking: Installation and Configuration Guide*, GC32-1907, for more detailed information.

The implementation process consists of the following steps:

1. Initially, we need to plan for the configuration. Decide upon the server configuration with your managed environment and the capability of the product. ITCAM for Response Time Tracking supports both single server environments and clustered environments. We describe the methods to deploy the ITCAM for Response Time Tracking management server installation and management agent.
2. The ITCAM for Response Time Tracking management server is the control center of the monitoring software. We recommend installing the ITCAM for Response Time Tracking management server before any other component.
3. The ITCAM for Response Time Tracking management agent is installed on the managed servers. This is a basic management agent without any specific components deployed. Deploy management components using the management console Web interface. The management agent can be deployed as an offline agent without checking the management server. However, the preferred method is checking communication at installation time.
4. The management components specify the functions of the management agent. In some cases, you can deploy multiple management components on the same management agent.
5. (Optional) Integrate the ITCAM for Response Time Tracking environment with the IBM Tivoli Monitoring V6.1 environment.
6. Create management objects, such as schedules, agent groups, and reporting groups, to prepare for the overall configuration setup. The management server installation or the monitoring components deployment creates the default schedules and agent groups. You can use the default schedules and agent groups for the initial configuration.

ITCAM for Response Time Tracking V6.1 has changed the terminology for product components, as shown in Table 5-1.

Table 5-1 Terminology changes in this release

New terminology	Old terminology
Monitor	Policy
Reporting group	Policy group
Discovery	Discovery policy
Listening monitor	Listening policy
Robotic monitor	Playback policy
Robotic script	Transaction monitoring

ITCAM for Response Time Tracking monitors using three methods:

- Discovery: defines an area of the Web environment to investigate and samples the transaction activity of real customer and average performance times. This is often the first step in creating a listening monitor.
- Listening monitor: collects performance data and produces detailed information about transaction performance times so that you can see detailed information about the transaction availability and performance. It can pinpoint problems as they occur with a listening monitor that monitors every step of real customer transactions.
- Robotic monitor: runs typical customer transactions and performance data from a robotic workstation. Problems can be identified as they occur using this robotic monitor.

The planning consideration topics include:

- 5.3.1, “Management server installation options” on page 215
- 5.3.2, “Management agent deployment” on page 216

5.3.1 Management server installation options

This section provides an overview of the ITCAM for Response Time Tracking main component’s installation methods and when to use them.

The ITCAM for Response Time Tracking management server is the control center for the monitoring software. It collects information from and provides services to the deployed management agents. The ITCAM for Response Time Tracking management server must be the first component of the management environment to be installed, so it is important to choose the best method to deploy it. Use the following methods to deploy the ITCAM for Response Time Tracking management server installation:

- Upgrade: Choose this option if you want to upgrade from a previously installed IBM Tivoli Monitoring for Transaction Performance Server or ITCAM for Response Time Tracking V6.0 Server. Refer to *ITCAM for Response Time Tracking: Installation and Configuration Guide*, GC32-1907, for the supported upgrade scenarios.
- Single Server - Silent: Choose this option to install a ITCAM for Response Time Tracking management server using a command-line interface without interaction (automated process).
- Single Server - Typical: Choose this option to install the ITCAM for Response Time Tracking management server and the required DB2 database and IBM WebSphere Application Server software.
- Single Server - Custom: Choose this option to install the ITCAM for Response Time Tracking management server, and use either an existing DB2 or Oracle database, an existing version of WebSphere Application Server on the target computer, or both on the target computer.

  > **Note**: If you install the DB2 database using the typical installation method of ITCAM for Response Time Tracking, it is configured with the default parameters for the database. However, using the custom installation enables you to customize your database, which is important for a high transaction volume or large-scale environment.

- Cluster: Choose this option to deploy multiple ITCAM for Response Time Tracking management servers into an existing WebSphere Application Server cluster environment to provide fault tolerance and increased scalability. This installation is manual, and there is no upgrade option. More information about this option is provided in *Installing a Management Server in a WebSphere Cluster Environment,* SC32-1804.

We strongly recommend that you split the components when dealing with large-scale environments. Place separate servers for WebSphere Application Server with ITCAM for Response Time Tracking and the database server. Also consider using a clustering solution when needed.

5.3.2 Management agent deployment

ITCAM for Response Time Tracking management agents are responsible for identifying transactions that might need monitoring, collecting performance data by running regularly scheduled listening and robotic monitoring, and sending generated events to the management server. Install ITCAM for Response Time

Tracking management agent on each server where those operations are required (that is, every Windows, Linux, UNIX, or mainframe that needs to be managed). The operations are performed by a monitoring component, but you need a management agent to install a monitoring component.

> **Note:** Before deploying any monitoring component, a management agent installation is required on the target box.

You decide which monitors to use before the monitoring components' deployment. It is important to have good understanding of the management agent deployment. Table 5-2 helps you determine which monitoring components to use for deployment.

Table 5-2 Considerations for the monitoring components deployment

If you want to...	**Monitoring components**
Monitor user transactions on IBM HTTP Server. For example, use this if you want the following measurements: ► The time it takes the Web server to process and respond to the HTTP request. ► The time it takes to display a Web page on a browser. ► The time it takes to complete the entire page request (round-trip time). This includes the previous two times and the network and data transfer time.	Web Response Monitor.
Monitor the internals of the infrastructure server, such as WebSphere Application Server, WebSphere Portal products, WebLogic, Tomcat, or JBoss so that you can: ► View detailed reports to perform root-cause analysis. ► Identify the exact cause of slowdowns and bottlenecks. ► Monitor Business Process Execution Language (BPEL) processes running on a WebSphere Process Server.	For WebLogic and WebSphere, use J2EE servlet. For Tomcat and JBoss, use the Tomcat and JBoss monitors.
Monitor response time data for Web services invocations.	J2EE Web Services.
Monitor the performance of ARM-instrumented applications, such as Siebel®.	ARM Application.
Monitor user response times for IBM Lotus Notes®, Microsoft Outlook®, SAP, 3270 emulators, or applications running in a Citrix environment.	Client Application Tracker.

If you want to...	Monitoring components
Run a custom application, script, or command and see results. For example: ▶ Testing server availability with FTP, telnet, or ping ▶ Querying a database with a custom SQL command ▶ Running a custom shell script	Generic Playback.
Perform transaction decomposition by one of the following methods: ▶ Viewing detailed reports so that you can perform root-cause analysis ▶ Identifying the exact cause of slowdowns and bottlenecks ▶ Monitoring back-end services, page render, and round-trip times	Synthetic Transaction Investigator with both Quality of Service and J2EE thresholds.

We decided to deploy the management agent on:

- All WebSphere Application Servers that host the trader applications
- CICS and IMS servers
- Tomcat and JBoss Application Servers
- Generic Windows (it runs Rational Robot), ARM application, Web Response Monitor, Quality of Service, Client Application Tracker, Generic Playback, and Synthetic Transaction Investigator servers

Figure 5-12 shows our overall configuration for ITCAM for Response Time Tracking.

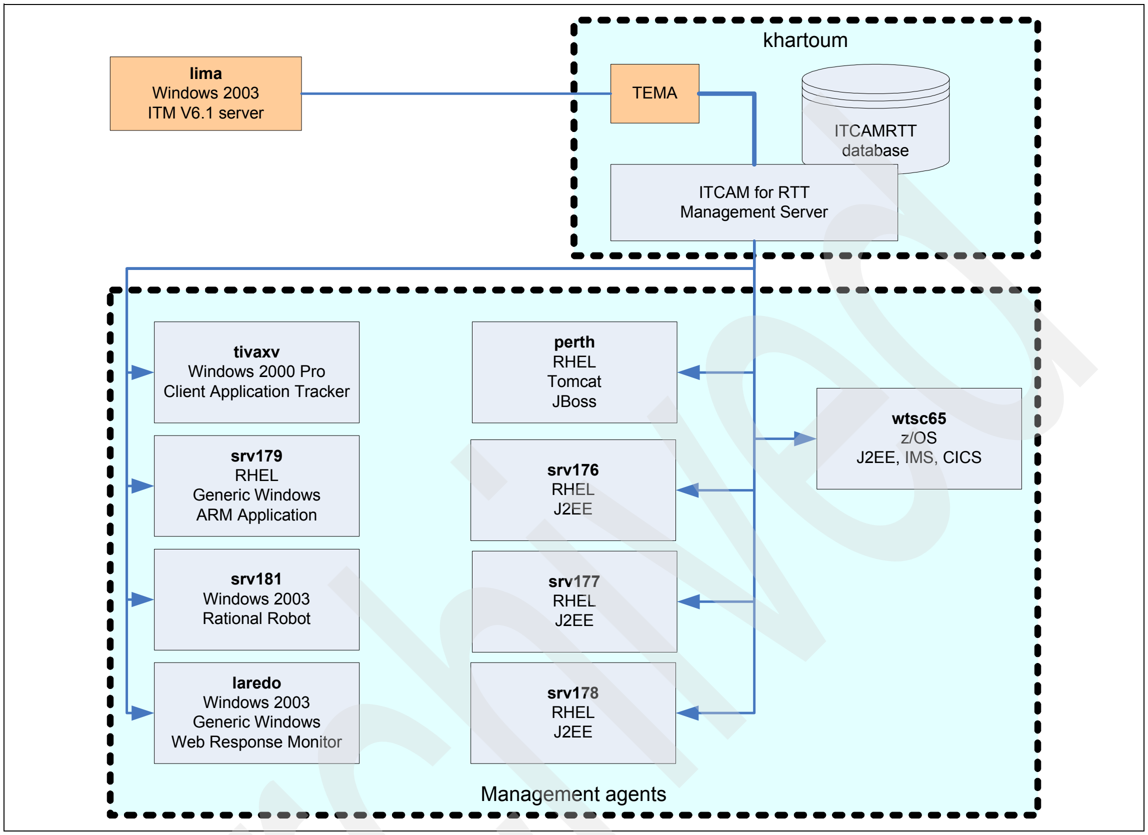


Figure 5-12 ITCAM for Response Time Tracking configuration

5.4 Installing ITCAM for Response Time Tracking

We divide the installation discussion into the following topics:

- 5.4.1, "Installing the management server" on page 220
- 5.4.2, "Installing the store and forward agent" on page 229
- 5.4.3, "Installing the distributed management agent" on page 233

Note: We discuss the z/OS-based management agent implementation separately in Chapter 10, "Implementation of ITCAM products on z/OS" on page 521.

5.4.1 Installing the management server

This section describes the installation of the ITCAM for Response Time Tracking management server. We followed the procedure for the single-server typical installation process. We illustrate the main steps of the typical installation. If you select the other installation process, refer to *ITCAM for Response Time Tracking Installation and Configuration Guide*, GC32-1907. We install the management server on a Linux Intel® R2.4 platform using the following procedure:

1. Prepare the operating system:
   a. Apply the required permissions to DB2 and WebSphere users. See *ITCAM for Response Time Tracking Installation and Configuration Guide*, GC32-1907.
   b. Modify the OS parameters, such as the kernel parameters for HP-UX or Solaris™.
   c. Allocate the file systems that are needed for the product path, such as /opt/IBM/itcam/RTT/MS or C:\Program Files\IBM\itcam\RTT\MS.
2. Prepare the information for the database and WebSphere Application Server installation. The database for ITCAM for Response Time Tracking can be either a DB2 or Oracle database. Install the database locally or on a remote database server. For a remote database server, install the database client on the management server. The management server is a J2EE application on WebSphere Application Server, and WebSphere Application Server must be installed before the management server. We use the typical installation, which can support an embedded installation for DB2 and WebSphere.
3. For DB2 stability, install the DB2 fix pack. We also describe this installation process.

Note: For coexistence with ITCAM for WebSphere or ITCAM for J2EE V6.1, install Fix Pack 1 for ITCAM for Response Time Tracking V6.1. See 5.4.4, "Installing ITCAM for Response Time Tracking fix pack 1" on page 237.

The typical installation supports an embedded installation of DB2 or WebSphere Application Server. The management server installation wizard starts from the management server CD-ROM for using LaunchPad. If you use a CD-ROM on UNIX or Linux, you must run the installer directly. Before starting the installation process, note all required installation parameters. Those are:

- WebSphere Application Server parameter:
  - WebSphere Application Server installation image and fix pack path
  - Whether to enable Secure Sockets Layer (SSL)
  - User account information for WebSphere Application Server process
  - Administrative console port

  - WebSphere Application Server installation directory
- DB2 database parameter:
  - DB2 installation image path
  - DB2 administration server, fenced and instance owner user accounts
  - DB2 installation directory

The installation of the ITCAM for Response Time Tracking management server uses two CD-ROMs. The first CD is platform specific and the second is platform neutral. The wizard copies files from both CD-ROM files onto the disk. If you copy these CD-ROM images to disk manually, you must copy both CD-ROMs' content into the same path. Otherwise, the installation wizard fails.

The installation wizard is launched from the first disk using the **`launchpad`** command or by invoking the appropriate setup_MS executable. The installation procedure is:

1. Run launchpad.sh where the installation files are located. LaunchPad provides installation prerequisites, an installation guide, product support, and product updates in addition to installing ITCAM (Figure 5-13).

Figure 5-13 LaunchPad

2. We use the installation directory /opt/IBM/itcam/RTT/MS as the default location (Figure 5-14).

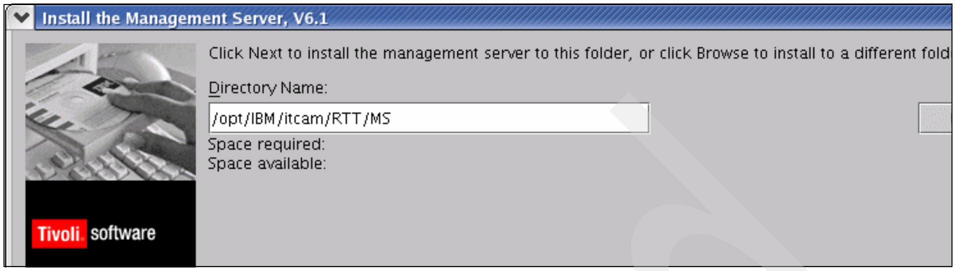


Figure 5-14 Installation path

3. The typical installation wizard can install an embedded WebSphere Application Server, as shown in Figure 5-15.

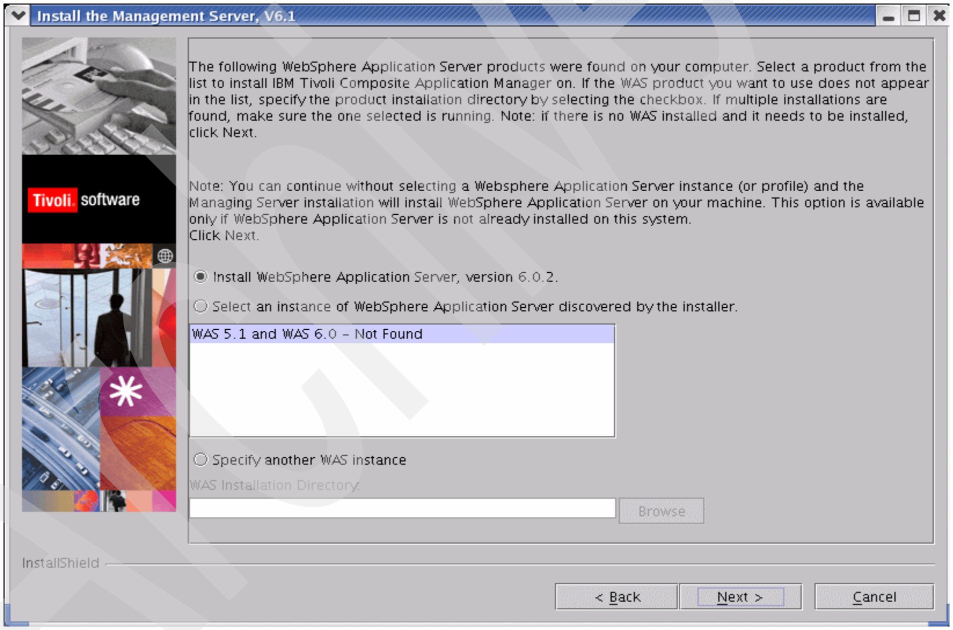


Figure 5-15 WebSphere Application Server installation

4. The management server uses several ports. We use SSL for the management server communication instead of non-SSL. It uses the port for the management server console with 9445. The non-SSL connection from the management agents or CLI uses port 9081. The SSL port is 9446, as shown in Figure 5-16.

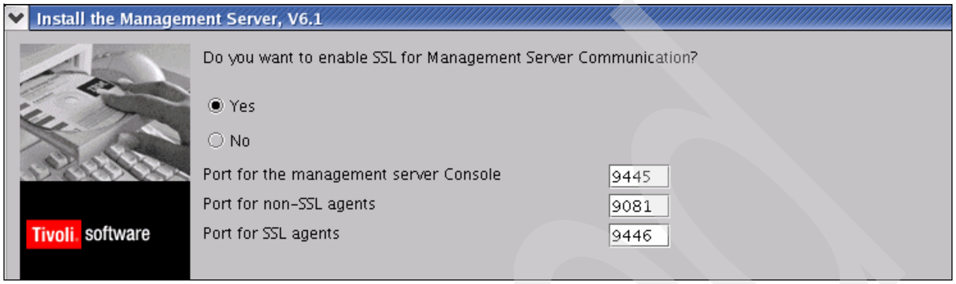


Figure 5-16 Port and SSL usage

> **Note:** To enable SSL for management server communication, you also must enable WebSphere Application Server global security. The typical installation does not support enabling the WebSphere Application Server global security in the installation process. We enable global security through WebSphere Application Server after ITCAM for Response Time Tracking installation. See Figure 5-17.

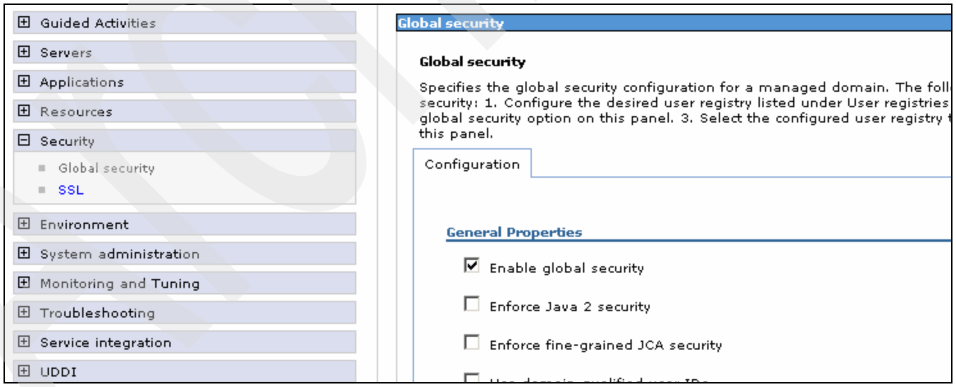


Figure 5-17 Enable global security

5. Next we install the WebSphere Application Server. Provide the configuration information for the WebSphere Application Server, as shown in Figure 5-18.

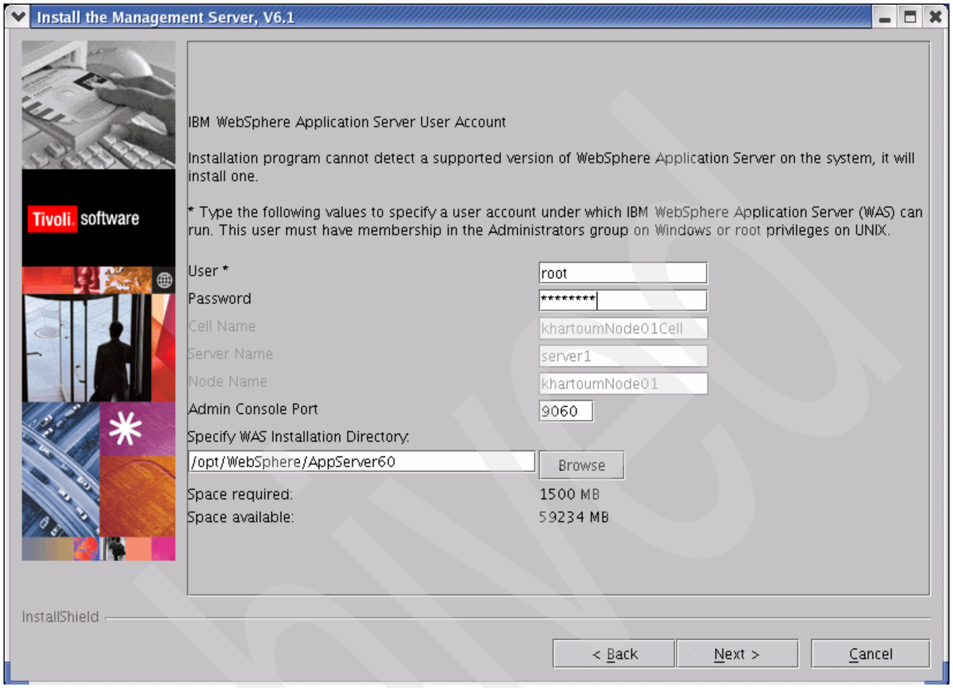


Figure 5-18 WebSphere Application Server parameters

6. The installation connects to the database installation. We select **Install DB2** for the database options (Figure 5-19).

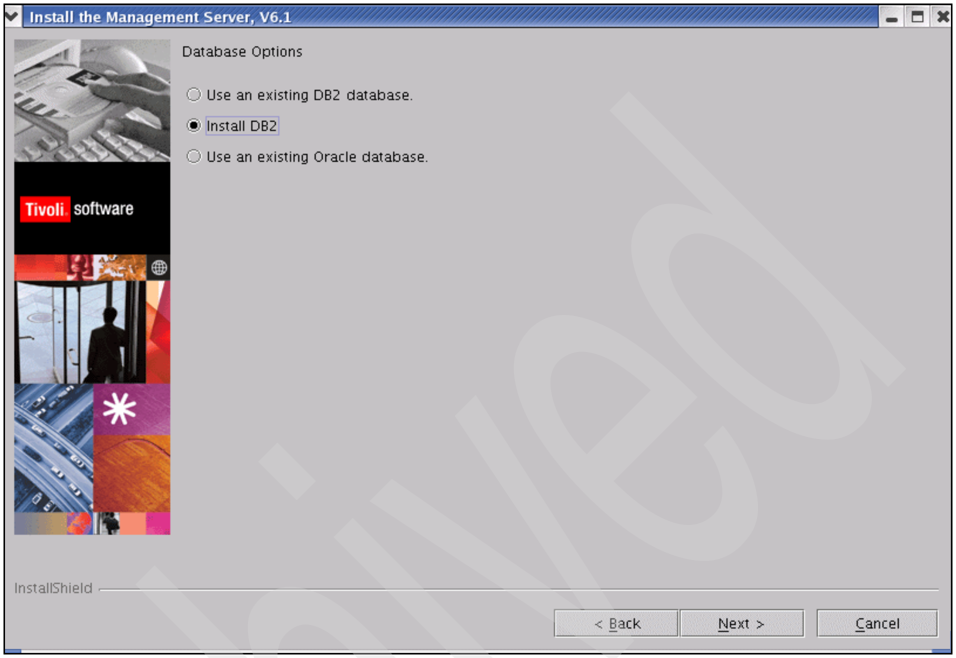


Figure 5-19 DB2 installation

7. The DB2 installation requires database user account information. Figure 5-20 shows the DB2 Administration Server User Account input window.

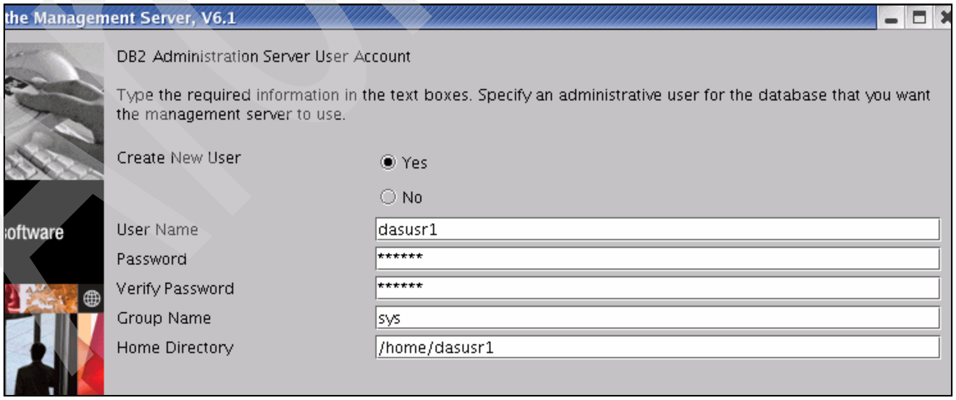


Figure 5-20 DB2 Administration Server User Account

Figure 5-21 shows the DB2 Fenced User Account input window.

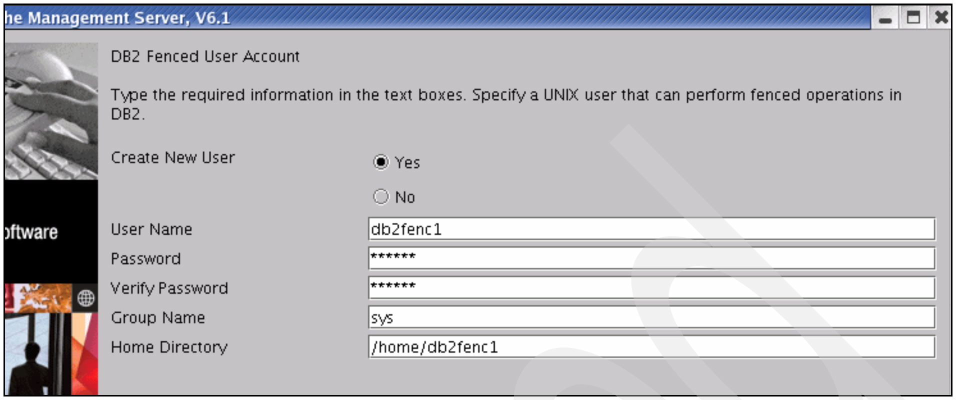


Figure 5-21 DB2 Fenced User Account

Figure 5-22 shows the DB2 Instance Owner User Account input window.

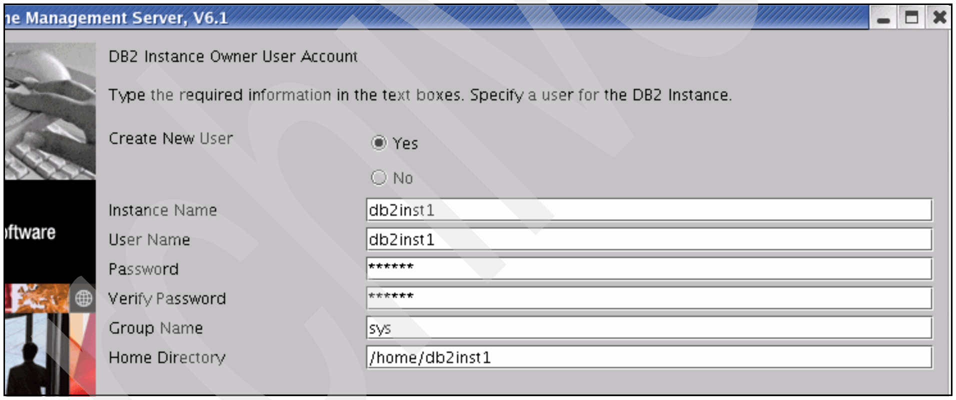


Figure 5-22 DB2 Instance Owner User Account

8. Fill in the installation path for embedded product installation. The wizard requests the installation image path of the WebSphere Application Server and DB2. WebSphere Application Server must be Version 6.0 with Refresh Pack 2, and DB2 must be Version 8.2, Modification Level WR21342.

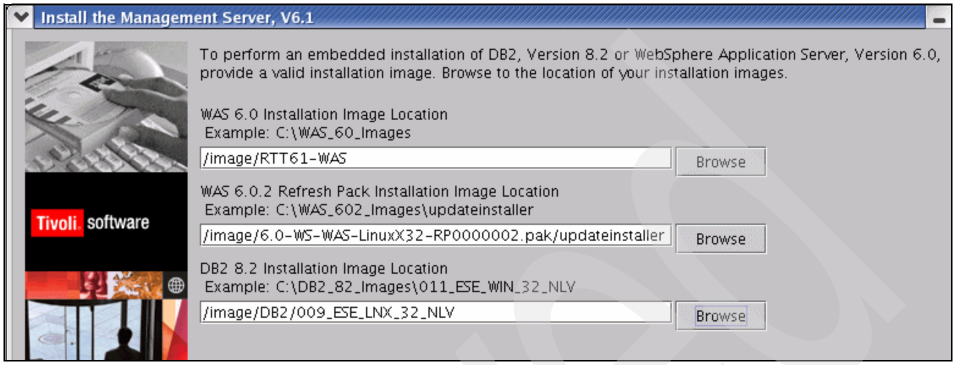


Figure 5-23 Installation image location

9. Click **Next** to accept the embedded product installation directories and ITCAM for Response Time Tracking management server installation directory.

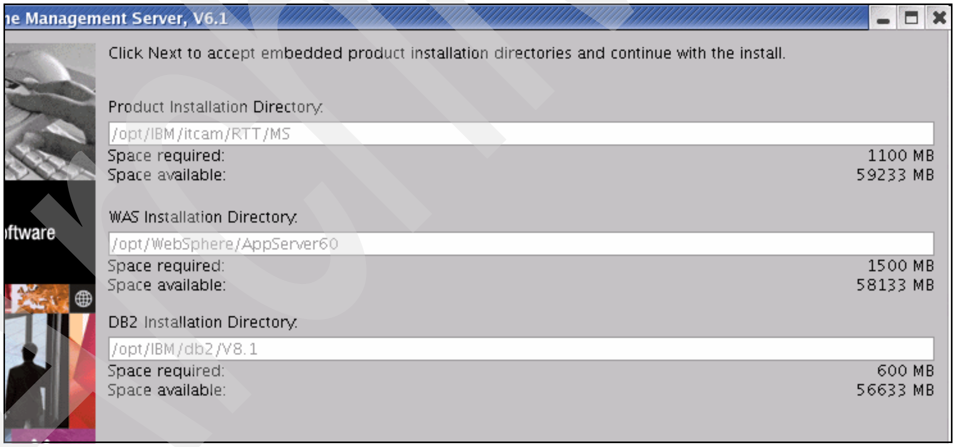


Figure 5-24 Product installation directory

10. The wizard starts an embedded WebSphere Application Server and DB2 installation. It then installs the ITCAM for Response Time Tracking management server, creates the database, and configures the management server application.

11. After the typical installation, we can access the management server using Microsoft Internet Explorer® (Figure 5-25):

```
https://<management server hostname>:9445/tmtpUI/
```

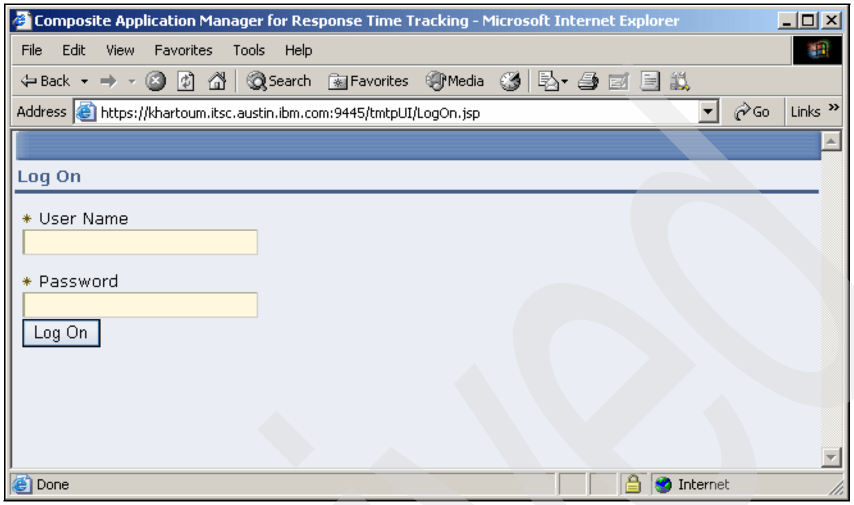


Figure 5-25 Management server console

Note: If you change the password of the db2inst1 account after installing ITCAM for Response Time Tracking, you get the following error message in the logon window:

```
BWMVZ2175E The server could not find the table USERSETTINGS in the database
```

You must change the password for the JDBC provider in the WebSphere Application Server:

1. In the WebSphere Application Server administrative console, go to **Resources** → **JDBC Providers** → **DB2 Universal JDBC Driver Provider - TMTP** → **Data Sources**.
2. Correct the password that is used to access the DB2 database for all TMTP-related data sources. In the data sources window, click **TMTP DataSource** and then go to **J2EE Connector Architecture(J2C) authentication data entries** → **db2Alias** and change the password for the db2inst1 user account.
3. Save the changes and restart the WebSphere Application Server.

We recommend installing DB2 Fix Pack 8 for DB2 stability. However, the ITCAM for Response Time Tracking management server prerequisite database version is DB2 8.2. You can download DB2 fix packs from the following site:

http://www.ibm.com/software/data/db2/udb/support/downloadv8.html

When applying DB2 fix packs on Linux servers, refer to the steps for completing the DB2 update shown in Example 5-1. This shows stopping the DB2 instances, installing the fix pack, updating the instances, and restarting the DB2 instances.

Example 5-1 DB2 fix pack installation

```
# su - db2inst1 -C db2 force application all
# su - db2inst1 -C db2stop
# su - dasusr1 -C db2admin stop
# ./installFixPak
# /opt/IBM/db2/V8.1/instance/db2iupdt db2inst1
# /opt/IBM/db2/V8.1/instance/dasupdt dasusr1
# su - db2inst1 -C db2start
# su - dasusr1 -C db2admin start
```

5.4.2 Installing the store and forward agent

Store and forward agents are needed when firewalls separate management agents from the management server. Installing the store and forward agent requires the IBM WebSphere Caching Proxy V6.0. This is part of the WebSphere V6.0 Edge Components. We found that the following tips help when installing the store and forward agent:

- Specify install from the CD so that the installation files are copied to a temporary directory on the local disk.
- Install the WebSphere Caching Proxy separately to avoid restarting in the middle of the installation.

Do not install from disk because the installation directory will be wiped out.

To install the store and forward agent:

1. We install the store and forward agent in C:\IBM\itcam\RTT\SnF (Figure 5-26).

Figure 5-26   Installation path for the store and forward agent

2. The window in Figure 5-27 shows the store and forward agent attributes. By default, the port that the agent listens to for the management agent connection is 9446 for SSL and 80 for non-SSL. Note that this is not the same as the management server default of 9446 and 9081.

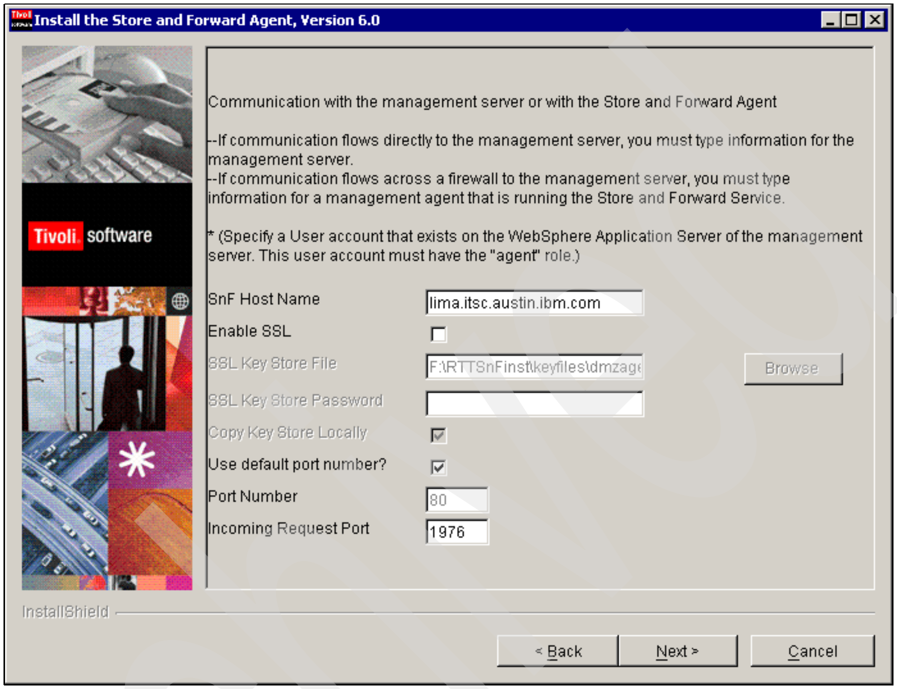


Figure 5-27 Store and forward agent attribute

3. Further attributes for the store and forward agent indicate the management server address and the authentication user. We use the primary ITCAM for Response Time Tracking user that we defined when installing the management server (Figure 5-28). Here we also define the management agent endpoint filter. The filter is a regular expression filter for the management agent IP addresses.

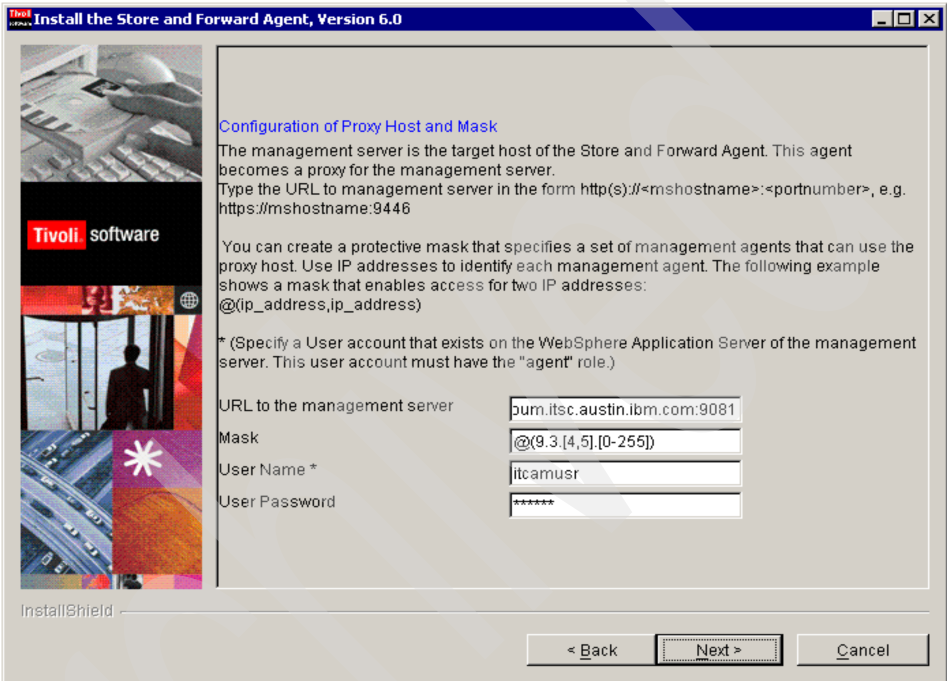


Figure 5-28 Additional store and forward agent properties

4. The wizard requests a certificate that might need to be used and the user ID to start the agent.

   If the WebSphere Caching Proxy is not installed, the wizard requests the path of the WebSphere Edge Components V6 CD, as shown in Figure 5-29.

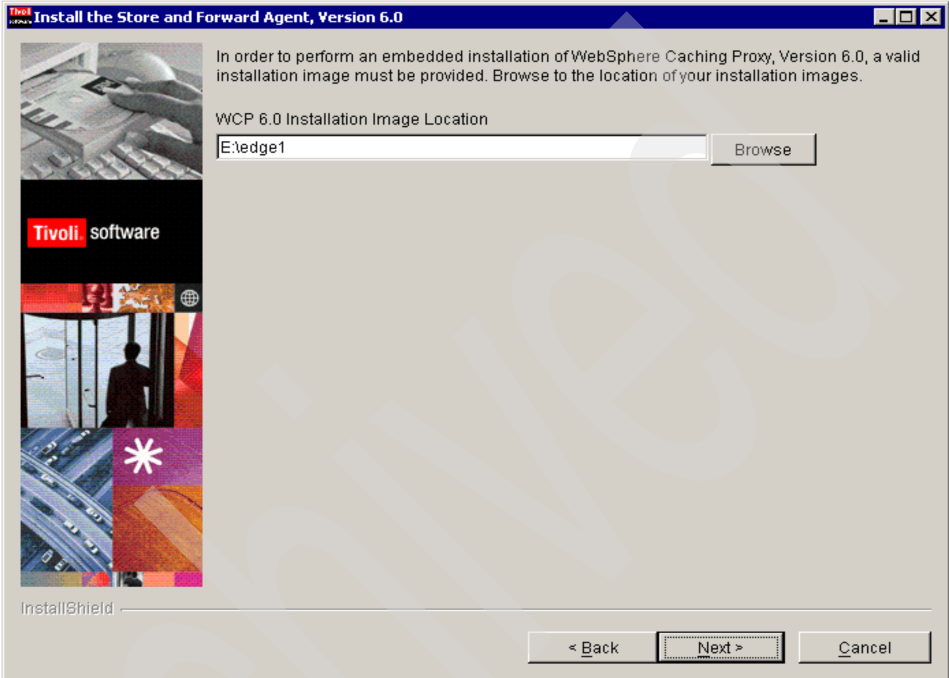


Figure 5-29 WebSphere Edge Component CD image

5.4.3 Installing the distributed management agent

In this section, we describe the main steps of the management agent installation. Installation considerations:

- For a Windows system:
  - Ensure access to the following user rights:
    - Act as part of the operating system.
    - Create a token object.
    - Log on as a service.
    - Replace a process level token.
    - Shut down the system.

– Configure Windows component permissions using the **dcomcnfg** command. Figure 5-30 shows the windows for Microsoft Windows 2003. This is required for the management agents running Robotic monitoring components.

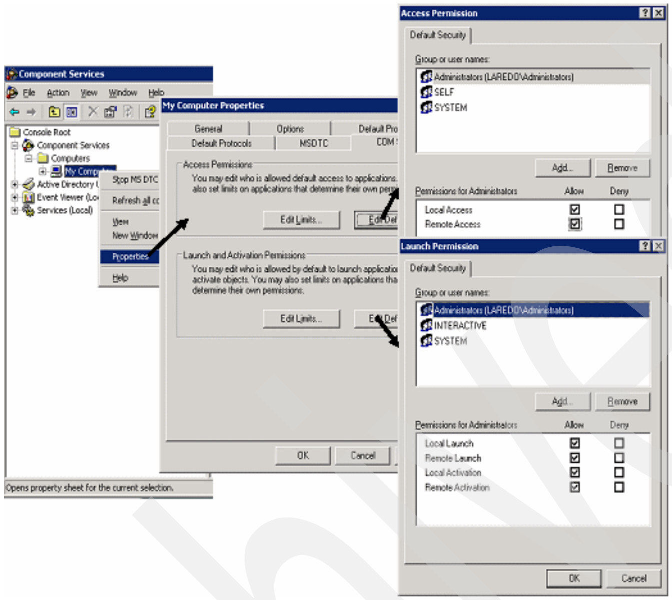


Figure 5-30 Component permission

- For UNIX or Linux systems: Ensure that you have X Window System access. If you are running from a command-line session, make sure that you have access to an X Server and export your DISPLAY environment variable accordingly.

To install the management agents:

1. We install all management agents for our environment, as shown in Figure 5-12 on page 219. The installation is platform dependent. In general, these are the options that we use:

   – Use a path without spaces in Windows, such as C:\IBM\itcam\RTT\MA, and the default path in Linux, such as /opt/IBM/itcam/RTT/MA. This is useful for command-line processing with ITCAM for Response Time Tracking.

   – The management server host name field must be the proper host name for your management server. Our server is khartoum.itsc.austin.ibm.com.

– For the Linux installation, we can use the root user, as shown in Figure 5-31.

– For the nonsecure installation, clear the Enable SSL option, as shown in Figure 5-31.

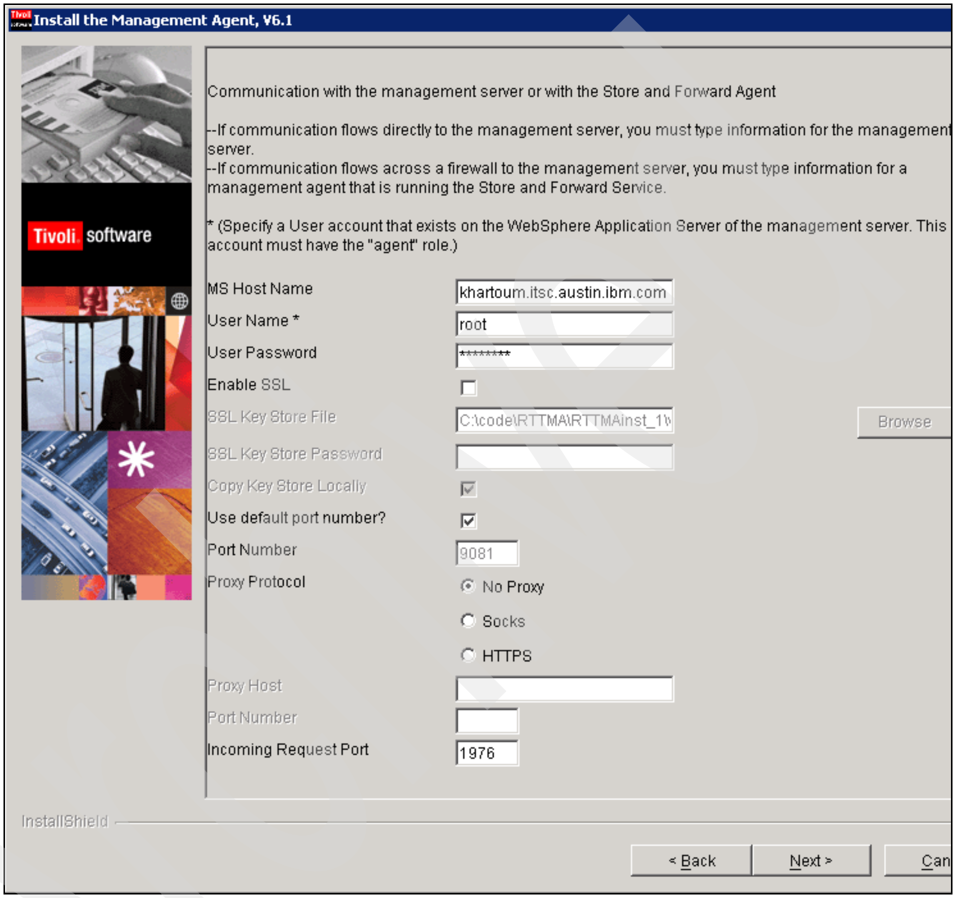


Figure 5-31 Management Agent options

Use an existing user account in Windows, Administrator, and specify the user account in Linux, root, as shown in Figure 5-32.

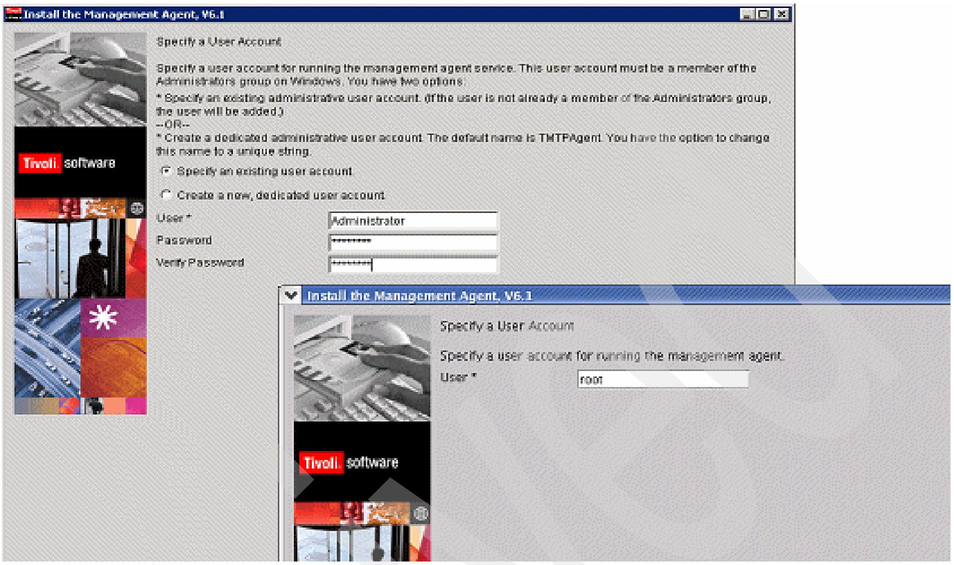


Figure 5-32 User Account for Management Agent

2. The installation wizard installs the ITCAM for Response Time Tracking management agent. After the installation, the ITCAM for WebSphere data collector automatically launches (Figure 5-33). If you select **Yes, Install**, it starts with the ITCAM for WebSphere data collector installation.

Figure 5-33 Launching ITCAM for WebSphere data collector installation

3. Verify the ITCAM for Response Time Tracking management agent installation by looking at the Status field on **System Administration** → **Agents**.

5.4.4 Installing ITCAM for Response Time Tracking fix pack 1

In this section, we describe the coexistence method of the ITCAM for J2EE data collector and the J2EE monitoring component of ITCAM for Response Time Tracking on the same WebSphere Application Server. The 6.1.0.0-TIV-RTT-FP0001 for ITCAM for Response Time Tracking V6.1 fixes a previous issue of the toolkit's setting being overridden by the ITCAM for Response Time Tracking installation. You can download the product fix pack from this site:

`http://www3.software.ibm.com/ibmdl/pub/software/tivoli_support/patches/`

To install the fix pack on the management server:

1. Extract the 6.1.0.0-TIV-RTT-FP0001 tar file.

2. Run the 6.1.0.0-TIV-RTT-FP0001_lin.bin, which opens the installation wizard, as shown in Figure 5-34. To continue, click **Next**.

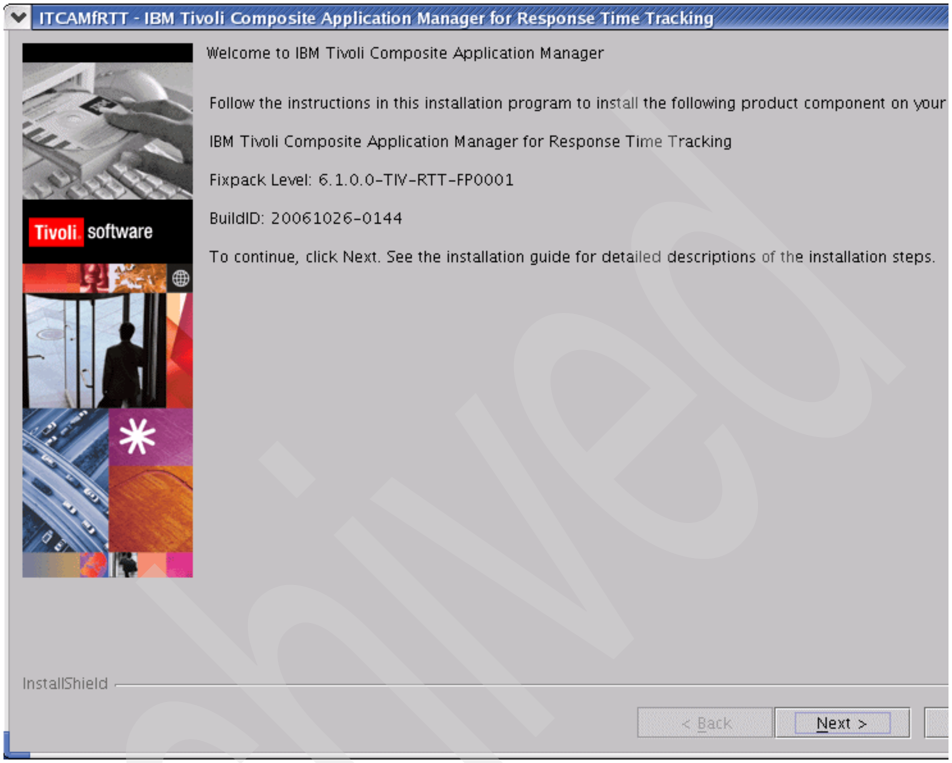


Figure 5-34 The fix pack installation wizard

3. Read the information and click **Next**.
4. Check the management server location and click **Next**.
5. Confirm the WebSphere Application Server value and click **Next**.
6. Check the installation options and click **Next**. The the fix pack installation on the management server starts.
7. After installing the fix pack, you can check errors in the installation_log.txt file.

Note: For an installation error with messages similar to the following messages in the installation_log.txt file, the cause is a too-small SOAP timeout:

```
wsadmin.sh -javaoption -Xms128m -javaoption -Xmx256m -conntype SOAP
-port 8880 -user root -password its0g00d -f
/opt/IBM/itcam/RTT/MS/ear/updateEAR.jacl Failed status code: 105

Setup.product.install, com.tivoli.ismp.ExecCommandProductAction,
err, An error occurred while updating the EAR Binaries for
WebSphere. For more information refer to the
$(absoluteInstallLocation)/6.1.0.0-TIV-RTT-FP0001/installation_log.t
xt file.
```

The solution is to change the default timeout of 180 from the soap.client.props file in <WAS_HOME>/profiles/<profileName>/properties/ and modify:

```
com.ibm.SOAP.requestTimeout = 600
```

Then restart WebSphere Application Server.

The following steps describe how to install the fix pack on the management agent:

1. To deploy the fix pack on the management agent, select **System Administration** → **Agent Updates**. Choose **6.1.0.0-TIV-RTT-FP00001** from the Available Updates menu. Select the agent on which to upgrade from the table. Select **Install Update** from the menu and click **Go**. This starts the fix pack installation on the management agent.

   Note: If the upgrade fails with agent updates, you can remove the monitoring components and redeploy the monitoring components afterward.

2. After installing the fix pack on the management agent, verify the management agents level. Select **System Administration** → **Agents Updates** and look at the Agent Update Status field and component update status, as shown in Figure 5-35. Both should say `Up-to-date`.

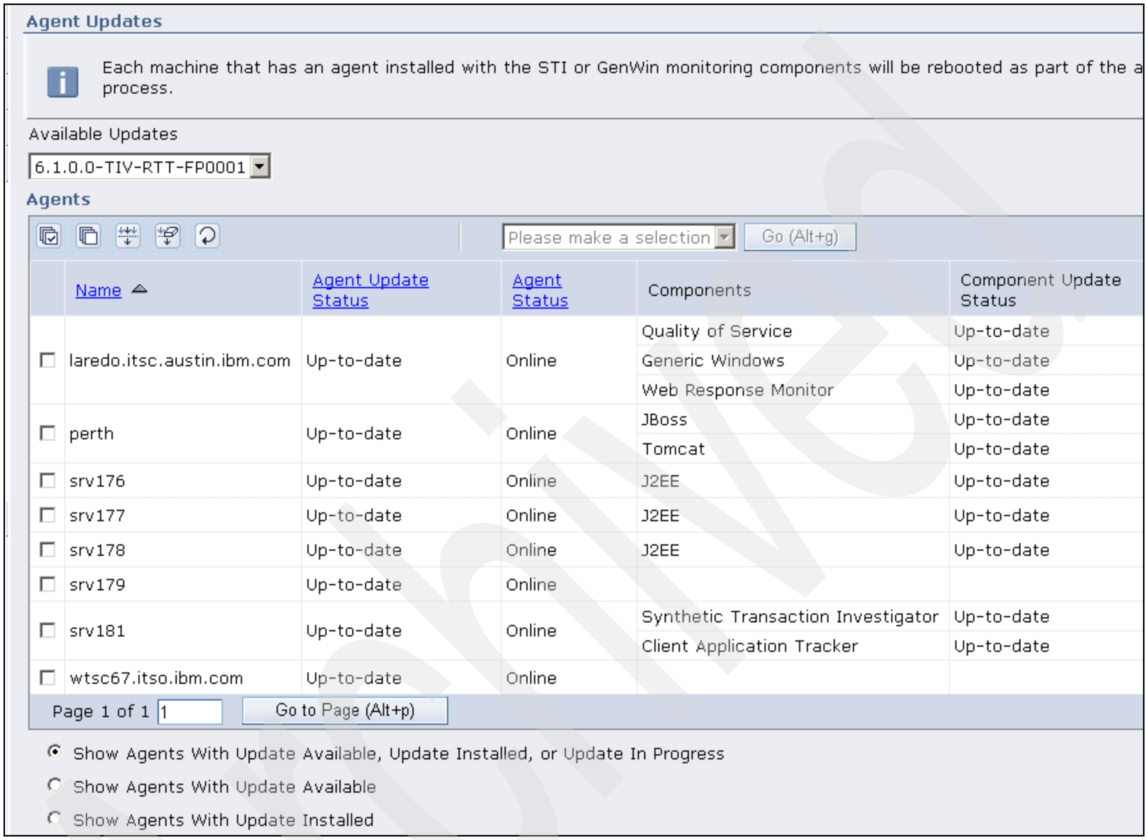


Figure 5-35 Complete agent updates

The following steps describe how to install the fix pack on the z/OS management agent. Get the z/OS package and put it on a temporary location in the UNIX System Services part of the z/OS machine. Perform the following steps:

1. Stop the management agent:

   `/etc/tmtp/MA/config/stop_tmtpd.sh`

2. Extract the parts.tar file to the base management agent directory. Specify the o option of the **tar** command to preserve the owner ID on your system. For example:

```
cd /usr/lpp/tmtp/V6R0M0/MA
tar xvfo parts.tar
```

3. The package contains the files used to deploy the J2EE component:

   – MA/lib/j2eeinst.jar.EJBs
   – MA/lib/j2eeinst.jar.noEJBs

   Verify that the j2eeinst.jar file in the MA/app/instrument directory corresponds to the desired behavior when deploying the J2EE component to this management agent.

   For performance reasons, use the MA/lib/j2eeinst.jar.noEJBs file as the file to be copied over the MA/app/instrument location.

   If your monitoring requirements need the enablement of EJB edges, you can copy the j2eeinst.jar.EJBs file to the MA/app/instrument location.

4. Start the management agent:

```
#/etc/tmtp/MA/config/start_tmtpd.sh
```

5.4.5 Integrating with Tivoli Enterprise Monitoring Server

ITCAM for Response Time Tracking can be integrated into the IBM Tivoli Monitoring V6.1 infrastructure, enabling you to exploit the full IBM Tivoli Monitoring V6.1 functionality such as Tivoli Data Warehouse V2.1, correlate response time information with data from other IBM Tivoli Monitoring V6.1 agents, and have this information presented in the Tivoli Enterprise Portal.

The integration into IBM Tivoli Monitoring V6.1 infrastructure uses Tivoli Enterprise Monitoring Agent for ITCAM for Response Time Tracking. The Tivoli Enterprise Monitoring Agent task retrieves information from the ITCAM for Response Time Tracking management server and forwards response time information to the Tivoli Enterprise Monitoring Server.

The Tivoli Enterprise Monitoring Agent is shipped as part of ITCAM for Response Time Tracking. It connects to the ITCAM for Response Time Tracking management server and collects information at customizable intervals. This enables you to install it on any box with a TCP/IP connection to the ITCAM for Response Time Tracking management server. It is much easier to have the management server itself host the Tivoli Enterprise Monitoring Agent because the Tivoli Enterprise Monitoring Agent host name is shown in the Tivoli Enterprise Portal as the source of the monitor.

Note: You must have an existing IBM Tivoli Monitoring V6.1 infrastructure set up. ITCAM for Response Time Tracking does not contain any IBM Tivoli Monitoring V6.1 infrastructure components such as Tivoli Enterprise Monitoring Server or Tivoli Enterprise Portal Server. For additional information about the IBM Tivoli Monitoring product, see the IBM Tivoli Monitoring documentation.

The installation process basically has two steps, which we discuss in the following sections:

- "Preparing IBM Tivoli Monitoring V6.1" on page 242
- "Implementing Tivoli Enterprise Monitoring Agent" on page 244

Preparing IBM Tivoli Monitoring V6.1

To integrate ITCAM for Response Time Tracking information into the IBM Tivoli Monitoring V6.1 infrastructure, you must install application support files so that the infrastructure can 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 Tivoli Enterprise Portal clients with product-specific information such as help files

This section covers these steps from a process point of view and guides you through the installation process. For installation and customization, we follow the instructions in *ITCAM for Response Time Tracking: Installation and Configuration Guide,* GC32-1907.

Note: Before you begin installing the Tivoli Enterprise Monitoring Agent, you must have installed at least IBM Tivoli Monitoring V6.1 fix pack 001 or later. For your reference, we use IBM Tivoli Monitoring V6.1 fix pack 003.

Perform the following steps:

1. Start the installation by clicking **Setup** in the Windows directory of the ITCAM for WebSphere Tivoli Enterprise Monitoring Agent CD-ROM or by running install.sh in your Linux or UNIX environment.

   Figure 5-36 shows the Welcome window for the installation of ITCAM for Response Time Tracking. Click **Next**.

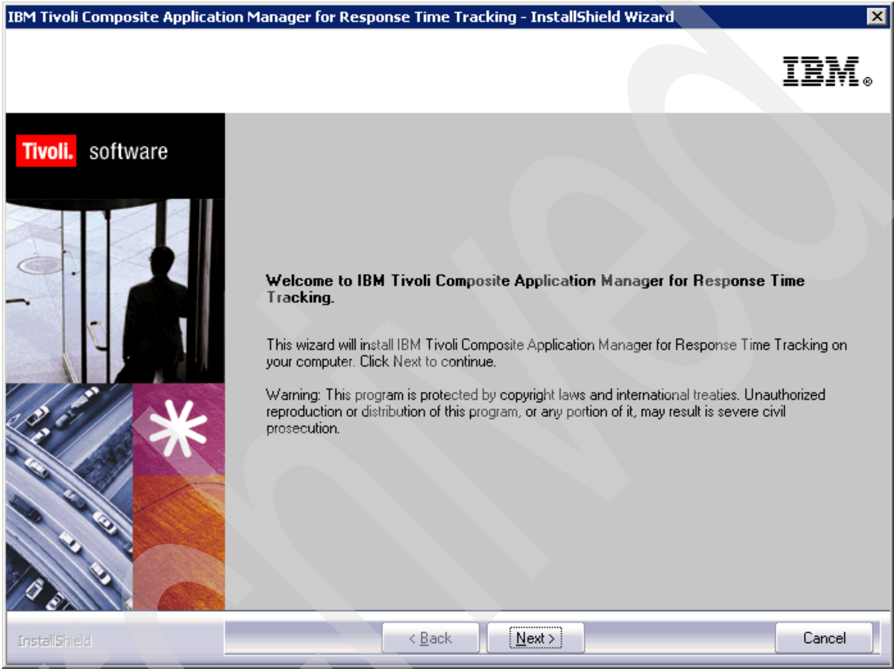


Figure 5-36 Tivoli Enterprise Monitoring Agent Welcome window

2. Click **Accept** to accept the license agreement.

3. On the Select Features window, select the following 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 Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server in a single machine. Therefore, we selected the following features (Figure 5-37 on page 244):

   – Tivoli Enterprise Monitoring Server
   – Tivoli Enterprise Portal Server
   – Tivoli Enterprise Portal Desktop Client

> **Note**: You might have to install this on your Tivoli Enterprise Monitoring Server machine, Tivoli Enterprise Portal Server machine, and individual Tivoli Enterprise Portal desktop clients. You only have to install the component that is available in each machine.

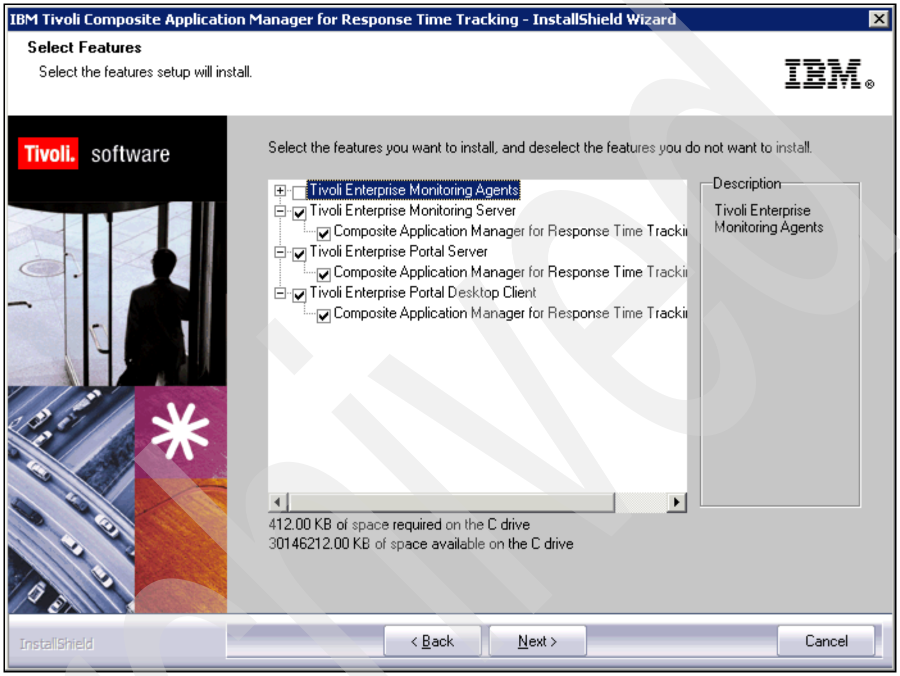


Figure 5-37 Tivoli Enterprise Monitoring Agent components

4. Select the agents to configure for remote deployment. If your requirements enable deploying a remote agent, select the component installation and click **Next**.
5. Follow the wizard to install and configure the components. Most components are already preconfigured, because we install to an existing IBM Tivoli Monitoring environment. The important option is to specify the communication to the Tivoli Enterprise Monitoring Server and the Tivoli Enterprise Portal Server. We use IP:PIPE, which is a TCP connection.

Implementing Tivoli Enterprise Monitoring Agent

Start the setup from the ITCAM for Response Time Tracking Tivoli Enterprise Monitoring Agent on the ITCAM for Response Time Tracking management server. The Windows installation uses the same wizard as in “Preparing IBM Tivoli Monitoring V6.1” on page 242.

Note the following differences on Windows:

- Install only the Tivoli Enterprise Monitoring Agent component.
- When the installation is finished, configure the agent communication to the Tivoli Enterprise Monitoring Server. The communication uses IP:PIPE, and you must specify the Tivoli Enterprise Monitoring Server host name.
- Enter ITCAM for Response Time Tracking management server information.

> **Note**: The ITCAM for Response Time Tracking Management Server port is defined in the server.properties file, which was in /opt/IBM/itcam/RTT/MS/config in our lab environment.

We configured the ITCAM for Response Time Tracking management server on our Red Hat Linux Enterprise Server. Start the installation process using the install.sh script, as shown in Example 5-2.

Example 5-2 Running install.sh

```
[root@khartoum rtt]# ./install.sh
Enter the name of the IBM Tivoli Monitoring directory
[ default = /opt/IBM/ITM ]:/opt/IBM/ITM

ITM home directory "/opt/IBM/ITM" already exists.
OK to use it [ y or n; "y" is default ]? y
Select one of the following:

1) Install products to the local host.
2) Install products to depot for remote deployment (requires TEMS).
3) Exit install.

Please enter a valid number:  1

Software Licensing Agreement
1. Czech
2. English
3. French
   .  .  .

Please enter the number that corresponds to the language
you prefer.

2
```

```
Software Licensing Agreement
   . . .
   . . .
Enter "1" to accept the agreement, "2" to decline it or
"99" to go back to the previous screen.

1
Preparing to install the IBM Global Security Kit (GSkit)
Preparing packages for installation...
gsk7bas-7.0-3.18
 Will enable automatic agent initiation after reboot.

Enter a 32-character encryption key, or just press Enter to use the
default
        Default = IBMTivoliMonitoringEncryptionKey
....+....1....+....2....+....3..
[Enter]
GSkit encryption key has been set.
Key File directory: /opt/IBM/ITM/keyfiles
Product packages are available for the following operating systems and
component support categories:

 1) Linux Intel R2.4 (32 bit)
 2) Linux Intel R2.4 (64 bit)
 3) Linux Intel R2.4 GCC 2.9.5 (32 bit)
 4) Linux Intel R2.4 GCC 2.9.5 (64 bit)
 5) Linux Intel R2.6 (32 bit)
 6) Linux Intel R2.6 (64 bit)
 7) Linux Intel R2.6 GCC 2.9.5 (32 bit)
 8) Linux Intel R2.6 GCC 2.9.5 (64 bit)
 9) Linux S390 R2.4 (32 bit)
10) Linux S390 R2.4 (64 bit)
11) Linux S390 R2.6 (32 bit)
12) Linux S390 R2.6 (64 bit)
13) Linux S390 R2.6 GCC 2.9.5 (64 bit)
14) Tivoli Enterprise Portal Browser Client support
15) Tivoli Enterprise Portal Desktop Client support
16) Tivoli Enterprise Portal Server support
17) Tivoli Enterprise Monitoring Server support

Type the number for the OS or component support category you want, or
type "q"
to quit selection
[ number "1" or "Linux Intel R2.4 (32 bit)" is default ]:
```

```
You selected number "1" or "Linux Intel R2.4 (32 bit)"

Is the operating system or component support correct [ y or n; "y" is
default
]? y

The following products are available for installation:

 1) IBM Tivoli Composite Application Manager for Response Time Tracking
V06.10.00.00
 2) Tivoli Enterprise Services User Interface  V06.10.01.00
 3) all of the above

Type the numbers for the products you want to install, or type "q" to
quit selection.
If you enter more than one number, separate the numbers by a comma or
a space.

Type your selections here:  1
The following products will be installed:

  IBM Tivoli Composite Application Manager for Response Time Tracking
V06.10.00 .00

Are your selections correct [ y or n; "y" is default ]? y
... installing "IBM Tivoli Composite Application Manager for Response
Time
Trac king  V06.10.00.00 for Linux Intel R2.4 (32 bit)"; please wait.

 => installed "IBM Tivoli Composite Application Manager for Response
Time
Tracki ng  V06.10.00.00 for Linux Intel R2.4 (32 bit)."
 ... Initializing database for IBM Tivoli Composite Application Manager
for
Resp onse Time Tracking  V06.10.00.00 for Linux Intel R2.4 (32 bit).
 ... IBM Tivoli Composite Application Manager for Response Time
Tracking
V06.10 .00.00 for Linux Intel R2.4 (32 bit) initialized.

Do you want to install additional products or product support packages
[ y or
n;  "n" is default ]? n

 ... postprocessing; please wait.
```

```
... finished postprocessing.

 Installation step complete.

 As a reminder, you should install product support on each of your TEM
 servers for any agents you have just installed. This is done via the
 "[ITM home]/bin/itmcmd support" command on your TEM servers.

 You may now configure any locally installed IBM Tivoli Monitoring
product via
 the "/opt/IBM/ITM/bin/itmcmd config" command.
```

To connect, the Tivoli Enterprise Monitoring Server and Agent need to be configured. Use the **itmcmd** command for the configuration (Example 5-3).

Example 5-3 Running itmcmd config

```
root@khartoum bin]# ./itmcmd config -A t2
Edit 'ITCAM for Response Time Tracking Management Server Identity'
settings? (default is: Yes): Yes
Response Time Tracking Management Server Host (default is: ): khartoum
Response Time Tracking Management Server Port (default is: ): 9445
Response Time Tracking User Login ID (default is: ): root
Response Time Tracking Login Password (default is: ): ********
Is Response Time Tracking Management Server SSL Enabled?
        Type number of item from the below list
        1. Yes
        2. No
         (default is: ): 1
Response Time Tracking Keystore File (default is: ):[Enter]
Response Time Tracking Keystore Password (default is: ):[Enter]
Edit 'Response Time Tracking Agent Configuration Options' settings?
(default is:  Yes): Yes
Maximum Timespan for Transaction Reporting in hours (default is: 8): 8
Maximum Timespan for Instance Selection in hours (default is: 1): 1
Maximum number of log messages (default is: 100): 100
Agent Messsage expiration (default is: 7): 7
Edit 'Response Time Tracking Managing Server Database Configuration
Options' set tings? (default is: Yes): Yes
Select the Database type for Managing Server Database
        Type number of item from the below list
        1. DB2
        2. ORACLE
         (default is: ): 1
```

```
Fully qualified hostname of the Managing Server Database Machine
(default is: ):  khartoum.itsc.austin.ibm.com
Specify Database Port: default for DB2 is 50000 and for Oracle is 1521
(default is: 50000): 50000
Database Name or SID Name (default is: ): TMTP
RTT MS Schema User Login Name (default is: ): db2inst1
RTT MS Schema User Login Password (default is: ): ****
Will this agent connect to a TEMS? [YES or NO] (Default is: YES): YES
TEMS Host Name (Default is: khartoum): lima
Network Protocol [ip, sna, ip.pipe or ip.spipe] (Default is: ip.pipe):

     Now choose the next protocol from one of these:
     - ip
     - sna
     - ip.spipe
     - none
Network Protocol 2 (Default is: none): none
IP.PIPE Port Number (Default is: 1918): 1918
Enter name of KDC_PARTITION (Default is: null):[Enter]

Configure connection for a secondary TEMS? [YES or NO] (Default is:
NO): NO
Enter Optional Primary Network Name or "none" (Default is: none): none
Agent configuration completed...
```

Use the **itmcmd** command to start the Tivoli Enterprise Monitoring Agent (Example 5-4).

Example 5-4 Running itmcmd agent

```
[root@khartoum bin]# ./itmcmd agent start t2
Starting agent...
Agent Started...
```

5.5 Deploying the monitoring components

This section describes the deployment of the monitoring components. After the management server and agents are set, you can deploy monitoring components to the installed management agents. This section provides the main steps of the deployment process of monitoring components.

The deployment considerations include:

5.5.1 Monitoring components in general

ITCAM for Response Time Tracking has 12 monitoring components, as shown in Table 5-3.

Table 5-3 ITCAM for Response Time Tracking monitoring components

Component	Usage
ARM Application	Monitors the performance of ARM 2.0 or 4.0-instrumented applications from ISVs (such as Siebel) or custom in-house applications. The ARM component can also detect and monitor custom metrics that are recorded from these ARM instrumented applications.
Client Application Tracker	Measures the round-trip response time of transactions originating at the client desktop. You can monitor Lotus Notes, Microsoft Outlook, SAP GUI, IBM Personal Communication 3270, Exceed, or Extra with this monitoring component.
Generic Playback	Enables you to schedule regular playback of any command on a management agent. It can upload and run any set of custom scripts or invoke commands that already exist on the agent.
Generic Windows	VU: enables you to monitor your Web applications by recording a script and playing it back as a robotic user. GUI: monitors your Microsoft Windows or Java applications by recording a script and playing it back as a robotic user. This agent runs Rational Robot.

Component	Usage
J2EE	Monitors J2EE application servers, such as WebSphere Application Server and BEA WebLogic. A single management agent can monitor multiple J2EE application servers on the management agent's host.
JBoss	Monitors JBoss Application Servers.
Mercury RoadRunner	Enables you to play back and monitor your Web applications by recording a Mercury LoadRunner script and playing it back as a robotic user on a management agent. HTTP(S) transactions generated by Mercury LoadRunner are automatically correlated with back-end monitored systems.
Quality of Service	Collects performance data for HTTP transactions for one or more Web servers, which includes the entire round-trip time of the transaction, the back-end service time, and the page display time.
Rational Performance Tester	Schedules regular playback of recorded scripts on a management agent, similar to the robot function.
Synthetic Transaction Investigator	Simulates the user experience while doing Internet transactions through a Web browser and traces a transaction that you can decompose using quality of service and J2EE performance measurements.
Tomcat	Monitors the Apache Tomcat server.
Web Response Monitor	Measures the performance of Web-based applications, provides response time and other performance data, and tracks navigation paths and usage behavior.

Some considerations regarding agent components include:

- You can use the components independently or in conjunction with another component. For example, using the Generic Windows component to play back a recorded transaction that targets a URL monitored by the Web Response Monitor component and is routed to an application server monitored by a J2EE endpoint provides the performance data available for that specific instance of the transaction.
- You can spread out the playback agents to collect measurements from different sites of the enterprise to get an overall response time measurement.

Figure 5-38 shows our final monitoring component assignment.

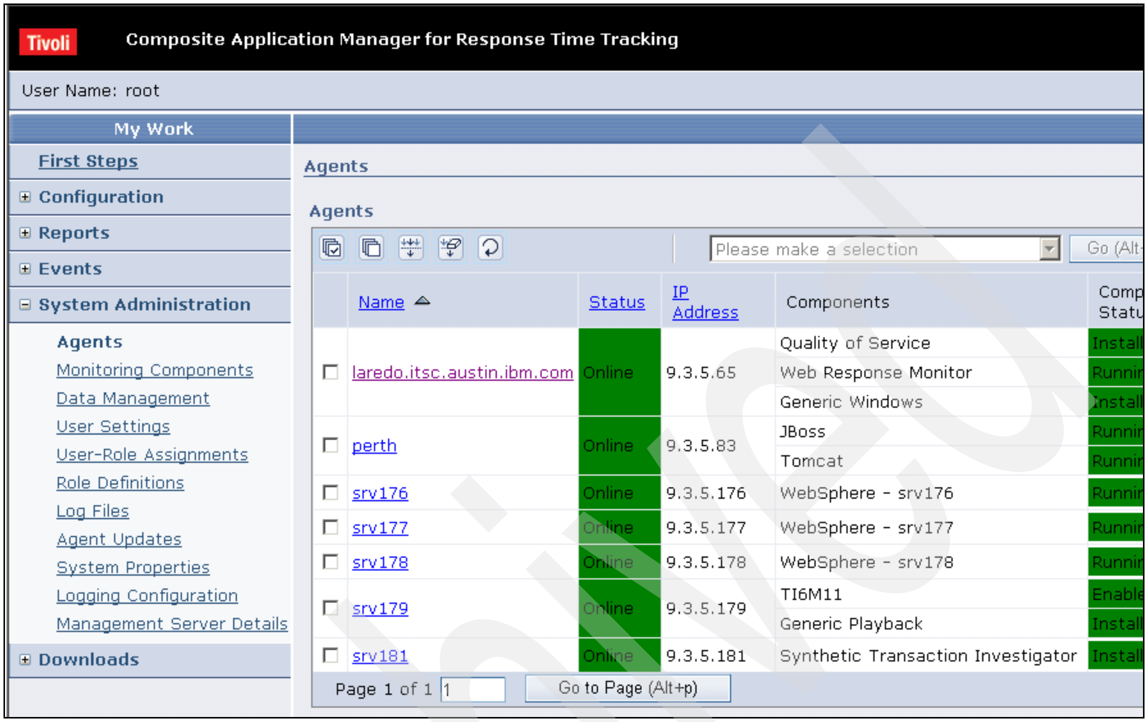


Figure 5-38 Monitoring component assignment

5.5.2 Installing the Rational Robot and Generic Windows components

The Generic Windows (GenWin) component uses Rational Robot to gauge how users experience Windows-based applications, Web-based applications, and custom applications. The Rational Robot software must be installed on the management agents that you want to play back Generic Windows scripts. You must install Rational Robot before you can deploy the Generic Windows monitoring component.

To install Rational Robot:

1. Download the robot2003.zip file from the management server by selecting **Downloads** → **Generic Windows Recorder**.
2. Extract the file on the local machine and run Robot2003_SR5.exe.

3. Accept the default setting by clicking **Next** until the License Key Administrator Wizard window. Select **Import a Rational License File**, as shown in Figure 5-39, and click **Next**.

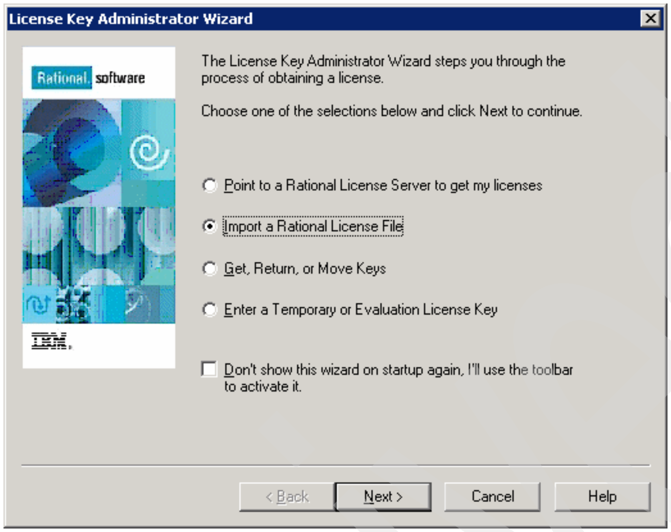


Figure 5-39 Select Import a Rational License File option

4. Click **Browse** and navigate to the directory where you extracted robot2003.zip and select the **ibm_robot.upd** file, as shown in Figure 5-40.

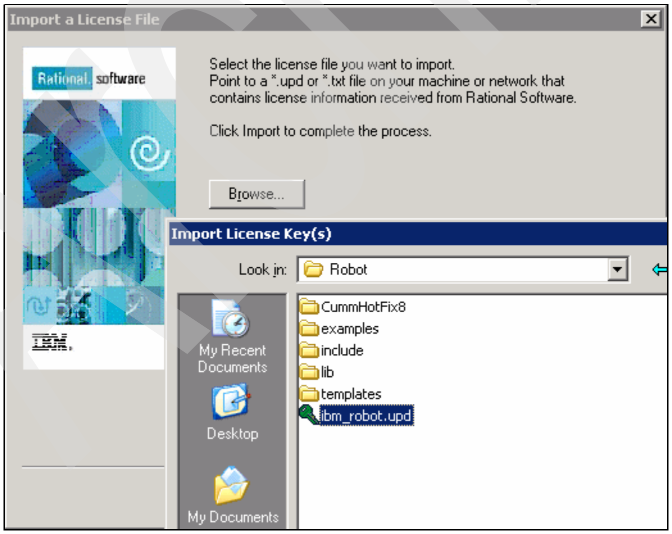


Figure 5-40 Import ibm_robot.upd file

5. Click **Import** again to confirm the import (Figure 5-41).

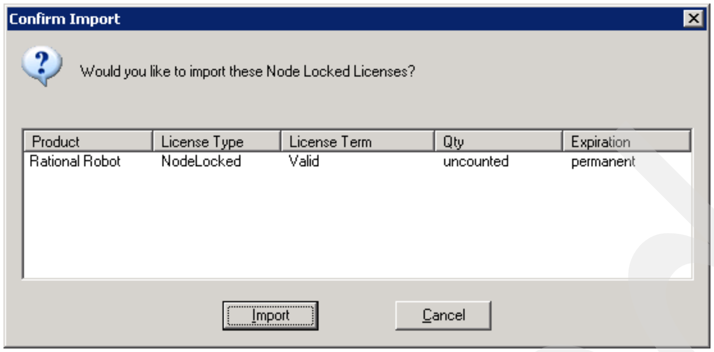


Figure 5-41 Confirm Import

6. After installing Rational Robot, restart the computer.
7. After installing Rational Robot, you must configure DCOM security permissions for the administrator using the DCOMCNFG utility. The default access and launch permission of DCOM do not give the Rational Robot Player account permission to launch Microsoft Internet Explorer. Permission must be given to the ITCAM for Response Time Tracking user and the user that runs Rational Robot recording. Refer to 5.4.3, “Installing the distributed management agent” on page 233, and *ITCAM for Response Time Tracking Installation and Configuration Guide*, GC32-1907.

Tip: Even after this procedure, you might get the following messages:

```
Setup failed to launch installation engine: Access is denied
```

Or:

```
Error installing ikernel.exe, access is denied
```

If these messages occur, follow the procedure described here:

http://support.microsoft.com/default.aspx?scid=kb;en-us;295278

To deploy the Generic Windows component:

1. Select **System Administration** → **Monitoring Components**. Select the **Generic Windows** component, and select the agent on which to deploy Generic Windows from the agent table, as shown in Figure 5-42.

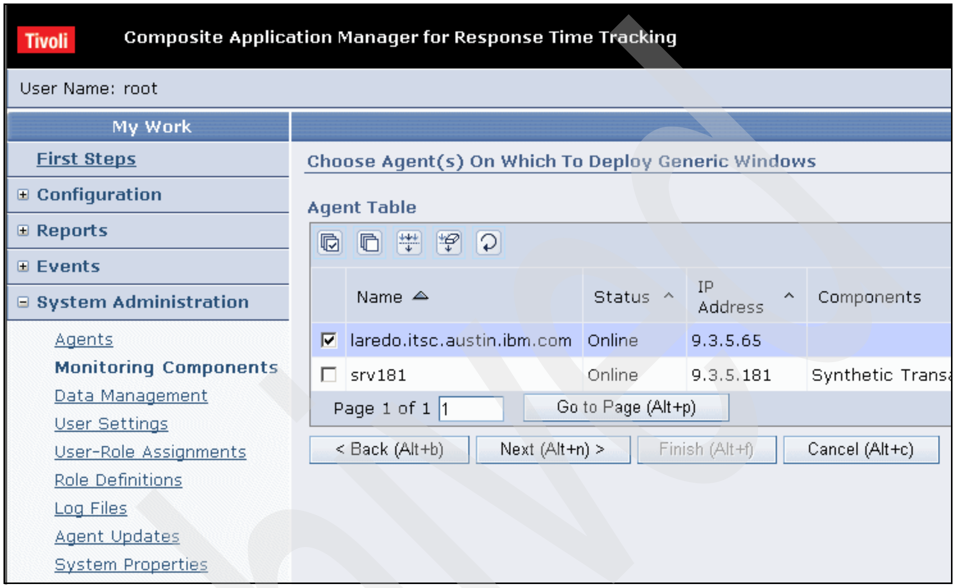


Figure 5-42 Choose agent for deploying Generic Windows

2. Enter values for the following fields to deploy the Generic Windows component. Figure 5-43 shows the deployment settings. We use the User ID, Password, and Project Name fields for running Robot scripts.

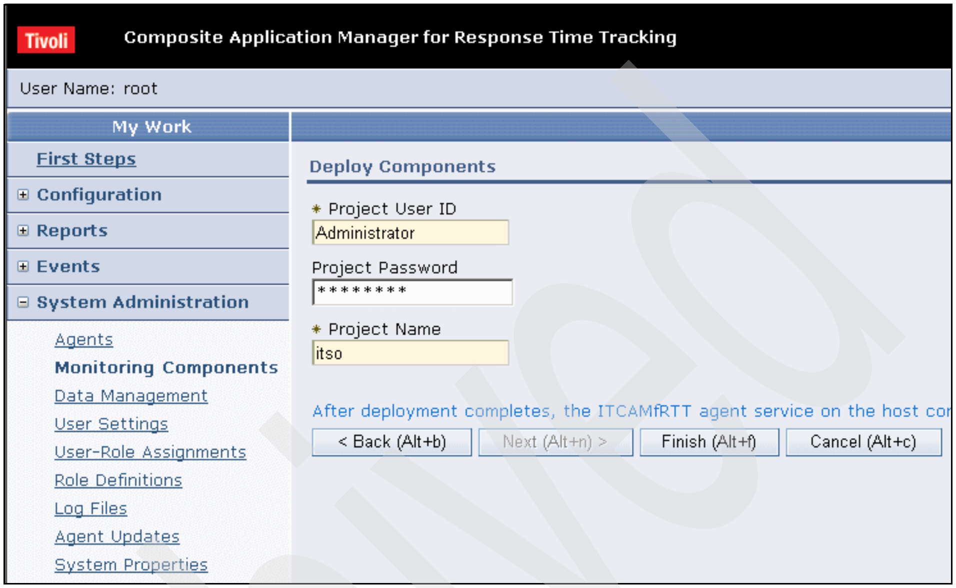


Figure 5-43 Deployment information for Generic Windows

3. After deploying Generic Windows, the agent reboots. Verify the Generic Windows deployment by looking at the Status field on the **System Administration** → **Agents** panel, as shown in Figure 5-44.

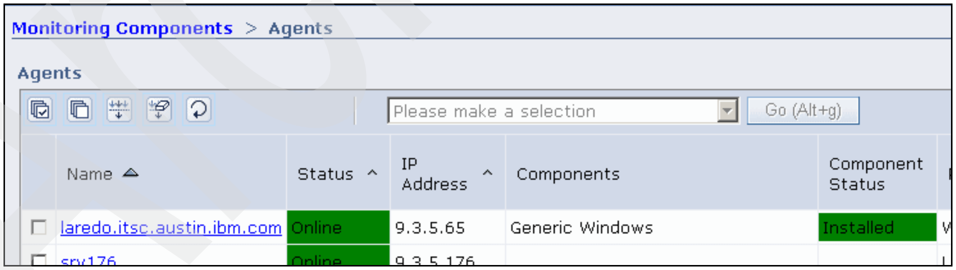


Figure 5-44 Complete the Generic Windows deployment

5.5.3 Deploying the Client Application Tracker component

The Client Application Tracker measures the round-trip response time of transactions originating at the client desktop. An agent sits invisibly at the client desktop and clocks transactions by setting a software stopwatch when the transaction begins. The monitoring component provides the ability to monitor the following versions of applications out-of-the box. You can download additional components (or behavior modules):

- Lotus Notes Versions 5 and 6
- Microsoft Outlook 2000 and 2003
- SAP Versions 6.2 and 6.4
- IBM Personal Communications, Hummingbird Exceed, and Attachmate Extra 3270 emulators

To deploy the Client Application Tracker component:

1. Select **System Administration** → **Monitoring Components**. Select **Client Application Tracker** from the available monitoring components table, and select **Deploy Monitoring Component** from the menu. Click **Go** (Figure 5-45).

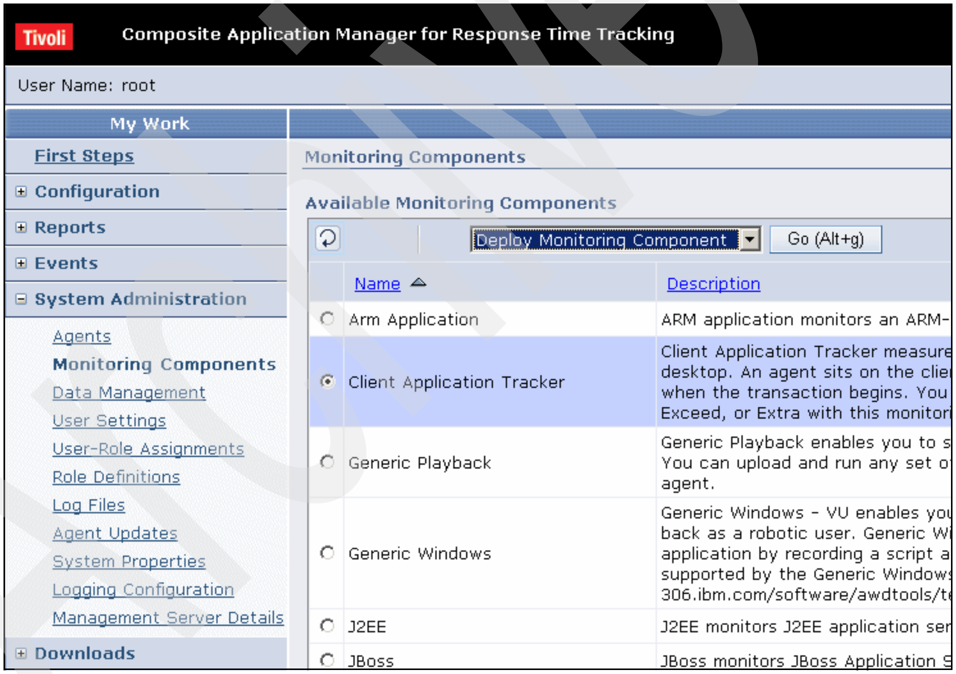


Figure 5-45 Deploy client application tracker component

2. Select the agent on which to deploy the Client Application Tracker from the agent table and click **Next**.

3. After deploying the Client Application Tracker, the agent reboots. Verify the Client Application Tracker deployment by looking at the Status field on the **System Administration** → **Agents** panel.

5.5.4 Deploying the J2EE component

The J2EE monitoring component provides a high-level view of transactions by collecting ARM data generated by J2EE applications. ITCAM for Response Time Tracking uses Java byte code instrumentation. You can monitor the following application servers with this components:

- WebSphere Application Servers
- BEA WebLogic
- WebSphere Portal products
- Business Process Execution Language (BPEL) processes running on a WebSphere Process Server

To deploy the J2EE monitoring component:

1. Select **System Administration** → **Monitoring Components**. Select **J2EE** from the Monitoring Components table, and choose **Deploy Monitoring Component** from the menu. Click **Go**. Choose the J2EE deployment method, as shown in Figure 5-46, and click **Next**.

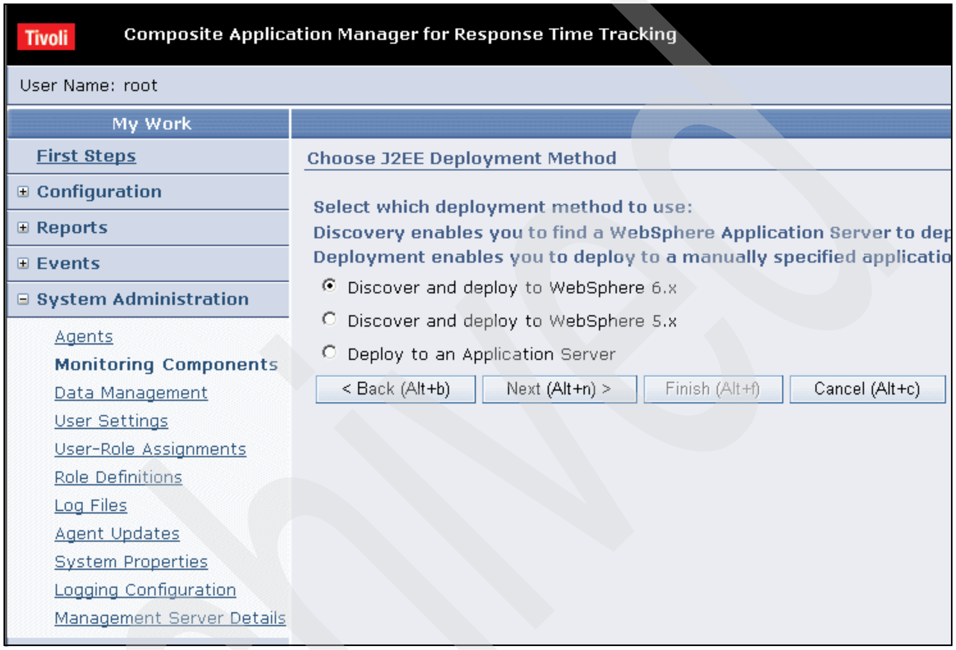


Figure 5-46 Choose J2EE deployment method

2. Select the agent on which to deploy J2EE from the agent table and click **Next**. Enter values for the following fields to discover J2EE in the agent and click **Next**. Figure 5-47 shows the J2EE discovery settings.

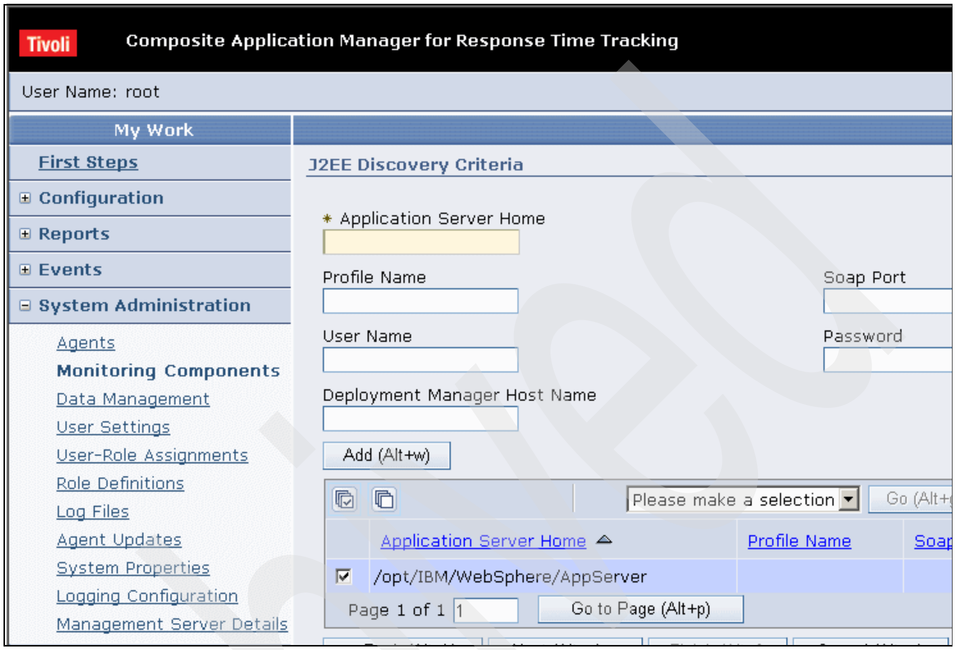


Figure 5-47 J2EE discovery criteria

3. After discovery of the J2EE component, select to deploy the J2EE component in the table and click **Finish** (Figure 5-48). It starts to deploy the J2EE component on the J2EE application server.

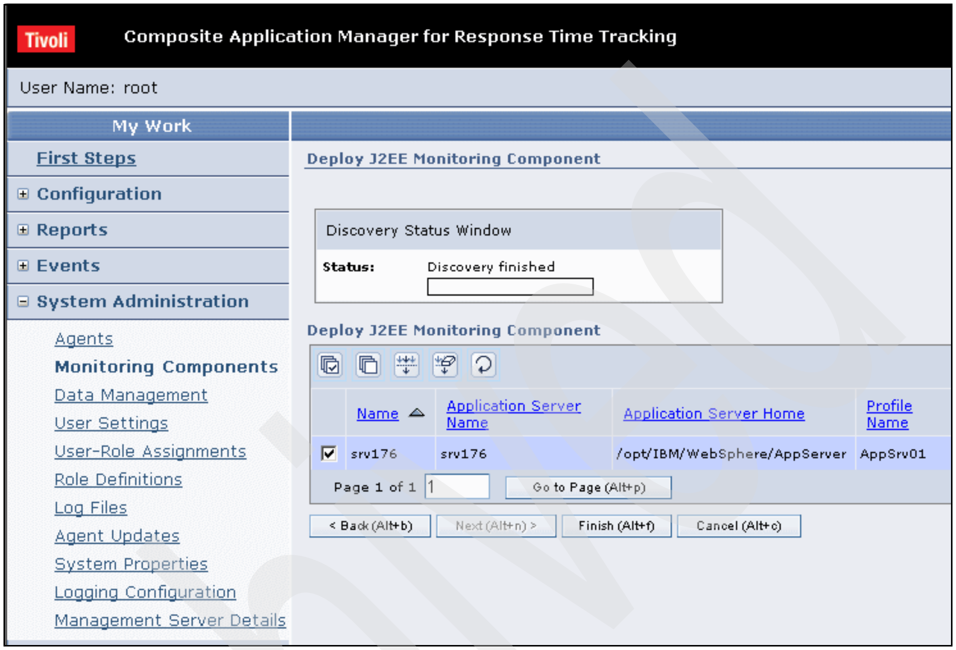


Figure 5-48 Deploy J2EE monitoring component

4. After deploying the J2EE monitoring component, the J2EE applicaiton server needs to restart. You can automatically restart the J2EE server while deploying the J2EE component by selecting the restart option in Figure 5-48. Verify the J2EE deployment by looking at the Status field in **System Administration** → **Agents**.

5.5.5 Deploying the JBoss and Tomcat components

With the JBoss and Tomcat monitoring components, you can monitor the JBoss Application Server and Apache Tomcat server. These components provide a high-level view of transactions by collecting ARM data generated by J2EE applications. You can monitor the following types of applications with these components:

- ► JBoss Application Server
- ► Apache Tomcat

To deploy the JBoss and Tomcat monitoring components:

1. Select **System Administration** → **Monitoring Components**. Select **JBoss** or **Tomcat** from the monitoring components table, and choose **Deploy Monitoring Component** from the menu. Click **Go**. Select the agent on which to deploy the JBoss or the Tomcat component in the agent table and click **Next**.
2. Supply the following information for the deployment monitoring components:
   – JBoss: JBoss Application Server Home, Java Home, and JBoss JMX Port (optional), as shown in Figure 5-49
   – Tomcat: Apache Tomcat Home and Java Home, as shown in Figure 5-50 on page 263

Figure 5-49 Required information to deploy JBoss

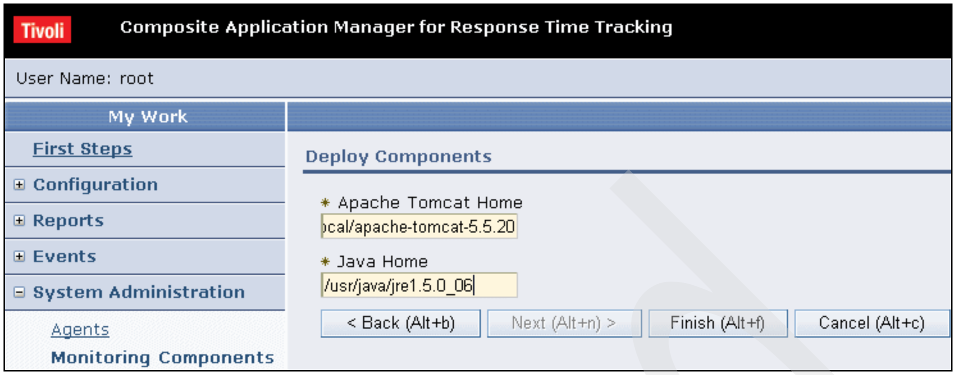


Figure 5-50 Required information to deploy Tomcat

3. Select the agent on which to deploy JBoss or Tomcat from the agent table and click **Next**.
4. After deploying JBoss or Tomcat, you need to restart the application server. Verify the JBoss or Tomcat deployment by looking at the Status field on the **System Administration** → **Agents** panel.

> **Note:** The JBoss and Tomcat J2EE component deployments change the following files:
>
> - <JBoss Application Home>/bin/run.sh
> - <Apache Tomcat Home>/bin/startup.sh
>
> The original file is backed up with the .orig extension.

5.5.6 Deploying the quality of service component

The quality of service monitoring component collects performance data for HTTP transactions for one or more Web servers, including the entire round-trip time of the transaction, the back-end service time, and the page render time. It is a reverse proxy. A forward proxy serves as a gateway for a client's browser, sending HTTP requests on the client's behalf to the Internet and protecting the internal network by masking the client's IP address and using its own. The quality of service component includes origin server, proxy server, and digital certificates.

To deploy the quality of service monitoring component:

1. Select **System Administration** → **Monitoring Components**. Select **Quality of Service** from the monitoring components table, and choose **Deploy Monitoring Component** from the menu. Click **Go**. Select the agent on which to deploy the quality of service component from the agent table and click **Next**.
2. To deploy the quality of service monitoring component, enter the required information in the fields, as shown in Figure 5-51.

Figure 5-51 Required information to deploy quality of service

3. After deploying the quality of service component, the agent reboots without a warning message. Be sure to make preparations for the reboot in advance. Verify the quality of service deployment by looking at the Status field on the **System Administration** → **Agents** panel.

5.5.7 Deploying the Synthetic Transaction Investigator component

The Synthetic Transaction Investigator (STI) component simulates a user performing Web transactions. STI is particularly well-suited for Web transaction playback.

To deploy the STI monitoring component:

1. Select **System Administration** → **Monitoring Components**. Select **Synthetic Transaction Investigator** from the monitoring components table, and choose **Deploy Monitoring Component** from the menu. Click **Go**. Select the agent on which to deploy the STI component in the agent table, as shown in Figure 5-52, and click **Next**.

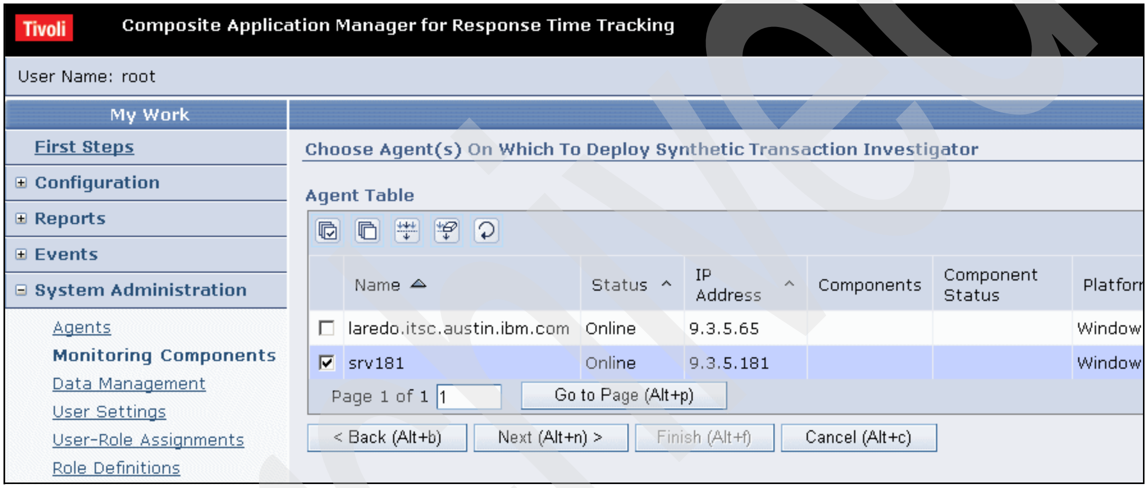


Figure 5-52 Select the agent to deploy STI

2. After deploying the STI component, the agent reboots. Verify the STI deployment by looking at the Status field on the **System Administration** → **Agents** panel.

5.5.8 Deploying the Web Response Monitor component

The Web Response Monitor monitoring component measures the performance of Web-based applications. It provides response time and other performance data, and tracks navigation paths and usage behavior for Web pages and embedded objects in Web pages, such as graphics files.

To deploy the Web Response Monitor component:

1. Before deploying the Web Response Monitor component, install the Network Monitor Driver:
   a. Open Network Connections.
   b. In Network Connections, click **Local Area Connection**.
   c. Select **File** → **Properties**.
   d. In the Local Area Connection Properties window, click **Install**.
   e. In the Select Network Component Type window, click **Protocol**, and then click **Add**.
   f. In the Select Network Protocol window, click **Network Monitor Driver**, and then click **OK**.

   After installing the Network monitor driver, it is listed, as shown in Figure 5-53.

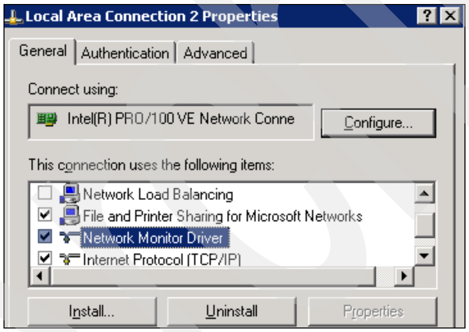


Figure 5-53 Network Monitor Driver installation

2. To deploy the Web Response Monitor monitoring components, select **System Administration → Monitoring Components**. Select **Web Response Monitor** from the monitoring components table, and choose **Deploy Monitoring Component** from the menu. Click **Go**. Select the agent on which to deploy the Web Response Monitor component from the agent table, as shown in Figure 5-54, and click **Next**.

> **Note:** The Web Response Monitor component runs on a management agent on the Web server machine. You can deploy on the following Web server machines:
>
> - IBM HTTP Server
> - Microsoft Internet Information Server (IIS)
> - Sun Java Web Server™
>
> We deploy the Web Response Monitor component on the quality of service component, as shown in Figure 5-54 on page 267. It is a reverse proxy that runs within IBM HTTP Server, Version 1.3.26.1.

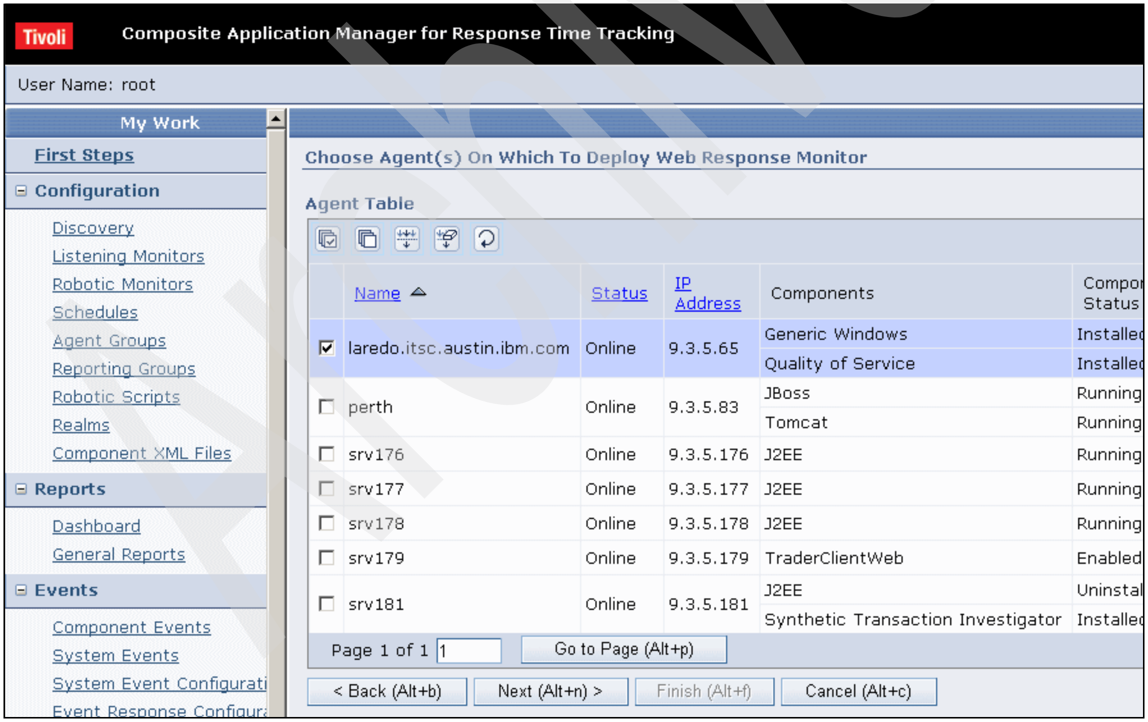


Figure 5-54 Installing Web Response Monitor on Quality of Service component

3. To deploy the Web Response Monitor monitoring component, enter the required information in the fields, as shown in Figure 5-55, and click **Finish**.

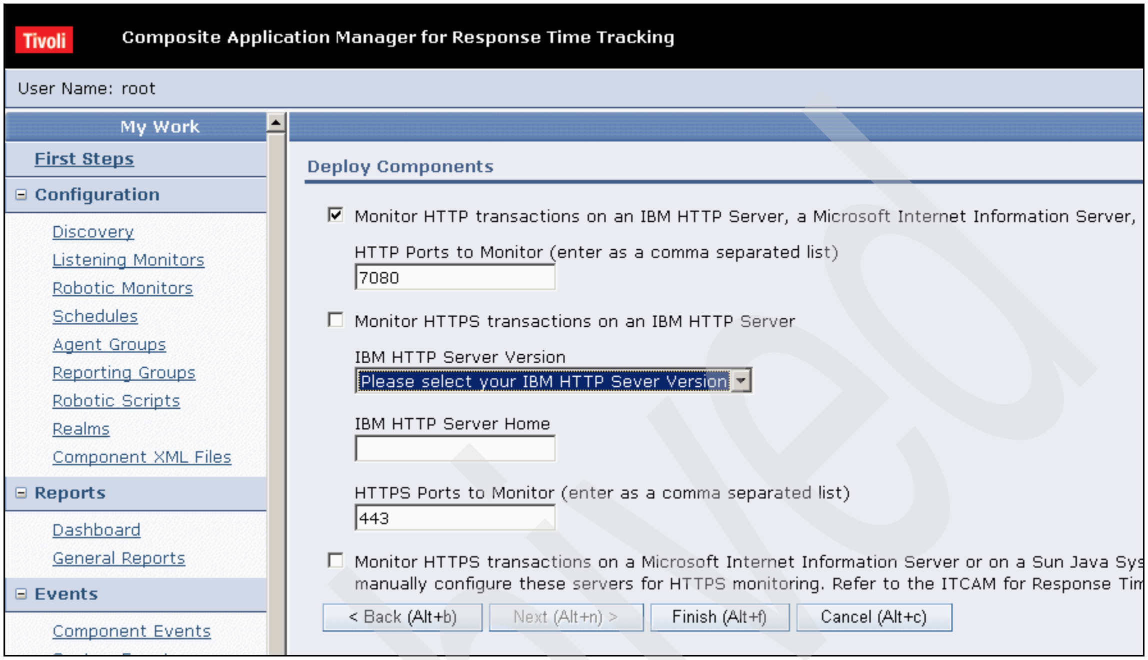


Figure 5-55 Required information to deploy Web Response Monitor

4. After deploying the Web Response Monitor, verify the Web Response Monitor component deployment by looking at the Status field on the **System Administration** → **Agents** panel.

5.5.9 Deploying the Generic Playback component

The Generic Playback component runs on a regular schedule any command on a management agent and collects timing information for the executing command.

To deploy the Generic Playback monitoring component:

1. Select **System Administration** → **Monitoring Components**. Select **Generic Playback** from the monitoring components table. Choose **Deploy Monitoring Component** from the menu, as shown in Figure 5-56, and click **Go**. Select the agent on which to deploy the Generic Playback component from the agent table and click **Next**. To deploy the component, click **Finish**.

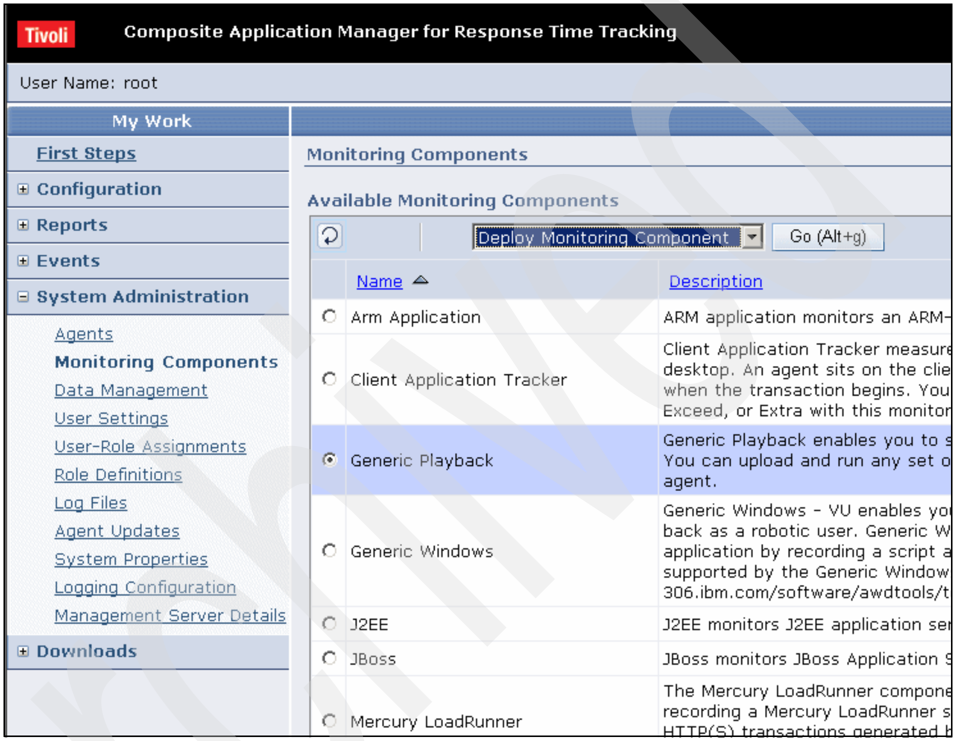


Figure 5-56 Deploying Generic Playback

2. After deploying the Generic Playback, verify the Generic Playback deployment by looking at the Status field on the **System Administration** → **Agents** panel.

5.5.10 Deploying the ARM Application component

The ARM Application component monitors the performance of ARM-instrumented applications. The applications can be from ISVs or custom in-house applications. Some commercial applications already contain ARM instrumentation, for example, WebSphere Application Server, Apache Web server, IBM HTTP Server, DB2 Universal Database™, Siebel, Baan ERP, and SAS.

To deploy the ARM Application monitoring component:

1. Select **System Administration** → **Monitoring Components**. Select **ARM Application** from the monitoring components table. Choose **Deploy Monitoring Component** from the menu and click **Go**. Select the agent on which to deploy the ARM Application component from the agent table and click **Next**. To enable the ARM Application, enter the required information, as shown in Figure 5-57, and click **Finish**.

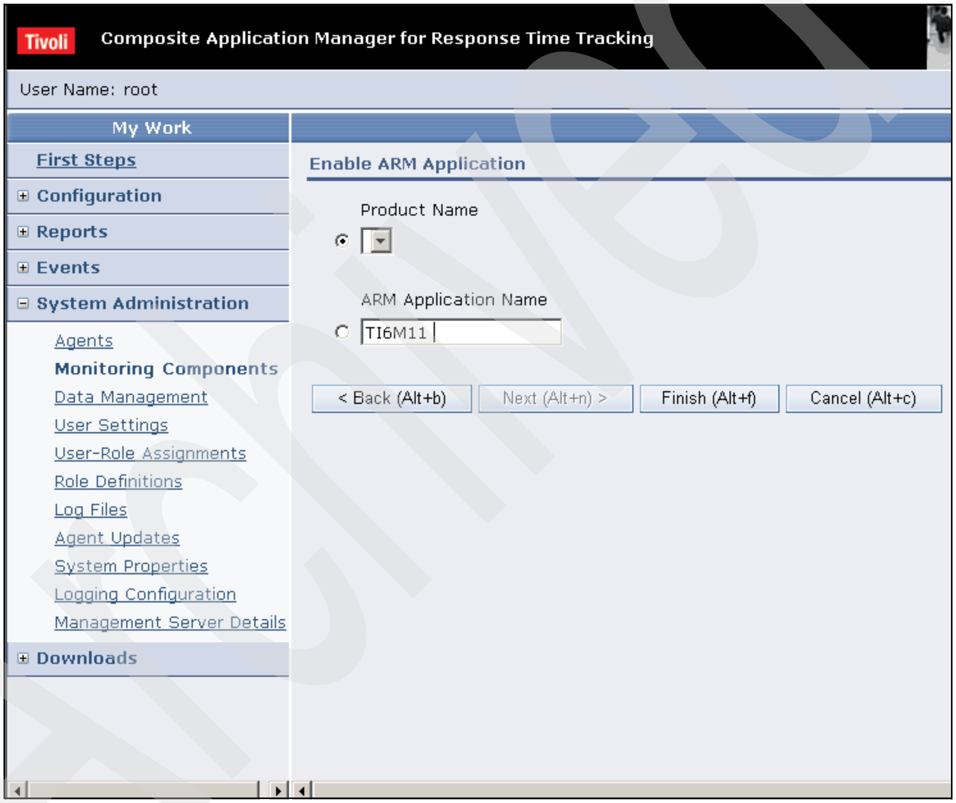


Figure 5-57 Deploying the ARM Application

2. After deploying the ARM Application component, verify the ARM Application deployment by looking at the Status field on the **System Administration** → **Agents** panel.

5.6 Initial configuration and usage

After performing the setup, perform most of the initial configuration tasks. The tasks are:

- Defining schedules, as described in 5.6.1, “Schedules” on page 271
- Defining the reporting group, as explained in 5.6.2, “Reporting groups” on page 274
- Defining the agent group, as shown in 5.6.3, “Agent groups” on page 275
- Defining the user role assignment, as discussed in 5.6.4, “Assigning user roles” on page 278

5.6.1 Schedules

Use the schedules to define when discovery, listening, and robotic monitors run. ITCAM for Response Time Tracking uses two types of schedules:

- Discovery and listening monitor schedules have start times and stop times, so you can specify when the monitor runs. A discovery and listening schedule can run continuously.
- Robotic monitor schedules for robotic transactions also have start and stop times. You specify how many times the monitor runs between the start and stop times.

The ITCAM for Response Time Tracking V6.1 has default schedules (Figure 5-58):

- Default listening schedule: It runs now and forever.
- Default playback schedule: It runs every 15 minutes forever.

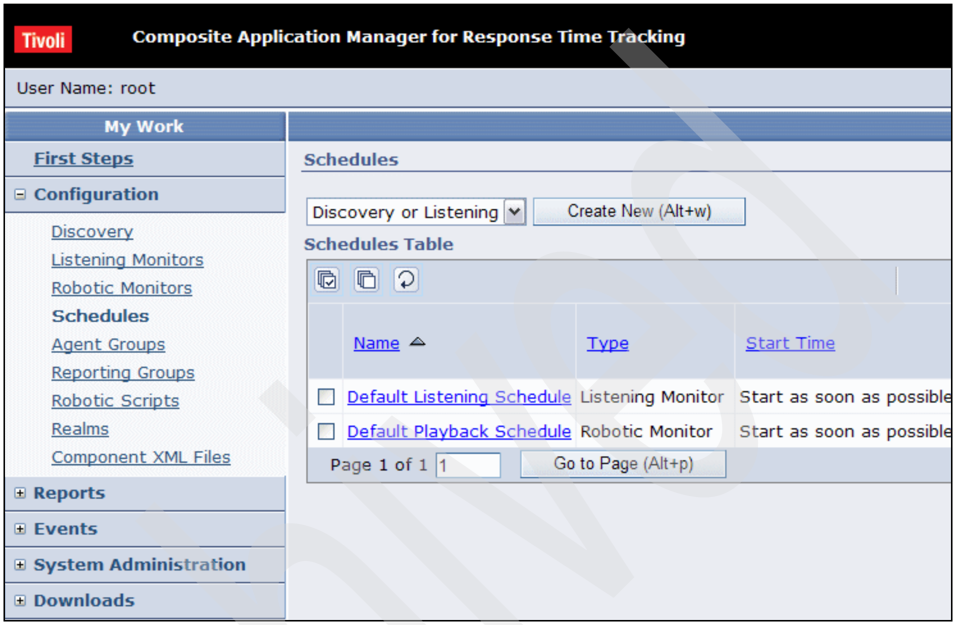


Figure 5-58   Default schedules

Design schedules generate the most meaningful data and minimize disruption to the running environment. Schedules also might be influenced by the service-level agreement monitoring requirement.

We are running in a test environment, so we have no real scheduling concerns such as peak time, prime time, and other transaction volume fluctuations. We define the following schedules:

- Discovery and listening schedules:
  - Workdays: runs 24 hours Monday through Friday
  - Workhours: runs 9 a.m. to 6 p.m. Monday through Friday
  - Weekend: runs 24 hours Saturday through Sunday
  - Discovery: runs every day 8 a.m to 9 p.m
- For the playback schedules, we wanted to capture transaction and response time trends. Therefore, we leave the schedule running all the time except for

scheduled maintenance. In our environment, we define continuous schedules with intervals, depending on the need of the application:

– Every_30_min
– Every_1_hour

Create schedules by selecting **Configuration** → **Schedules**. Figure 5-59 shows the window for defining discovery and listening monitor schedules.

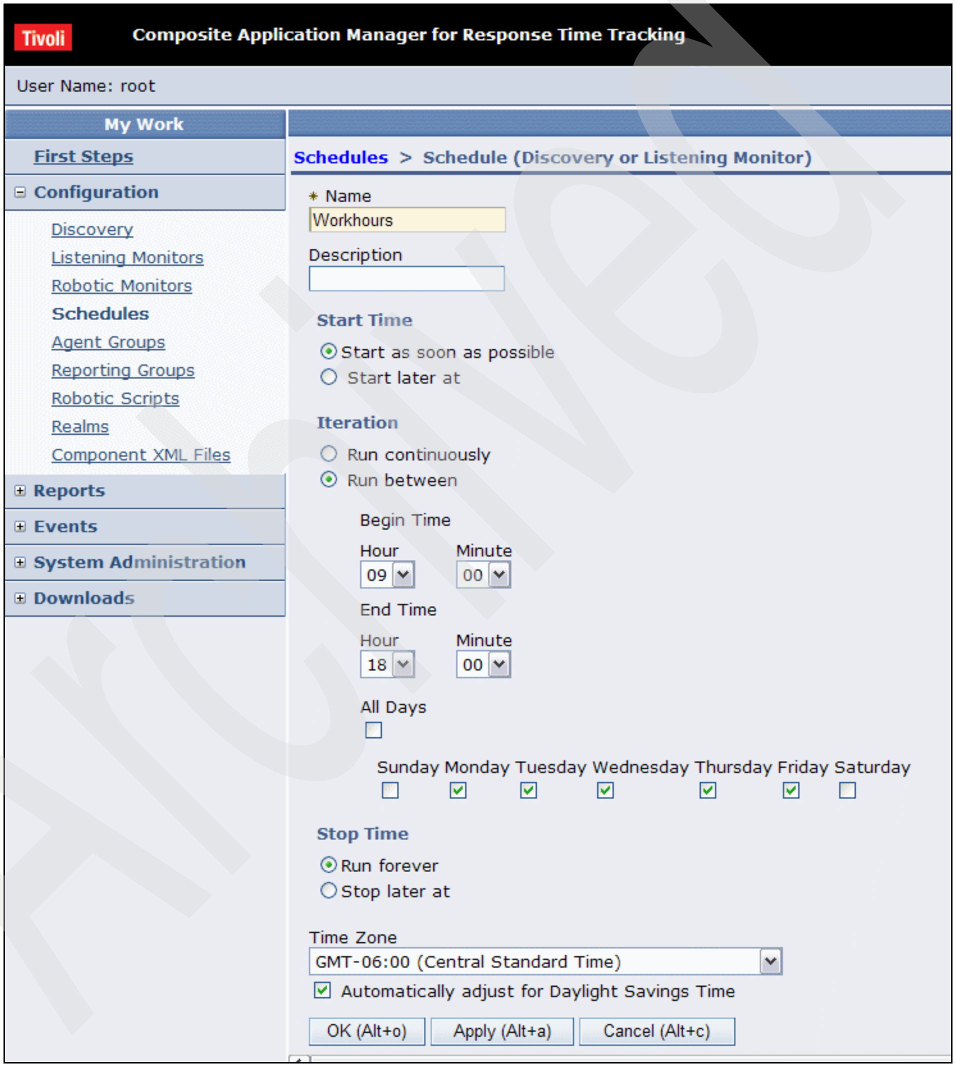


Figure 5-59 Configure discovery and listening schedules

Define a robotic monitor schedule using the window shown in Figure 5-60.

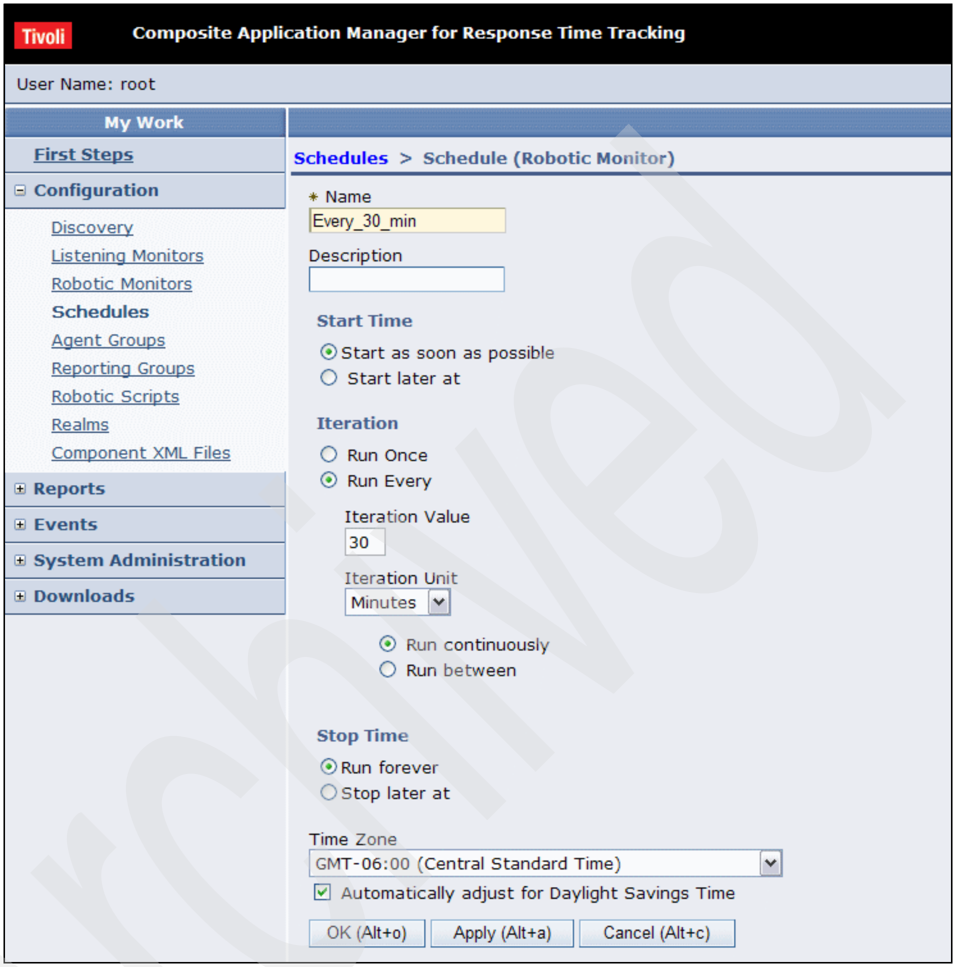


Figure 5-60 Configure a robotic monitor schedule

5.6.2 Reporting groups

The reporting groups group reports for monitors by function or business use, such as by site or applicaiton. Reporting groups enable you to group monitors by one of three types: application, customer, or location. For example, if you have a service-level agreement for a customer called BookSeller, you might create a reporting group called MonitorBookSeller that is of type customer. Then you can know whether you are meeting the requirements for BookSeller. Every monitor must belong to at least one reporting group and can belong to multiple reporting groups. In our environment, we have the trader application, which we explain in

Appendix A, “Trader application description” on page 619. For our monitoring of the application, we define these reporting groups by application:

TraderWeb We define that all reports must belong to this group for an overall view of the enterprise. This reporting group also makes a global operator easier to define.

Trader_DB2_Appl Reports the DB2-based transaction performance.

Trader_CICS_Appl Reports the CICS-based transaction performance.

To define the reporting group, select **Configuration** → **Reporting Groups**. Define a new reporting group by clicking **Create New**. A reporting group has the name, description field, and select type in drop-down list. Figure 5-61 shows our completed reporting groups list.

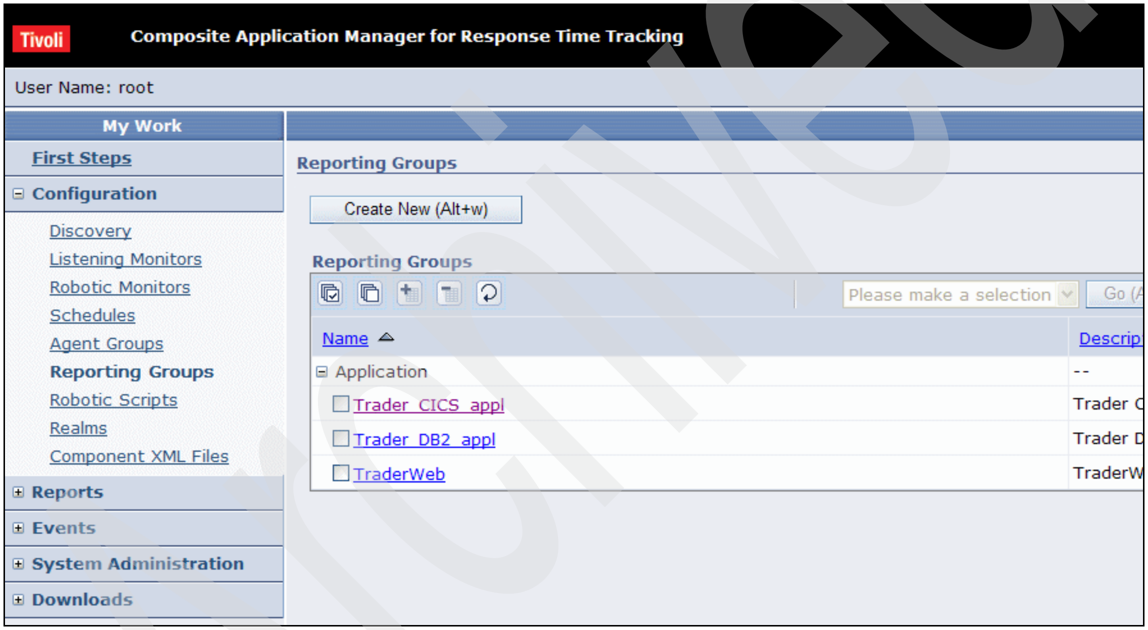


Figure 5-61 Reporting groups list

5.6.3 Agent groups

The agent group is a group of management agents that run the same monitor or monitoring components. An agent group can be used for authorization purposes. Each management agent is associated with one or more listening and playback components. If a transaction spans multiple management agents and you want to perform thresholding on subtransactions, you must include all of the management agents in the same group. For example, if you create an STI monitor and know that the played-back transaction runs through a Web server monitored by the quality of service component, include the agent that is running the quality of service component. Otherwise, you cannot establish thresholds for the part of the played-back transaction that runs through the Web server. You can obtain performance data, but thresholding is not enabled.

ITCAM for Response Time Tracking provides the default agent groups when you deploy a new monitoring component type automatically, as shown in Figure 5-62.

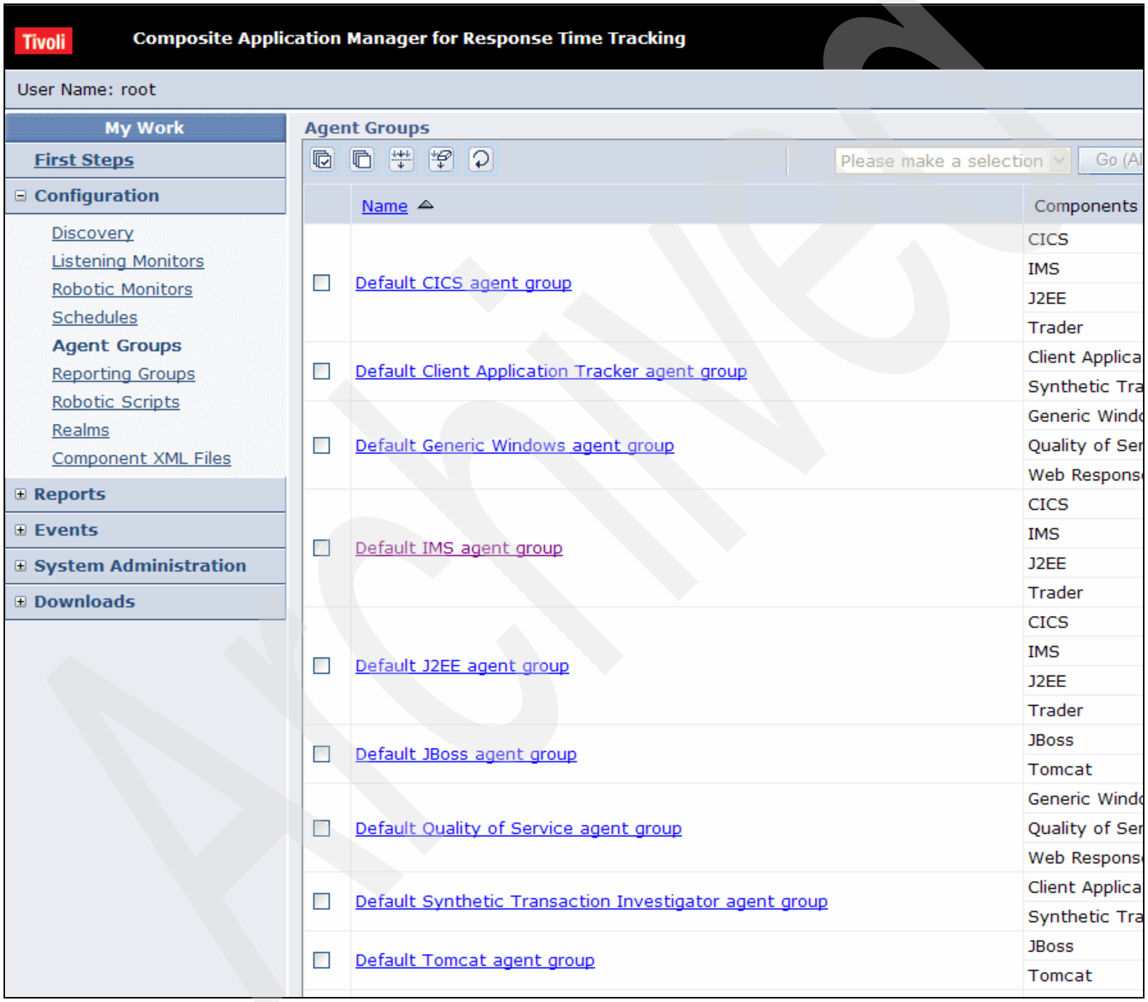


Figure 5-62   Default agent groups list

In our environment, we define the following additional groups:

STI_QoS_J2EE_MA All management agents that run the trader Web Client using STI, QoS, and J2EE

Robot_MA All management agents that run the trader application and Rational Robot

JBoss_Tomcat_MA All management agents that run JBoss and Tomcat

To define an agent group, select **Configuration** → **Agent groups**, click **Create New**, and specify agents for the group, as shown in Figure 5-63.

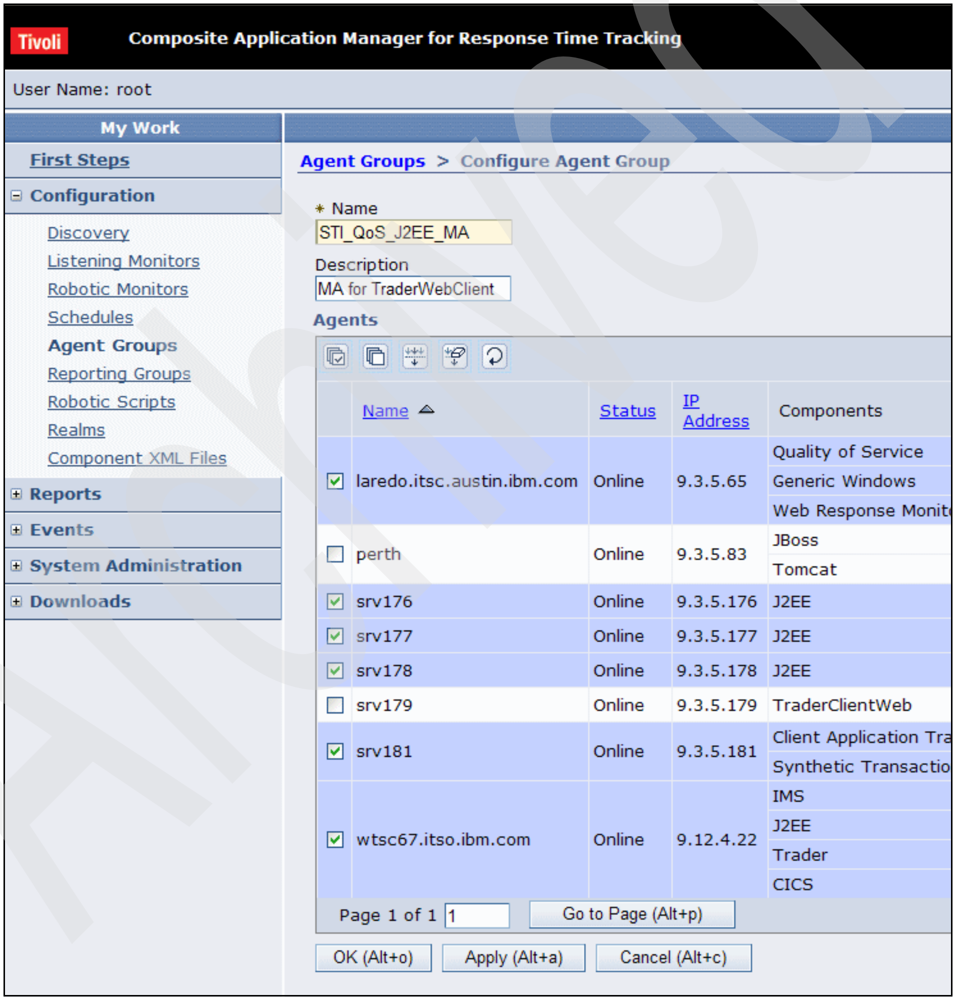


Figure 5-63 Configure agent group

5.6.4 Assigning user roles

This section provides information about how to assign users to preassigned roles or create granular, restricted profiles for different users so that they can only see and work on the resources that matter to them. If you provide service to competing groups, this ensures that they cannot see each other's data. Users can have multiple roles. Each role has the following characteristics:

- Associated operations, each of which can have view or modify permission
- A set of reporting groups restricting the scope of the operations or unrestricted reporting groups
- A set of agent groups restricting the scope of the operations or unrestricted agent groups

The default role of admin can perform all the procedures. There must always be at least one user in the system that is assigned the admin role.

ITCAM for Response Time Tracking operators are assigned roles from the Web interface. These roles are not WebSphere roles, although users are still authenticated using WebSphere security and authentication mechanisms that can be based on OS users or LDAP users. To define users select **System Administration** → **User-Role Assignments**. Create new users by clicking **Add User**. Figure 5-64 shows the new user assignment window.

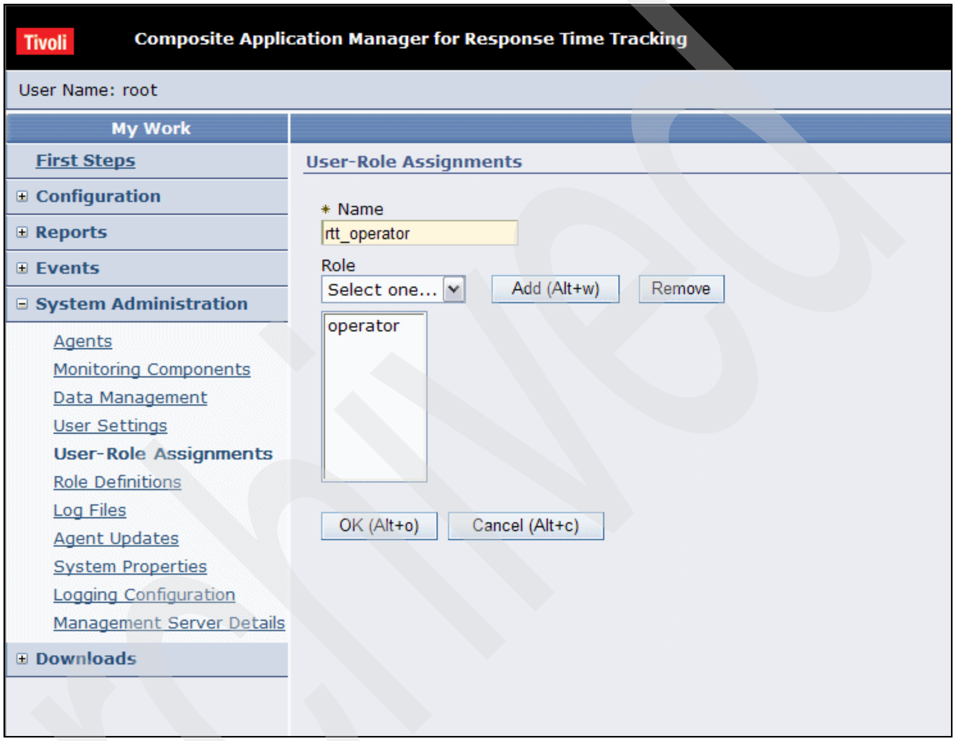


Figure 5-64 New user assignments

Table 5-4 shows the existing default roles.

Table 5-4 Default roles

Role	Default permissions for this role
admin	Allows the user to perform all possible actions on every resource. If you want to define custom roles, you must be assigned to this role.
agent	Can install or view management agents. This role cannot create monitors, schedules, and so on.
operator	Can view all reports and events.

Role	Default permissions for this role
tap	Has view capability for the DATA_QUERY permission - unrestricted resources. This role is provided to facilitate the authorization set up for the IBM Tivoli Enterprise Monitoring Agent user. The tap role can only retrieve data for the reporting groups and agent groups for which the user has been granted access.
user	Has view capability for all tasks with no restrictions.

You can create additional roles to segregate users based on reporting groups. Define roles using **System Administration** → **Role Definitions**, as shown in Figure 5-65.

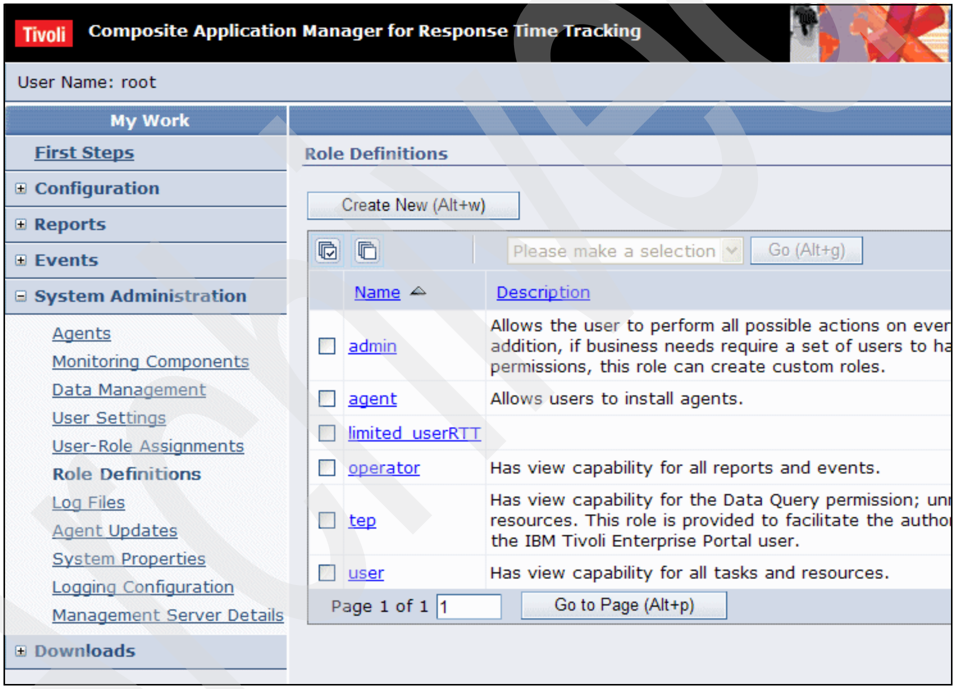


Figure 5-65 Role definitions

To create a new role, click **Create New**. For a new role, you must define the following elements:

- Task access: Set as shown in Figure 5-66.

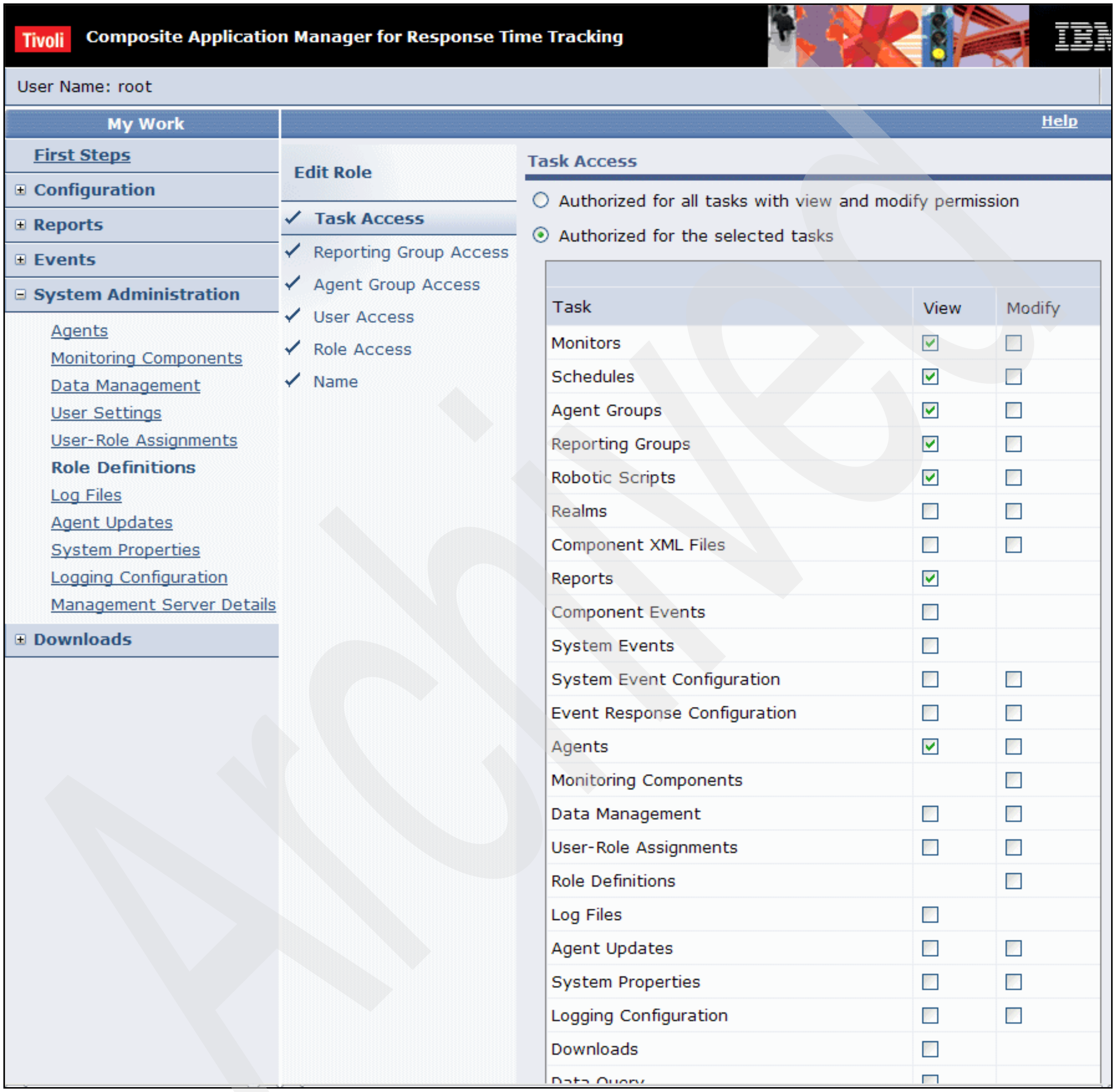


Figure 5-66 Task access

- Reporting group access: defines authorizations for the reporting group.
- Agent group access: defines authorizations for the agent group.
- User access: defines which users can be assigned to this role.
- Role access: defines which roles can manage this role.

Figure 5-67 shows our role definitions list with the new role.

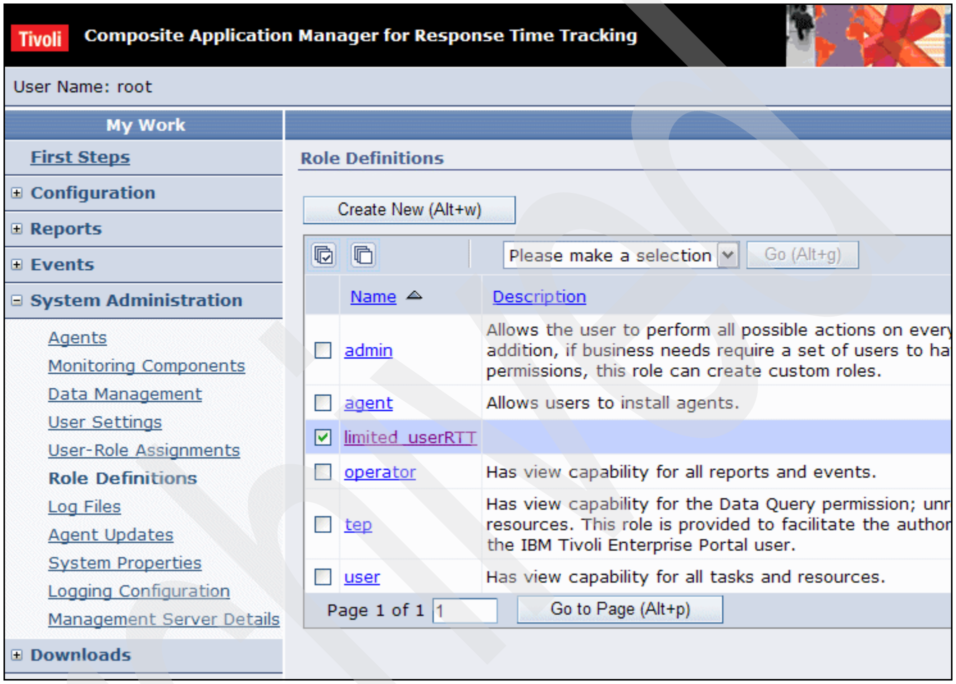


Figure 5-67 Role definitions list

6

ITCAM for Response Time Tracking usage

This chapter describes the basic usage for IBM Tivoli Composite Application Manager (ITCAM) for Response Time Tracking monitoring components and describes it as applied to our sample environment. This chapter includes the following topics:

6.1 Usage scenarios overview

We configured a distributed monitoring component. We want to manage its response time with ITCAM for Response Time Tracking. It can be monitored using several components, such as:

- Collecting existing response times from the J2EE application server by the J2EE monitoring component. It is collected by the J2EE instrumentation on the WebSphere Application Server. For CICS, this instrumentation information is from ITCAM for CICS Transactions. See 6.2, “J2EE monitoring” on page 285.
- Collecting Web-based transaction response time using the Web Response Monitor component. See 6.3, “Web Response Monitor” on page 292.
- Performing the sample application from a user site on scheduled intervals. These sample trader applications can be run using Rational Performance Tester. See 6.9, “Rational Performance Tester” on page 320.

> **Note:** As a general usage tip, ITCAM for Response Time Tracking uses the string .* wildcard, not just the asterisk character (*). A discovery will not discover transactions when a listening monitor is already in effect. Disable the discovery when the transactions have been discovered. A discovery or listening monitor will be effective if the transaction has been tagged with an ARM correlator from its upstream.

For additional monitoring components usage, we also perform monitoring for the following:

- The Rational Robot transaction response time with Generic Windows. See 6.4, “Rational Robot and Generic Windows” on page 296.
- JBoss and Tomcat on J2EE application transaction response time with JBoss and Tomcat monitoring components. See 6.5, “JBoss and Tomcat monitoring” on page 301.
- ARM instrumented application transaction response time with Application Response Measurement. See 6.6, “Application Response Measurement” on page 303.
- Lotus Notes client response time with Client Application Tracker using ETE™ Watch. See 6.7, “Client application tracker” on page 310.
- Command execution response time with Generic Playback. See 6.8, “Generic playback” on page 316.

We describe using Tivoli Enterprise Portal with ITCAM for Response Time Tracking in 6.10, “Working with Tivoli Enterprise Portal” on page 326.

6.2 J2EE monitoring

For getting the J2EE transaction response time component path, we configure the J2EE monitoring component. It provides the detailed response time information for the J2EE application. The J2EE monitor can be shown as Web services calls or servlet access. The Web-based trader application uses servlets in the TraderClientWeb application, while the Java-based trader application performs direct Web services calls.

6.2.1 J2EE discovery

We create the J2EE discovery for both servlets and Web services calls:

1. To configure the J2EE servlets discovery, select **Configuration → Discovery**. Choose **J2EE Servlet** from the menu and click **Create New**. Figure 6-1 shows the servlet discovery settings.

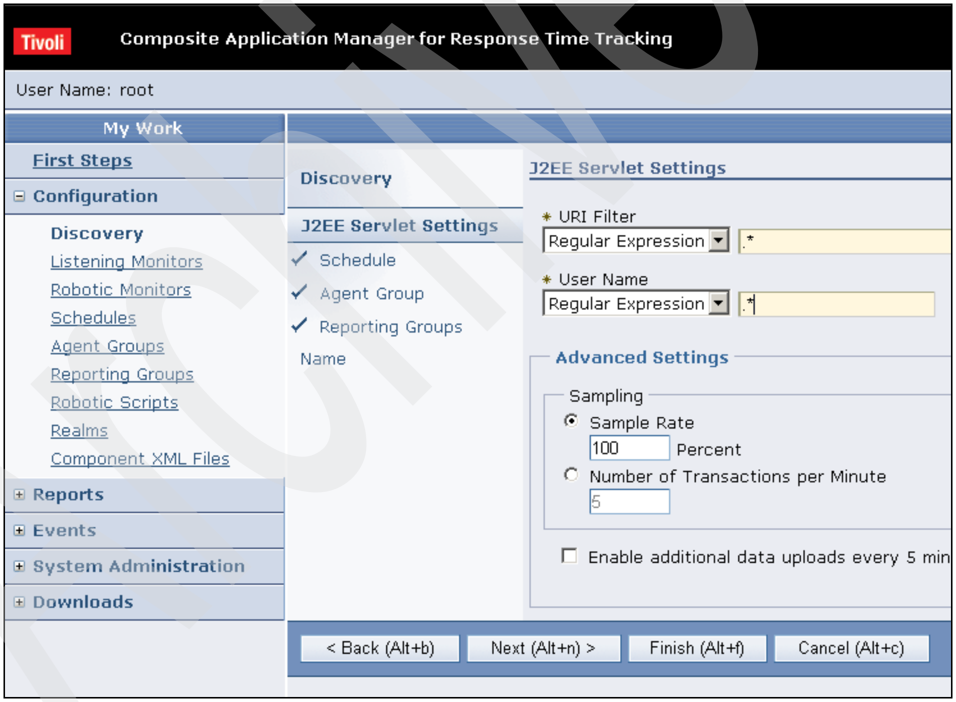


Figure 6-1 J2EE servlet discovery settings

2. We also create discovery for Web services that are called from the Java-based application. Figure 6-2 shows the Web services discovery settings.

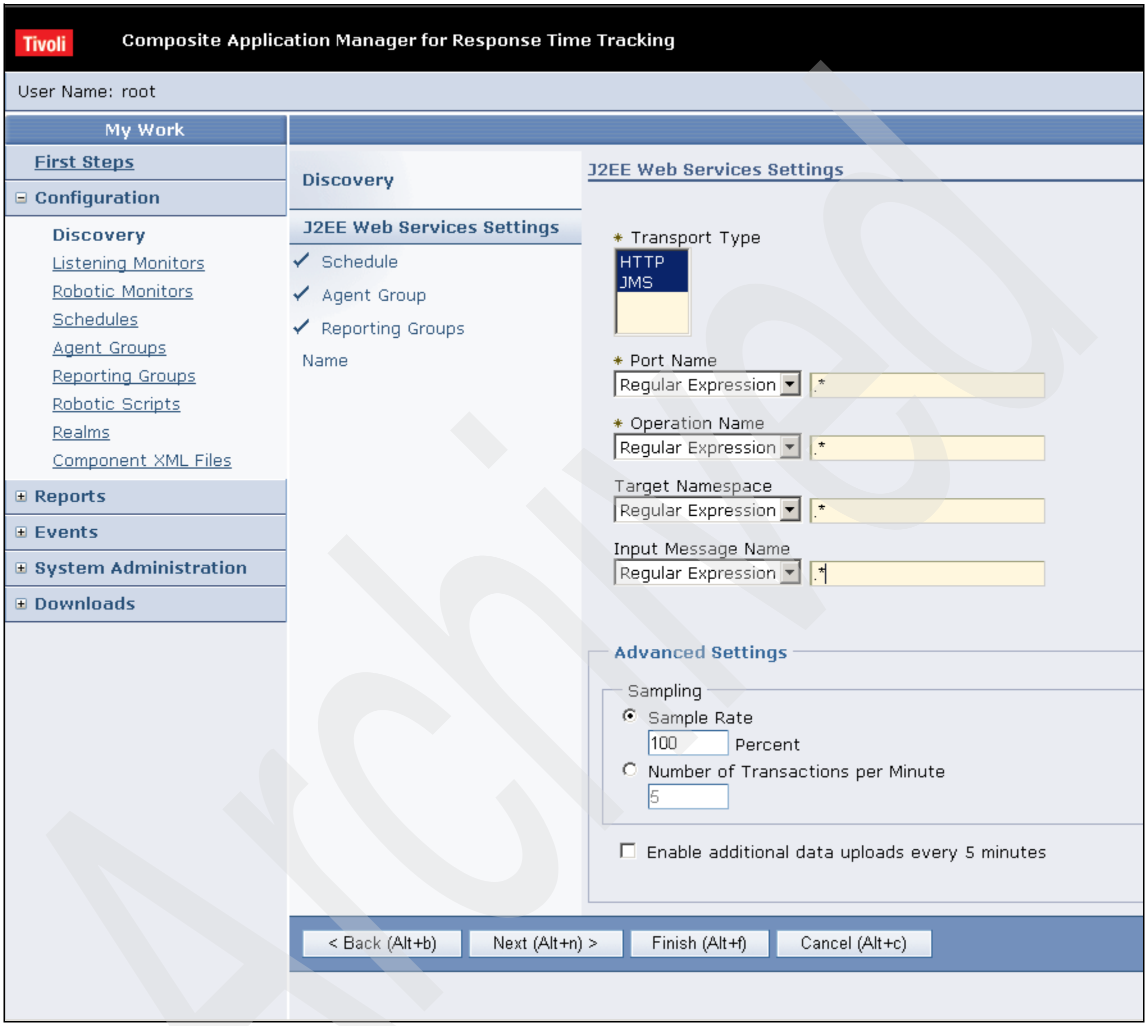


Figure 6-2 J2EE Web services discovery settings

3. After capturing some transactions, you can get the discovered transaction. Figure 6-3 shows the sample discovered the J2EE transactions.

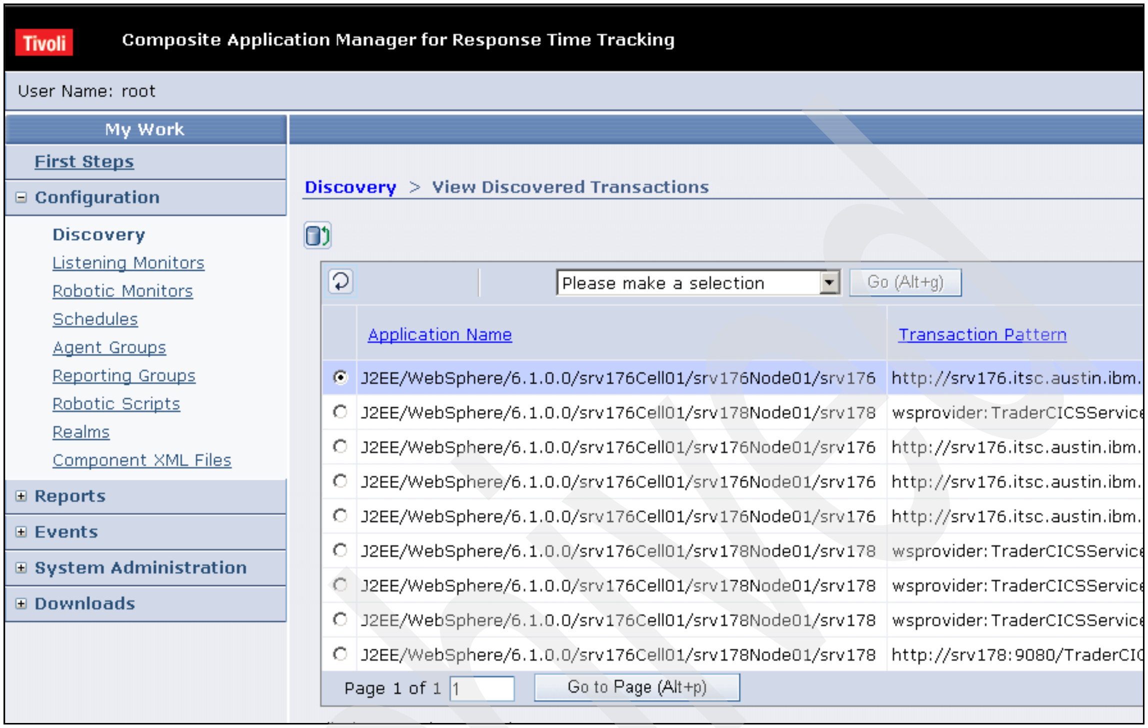


Figure 6-3 Discovered J2EE transactions

6.2.2 J2EE listening monitor

From the discovered J2EE transactions in Figure 6-3, we can define the J2EE listening monitor. We create the J2EE listening monitor for TraderCICSECIServlet and the discovered Web services.

To create the J2EE listening monitor, select the discovered servlet and choose **Create Listening Monitor From** from the menu. Click **Go**, and configure the J2EE servlet and Web services settings for the J2EE listening monitor, as shown in Figure 6-4.

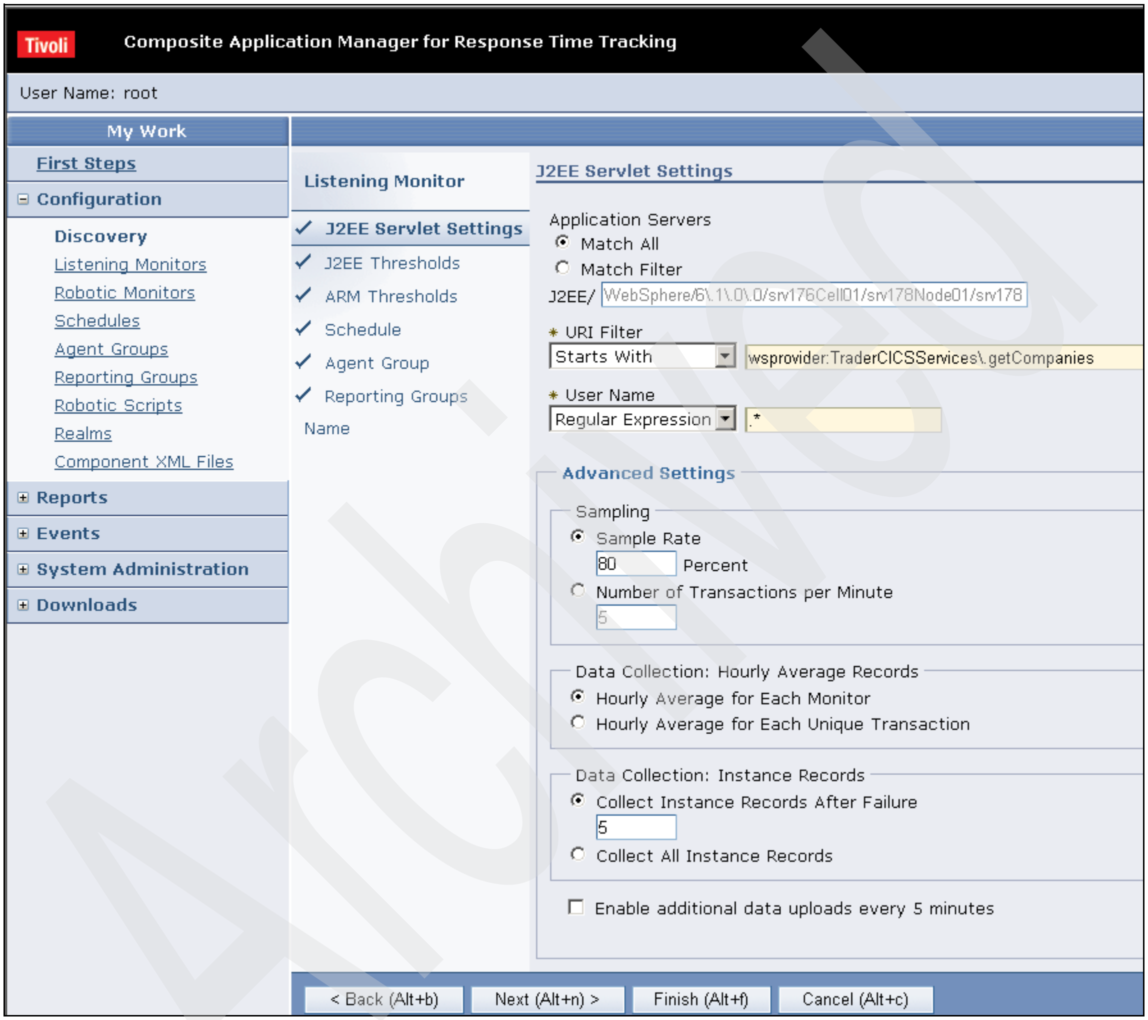


Figure 6-4 getCompanies Web services Listening Monitor settings

6.2.3 J2EE report

After the J2EE listening monitors are running for a while, we get the composite transaction report. We can see the performance of the J2EE applications using **Reports** → **Dashboard**. Figure 6-5 shows the J2EE composite transaction topology.

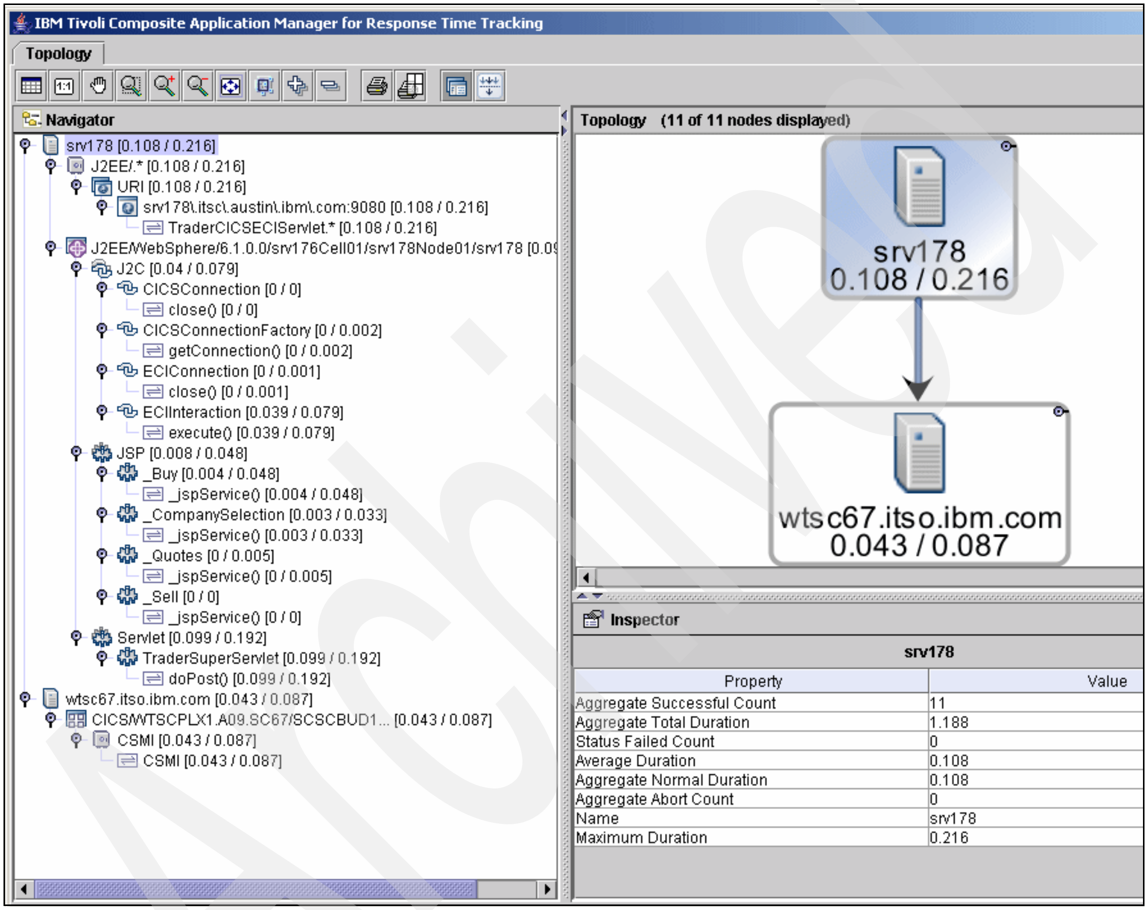


Figure 6-5 J2EE transaction node overview

From the transaction node, we can expand and analyze its behavior for each node. For the J2EE servlet in srv178, the transaction is broken down as shown in Figure 6-6.

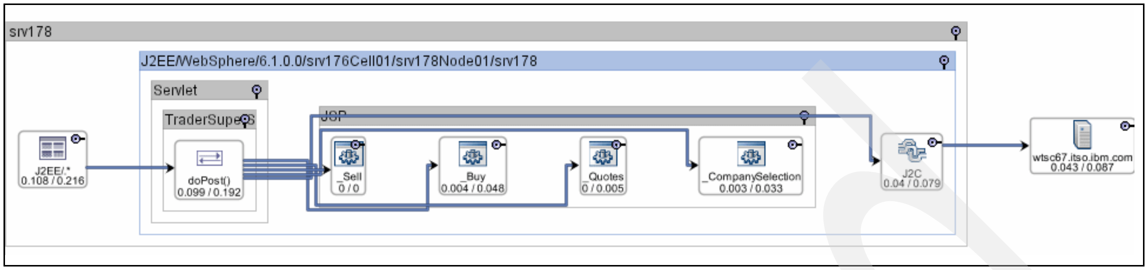


Figure 6-6 srv178 breakdown

For the Web services transaction in srv178, Figure 6-7 shows the transaction.

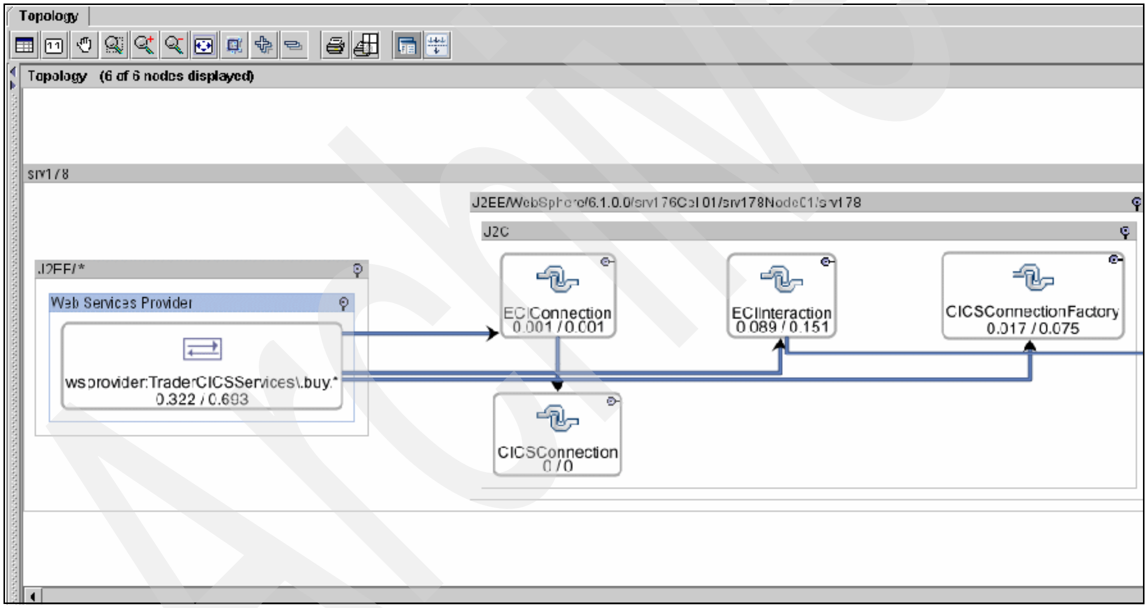


Figure 6-7 Web services Web application modules

The J2C structure can be further broken down depending whether it is an CICS-related connection, as shown in Figure 6-8.

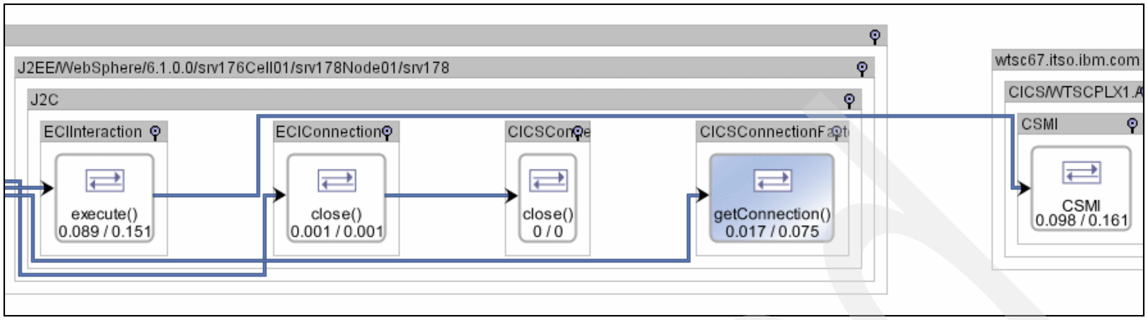


Figure 6-8 J2C break down

Figure 6-9 shows the transaction that is invoked on the z/OS side. For more detailed CICS configuration information, refer to Chapter 10, "Implementation of ITCAM products on z/OS" on page 521.

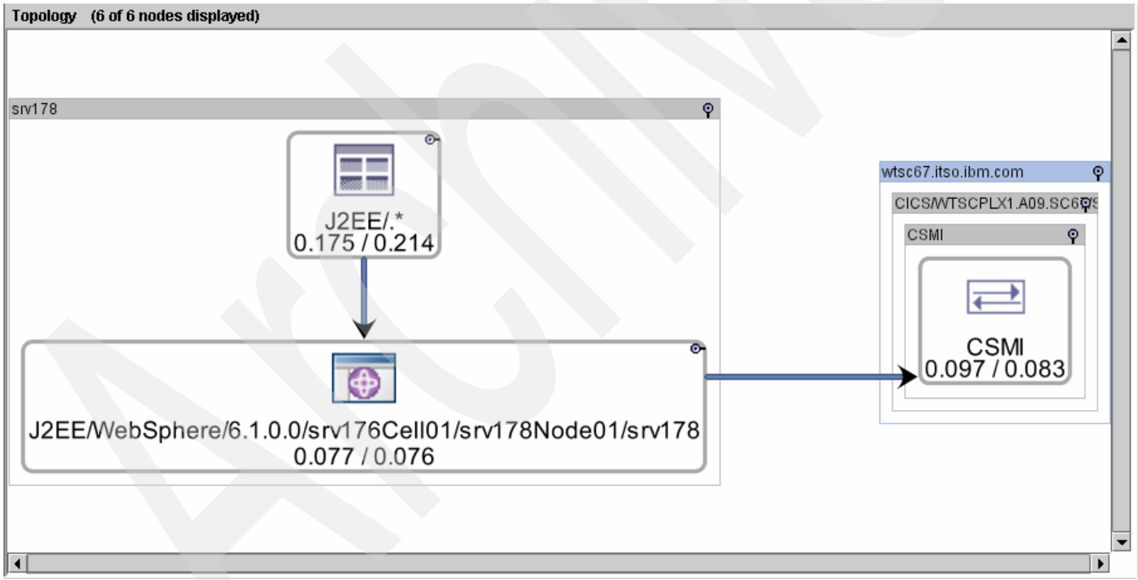


Figure 6-9 CICS transaction

6.3 Web Response Monitor

The Web Response Monitor (WRM) is a new monitoring component in this release. It monitors the performance of Web-based applications. The Web Response Monitor monitoring component resembles the quality of service monitoring component. These are different methods of monitoring. The Web Response Monitor calculates the response time for Web transactions that use the TCP/IP protocol. For more information, refer to *ITCAM for Response Time Tracking Administrator's Guide*, SC32-1905.

6.3.1 Web Response Monitor discovery

We deployed the Web Response Monitor to the quality of service monitoring component. It runs within IBM HTTP Server V1.3.26.1. We did not configure a separate Web server for the Web Response Monitor.

To monitor with the Web Response Monitor, you need to create the discovery to determine what transactions are in the Web server.

To configure the Web Response Monitor discovery:

1. Select **Configuration** → **Discovery**. Choose **Web Response Monitor** from the menu, as shown in Figure 6-10, and click **Create New**.

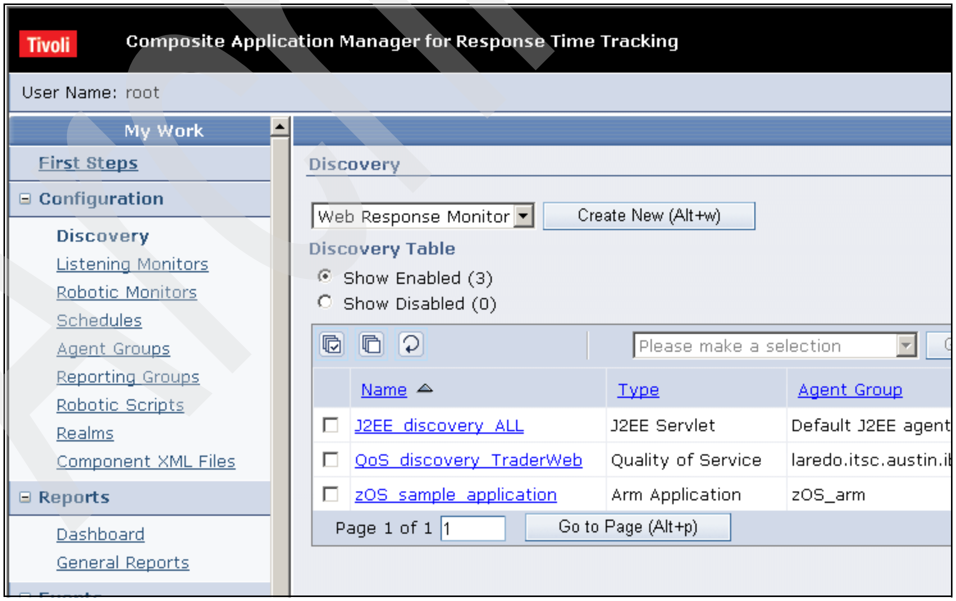


Figure 6-10 Web Response Monitor discovery

2. Enter the WRM settings to discovery the transaction in the Web server, and click **Next**. Figure 6-11 shows the WRM discovery settings.

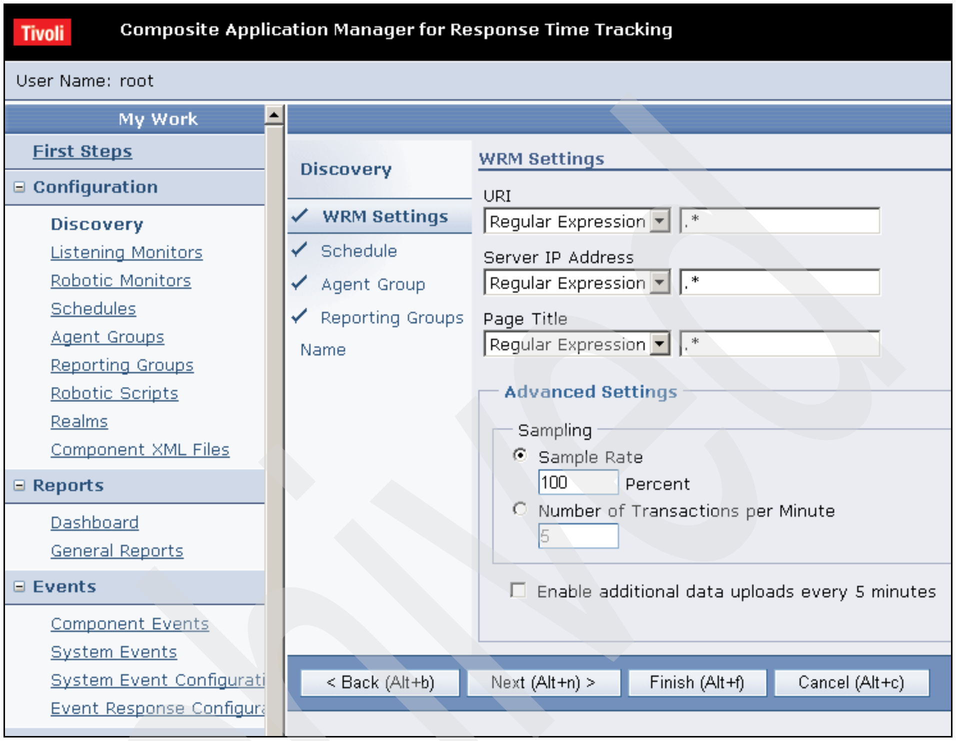


Figure 6-11 Web Response Monitor discovery settings

3. After the discovery monitor configuration, we generated the trader application. After capturing some transactions, you get the discovered transactions, as shown in Figure 6-12.

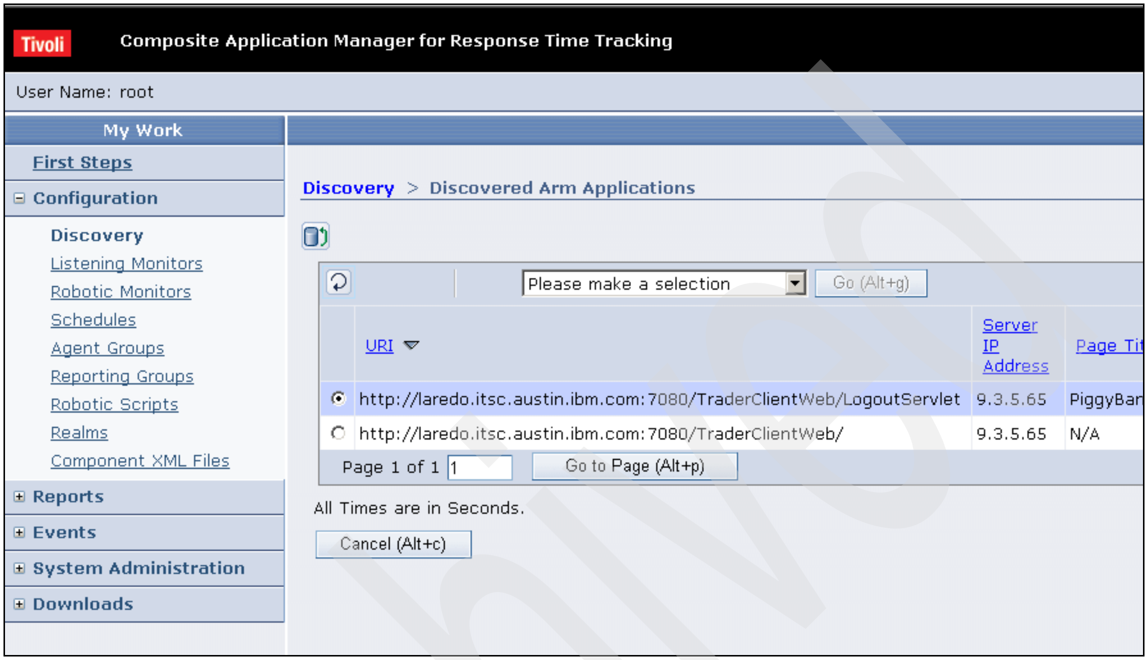


Figure 6-12 Discovered Web transactions by the Web Response Monitor

6.3.2 Web Response Monitor listening monitor

From the discovered Web transactions shown in Figure 6-12, we define the listening monitor.

To create the Web Response Monitor listening monitor, select the URL of the discovered Web transaction and choose **Create Listening Monitor From** from the menu. Click **Go**. Configure the WRM settings for the listening monitor, as shown in Figure 6-13.

Figure 6-13 Web Response Monitor listening monitor settings

6.3.3 Web Response Monitor report

After the listening monitor is running, you can see the performance of Web-based applications using **Reports** → **Dashboard**. Figure 6-14 shows the transaction topology, separating the load time, resolve time, and overlap time.

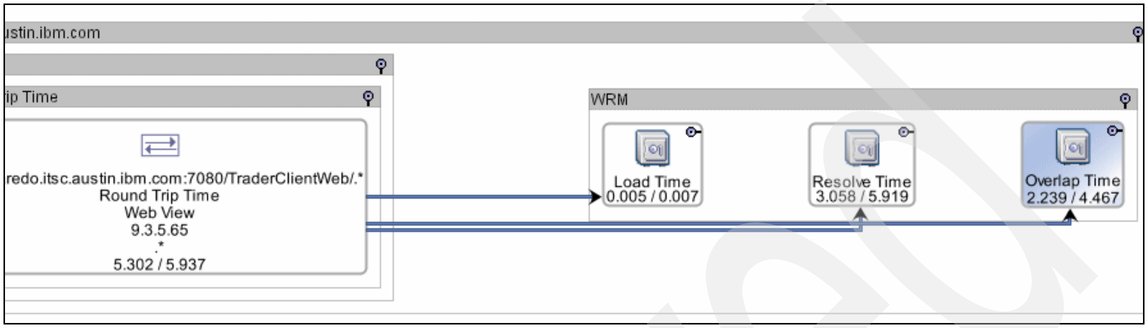


Figure 6-14 Transaction topology by Web Response Monitor listening monitor

6.4 Rational Robot and Generic Windows

The Generic Windows (GenWin) component runs with Rational Robot to monitor Windows-based applications. To monitor with the Generic Windows component, you need to create a robot script first.

6.4.1 Creating a Robot script

We describe the Rational Robot record procedure with the trader application.

Perform the following steps:

1. On the Rational Robot installed machine, click **Start** → **All Programs** → **Rational Software** → **Rational Robot**. Type the user name, password, and project in the Rational Test Login window, which are set in 5.5.2, “Installing the Rational Robot and Generic Windows components” on page 252.

2. Click the **Record GUI Script** button (Ctrl+R) to open the Record GUI window. Define the name and then click **OK**, as shown in Figure 6-15.

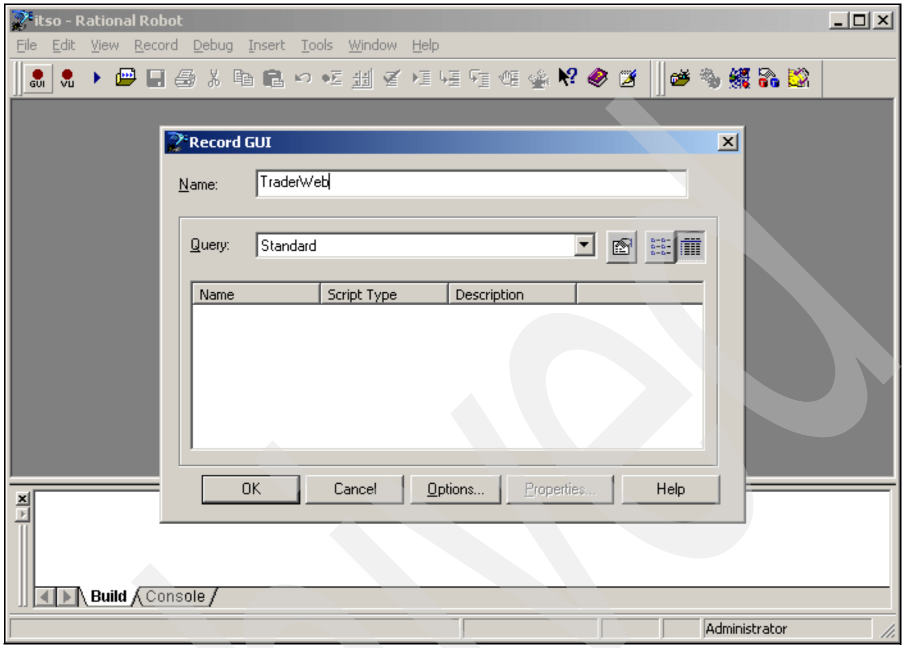


Figure 6-15 Starting record GUI

3. We record the trader application transaction using Microsoft Internet Explorer. Click **Stop**. The script is available to edit, as shown in Figure 6-16.

> **Note:** We use the GUI Robot only for Microsoft Internet Explorer. For more information about Rational Robot usage, refer to *ITCAM for Response Time Tracking Administrator’s Guide*, SC32-1905.

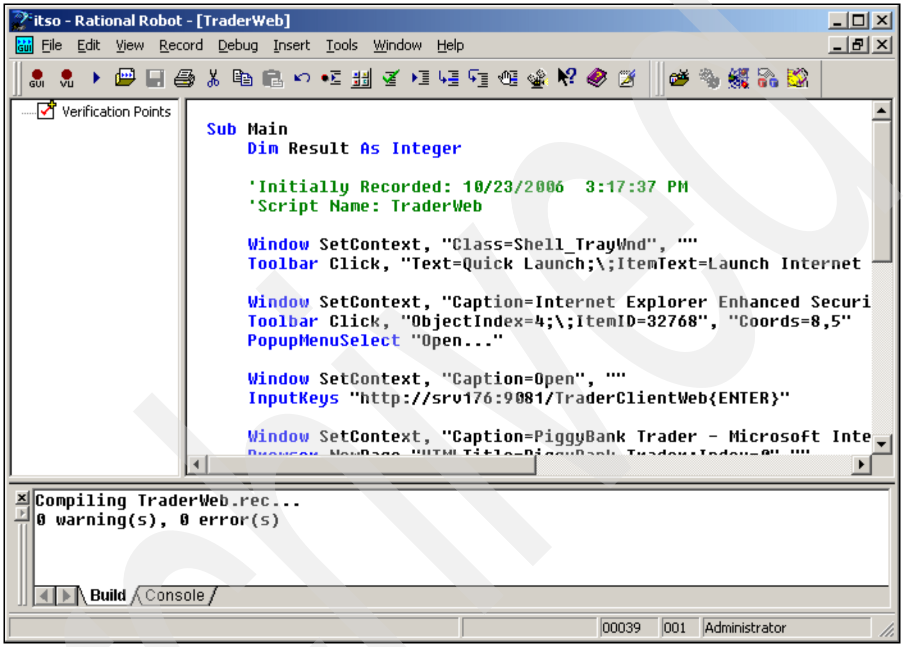


Figure 6-16 Created transaction script by Rational Robot

4. To use the Rational Robot script on the management server, upload the recorded transaction. Select **Configuration** → **Robotic Scripts** on the management server console. Select **Generic Windows - GUI** from the menu and click **Create New**. Use this with the Java Web start upload or manual script upload. We use the manual script upload, as shown in Figure 6-17.

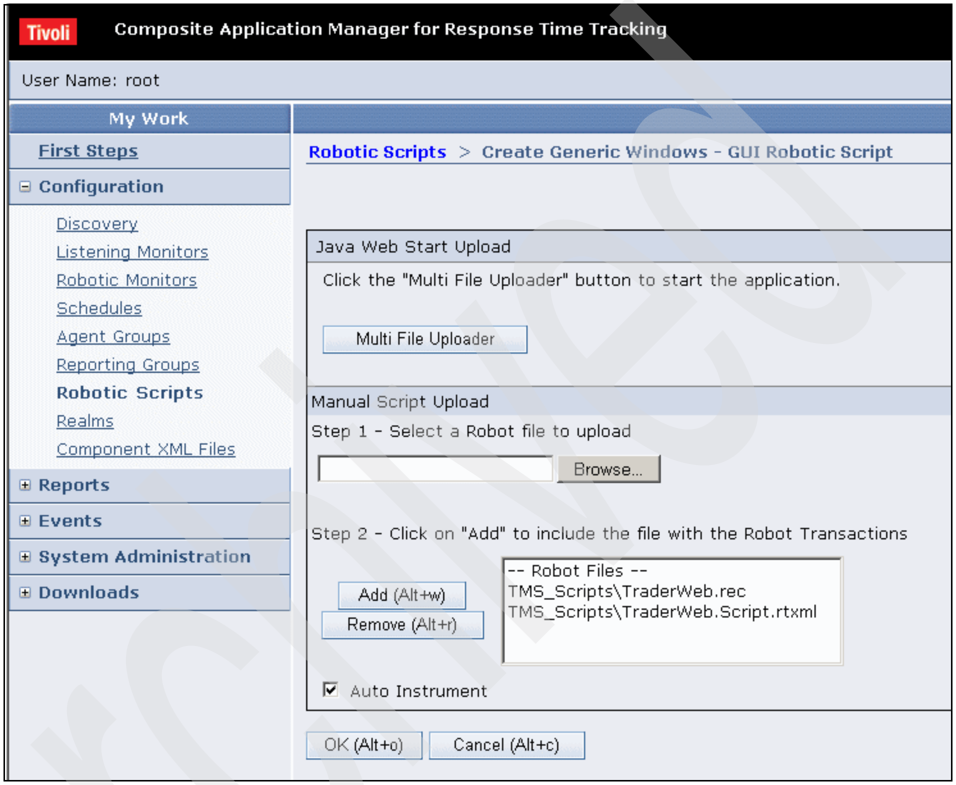


Figure 6-17 Uploading the recorded transaction by Rational Robot

6.4.2 Generic Windows robotic monitor and report

After creating and uploading the Rational Robot script, it is used by the robotic monitor for the transaction monitoring.

To monitor with Generic Windows:

1. Select **Configuration** → **Robotic Monitors**. Choose **Generic Windows - GUI** from the menu, and then click **Create New**. Figure 6-18 shows the Generic Windows - GUI Settings.

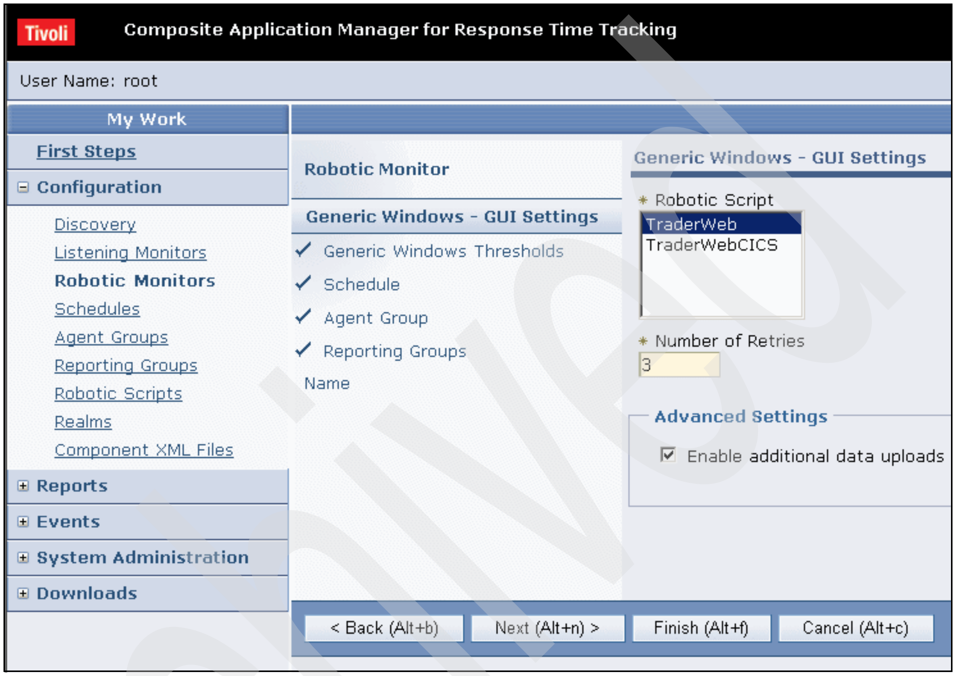


Figure 6-18 Generic Windows - GUI Robotic monitor settings

2. We created the Generic Windows robotic monitors using the default playback schedule. After the robotic monitor is running, you can see the report using **Reports** → **Dashboard**, as shown in Figure 6-19.

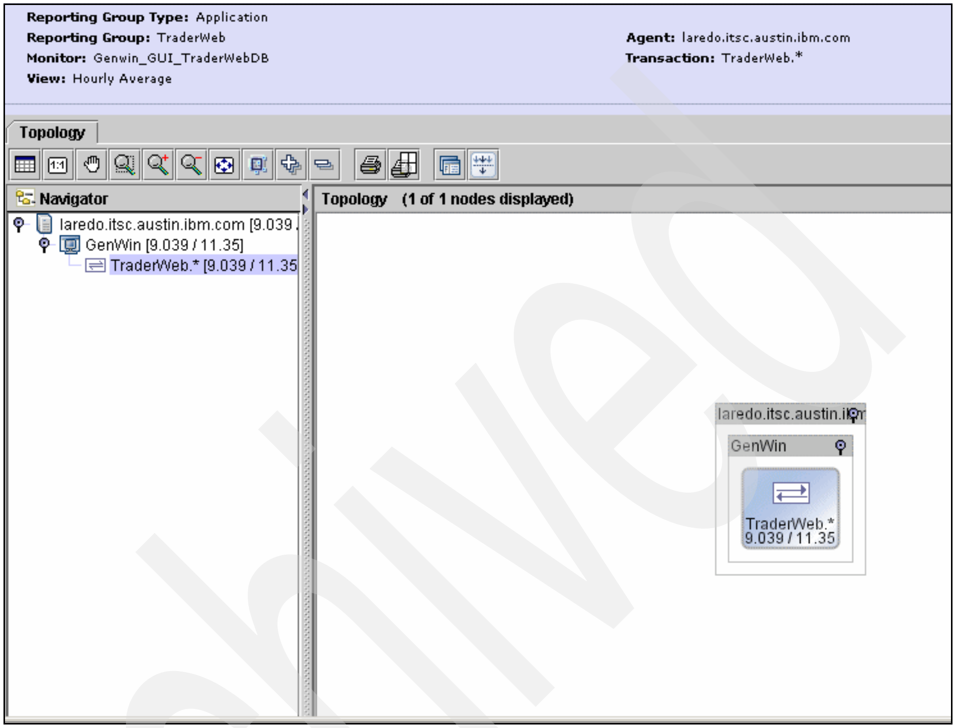


Figure 6-19 Sample transaction topology report for the GenWin robotic monitor

6.5 JBoss and Tomcat monitoring

JBoss and Tomcat are new monitoring components in this release. These components collect ARM data generated by J2EE applications. The monitoring process is similar to the J2EE monitoring component.

To monitor J2EE transactions on JBoss and Tomcat, we discovered the transaction using **Configuration** → **Discovery**. After capturing the transactions, we configured the listening monitor from the discovered list using **Create Listening Monitor From**. We used the sample J2EE application to monitor JBoss and Tomcat J2EE applications.

Figure 6-20 shows the transaction topology report from running on JBoss.

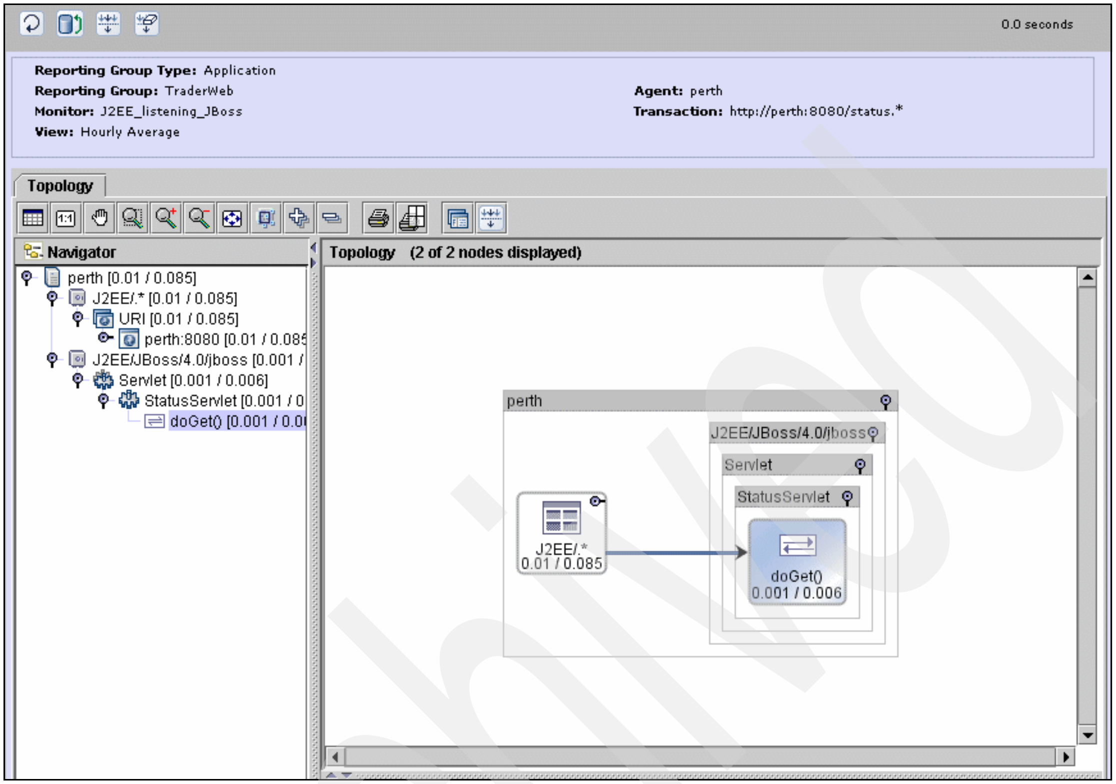


Figure 6-20 J2EE transaction topology of JBoss

Figure 6-21 shows the transaction topology report from running a Tomcat sample J2EE application.

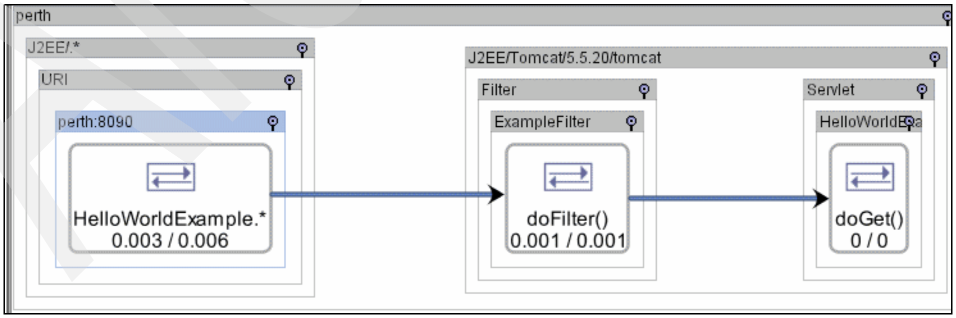


Figure 6-21 J2EE transaction topology of Tomcat

6.6 Application Response Measurement

The Application Response Measurement (ARM) component monitors the performance of ARM 2.0 or 4.0-instrumented applications. This section describes the ARM instrumentation using the sample Java application.

6.6.1 Instrumenting the sample application

We used the sample Java application to monitor using ARM. The sample application is simple: Click the **Simple** or **Complex** button, and then it returns the elapsed time, as shown in Figure 6-22.

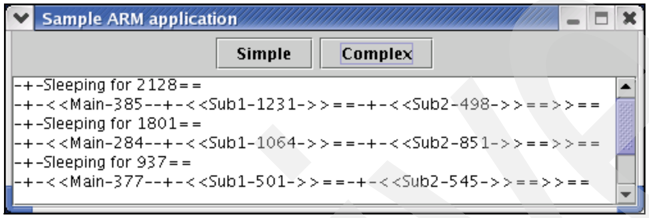


Figure 6-22 Sample Java application for ARM monitor

Perform the following steps:

1. For our instrumentation, we only need to import the org.opengroup.arm40.transaction package, as shown in Example 6-1.

Example 6-1 Importing the ARM instrumentation

```
import org.opengroup.arm40.transaction.*;
```

2. Declare the ARM instrumentation variables, as shown in Example 6-2.

Example 6-2 ARM instrumentation variables

```
public class ARMAppsMain extends JApplet implements ActionListener {
   private static final long serialVersionUID = 1;
   private Random r;
   private static boolean armOn;
   public static ArmTransactionFactory armTranFactory   = null;
   static {
      try {
         String tranFactoryName =
"com.ibm.tivoli.transperf.arm4.transaction.Arm40TransactionFactory";
         Class tranFactoryClass =
Class.forName(tranFactoryName,true,ClassLoader.getSystemClassLoader());
```

```
        armTranFactory =
(ArmTransactionFactory)tranFactoryClass.newInstance();
        armOn = true;
        System.out.println("ARM interface activated");
      } catch ( Exception ex ) {
        ex.printStackTrace();
        armOn = false;
      }
   }
   private ArmApplicationDefinition armAppDef;
   private ArmTransactionDefinition armTranDefs;
   private ArmTransactionDefinition armTranDefc;
   private ArmTransactionDefinition armTranDefc1;
   private ArmTransactionDefinition armTranDefc2;
   private ArmApplication armClient;
   private ArmTransaction armTran;
   private ArmCorrelator corr;
   private JTextArea ta;
```

3. Initialize the ARM environment when the servlet is initialized, as shown in Example 6-3.

Example 6-3 Initializing the ARM environment

```
public ARMAppsMain() {
      r = new Random();
      if (armOn) {
        armAppDef =
armTranFactory.newArmApplicationDefinition("TI6M11",null,null);
        System.out.println("ARM appl TI6M11 defined");
        armTranDefs =
armTranFactory.newArmTransactionDefinition(armAppDef,"SimpleTran",null,
null);
        armTranDefc =
armTranFactory.newArmTransactionDefinition(armAppDef,"ComplexTran",null
,null);
        armTranDefc1 =
armTranFactory.newArmTransactionDefinition(armAppDef,"ComplexSub1",null
,null);
        armTranDefc2 =
armTranFactory.newArmTransactionDefinition(armAppDef,"ComplexSub2",null
,null);
        System.out.println("ARM transactions defined");
      }
```

4. Instrument the start of the transaction, as shown in Example 6-4. The armStatus variable contains the transaction status. We initialize it with a STATUS_GOOD indicator. The ta.append can only be retrieved after the transaction is started using the start() method.

Example 6-4 Starting the ARM transaction

```
int armStatus = ArmConstants.STATUS_GOOD;
   if (armOn) {
      armClient =
armTranFactory.newArmApplication(armAppDef,"ITSO","Austin",null);
      armTran =
armTranFactory.newArmTransaction(armClient,armTranDefs);
      armTran.start();
      ta.append("-+-");
   }
```

5. Stop the ARM transaction, as shown in Example 6-5.

Example 6-5 Stopping the ARM transaction

```
if (armOn) {
            armTran.stop(armStatus);
            armClient.end();
            ta.append("==");
         }
```

6. The complex part of the sample Java application is calling the subtransaction, as shown in Example 6-6.

Example 6-6 Complex part of the sample application

```
long i = r.nextInt(300)+100;
         ta.append("Main-"+i+"-");
         try {
            Thread.sleep(i);
         } catch (InterruptedException e) {
         }
         sub1();
         sub2();
         if (armOn) {
            armTran.stop(armStatus);
            armClient.end();
            ta.append(">>==");
         }
         ta.append("\n");
      }
```

```
   }

   public void sub1() {
      int armStatus = ArmConstants.STATUS_GOOD;
      if (armOn) {
         armClient =
armTranFactory.newArmApplication(armAppDef,"ITSO","Austin",null);
         armTran =
armTranFactory.newArmTransaction(armClient,armTranDefc1);
         armTran.start(corr);
         ta.append("-+-<<");
      }

      long i = r.nextInt(1000)+300;
      ta.append("Sub1-"+i+"-");
      try {
         Thread.sleep(i);
      } catch (InterruptedException e) {
      }
      if (armOn) {
         armTran.stop(armStatus);
         armClient.end();
         ta.append(">>==");
      }
   }

   public void sub2() {
      int armStatus = ArmConstants.STATUS_GOOD;
      if (armOn) {
         armClient =
armTranFactory.newArmApplication(armAppDef,"ITSO","Austin",null);
         armTran =
armTranFactory.newArmTransaction(armClient,armTranDefc2);
         armTran.start(corr);
         ta.append("-+-<<");
      }

      long i = r.nextInt(1000)+300;
      ta.append("Sub2-"+i+"-");
      try {
         Thread.sleep(i);
      } catch (InterruptedException e) {
      }
      if (armOn) {
         armTran.stop(armStatus);
```

```
            armClient.end();
            ta.append(">>==");
        }
    }
```

6.6.2 ARM discovery

To define each transaction monitor, do one of these actions:

- Incorporate the ARM-instrumented applications into the monitoring environment by uploading an XML file for the application.
- Use ARM discovery to identify ARM applications if you do not know which applications you want to monitor.

The easiest way to create a listening policy is to define a discovery to find the transaction. To configure the ARM discovery, select **Configuration** → **Discovery**. Choose **ARM Application** from the menu and click **Create New.** Enter the ARM settings to discover the transaction in the sample application, as shown in Figure 6-23. Click **Next**.

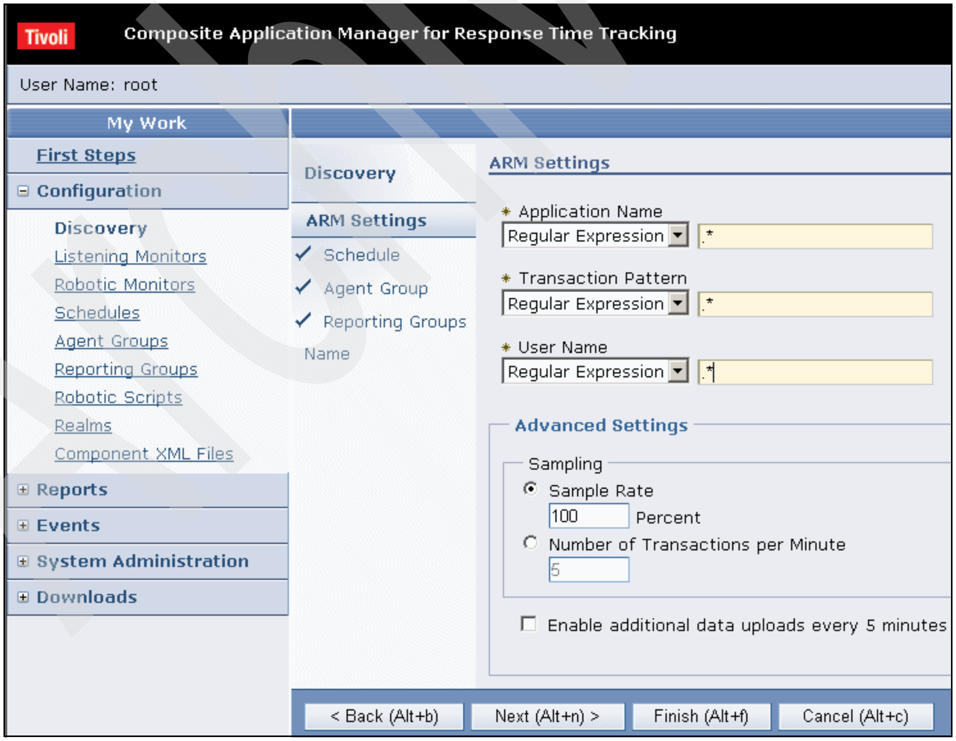


Figure 6-23 ARM discovery settings

After the ARM discovery monitor configuration, we run the sample application. After capturing some transactions, you get the discovered ARM application, as shown in Figure 6-24.

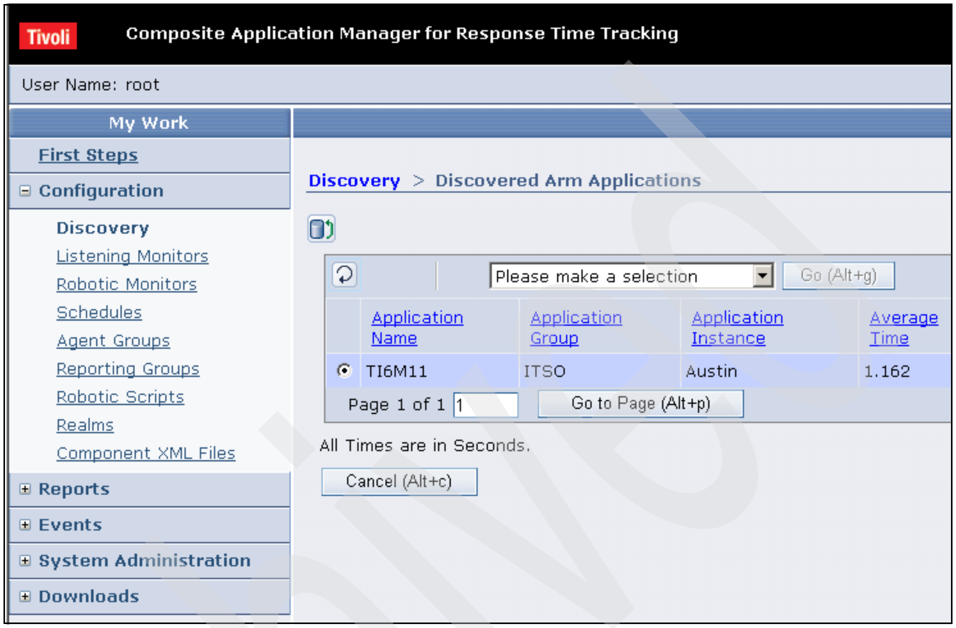


Figure 6-24 Discovered ARM instrumentation application by ARM discovery

6.6.3 ARM listening monitor

From the discovered ARM transactions in Figure 6-24, we define the listening monitor.

To create the ARM listening monitor, select the discovered ARM instrumentation application and choose **Create Listening Monitor From** from the menu. Click **Go** and configure ARM 4.0 settings for the listening monitor, as shown in Figure 6-25.

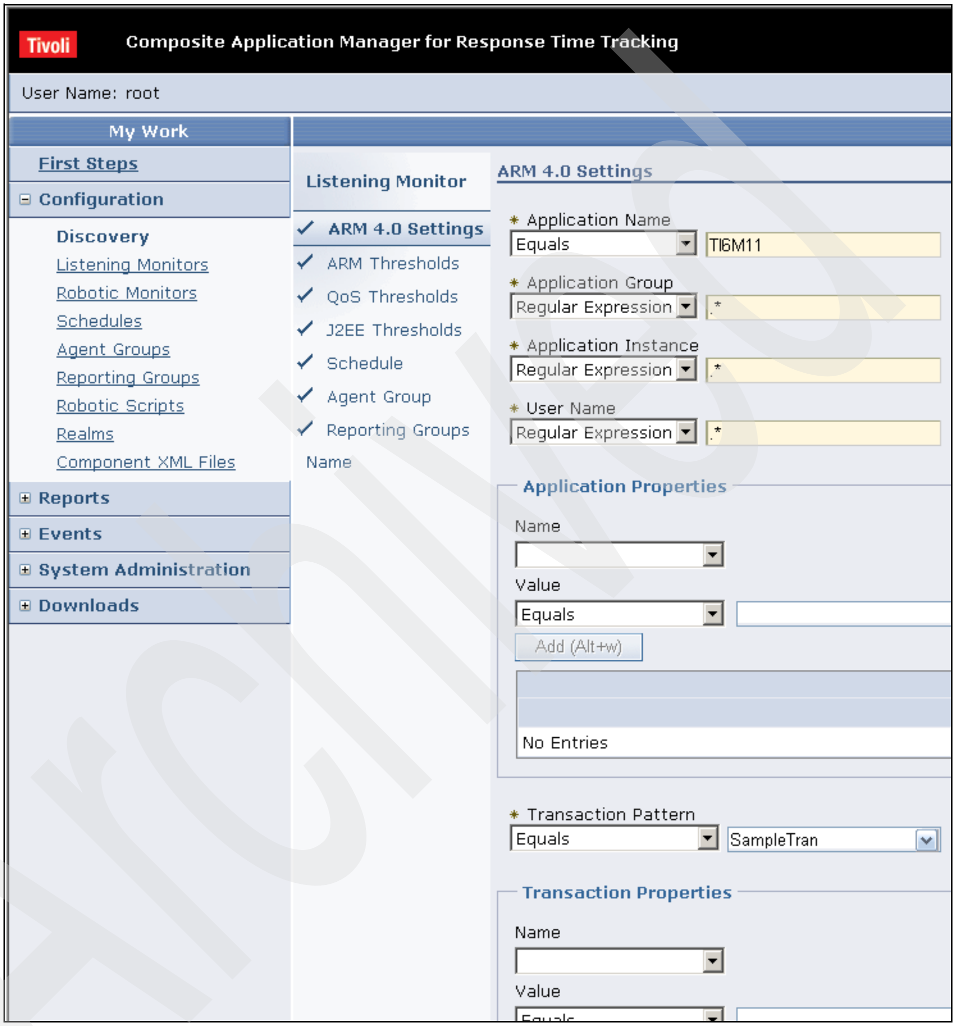


Figure 6-25 ARM listening monitor settings

6.6.4 ARM report

After the listening monitor is running, you can see the response time and topology information using **Reports** → **Dashboard**. Figure 6-26 shows the transaction topology for the sample application.

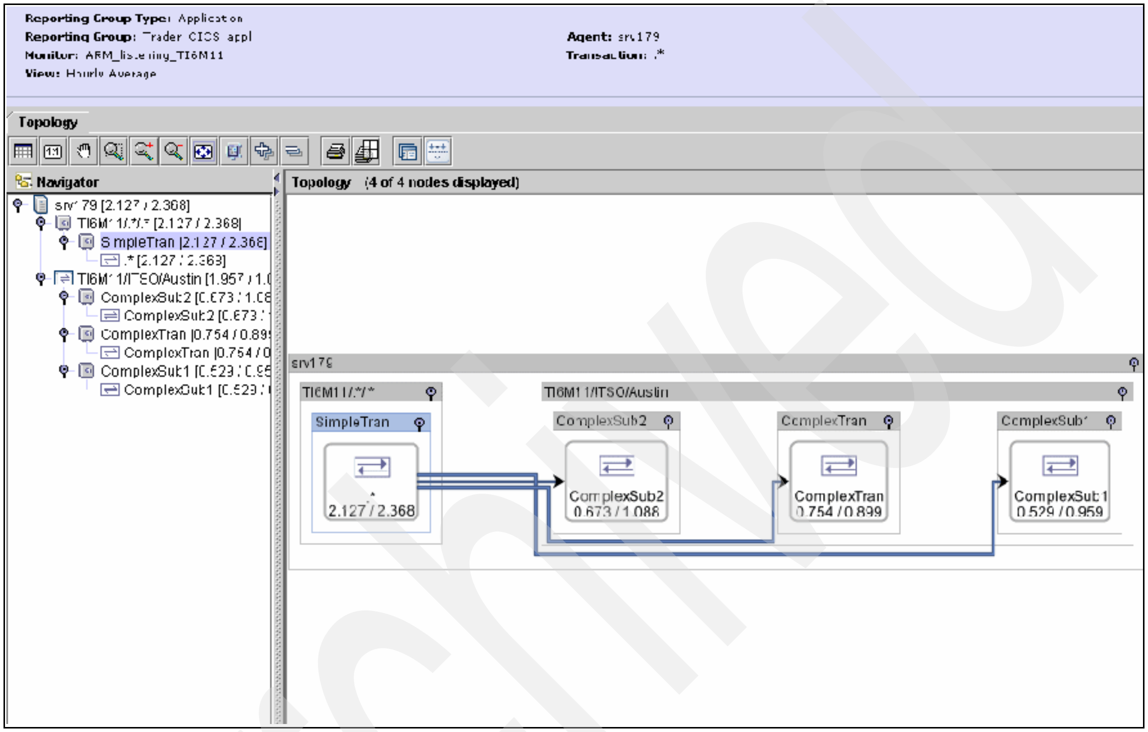


Figure 6-26 Sample application transaction topology

6.7 Client application tracker

The client application tracker monitors the round-trip response time of transactions originating at the client desktop. The client application tracker is an evolution from the IBM Tivoli ETEWatch® product (formerly Candle ETEWatch).

> **Note:** Do not use the client application tracker agent on the same machine as STI.

The client application tracker provides the capability to monitor Lotus Notes, Microsoft Outlook, SAP, IBM Personal Communications, and so on. For more information, refer to *ITCAM for Response Time Tracking Administrator's Guide*, SC32-1905.

We monitor IBM Personal Communications, which is a new object monitoring application in this release. For monitoring IBM Personal Communications, you need to do the following process in advance:

1. Check that the behavior modules file is named pcom_5x.dat in MA_install_directory/app/CAT/Mgmt/Collector/TC directory, as shown in Figure 6-27.

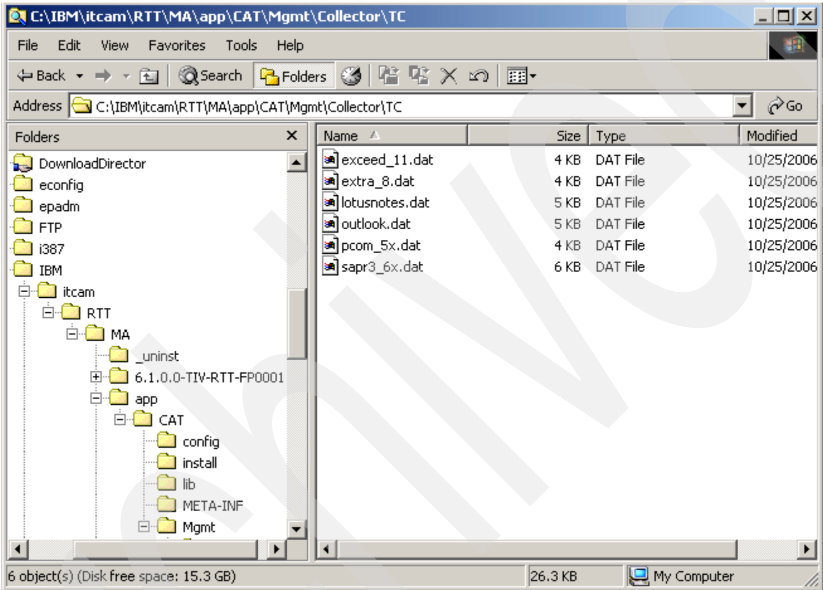


Figure 6-27 Behavior modules directory

2. Edit the rule file tn3270.rul, which identifies strings of characters that appear in your application and identifies applications and transactions. The tn3270.rul file is in MA_install_directory/app/CAT/Mgmt/Collector directory. We add the line for the default rule for unidentified transactions, as shown in Example 6-7. For details about creating this custom rule file, see the chapter about 3270 applications in the *CandleNet ETEWatch User's Guide*, GC32-9178.

Example 6-7 Modify the tn3270.rul file

```
<...>
#
# Default rule for unidentified transactions
#
tranid: return("Unknown");
```

3. To compile the modified tn3270.rul rule file, execute the **`tdl tn3270.rul`** command from MA_install_directory/app/CAT/Mgmt/Collector.
4. Restart the monitoring agent using **Start** → **Settings** → **Control Panel** → **Administrative Tools** → **Services**. Execute **Restart** for the IBM Tivoli Monitoring for Transaction Performance service, as shown in Figure 6-28.

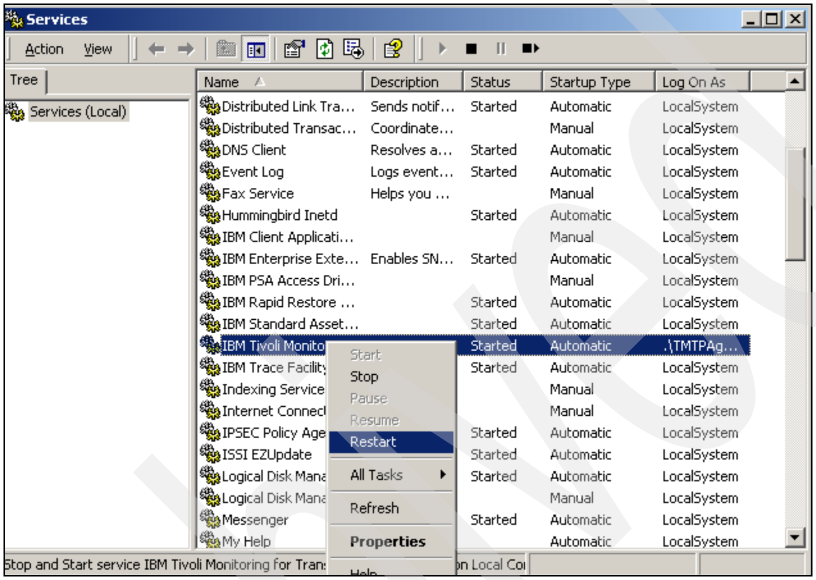


Figure 6-28 Restart the monitoring agent using services

6.7.1 Client application tracker discovery

To monitor the IBM Personal Communications application, create the discovery using the ARM application for the behavior modules:

1. To configure the ARM discovery, refer to 6.6.2, “ARM discovery” on page 307.

2. We executed the IBM Personal Communications emulator to create the transaction, as shown in Figure 6-29.

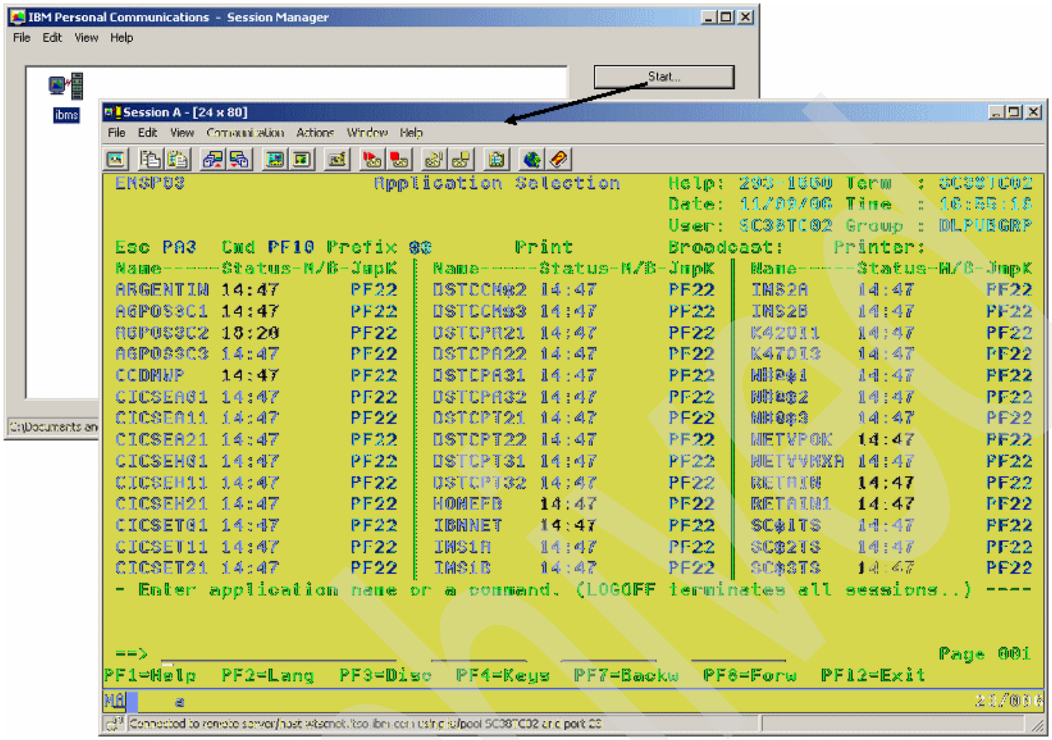


Figure 6-29 Execute the IBM Personal Communications emulator

> **Note:** You can check that the client application tracker runs using the log files. If the product runs, the file size grows when you execute the application. The log files are:
>
> - MA_install_directory/app/CAT/Mgmt/Log/KeeAgent.log
> - MA_install_directory/app/CAT/Mgmt/Log/KeeHook.log

3. After the IBM Personal Communications emulator executes, get the discovered application, as shown in Figure 6-30.

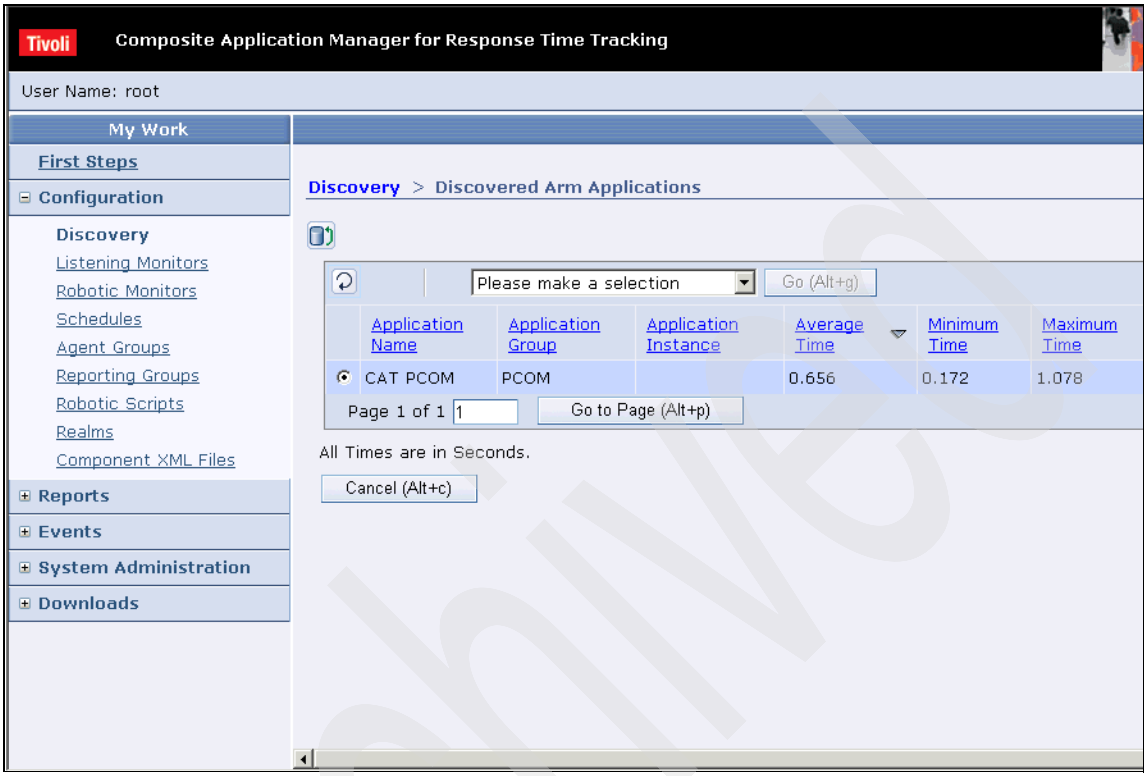


Figure 6-30 Discovered IBM Personal Communications application by ARM Discovery

6.7.2 Client Application Tracker listening monitor

From the discovered CAT PCOM application in Figure 6-30, we can define a listening monitor.

To create the client application tracker listening monitor, select the discovered CAT PCOM application and choose **Create Listening Monitor From** from the menu. Click **Go** and configure the ARM 4.0 settings for the listening monitor, as shown in Figure 6-31.

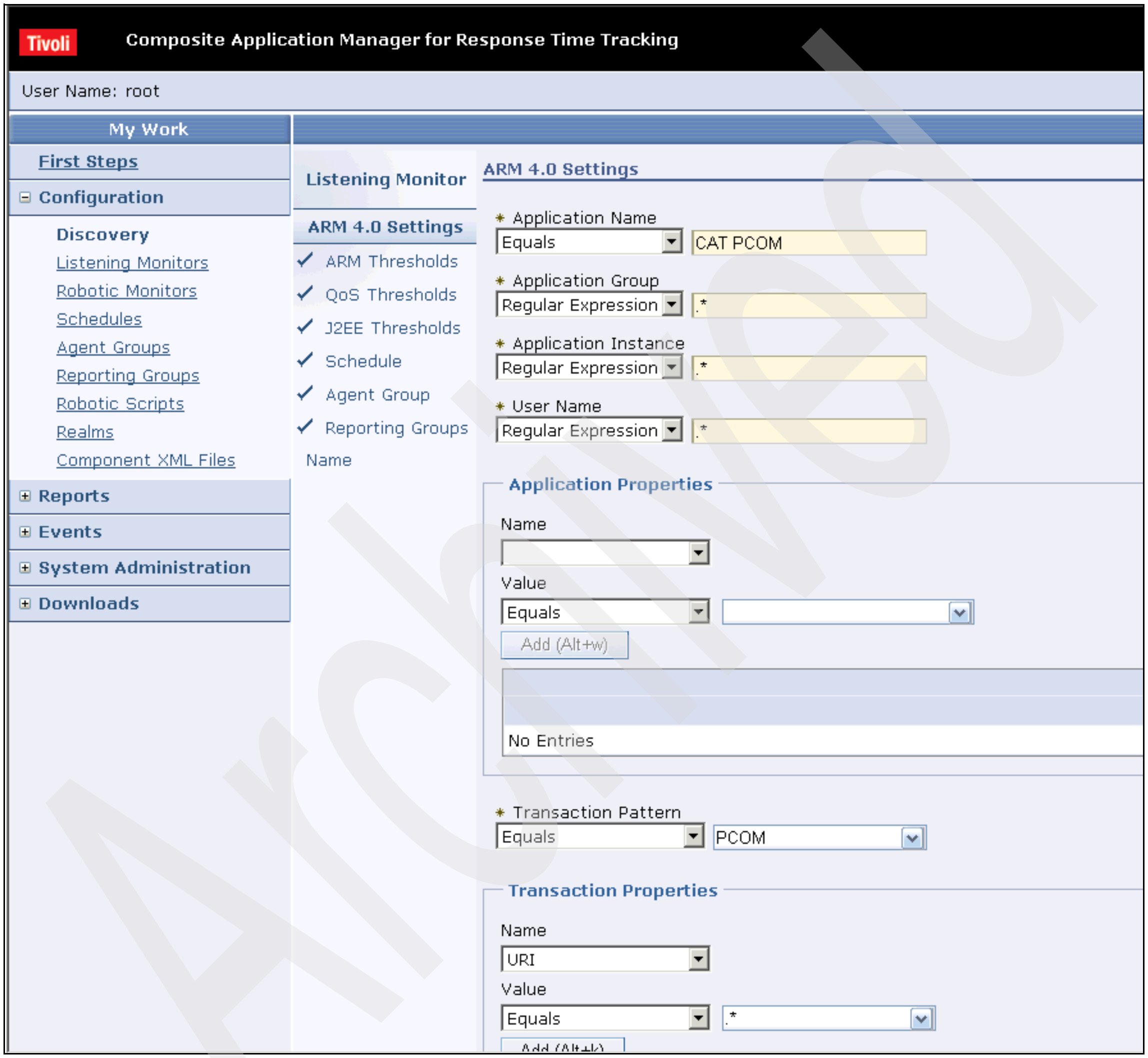


Figure 6-31 Client application tracker listening monitor settings for CAT PCOM

6.7.3 Client application tracker report

After the listening monitor is running, you can see the separate response times and transaction topology using **Reports** → **Dashboard**. Figure 6-32 shows the transaction topology of the IBM Personal Communications application.

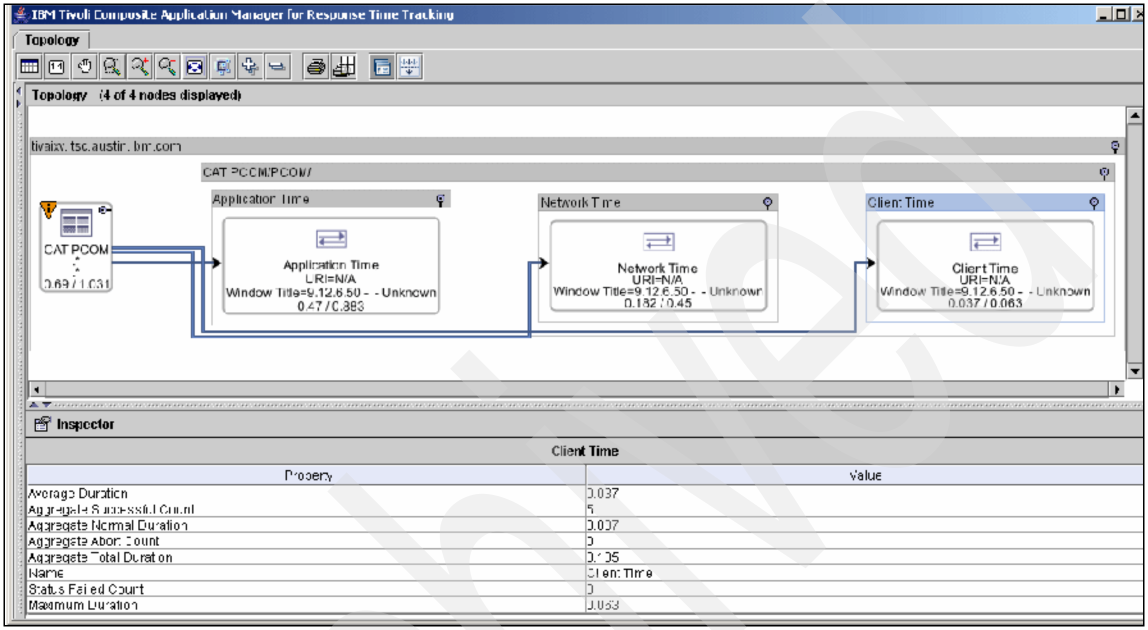


Figure 6-32 Transaction topology by the client application tracker monitoring component

6.8 Generic playback

Generic playback is a new monitoring component in this release. It runs any command on a monitoring agent and collects timing information for the command execution. The generic playback component automates running scripts based on robotic schedule. For example, it can be used to test server availability with FTP, telnet, or ping to query a database with a custom SQL command and to run a custom shell script.

6.8.1 Using the command

Use the following steps to use a command-based generic playback monitor:

1. To use generic playback, create a generic playback robotic monitor. We create the robotic monitor for server availability with the **ping** command. Select **Configuration** → **Robotic Monitors** and choose **Generic Playback** from the menu. Click **Create New**. Figure 6-33 shows the generic playback robotic monitor settings.

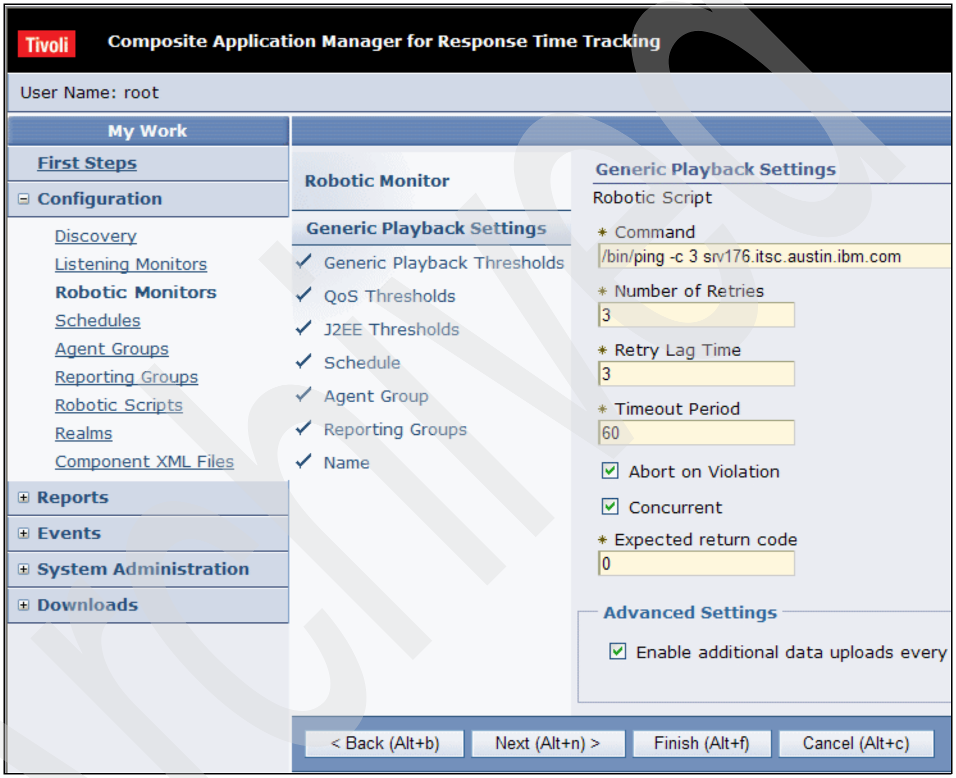


Figure 6-33 Generic Playback robotic monitor settings

> **Note:** If you put the command on a monitoring agent instead of uploading files, it needs to know the full path to the command when creating the playback monitor, as shown in Figure 6-33.

2. Continue to enter the generic playback settings and name. Click **Finish**.

3. After the robotic monitor is running for a while, you can see the report from **Reports** → **Dashboard**. Figure 6-34 is our output from the generic playback robotic monitor. It shows the `ping` command execution performance result within a certain period of time.

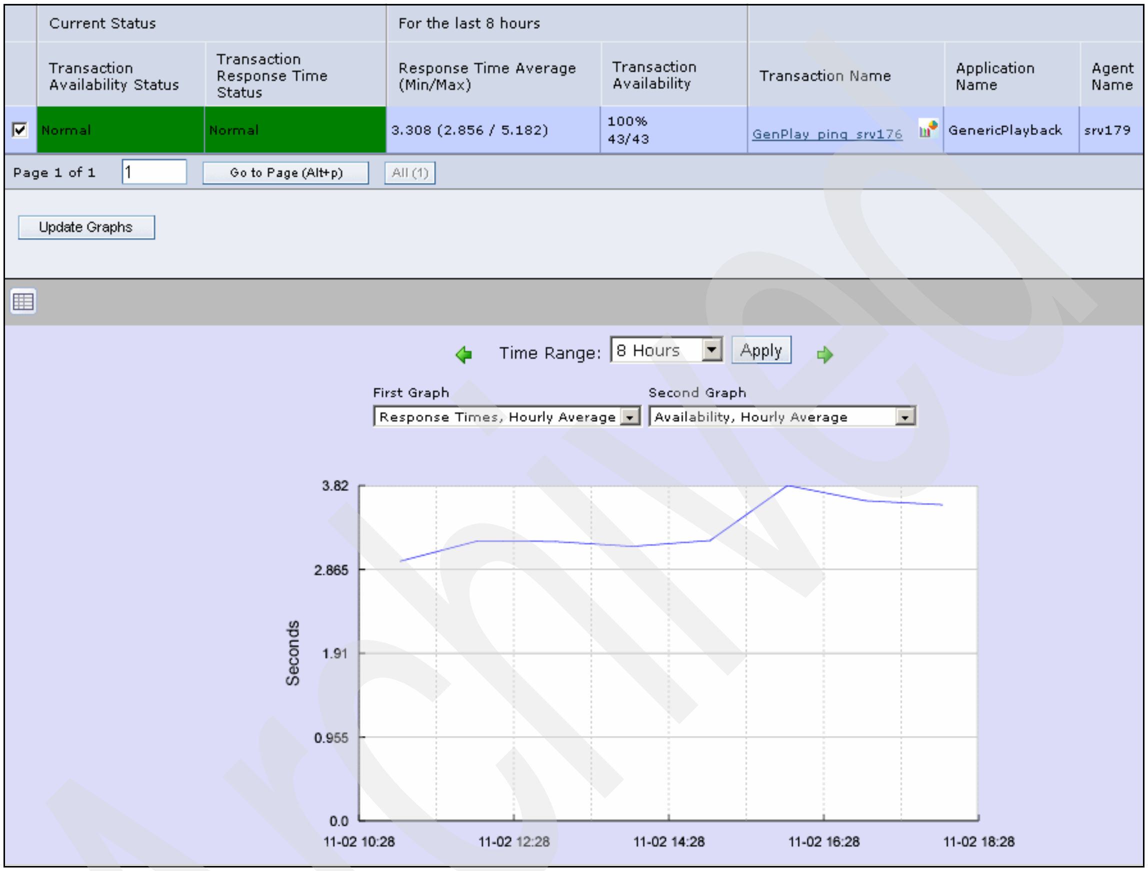


Figure 6-34 Generic playback report

6.8.2 Using an uploaded file

To use an uploaded file for the generic playback monitor:

1. Use the robotic scripts. Select **Configuration** → **Robotic Scripts** and choose **Generic Playback** from the menu. Click **Create New**. Figure 6-35 shows the generic playback robotic monitor settings. Click **Browse** → **Open** and select the file. Click **Add**. This uploads the file to the management server.

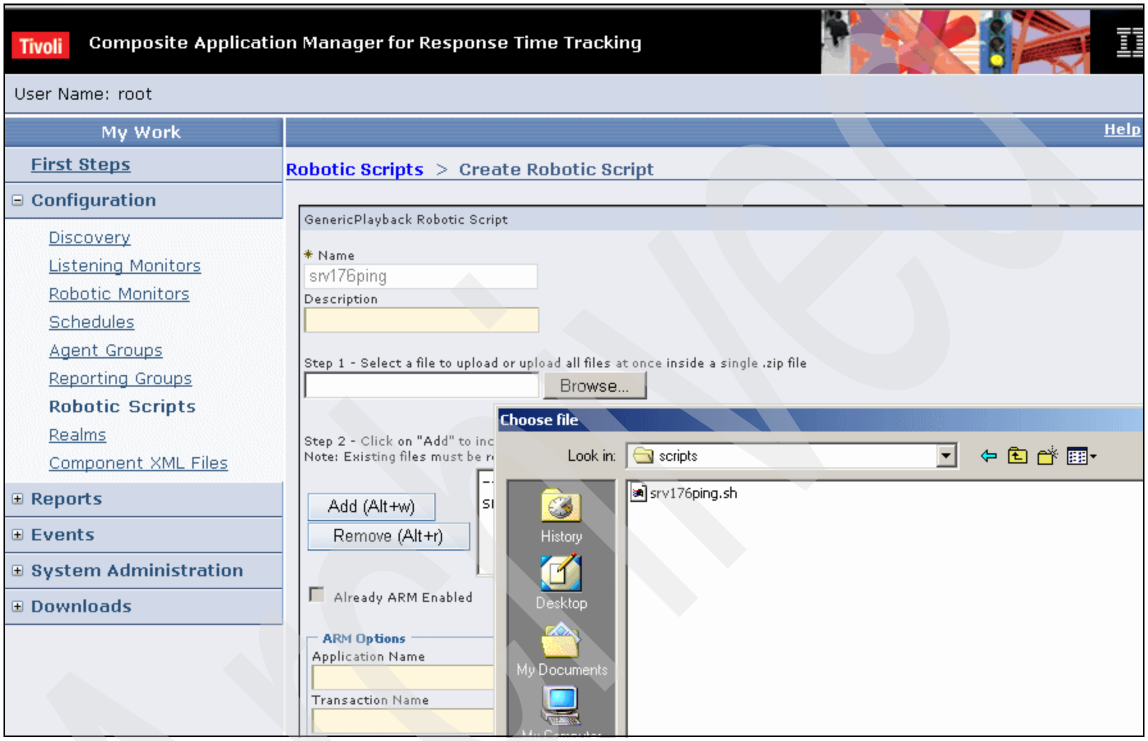


Figure 6-35 Creating a robotic script

2. To monitor with the uploaded file, select **Configuration** → **Robotic Monitors**. Choose **Generic Playback** from the menu, and then click **Create New**. Figure 6-36 shows the generic playback robotic monitor settings.

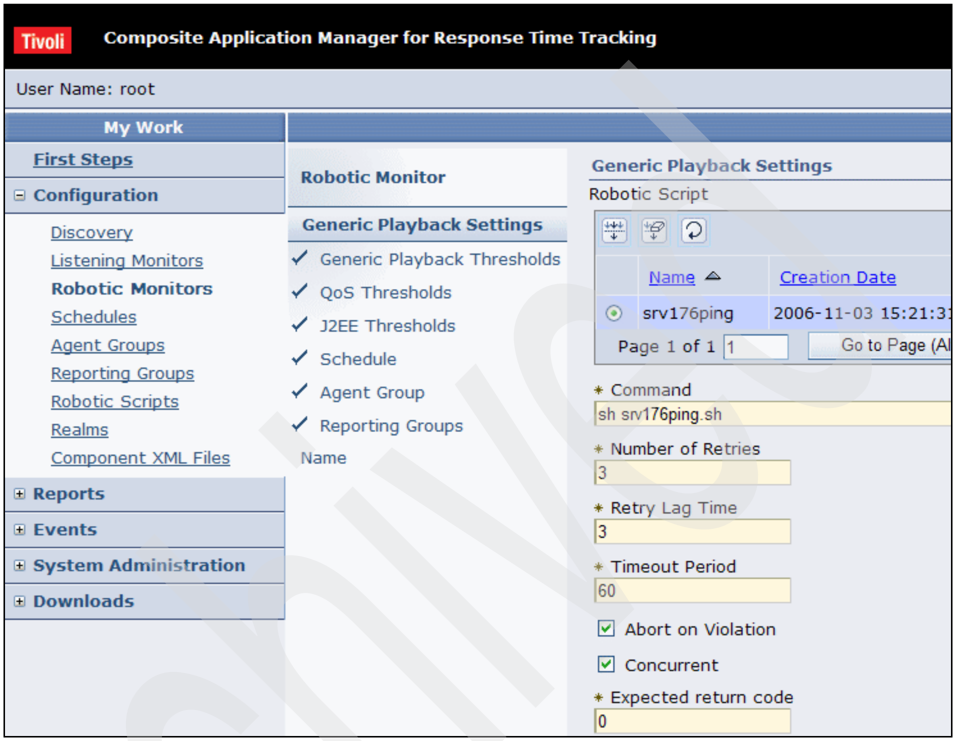


Figure 6-36 Generic playback robotic monitor settings with robotic script

The generic playback works by wrapping a recording in ARM with start and stop calls so that the software can measure the total time the command or recording took. For information about how to do this, see *ITCAM for Response Time Tracking Administrator's Guide*, SC32-1905.

6.9 Rational Performance Tester

The Rational Performance Tester (RPT) component runs with Rational Performance Tester runtime to monitor protocol-based transactions. To monitor with the Rational Performance Tester component, you need to create a Rational Performance Tester script first.

You need to install a Rational Performance Tester workbench before you can create a Rational Performance Tester script. Use the Rational Performance

Tester workbench image in the RTT 6.1 IF0006 package instead of the image in fix pack 1. To install the Rational Performance Tester workbench, unzip the files in image 6.1.0.1-TIV-RTT-IF0006_RPT.zip, then run the ..\RPT\w32-ix86\disk1\setup_RPT.exe. This installs the Rational Performance Tester workbench with the ITCAM for Response Time Tracking 6.1 plug-in for you.

6.9.1 Creating a Rational Performance Tester script

We describe the Rational Performance Tester record procedure with the trader application.

Perform the following steps:

1. On the Rational Performance Tester installed machine, click **Start** → **All Programs** → **RPT** → **IBM Rational Performance Tester** → **IBM Rational Performance Tester**. Type in a project file location.

2. Click **File** → **New** → **Test From Recording**. Then click **Create Test From New Recording** and **HTTP Test** and open the Create New Test From Recording dialog. Define the project name and recording file name, then click **OK**, as shown in Figure 6-37.

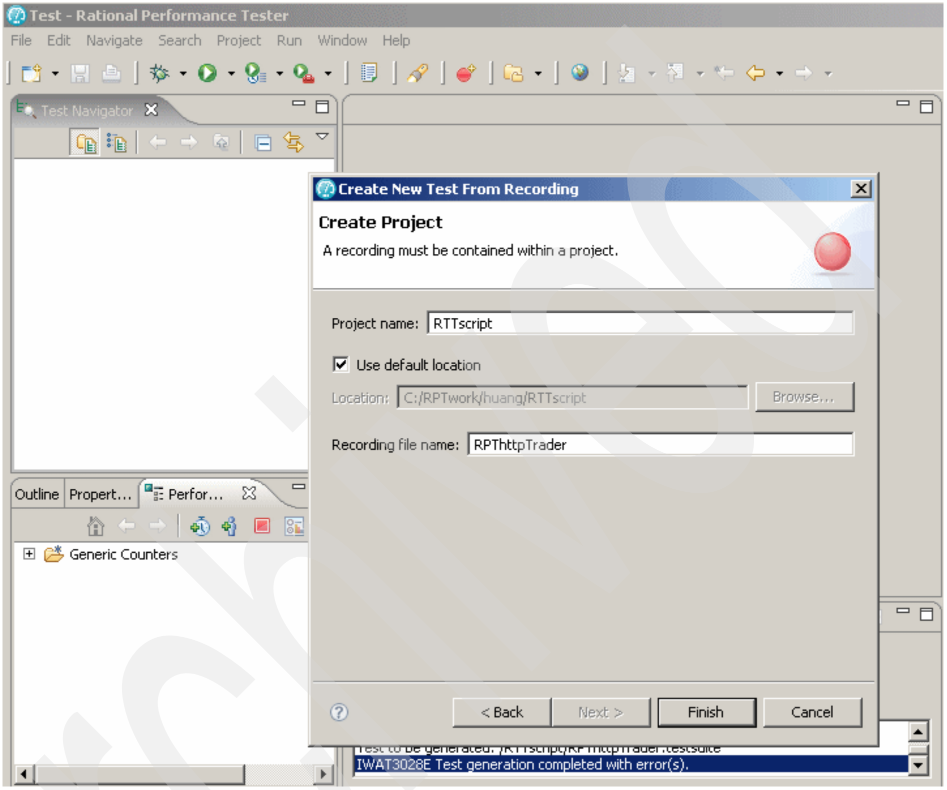


Figure 6-37 Starting Record Rational Performance Tester http script

3. We record the trader application transaction using Microsoft Internet Explorer. Close the Internet Explorer browser. The script is available to edit, as shown in Figure 6-38.

> **Note:** We only use the Rational Performance Tester HTTP script for Microsoft Internet Explorer. For more information about Rational Performance Tester usage, refer to the online help of Rational Performance Tester.

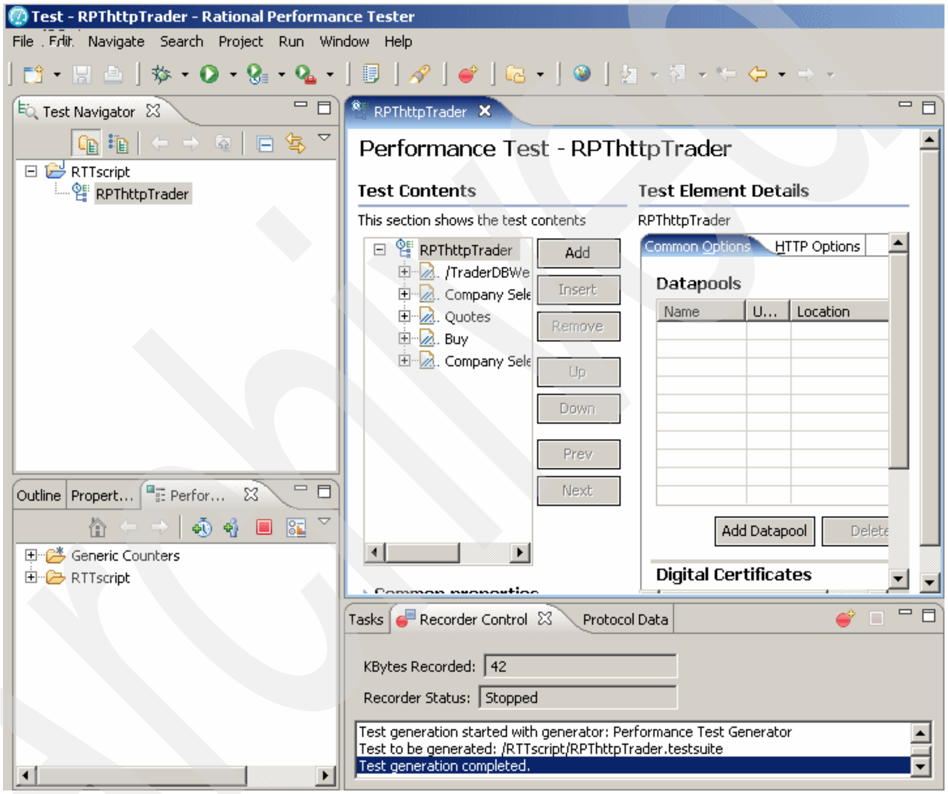


Figure 6-38 Created transaction script by Rational Performance Tester

4. To use the Rational Performance Tester script on the management server, export the recorded transaction to the management server via RTT plug-ins. Right-click the script name in the test navigator panel and select **Export**, Select **ITCAM Response Time Tracking** as the export destination and click **Next**. Type in the management server information, as shown in Figure 6-39.

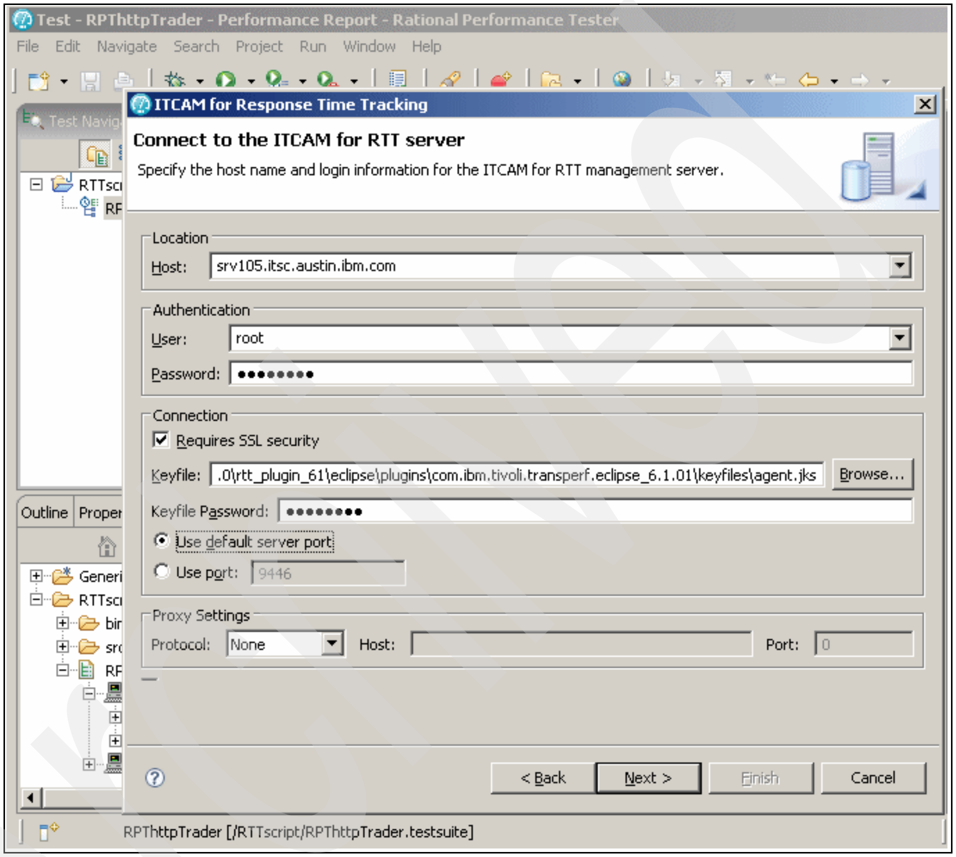


Figure 6-39 Export the recorded transaction by Rational Performance Tester

6.9.2 Rational Performance Tester HTTP monitor and report

After creating and exporting the Rational Performance Tester script, you can use the script to perform transaction monitoring. To monitor with Rational Performance Tester:

1. Select **Configuration** → **Robotic Monitors**. Choose **Rational Performance Tester** from the menu, and then click **Create New**. Figure 6-18 on page 300 shows the Rational Performance Tester Settings.

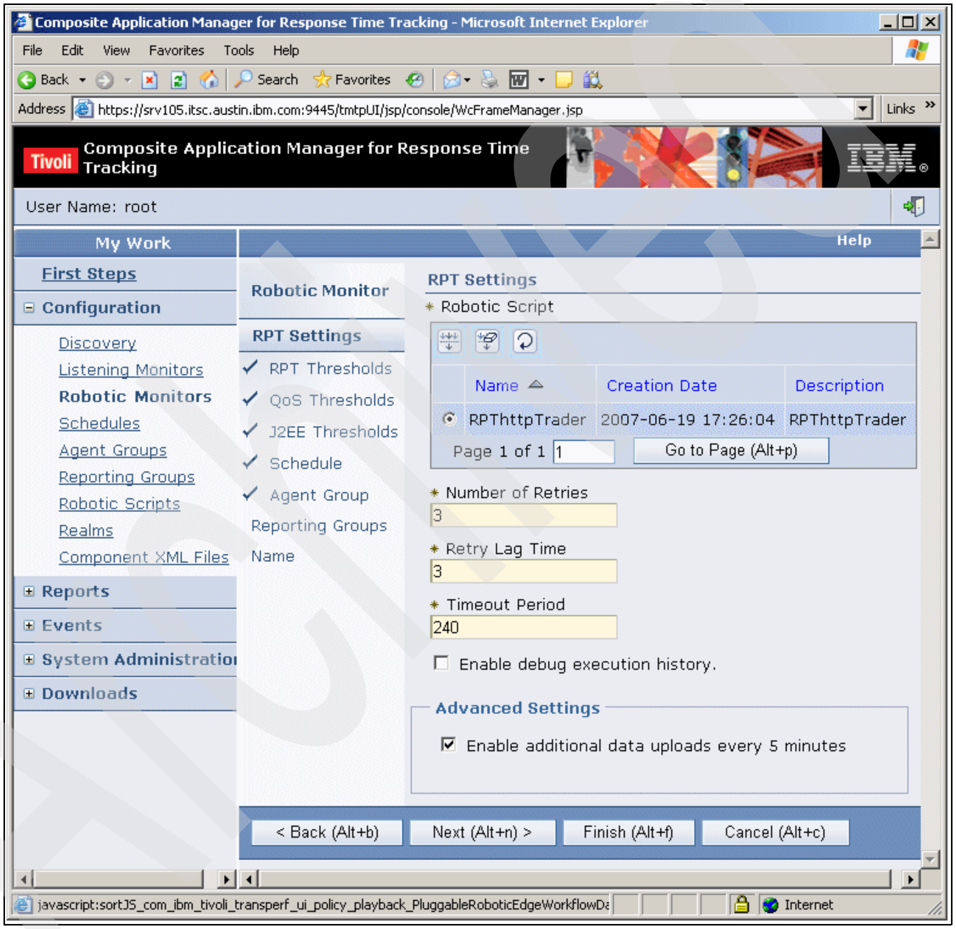


Figure 6-40 Rational Performance Tester robotic monitor settings

2. We created the Rational Performance Tester robotic monitors using the default playback schedule. After the robotic monitor is running, you can see the report using **Reports** → **Dashboard**, as shown in Figure 6-41.

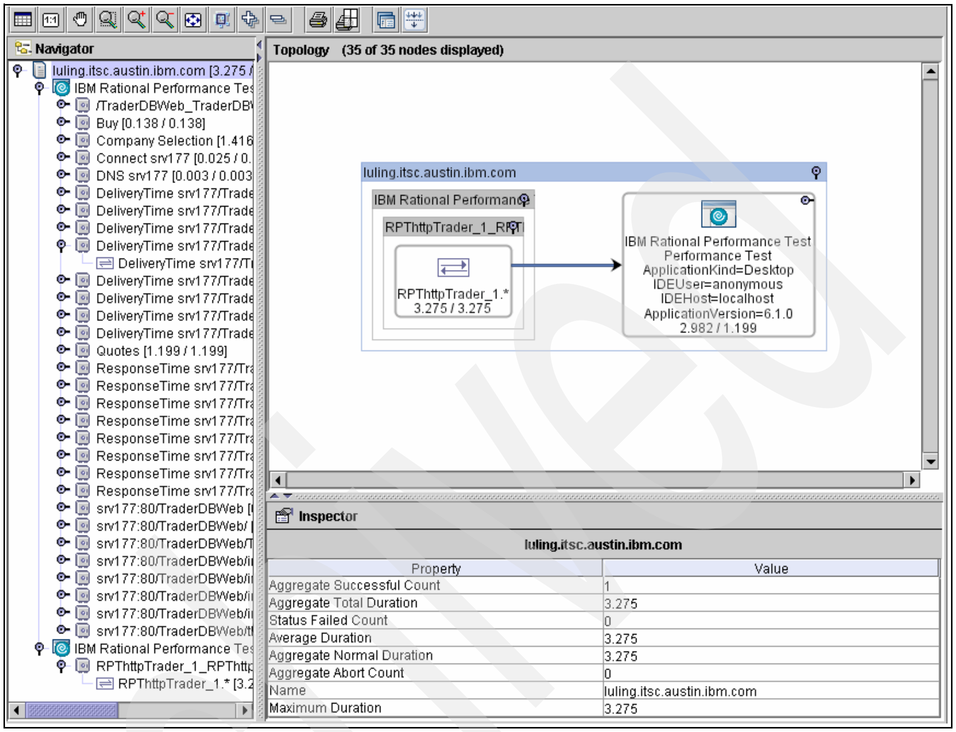


Figure 6-41 Transaction topology report for Rational Performance Tester robotic monitor

6.10 Working with Tivoli Enterprise Portal

This section describes Tivoli Enterprise Portal information from ITCAM for Response Time Tracking. The Tivoli Enterprise Monitoring Agent is installed on the ITCAM for Response Time Tracking management server, which is khartoum. The Tivoli Enterprise Portal Client runs either as a desktop or a Web-based application. To use Tivoli Enterprise Portal Client as a desktop, it needs to be installed on your desktop. To access the Tivoli Enterprise Portal Client in the browser mode, type the URL in the browser and enter the user ID and password. The default user ID is sysadmin. The URL is:

```
http://lima:1920//cnp/client
```

We customized the logical view for ITCAM for Response Time Tracking. Figure 6-42 shows the agent status and message.

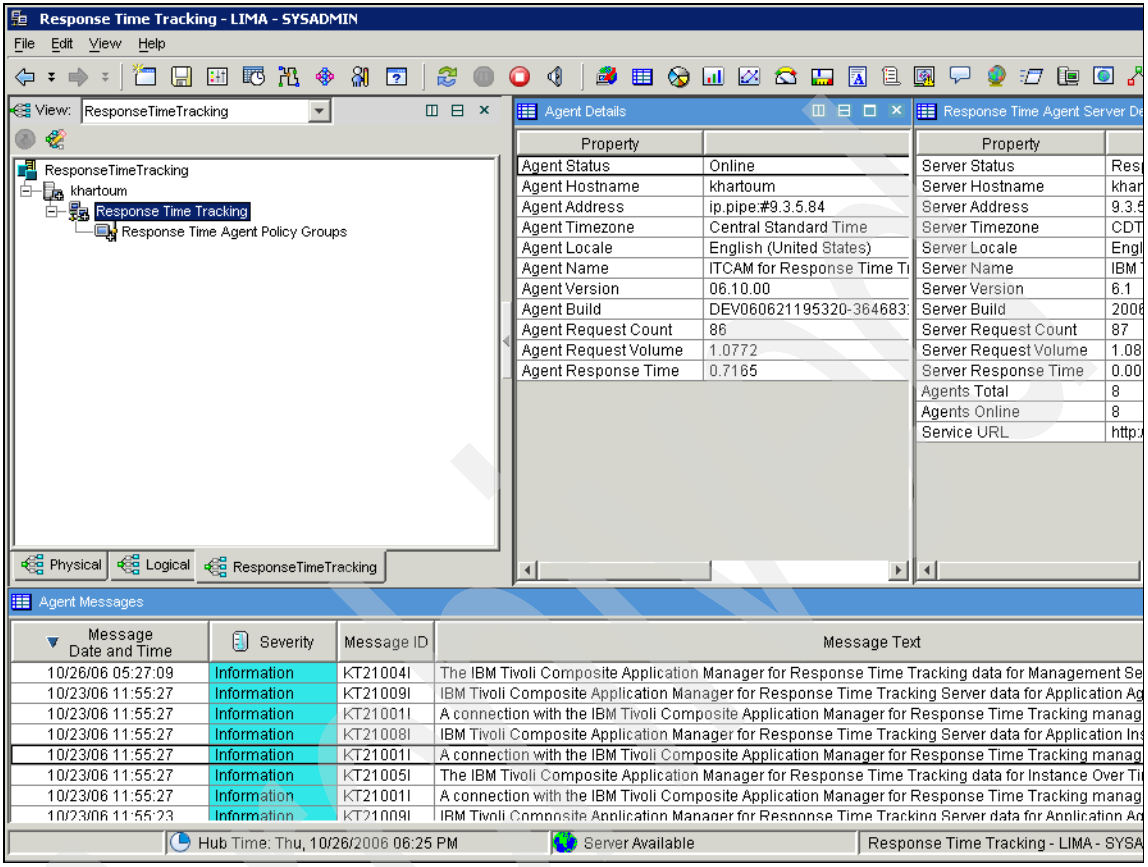


Figure 6-42 Response Time Tracking portal workspace

Select **Response Time Tracking Agent Policy Groups**. This opens the primary interface for ITCAM for Response Time Tracking, as shown in Figure 6-43. The policy groups summary shows the reporting groups in ITCAM for Response Time Tracking defined. To view the detailed reporting groups information, click the icon by the TraderWeb reporting group.

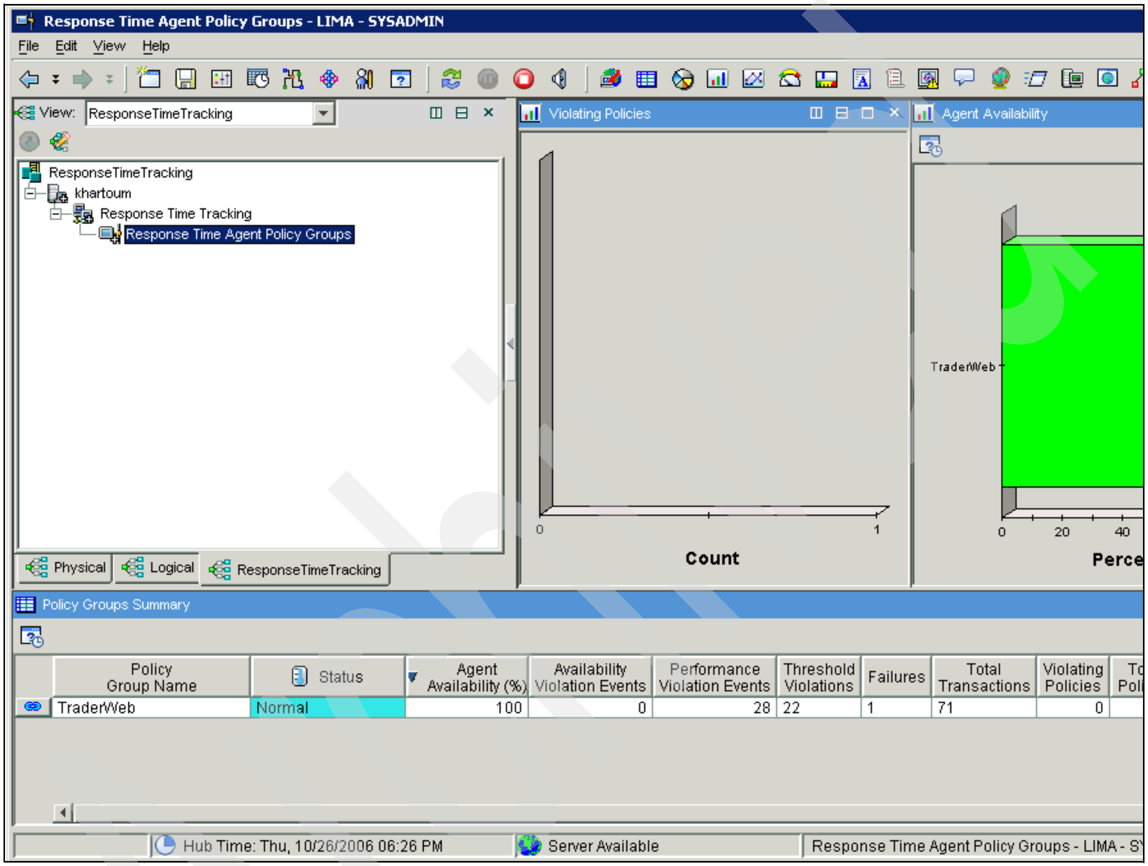


Figure 6-43 Response time tracking agent policy groups

This links to the reporting group TraderWeb under monitor summary, as shown in Figure 6-44. You can see the monitor status of the reporting group TraderWeb. To check each monitor's status, click the icon  by the policy name.

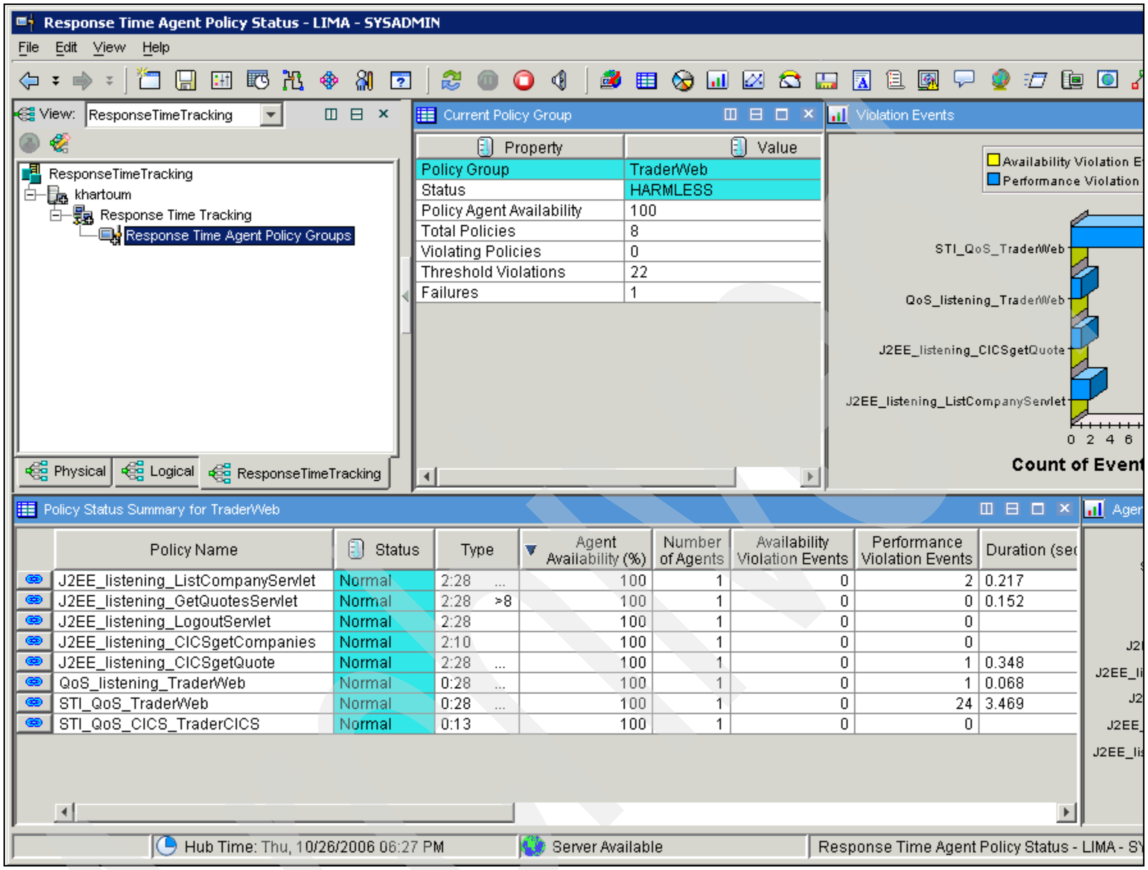


Figure 6-44 Policy status for policy group

In Figure 6-45, Tivoli Enterprise Portal shows STI_QoS_TraderWeb, the STI robotic monitors status. For more a detailed status, click the icon . This icon links to the related workspace.

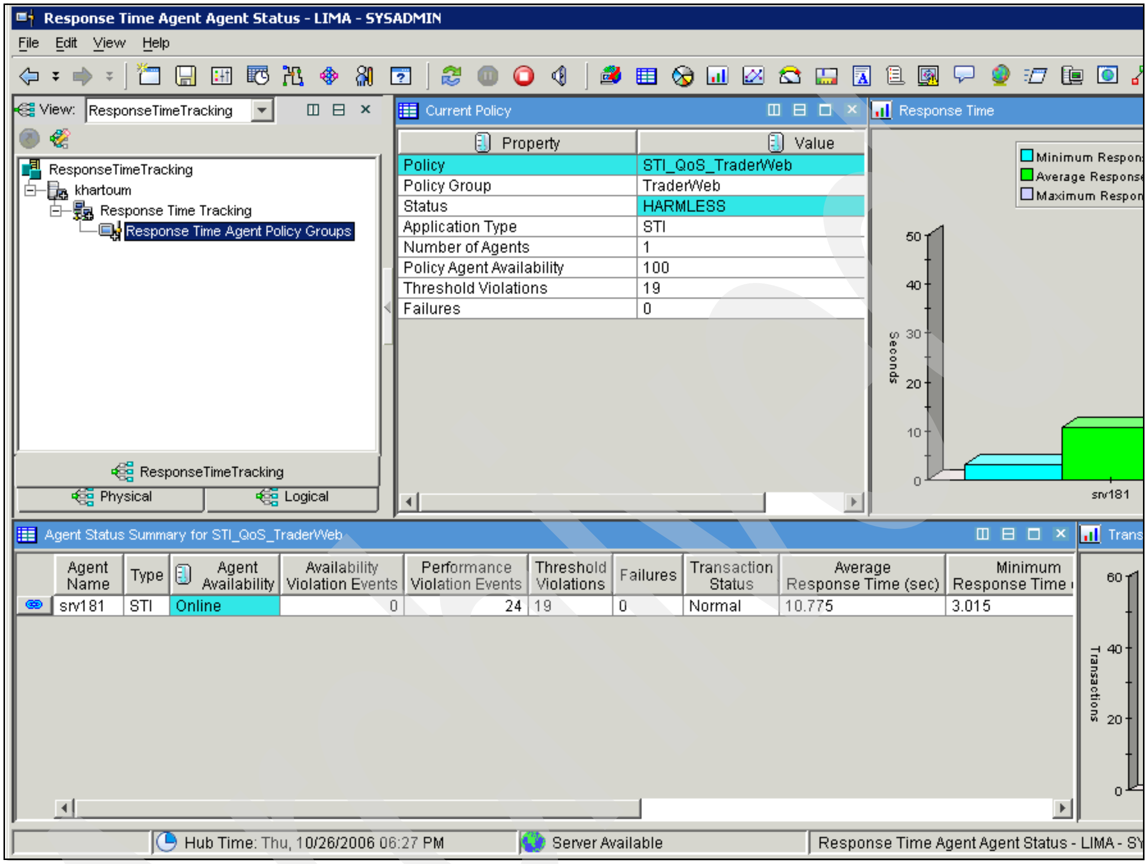


Figure 6-45 STI_QoS_TraderWeb robotic monitor status

The Tivoli Enterprise Portal provides historical data by configuring and starting historical data collection. For more information about the historical data collection settings, refer to IBM Tivoli Monitoring documentation. Figure 6-46 shows an example of the agent availability historical data over the last 8 hours.

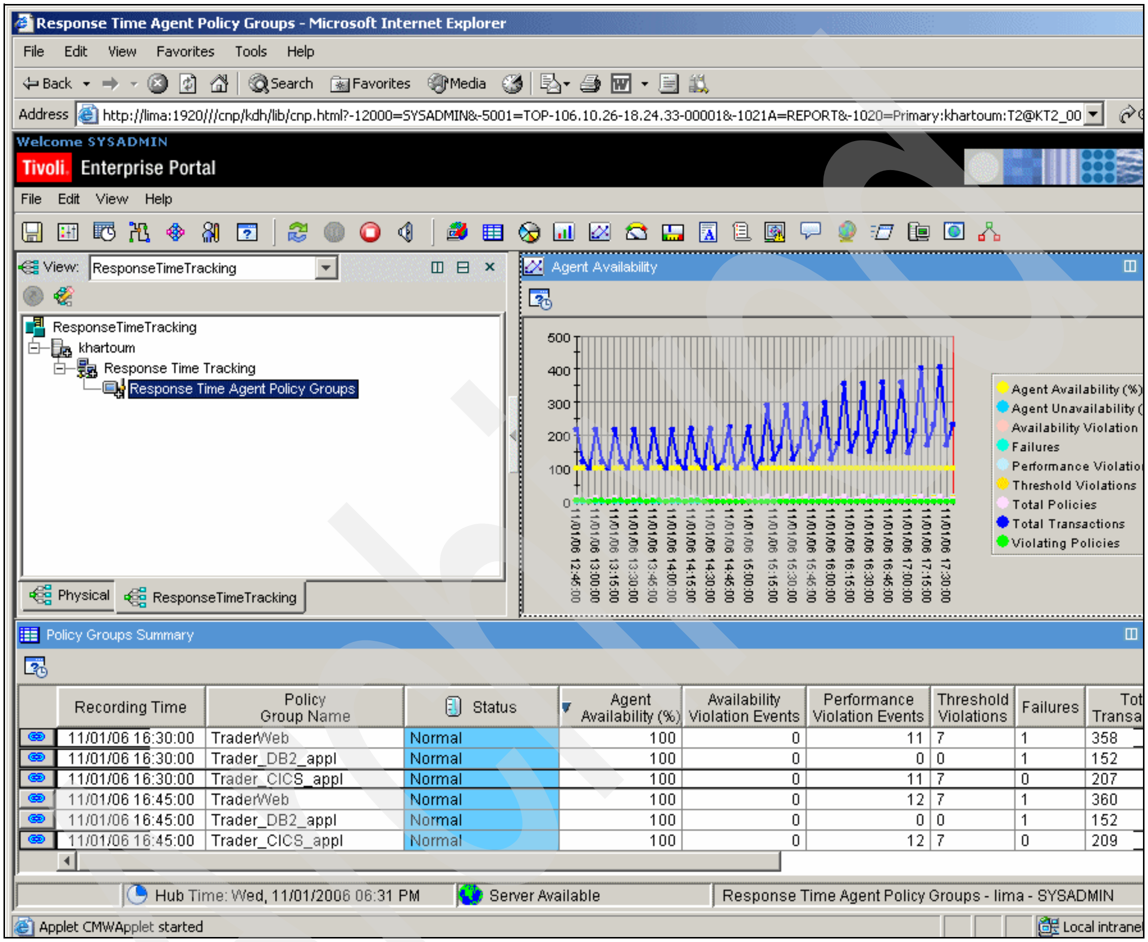


Figure 6-46 Display the agent availability historical data in Tivoli Enterprise Portal

7

ITCAM for Response Time concepts, installation, usage

This chapter describes the implementation of IBM Tivoli Composite Application Manager for Response Time in our environment. It includes:

7.1 Product features

ITCAM for Response Time provides the ability to collect response time and availability information by including a hook in the user application or performing an automated transaction to sample the user experience.

ITCAM for Response Time collects the response time information from user application instrumentation, such as Web traffic, 3270 interface, Lotus Notes, Microsoft Outlook, and other graphical applications. It also provides robotic simulation to check availability and response time of a scripted transaction. The transaction can be a Windows-based, Web-based, Citrix, SAP, or based on Mercury LoadRunner.

ITCAM for Response Time collects this response time information in the scope of an IBM Tivoli Monitoring environment. Data is collected into a common monitoring platform. This allows monitoring to be performed together with the resource monitoring and allows cross reference and correlation of events.

The major features of ITCAM for Response Time are:

- Helps monitor real user response time. It can identify sporadic problems that would otherwise get lost in the averages and show what users are experiencing.
- Records and plays back synthetic transactions. This provides both availability and response time monitoring, which can be useful for testing different locations and service providers and helps proactively find problems.
- Helps automate business practices. It creates comprehensive automated policies and situations to proactively help manage the user experience and provides expert advice to help understand how to best resolve specific response time issues.
- Delivers end-to-end integration using a common user interface. The integration of data and events with other IBM Tivoli Monitoring based solutions from IBM Tivoli Campsite Application Manager, IBM Tivoli Monitoring, and IBM Tivoli OMEGAMON to help provide comprehensive management of business applications. It thus allows management of the entire enterprise with a single user interface, therefore eliminating the need to learn multiple tools with different user interfaces and resulting in faster return on investment.

7.2 Product architecture

ITCAM for Response Time Version 6.2 is an evolution from ITCAM for Response Time Tracking Version 6.1. It inherits some of the major components and functions of that product.

Figure 7-1 shows the ITCAM for Response Time V6.2 component relationships and logical architecture.

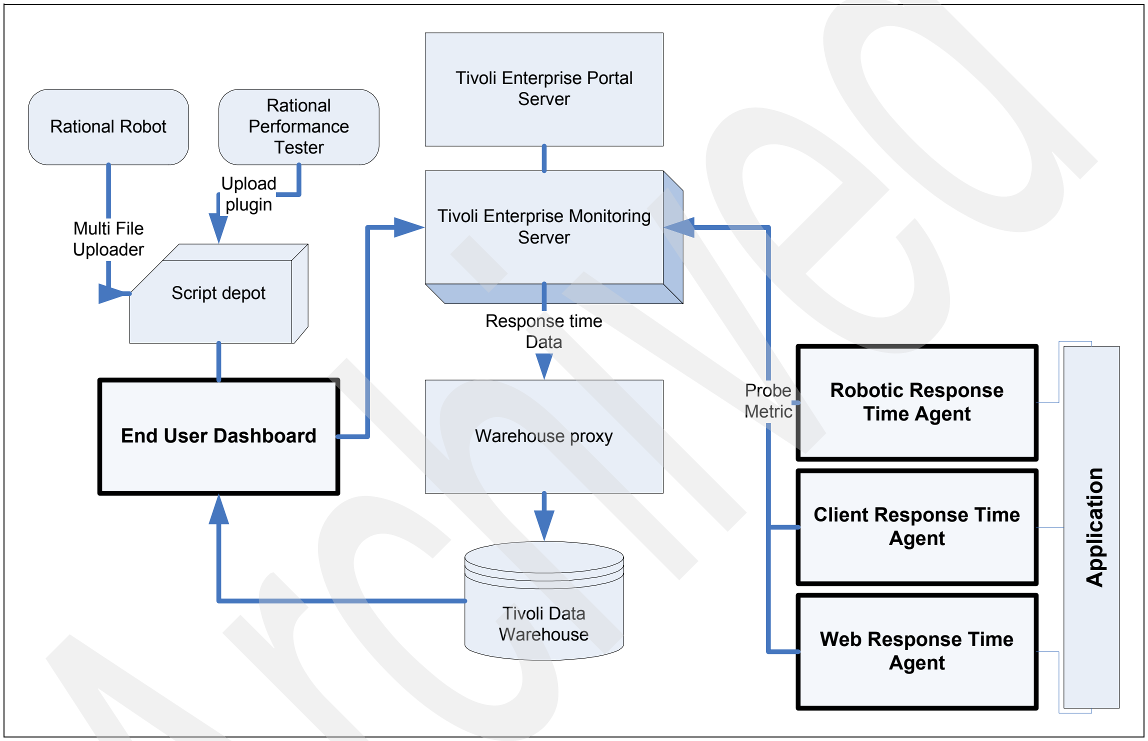


Figure 7-1 ITCAM for Response Time architecture

ITCAM for Response Time is an IBM Tivoli Monitoring 6.1 application. The IBM Tivoli Monitoring base architecture is explained in <Appendix n.n on page nnn>. The operator accesses ITCAM for Response Time using Tivoli Enterprise Portal, either the browser or desktop version.

In Figure 7-1, the bolded boxes represent the ITCAM for Response Time agents. The Robotic, Client, and Web Response Time agents connect to the application and retrieve response time information. Response time data is then stored in the Tivoli Data Warehouse. The End-user Dashboard component provides a comprehensive response time interface for all applications and agents on a specified IBM Tivoli Monitoring instance.

The End-user Dashboard also acts as robotic file depot. It stores the robotic scripts from either the Rational Robot or the Rational Performance Tester. These scripts are loaded by the robotic response time agent for execution.

Response time information from the Tivoli Enterprise Monitoring Agent is collected whenever one of the following happens:

- An online request is issued by the Tivoli Enterprise Portal (manual refresh or autoupdate interval expires)
- A situation interval expires and the situation requests the data
- The historical collection interval expires and the current data is written out to the historical file

We discuss the four components of ITCAM for Response Time in the following sections:

- 7.2.1, “Web Response Time agent” on page 336
- 7.2.2, “Client Response Time Agent” on page 338
- 7.2.3, “Robotic response time agent” on page 340
- 7.2.4, “End User Response Time Dashboard agent” on page 341

7.2.1 Web Response Time agent

The Web Response Time agent collects user response time for HTTP and HTTPS Web transactions.

- For HTTP traffic, the agent can listen to the local TCP/IP stack and measure the response time of the transaction.
- For HTTPS traffic, as the product needs to access unencrypted HTTP datastream, the agent runs on the Web server machine and makes use of the Web server exits to get access to the datastream.
- Appliance mode allows the agent to collect HTTP traffic from another machine in the same network segment by enabling a collection of network packets in promiscuous mode.

The agent is made up of three components, as displayed in Figure 7-2.

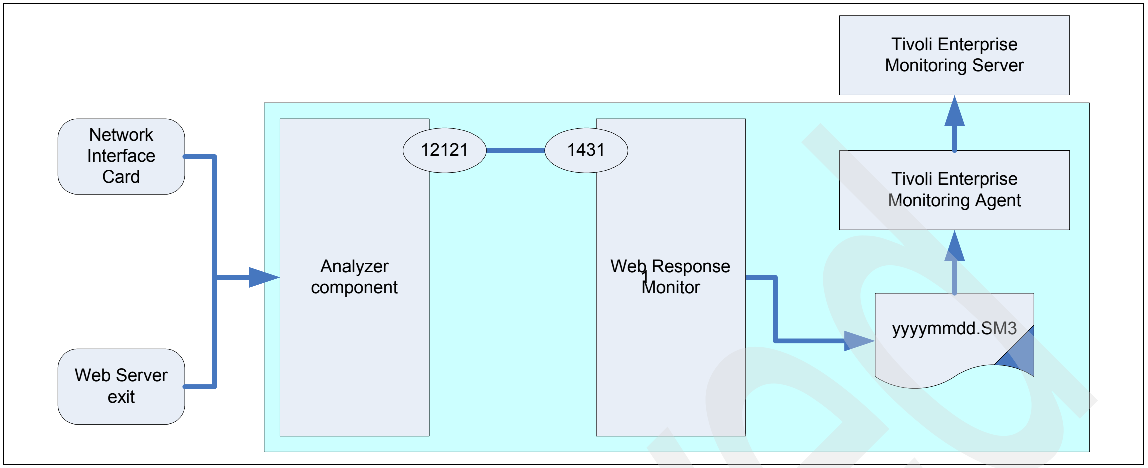


Figure 7-2 Web Response Time agent structure

The components are:

- Analyzer component

  The analyzer operates as a TCP/IP network component logically at the NIC card physical layer. This component observes the network packets and applies a patented response time technique to collect the end-to-end response time of outgoing and incoming sockets. This collects response times all the way down to the requesting source without requiring a software agent at the source.

  In a Windows platform, the analyzer runs as a service called Candle Media Analyzer. In UNIX or Linux, it runs a separate process.

- Web Response Monitor component

  The Web Response Monitor component collects the socket information from the analyzer component and correlates all the subcomponents of the Web transaction. When the complete transaction's information has been collected, the record is written out to a file (*YYYYMMDD*.sm3, where *YYYY* is the year, *MM* is the month, and *DD* is the day). The YYYMMDD.sm3 file can be found under the IBM Tivoli Monitoring directory.

  The analyzer typically communicates to the Web Response Monitor using port 12121 and 1431.

  The Web Response Monitor runs as a service under Windows known as Candle Web Response Monitor - Collection.

- Tivoli Enterprise Monitoring Agent

  The Tivoli Enterprise Monitoring Agent reads the SM3 file and aggregates the data over a user-defined interval (5 minutes by default). It then calculates the information for displays on the workspaces.

  The Tivoli Enterprise Monitoring Agent also starts and stops the analyzer and Web Response Monitor components, so that they are started when the Tivoli Enterprise Monitoring Agent is started, and the Tivoli Enterprise Monitoring Agent stops them when it is stopped.

  > **Note:** When the duration between a stop request and start request is too close (such as using the restart process), the analyzer or Web Response Monitor processes may not be stopped properly.

  The TEMA agent performs all the usual IBM Tivoli Monitoring 6.1 agent activities:

  – Responds to requests for data
  – Logs data to binary history files if historical recording is turned on
  – Runs situations when their intervals expire
  – Heartbeats the TEMS to make sure it is still available
  – Responds to TEMS heartbeats

  The TEMA agent runs as a service under windows called ITCAM for Web Response Time agent, and as a kt5agent process in UNIX and Linux.

7.2.2 Client Response Time Agent

The Client Response Time Agent is installed on a Windows desktop to monitor desktop applications that run on that machine. This machine is typically a user machine on which someone is working with the monitored application.

It analyzes a combination of windows messages and TCP/IP network traffic to compute the user response time for transactions created by monitored GUI applications.

The logical architecture of the Client Response Time agent is shown in Figure 7-3.

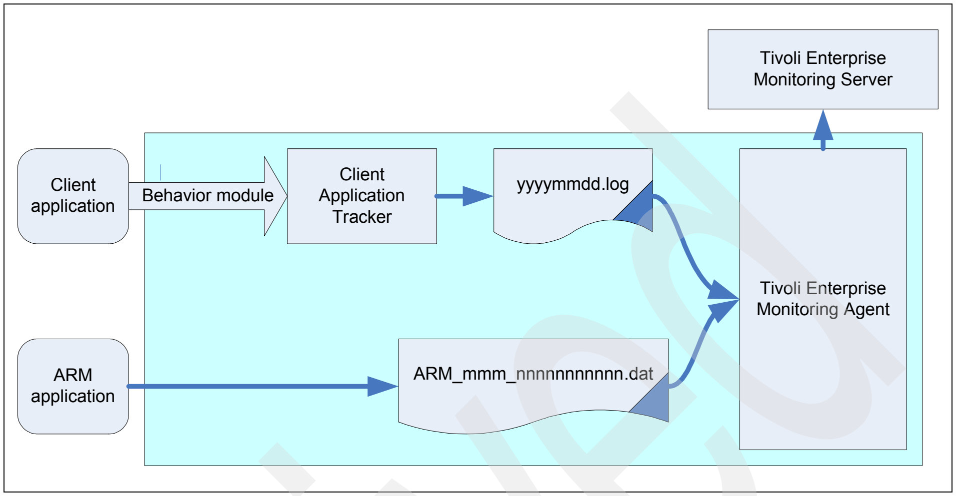


Figure 7-3 Client Response Time agent

The Client Response Time agent consists of:

- Client Application Tracker

  The Client Application Tracker monitors the application as defined by behavior modules. The behavior modules comes as part of the product, or they can be developed by a user using the ETEWatch Customizer component that is available on the IBM Tivoli Open Process Automation Library at:

  “http://catalog.lotus.com/wps/portal/topal

  The product comes with support for:

  – Lotus Notes Versions 6 and 7
  – Microsoft Outlook 2000 and 2003
  – SAP GUI 6.x
  – IBM Personal Communications 5.X (TN3270 protocol only)
  – Hummingbird (TN3270 protocol only)
  – Exceed 11 (TN3270 protocol only)
  – Attachmate Extra 8 TN3270 emulators

  Behavior modules reside in the <itm>\TMAITM6\cat\Mgmt\Collector\TC directory on Windows and have a .dat file suffix.

  The Client Application Tracker component reads the behavior files at startup and monitors the applications defined in those behavior files. It makes use of windows messages and TCP/IP datastream to decipher the start and stop times of transactions caused by the users interacting with the monitored applications. When a transaction record is completed it is written out to a daily log file called *YYYYMMDD*.log (where *YYYY* is the year, *MM* the month, and *DD* the day), which is aggregated in a similar way to the how the Web Response Time agent aggregates its sm3 file. This file resides in <itm>\TMAITM6\cat\Mgmt\Log. In Windows, the Client Application Tracker runs as a service called IBM Client Application Tracker.

- The Tivoli Enterprise Monitoring Agent

  The Tivoli Enterprise Monitoring Agent aggregates the data over a user-defined interval (5 minutes by default) and calculates the information necessary to populate the IBM Tivoli Monitoring tables that are reported in Tivoli Enterprise Portal workspaces.

  The Tivoli Enterprise Monitoring Agent performs all the usual IBM Tivoli Monitoring 6.1 agent activities:

  – Responds to requests for data
  – Logs data to binary history files if historical recording is turned on
  – Runs situations when their intervals expire
  – Sends heartbeats to indicate its availability
  – Responds to Tivoli Enterprise Monitoring Server heartbeats

  In Windows, the Tivoli Enterprise Monitoring Agent runs as a ITCAM for Client Response Time agent service and as the process kt4agent under Linux and UNIX.

  The Tivoli Enterprise Monitoring Agent can also collect Application Response Measurement (ARM) Version 2 and ARM Version 4 records that are produced by any ARM instrumented application. The ARM_*mmmm_nnnnnnnnnn*.dat file (where *mmmm* is the pid of the program creating the ARM records and *nnnnnnnnnn* is the time stamp) is created in the <itm>\TMATITM6\cat\Mgmt\Collector\TC directory on Windows.

7.2.3 Robotic response time agent

The robot response time agent is installed on Windows, Linux, or UNIX to accept response and availability information from the supported robotic runtime environment. The robot runtime environments currently supported are:

- Rational Performance Tester
- Rational Robot
- Command-line interface (CLI)
- Mercury LoadRunner

The logical architecture of the agent is displayed in Figure 7-4.

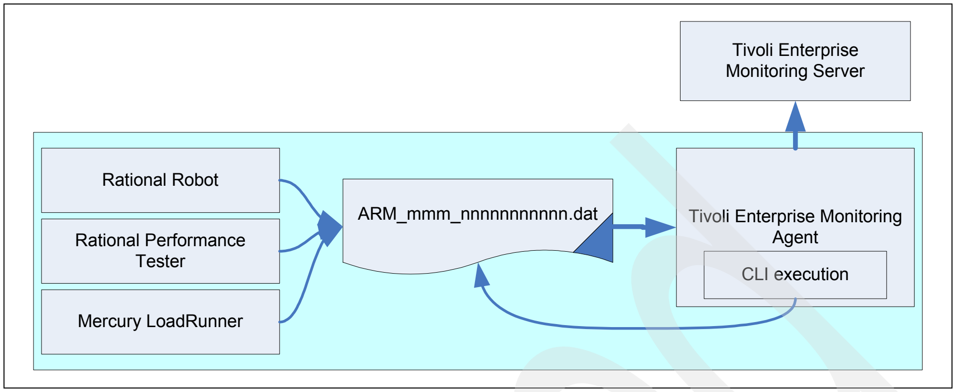


Figure 7-4 Robotic response time agent

The agent collects response and availability information in the form of ARM V2 and ARM V4 records and logs these to the file. The full name of the file is ARM_`nnnn_mmmmm`.dat, where *mmmm* is the pid of the program creating the ARM records and *nnnnnnnnnn* is the time stamp. The robots execute at the interval defined at install time (the default is every 15 minutes). This can be changed subsequently.

The agent runs as the service ITCAM for Robot Response Time under Windows and as the process kt6agent under UNIX and Linux.

7.2.4 End User Response Time Dashboard agent

The End User Response Time Dashboard agent consolidates the historical data from the all other ITCAM for Response Time agents:

- Client response time
- Robotic response time
- Web response time

As a consolidator, there is only one End User Response Time Dashboard agent in each IBM Tivoli Monitoring environment. The End User Response Time Dashboard agent also acts as the file depot for the robotic scripts.

The ITCAM for End User Response Time Dashboard agent integrates up to eight hours of historical data from the Tivoli Data Warehouse (TDW). In Windows, the agent runs as a service called ITCAM for End User Response time and as a process in Linux and UNIX called kt3agent.

The product provides automated ways of distributing scripts to the agents. Manual methods are also documented. There are two deployments required:

- “Deploying the script to the repository” on page 342
- “Deploying the script from the repository to the agent” on page 343

Figure 7-5 shows the high-level view of the architecture.

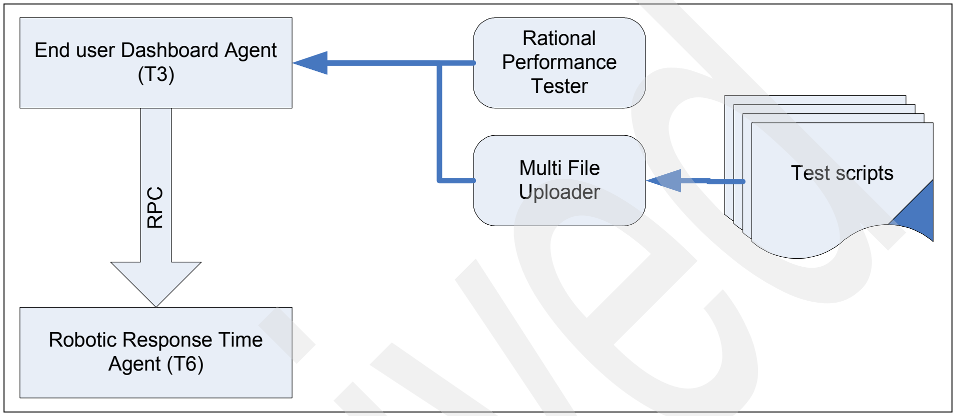


Figure 7-5 Script deployment

Deploying the script to the repository

Deploying the script to the repository can be achieved in two ways:

- Automated deployment to the repository

  The repository is a directory that is part of the ITCAM for Response Time Dashboard agent. It is typically found at <*itm*>/kt1depot/T3/<*type*>, where <*itm*> is the location of the IBM Tivoli Monitoring install directory and <*type*> is dependent on the type of script, such as RPT, CLI, ROBOT_GUI, and ROBOT_VU.

  Rational Performance Tester has the ability to send packaged scripts directly to the dashboard agent. The Rational Performance Tester workbench contains an export destination for ITCAM for Response Time. This uses the SOAP mechanism to send the scripts to the repository. It uses port 1976 by default.

  Automated sending of CLI, Load Runner, or Rational Robot scripts to the repository uses the Multiple File Uploader (MFU) facility. This can be accessed from the Tivoli Enterprise Portal navigator or using the remote Java WebStart facility. The MFU also uses SOAP calls from port 1976 by default.

  For Rational Robot, the Multi File Uploader discovers scripts stored under <itm>\tmaitm6\app\genwin\project\<projectname>\TestData\TestDataStore\DefaultTestScriptDatastore\TMS_Scripts.

- Manual deployment to the repository

  The scripts (as zip files) can be manually placed directly in the repository directory <*itm*>/kt1depot/T3/*type*, where <*itm*> is the location of the IBM Tivoli Monitoring install directory and *type* is dependent on the type of script, such as ROBOT_GUI, ROBOT_VU, RPT, and CLI.

Deploying the script from the repository to the agent

The scripts can be deployed in two ways between the respository and the agent:

- Automatically deploying the script

  At user-defined intervals (15 minutes by default), the End User Response Time Dashboard agent pulls the new scripts from the repository to the executing agent. They are copied to a directory called <itm>/kt1depot/t6/<*type*>, where *type* is the type of script, such as CLI, RPT, Robot_GUI, and Robot_VU.

- Manual deployment from the repository to the agent

  Scripts can be manually distributed by copying the scripts from the repository at <itm>/kt1depot/T3/<type> to the machine running the robotic agent under <itm>/kt1depot/t6/<type>.

Both the manual and automated deployment to the agent make use of an RPC capability between the repository and the ITCAM for Robotic Response Time agent. This capability exists in the Tivoli Enterprise Monitoring Server, and it is installed automatically on Windows and needs to be manually installed on other platforms.

7.3 Installation overview

In our environment, we installed all of the four agents provided with the ITCAM for Response Time 6.2. You do not need to install all of the agents provided. You should install the agent that fits your needs, as described in Table 7-1.

Table 7-1 Choosing which monitoring agent to use

If you want to	Use this agent
Monitor real user response times for Lotus Notes, Microsoft Outlook, or applications running in a Citrix or Terminal Services environment. Understand real user client experience. Monitor custom Windows applications. Monitor real user 3270 transactions.	ITCAM for Client Response Time

If you want to	Use this agent
Upload and store the robotic scripts that you use with ITCAM for Robotic Response Time. This agent is required in order to upload the IBM Rational Performance Tester scripts you recorded. Have a consolidated enterprise view of all application performance and availability. Create custom roles to limit access to application data.	ITCAM for End User Response Time Dashboard
Run an existing Mercury Load Runner script and monitor the results with this product. Run robotic monitoring for Web applications, Siebel, SAP, and Citrix. Run a custom application, script, or command and see results.	ITCAM for Robotic Response Time
Monitor real user transactions on an HTTP server. Discover new URLs.	ITCAM for Web Response Time

Once you have decided what you want to monitor and which agents to install, verify the prerequisites at:

http://publib.boulder.ibm.com/tividd/td/ITCAMRT/prereq62/en_US/HTML/Version62.html

For this deployment guide, we installed the Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, Tivoli Enterprise Portal Desktop Client, Warehouse Proxy, Warehouse Summarization and Pruning agent, and Warehouse Database on the same server. This type of installation is recommended only for proof-of-concepts or small-size environments. For more information about IBM Tivoli Monitoring design considerations see *IBM Tivoli Monitoring Installation and Setup Guide,* GC32-9407. You should consult also the *Deployment Guide Series: IBM Deployment Guide Series: IBM Tivoli Monitoring 6.1*, SG24-7188. That is a step-by-step deployment guide for IBM Tivoli Monitoring 6.1 that covers small to large environments and discusses best practices for a deployment plan.

For *each* ITCAM for Response Time Agent you are going to install you must first install application support files on the Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, and Tivoli Enterprise Portal Desktop Client.

You just need to install the application support files *once* for *each* ITCAM for Response Time Agent that you are going to install. For example, if you plan to install four ITCAM for Robotic Response Time Agents on your environment to play back Rational Performance Tester scripts from different locations, you just need to install ITCAM for Robotic Response Time application support files once.

We discuss application support files installation in 7.3.1, "Installing application support files for Windows" on page 346. The process is the same for all of the four agents for ITCAM for Response Time (End User Dashboard, Client, Web, and Robotic), so we demonstrate it only once.

You can install an agent using different methods:

- We install the End User Response Time Dashboard agent on the same server where the Tivoli Enterprise Portal Server, Tivoli Enterprise Monitoring Server, and Tivoli Enterprise Portal Desktop Client are installed. See 7.3.2, "Installing an agent with a wizard" on page 351.
- We install the other agents using the remote deployment on machines that have IBM Tivoli Monitoring Operating System agent installed. The installation process for the Client Response Time agent, Web Response Time agent, and Robotic Response Time agent is the same. See 7.3.3, "Installing using remote deployment" on page 357.

Note: If you have previously installed ITCAM for Response Time Tracking Version 6.1, you must uninstall it from your environment and verify that the following DLLs are deleted before installing ITCAM for Robotic Response Time or ITCAM for Client Response Time:

- c:\windows\system32\libarm4.dll
- c:\windows\system32\libarm32.dll
- c:\windows\system32\armjni4.dll
- c:\windows\system32\armjni.dll
- c:\windows\system32\armcli.dll
- c:\windows\system32\libarm4net.dll

The installation environment that we use is shown in Figure 7-6.

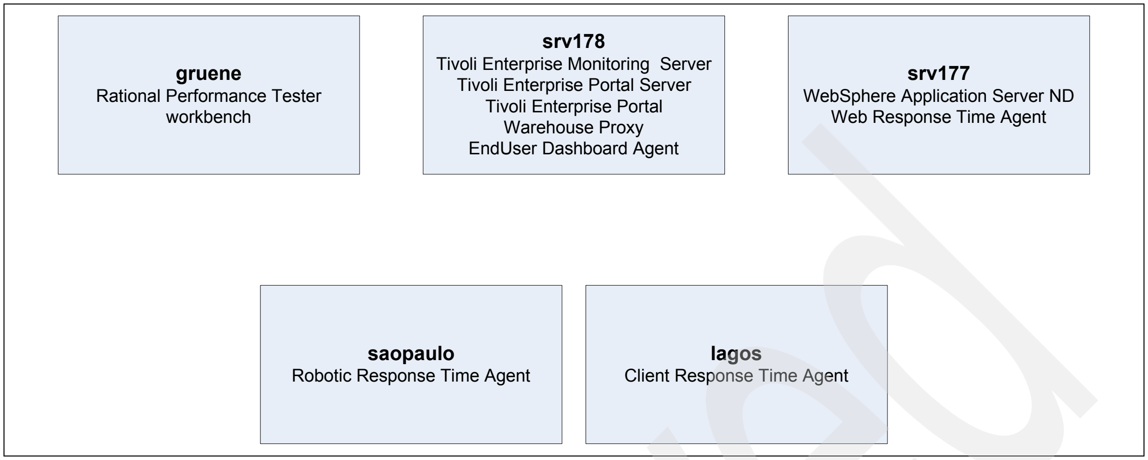


Figure 7-6 Sample environment

7.3.1 Installing application support files for Windows

Before you can view data collected by monitoring agents, you must install and enable application support for the agents. Application support files provide agent-specific information for workspaces, helps, situations, template, and other data. This section describes how to install application support files for Windows.

All monitoring agents require that application support be configured on all instances of the following infrastructure components:

- Tivoli Enterprise Monitoring Server (both hub and remote monitoring serves)
- Tivoli Enterprise Portal Server
- Tivoli Enterprise Portal Desktop Client

See your IBM Tivoli Monitoring administrator and collect the Tivoli Enterprise Portal Server host name, Tivoli Enterprise Monitoring Server name, Tivoli Enterprise Monitoring Server host name, and the protocol settings for the Tivoli Enterprise Monitoring Server that you are going to connect to.

> **Note:** The monitoring server will be stopped during this process.

The installation of the application support file installs the components shown in Figure 7-7.

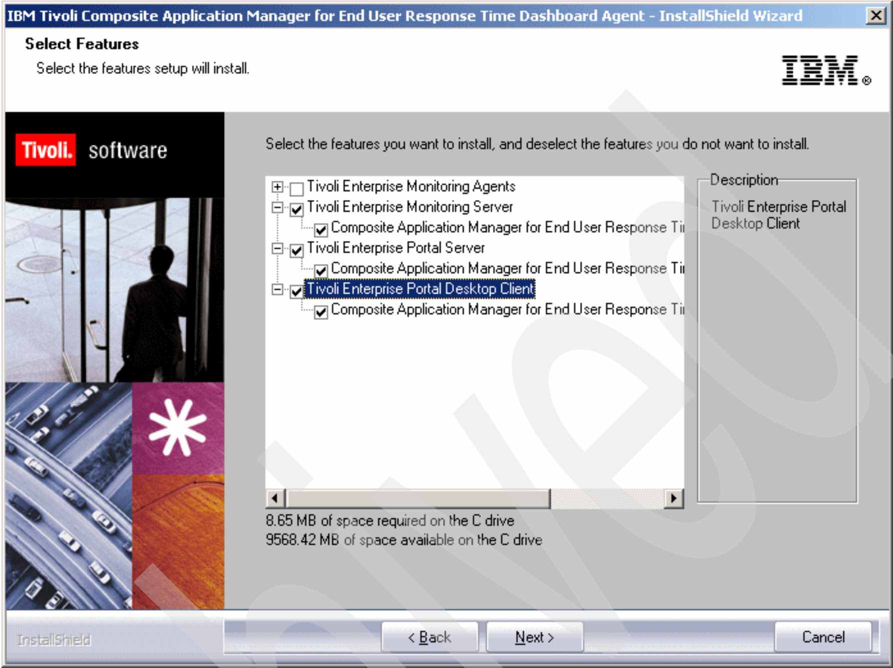


Figure 7-7 Select features

We recommend that you implement the remote deployment option for the Web Response Time agent, Robotic Response Time, and Web Response Time agents, as you may need to install them on multiple machines. The remote deployment installation is discussed in 7.3.3, "Installing using remote deployment" on page 357. This remote deployment option is available if the Tivoli Enterprise Monitoring Server is detected in the current machine. Also, the remote deployment uses the Operating System agent component to perform the remote deployment.

After installation is complete, you can start the configuration. The options for the configuration are shown in Figure 7-8.

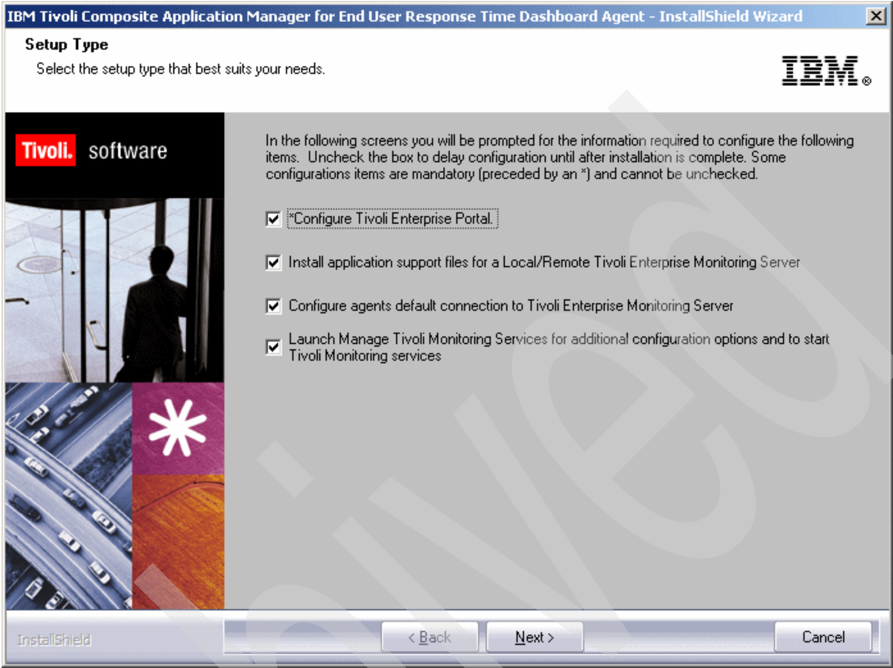


Figure 7-8 Configuration choices

The process is:

1. The Tivoli Enterprise Portal and Tivoli Enterprise Portal Server configuration starts. It uses the host name of the Tivoli Enterprise Portal Server and rebuilds the Tivoli Enterprise Portal Server presentation files.

2. Install application support for the Tivoli Enterprise Monitoring Server:

   a. First to check the communication protocols and the parameters for each defined communication protocol, as shown in Figure 7-9.

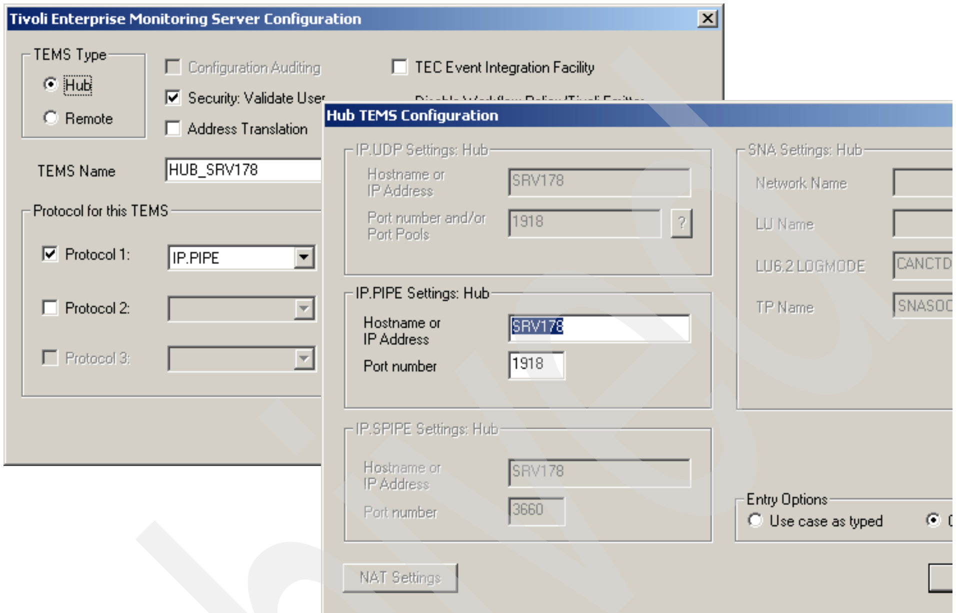


Figure 7-9 Tivoli Enterprise Monitoring Server communication protocols

b. Check the Tivoli Enterprise Monitoring Server location and add the application support files (execute SQL files). This file must complete with rc=0. See Figure 7-10.

> **Note:** If the Application Support Addition Complete window is not displayed after a while, look in the IBM\ITM\CNPS\Logs\seedk*pp*.log files (where *pp* is the two-character code for each monitoring agent) for diagnostic messages that help you determine the cause of the problem.

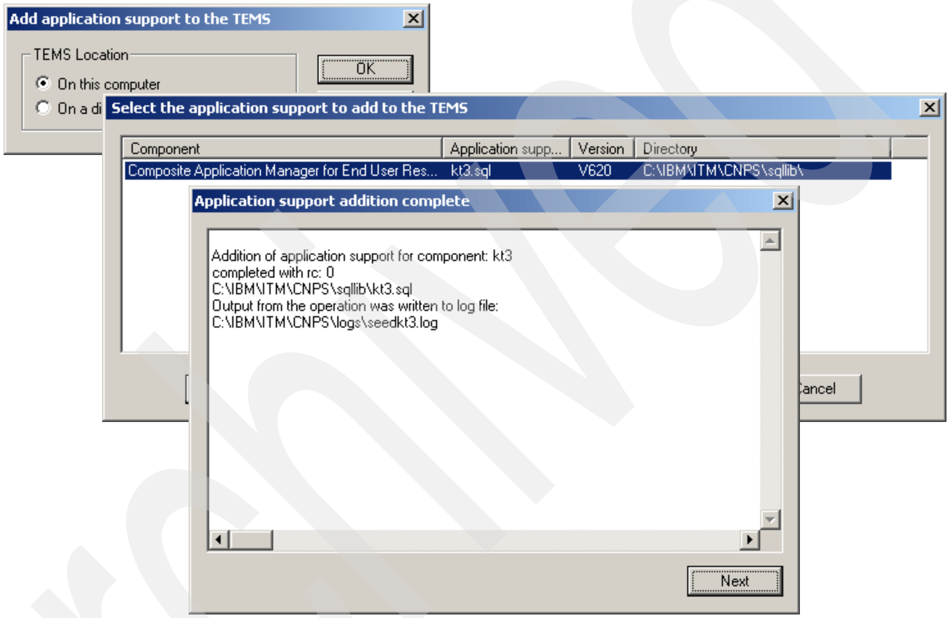


Figure 7-10 Selecting Tivoli Enterprise Monitoring Server location

3. The next figures show the configuration of the agent default connection to the Tivoli Enterprise Monitoring Server. Specify the communication protocol default and communication parameter defaults for the agent to use when it communicates with the Tivoli Enterprise Monitoring Server and click **OK**, as shown in Figure 7-11.

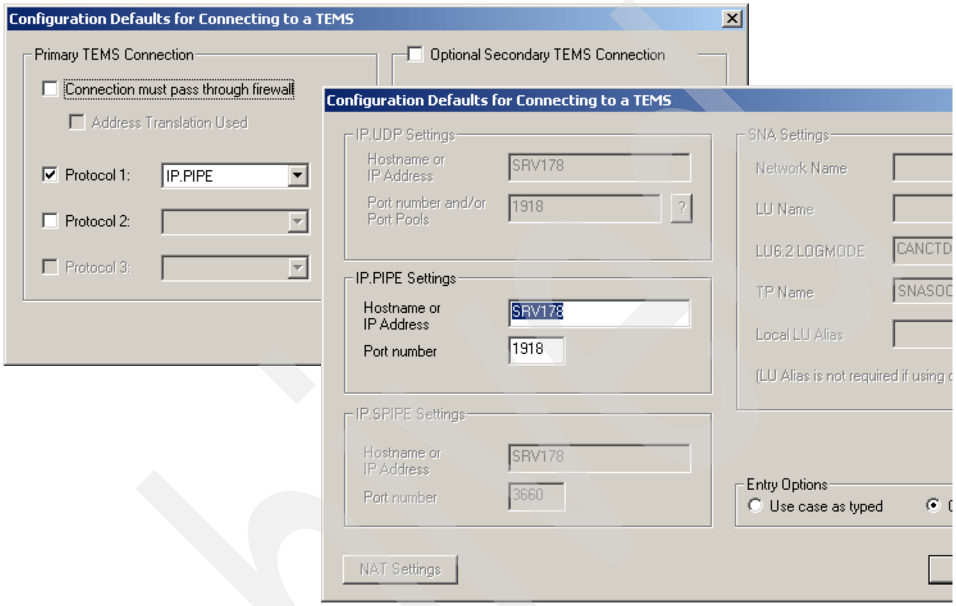


Figure 7-11 Communication protocol defaults

Note: Repeat these steps for all types of agent that you are going to install in your environment. In a deployment environment we repeated the steps to install application support files for the Robotic, Web, and Client Response Time agents.

7.3.2 Installing an agent with a wizard

This chapter describes how to install the End User Response Time Dashboard on Windows using the installation wizard.

The End User Response Time Dashboard works with data from the Tivoli Data Warehouse. Therefore, we recommend that it is installed close to the Tivoli Data Warehouse, but on a separate system in the same subnet. It could be on the same machine as the Warehouse Proxy or Summarization and Pruning agents, as they all access the warehouse.

For demonstration purposes we install the Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, Tivoli Data Warehouse, Warehouse Proxy Agent, and Summarization and Pruning Agent all on one server.

Before you install:

1. Install the agent as administrator.
2. Install only one ITCAM for End User Response Time Dashboard Tivoli Enterprise Monitoring Agent in the monitoring environment. The ITCAM for End User Response Time Dashboard is also the robotic script file depot. There should be only one file depot per IBM Tivoli Monitoring environment.
3. See your IBM Tivoli Monitoring administrator and collect the Tivoli Enterprise Portal Server host name, Tivoli Enterprise Monitoring Server name, Tivoli Enterprise Monitoring Server host name, and the protocol settings for the Tivoli Enterprise Monitoring Server that you are going to connect to. Also get the Tivoli Data Warehouse configurations.
4. Install application support files, as described in 7.3.1, “Installing application support files for Windows” on page 346.

The Tivoli Enterprise Monitoring Agent installation is a two-part process, the installation and the configuration. The installation part would detect existing components and decide whether this is a fresh installation or maintenance. It also detects any presence of Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, or Tivoli Enterprise Portal that allows you to install the application support file accordingly.

For maintenance mode, in which some of the components are already installed, you *should not* deselect items, otherwise you will uninstall them. As we install the End User Dashboard Agent on the same machine as the Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server, it already has the application support files. We use the default for the installation wizard. Figure 7-12 shows the installation summary.

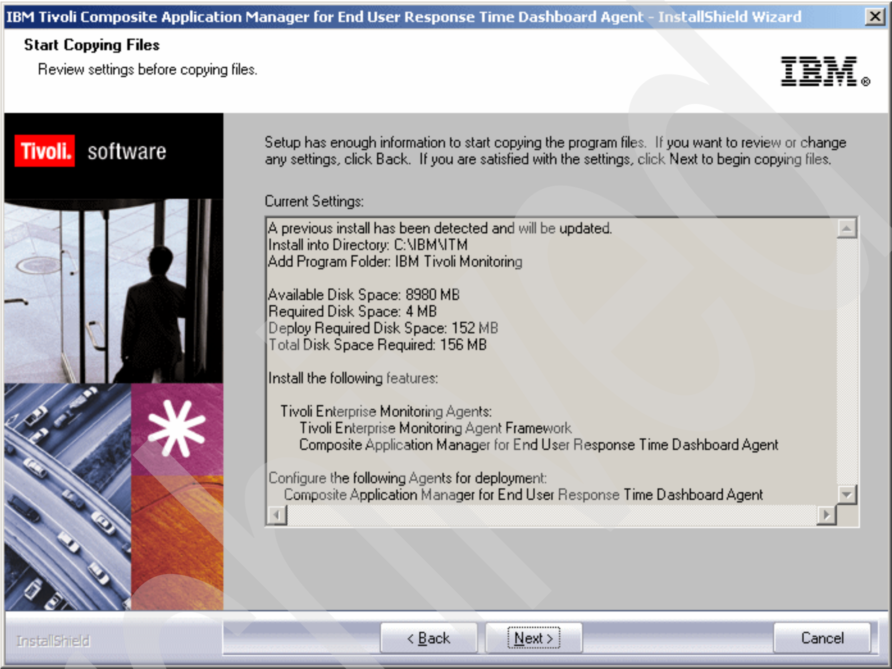


Figure 7-12 Installation summary

When the configuration steps start, we select to only launch Manage Tivoli Monitoring Services to configure the End User Dashboard agent. See Figure 7-13.

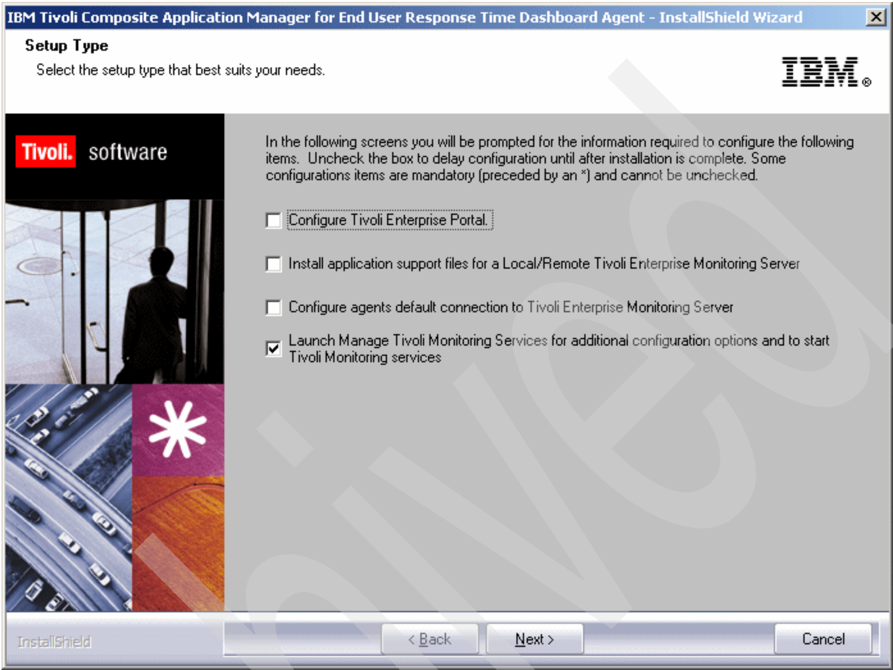


Figure 7-13 Configuration options

The agent configuration is shown in the subsequent list:

- Figure 7-14 shows the Tivoli Data Warehouse configuration.

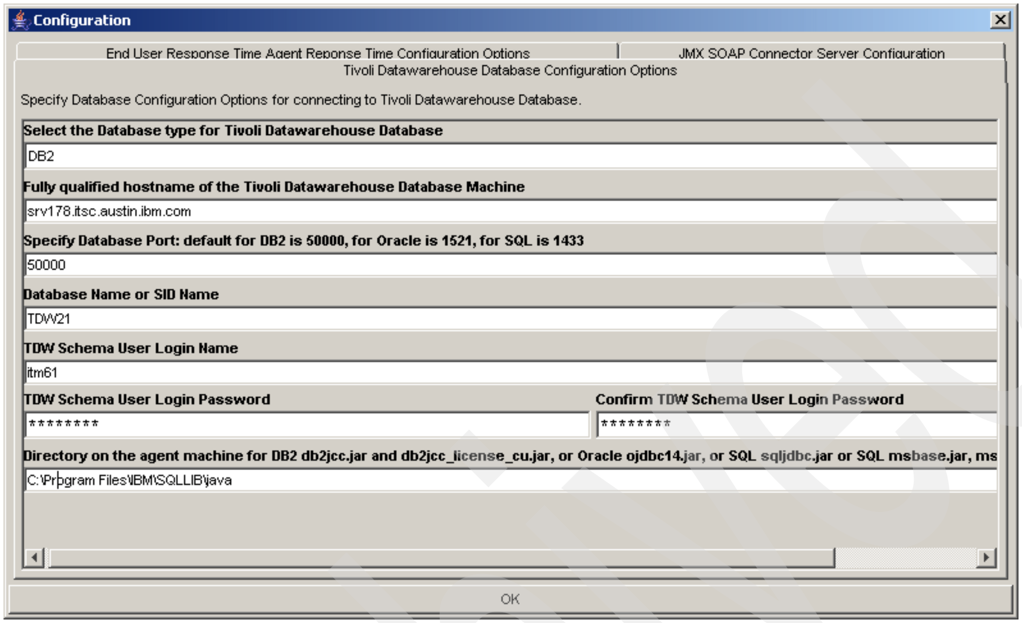


Figure 7-14 Warehouse database configuration

- Figure 7-15 shows the JMX SOAP Server Configuration tab. This configuration is used for the file uploader utility, either the Multi-file Uploader or the Rational plug-in.

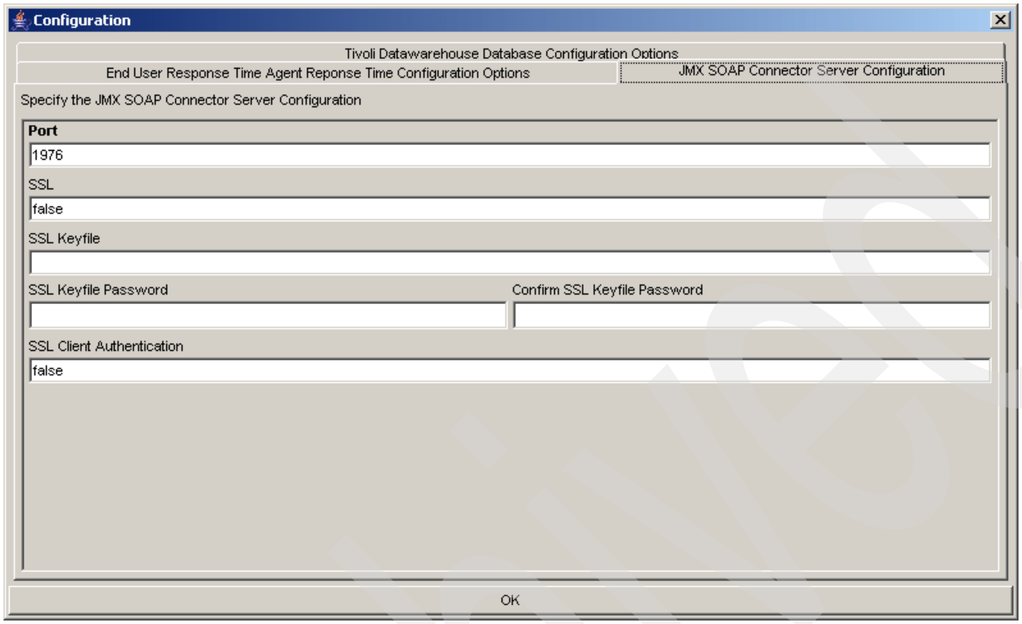


Figure 7-15 JMX SOAP connector server configuration

- Figure 7-16 shows the End User Time Agent Response Time Configuration options.

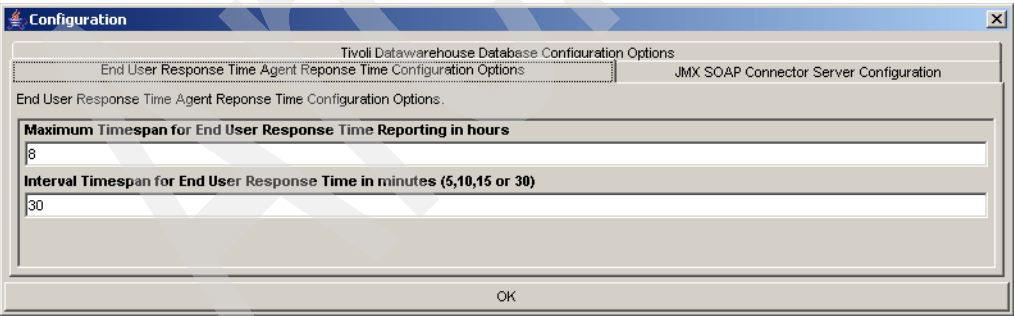


Figure 7-16 End User Response Time Agent configuration

7.3.3 Installing using remote deployment

The installation for other agents can be performed using remote deployment, as there may be more than one agent in your environment for the type. ITCAM for Response Time has these agents:

- Client Response Time
- Web Response Time
- Robotic Response Time

Be sure that you have all the requirements of a specific agent that you want to deploy, such as:

- The Robotic Response Time agent requires Rational Robot to already be installed if you want to play back the Rational Robot script.
- The Windows-based Web Response Time agent requires the Windows Network Monitor to be installed. See 7.3.4, “Installing Windows Network Monitor” on page 361.

For this example, we install the ITCAM for Client Response Time Agent at LAGOS, which already has an IBM Tivoli Monitoring Windows OS agent running. We also already loaded the IBM Tivoli Monitoring repository when we installed the application support files, as described in 7.3.1, “Installing application support files for Windows” on page 346.

1. Log on to the Tivoli Enterprise Portal and find the server where you want to deploy ITCAM for Client Response Time. In this demonstration, we want to deploy it at LAGOS, which is accessible from the Tivoli Enterprise Portal.

2. Right-click **LAGOS** then click **Add Managed System**, as shown in Figure 7-17.

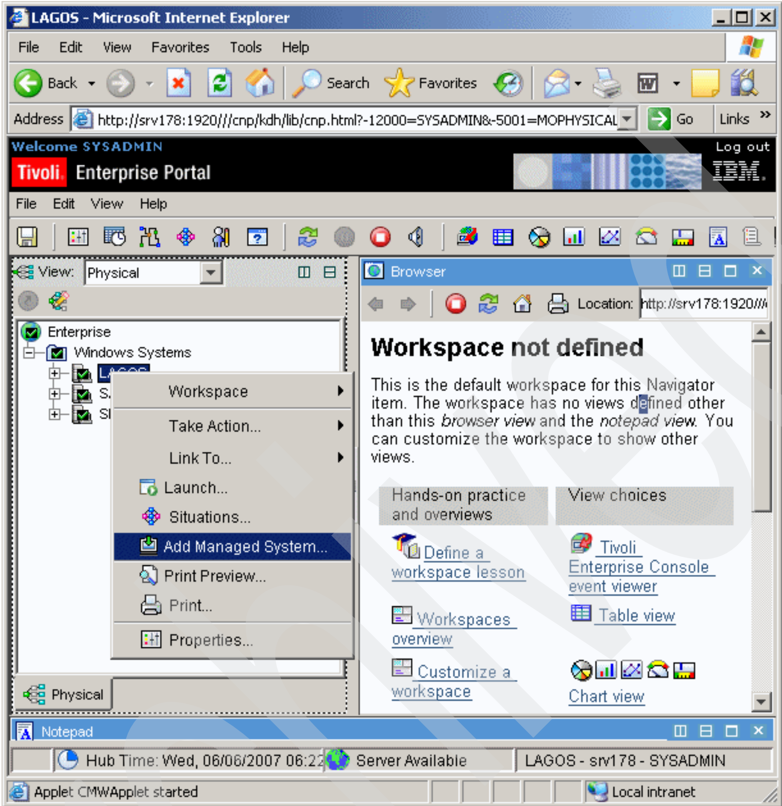


Figure 7-17 Add managed system

3. Select **Composite Application Manager for Client Response Time Monitoring Agent** and click **OK**, as shown in Figure 7-18.

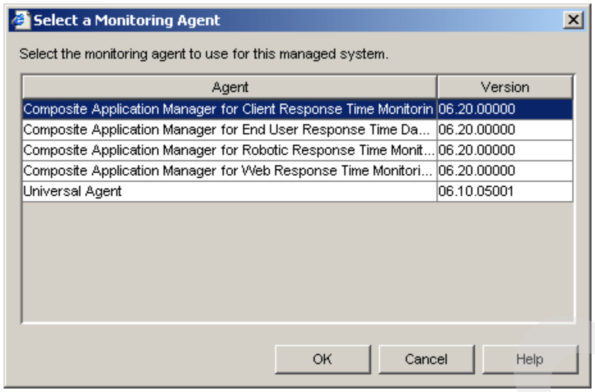


Figure 7-18 Select a Monitoring Agent

4. The configuration for the new managed system is shown in Figure 7-19. Leave the defaults for both tabs and click **Next**.

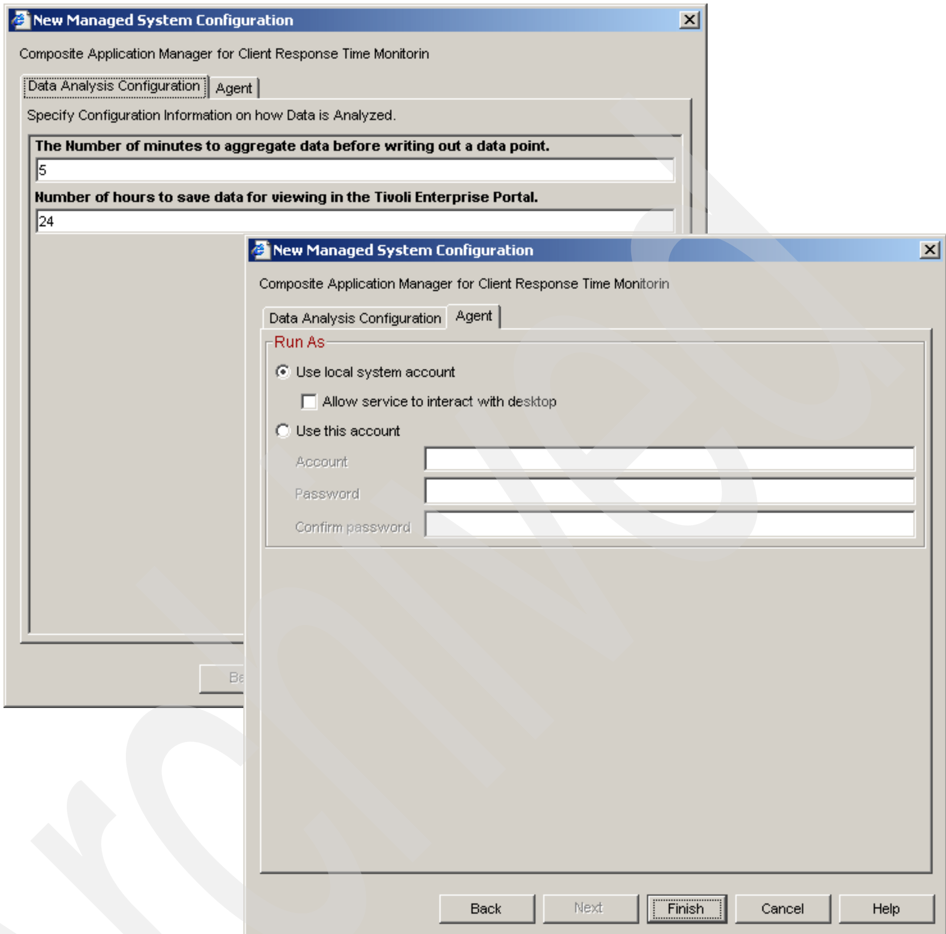


Figure 7-19 New managed system configuration

5. Wait until the configuration finishes, then update the navigator by clicking the arrow that you see inside a green balloon underneath the physical view.

6. The installation has finished and the Client Response Time Workspace can be accessed by the portal, as shown in Figure 7-20.

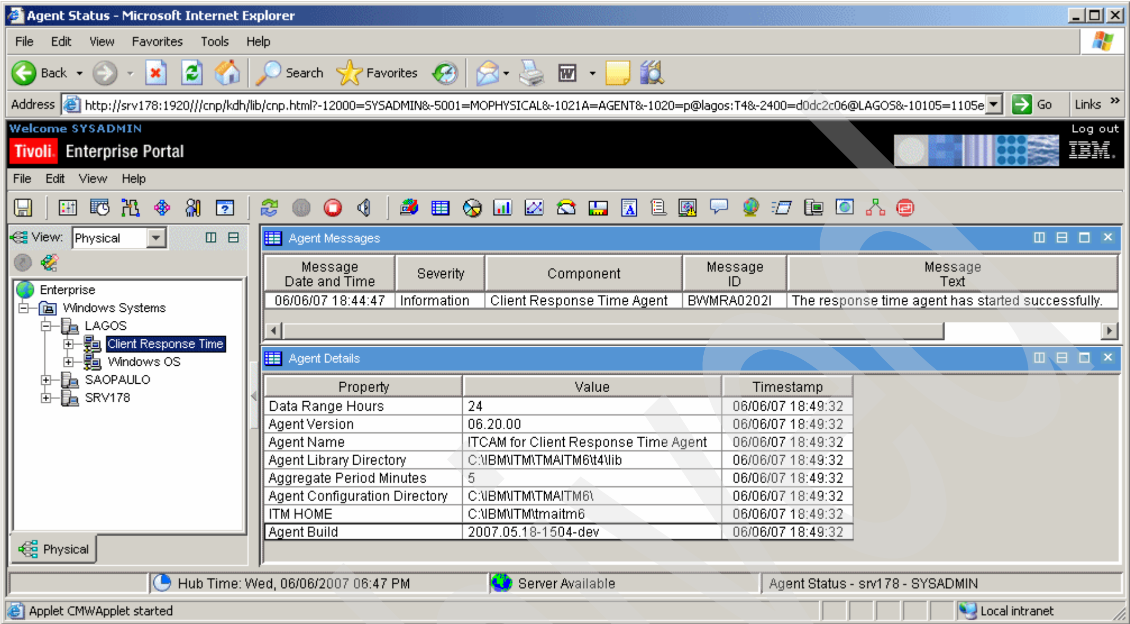


Figure 7-20 Client Response Time workspace

7.3.4 Installing Windows Network Monitor

Before installing the ITCAM for Web Response Time agent on a Windows platform, you must install the Windows Network Monitor.

1. Do one of the following:
   - For Windows 2000: Click **Control Panel** → **Network and Dial-up Connections** → **Local Area Connection**.
   - For Windows 2003 and Windows XP: Click **Control Panel** → **Network Connections** → **Local area Connection**.
2. Right-click **Local Area Connection**.

3. In the pop-up menu, click **Properties**, as shown in Figure 7-21.

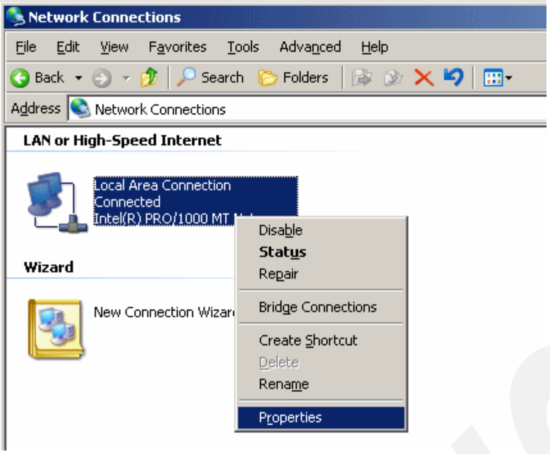


Figure 7-21 Network Connections properties

4. In the Local Area Connection Properties window, click **Install** if there is no Network Monitor Driver available, as shown in Figure 7-22.

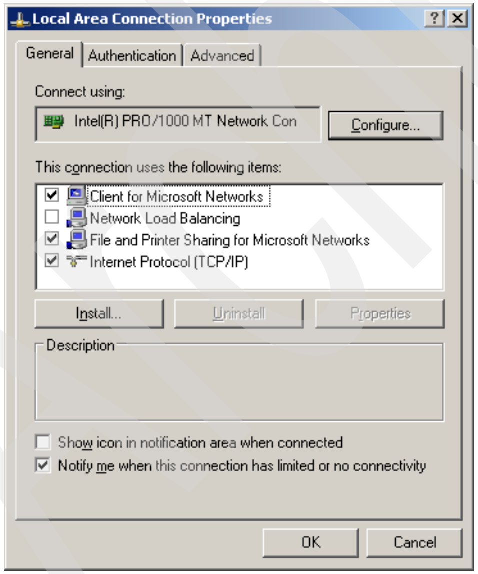


Figure 7-22 Network drivers

5. Select **Protocol** from the Select Network Component window and click **Add**, as shown in Figure 7-23.

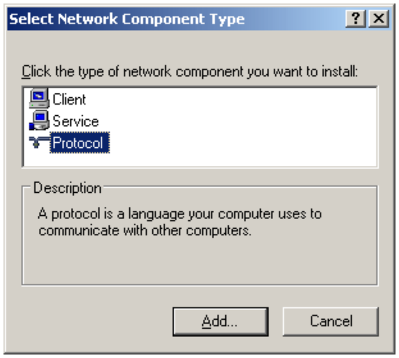


Figure 7-23 Choosing protocol network component

6. Select **Network Monitor Driver** from the Select Network Protocol window and click **OK**, as shown in Figure 7-24.

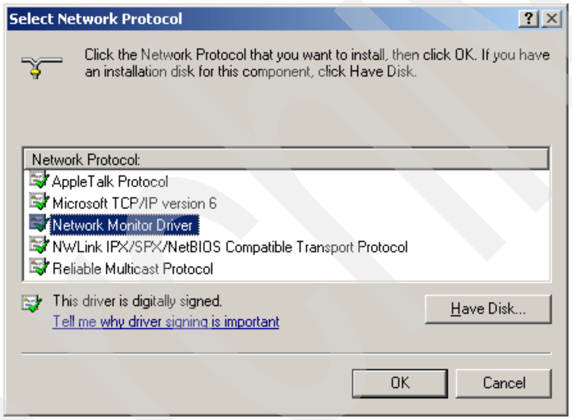


Figure 7-24 Select network monitor driver protocol

7. After the Network Monitor Driver is displayed on the Local Area Connections Properties window, click **Close**, as shown in Figure 7-25.

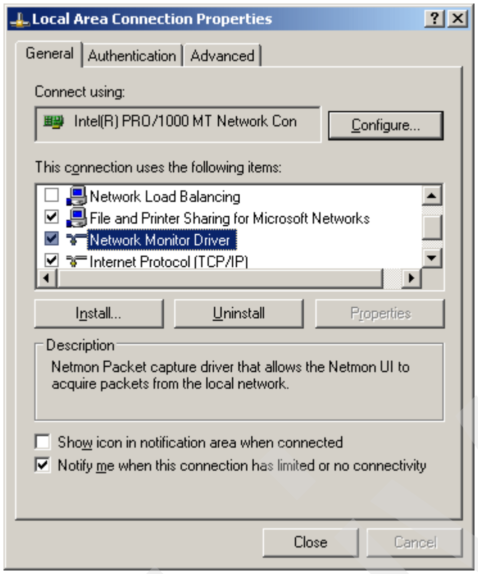


Figure 7-25 Network Monitor driver installed

7.3.5 Installing Rational Performance Tester Workbench

This section describes the installation of the Rational Performance Tester Workbench on a Windows workstation. The Rational Performance Tester Workbench is needed to record and upload Rational Performance Tester tests. You should install the Rational Performance Tester Workbench if you want to monitor Web applications, SAP, Siebel, or Citrix applications.

You can install the Rational Performance Tester Workbench on any system from which you want to record the tests. You do not need to install it on the same system as the Robotic Response Time Agent. The Rational Performance Tester Workbench needs a connection to the End User Response Time Dashboard Agent. Make sure that there are no firewalls between them.

For the installation we used the following media:

- IBM Tivoli Composite Application Manager for Response Time V6.2: Rational Performance Test V7.0 Integration Support Windows, English. This contains the setup_RPT.exe that uses the Rational Performance Installation Media (Part 1, 2, and 3) and performs the installation of all required components:
  - IBM Installation Manager 1.0
  - IBM Rational Performance Tester 7.0
  - IBM Rational Performance Tester 7.0 license key
  - ITCAM for Response Time 6.2 Integration plug-in
- IBM Rational Performance Tester V7.0 Multiplatform Multilingual Part 1
- IBM Rational Performance Tester V7.0 Multiplatform Multilingual Part 2
- IBM Rational Performance Tester V7.0 Multiplatform Multilingual Part 3

If you are downloading the software to a disk, we recommend that you to uncompress all four installation images on the same directory, for example, C:\ITCAMRational, so that the installation wizard does not ask you for the Rational media. You can check the supported platforms for the Rational Performance Tester at:

http://www-306.ibm.com/software/awdtools/tester/performance/sysreq/index.html

Install the software from the RPT\w32-ix86\disk1 directory of the downloaded installation images. This is where you uncompressed IBM Tivoli Composite Application Manager for Response Time V6.2: Rational Performance Test V7.0 Integration Support Windows, English. Run setup_RPT.exe. You can use the default options. See Figure 7-26 for some of the installation dialogs.

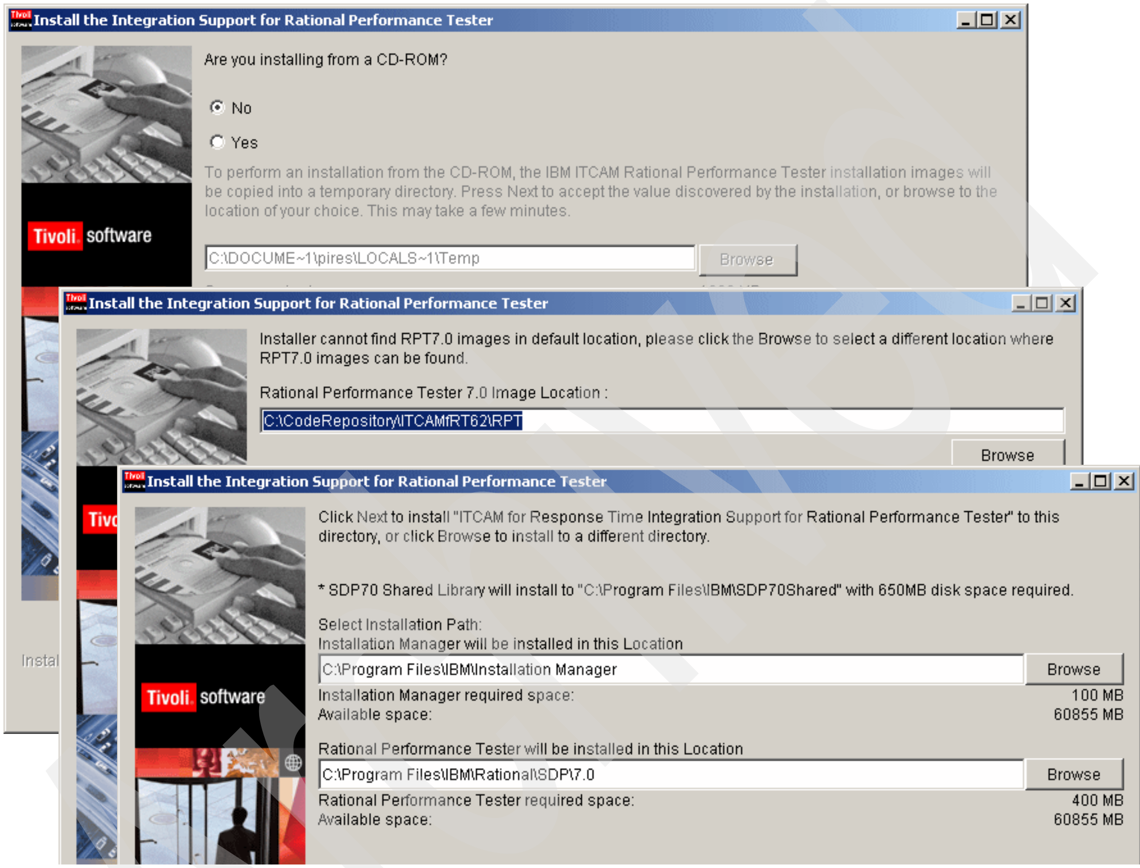


Figure 7-26 Rational Performance Tester installation dialogs

7.4 Usage scenarios

This chapter aims to demonstrate the features of ITCAM for Response Time. The demonstration uses the trader application. A description of the trader application is provided in <app>.

The trader application is a Web-based J2EE application that has a back end processing in CICS. The application can be accessed from the Web browser or

3270 interface. Based on this structure, we decided to collect the trader application response time using the following methods:

- Using Rational Performance Tester to run a transaction simulation to the Web interface
- Using the Web Response Monitor to collect HTTP traffic performance on the Web server
- Using the Client Application Tracker to collect CICS's 3270 interface response time

Interaction with ITCAM for Response Time is mainly performed through the workspaces of the Tivoli Enterprise Portal. Except for the recording portion of Rational Performance Tester using the Rational Performance Tester Workbench, all activities are performed using the Tivoli Enterprise Portal.

In 7.4.1, "Workspaces and interfaces" on page 367, we describe the Tivoli Enterprise Portal interface. Further along, we start to build the environment using:

- Recording the Rational Performance Tester script in 7.4.2, "Using Rational Performance Tester" on page 369
- Running and monitoring the Rational Performance Tester script in 7.4.3, "Monitoring Robotic Response Time" on page 385
- Working with the Web response time in 7.4.4, "Monitoring Web Response Time agent" on page 402
- Collecting 3270 transaction performance in 7.4.5, "Using the Client Response Time Agent" on page 408

7.4.1 Workspaces and interfaces

The workspaces for ITCAM for Response Time in the physical tree are organized based on agent. Each agent type provides a different set of workspaces that are collected under it. The End User Dashboard agent is a unique agent in that it only collects data from the historical data warehouse, instead of a live system. The dashboard provides an overall view of the cross-enterprise response time.

The hierarchy tree of the workspace for ITCAM for Response Time agents is shown in Figure 7-27.

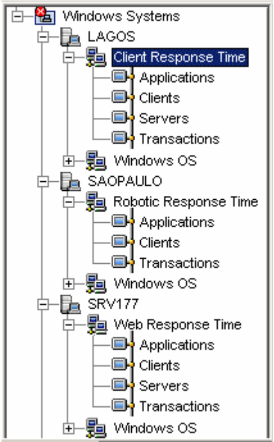


Figure 7-27 Workspace structure

As shown in Figure 7-27, the workspace for ITCAM for Response Time is divided by:

- Application: An application is defined as a larger entity that encompasses multiple transactions.
- Transaction: individual element of an application.
- Client: the user machine or IP addresses that accesses the application.
- Server: the request processing server, such as 3270 host, application server, machine and so on. The robotic response time does not have a server aggregation, as the transaction can be directed to any server within the robotic script.

Furthermore, the end user dashboard workspace aggregates information from all other ITCAM for Response Time agents. The end user dashboard workspace is shown in Figure 7-28.

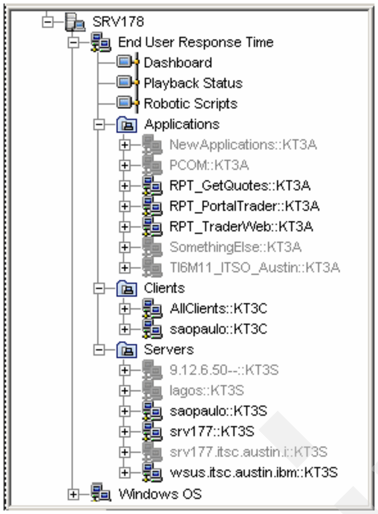


Figure 7-28 Workspace structure

The dashboard contains the overall summary or aggregate of all ITCAM for Response Time agent status. The playback status and robotic scripts workspace are related to the robotic script depot. The application, client, and server workspaces show the historical status of the available response time measurement broken down by application or client or server.

7.4.2 Using Rational Performance Tester

Before you start you must install the Rational Performance Tester Workbench on a workstation, as described in 7.3.5, “Installing Rational Performance Tester Workbench” on page 364.

Rational Performance Tester monitors Web applications, SAP, Citrix, and Siebel using protocol-based record and playback. Rational Performance Tester features include:

- Immediate productivity by hiding all complexity. No coding is required.
- An integrated solution that is built on Rational Application Developer.
- Advanced data access and manipulation.
- Automatic data correlation and synchronization of input parameters.
- Verification points for content matching and response code checking.

- Automatic ARM instrumentation for HTTP transactions to correlate with downstream applications and resources for problem isolation.
- Improved scalability allows for higher playback rates and more concurrent playbacks on a single system, which limits costs of licenses and maintenance.
- Network-level statistics for DNS, SSL Connect time, Server Response Time, and Delivery.
- Rational Robot VU functionality, including a scalable and concurrent playback solution of HTTP transactions and a generic framework that allow you to add new protocols.

> **Note:** You cannot migrate a previously recorded Rational Robot VU script to RPT scripts. You must record new scripts with the Rational Performance Tester.

In this section, we create an HTTP script to access our trader application. We record a script called RPT_TraderWeb. This name is important, as it characterized to the target system, not the agent host name on which the script would run. The process is:

1. Open the Rational Performance Tester:
   a. Click **All Programs** → **RPT** → **IBM Rational Performance Tester** → **IBM Rational Performance Tester**.
   b. If this is the first time you open Rational Performance Tester, select a workspace in which to store your projects, as shown in Figure 7-29. Click **OK**.

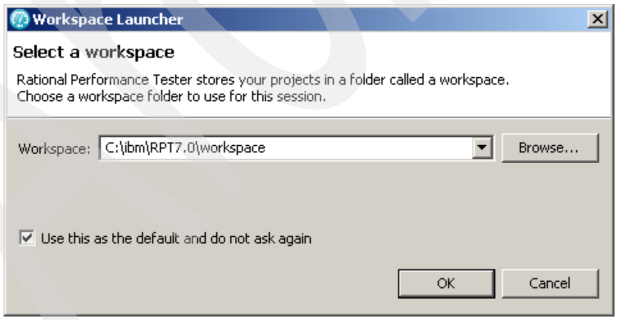


Figure 7-29 Workspace launcher

2. Create a new transaction recording:

   a. Select **File** → **New** → **Test From Recording**, as shown in Figure 7-30.

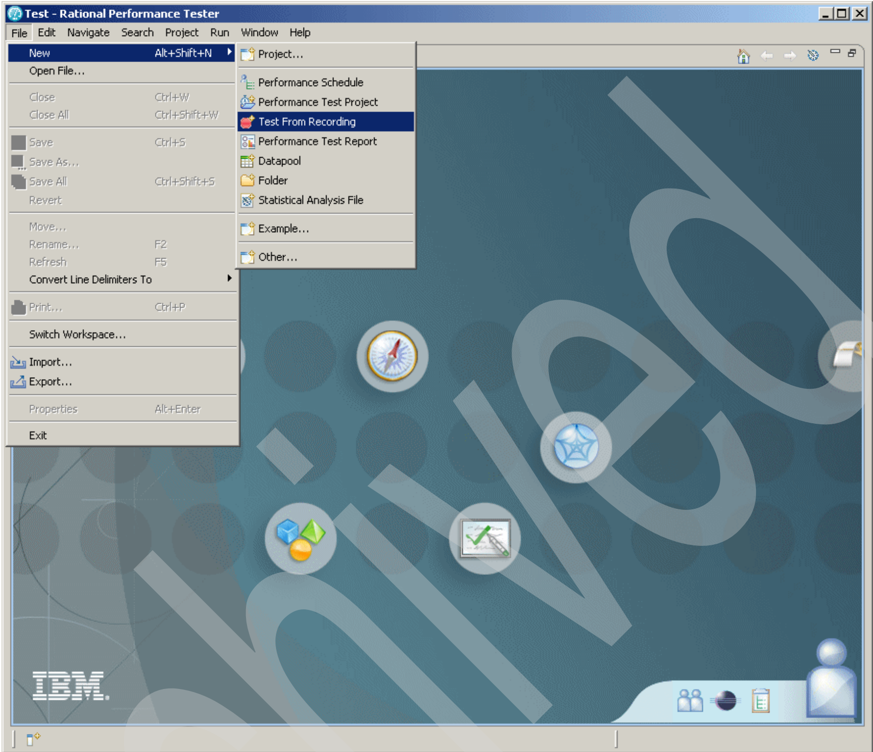


Figure 7-30 Recording a new transaction

b. From the Create New Test from Recording window, select **Create Test From New Recording**, then choose **HTTP Recording**, as shown in Figure 7-31. Click **Next**.

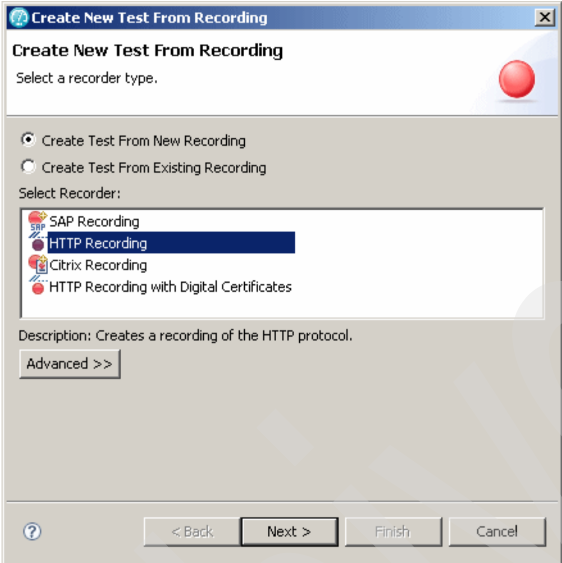


Figure 7-31 Create New Test From Recording

c. In the Create Project window, fill in a project name, a location, and a recording file name, as shown in Figure 7-32. Click **Finish**. The recording file name is used to identify the robotic script and robotic transaction. This name should match your naming convention standard, especially if you want to use an automation application based on the robotic events.

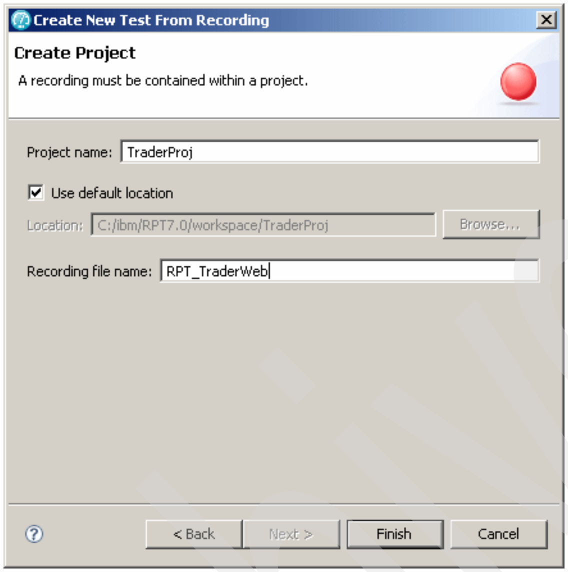


Figure 7-32 Create project

d. The browser opens the readme.html file, as shown in Figure 7-33. Read and take the actions described in the section “BEFORE YOU BEGIN.” Also read the privacy warning.

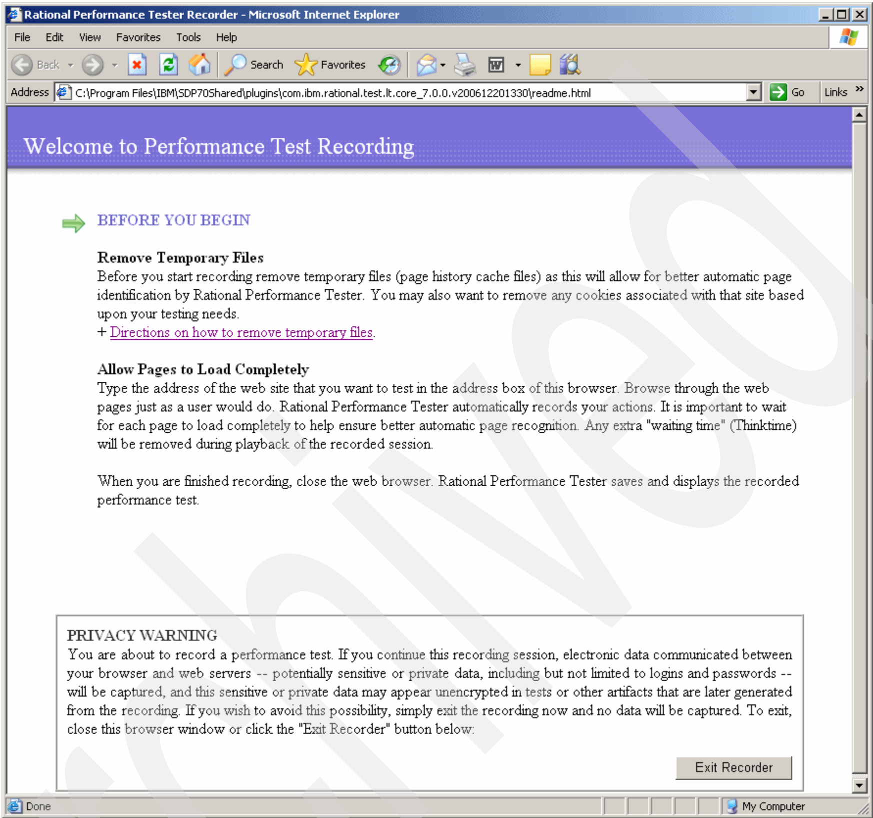


Figure 7-33 Recording welcome page

3. Start recording the transaction that you want to monitor. Figure 7-34 shows some of the steps that we took when recording the RPT_TraderWeb transaction. When you finish your transaction recording, close the browser.

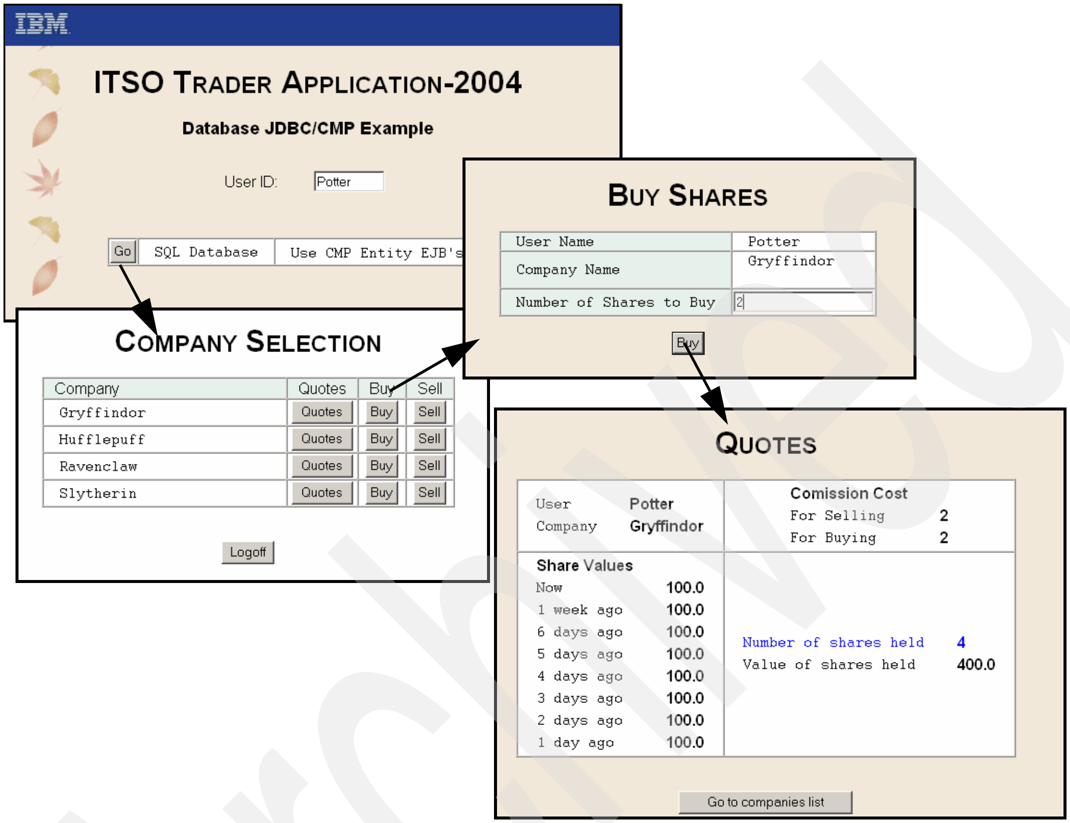


Figure 7-34 Recording a transaction

4. Once the Web browser window is closed, the test generation is complete. We closed the Welcome tab to give more room to view the recorded test. We also selected the Browser tab in the Protocol Data so that we can see the Web page being tested. See the test result in Figure 7-35.

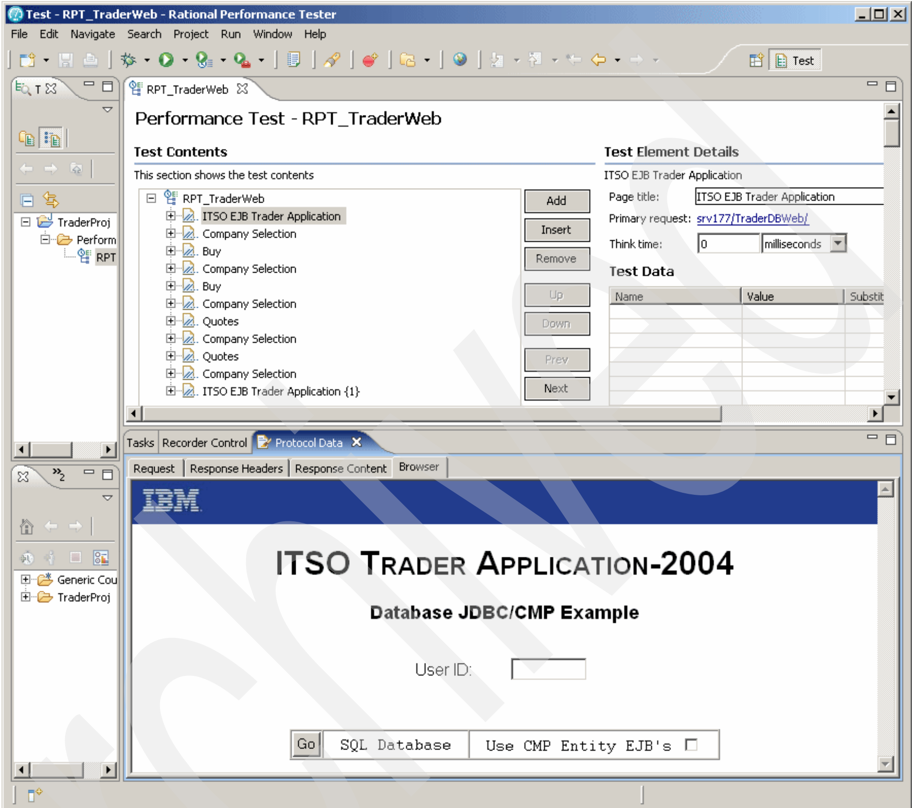


Figure 7-35 RPT workbench - test showing the Web page

5. The test result in Figure 7-35 on page 376 shows generic page titles, such as company selection, buy, and quotes. These page titles becomes subtransactions in the robotic monitor execution. We recommend changing the page titles to allow ease of identification for sub transactions. We customized the page titles, as shown in Figure 7-36.

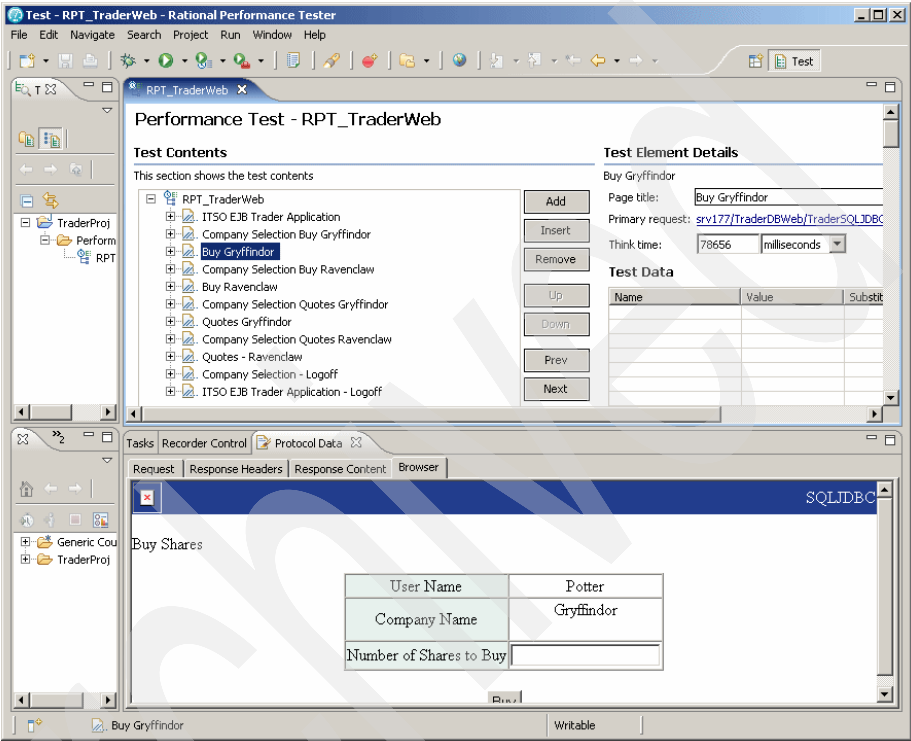


Figure 7-36 Customized page title in workbench

6. Optionally, we also added response code verification points, as shown in Figure 7-37.

> **Note:** *Verification points* are ways to detect problems in an application. Web verification points you can use are:
>
> - *Content* searches for a specific string on a page.
> - *Page title* compares the page title against the recorded title.
> - *Response code* compares the HTTP code returned against the recorded code.
> - *Response size* compares the page size returned against the recorded size.
> - *Custom* is a Java API that is available only when custom code is added to a test.
>
> For more information, access the help provided in the Rational Performance Tester Workbench.

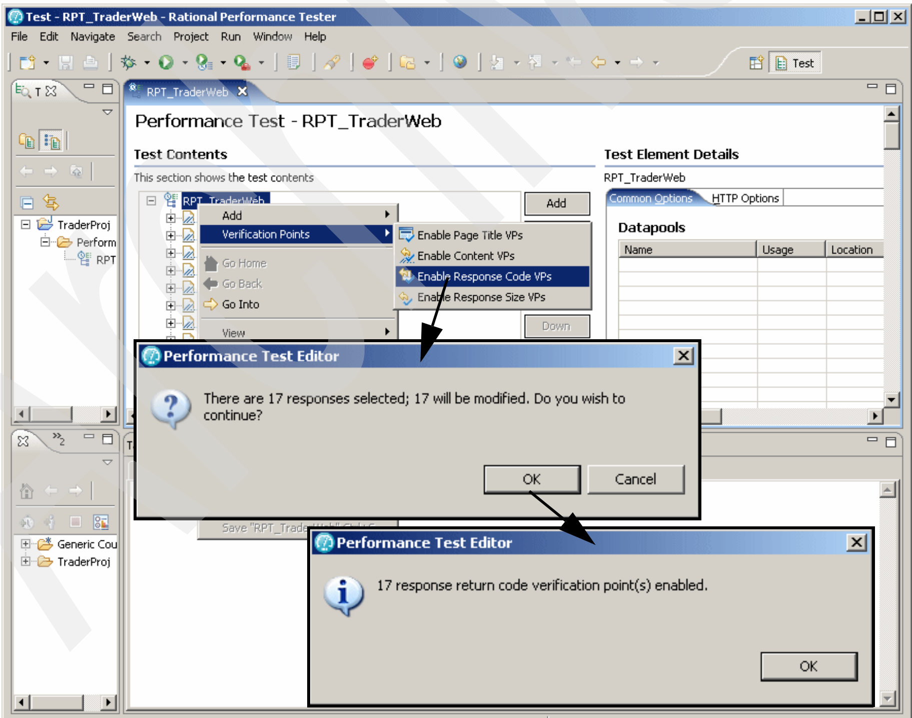


Figure 7-37 RPT workbench - enabling response code verification points

The response code verification default is inserted to all sub-transactions of the pages. This is shown in Figure 7-38.

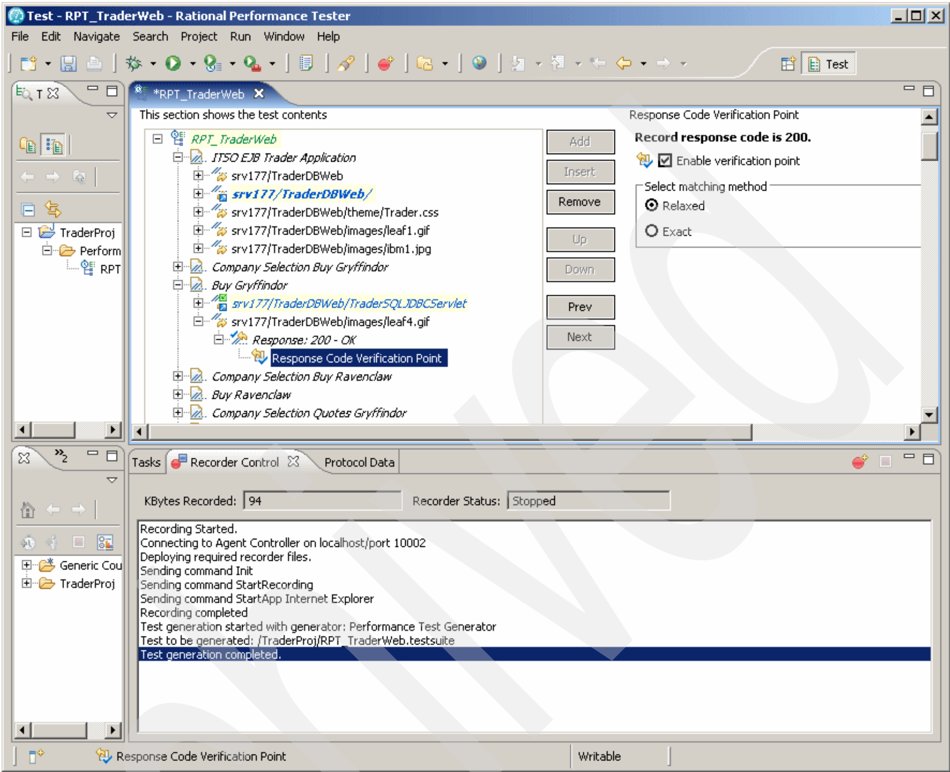


Figure 7-38 Response code verification point in the workbench

7. Save the updates by selecting **File** → **Save**. Make sure that you test that your script is working properly before uploading it to ITCAM for Response Time. You can test your script using the context menu and selecting **Run As** → **2 Performance Test**, as shown in Figure 7-39.

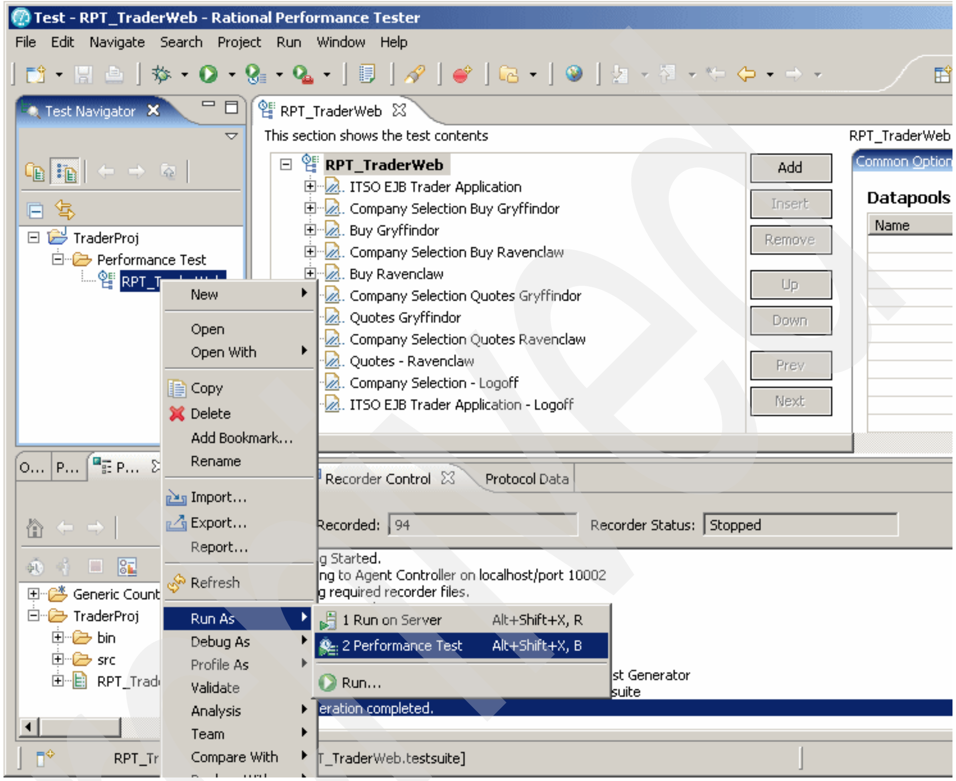


Figure 7-39 RPT workbench - running a test before uploading

8. If all tests are successful, you are ready to upload the script to the End user Dashboard Agent. The ITCAM for Response Time uploader is a plug-in in the Eclipse workbench that runs Rational Performance Tester Workbench.

   a. Before you upload the script, we recommend that you stop the ITCAM for Robotic Response Time, as shown in Figure 7-40. You can also use the **`./itmcmd agent stop t6`** command or using Manage Tivoli Enterprise Monitoring Services.

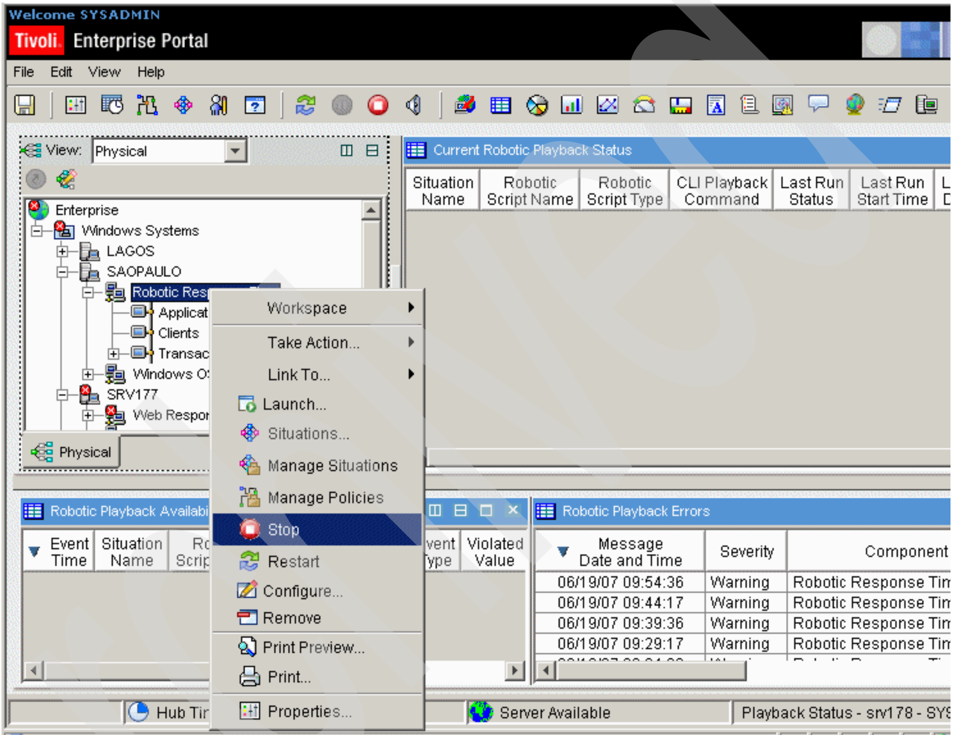


Figure 7-40 Stopping Robotic Response Time Agent

b. On the Rational Performance Tester Workbench select **File** → **Export**. Click **Other** → **ITCAM for Response Time** and click **Next**, as shown in Figure 7-41.

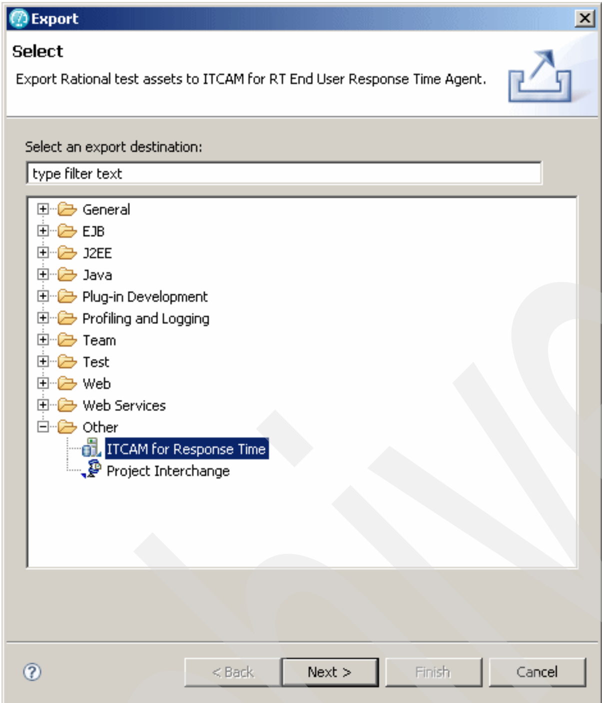


Figure 7-41 RPT workbench - export to ITCAM for Response Time

c. Figure 7-42 shows the Connect to ITCAM for RT End User Response Time Agent window. Fill in the host name and port for the End User Response Time Dashboard Agent. Click **Next**.

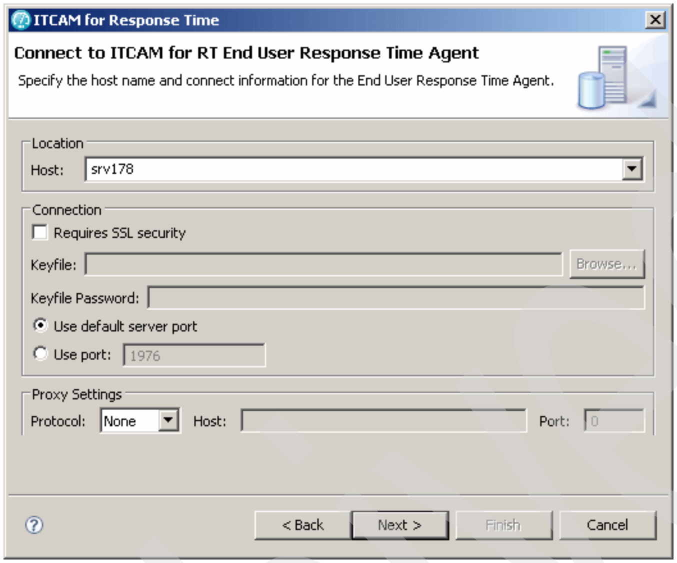


Figure 7-42 Connecting to End User Response Time Agent

d. Figure 7-43 shows the Select Projects window. Select the project that you want to export and click **Next**.

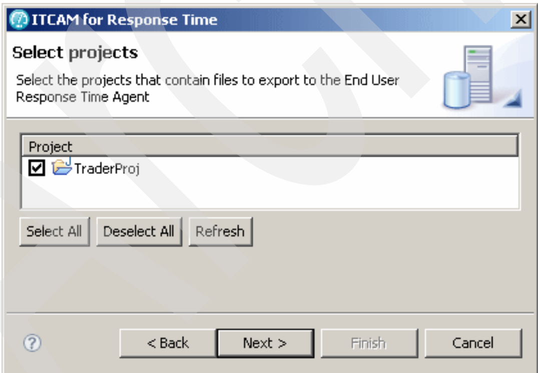


Figure 7-43 Select project to export

e. Select the files to export, as shown in Figure 7-44. Click **Finish**.

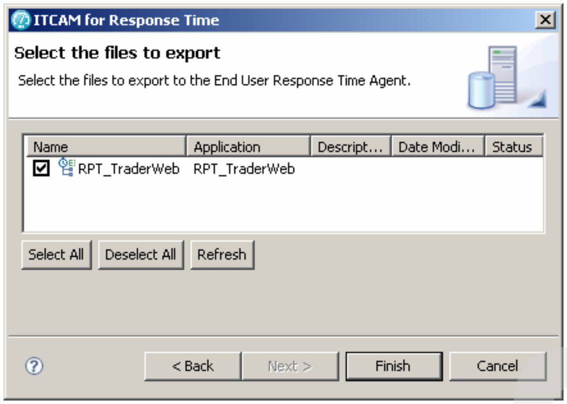


Figure 7-44 Select the files to export

f. Wait until you get the message that upload is successful, as shown in Figure 7-45.

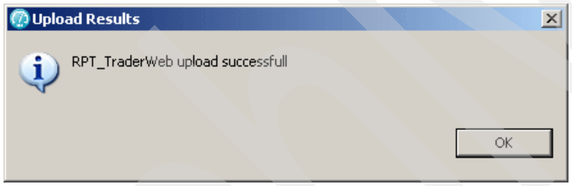


Figure 7-45 Upload successful

g. Start the ITCAM for Robotic Response Time, as shown in Figure 7-46. You can also use the `./itmcmd agent start t6` command or you can use the Manage Tivoli Enterprise Monitoring Services.

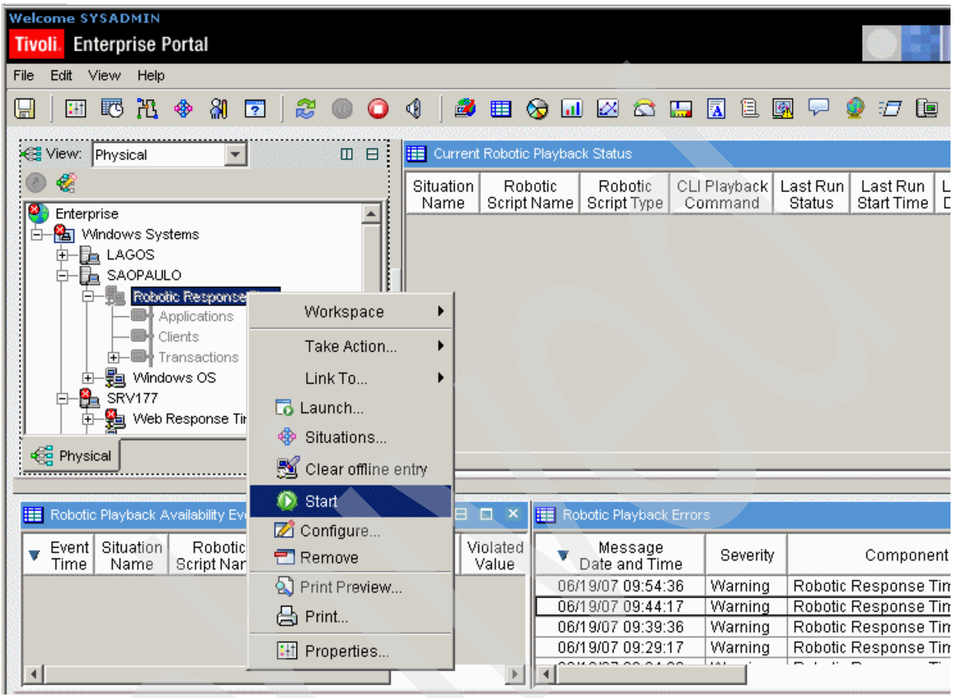


Figure 7-46 Starting the ITCAM for Robotic Response Time Agent

9. You have finished the record and upload/export steps.

7.4.3 Monitoring Robotic Response Time

In this section, we discuss the following:

- Selective execution of robotic script
- Verifying whether the playback is working
- Working with a robotic situation event
- Customizing the playback schedule

Selective execution of robotic script

Once you have the robotic script uploaded, robotic monitoring is performed using situations. The Robotic Response Time Agent downloads all robotic scripts regularly (by default every 15 minutes) from the depot. It executes any script whose name matches a Robotic Agent situation. Figure 7-47 shows the default Robotic Response Time Playback situation that matches all scripts. This situation executes all stored robotic scripts based on the sampling interval defined. See Figure 7-47.

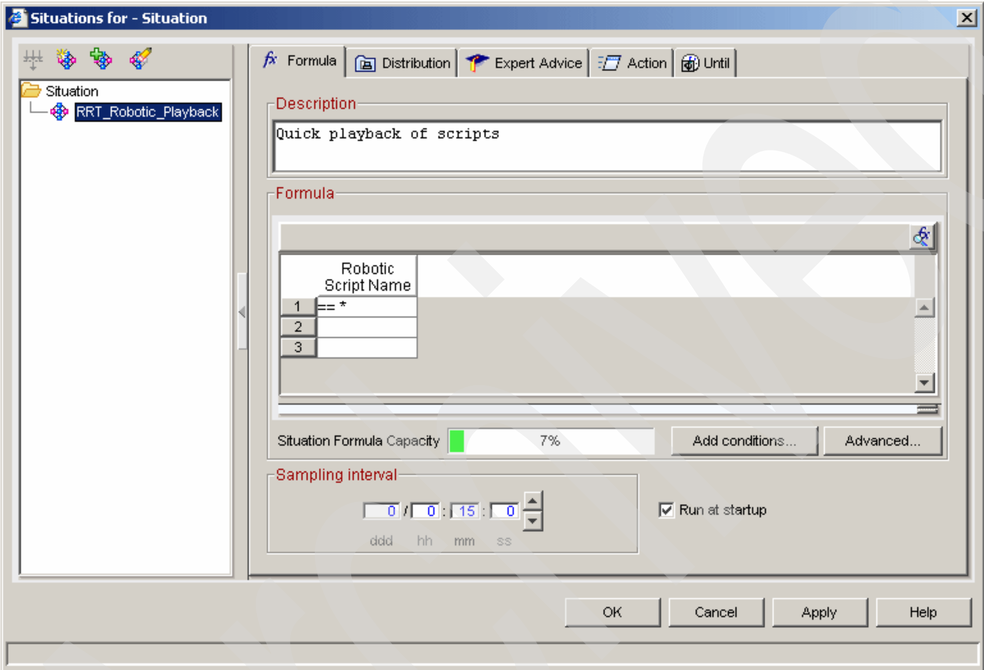


Figure 7-47 RRT_Robotic_Playback situation

You can change this behavior by modifying the condition of an existing situation and changing the distribution scope of the situation.

Verifying whether the playback is working

Now that we have uploaded the script to the ITCAM for End User Response Time Dashboard, we check whether it is being played back.

The next windows show you how to verify this by navigating some of the workspaces.

Note: For a detailed description of the workspaces, situations, and attributes, read the ITCAM for Robotic Response Time and ITCAM End User Response Time Dashboard Users Guide, SCxx-xxxx.

Do the following:

1. To identify what scripts are stored on the End User Response Time Dashboard, you can access the robotic scripts workspace, as shown in Figure 7-48.

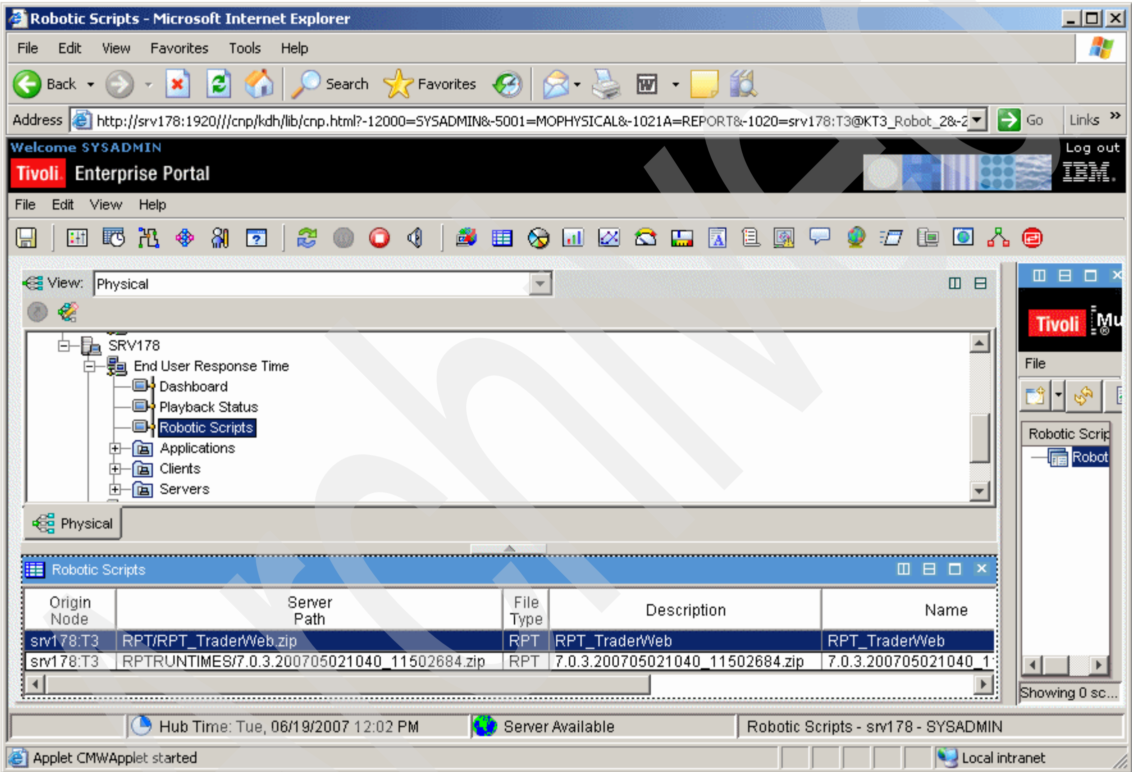


Figure 7-48 End user response time - robotic scripts workspace

2. From the End User Response Time workspace in Figure 7-49, you see the status of all the ITCAM for Response Time agents, and from that workspace you can link to the agent-specific workspaces.

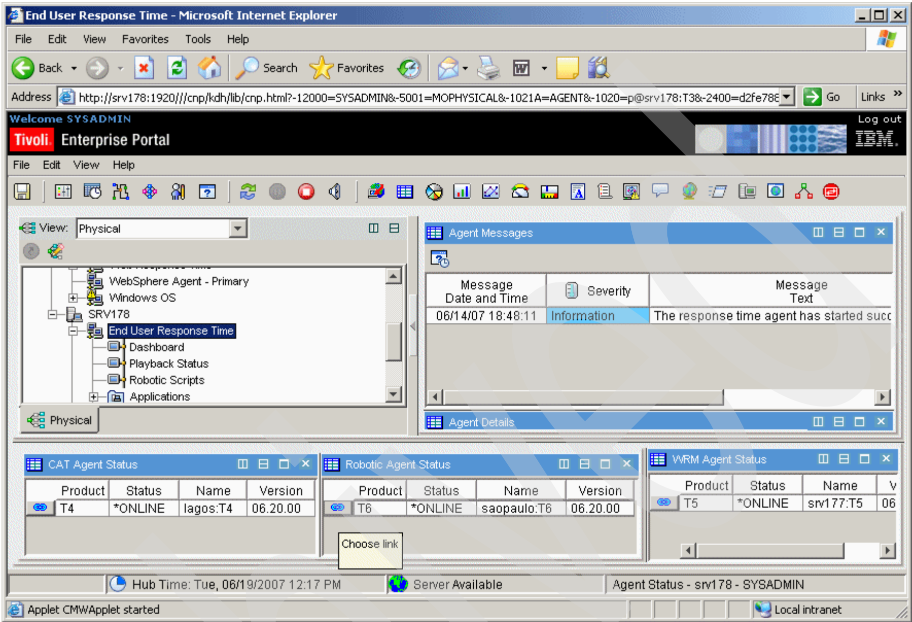


Figure 7-49 End user response time workspace

3. The Current Robotic Playback Status view (Figure 7-50) provided in the Robotic Response Time Playback Status workspace provides information about each script running on the agent. From here you can monitor the current status of any robotic scripts running on the server.

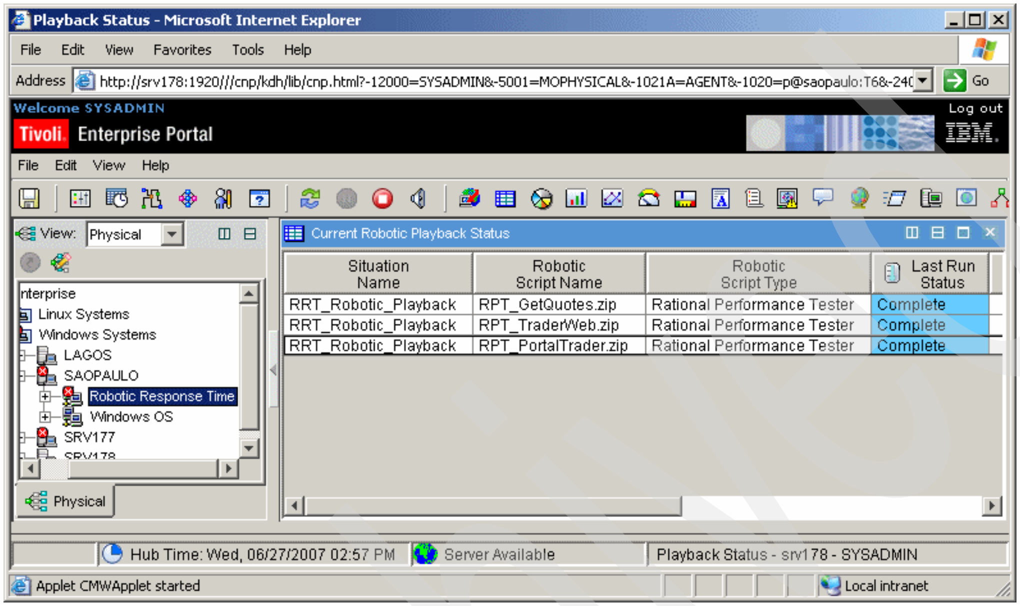


Figure 7-50 Current Robotic Playback Status view

Working with a robotic situation event

In our environment there are various monitoring agents generating situation events. We can demonstrate working with situations using the following flow:

1. To have a better view of what we want to demonstrate, we filter the situation event console events. Figure 7-51 shows the Situation Event Console provided in the Enterprise Status workspace.

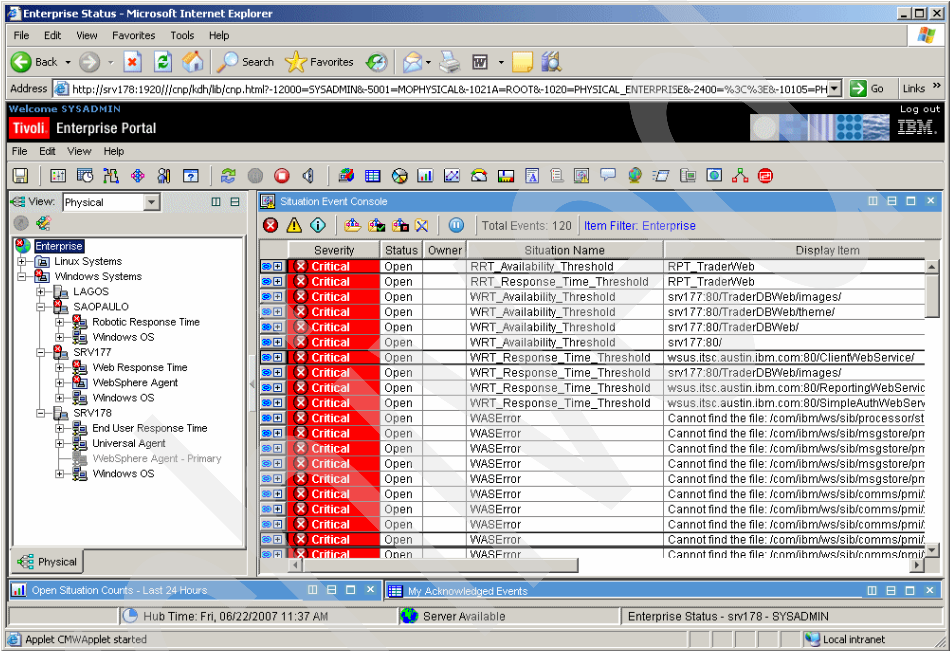


Figure 7-51 Situation event console

2. To filter the situation event console so it would only show the Robotic Response Time agent events, you can drag the Robotic Response Time agent into the situation console, as shown in Figure 7-52.

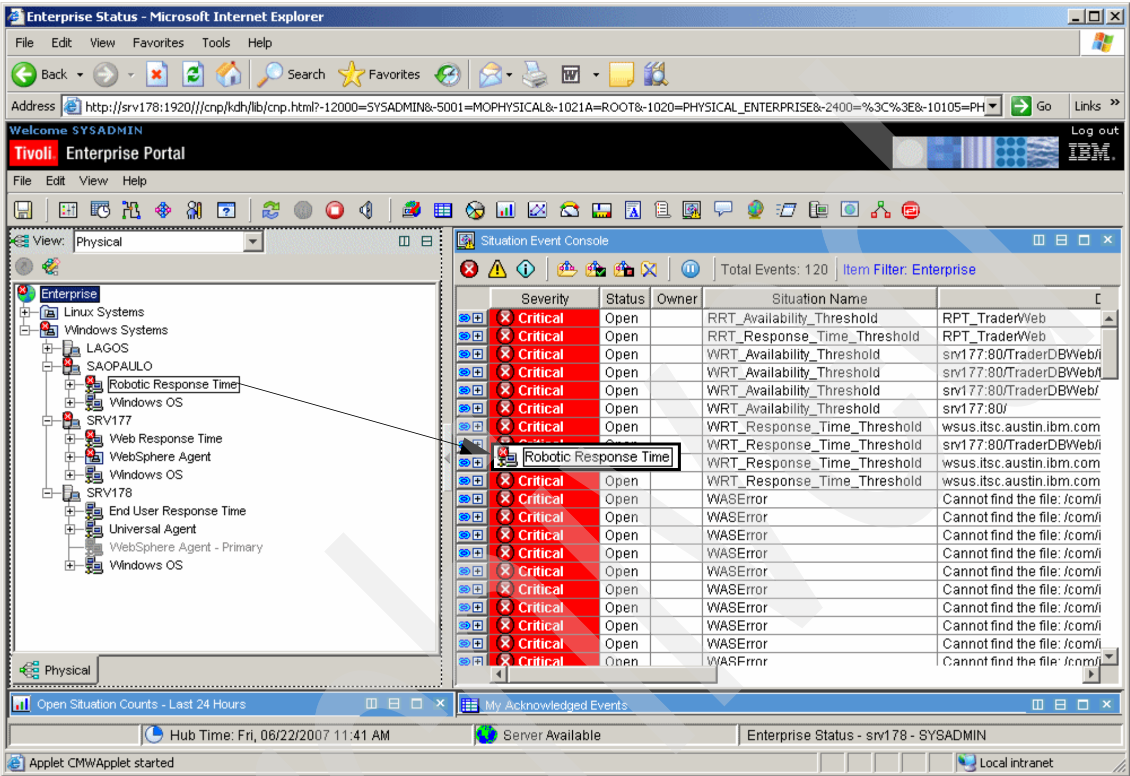


Figure 7-52 Drag Robotic Response Time and drop into situation event console

3. The filter is then active, as shown in Figure 7-53.

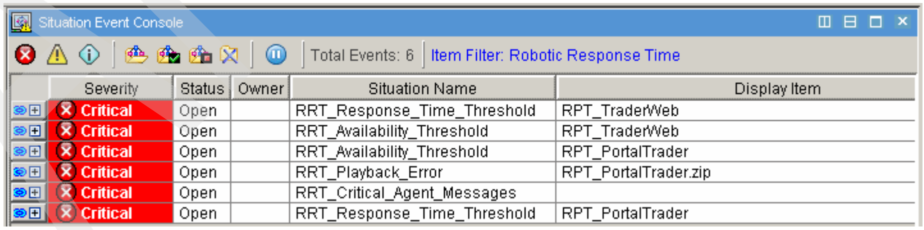


Figure 7-53 Situation event console filter active

4. We can then analyze an event by right-clicking it and selecting **Situation Event Results**, as shown in Figure 7-54.

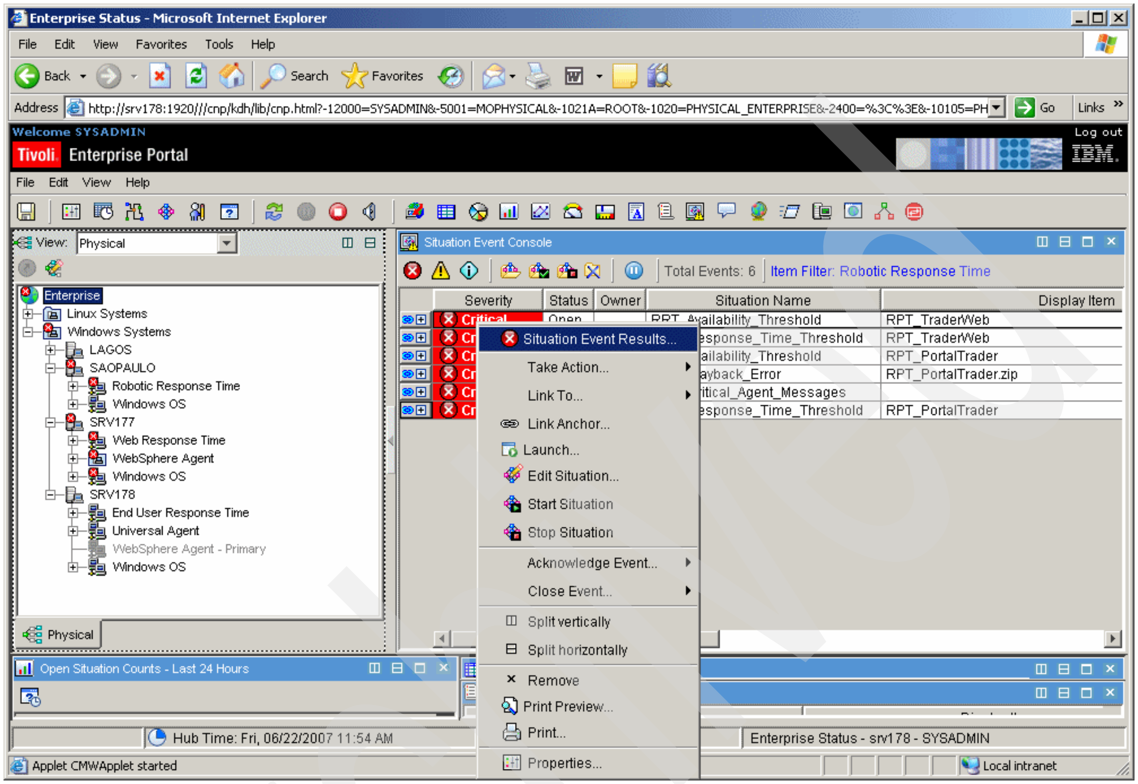


Figure 7-54 Selecting situation event results

5. As we modified the workspace by adding a filter, Tivoli Enterprise Portal confirms (Figure 7-55) whether we want to save the workspace. Click **No** to keep the existing filter.

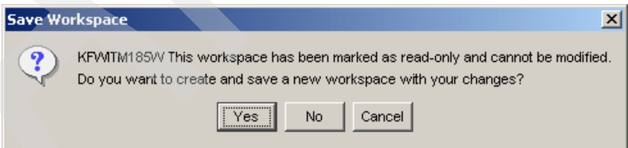


Figure 7-55 Do not save the workspace

6. Figure 7-56 shows details of the critical alert raised by the RRT_Availability_Threshold. This means that the RPT_TraderWeb script did not complete successfully.

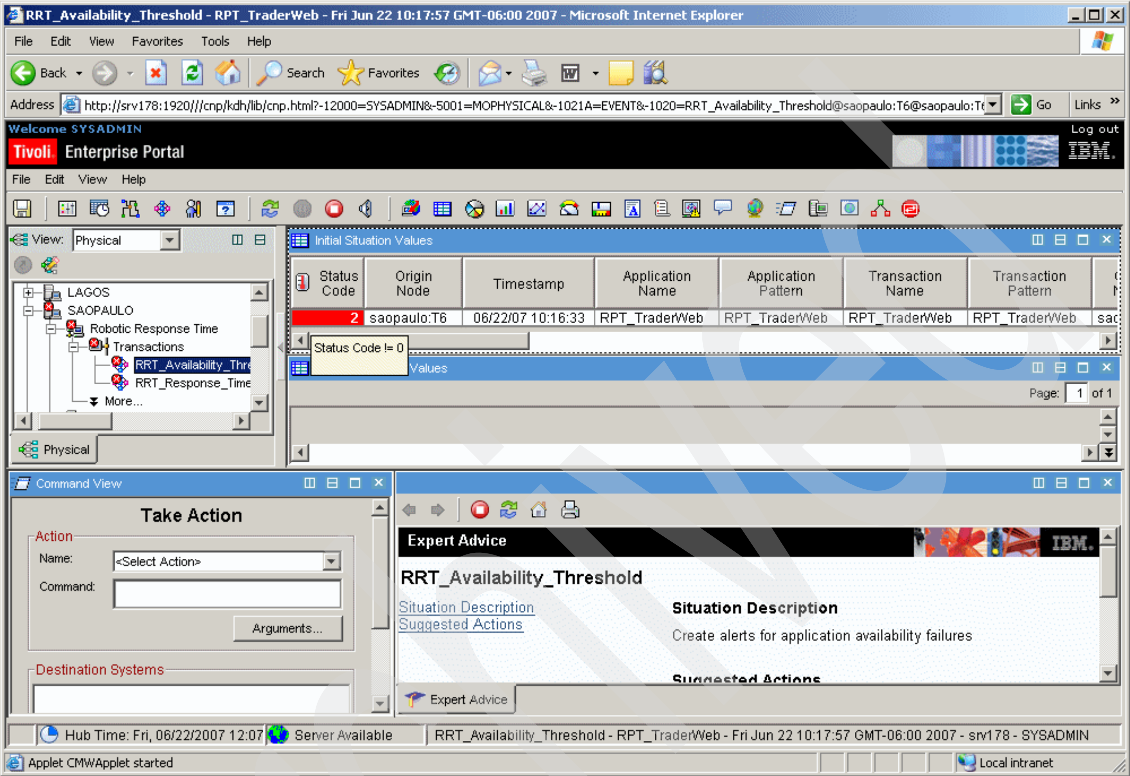


Figure 7-56 RRT_Availability_Threshold event

7. Using the Top 5 Worst Applications Current Status workspace (Figure 7-57), we know that the problem is not occurring anymore.

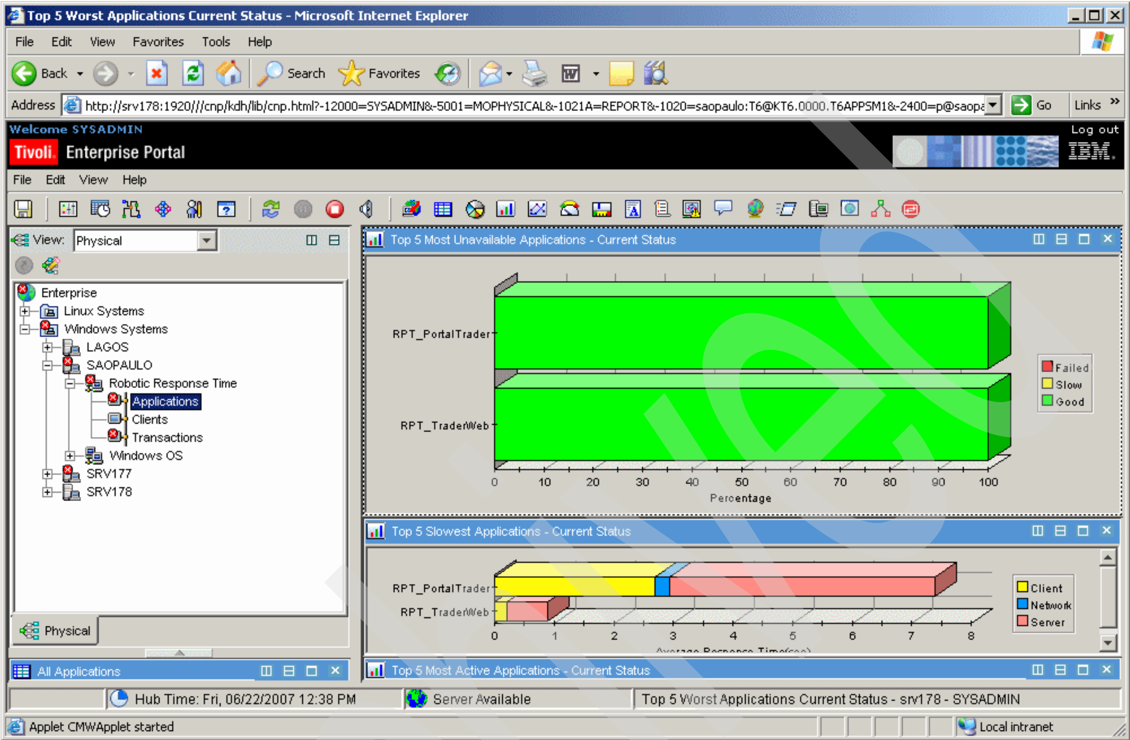


Figure 7-57 Top 5 Most Unavailable Applications - current status

8. You can manually close the event, as shown Figure 7-58.

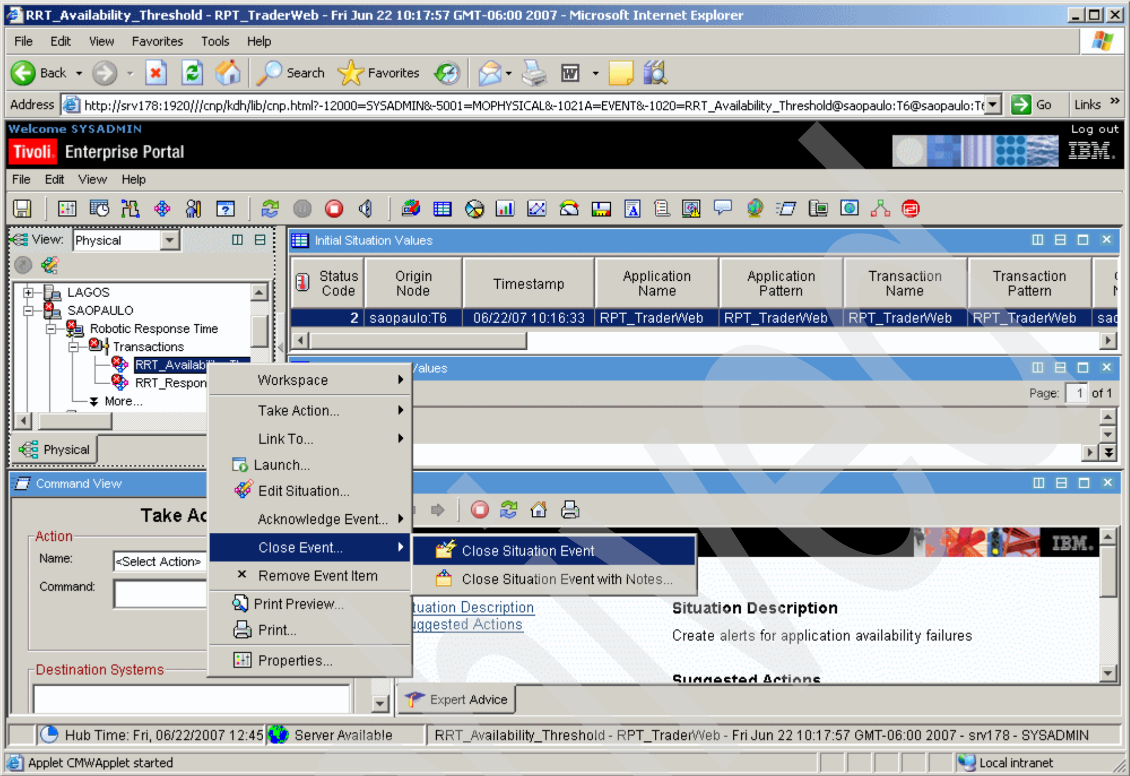


Figure 7-58   Closing the event

Customizing the playback schedule

In this section we explain how to customize a playback schedule for a robotic script called RPT_Get_Quotes.zip. We want to run the script only on work days Monday to Friday within the prime shift from 8 a.m. to 5 p.m. The script will run every 10 minutes with the possibilities o three retries. Each retry will have a 10-second lag time. The script will time out if there is no response within 30 seconds.

> **Note:** For more information about Situations and workflows, refer to the *IBM Tivoli Monitoring User's Guide*, SCxx-xxxx.

To accomplish this scenario:

1. Create a robotic playback configuration situation and do *not* select Run At startup. The situation will be controlled by IBM Tivoli Monitoring workflow.

a. Open the situation editor from **Edit** → **Situation Editor**, as shown in Figure 7-59.

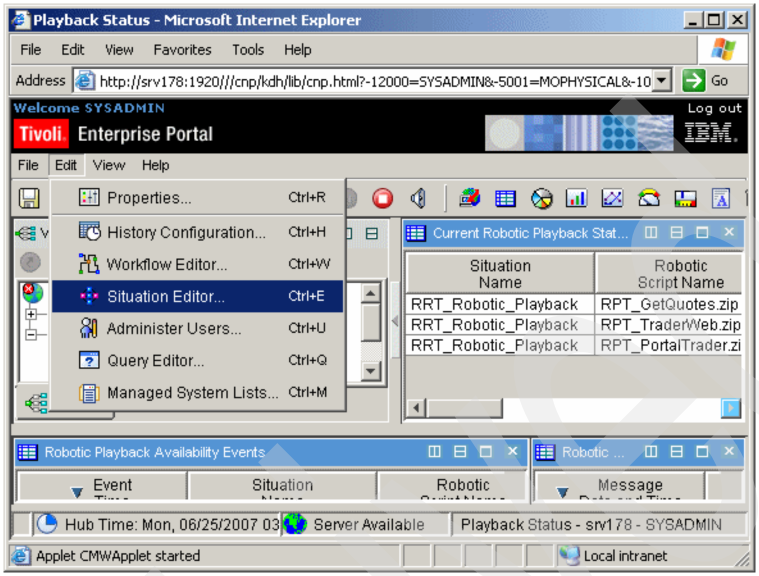


Figure 7-59 Situation Editor

b. Right-click **RRT_Robotic_Playback_Advanced** and select **Create another**, as shown in Figure 7-60.

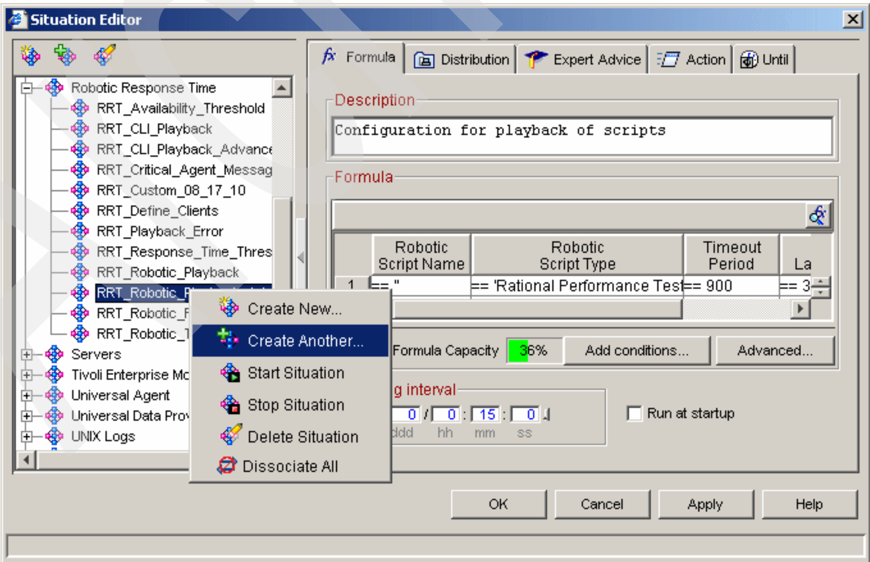


Figure 7-60 Cloning situation RRT_Robotic_Playback_Advanced

c. Fill in name and description in Figure 7-61 and click **OK**.

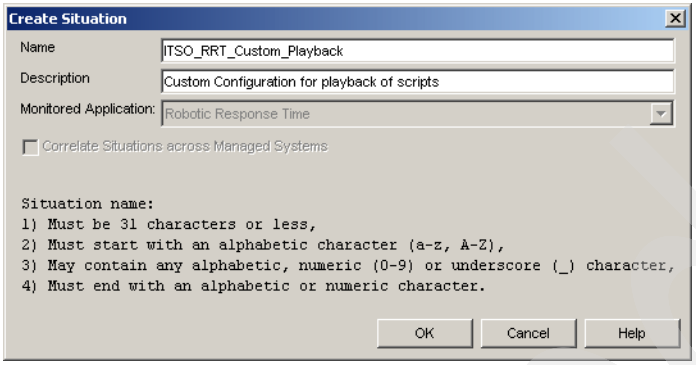


Figure 7-61 Create new situation

d. Customize robotic script name, robotic script type, time out period, retry lag time, number retries, and sampling interval, as shown in Figure 7-62. Do *not* select Run at startup.

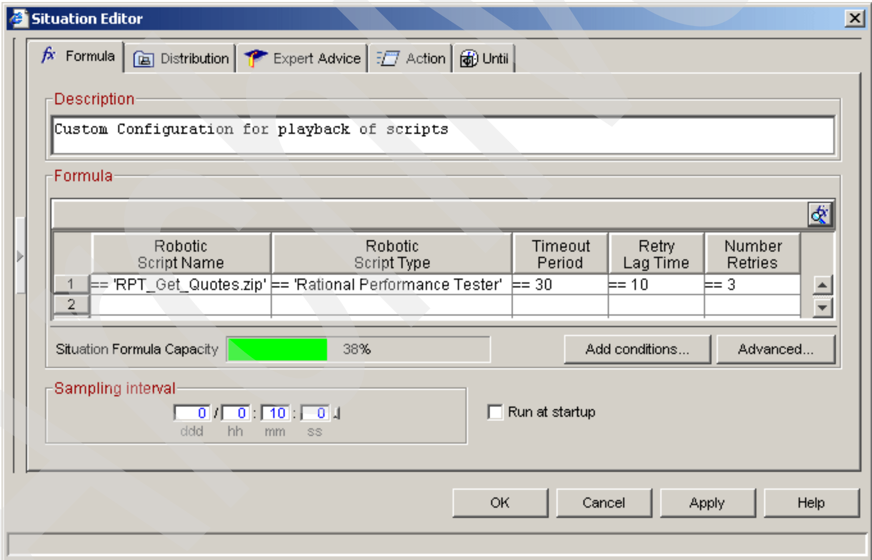


Figure 7-62 Customizing Robotic Agent situation

e. Figure 7-63 shows the Distribution tab. Choose the agents where you want the script to be played back and click **OK**.

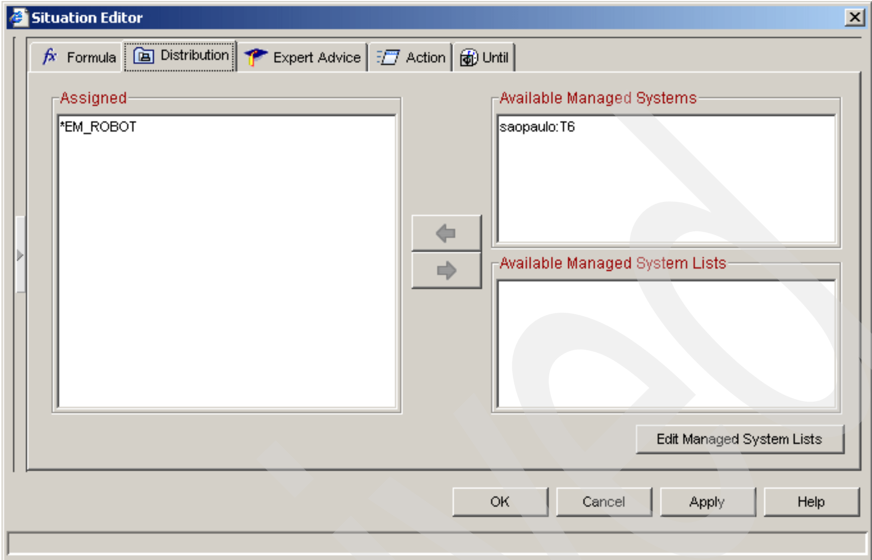


Figure 7-63 Defining situation distribution

2. Create a scheduling situation for starting the robotic monitor:

   a. We use the PrimeShift situation provided by IBM Tivoli Monitoring. For other schedule requirements, you can also clone the PrimeShift situation and define your own scheduling rule. See Figure 7-64.

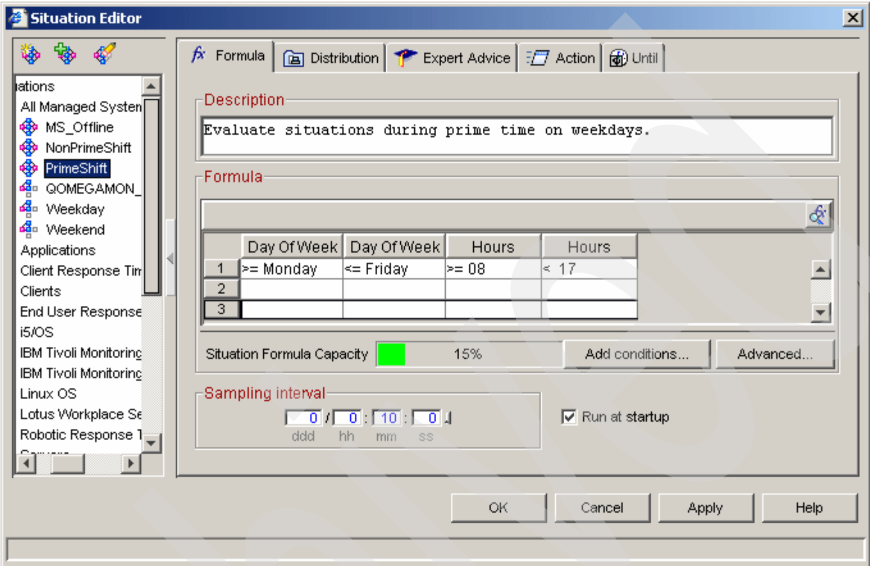


Figure 7-64 PrimeShift situation

b. Make sure to distribute the scheduling situation to the same robotic agents as the robotic configuration situation created in step 1, as shown in Figure 7-65.

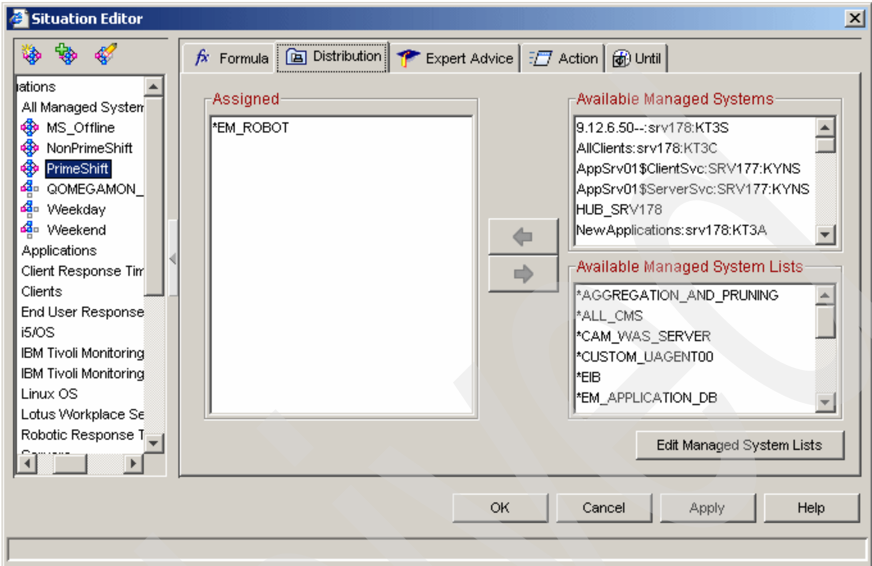


Figure 7-65 PrimeShift situation - Distribution tab

3. Create a scheduling situation to stop the robotic playback:

   a. We use the NonPrimeShift situation provided by IBM Tivoli Monitoring. Do not use Run at startup. Figure 7-66 shows the NonPrimeShift situation.

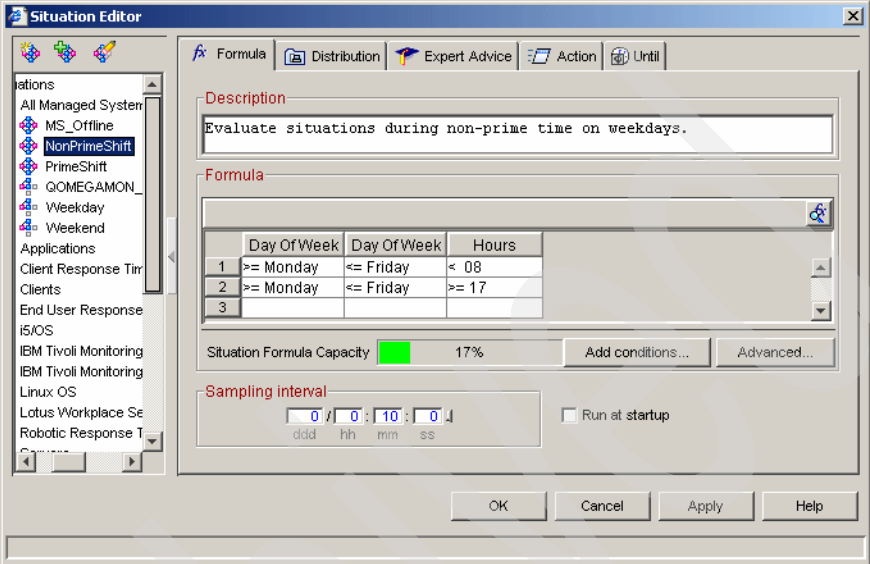


Figure 7-66 NonPrimeShift situation

   b. Make sure to distribute the scheduling situation to the same robotic agents as the robotic configuration situation created in step 1, as shown in Figure 7-67.

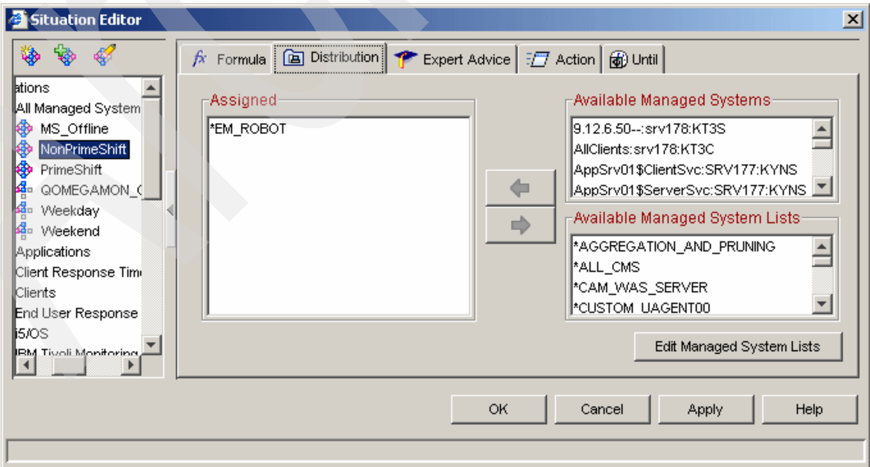


Figure 7-67 NonPrimeShift situation - Distribution tab

4. Create a workflow to start and stop the robotic playback situation. Distribute the workflow to the same robotic agents that you distributed in the previous steps. Figure 7-68 shows the workflow.

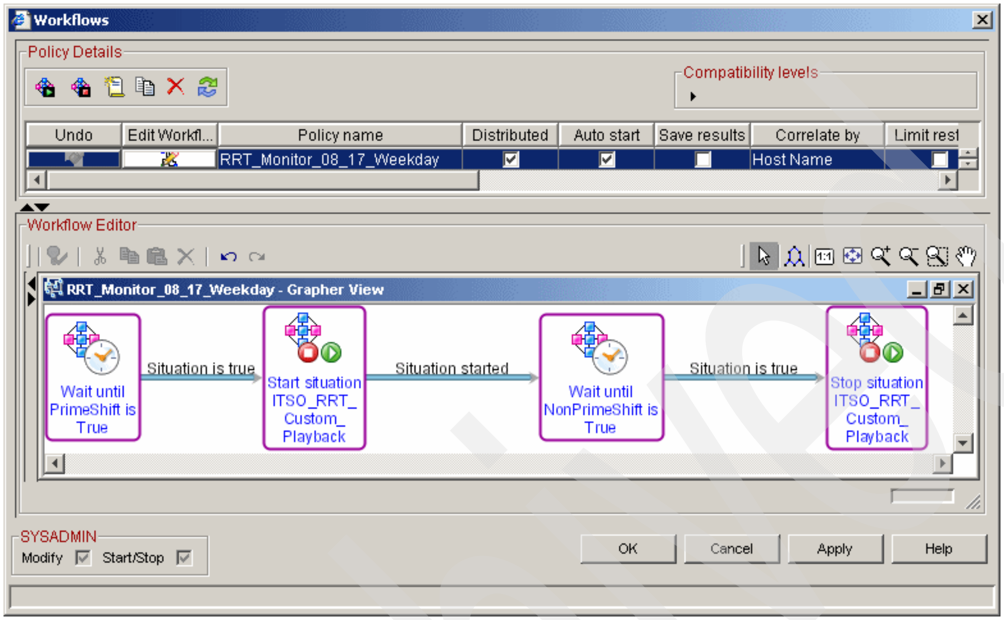


Figure 7-68 Workflow for start and stop the customized schedule

7.4.4 Monitoring Web Response Time agent

The Web Response Time agent is used to collect HTTP response time for the trader application running on srv177. WebSphere provides HTTP access at ports 9081 and 9082. By default, Web Response Monitor collects all HTTP traffic and measures its response time. We define situations to indicate the data to be collected by the Tivoli Enterprise Monitoring Server.

First let us evaluate the workspaces of the Web Response Time agent. Figure 7-69 shows the agent summary workspace that lists important information about the agent.

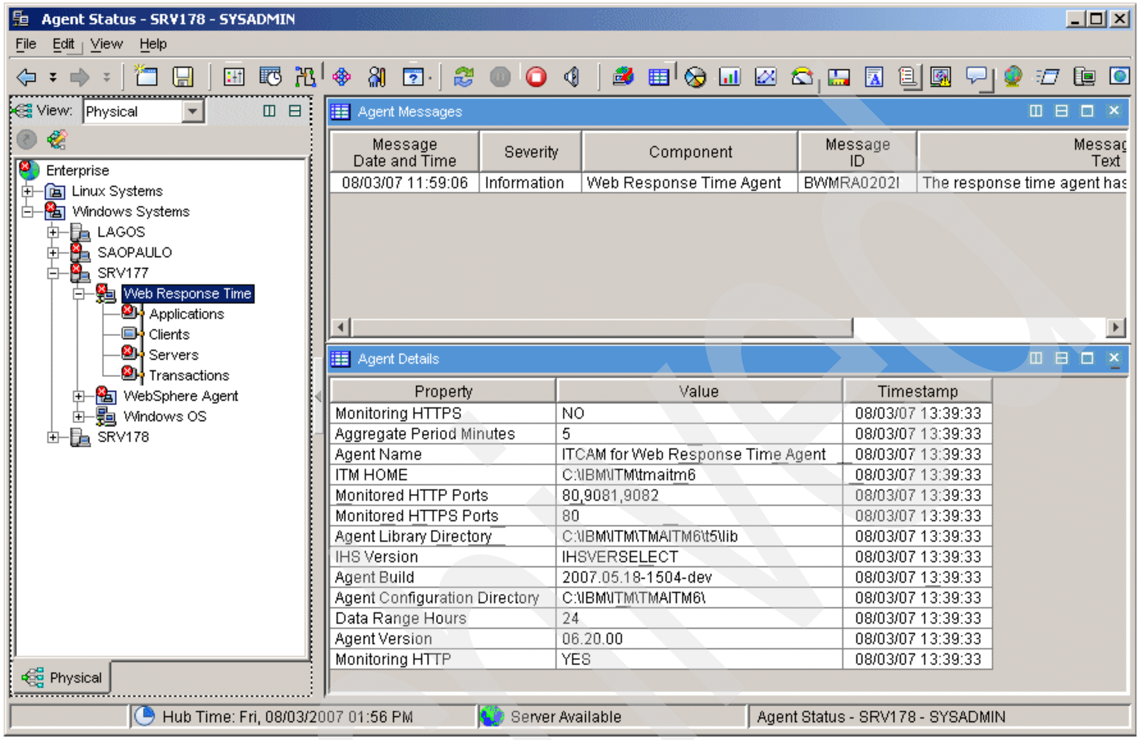
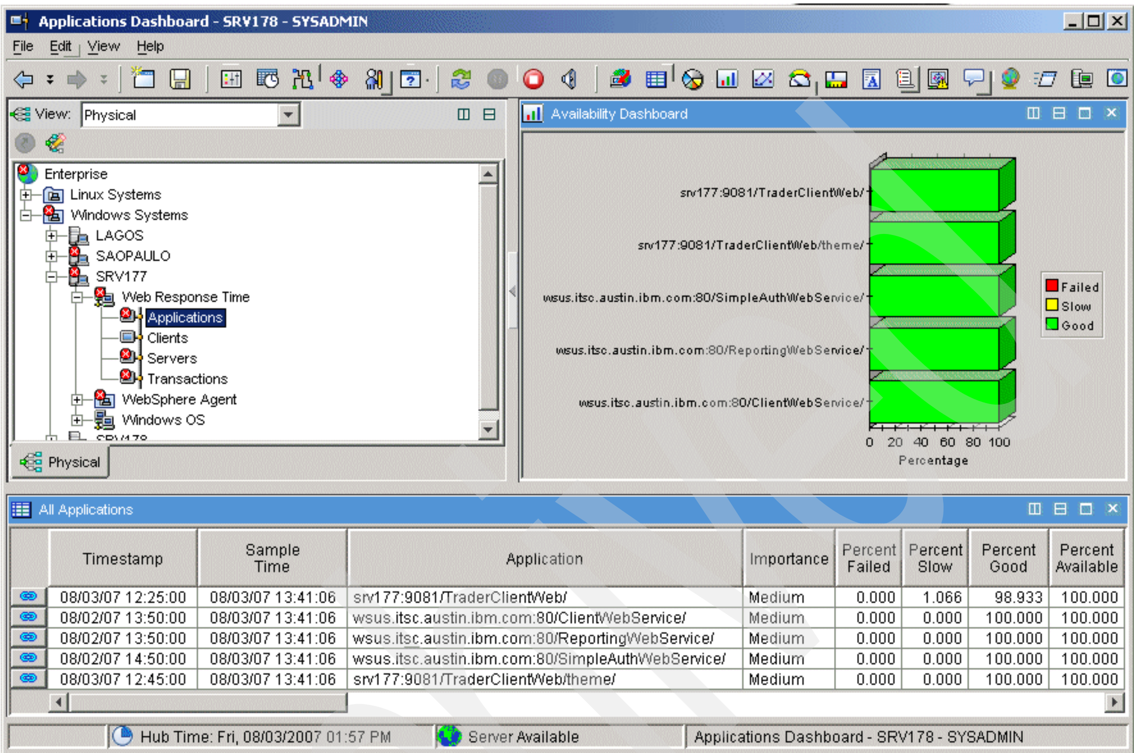


Figure 7-69 Agent summary workspace

The workspaces shows the application response time breakdown that is collected from this agent by application, transaction, client, and server.

The application workspace differentiates response time by the URL path. Figure 7-70 shows the application grouping.



	Timestamp	Sample Time	Application	Importance	Percent Failed	Percent Slow	Percent Good	Percent Available
	08/03/07 12:25:00	08/03/07 13:41:06	srv177:9081/TraderClientWeb/	Medium	0.000	1.066	98.933	100.000
	08/02/07 13:50:00	08/03/07 13:41:06	wsus.itsc.austin.ibm.com:80/ClientWebService/	Medium	0.000	0.000	100.000	100.000
	08/02/07 13:50:00	08/03/07 13:41:06	wsus.itsc.austin.ibm.com:80/ReportingWebService/	Medium	0.000	0.000	100.000	100.000
	08/02/07 14:50:00	08/03/07 13:41:06	wsus.itsc.austin.ibm.com:80/SimpleAuthWebService/	Medium	0.000	0.000	100.000	100.000
	08/03/07 12:45:00	08/03/07 13:41:06	srv177:9081/TraderClientWeb/theme/	Medium	0.000	0.000	100.000	100.000

Figure 7-70 Application workspace

The server workspace shows the target server of the HTTP transactions. Figure 7-71 shows the sample HTTP transactions.

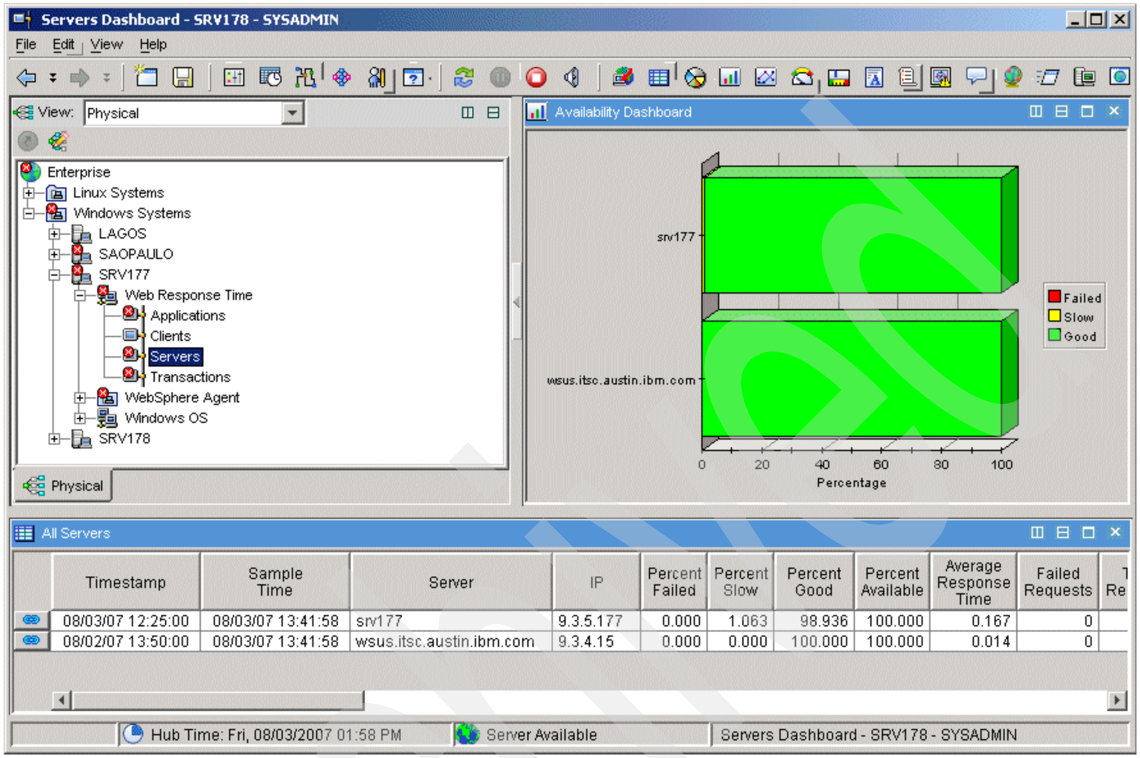


Figure 7-71 Server workspace

The transaction workspace gets you into the individual observed transactions. Figure 7-72 shows the sample workspace.

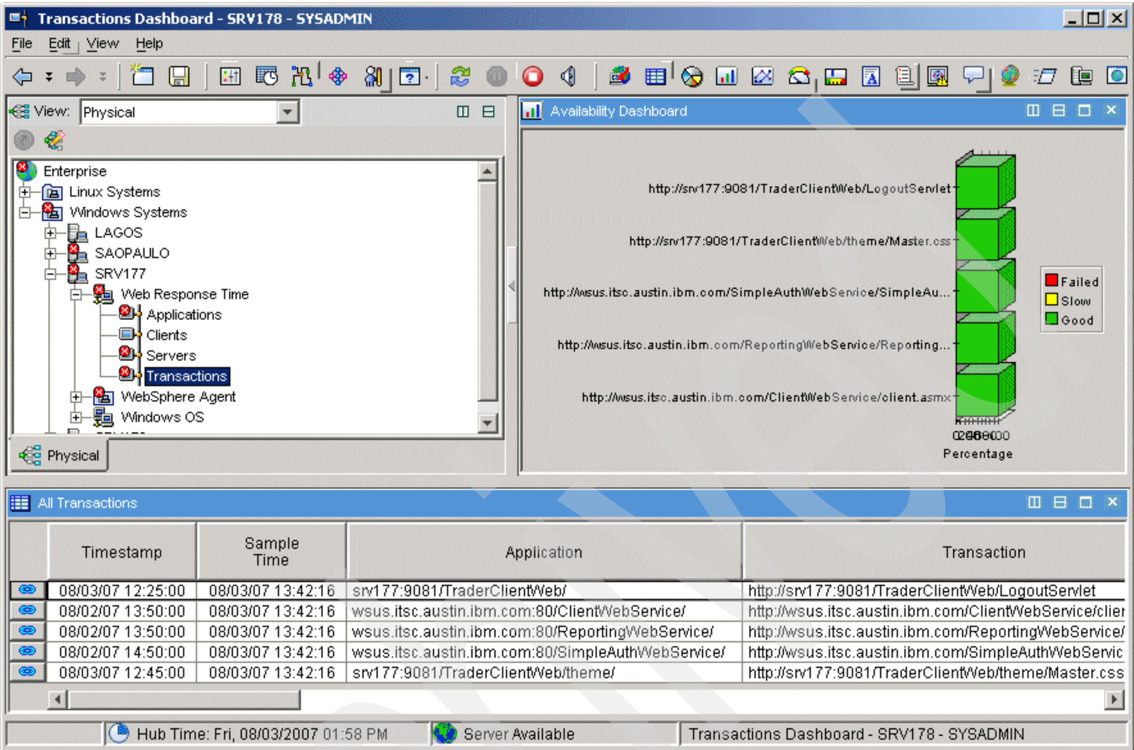


Figure 7-72 Transaction workspace

The rule to determine transaction and client grouping is defined as situations. The list of active situations for the Web Response Time agent can be retrieved by right-clicking the Web Response Time agent in the navigation tree and selecting **Manage Situations**. The situation list is shown in Figure 7-73.

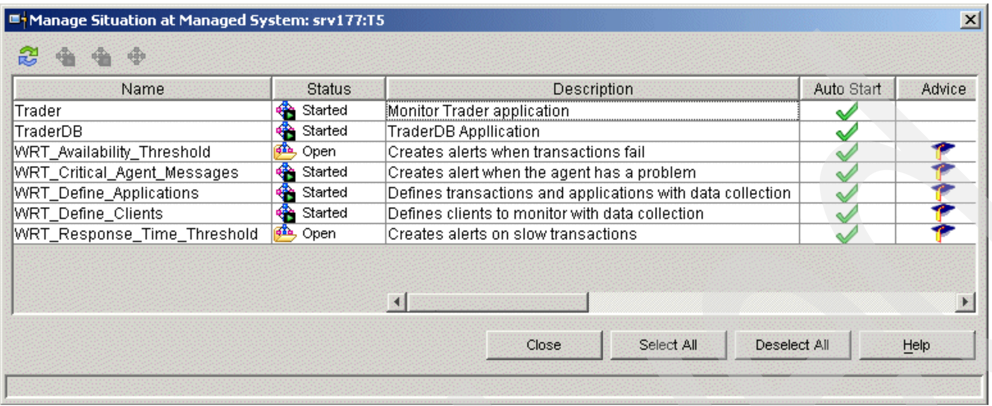


Figure 7-73 Situation list

As an example, let us right-click the WRT_Defining_Clients situation and select **Edit Situation**. Figure 7-74 shows that all clients are grouped in the all clients group.

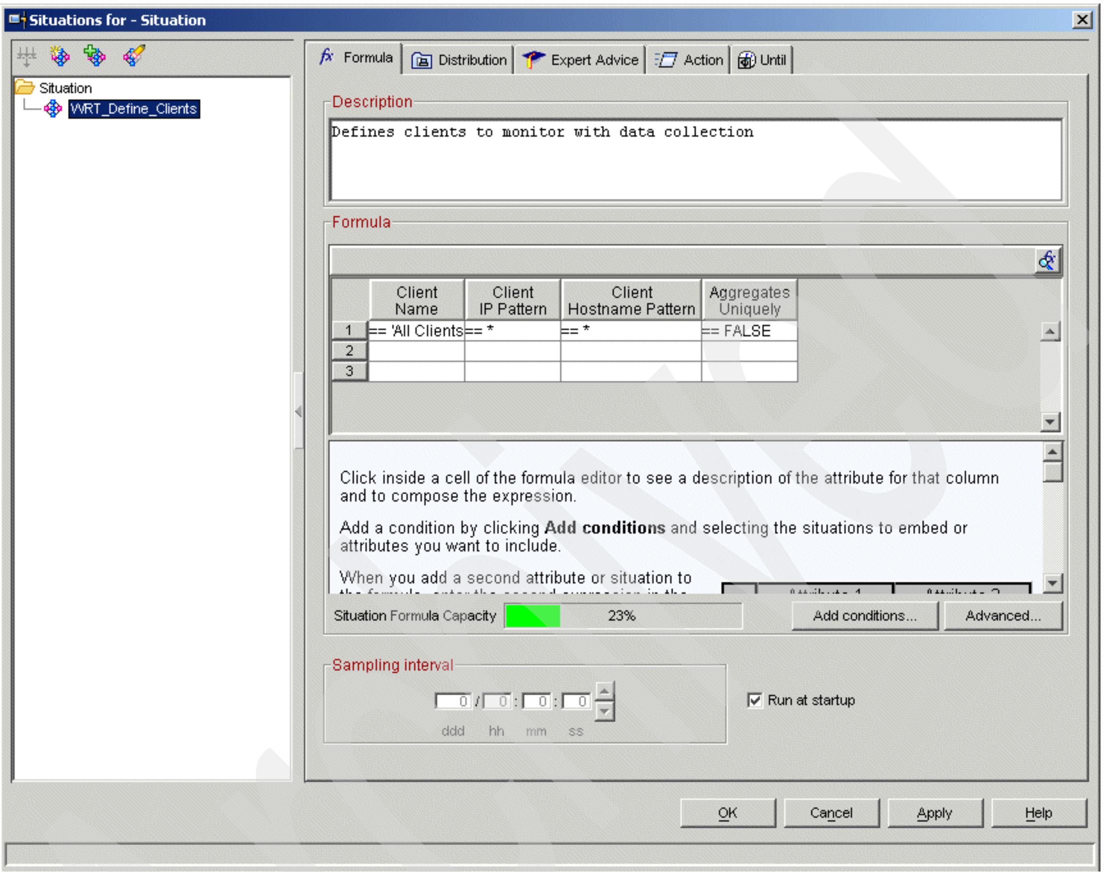


Figure 7-74 Situation for defining clients

Note: The Web Response Time situation does not support multiple conditions.

7.4.5 Using the Client Response Time Agent

For the Client Response Time Agent, we monitor the response time for accessing the trader application on the mainframe using IBM Personal Communications 3270 application.

To monitor the Client Response Time we need to create a situation from the Tivoli Enterprise Portal. The response time would be monitored according to the situation and the filter rule set up in the situation. Creating a situation for the Client Response Time is the same as creating a situation for the Web Response Time.

Once the situation is created and started, the Client Response Time agent on lagos starts monitoring the application and filtering events based on the rule defined in the situation, and starts sending data to the Tivoli Enterprise Portal. The trader CICS 3270 application windows are shown in Figure 7-75.

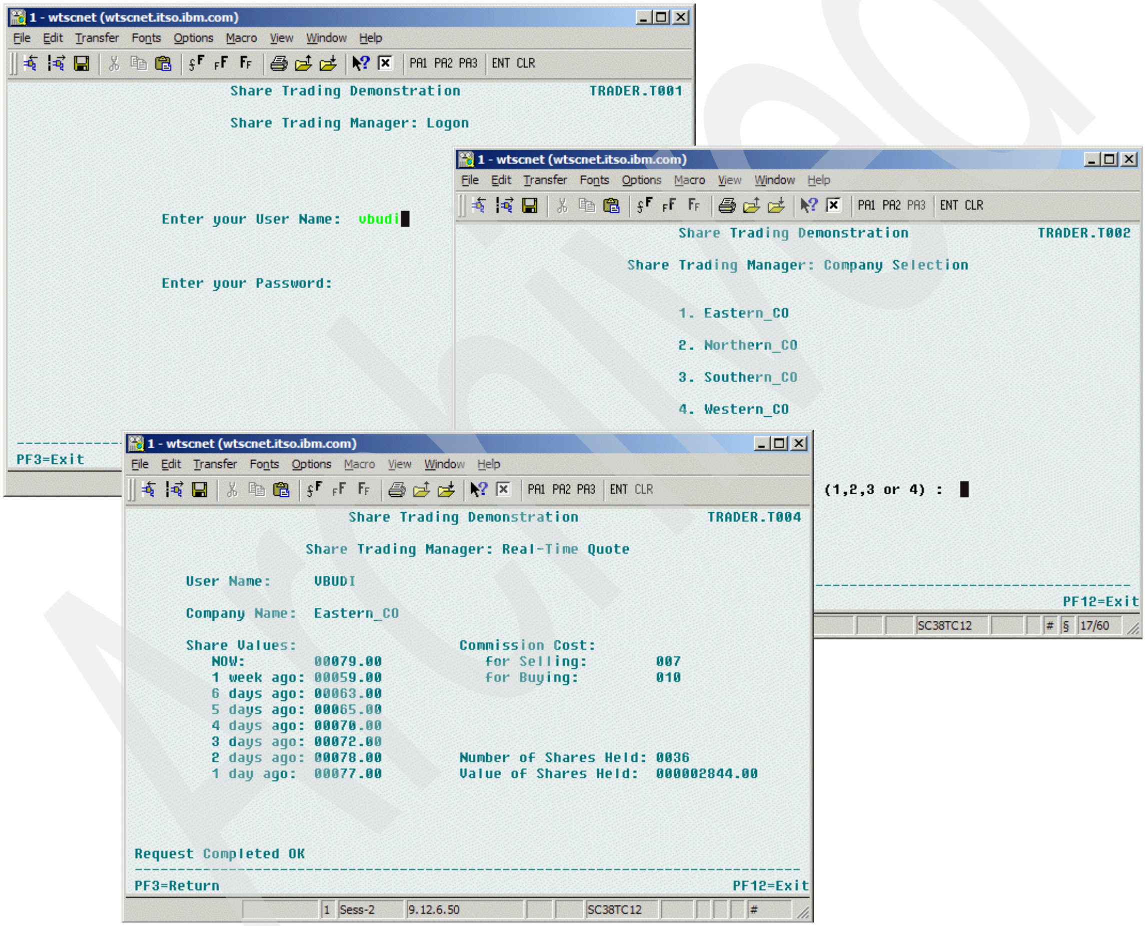


Figure 7-75 Accessing trader using IBM Personal Communications

Client Response Time data can now be seen on the Tivoli Enterprise Portal under Transactions for Client Response Time, as shown in Figure 7-76.

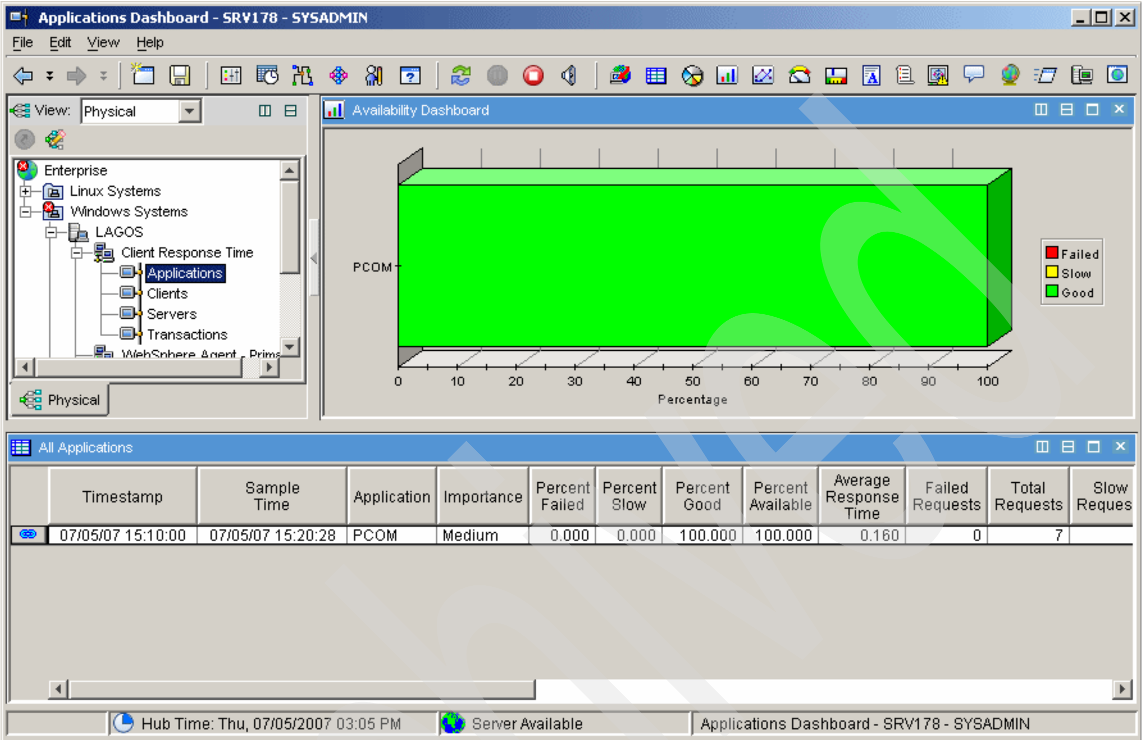


Figure 7-76 Transaction Data for Client Response Time

Clicking the chain icon ( ) takes you to the transaction detail shown in Figure 7-77.

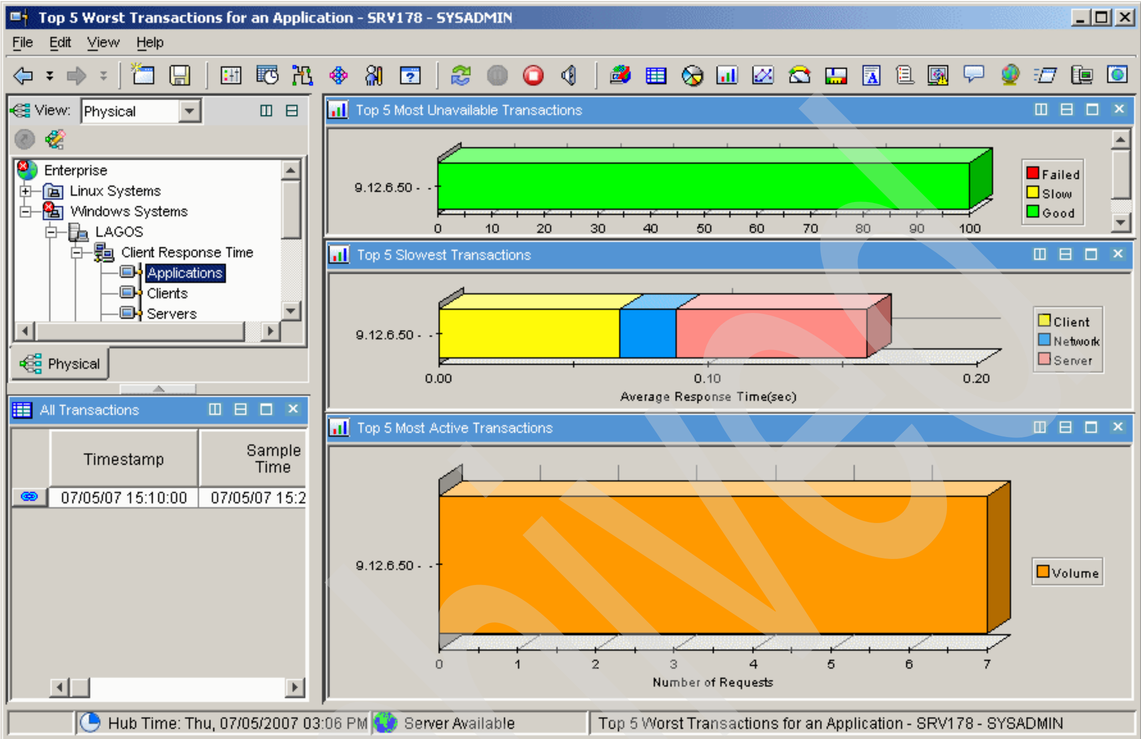


Figure 7-77 Client response time

8

ITCAM for Internet Service Monitoring

This chapter contains information about the implementation and monitoring usage of IBM Tivoli Composite Application Manager for Internet Service Monitoring in our environment. The discussion includes:

8.1 Internet Service Monitoring

ITCAM for Internet Service Monitoring is the new name for Netcool/Internet Service Monitors. It provides monitoring for Internet application availability and response monitoring. ITCAM for Internet Service Monitoring provides lightweight service availability testing for more than 20 TCP/IP-based protocols. This infrastructure integrates with other Netcool-based solutions for systems monitoring and network management.

For an overview of the Netcool solution for network and event management, refer to *Migrating to Netcool/Precision for IP Networks --Best Practices for Migrating from IBM Tivoli NetView*, SG24-7375.

ITCAM for Internet Service Monitoring probes Internet applications using a standard interface to collect data from:

- Host systems
- Applications
- Internet applications

8.2 ITCAM for Internet Service Monitoring architecture

ITCAM for Internet Service Monitoring consists of a set of interrelated components, each of which performs a monitoring, configuration, data processing, and reporting function. The main components of ITCAM for Internet Service Monitoring are the monitors, which regularly poll or test Internet services to check their status. The test results generate data for evaluation, reporting, and alert generation. This data can be used for service-level monitoring. Figure 8-1 shows the components of ITCAM for Internet Service Monitoring.

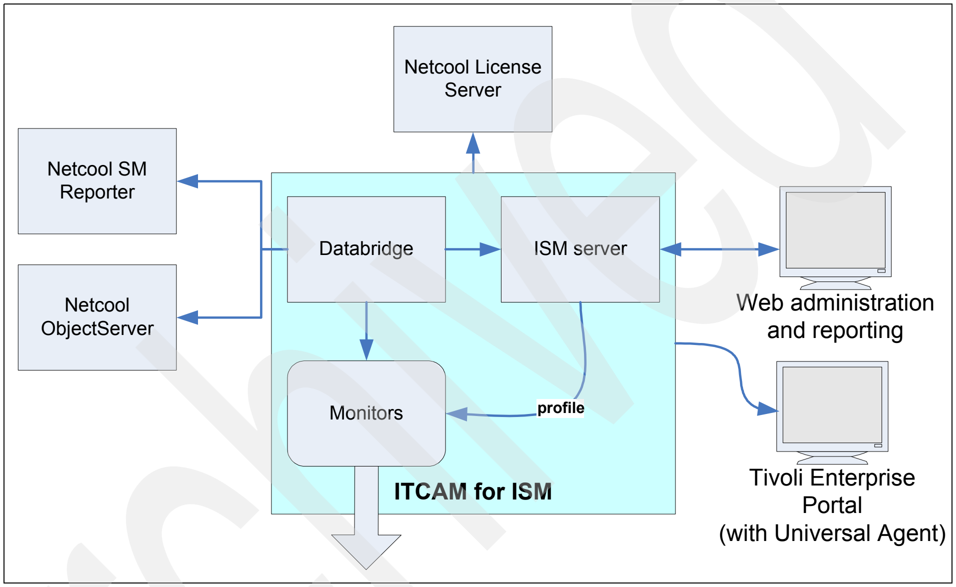


Figure 8-1 ITCAM for Internet Service Monitoring architecture

Based on Figure 8-1, the main components are:

- Monitors

  The monitors test specific Internet services and forward test results to the Databridge. They emulate the actions of a real user of the service. For example, the HTTP monitor periodically attempts to access a Web page by emulating requests that a Web browser would usually send when a user visits the Web page. It generates an event recording the result of the test, which is sent to the Databridge. The key functions of monitors are their polling functions. The monitors actively poll or test services at regular intervals by injecting transactions or queries into the target service and generating performance evaluation data.

- ISMServer

  ISMServer provides configuration, management, and reporting functions. It uses monitoring profiles for configuring and scheduling the testing performed by monitors. The profiles include criteria for evaluating SLAs and service availability based on the results of each monitor's service tests. The ISMServer provides a Web-based interface for accessing the ISMServer configuration, management, and reporting functions. All reports and status pages are provided in the Web browser-based interface.

- Databridge

  The Databridge is the communications bridge between the monitors with the ISMServer and any other component that requires data from the monitors. It also interfaces with the ObjectServer and Netcool/SM Reporter. It receives the results of service tests performed by the monitors, converts this data into different formats for processing by ISMServer, an ObjectServer, or Netcool/SM Reporter. The formats are:

  – XML data logs, which ISMServer uses to compile SLA views and performance graphs.

  – Alerts are sent to ObjectServers.

  – Service Monitor Data Records (SMDRs), which Netcool/SM Reporter uses to generate reports.

- Netcool License Server is a stand-alone server component that provides licensing functionality for the Netcool portfolio of products. This component is based on the premise that license administration and maintenance can be simplified by centralizing license data on one or more designated license servers, with licenses being drawn from a server as necessary. This is a required component for running any Netcool software.
- Netcool SM Reporter is a comprehensive reporting solution for the Netcool product.
- Netcool ObjectServer is an interface to store overall monitoring information for Netcool products.

The interface to IBM Tivoli Monitoring Server is provided by the Tivoli Open Process Automation Library, available at:

http://catalog.lotus.com/topal?NavCode=1TW10TM31

8.3 Installing ITCAM for Internet Service Monitoring

In our environment, we decided to run only the minimal ITCAM for Internet Service Monitoring component. This includes installing the license server and the main ITCAM for Internet Service Monitoring component on a single machine. We did not implement the ObjectServer or the SM Reporter. We discuss the installation in the following sections:

- 8.3.1, “Installing Netcool License Server” on page 417
- 8.3.2, “Installing Internet Service Monitoring” on page 422
- 8.3.3, “Problem determination and log files” on page 428

8.3.1 Installing Netcool License Server

The Netcool License Server must be installed and configured before running any Netcool product in your environment. You must also ensure that you have the required license files containing the license feature codes for the Netcool products and related components that you are authorized to use.

The license server can be a single stand-alone system or installed as a redundant server to accommodate fault tolerance. We implemented the single license server scenario, which is the simpler configuration, recommended for smaller implementations.

To install the Netcool License Server:

1. Download the TAR and LIC files. Extract the TAR file into a temporary directory.
2. To start the installer, execute the following command from the temporary directory to which you extracted the archive:

   ```
   ./INSTALL [-errorlevel (debug|info|warning}]
   ```

   Use the optional -errorlevel parameter to set the level of detail in the installation log file for information or troubleshooting purposes.

3. The installer prompts you for the location of the NCHOME variable, if not defined. Accept the proposed location by pressing Enter, or provide a new location and press Enter. We use the default NCHOME variable of /opt/netcool. Click **Next** on the installer window, as shown in Figure 8-2.

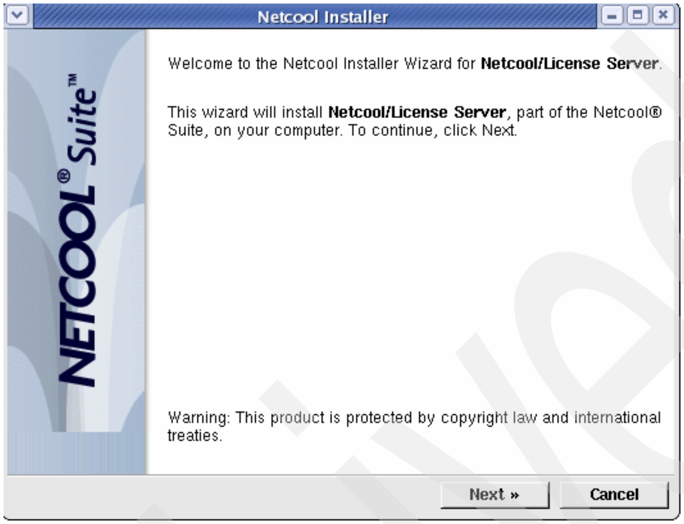


Figure 8-2 Net cool License Server Installation

4. Select **I accept the terms in the license agreement** and click **Next**, as shown in Figure 8-3.

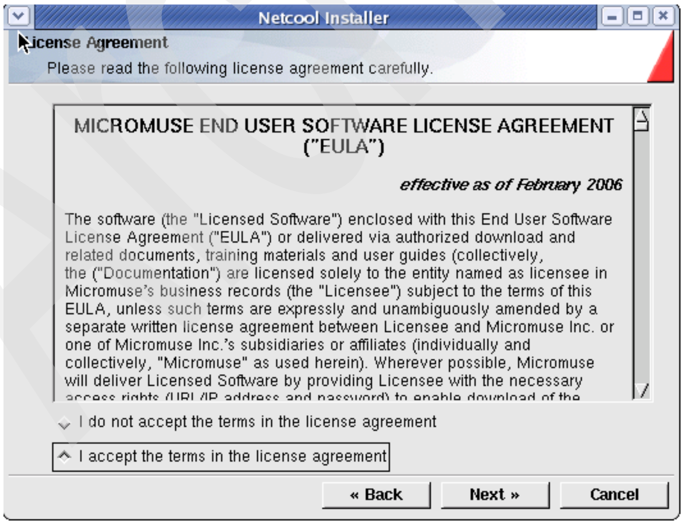


Figure 8-3 Net cool License Server installation

5. The installer displays the location of the NCHOME variable. Click **Next**, as shown in Figure 8-4.

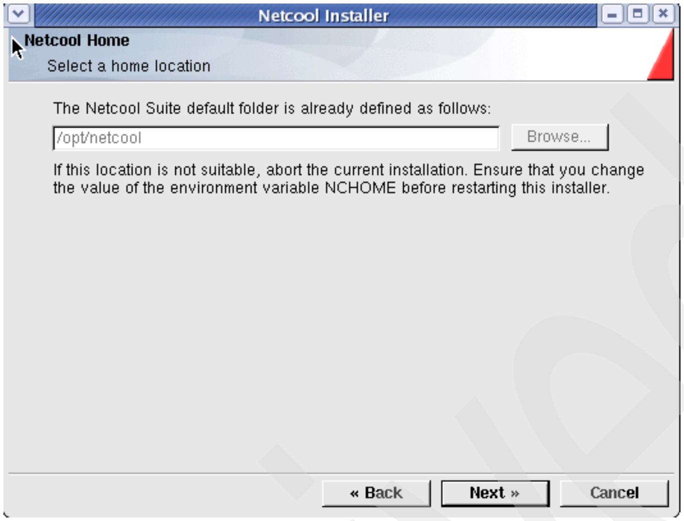


Figure 8-4 Netcool License Server Installation

6. The installation window shows information about the setup type of installation. We select the **Default** option, as shown in Figure 8-5.

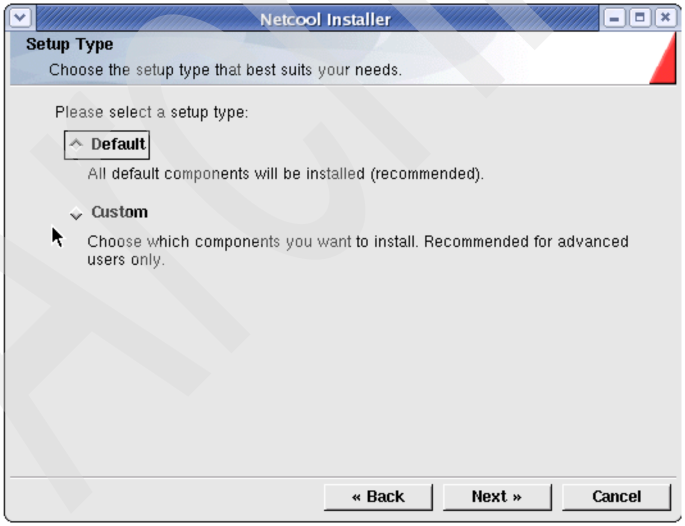


Figure 8-5 Netcool License Server installation

7. The Netcool License Server installer provides a summary of the operations it will perform during the installation process. Review the installation settings, as shown in Figure 8-6, and click **Install** to complete the installation.

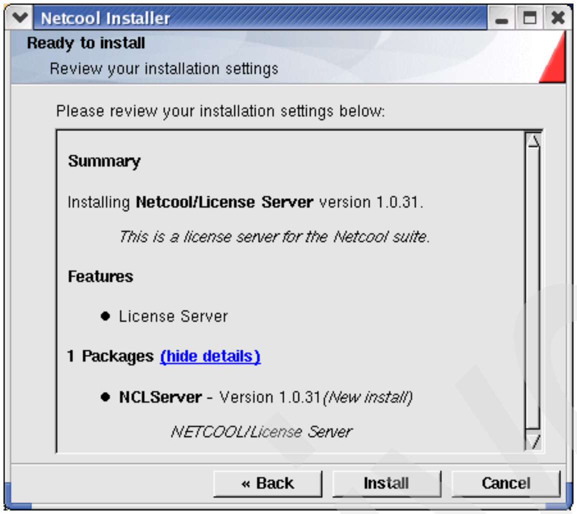


Figure 8-6 Installation review of Netcool License Server

8. Once the installation is complete, the following window appears (Figure 8-7). Click **Finish**.

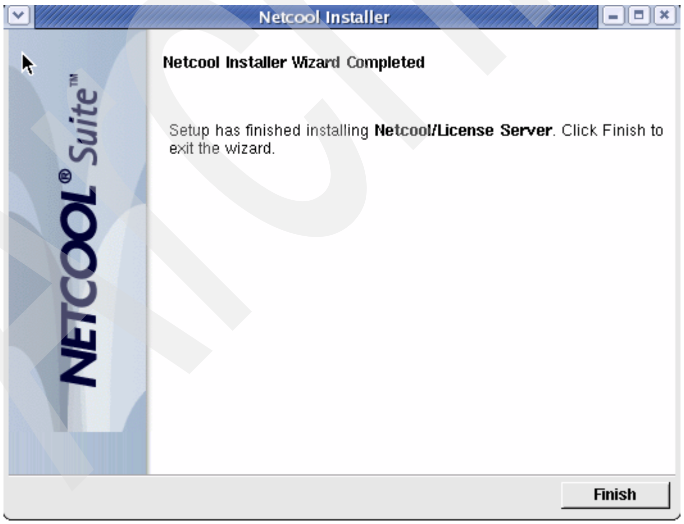


Figure 8-7 Net cool License Server installation

9. The Netcool License Server files are stored in the location $NCHOME/license. In our case, it is /opt/netcool/license. Save the license file (.lic) in the $NCHOME/license/etc directory of the system running the Netcool License Server. Use a unique file name.
10. Make sure to edit the license file (.lic) and add your server name, as shown in Example 8-1.

Example 8-1 Sample contents of .lic file

```
SERVER khartoum.itsc.austin.ibm.com ANY 27000
VENDOR netcool
USE_SERVER
FEATURE ncism_dhcp netcool 1.0 permanent 9999 ck=177 SIGN=35C0CCFC9DC0
```

11. Start the license server by entering following command:

    ```
    $NCHOME/license/bin/nc_start_license &
    ```

    When the license server starts, it reads all license files in $NCHOME/license/etc and then waits for incoming license requests. The license server writes information to license.log in the $NCHOME/license/log directory.

    > **Note:** When checking for the license server process using the UNIX **ps** command, the license server daemon is identified as lmgrd.

12. Verify the installed license file. Enter following command:

```
$NCHOME/license/bin/nc_print_license
```

This command lists all served licenses of the Netcool License Server. Figure 8-8 shows a excerpt of our configuration.

```
ncadmin@srv179:/opt/netcool/license/bin #./nc_print_license
lmstat - Copyright (c) 1989-2003 by Macrovision Corporation. All rights
reserved.Flexible License Manager status on Fri 11/3/2006 15:49
License server status: 27000@srv179
License file(s) on srv179: /opt/netcool/license/etc/license.lic:
srv179: license server UP (MASTER) v9.2
Vendor daemon status (on srv179):
netcool: UP v9.2
Feature usage info:
Users of ncism_dhcp: (Total of 9999 licenses issued; Total of 0 licenses in use)
Users of ncism_dns: (Total of 9999 licenses issued; Total of 0 licenses in use)
Users of ncism_ftp: (Total of 9999 licenses issued; Total of 0 licenses in use)
Users of ncism_http: (Total of 9999 licenses issued; Total of 0 licenses in use)
.....
```

Figure 8-8 Command nc_print_license

8.3.2 Installing Internet Service Monitoring

This section describes the installation of ITCAM for Internet Service Monitoring. We use ITCAM for Internet Service Monitoring V2.4 on Red Hat Enterprise Linux V3. The following steps outline the installation of ITCAM for Internet Service Monitoring:

1. Create the necessary user ID for ITCAM for Internet Service Monitoring. ITCAM for Internet Service Monitoring requires a user ID as the installation owner and the default monitor process. We use:

   ncadmin Owner of ITCAM for Internet Service Monitoring files and directories

   netcool Default group for user ID ncadmin

   Use the appropriate operating system facility to create the group and user. In our environment, we use the **groupadd** and **adduser** commands. These commands create the group and user as follows:

   ```
   groupadd netcool
   adduser -c Netcool_Admin_ITSO -d /home/ncadmin -g netcool -n -p
   xxxxxx -s /bin/sh -u 10500 ncadmin
   ```

2. Start the installer. You can invoke the installer in a graphical interface or using a console mode. The default is the GUI installer. Invoke the console mode

using the -console option. Invoke the installer using the INSTALL command from the archive directory.

3. The installer prompts you for the location of the NCHOME variable, if not defined. Accept the proposed location by pressing Enter, or provide a new location and press Enter. Our location of variable NCHOME is defined with the value of /opt/netcool. Because we install this in the same machine as our license server, the NCHOME variable should already been configured.
4. The installation process window opens, as shown in Figure 8-9. Follow the instructions by clicking **Next**. After agreeing to the terms of the license, we select the default setup type.

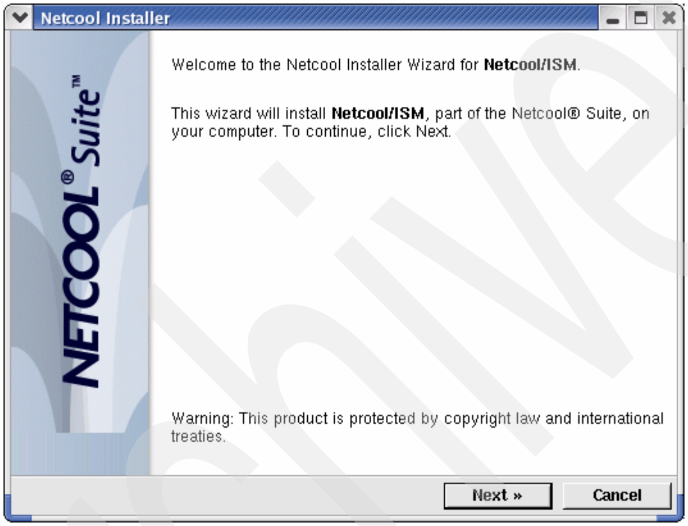


Figure 8-9 Netcool/Internet Service Monitor welcome window

5. Enter the required settings and configuration options. The installer presents a series of dialog boxes and menus that provide options and settings for configuring the Netcool/Internet Service Monitor installation. In our environment, we do not have an ObjectServer. We use the default settings for ObjectServer, as shown in Figure 8-10.

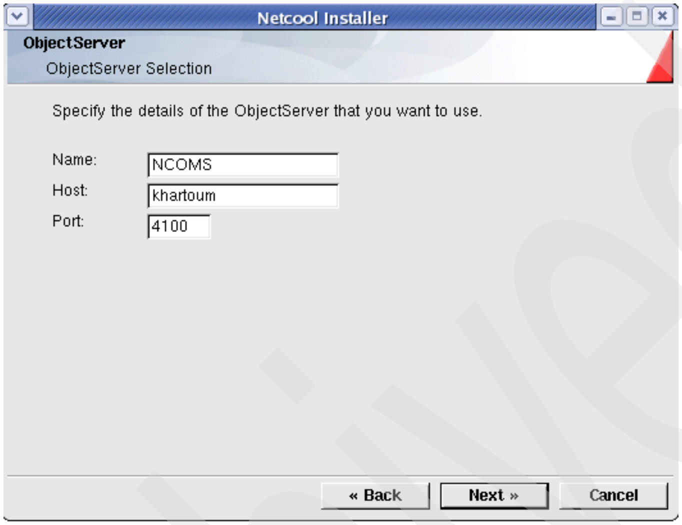


Figure 8-10 ObjectServer default configuration

Table 8-1 provides a description of the options and settings.

Table 8-1 Netcool/Internet Service Monitor installation options and settings

Option or setting name	Description
ISMServer Distribution Port	The port on which ISMServer distributes profiles.
ISMServer Port	The port that ISMServer uses to serve the browser-based interface.
ObjectServer Host	The DNS name or IP address of the ObjectServer to which the Databridge forwards events.
ObjectServer Name	The name of the ObjectServer to which the Databridge and ISMServer send events and SLA data.
ObjectServer Port	The ObjectServer port to which the Databridge forwards events.
Run Netcool/ISM as User	Specifies the name of the user under which all components of Netcool/ISM run. Note: The DHCP, ICMP, and TRANSX monitors always run as root.

Option or setting name	Description
Standard Time Zone	The standard time zone used by ISMServer when producing SLA data and reports.
Summer Time Zone	The time zone used by ISMServer when producing SLA data and reports during the summer time.

6. The next configuration window asks for the appropriate monitor user ID. In addition, the installer provides a selection of features to install, such as ISM Server reporting, Object Server connectivity, and database module. Figure 8-11 shows the settings of our environment.

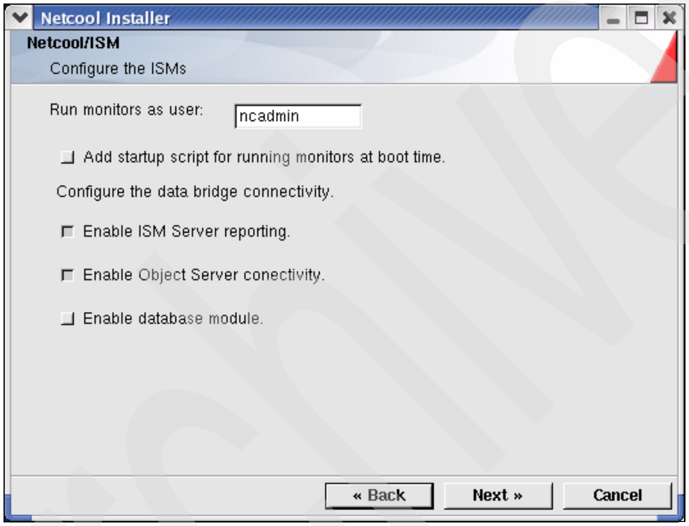


Figure 8-11 Configure ISMServer user ID

7. In the last configuration window, specify the settings for communication ports and time zones. We use the default port settings, as shown in Figure 8-12.

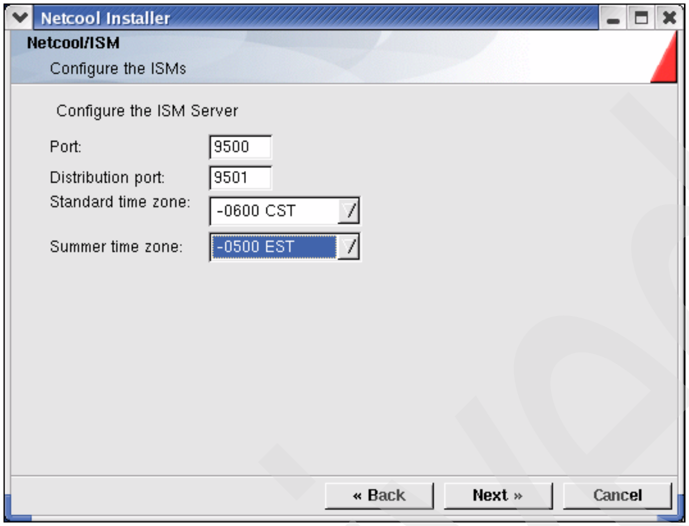


Figure 8-12 Configure ISMServer settings

8. The ITCAM for Internet Service Monitoring installer provides a summary of the operations it will perform during the installation process. Review the installation settings, as shown in Figure 8-13, and click **Install** to complete the installation.

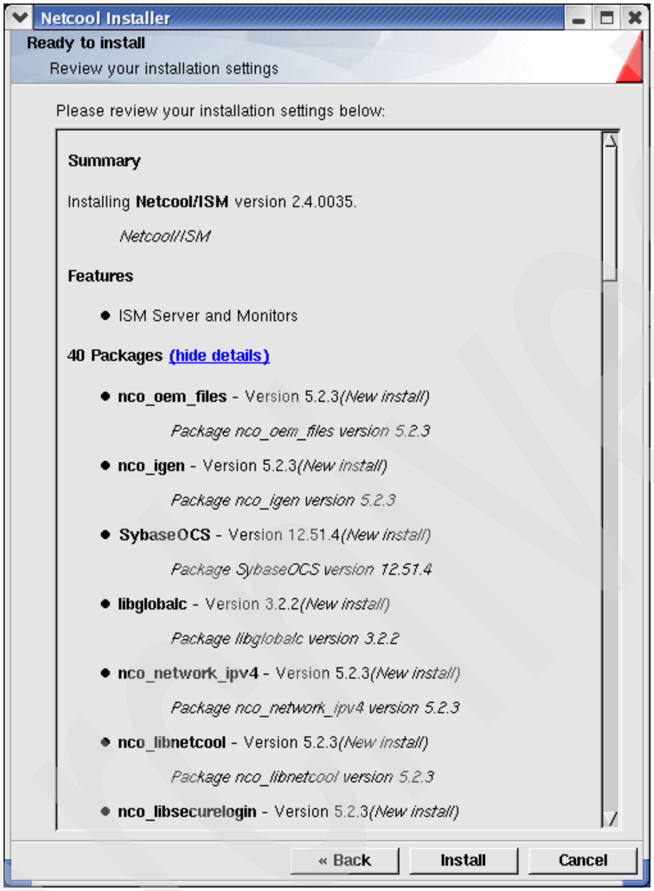


Figure 8-13 Installation review

9. Start the ITCAM for Internet Service Monitoring with following shell script:

```
$NCHOME/ism/bin/ism_startup.sh start
```

This script starts the Databridge, the Netcool/Internet Service Monitor monitors, and the ISMServer. In our environment, the value of variable NCHOME is /opt/netcool.

We discuss the Web console interface in 8.4, “Configuration and usage” on page 429. The default user name and password are ism and netcool.

8.3.3 Problem determination and log files

This section provides some tips for problem determination and the log file locations that we encounter during our work:

- The ISMServer, monitors, and Databridge provide their log files in the directory $NCHOME/log/ism. In our case, the directory is /opt/netcool/log/ism:
  - bridge.log
  - ismserver_startup.log
  - ismserver.log
- For problems in the Databridge from executing the ism_startup.sh shell script. This includes when the ISMServer seems not to be running, or the monitors collect no information data. In this case, check the property file $NCHOME/etc/ism/props/bridge.props for the Databridge. Check the setting for variables Module0SharedLib and Module0PropFile. Example 8-2 shows the content.

  Example 8-2 The bridge.props

  ```
  Manager: 'Internet Service Monitors'
  MaxLogFileSize: 1048576
  MessageLevel: 'warn'
  MessageLog: '/opt/netcool/log/ism/bridge.log'
  Module0SharedLib:'libSMModuleDatalog'
  Module0PropFile:''
  Module1SharedLib:'NONE'
  Module1PropFile:''
  Module2SharedLib:'NONE'
  Module2PropFile:''
  ```

  For running a configured ObjectServer, the setting for variables Module0SharedLib and Module0PropFile in the property file are as shown in Example 8-3.

  Example 8-3 bridge.props with ObjectServer

  ```
     . . .
  Module0SharedLib:'libSMModuleObjectServer'
  Module0PropFile:'/opt/netcool/etc/ism/props/objectserver.props'
     . . .
  ```

- You can also try to start the Databridge manually using the user ID for the ISMServer. Therefore, be sure to set the appropriate runtime definitions for the variables LD_LIBRARY_PATH, LIBPATH, and NCHOME. Use the following commands or a self-made script (Example 8-4).

Example 8-4 Starting the Databridge manually

```
export NCHOME=/opt/netcool

LIBPATH=$NCHOME/platform/linux2x86/lib:$LIBPATH
export LIBPATH=$NCHOME/ism/platform/linux2x86/lib:$LIBPATH

LD_LIBRARY_PATH=$NCHOME/platform/linux2x86/lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH=$NCHOME/ism/platform/linux2x86/lib:$LD_LIBRARY_P
ATH
##
## start the databridge
##
$NCHOME/ism/platform/linux2x86/bin/nco_m_bridge
```

8.4 Configuration and usage

This section illustrates the usage of ITCAM for Internet Service Monitoring using the Web-based interface. The discussions include:

- 8.4.1, "Defining profiles and profile elements" on page 429
- 8.4.2, "Status reports" on page 433
- 8.4.3, "Using Tivoli Enterprise Portal" on page 440

8.4.1 Defining profiles and profile elements

The usage scenario performs a sample configuration for monitoring and reporting a Web application using HTTP protocol. We have a distributed application and we want to manage its response time with ITCAM for Internet Service Monitoring.

The following steps outline the procedure:

1. To access ISMServer, open a Web browser and enter the URL of the ITCAM for Internet Service Monitoring server with the port defined, while installing the server. In our case, we use the default port of 9500. Our URL is:

   `http://khartoum.itsc.austin.ibm.com:9500/`

Figure 8-14 shows the login page.

Figure 8-14 ITCAM for Internet Service Monitoring login page

2. The default user name and password are ism and netcool. After successfully logging in, the Profiles page opens with an empty list.
3. Define the first profile for the environment to be monitored using **Create a new profile**. We create a new profile named netc1, as shown in Figure 8-15.

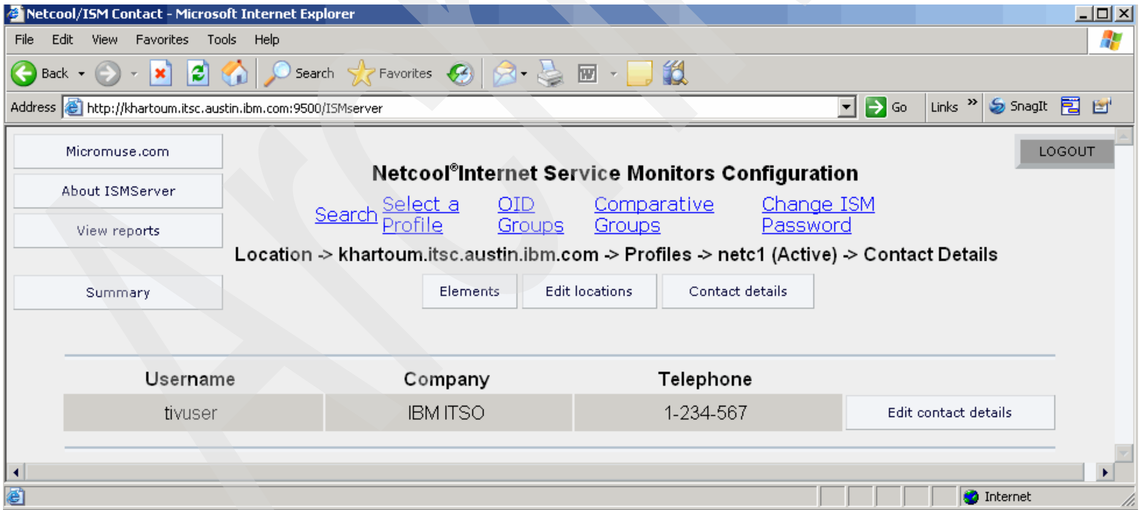


Figure 8-15 Netcool/ISM view profiles

4. Choose **Select profile** for opening a specific profile. Click **Add new element** on the profile elements page, as shown in Figure 8-16.

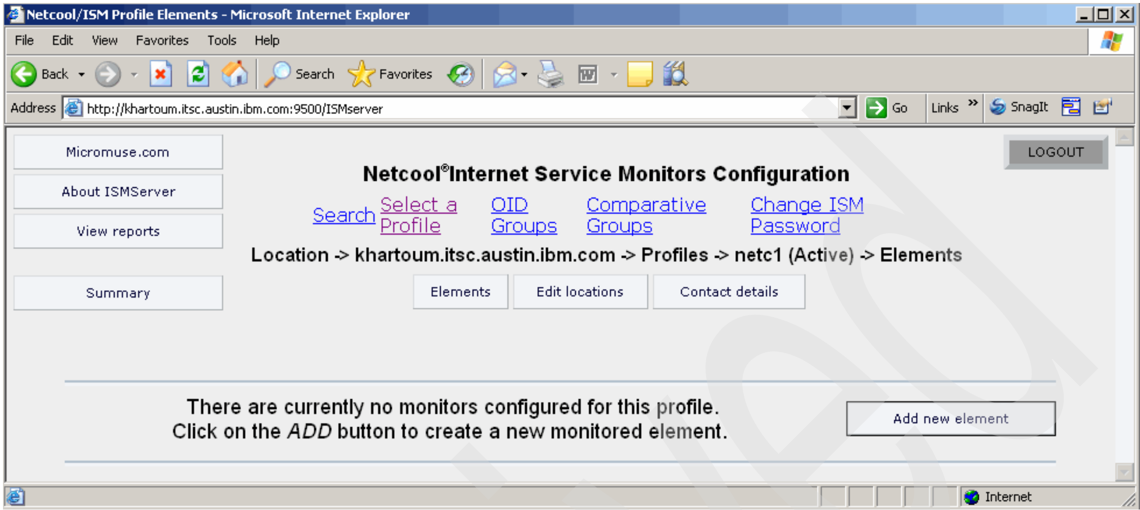


Figure 8-16 Profile page

5. Click the button corresponding to adding the monitor appropriate to the type of protocol. We select the **HTTP** protocol (Figure 8-17) to configure the monitor.

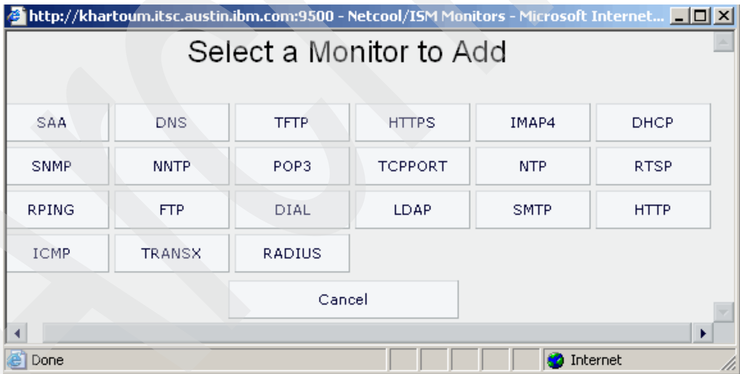


Figure 8-17 Selecting a monitor

6. The window to edit the element would opens. Enter the monitor configuration appropriate to your monitoring task. Our sample configurations are shown in Figure 8-18.

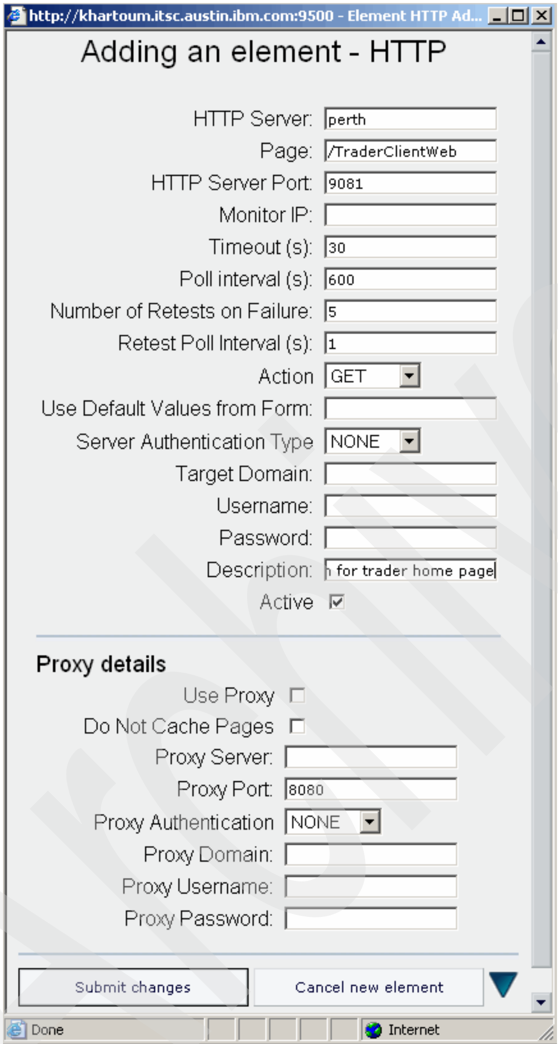


Figure 8-18 Editing an element - HTTP

The configuration is activated immediately and the monitor starts polling the Web application. ITCAM for Internet Service Monitoring displays the status of each profile element using color-coded indicators. Over time, as the status of the service changes, the colors of these indicators also change so that you can see how the service is performing in relation to service level. This performance is shown as both the current status and the service level compliance over the last hour, as shown in Figure 8-19.

> **Note:** In case your monitor does not show anything (the status is displayed as unknown), make sure that all the components, like license server, ITCAM for Internet Service Monitoring server, and datalog component, are running. Check the log files in the $NCHOME/log/ism folder to see whether there is any error while starting bridge, ismserver, or http monitor. Also make sure that the DATALOG entry in $NCHOME/etc/ism/props/ http.props is enabled. You can enable it by adding the DATALOG : 1 line, if its not there already.

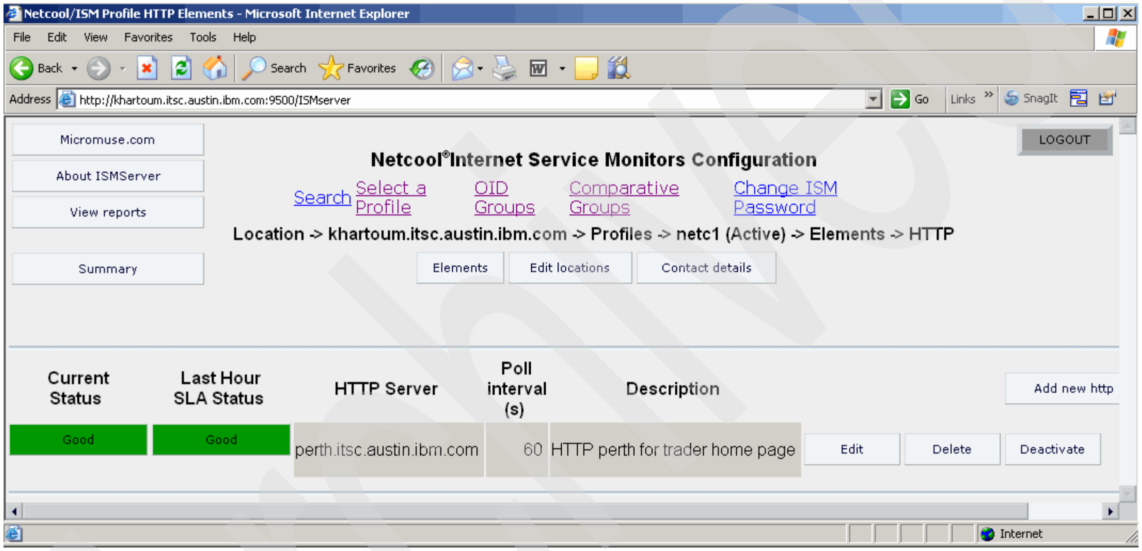


Figure 8-19 Profile element page

8.4.2 Status reports

Using ITCAM for Internet Service Monitoring, you can view the status reports that provide information about how the services are performing. Reports provide SLA status indicators, detailed service level graphs, and performance graphs.

In this section we first discuss how to create SLAs. Then we look at different graphs that can be used to monitor different aspects of the services.

Creating SLAs for a profile

To do this:

1. Select the profile for which you want to define the SLA, as shown in Figure 8-20.

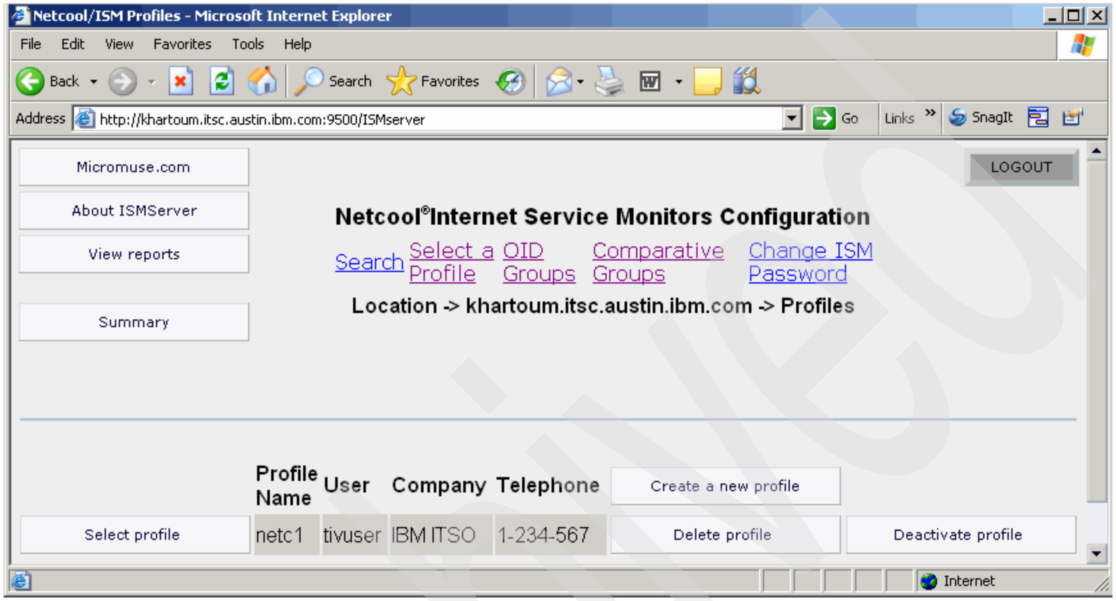


Figure 8-20   Select a profile

2. Select the element from the Elements page, as shown Figure 8-21.

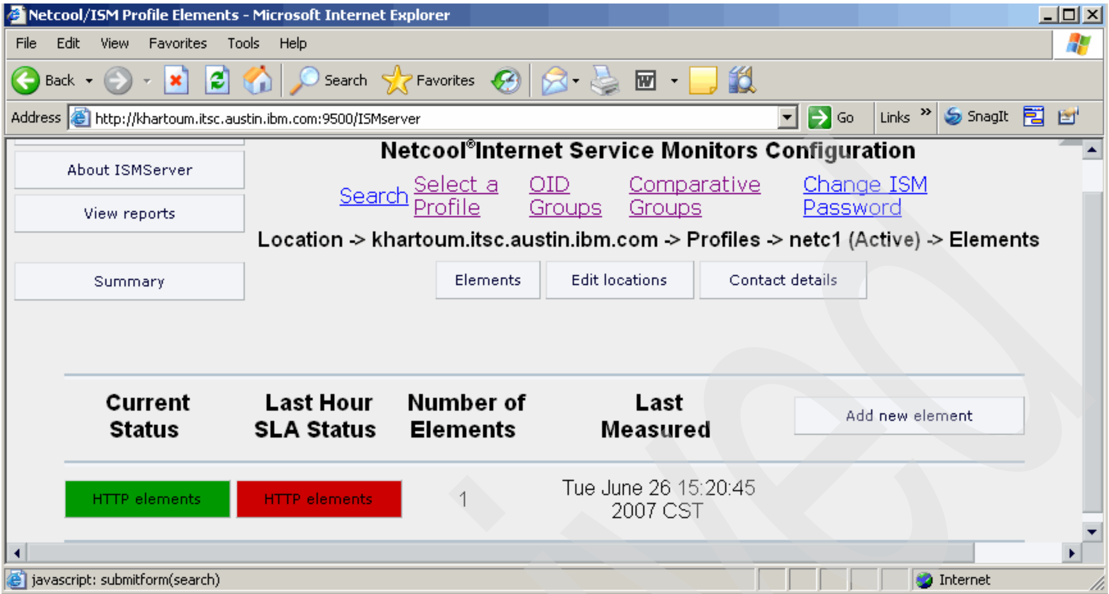


Figure 8-21 Element page

3. Once you have selected the element, click **Edit**. Editing an element - HTTP shows up. Click the arrow icon displayed next to the Cancel changes button. The editing window expands, as shown in Figure 8-22. You can define multiple conditions using the if else if block. You can use the IGNORE keyword in the condition drop down to ignore a particular test.

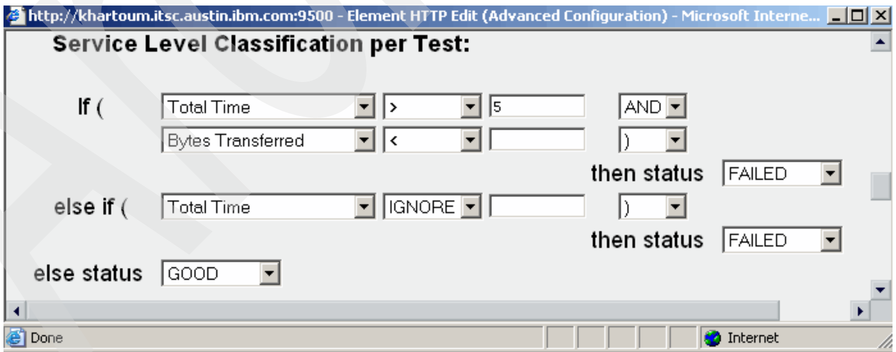


Figure 8-22 Editing element - service level classification

4. Once you have defined the conditions, which define the status as GOOD or FAILED, and so on, you can define the SLAs, as shown in Figure 8-23.

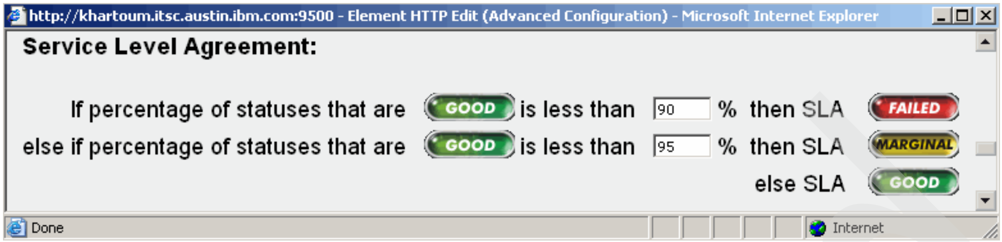


Figure 8-23 SLA

To view how your application is performing against SLAs, perform the following steps:

1. Select the **Last hour SLA Status** button in the Profile Element page. The report for measuring the SLA statistics opens. You can get the SLA report based on the SLAs defined and how your application is performing (Figure 8-24).

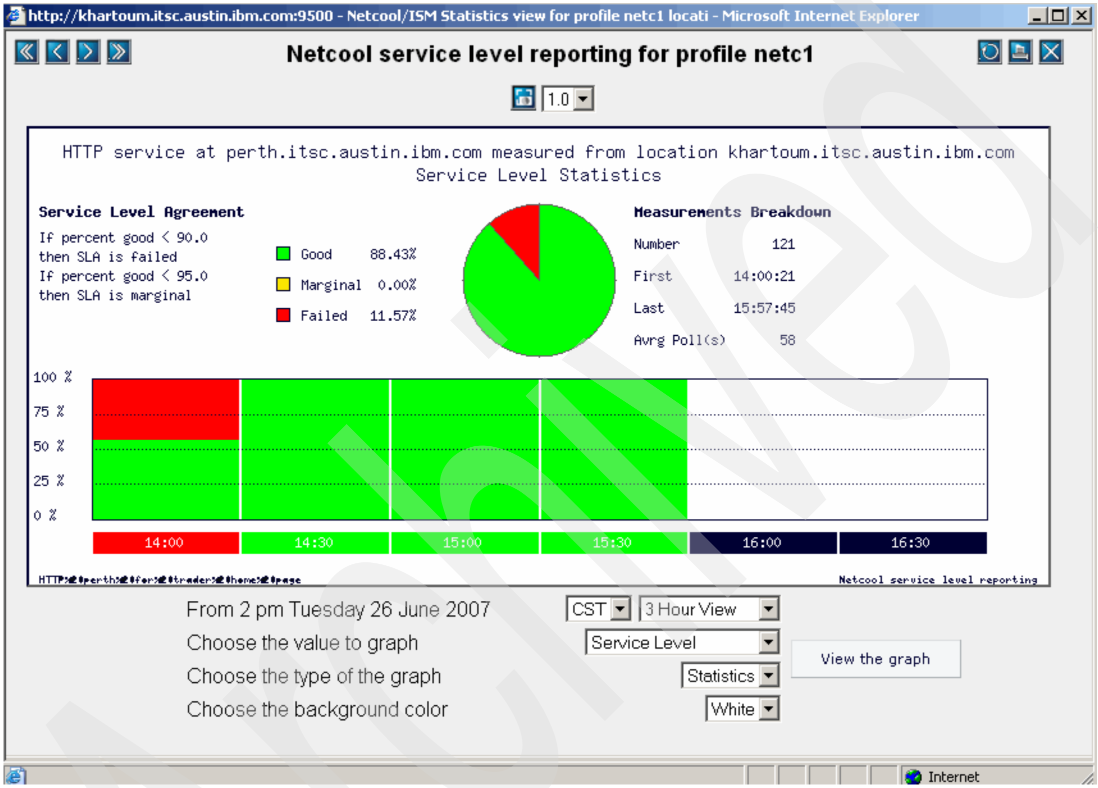


Figure 8-24 SLA report

2. You can view different parameters related to your service performance by choosing appropriate options from drop-down boxes in the SLA report.

3. Beginning from the Profile Element pages, click the **Current Status** to view the response time report, as shown in Figure 8-25.

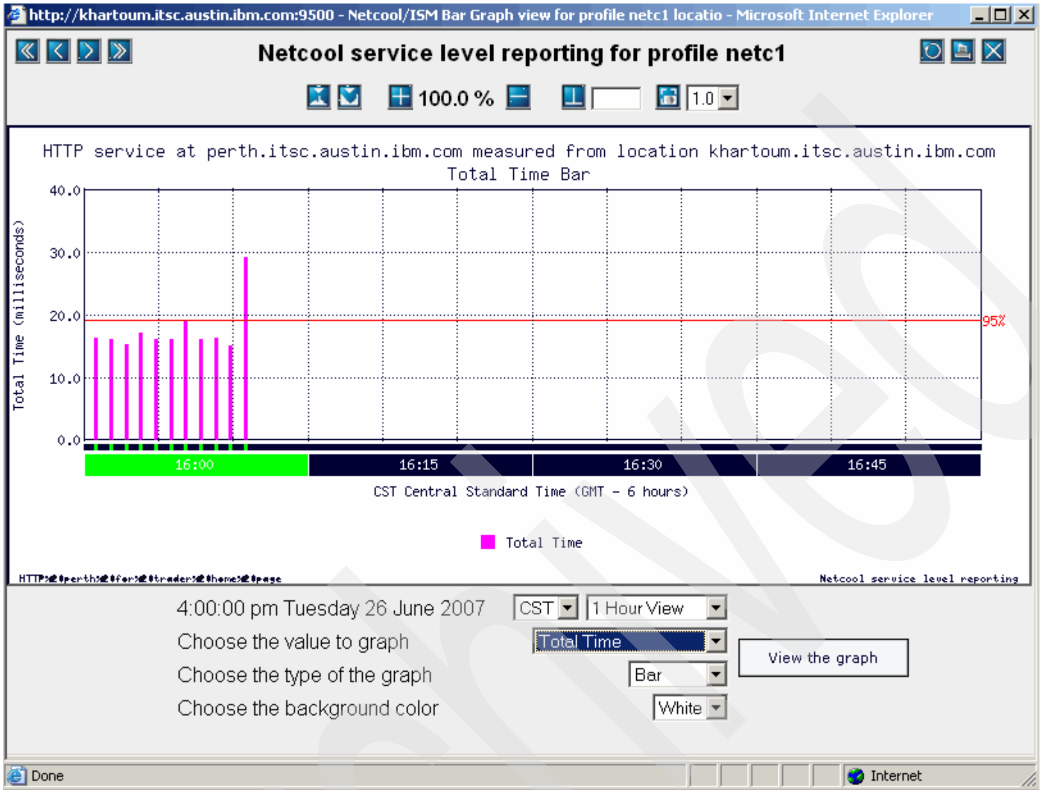


Figure 8-25 Current status report

4. You can also view the report using the calender view.Click **View reports** on the top left side of the ITCAM for Internet Service Monitoring home page. A calender is displayed. Select the date for which you want to view the report, as shown in Figure 8-26.

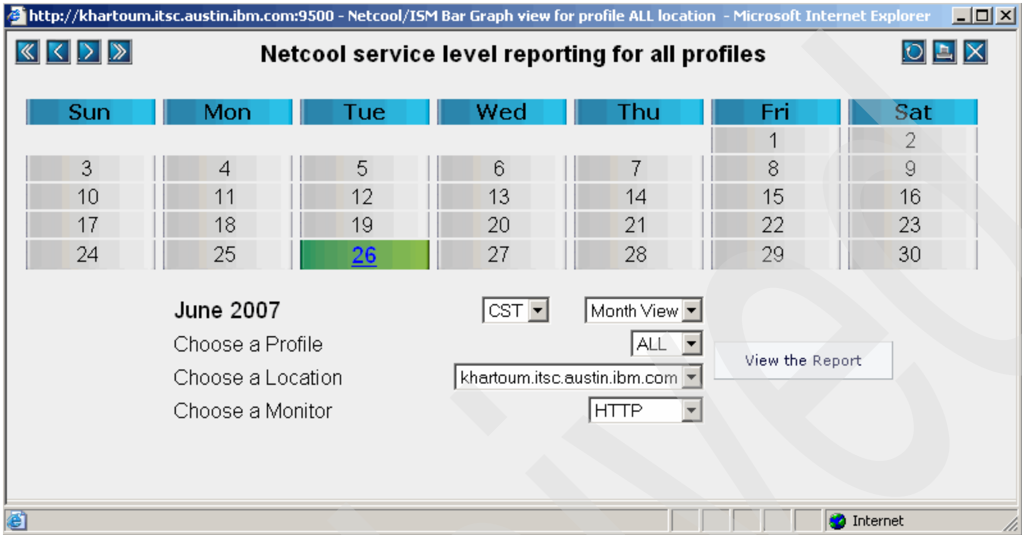


Figure 8-26 Calender View for reports

5. Select the appropriate profiles and monitor and click the date.

6. You get the SLA report for the selected date, as shown in Figure 8-27.

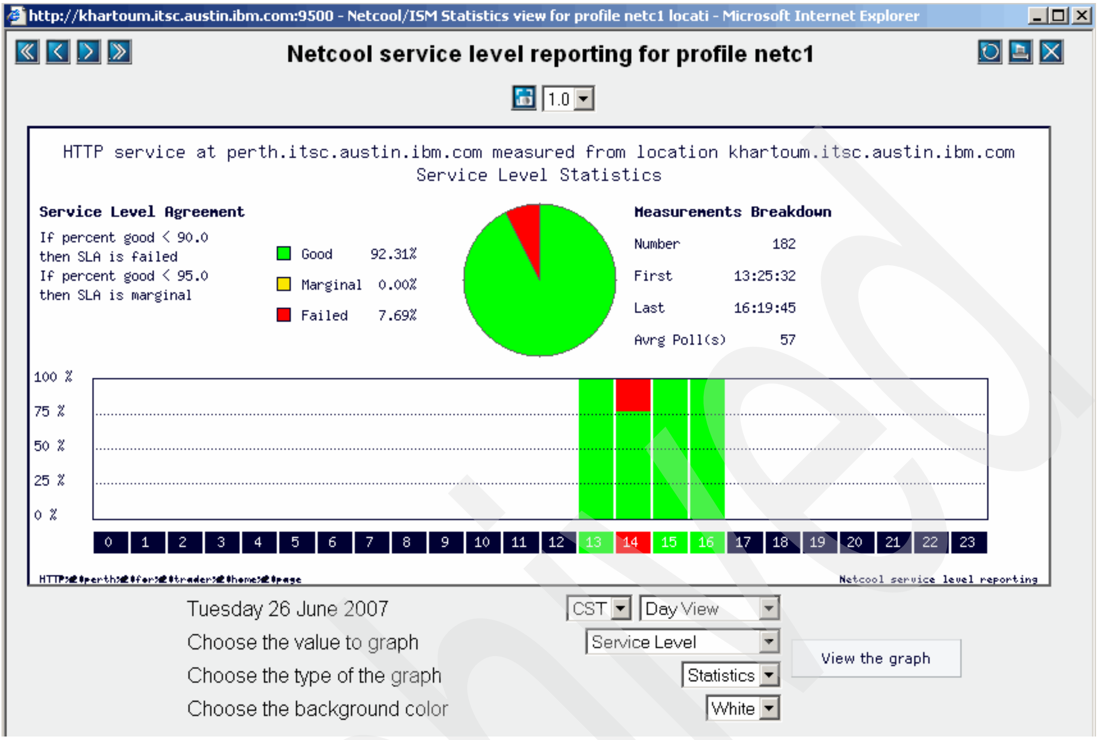


Figure 8-27 Date wise SLA

8.4.3 Using Tivoli Enterprise Portal

This section explains how to use ITCAM for Internet Service Monitoring with Tivoli Enterprise Portal. If you already have a Tivoli Enterprise Portal installed in your environment, you can use IBM Tivoli Monitoring Universal Agent to send data from ITCAM for Internet Service Monitoring to the Tivoli Enterprise Portal. This topic assumes that you have the following components installed and running and does not detail the installations of these components.

- ITCAM for Internet Service Monitoring and its components like license server, monitor, and so on.
- IBM Tivoli Monitoring (ITM).
- Universal Agent is running and configured to connect to TEMS.

To enable integration between ITCAM for Internet Service Monitoring and IBM Tivoli Monitoring and to view the data of ITCAM for Internet Service Monitoring with Tivoli Enterprise Portal, you need to install the ITCAM for Internet Service

Monitoring Tivoli Enterprise Portal Server Integration module, available from IBM OPAL site at the following address:

http://catalog.lotus.com/tm?NavCode=1TW10TM31

This module is called IBM Tivoli Monitoring Workspace Package for ITCAM for Internet Service Monitoring, and provides a subscriber program that receives data from ITCAM for Internet Service Monitoring and pushes it to IBM Tivoli Monitoring Universal Agent via the socket data provider. This package also provides IBM Tivoli Monitoring workspaces to visualize this data.

To install this package on a Linux platform:

1. Extract the tar file downloaded from the above link and use the file corresponding to your platform (ISM_ITMPATCH-1.0.0-linux2x86.tar for Linux).
2. Untar the ISM_ITMPATCH-1.0.0-linux2x86.tar.
3. Run ./INSTALL and the installer window opens, as shown Figure 8-28. Click **Next**.

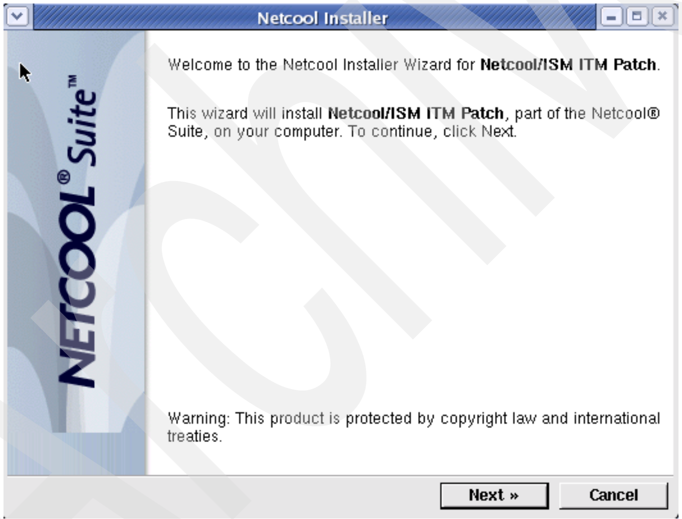


Figure 8-28 Installer window for IBM Tivoli Monitoring patch

4. Select **I accept the terms in the license agreement** and click **Next**, as shown in Figure 8-29.

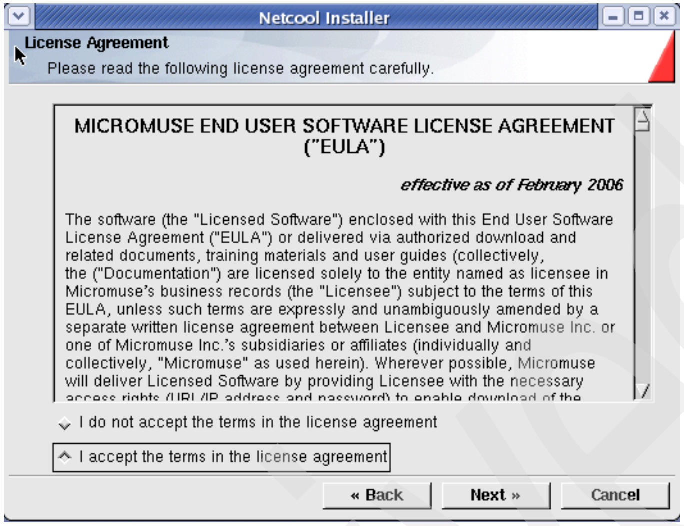


Figure 8-29 License agreement for patch

5. We choose the setup type as Default, as shown in Figure 8-30.

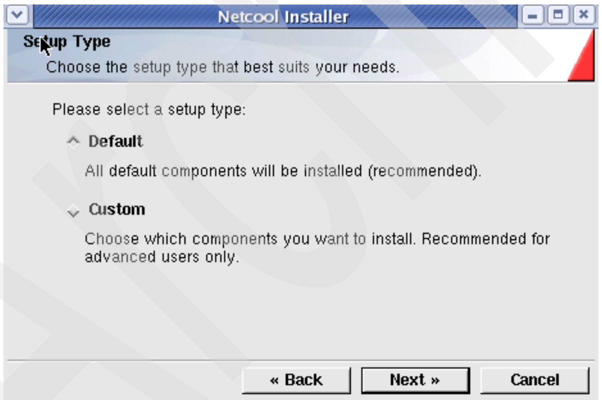


Figure 8-30 Installation type for patch

6. The Configure the ISMs window opens. Enter the host name of where IBM Tivoli Monitoring is running and the port where the Universal Agent is running (default is 7500). Select **Overwrite bridge.props** and **Enable ITM server connectivity**. Do not select Enable Object Server connectivity, unless you have Netcool/Omnibus server in your environment. The configuration window is shown in Figure 8-31.

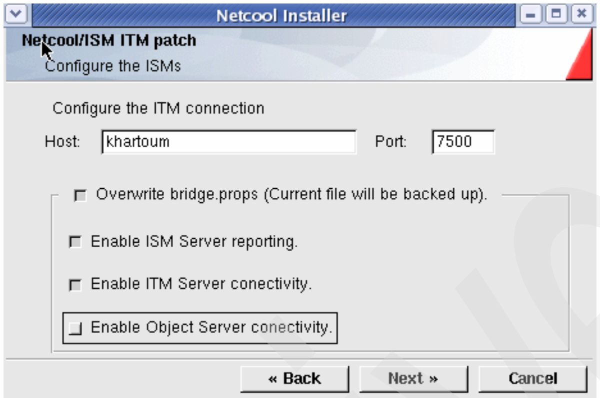


Figure 8-31 Installer option for patch

7. Click **Next** to start the actual copying and configuration and click **Finish** when the completion window appears.

Once the patch is installed, the next step is to configure it to start accepting data being sent by ITCAM for Internet Service Monitoring. This can be done by using the $NCHOME/ism/itm/itmconfig script. Follow the steps below.

> **Note:** Following these steps assumes that your ITCAM for Internet Service Monitoring and IBM Tivoli Monitoring are on same machine. If this is not the case, copy the $NCHOME/ism/itm folder where IBM Tivoli Monitoring is installed, and perform the following steps.

1. Change to the $NCHOME/ism/itm folder.
2. Ensure that the CANDLEHOME variable is set correctly. The default is /opt/IBM/ITM.
3. Run `./itmconfig.sh -u` to configure the Universal Agent.
4. Run `./itmconfig.sh -p` to configure the Tivoli Enterprise Portal Server.

Note: While running the **`./itmconfig.sh -u`** command on Linux, you might get the error message:

```
KUMPV599E - Error Could not find any kumpmsg resource bundles in
directory ./rb Error code is U_MISSING_RESOURCE_ERROR
```

In that case you have to run the um_console manually, as shown in Example 8-5.

Example 8-5 Sample output of running $CANDLEHOME/bin/um_console command

```
ot@peoria bin]# ./um_console
KUMPS002I Enter console command <Application name or Metafile name
or file name>
import /tmp/ITM_ISMPatch/itm/ism.mdl
KUMPS001I Console input accepted.
KUMPS020I Import successfully completed for
/tmp/ITM_ISMPatch/itm/ism.mdl

Press ENTER to exit
```

5. Once you have configured the Universal Agent and Tivoli Enterprise Portal Server, restart your ITCAM for Internet Service Monitoring, Tivoli Enterprise Portal Server, and Universal Agent.

Some of the default ITCAM for Internet Service Monitoring workspaces uses the Tivoli Data Warehouse feature. To configure this feature on IBM Tivoli Monitoring:

1. Run **`$CANDLEHOME/bin/CandleManage`**.

2. From Manage Tivoli Enterprise Monitoring Services, select **Summarization and Pruning Agent**, and select **Configure**, as shown in Figure 8-32.

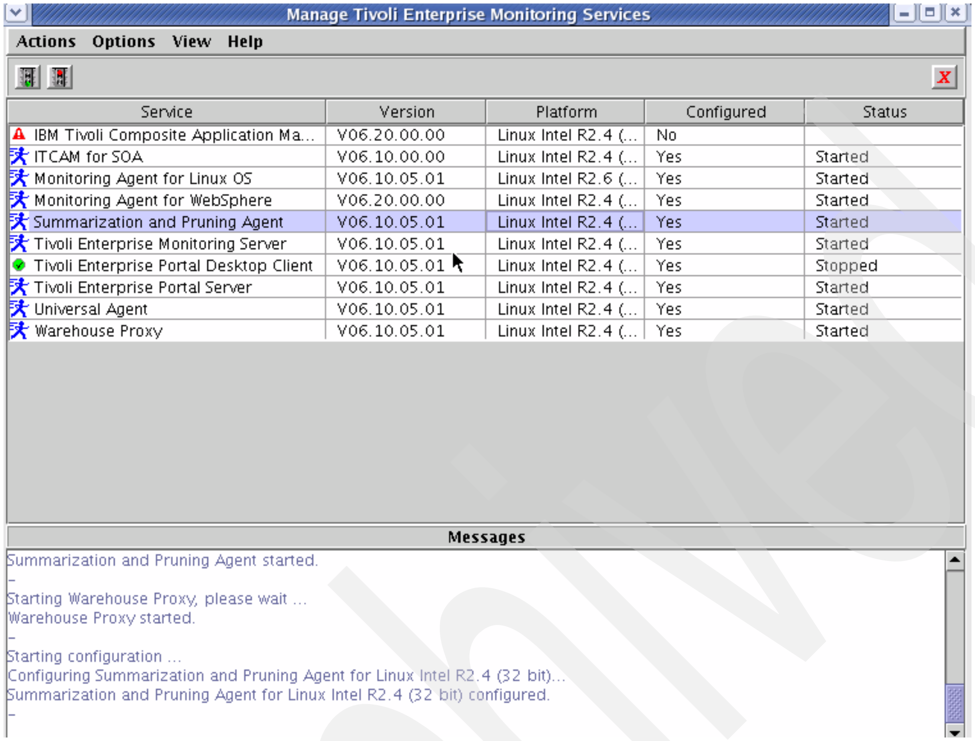


Figure 8-32 Configuring Summarization and Burning Agent

3. Select **Agent Parameters** → **Sources** tab and make sure that all the entries are correct and pointing to your Tivoli Data Warehouse. Make sure to click **Test database connection** to check that the database connection is working (Figure 8-33).

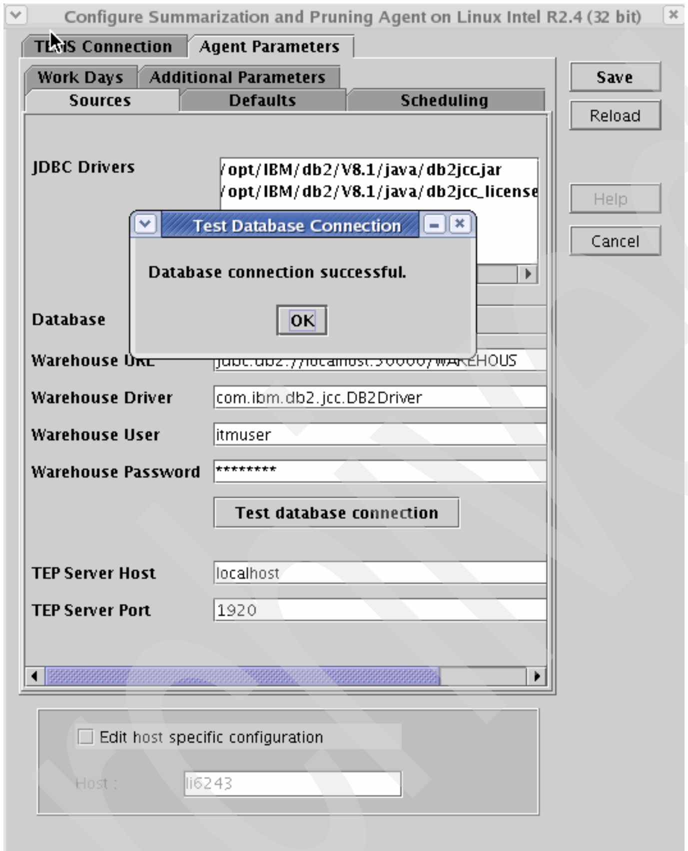


Figure 8-33 Configuration and test data warehouse connection

4. From within Tivoli Enterprise Portal, select **Edit** → **History Configuration**. You should be able to see ISM_MONTIORS in the product drop-down. Select ISM_MONITORS.
5. From the Attribute groups, select whichever group you want to monitor. We selected ISM_HTTP.
6. Make adjustment to collection and pruning as per your requirements.
7. Click **Configure Group**, with all the groups you configured selected.
8. Click **Start Collection**.

This is shown in Figure 8-34.

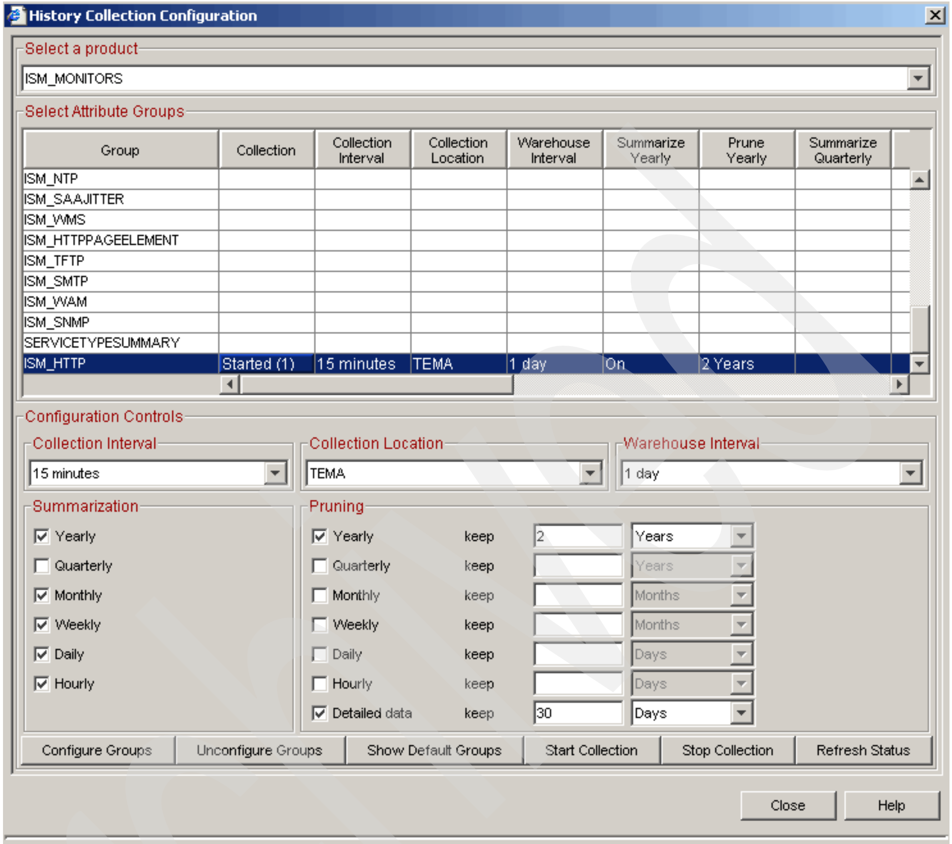
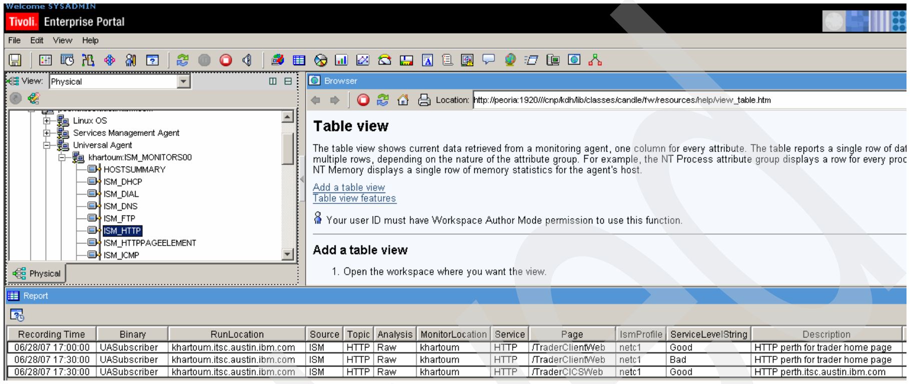


Figure 8-34 ITCAM for Internet Service Monitoring historical data configuration

Once these monitors are configured, you should be able to view this data in the Tivoli Enterprise Portal.

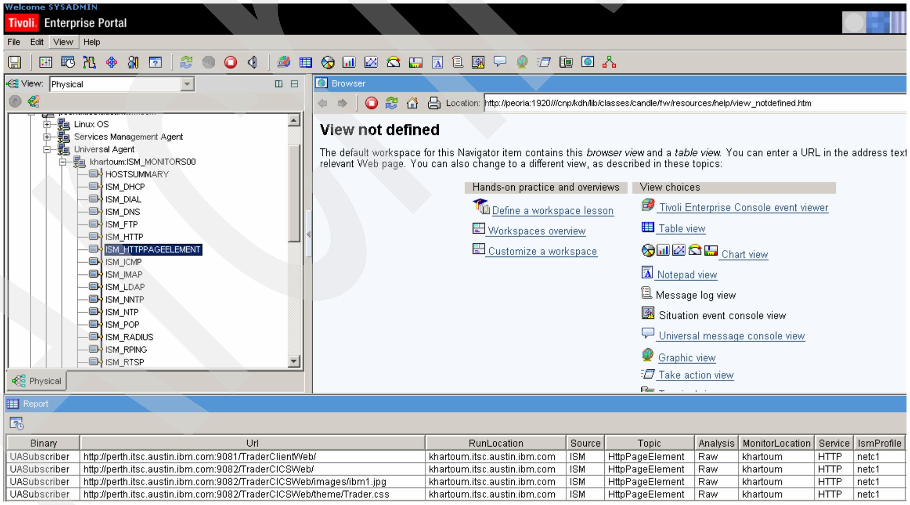
9. Expand the node where Universal Agent is installed, and expand the UA. You will see a ISM_MONITOR00 node under UA. Expand this and select the monitor. You should be able to see various profile elements, which you have defined in ITCAM for Internet Service Monitoring appearing in under Reports, as shown in Figure 8-35.



Recording Time	Binary	RunLocation	Source	Topic	Analysis	MonitorLocation	Service	Page	IsmProfile	ServiceLevelString	Description
06/28/07 17:00:00	UASubscriber	khartoum.itsc.austin.ibm.com	ISM	HTTP	Raw	khartoum	HTTP	/TraderClientWeb	netc1	Good	HTTP perth for trader home page
06/28/07 17:30:00	UASubscriber	khartoum.itsc.austin.ibm.com	ISM	HTTP	Raw	khartoum	HTTP	/TraderClientWeb	netc1	Bad	HTTP perth for trader home page
06/28/07 17:30:00	UASubscriber	khartoum.itsc.austin.ibm.com	ISM	HTTP	Raw	khartoum	HTTP	/TraderCICSWeb	netc1	Good	HTTP perth.itsc.austin.ibm.com

Figure 8-35 ITCAM for Internet Service Monitoring HTTP monitor record

10. If you are viewing the ISM Http monitor, you can also see various elements of a page, as shown in Figure 8-36.



Binary	Url	RunLocation	Source	Topic	Analysis	MonitorLocation	Service	IsmProfile
UASubscriber	http://perth.itsc.austin.ibm.com:9081/TraderClientWeb/	khartoum.itsc.austin.ibm.com	ISM	HttpPageElement	Raw	khartoum	HTTP	netc1
UASubscriber	http://perth.itsc.austin.ibm.com:9082/TraderCICSWeb/	khartoum.itsc.austin.ibm.com	ISM	HttpPageElement	Raw	khartoum	HTTP	netc1
UASubscriber	http://perth.itsc.austin.ibm.com:9082/TraderCICSWeb/images/ibm1.jpg	khartoum.itsc.austin.ibm.com	ISM	HttpPageElement	Raw	khartoum	HTTP	netc1
UASubscriber	http://perth.itsc.austin.ibm.com:9082/TraderCICSWeb/theme/Trader.css	khartoum.itsc.austin.ibm.com	ISM	HttpPageElement	Raw	khartoum	HTTP	netc1

Figure 8-36 ITCAM for Internet Service Monitoring HTTP page element

You can use all the IBM Tivoli Monitoring Tivoli Enterprise Portal features, like creating situations, and so on, to handle this data.

9

ITCAM for SOA

This chapter describes the IBM Tivoli Composite Application Manager for SOA V6.1 concepts, installation, and usage in the following sections:

9.1 ITCAM for SOA overview

This section describes ITCAM for SOA V6.1. The discussion includes:

- 9.1.1, "Product features" on page 450
- 9.1.2, "Product components" on page 452
- 9.1.3, "ITCAM for SOA management resources" on page 457

9.1.1 Product features

ITCAM for SOA manages service-oriented architecture (SOA). It can monitor, manage, and control the Web services layer of IT architectures. ITCAM for SOA provides the following features:

- Service monitoring views in Tivoli Enterprise Portal. ITCAM for SOA workspaces consist of these data collector-based workspaces:
  - Service Management Agent Environment: This node represents the agent monitoring applications for all of the application servers on that system. This node is not initially displayed, and is displayed only when there is traffic in the environment. Sub nodes include the following:
    - Performance Summary: shows the response time information for Web services calls as viewed from the client or the server. A services inventory view is also provided that contains the list of discovered services, along with some basic performance metrics for each operation.
    - Message Summary: shows the message statistics, including the number of messages volume and size of message information.
    - Fault Summary: shows failure analysis for Web services calls including the actual fault code and fault string to assist in problem determination.
  - Service Management Agent: shows the monitoring agent configuration summary, data collectors, monitoring profiles, and filters.
  - Mediation Configuration: shows controlled mediation primitive entries for mediation on Service Component Architecture (SCA) components in WebSphere Process Server or WebSphere Enterprise Service Bus.
  - Message arrival: shows the message arrival rate and events based on the message arrival critical situation. This can be use for looking at the throughput rates.
- ITCAM for SOA navigation view that provides a visualization of data from the Discovery Library Adapters (DLA) discovered service information. It also provides topology views that show relationships between service operations and BPEL business processes for impact analysis.

- Leverages Tivoli Enterprise Portal situations to check thresholds. ITCAM for SOA provides some predefined situations that you need to tailor. The predefined situations concern:
  - Number of messages received by a service within a time window
  - Size of the messages
  - Faults
- Provides basic mediation support with the ability to filter or reject Web services call messages from a particular client or service. It can log request and response messages for analysis.
- Offers heterogeneous platform coverage:
  - Support for IBM WebSphere Application Server, CICS Transaction Server, Microsoft .NET, JBoss, BEA WebLogic, and other SOA clients and servers.
  - Enterprise Service Bus platforms: WebSphere Enterprise Service Bus, WebSphere Process Server, WebSphere Message Broker, and WebSphere DataPower SOA device.
  - Apache Axis 1.2 Data Collector support: Axis is an open source, XML-based Web service framework. This gives customers the flexibility to use an open source Web services framework and still maintain end-to-end services monitoring.
- Displays a list of services and operations monitored in the environment.
- Leverages Tivoli Enterprise Portal workflow and policy editor for threshold-triggered action sequences.
- Offers the ability to include services-layer views in Tivoli Enterprise Portal.

The context-rich views and inter-workspace linkages in Tivoli Enterprise Portal enable users to drill down to IT resources to identify Web service bottlenecks and failures. By providing built-in and extensible alerts, situations, and workflows, users can create powerful automated mediation scenarios using the Tivoli Enterprise Portal.

The service metrics, alerts, and automation workflows provided by ITCAM for SOA and other Tivoli products can be displayed in Tivoli Enterprise Portal with the cross-workspace linkages to provide a rich and multilayered source of information that can help to reduce the time and skills required for problem root-cause analysis and resolution.

ITCAM for SOA includes the Web Services Navigator, a plug-in to IBM Rational Application Development and other Eclipse-based tools. It provides a deep understanding of the service flow, patterns, and relationships for developers and architects. The Web Services Navigator leverages the existing ITCAM for SOA agent infrastructure. The data may originate from the IBM Tivoli Monitoring V6.1 Tivoli Data Warehouse or from the ITCAM for SOA log files using the Log Assembler tool.

In Version 6.1, ITCAM for SOA contains a new component for mediation service management based on Service Component Architecture (SCA). It enables you to modify mediation service settings on the fly. Mediation is a facility that sits between the Web services requester and the Web services provider that allows manipulation of Web services messages, includes format translation, routing, logging, and filtering.

9.1.2 Product components

ITCAM for SOA manages services, including Web services, SCA components, and services flowing though Datapower and Message broker. Web services can be viewed as a remote processing facility that is defined through the use of Web Services Description Language (WSDL). Usual access uses SOAP over HTTP. ITCAM for SOA utilizes a data collector for each of the supported environments:

- For J2EE Web services, ITCAM for SOA collects data as a JAX-RPC handler.
- For .NET-based Web services, ITCAM for SOA installs a service extension.
- For SCA, ITCAM for SOA acts as an SCA handler.
- For WebSphere Message Broker, ITCAM for SOA uses a Message broker exit.

Internally, Web services are implemented using the Java API for XML-based Remote Procedure Call (JAX-RPC). ITCAM for SOA installs itself as the JAX-RPC handler to capture and manage Web services calls.

ITCAM for SOA consists of these logical components:

- Web services data collector that collects the services data appropriate to the environment it is collecting from. It can intercept Web services calls to collect statistical information and write to a log file.
- Tivoli Enterprise Monitoring Agent that collects information from all of the data collectors on that machine and forwards them to the Tivoli Enterprise Monitoring Server. We discuss the data collectors and Tivoli Enterprise Monitoring Agent in “Monitoring agent data collector” on page 453.
- An Eclipsed-based viewer that processes log files that are generated by the Web services data collector. It generates visual representations of various characteristics of monitored Web services. See “IBM Web Services Navigator” on page 454.

- Mediation primitives SCA tools that allow operational control of mediation primitives within WebSphere Enterprise Service Bus or the WebSphere Process Server. See "Managing SCA mediation" on page 456.

Monitoring agent data collector

ITCAM for SOA works with several application server environments:

- IBM WebSphere Application Server V5.1.0.5 with PQ89492, V6.0, and V6.1
- IBM WebSphere Business Integration V5.1.1.1
- IBM WebSphere Process Server V6.0.1
- IBM WebSphere Enterprise Service Bus V6.0.1
- IBM CICS Transaction Server V3.1 and later
- BEA WebLogic Server V8.1.4
- Microsoft .NET V1.1 with Service Pack 1 and V2.0
- JBoss V4.03
- WebSphere Community Edition V1.0 and its service packs
- SAP NetWeaver V6.40 with Service Pack 9 or later service packs
- IBM WebSphere DataPower SOA Appliance Firmware V3.5.0.5 or later
- WebSphere Message Broker V5.3 or V6
- Axis 1.2 support

Figure 9-1 shows the ITCAM for SOA data collection conceptual architecture.

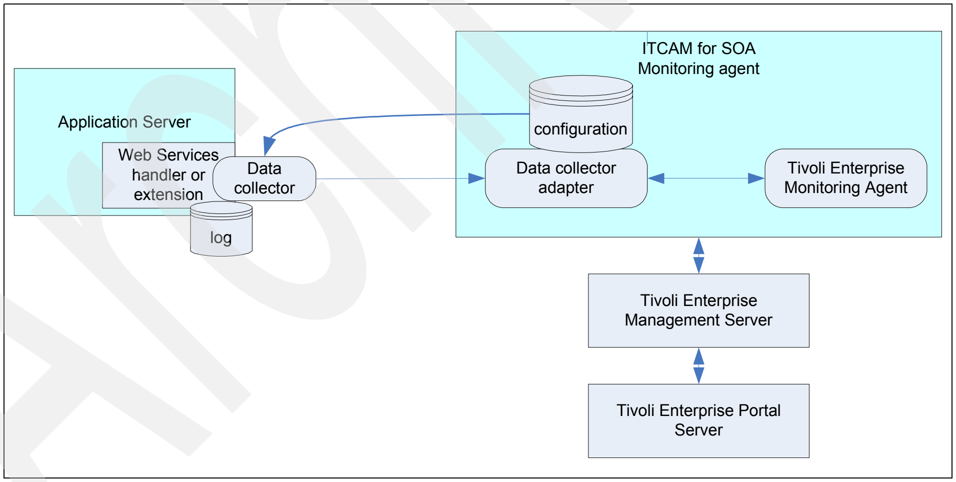


Figure 9-1 ITCAM for SOA structure

The monitoring agent data collector is implemented as a JAX-RPC handler or service extension that is installed into the application servers that are hosting the monitored Web services. The handler is given control when either of the following events occurs:

- A client application invokes a Web service, which is referred to as a *client-side interception*.
- The Web service request is received by the hosting application server, which is referred to as a *server-side interception*.

The monitoring agent records and collects monitored information into one or more local log files. The information is then transferred to the Tivoli Enterprise Monitoring Server and can be archived into a historical database for later retrieval with IBM Web Services Navigator.

ITCAM for SOA V6.1 focuses on the SOAP engine of IBM WebSphere Application Server, WebSphere Service Integration Bus, Microsoft .NET Framework, and BEA WebLogic.

The Web services data collector supports both J2EE application client and server container environments because JAX-RPC handlers are supported only by these environments. The Web services must be compliant with JSR-109 specifications.

To ensure proper operation of the JAX-RPC handler, verify that the client applications are written according to the conventions at the following location:

http://www.jcp.org/aboutJava/communityprocess/final/jsr109/

IBM Web Services Navigator

IBM Web Services Navigator is an Eclipsed-based tool used to visualize Web services in an SOA environment. It provides a graphical display of:

- Web services transaction flows
- Service topology
- Flow patterns

Web Services Navigator concepts can be illustrated as shown in Figure 9-2.

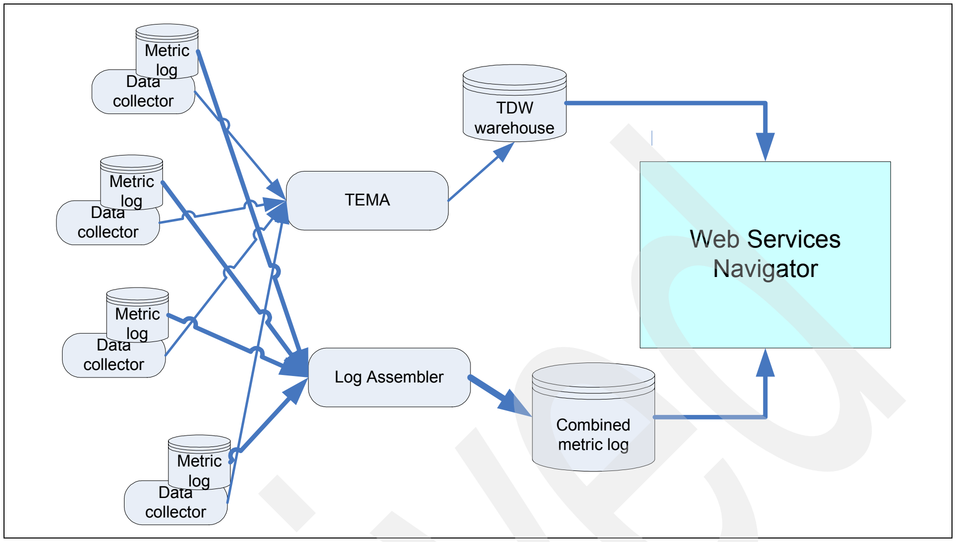


Figure 9-2 Web Services Navigator

The Web Services Navigator is a log-browsing tool intended for offline analysis of SOA Web services. The Web Services Navigator provides four primary views:

- Statistic tables:
  - Message statistics

    Per-message statistics including requestor, provider, send/receive time, and message size
  - Invocation statistics

    Response time, network delay, message size, and more for each Web service invocation
  - Transaction statistics

    Statistics for aggregated transactions, including elapsed time, number of faults, number of machines this transaction involves, and number of invocations comprising this transaction
  - Pattern invocation statistics

    Statistics for discovered patterns, including operation names, number of occurrences, response times, and message sizes

> **Note:** To see the message content from the ITCAM for SOA metric log:
>
> 1. Set a monitor control higher than *none* for any or all of the Web services being monitored.
> 2. Include the subsequent xxxx.content.log when running Log Assembler.

- Service topology view

  A graphical representation of the monitored Web services that displays aggregated information and information about the relationships between Web services.

- Transaction flows view

  The transaction flows view displays Universal Markup Language (UML) style sequence diagrams. The transaction flow shows a chronological view of each transaction, the flow between the various Web services over time, and the topology and statistics for each transaction. You can zoom in on the view to see the details of individual transactions.

- Flow pattern view

  The flow pattern view is a visual representation of the aggregated pattern of transactions represented in the log file. The view also represents each pattern as a distinct sequence of Web service calls and displays the frequency of each pattern.

Managing SCA mediation

The WebSphere Process Server and WebSphere Enterprise Service Bus introduce a new way to model services in an SOA, called the Service Component Architecture. SCA is designed to separate business logic from its implementation so that you can focus on assembling an integrated application without knowing implementation details.

There is a special type of SCA component called a mediation. In a service-oriented architecture (SOA), where services are loosely coupled rather than being connected directly to each other, mediations can be inserted between the services, where they can intercept and process messages that are being passed between the services. Mediations can process these messages and take appropriate actions, such as reroute, log, or transform a message, or create a notification or an event.

ITCAM for SOA provides the ability to dynamically enable and disable the deployed mediation functions. This facility is available for applications in the WebSphere Enterprise Service Bus or WebSphere Process Server runtime environment. The invocation is provided in a new workspace in the Tivoli Enterprise Portal called the Mediation Configuration workspace. The actions are:

- ConfigureMediation_610
- DeletePrimitiveProperty_610

9.1.3 ITCAM for SOA management resources

The management resources for ITCAM for SOA that we discuss here are:

- “Workspaces” on page 457
- “Attributes” on page 458
- “Situations” on page 459
- “Actions” on page 460

Workspaces

ITCAM for SOA delivers a set of predefined workspaces, which you can select from the Tivoli Enterprise Portal navigator view. Each workspace has its own set of views that display Web services data and metrics in various levels of detail. Figure 9-3 shows the workspace navigator area.

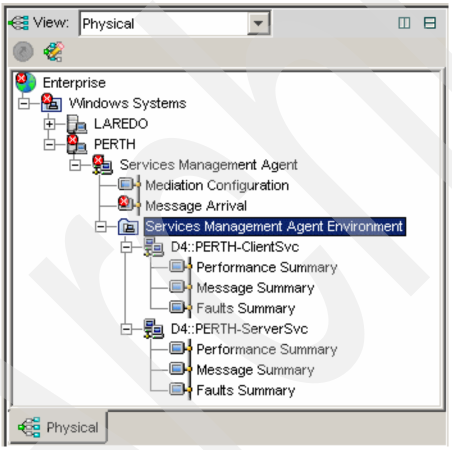


Figure 9-3 Navigator in the workspace

These workspaces are available:

- The Service Management Agent workspace displays the current configuration details for the monitoring agent data collectors that are configured in different application server instances. This workspace contains the following views:
  - Data Collector Global Configuration
  - Data Collector Monitor Control Configuration
  - Data Collector Filter Control Configuration

- The Mediation Configuration workspace that contains the SCA mediation configuration. This workspace can be used to launch the SCA mediation actions.
- The Message Arrival Summary workspace provides a summary of the number of messages that arrive at the data collector for each combination of service name, operation name, and remote IP address that has been configured as a situation. This workspace contains the following views:
  - Message Arrival Details
  - Message Arrival by Service
  - Message Arrival by Operation
- The Services Management Agent Environment workspace represents the agent monitoring applications for all of the application servers on that system. The Services Management Agent Environment workspace provides a set of views that summarize the performance, message activity, and fault occurrences associated with the Web services traffic through this monitoring agent. This workspace contains the following views:
  - Average Response Time by Operation
  - Number of Messages by Operation
  - Average Message Size by Operation
- The Performance Summary workspace provides the inventory of currently active and monitored services, as well as the response time of the services. This workspace contains the following views:
  - Average Response Time by Operation
  - Services Inventory
- The Messages Summary workspace provides details about the number and size of messages received for services and service/operation combinations. This workspace contains the following views:
  - Number of Messages by Service - Operation - Type
  - Average size of Messages by Service - Operation - Type
- The Faults Summary workspace provides a general faults summary. This workspace contains the following views:
  - Faults Summary by Operation
  - Fault Details

Attributes

Attributes are measurements that are collected by the IBM Tivoli Monitoring V6.1 family of products. ITCAM for SOA stores specific measurements or attributes relevant to its needs and function.

Refer to *IBM Tivoli Composite Application Manager for SOA Installation and User's Guide*, GC32-9492. An appendix lists the attribute groups or tables provided with the ITCAM for SOA product. The tables that are available for long-term historical data collection are indicated in the description of the table, and show the historical reference information that identifies each attribute within each table. Each table identifies the column name, attribute name, and a description of the data provided. These attributes are used in ITCAM for SOA situations.

Situations

In the IBM Tivoli Monitoring V6.1 context, a situation is a condition in which a set of attributes are tested against a threshold within any filtering rules (if necessary). The situation evaluates these conditions at predefined intervals and invokes the necessary automated responses and notification methods when needed.

ITCAM for SOA provides a set of predefined situations that are designed to help monitor critical activities and serve as templates for creating customized situations for your own use. Predefined situations are started automatically when the product is installed. After they have been configured, the situation alerts provided with ITCAM for SOA trigger event notification.

Table 9-1 lists the predefined situations that are provided with ITCAM for SOA.

Table 9-1 ITCAM for SOA situations

Situation name	Description
Fault	Monitors the messages in the Web services flow to determine whether a Web services fault has occurred.
Message Arrival Critical	Alerts you to excessive amounts of Web services traffic. (The number of messages received from one or more remote clients exceeds a threshold that you specify.)
Message Arrival Clearing	Clears a previously triggered message arrival critical situation. This situation can also be used to alert that a message has fallen below a specified threshold (lack of activity alert).
Message Size	Monitors the length, in bytes, of each message during the Web services flow. If the length of the message is more than the threshold value, this situation is triggered.
Response Time Critical	Monitors elapsed round-trip response time, in milliseconds, for the completion of a Web services request.
Response Time Warning	Monitors elapsed round-trip response time, in milliseconds, for the completion of a Web services request.

> **Note:** Each implementation provides its own set of thresholds, because the product's default will *not* fit your environment. You must tune your monitoring thresholds.
>
> The default situations are based on individual service calls stored in the Services_Metric table. Analyzing summarized information in the Services_Inventory table can reduce some of the monitoring processor usage.

Actions

In ITCAM for SOA V6.1, there are different sets of actions for different contexts. Although the available action types are the same, there are now four actions available for each type. The existing 6.0 actions are still kept for compatibility and coexistence with ITCAM for SOA V6.0 agents and data collectors. New generic ITCAM for SOA actions are labelled with suffix _610. These contain new parameters and functionality to be used with the new ITCAM for SOA V6.1 data collectors. When invoked from the workspaces with Service Metric or Service Inventory data, the data is inserted automatically as the arguments for the actions. The action name is prefixed with SM_* or SI_*, respectively.

The available take action methods for ITCAM for SOA are:

AddFltrCntrl Creates new filter control settings to reject messages.

AddMntrCntrl Creates new monitor control settings. These monitor settings affect the data logging for use with IBM Web Service Navigator.

DelFltrCntrl Deletes existing filter control settings.

DelMntrCntrl Deletes existing monitor control settings.

DisableDC Disables data collection and the ability to reject messages.

EnableDC Enables data collection and the ability to reject messages.

updateLogging Defines the level of logging information.

UpdMntrCntrl Updates existing message logging levels for monitor control.

updateTracing Enables or disables tracing.

These actions can be invoked manually or triggered by a situation. Actions can also be triggered by workflows, which are predefined automations that you can build on the IBM Tivoli Monitoring platform.

9.2 ITCAM for SOA installation

This is the overall implementation procedure for ITCAM for SOA:

1. Plan for the configuration. It is important to have a good understanding of the managed environment and the capability of the product. We discuss planning considerations in 9.2.1, "Planning the implementation" on page 461.
2. ITCAM for SOA is installed and operates within the management infrastructure of the Tivoli Enterprise Monitoring Server services platform. The installation of IBM Tivoli Monitoring V6.1 must be performed before any other ITCAM for SOA component. For Tivoli Monitoring installation information, see *IBM Tivoli Monitoring Installation and Setup Guide*, GC32-9407.
3. Install the application support component for ITCAM for SOA on your Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, and Tivoli Enterprise Portal systems (9.2.2, "Installing ITCAM for SOA application support" on page 464).
4. Install and configure the monitoring agents of ITCAM for SOA (see 9.2.3, "Installing the ITCAM for SOA monitoring agent" on page 470, and 9.2.4, "Enabling the monitoring agent" on page 473).
5. Metrics collected by the ITCAM for SOA data collector (DC) can be stored in the Tivoli Data Warehouse. The Data Warehouse Proxy must be configured on the Tivoli Enterprise Monitoring Server in order to enable historical data collection for ITCAM for SOA. We discuss this in 9.2.5, "Configuring the warehouse proxy" on page 475, and 9.2.6, "Installing IBM Web Services Navigator" on page 479.

> **Note:** This chapter gives an overview of the ITCAM for SOA main components and installation process. Refer to *IBM Tivoli Composite Application Manager for SOA Installation and User's Guide*, GC32-9492, for more detailed information.

9.2.1 Planning the implementation

Planning considerations include:

- "IBM Tivoli Monitoring services" on page 462
- "ITCAM for SOA application support" on page 462
- "ITCAM for SOA monitoring agent" on page 462
- "IBM Web Service Navigator" on page 463

IBM Tivoli Monitoring services

IBM Tivoli Monitoring services, which include Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server, must be already installed and configured in your environment. Refer to *IBM Tivoli Monitoring Installation and Setup Guide*, GC32-9407, to understand the available options for deploying the various components on one or more systems in your enterprise.

> **Note:** This chapter does not list the steps to install, set up, or implement Tivoli Monitoring Services. See *Getting Started with IBM Tivoli Monitoring 6.1 on Distributed Environments*, SG24-7143, to information about those topics.

Familiarize yourself with the Tivoli Enterprise Monitoring Server management infrastructure installed in your enterprise environment, including its various facilities to manage the system, such as workflows and situations. Use Tivoli Enterprise Portal Server and Tivoli Enterprise Portal to understand workspaces and views for operators and their implication in the overall monitoring.

ITCAM for SOA application support

To view the ITCAM for SOA monitored agents through Tivoli Enterprise Portal, you must install application support for the agent on the Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, and Tivoli Enterprise Portal. The Tivoli Enterprise Monitoring Server and Tivoli Enterprise Portal Server services are restarted during the ITCAM for SOA application support installation process.

The application support files include:

- Data structure definition for Tivoli Enterprise Monitoring Server attributes and attribute groups (tables). ITCAM for SOA contains two tables: Services_Metrics and Services_Inventory.
- Situation definitions that allow proactive monitoring to be performed in the IBM Tivoli Monitoring environment.
- Presentation information to be installed in the Tivoli Enterprise Portal Server, including help resources and workspace definitions.
- Additional resources such as sample workflow and historical collection information.

ITCAM for SOA monitoring agent

In a typical distributed environment, the Tivoli Enterprise Monitoring Server is installed on one system, the Tivoli Enterprise Portal Server is installed on another system, and the Tivoli Enterprise Monitoring Agent is installed on additional multiple application server systems where Web services traffic is to be monitored.

You must install ITCAM for SOA agents on each system that has one or more application server environments that run Web services, such as IBM WebSphere Application Server, Microsoft .NET, and BEA WebLogic. Review *IBM Tivoli Composite Application Manager for SOA Release Notes*, GI11-4096, which contains the most current information.

When you install ITCAM for SOA on the application server, select to install the agent support component. This includes the data collector component that intercepts request and response messages for the Web services that you want to monitor. You must configure the appropriate data collector after installing the agent component.

The IBM Tivoli Monitoring environment requires installation and configuration to be performed on both distributed and z/OS-managed systems where some of the platform components are installed and run.

For more information about installing ITCAM for SOA on a supported z/OS operating system, refer to *Configuring IBM Tivoli Composite Application Manager for SOA z/OS*, SN32-9493.

When the ITCAM for SOA monitoring agent is installed and the application server is enabled, the data collector is the monitoring component of the ITCAM for SOA. It is implemented as a SOAP message handler and used to monitor Web services that flow across an interception point.

The interception point is a JAX-RPC handler in IBM WebSphere Application Server and BEA WebLogic server environments and service extensions in the Microsoft .NET environment.

IBM Web Service Navigator

IBM Web Services Navigator has these features in the Eclipse environment:

- An import wizard to import Web services log files or retrieve data from ITCAM for SOA historical data into the IBM Web Services Navigator
- Web Services Profiling perspectives with a set of views of Web services transactions
- A separate Log Assembler tool that can be used to manually combine locally stored metric and content log files from the multiple application servers in your Web services environment into a single log file that can be imported to the IBM Web Services Navigator for viewing

> **Note:** Another component for managing SCA mediation is not tested in the project.

9.2.2 Installing ITCAM for SOA application support

This section provides installation procedures for application support for the ITCAM for SOA monitoring agent on the Tivoli Enterprise Monitoring Server. The installation for ITCAM for SOA is supplied with platform-specific CD-ROMs. The wizard copies the CD-ROM files onto the disk. Perform the following steps:

1. Navigate to the platform-specific directory on the ITCAM for SOA product CD. The installer is provided to support the IBM Tivoli Monitoring V6.1 installation. Select the appropriate operating system platform. For Windows, we invoke \WINDOWS\setup.exe.
2. The wizard can discover the IBM Tivoli Monitoring installation. It checks for the appropriate GSKit and Java environment. We choose the installation drive C:, as shown in Figure 9-4.

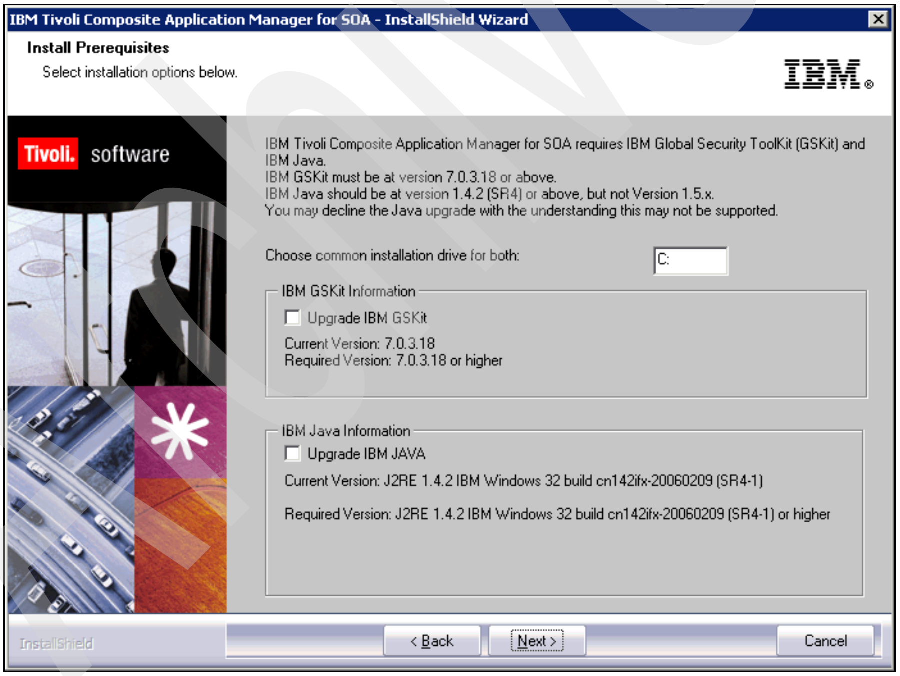


Figure 9-4 Installation drive

3. Because Tivoli Enterprise Monitoring Server, Tivoli Enterprise Portal Server, and Tivoli Enterprise Portal are installed on one system, all of the components are selected to install to the local machine. See Figure 9-5.

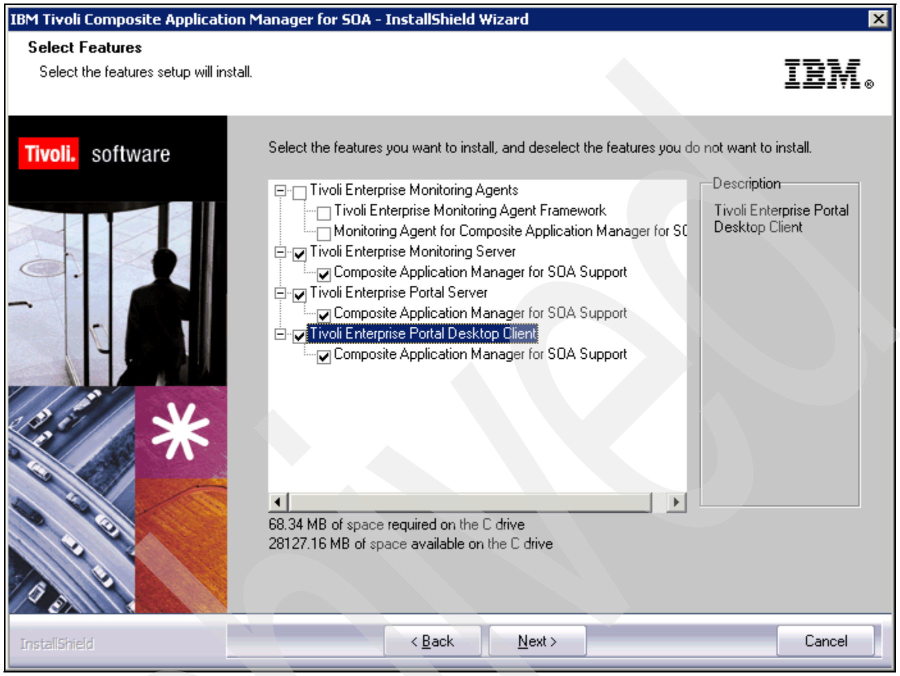


Figure 9-5 Installation features

> **Note:** If you select any of the application support components that are already installed, a warning window opens. You have the choice of overwriting the existing installation files or deselecting the desired component to avoid overwriting the files.

4. ITCAM for SOA also supports packaging of the agent for remote deployment. See Figure 9-6.

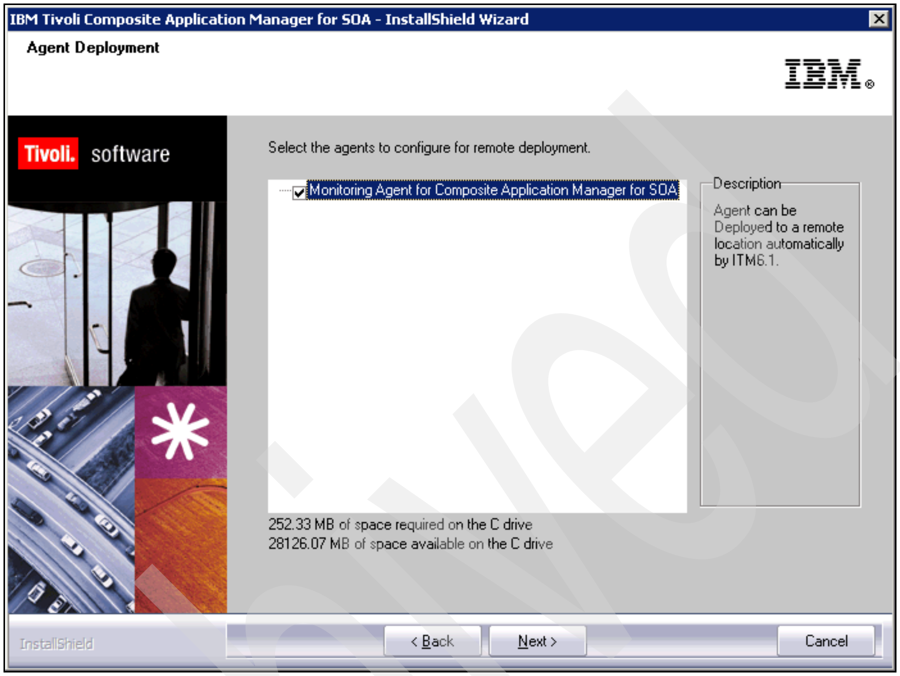


Figure 9-6 Application support component

5. The installation summary provides a brief list of actions it will perform. Click **Next** (Figure 9-7).

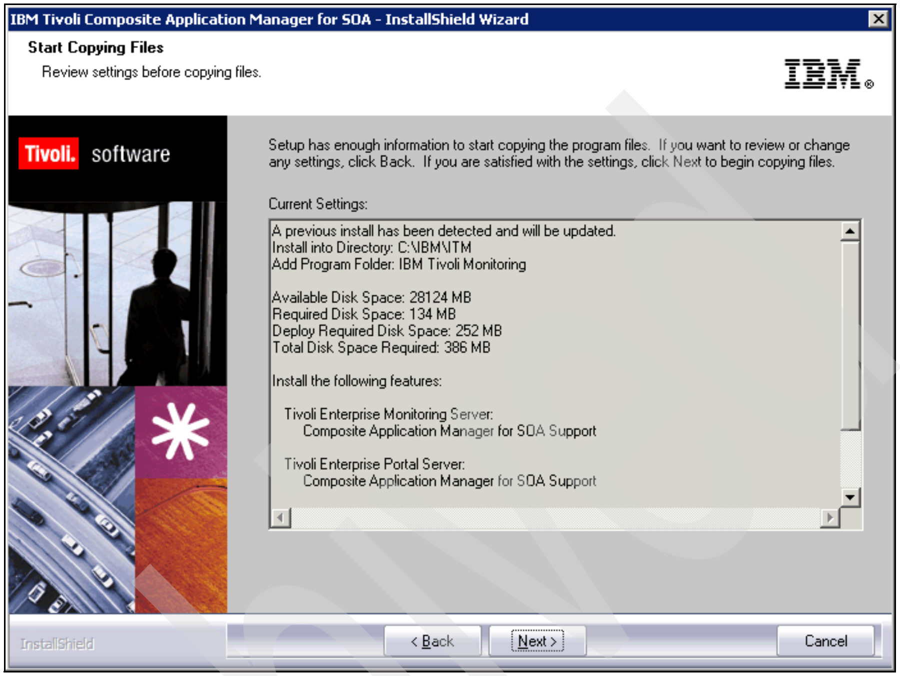


Figure 9-7 Installation action list

6. Select which components you want to configure, as shown in Figure 9-8.

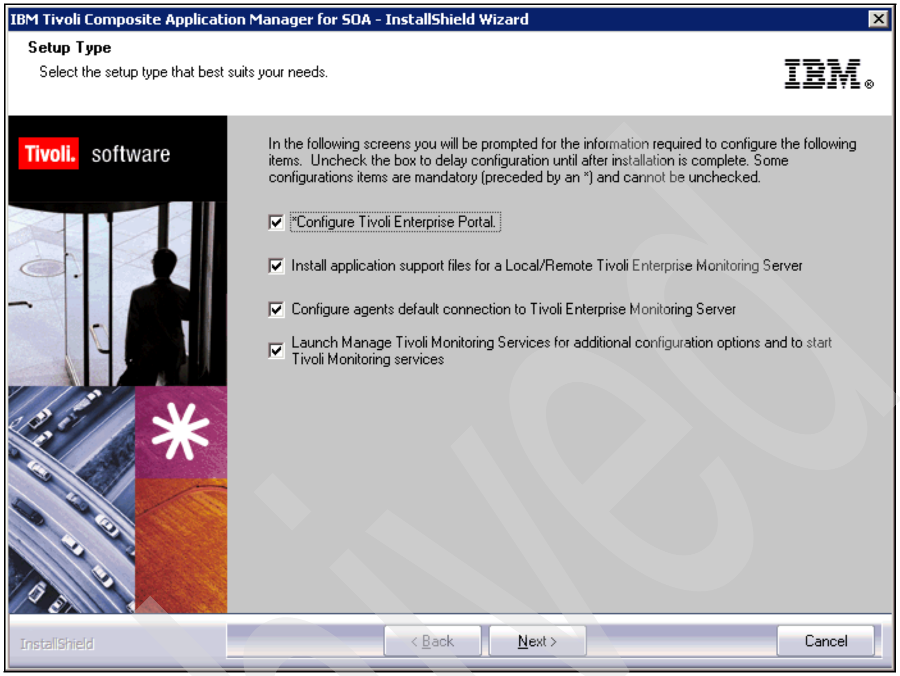


Figure 9-8 Configuration options

7. Enter the host name of the machine where Tivoli Enterprise Portal Server is installed. In our case, we enter `Lima`.
8. Enter the information for the Tivoli Enterprise Monitoring Server management hub or remote server, as shown in Figure 9-9.

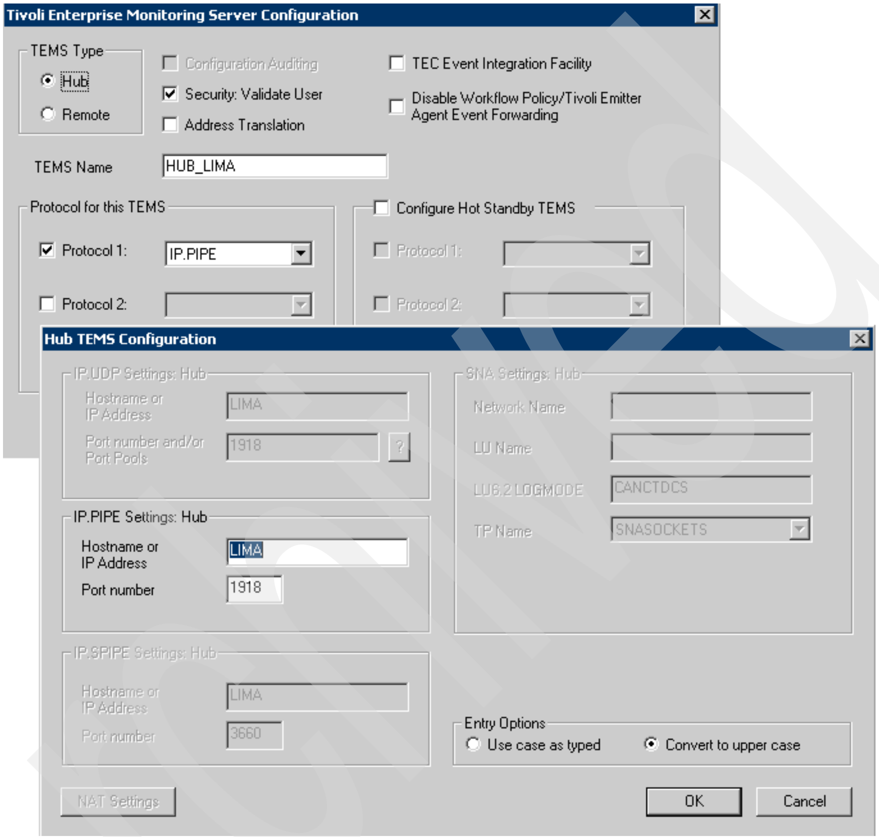


Figure 9-9 Tivoli Enterprise Monitoring Agent hub configuration

9. Select whether the Tivoli Enterprise Monitoring Server is located on the same system. In our installation, that is the case (Figure 9-10).

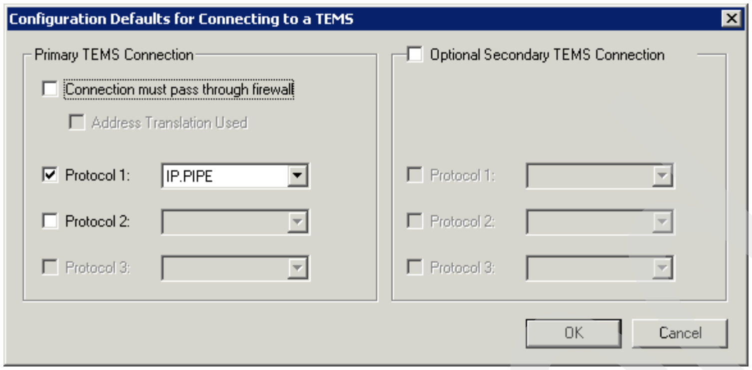


Figure 9-10 Tivoli Enterprise Monitoring Agent configuration part 1

10.We enter the default configuration for the Tivoli Enterprise Monitoring Server connection, as shown in Figure 9-10 and Figure 9-11.

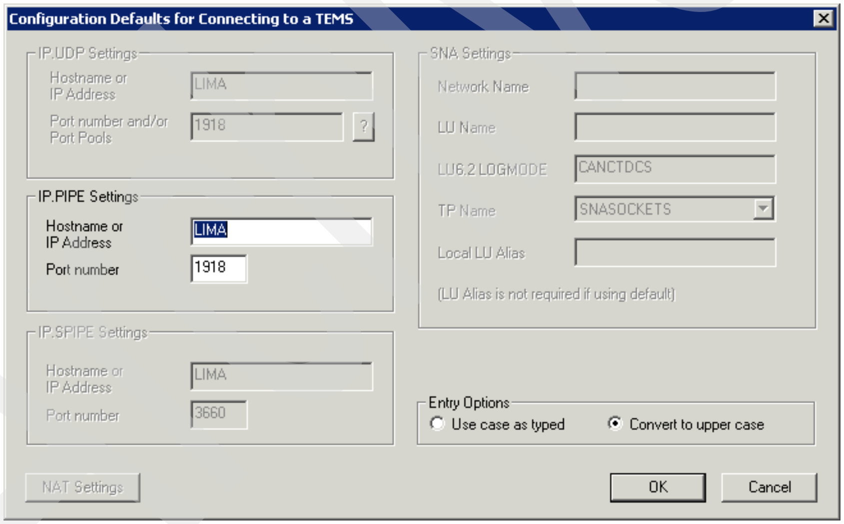


Figure 9-11 Tivoli Enterprise Monitoring Agent configuration part 2

9.2.3 Installing the ITCAM for SOA monitoring agent

The monitoring agent installation differs by environment. This section gives an overview of the monitoring agent installation. The ITCAM for SOA monitoring agent installation includes the data collector, which is installed into each application server environment where Web services traffic is to be monitored. Perform the following steps:

1. To launch the ITCAM for SOA monitoring agent Welcome window:
   - For a Windows installation, navigate to the /WINDOWS directory and select **setup.exe** from the ITCAM for SOA product CD.
   - For a UNIX installation, open a command session and navigate to the root directory of the ITCAM for SOA CD. Issue the **./install.sh** command.
2. We demonstrate the installation on a Linux machine in Example 9-1.

Example 9-1 Linux installation

```
# ./install.sh
INSTALL
Enter the name of the IBM Tivoli Monitoring directory
[ default = /opt/IBM/ITM ]:/opt/IBM/ITM
Select one of the following:
1) Install products to the local host.
2) Install products to depot for remote deployment (requires TEMS).
3) Exit install.
Please enter a valid number:  1
Initializing ...
Software Licensing Agreement
1. Czech
2. English
   .  .  .
Please enter the number that corresponds to the language you prefer. 2
Software Licensing Agreement
   .  .  .
Enter "1" to accept the agreement, "2" to decline it or
"99" to go back to the previous screen. 1
Product packages are available in /code/SOA61-linux/unix

The following products are currently installed in "/opt/IBM/ITM:"
   .  .  .
The following prerequisites should be installed now:
 IBM Tivoli Monitoring Shared Libraries V610R437 @ Linux Intel R2.4 (32 bit)
 Tivoli Enterprise Services User Interface V610R291 @ Linux Intel R2.4 (32 bit)
Do you want to install these prerequisites [ y or n; "y" is default ]? y
 ... installing package "axli6243"; please wait.
 => installed package "axli6243."
 ... installing package "uili6243"; please wait.
 => installed package "uili6243."
Product packages are available for the following operating systems and
component support categories:
 1) Linux AMD64 R2.6 (64 bit)
 2) Linux Intel R2.4 (32 bit)
…
```

```
[ number "2" or "Linux Intel R2.4 (32 bit)" is default ]:2
You selected number "2" or "Linux Intel R2.4 (32 bit)"
Is the operating system or component support correct [ y or n; "y" is default
]? y

The following products are available for installation:
 1) ITCAM for SOA  V06.10.00.00
 2) Tivoli Enterprise Services User Interface  V06.10.04.00
 3) all of the above
Type the numbers for the products you want to install, or type "q" to quit
selection.If you enter more than one number, separate the numbers by a comma or
a space.
Type your selections here: 1
The following products will be installed:
  ITCAM for SOA  V06.10.00.00
Are your selections correct [ y or n; "y" is default ]?y
 ... installing "ITCAM for SOA  V06.10.00.00 for Linux Intel R2.4 (32 bit)";
please wait.
 => installed "ITCAM for SOA  V06.10.00.00 for Linux Intel R2.4 (32 bit)."
 ... Initializing database for ITCAM for SOA  V06.10.00.00 for Linux Intel R2.4
(32 bit).

Do you want to install additional products or product support packages [ y or
n; "n" is default ]?n
 ... postprocessing; please wait.
 ... finished postprocessing.
 Installation step complete.

As a reminder, you should install product support on each of your TEM servers
for any agents you have just installed. This is done via the "[ITM
home]/bin/itmcmd support" command on your TEM servers.
You may now configure any locally installed IBM Tivoli Monitoring product via
the "/opt/IBM/ITM/bin/itmcmd config" command.
```

3. After finishing the installation, we configure the agent by running the **./itmcmd.sh config -A d4** command from the /bin directory of your installation, where d4 is the 2-character product code for ITCAM for SOA. For further steps, see the script in Example 9-2.

Example 9-2 Configuring the agent

```
# ./itmcmd config -A d4
Agent configuration started...

Will this agent connect to a TEMS? [YES or NO] (Default is: YES):YES
TEMS Host Name (Default is: srv178): lima

Network Protocol [ip, sna, ip.pipe or ip.spipe] (Default is: ip.pipe):
```

```
     Now choose the next protocol from one of these:
     - ip
     - sna
     - ip.spipe
     - none
Network Protocol 2 (Default is: none):none
IP.PIPE Port Number (Default is: 1918):1918
Enter name of KDC_PARTITION (Default is: null):null

Configure connection for a secondary TEMS? [YES or NO] (Default is:
NO):NO
Enter Optional Primary Network Name or "none" (Default is: none):none
Agent configuration completed...
```

> **Note:** You do not have to start the ITCAM for SOA monitoring agent until the monitoring agent is enabled for the application server.

9.2.4 Enabling the monitoring agent

This section describes how to enable the ITCAM for SOA monitoring agent data collector handler in the appropriate application server systems, depending on the application server environment you have installed on this application server.

After installing the ITCAM for SOA monitoring agent on the application server, the data collector directory structure is created in the Tivoli Enterprise Monitoring Agent base directory as follows:

For Windows	%TEMA_HOME%\TMAITM6\KD4
For UNIX	$TEMA_HOME/<OS_INTERP>/d4/KD4
For z/OS	<TEMA_HOME>

These directories contain all the files required to run the data collectors.

> **Note:** An install script called KD4configDC configures the data collector for all application server platforms. However, each platform requires its own additional parameters and steps that must be performed to enable the monitoring of Web services.

Depending on your operating system platform, issue **`KD4configDC.sh`** or **`KD4configDC.bat`**. The arguments for this command vary depending on the application server environment. Refer to *IBM Tivoli Composite Application Manager for SOA Installation and User’s Guide*, GC32-9492, for more detailed information about the **`KD4configDC`** options.

We only enable the data collection in our WebSphere Application Server environment. For IBM WebSphere Application Server, run this command at a command prompt:

```
KD4configDC -enable -env 1 <WAS_HOME>
```

Note: If you set the WAS_HOME environment variable, the configuration program can be invoked without this argument. The WAS_HOME environment variable is typically set using the setupCmdLine program from WebSphere.

This command enables the WebSphere Application Server to use the KD4 data collector as a JAX-RPC handler. The kd4dcagent.jar file is installed in the %WAS_HOME%/lib/ext directory.

After the ITCAM for SOA monitoring agent data collector has been configured, stop and restart IBM WebSphere Application Server.

9.2.5 Configuring the warehouse proxy

You can configure the warehouse proxy to start retrieving data in the Tivoli Data Warehouse from the Manage Tivoli Enterprise Monitoring Services windows. Perform the following steps:

1. Right-click **Warehouse Proxy** and select **Reconfigure** (Figure 9-12).

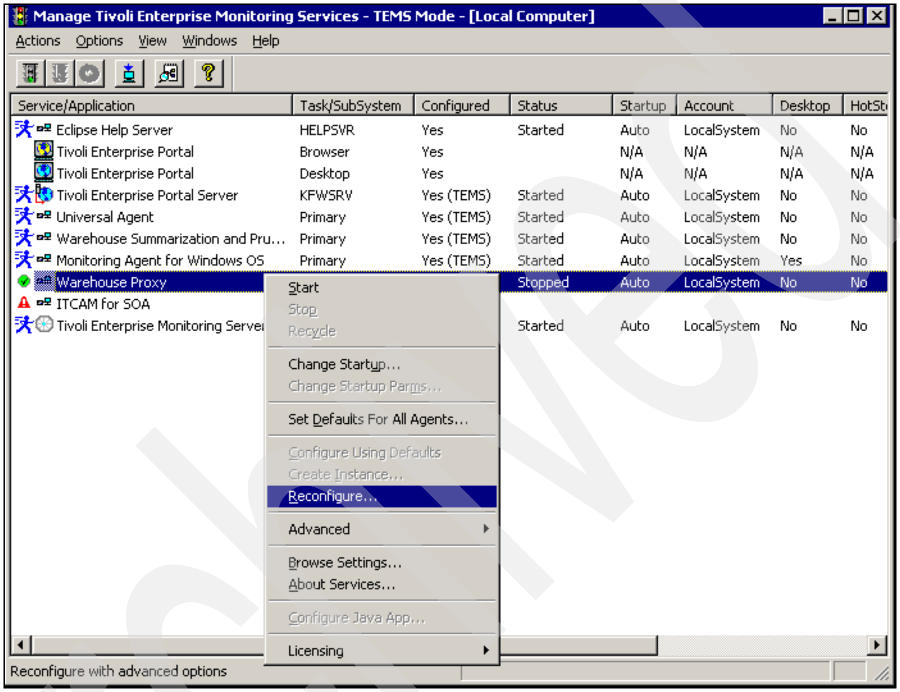


Figure 9-12 Manage Tivoli Enterprise Monitoring Services

2. In the first part of the configuration, specify Tivoli Enterprise Monitoring Server settings (Figure 9-13).

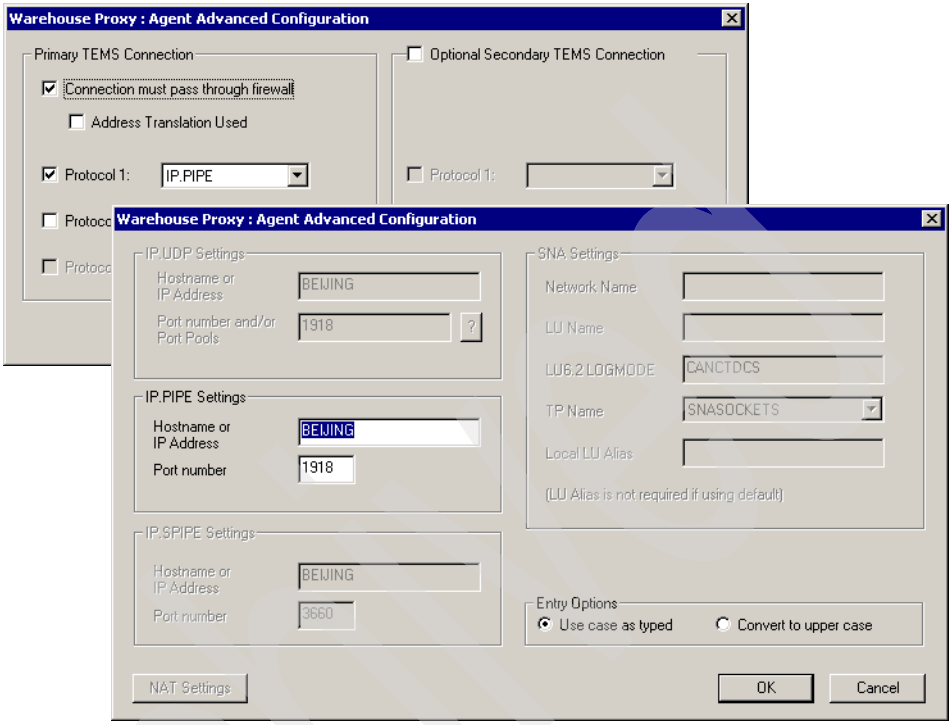


Figure 9-13 Warehouse Proxy configuration: Tivoli Enterprise Monitoring Server settings

3. Provide the database settings (Figure 9-14). We use the WAREHOUS database in our DB2 UDB database.

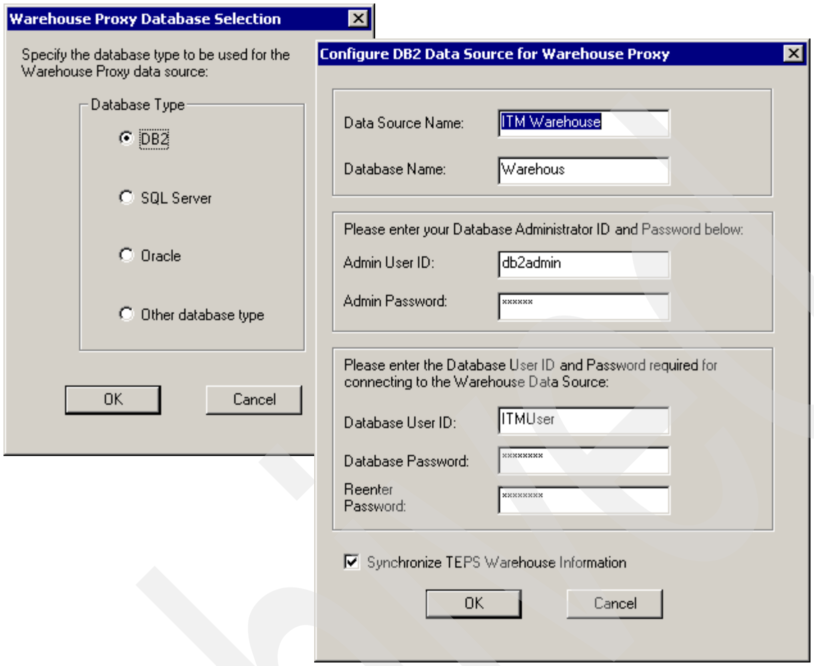


Figure 9-14 Warehouse Proxy configuration: Database settings

Now the warehouse proxy has been configured and started. You can configure data collection for the IBM Tivoli Monitoring Agents. To enable ITCAM for SOA collection, click the historical data configuration button in Tivoli Enterprise Portal. In the window shown in Figure 9-15, specify that the ITCAM for SOA information is to be collected.

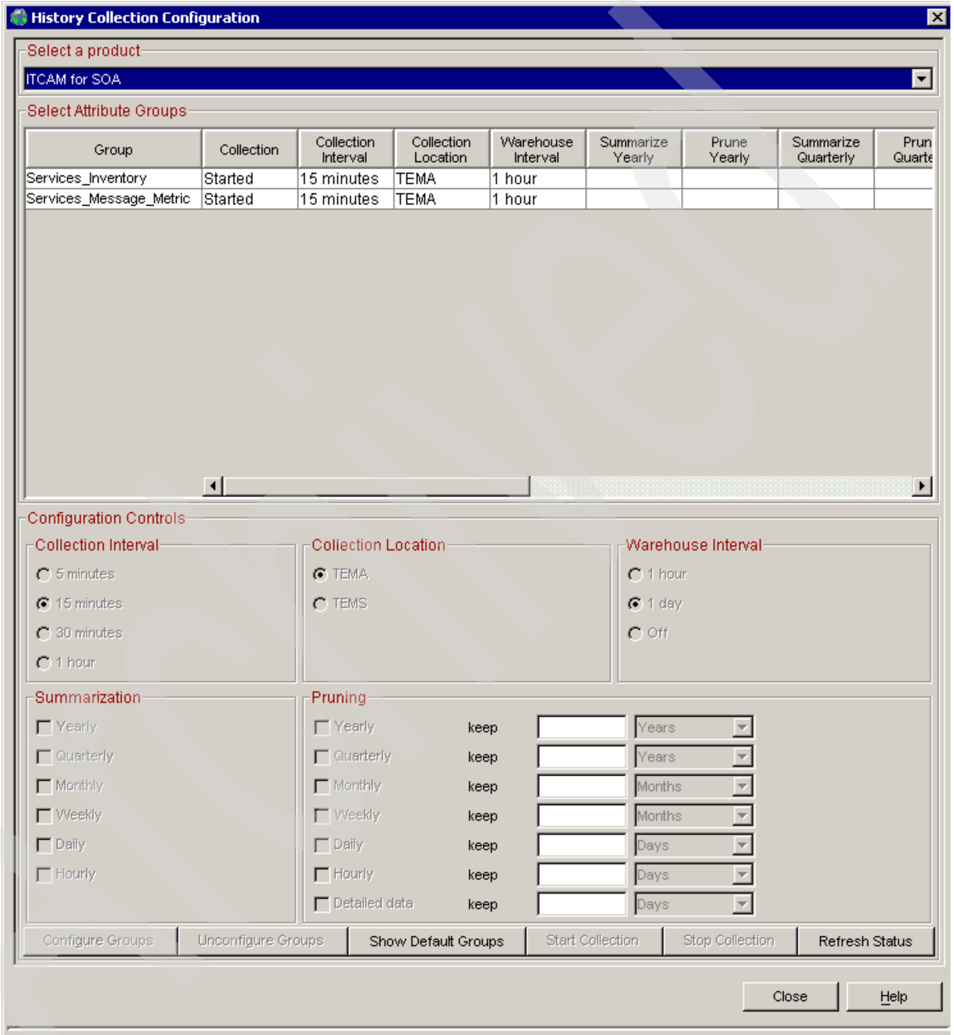


Figure 9-15 Configuring Historical Collection for ITCAM for SOA

ITCAM for SOA collects data in the Service_Inventory and Service_Metrics tables, as shown in Figure 9-15. The data in the Service_Inventory table can be useful for watching performance trends of Web services over time. Collection of historical data for the Services_Metrics table is *only* useful with the IBM Web

Services Navigator application. This Services_Metrics table generates a large volume of historical data, and should therefore only be collected for short periods of time.

9.2.6 Installing IBM Web Services Navigator

IBM Web Services Navigator runs in the Eclipse environment. ITCAM for SOA V6.1 is packaged with its own Eclipse V3.1.2 environment. You can install IBM Web Services Navigator into IBM Rational Application Developer V6.1 and IBM Rational Software Architect V6.1, which are built on the Eclipse platform. Refer to *IBM Tivoli Composite Application Manager for SOA Tools version 6.1.0*, GC32-9494.

IBM Web Services Navigator can be installed with a new Eclipse environment or to an existing Eclipse environment as an Eclipse plug-in. Currently, IBM Web Services Navigator is supported only on Windows and Linux-based operating systems. Figure 9-16 shows the installation path and option that we use.

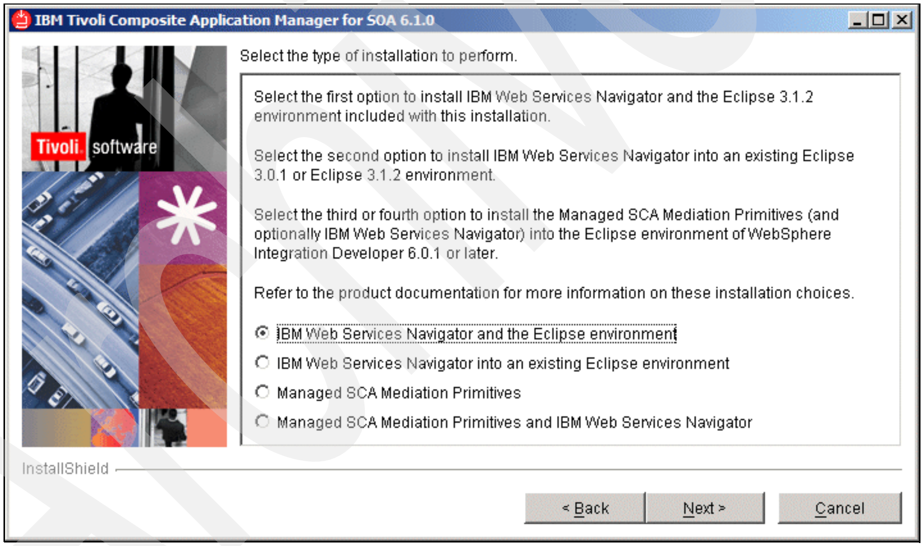


Figure 9-16 IBM Web Services Navigator installation

When the installation is complete, you can launch IBM Web Services Navigator from **Programs → IBM Tivoli Composite Application Manager for SOA 6.1.0 → IBM Web Services Navigator** or by executing **`runNavigator`**.

IBM Web Services Navigator uses a perspective in Eclipse called the Web Services Profiling perspective. For more information about Eclipse and perspective usage in Eclipse, refer to:

http://www.eclipse.org

The initial page when starting IBM Web Services Navigator is the task assistant page to guide you through the use of the navigator (Figure 9-17).

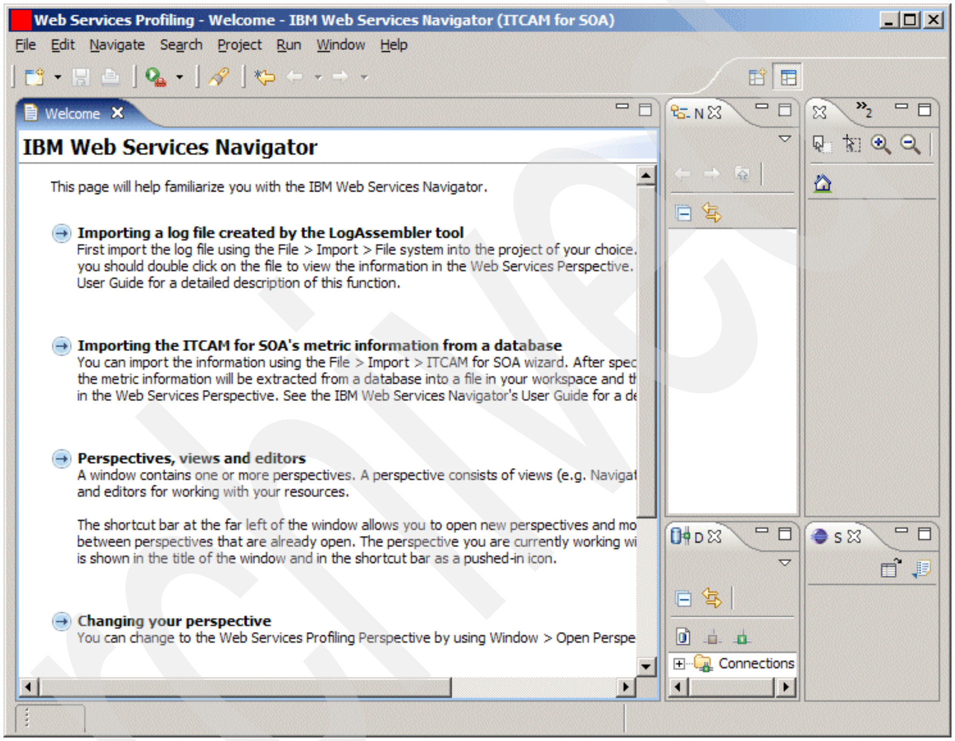


Figure 9-17 Initial help window for IBM Web Services Navigator

You can close this help window using the menu **Window** → **Reset Perspective**. Figure 9-18 shows an empty Eclipse workspace with the Web Services Profiling perspective.

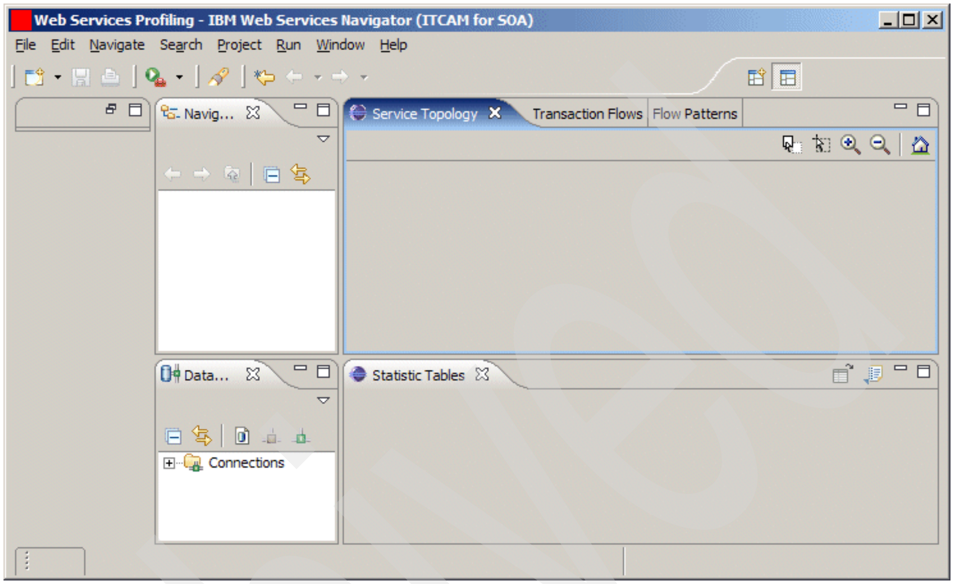


Figure 9-18 Empty IBM Web Services Navigator

9.3 ITCAM for SOA usage scenarios

For this section, we gather metrics from the monitored Web services messages. We discuss the use of ITCAM for SOA in the following sections:

- 9.3.1, “Monitoring Web services calls” on page 482
- 9.3.2, “Filtering Web services calls” on page 485
- 9.3.3, “Using a workflow” on page 489
- 9.3.4, “Using IBM Web Services Navigator” on page 489

9.3.1 Monitoring Web services calls

The workspace of ITCAM for SOA (Figure 9-19 shows the default) in the Tivoli Enterprise Portal is arranged to show Web services calls by servers. The Web services calls are typically identified by the following attributes:

- Frequency
- Response time
- Message length

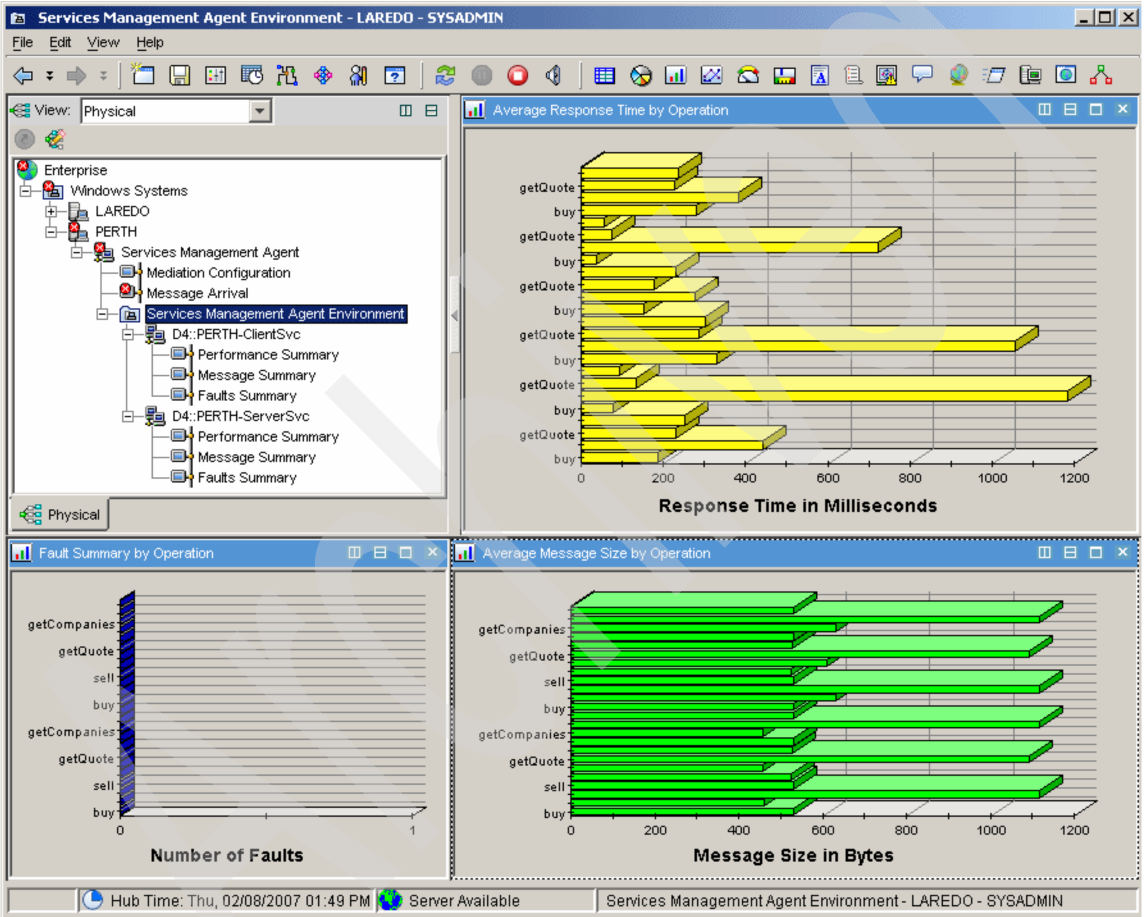


Figure 9-19 Primary workspace for ITCAM for SOA

The workspace in Figure 9-19 displays the primary metrics that are collected by ITCAM for SOA. It shows all active Web services calls in the duration. In our trader application, we have three Web modules, each one accessing DB2, CICS, and IMS. Each Web module serves four Web services calls: getCompanies, getQuote, buy, and sell.

In the performance summary display shown in Figure 9-20, the detailed information of the Web services call performance are shown in a table, along with a response time summary chart.

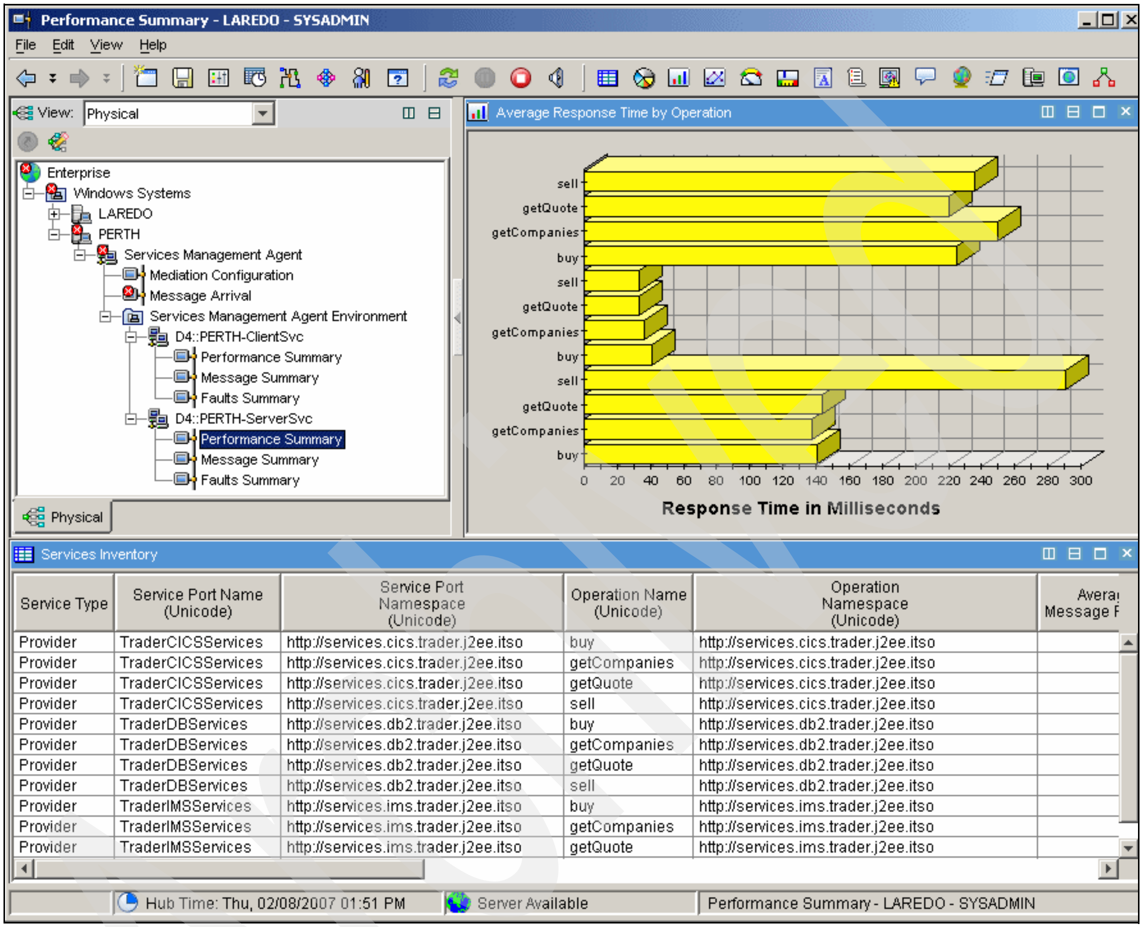


Figure 9-20 Performance summary

The message summary page shows the message statistics (Figure 9-21). This is typically useful to assess the network capacity requirement for the Web services, because it shows both the length and the number of messages for the server.

Note: The message size of zero for SCA monitoring would be encountered since we do not know what the size of an Service Data Object (SDO) is.

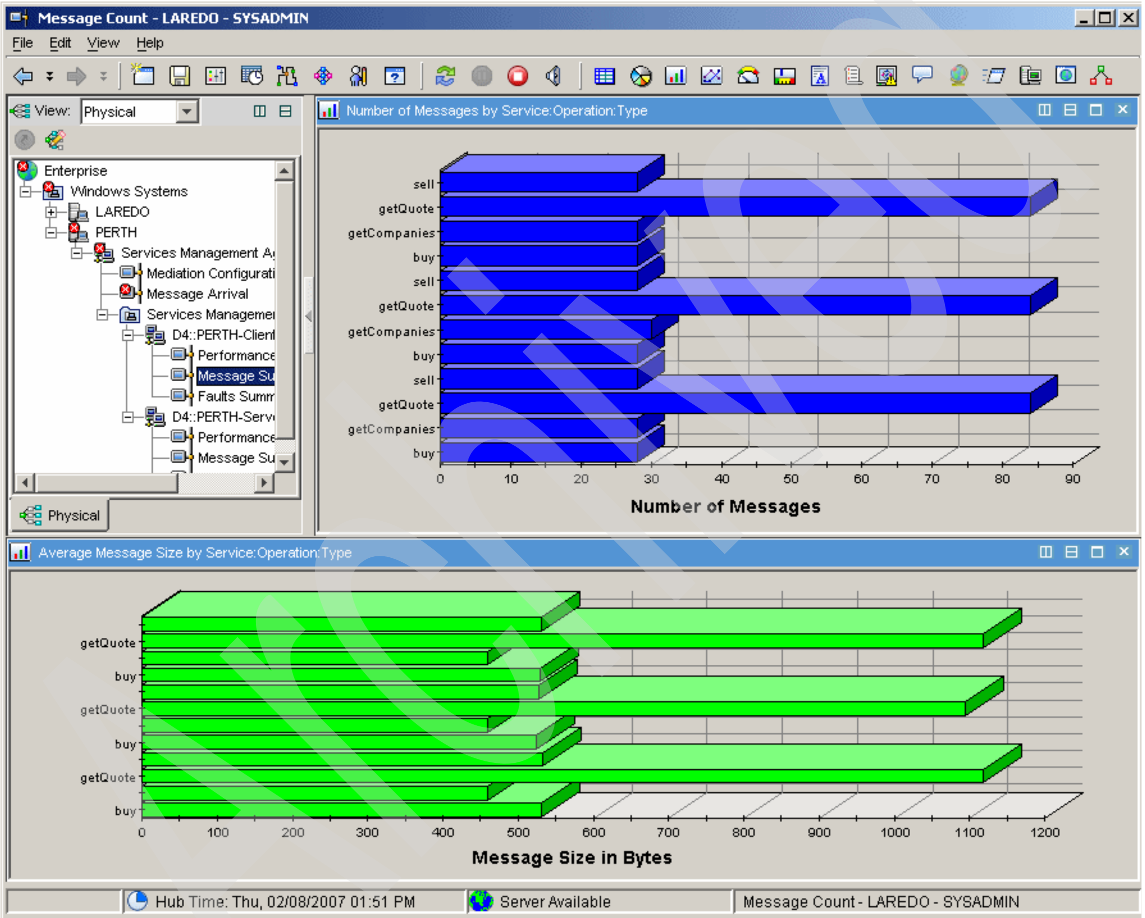


Figure 9-21 Message summary

From a different branch, the message arrival rate is shown in Figure 9-22. This shows the activity of the server in general.

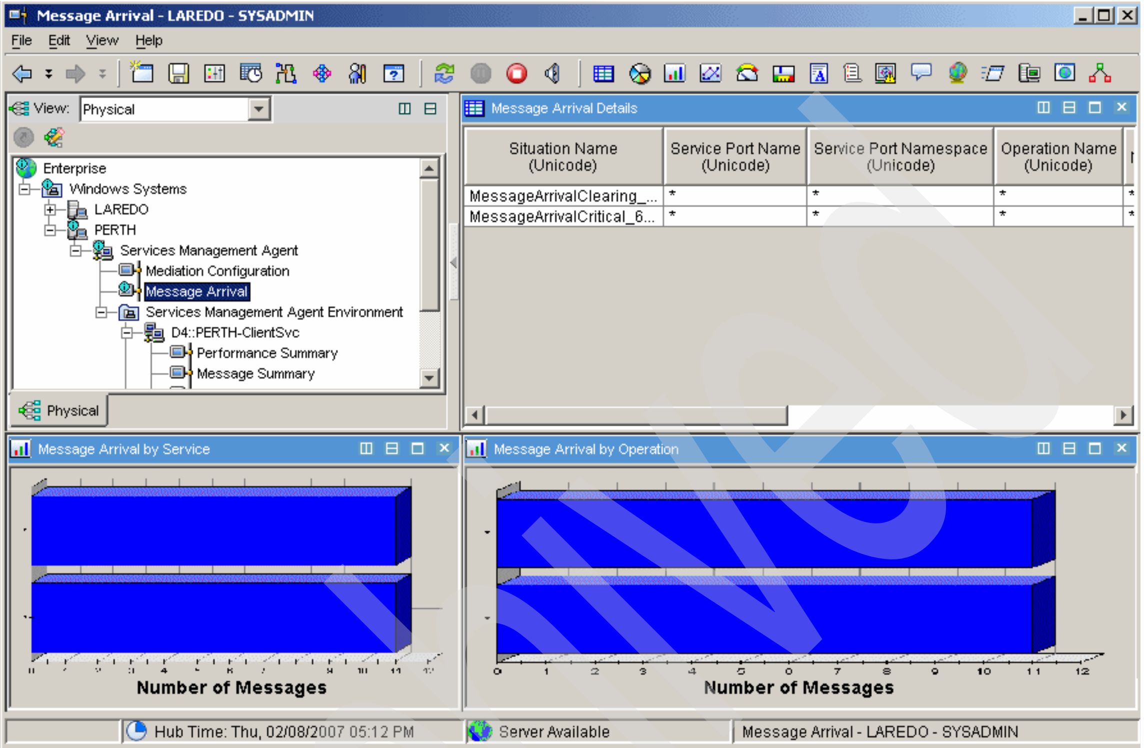


Figure 9-22 Message arrival rate

9.3.2 Filtering Web services calls

Web services calls can be rejected by ITCAM for SOA. This is provided as part of the default take action method in the IBM Tivoli Monitoring V6.1 environment. See “Actions” on page 460 for descriptions of the take action methods.

These actions can be invoked manually or triggered by a situation. This section demonstrates invoking the take action method manually to reject a certain Web services call.

It makes sense to invoke the filtering action from the message arrival table or from the message summary table when you see a lot of unexpected messages flowing through. You can invoke the action using the context menu, as shown in Figure 9-23.

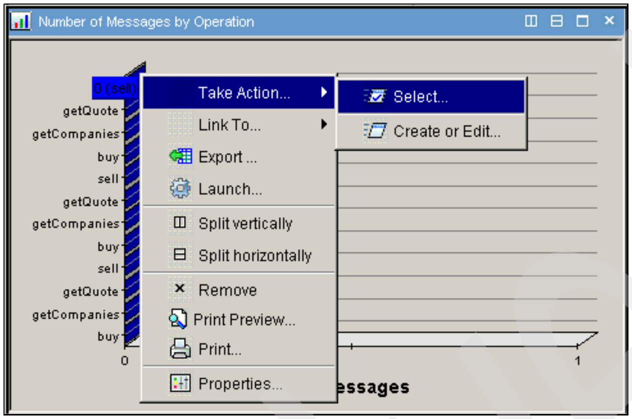


Figure 9-23 Invoking an action

We invoke the addFltrCntrl action. This action requires a set of parameters, as shown in Figure 9-24. Depending on where you initiate the take action, you can get pre-filled data for these arguments.

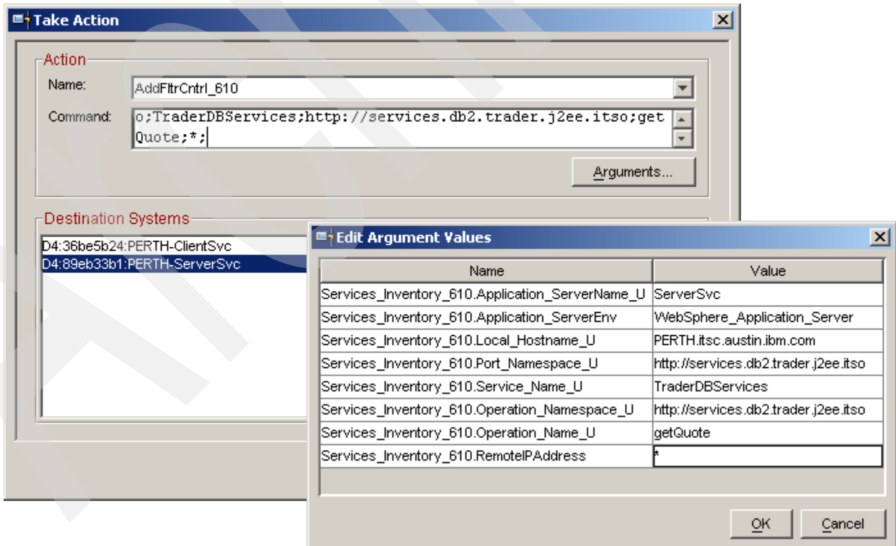


Figure 9-24 Action parameters

After the action has been completed, you can see the effective setting in the Service Management Agent workspace, as shown in Figure 9-25.

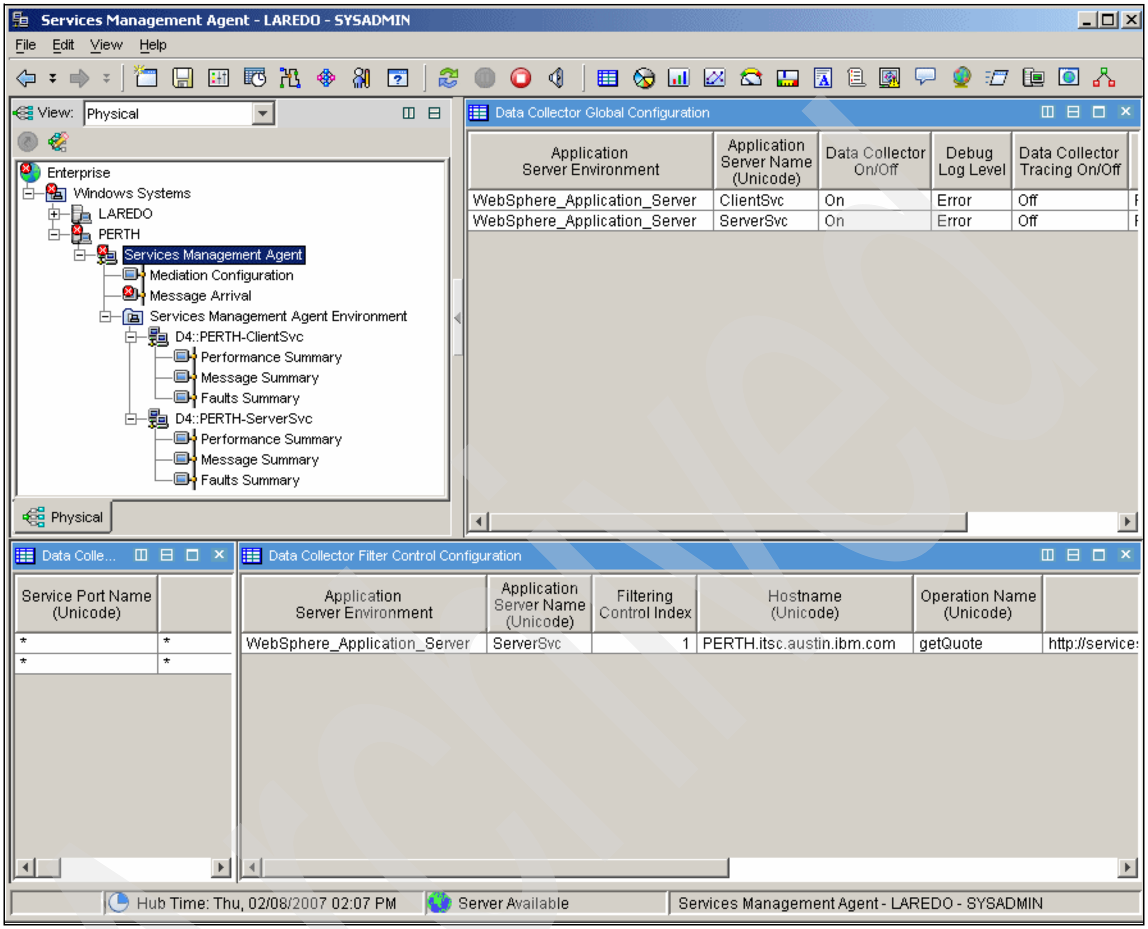
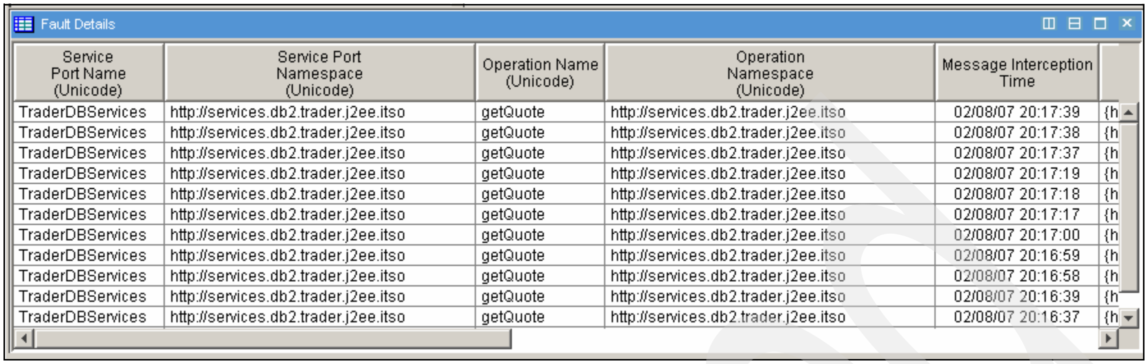


Figure 9-25 Rejected Web services call

The rejected Web services calls show up as fault in the fault summary list. Figure 9-26 shows the fault list.



Service Port Name (Unicode)	Service Port Namespace (Unicode)	Operation Name (Unicode)	Operation Namespace (Unicode)	Message Interception Time	
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:17:39	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:17:38	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:17:37	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:17:19	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:17:18	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:17:17	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:17:00	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:16:59	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:16:58	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:16:39	{h
TraderDBServices	http://services.db2.trader.j2ee.itso	getQuote	http://services.db2.trader.j2ee.itso	02/08/07 20:16:37	{h

Figure 9-26 Rejected Web services call faults

You can remove the filter using the DelFltrCntrl action, as shown in Figure 9-27. This is invoked from the filter list table in Figure 9-25 on page 487 directly. That is why the arguments are automatically filled in.

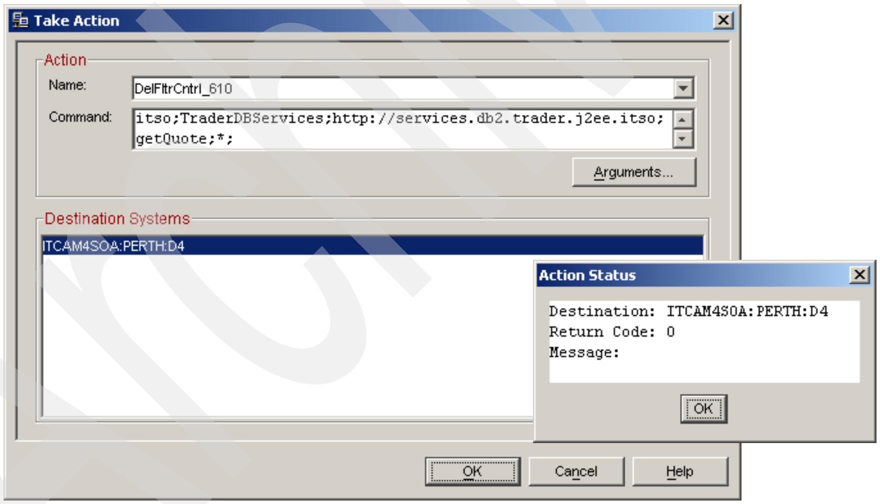


Figure 9-27 Removing a filter

9.3.3 Using a workflow

ITCAM for SOA provides a sample workflow for your customization. Figure 9-28 shows the workflow that is provided for ITCAM for SOA. (By default, this workflow is not started.)

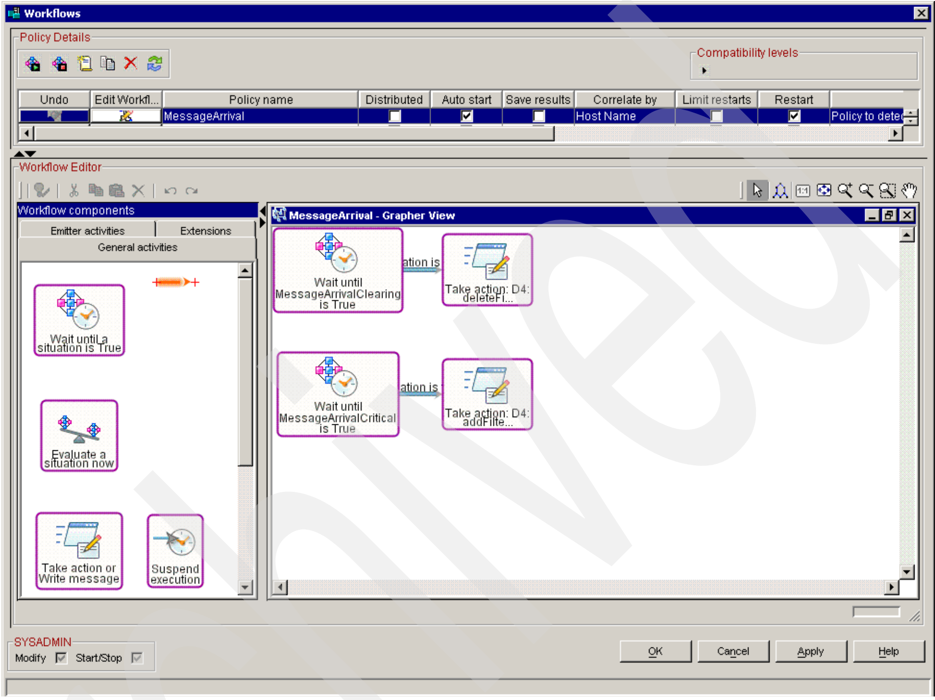


Figure 9-28 Workflow for ITCAM for SOA

The sample workflow uses MessageArrivalCritical and MessageArrivalClearing situations. This workflow is automatically started. When the message arrival approaches a critical threshold, a call filtering is performed for the source IP address. The workflow restores the access when the message arrival rate drops below the threshold.

This workflow is useful for detecting a denial-of-service attack on Web services systems.

9.3.4 Using IBM Web Services Navigator

This section demonstrates the IBM Web Services Navigator to explore our Web services environment. Web services information can be retrieved from the Tivoli

Data Warehouse V2.1 database or from a merged log using the Log Assembler tool. We illustrate both processes.

Using the warehouse data

Before you can work with the historical data from ITCAM for SOA, you must configure a connection from IBM Web Services Navigator to the Tivoli Data Warehouse V2.1 database. We use the DB2 UDB database as the warehouse. First, perform the configuration for historical data collection, as discussed in 9.2.5, “Configuring the warehouse proxy” on page 475.

Perform the following steps:

1. Create a new connection by right-clicking in the Database Explorer view and selecting **New Connection**, as shown in Figure 9-29.

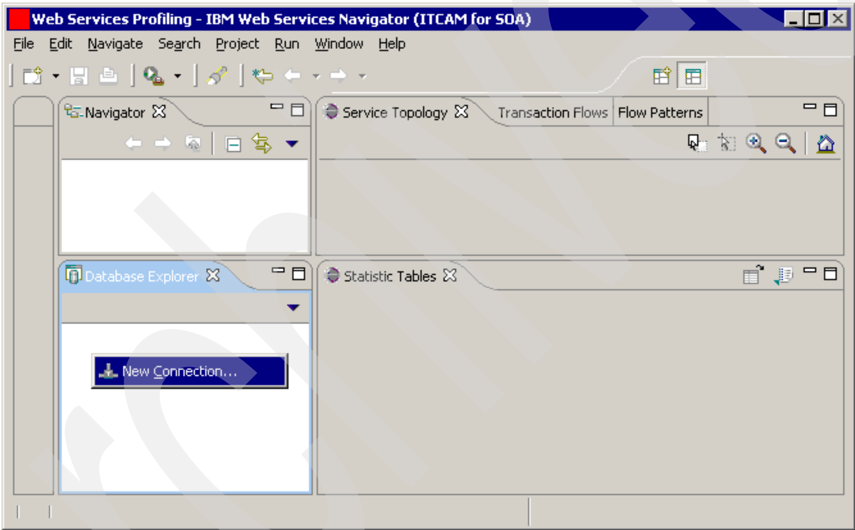


Figure 9-29 Creating a new connection

2. On the New Database Connection window, the Choose a DB2 alias option enables you to select from the local database alias. This is typically not a viable option because the warehouse typically does not reside on the same machine as the IBM Web Services Navigator. We select **Choose database manager and JDBC driver** and call the connection WAREHOUS, as shown in Figure 9-30.

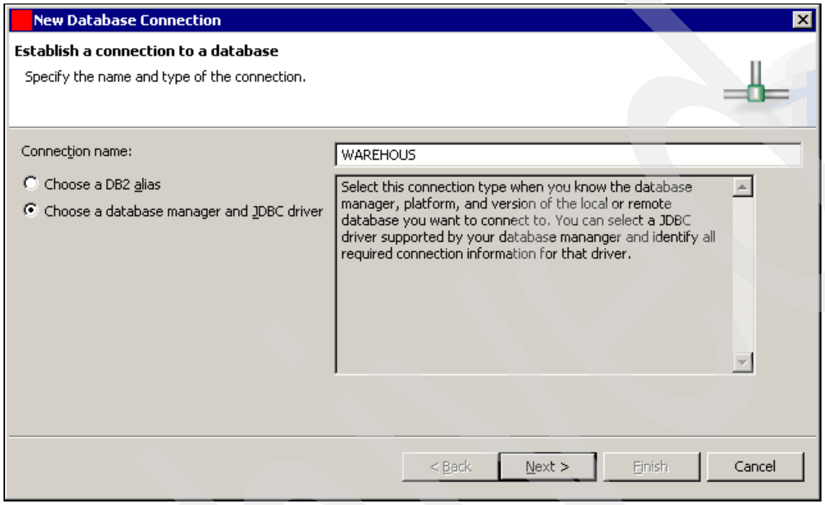


Figure 9-30 New Database Connection

3. The new connection property window (Figure 9-31) includes the database information and user access information for the Tivoli Data Warehouse. Click **Test Connection** to ensure that you can connect to the warehouse database.

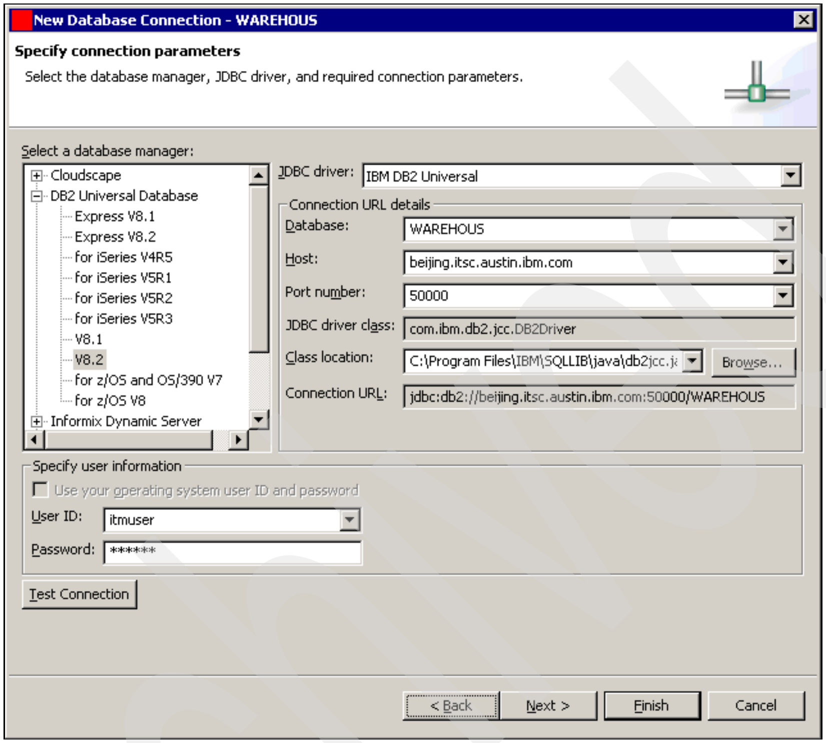


Figure 9-31 WAREHOUS database information

After we connect the warehouse database, we can start working with the IBM Web Services Navigator. Follow this procedure:

1. Create a new project by selecting **File** → **New** → **Project**. The new project type should be **Simple** (Figure 9-32).

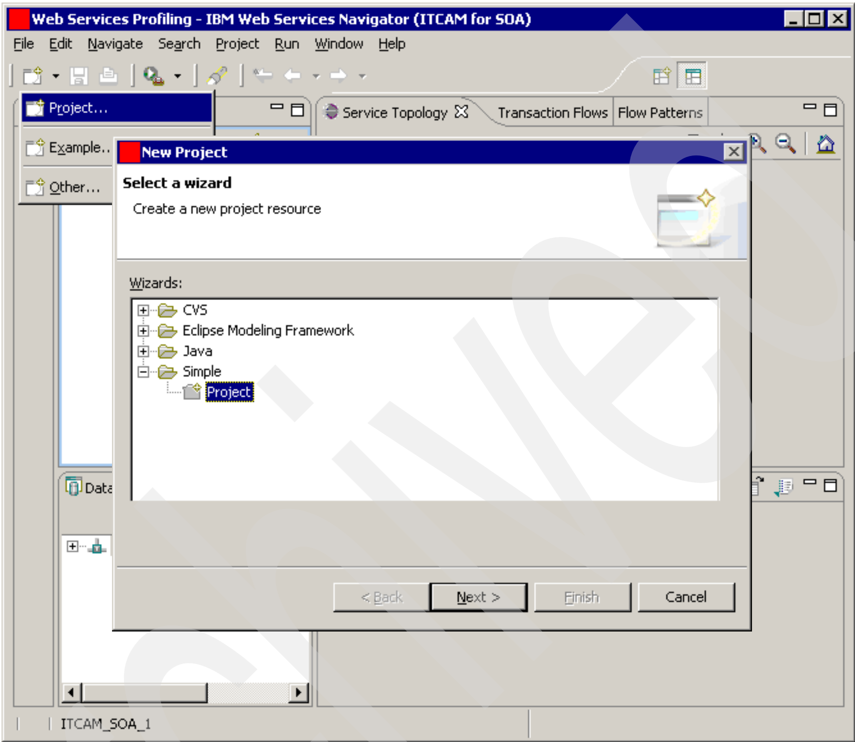


Figure 9-32 Defining a new project

2. Within the project, you can import the ITCAM for SOA data using the menu **File** → **Import** and choosing **ITCAM for SOA** data, as shown in Figure 9-33.

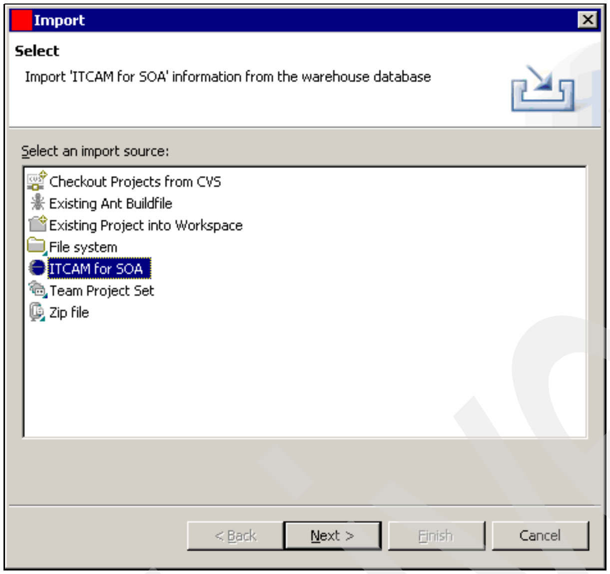


Figure 9-33 Importing ITCAM for SOA data

3. The data import wizard asks about the target workspace and time range to collect (Figure 9-34). The time range that you specify here must be converted to coordinated universal time (UTC). You then specify the database connection. We use an existing connection that we have defined.

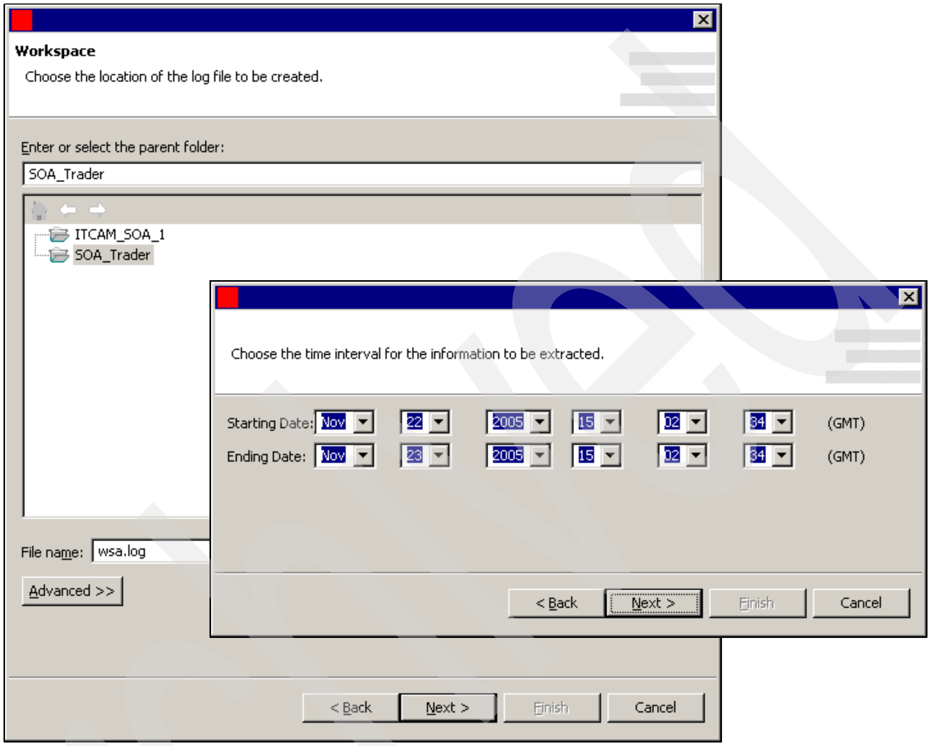


Figure 9-34 Target workspace and time interval

> **Note:** The tables that must exist in the warehouse for the IBM Web Services Navigator can be viewed with the **`db2 list table`** command if you are connected as the warehouse user to the warehouse database:
>
> ```
> Services_Inventory              ITMUSER     T
> 2005-11-23-16.28.03.687001
> Services_Message_Metric         ITMUSER     T
> 2005-11-23-18.28.04.640001
> ```
>
> Also, depending on the transaction volume and mix that you have, you might want to restrict data collection to a smaller time frame to make the results more readable. If you put an invalid value in the date range, the Next button is disabled and not enabled again unless you cancel and restart the import.

Figure 9-35 shows the Service Topology view of our environment.

Service Topology shows the Web services providers and transactions in our environment. It also shows the mix of clients that call each service. Our environment has a dedicated client, laredo, that issues all Web services calls. Laredo is the front-end WebSphere Application Server to which the user connects, and it issues all of the Web services calls.

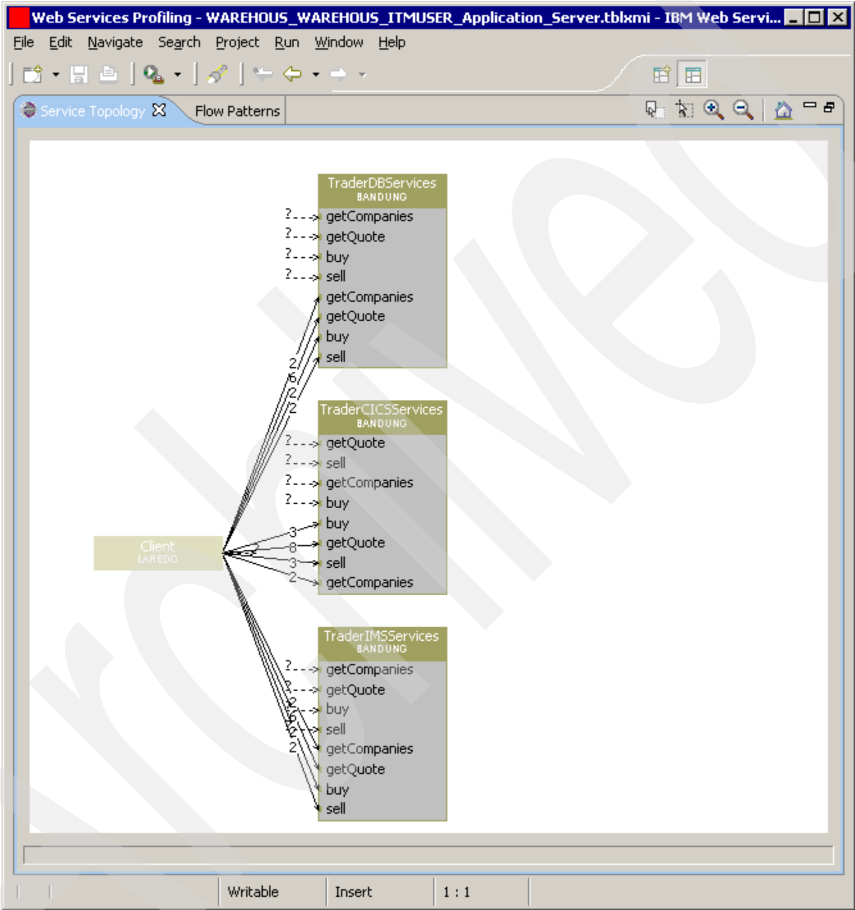


Figure 9-35 Service topology

Note: We discuss other views of the Web Services Navigator in “Using the Log Assembler tool” on page 497.

Using the Log Assembler tool

With the Log Assembler tool, you can specify the nature of transactions that you want to analyze. The data collection is performed using the monitor control action. The monitor control actions are:

- AddMntrCntrl: adding a new control
- UpdMntrCntrl: updating an existing control
- DelMntrCntrl: deleting an existing control

By default, each data collector has a global definition of no monitor control logging for all Web services calls. Individual control can be specified by:

- Service port namespace
- Service port
- Operation namespace
- Operation

> **Note:** Two important items are:
>
> - The ports here are *not* TCP/IP ports. These are port definitions in the wsdl file.
> - A monitoring control definition with all attributes set as wildcards existed. You must use **`UpdMntrCntrl`** to change this,

We decided to add monitor control for an individual service port for our Web services: TraderDBServices, TraderIMSServices, and TraderCICSServices. Figure 9-36 shows the action invocation for TraderIMSServices.

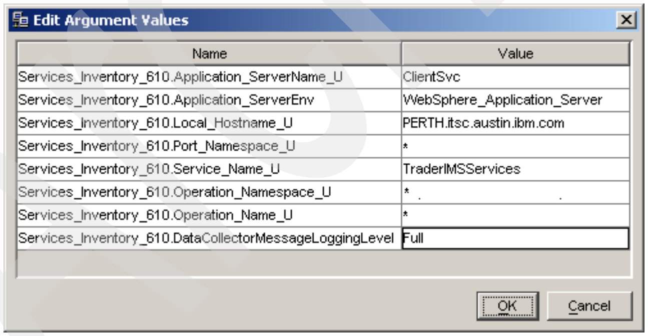


Figure 9-36 Monitor control action

The resulting control can be seen in the Service Management Agent workspace. Figure 9-37 shows the monitor list.

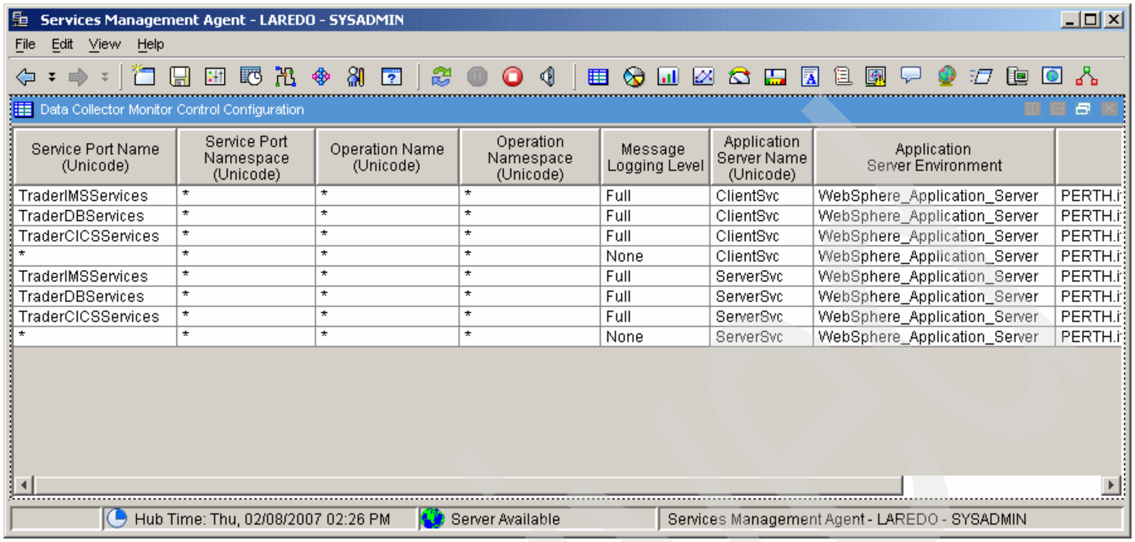


Figure 9-37 Monitor list

This monitor control specification generates the metric log and content log files in the $ITMhome\TMAITM6\KD4\logs:

- The content log is stored for each data collector. Each content log file can be up to 500 MB in size. The file name of the content log is in the format of:

  `KD4.<env>..<cell>.<node>.<server>.content.log`

- There are multiple metric log files. The metric logs are stored in either the KD4.DCA.CACHE or KD4.DCA.CACHE\archive subdirectory of the logs directory. These files have the name of:

  `KD4.<env>..<cell>.<node>.<server>.metric.log.<timestamp>`

You must select the appropriate content logs and metric logs and collect them to the machine on which you install the Web Services Navigator. The Log Assembler tool is installed with the Web Services Navigator. The Log Assembler processes and merges each metric log, while retrieving the message content from the content log file. The result is then a single merged metric log file from multiple servers, each with its own message contents attached.

The easiest way to process these logs is using a batch file, because the file names are long and you have to invoke the command multiple times. In our tests, we invoke 108 Web services calls within three minutes on two application servers. This gave us 28 metric log files and two content log files. Figure 9-38 shows our log files.

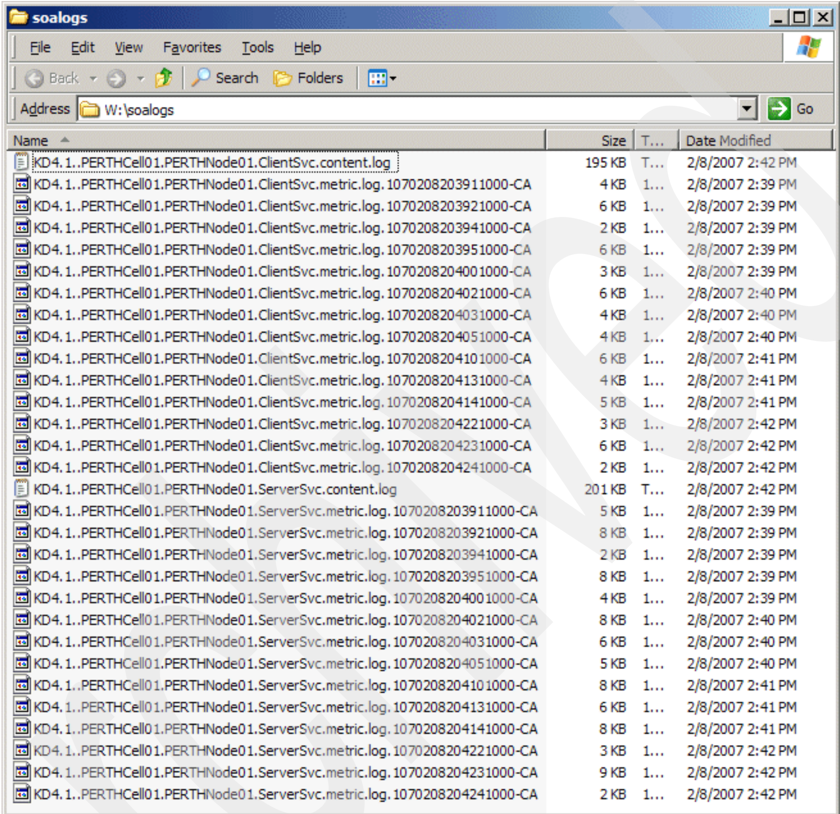


Figure 9-38 Log files

We use the batch file in Example 9-3 to process these log files.

Example 9-3 Running Log Assembler

```
SET TOOLPATH="C:\IBM\ITCAM for SOA 6.1.0\Tools"
SET LOGAJAR=com.ibm.websightView.LogAssembler.jar
cd \soalogs

for %%i in (*.ClientSvc.metric.log.*) do %toolpath%\_jvm\jre\bin\java
-cp %toolpath%\lib\%logajar%;%toolpath%\lib\jlog.jar -jar
%toolpath%\lib\%logajar% 0 a soa.Merged.log C:\soalogs\%%i
C:\soalogs\KD4.1..PERTHCell01.PERTHNode01.ClientSvc.content.log
for %%i in (*.ServerSvc.metric.log.*) do %toolpath%\_jvm\jre\bin\java
-cp %toolpath%\lib\%logajar%;%toolpath%\lib\jlog.jar -jar
%toolpath%\lib\%logajar% 0 a soa.Merged.log C:\soalogs\%%i
C:\soalogs\KD4.1..PERTHCell01.PERTHNode01.ServerSvc.content.log
```

Note that from Example 9-3:

- The Log Assembler must be invoked separately for each server. The metric log for ClientSvc must be processed with the content log for ClientSvc. You cannot combine the processing.
- The Log Assembler uses two JAR files, the LogAssembler.jar and jlog.jar.
- The arguments of the command are:

logging level	0, 1, or 2. Nonzero values generate a lot of output.
processing	o or a - represent overwrite or append to the log files.
target log file name	This is the merged log file name.
metric log file name	The metric log that you want to be processed.
content log files	The content log files for a single server.

We are now ready to import the log file into Web Services Navigator:

1. First, we need an empty project (or an existing one) to store the imported log file definitions. Select **File** → **New** → **Project**. We create a simple project (Figure 9-39).

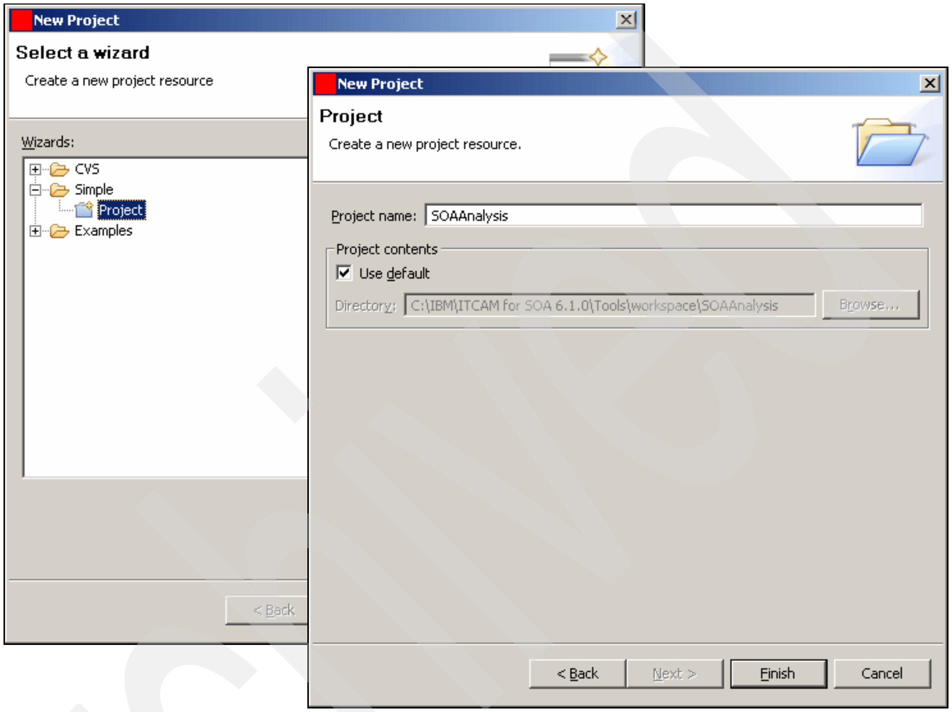


Figure 9-39 Creating a project

2. We then import the log file. Select **File** → **Import** and select from the file system the merged log from the Log Assembler tool. Figure 9-40 shows the import process.

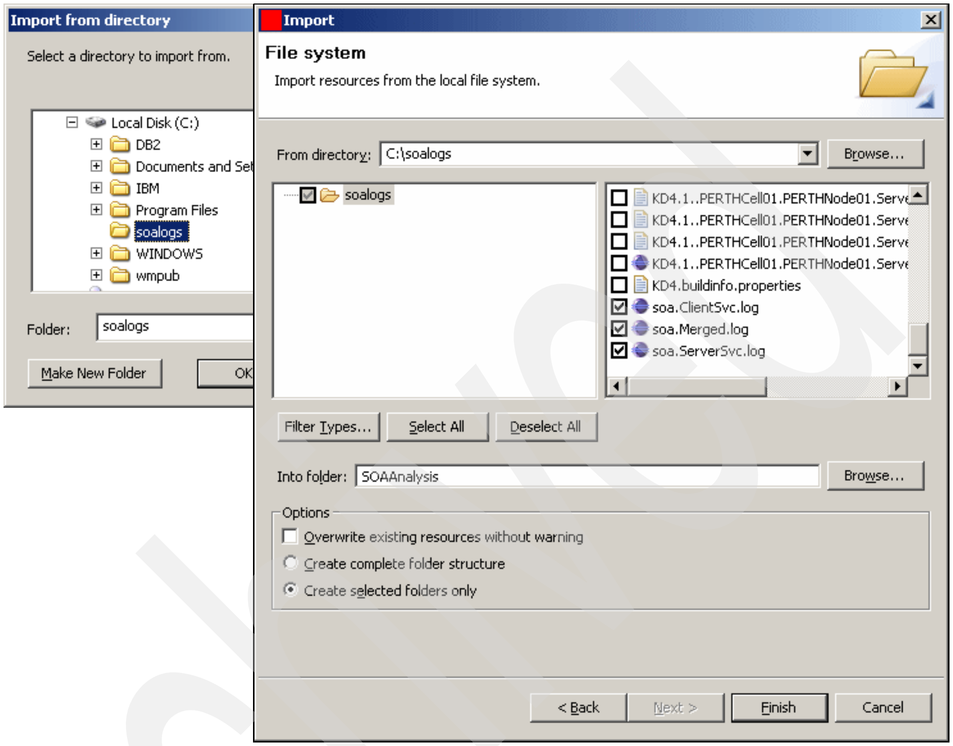


Figure 9-40 Importing log files

In Figure 9-40, note that we experimented with the log files. We have individual server log files and the merged log files. The individual server log files do not generate a full picture of the Web services calls. The following discussion covers the merged log file only.

We have the Service Topology view of the merge log shown in Figure 9-41.

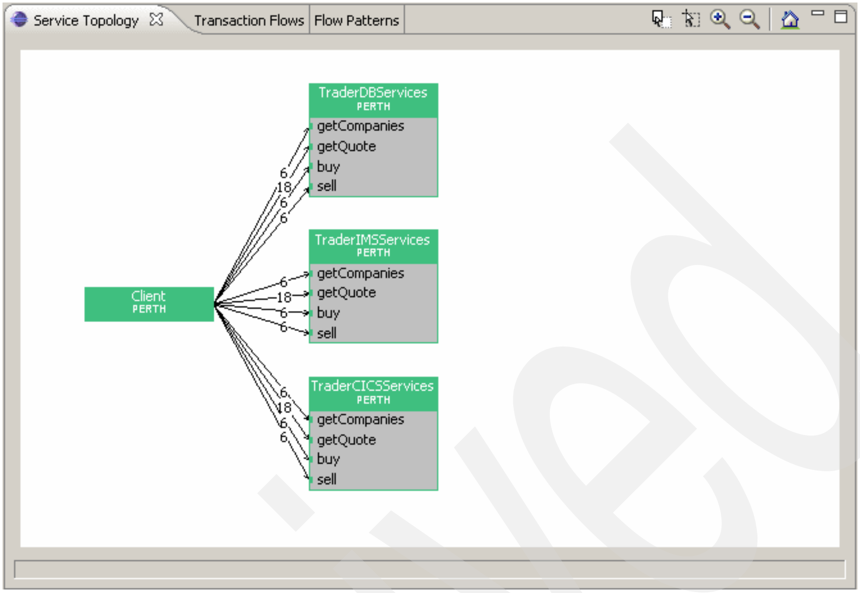


Figure 9-41 Service Topology

Figure 9-41 shows the calling pattern of the Web services. The green boxes indicate the service name, and the operations are listed within it. The arrows indicate service calls with numbers showing the number of invocations. You can also summarize the calls based on the service name instead of the operation by collapsing the boxes, as shown in Figure 9-42.

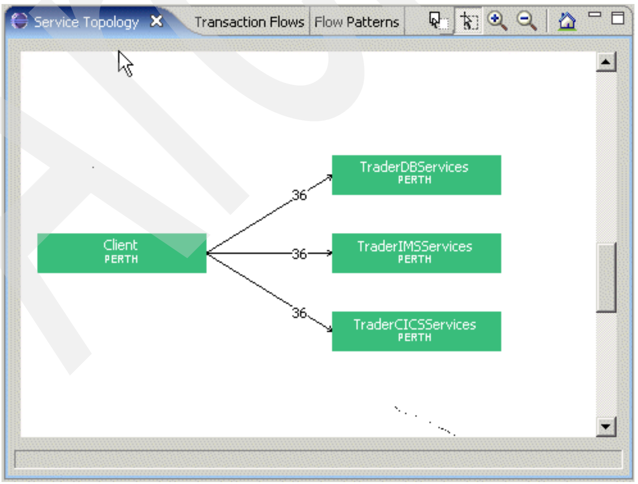


Figure 9-42 Operation summary

Figure 9-43 shows the transaction flows. It shows the time line of the Web services calling sequences. The busy chart shows all 108 Web services calls that we made.

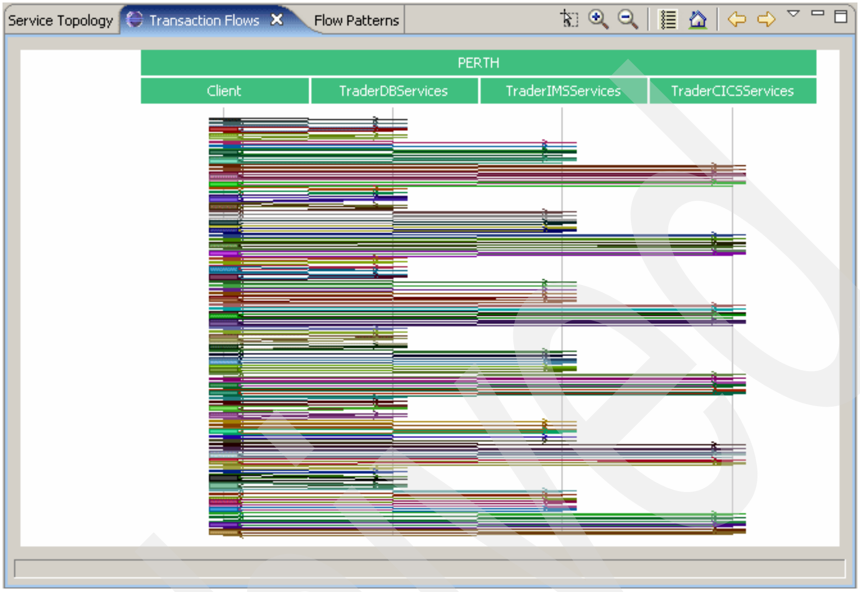


Figure 9-43 Transaction flows

Hovering the cursor over any of those lines provide the call details of the Web service. Selecting the service shows yellow highlighting and displays the details in the bottom pane (Figure 9-44).

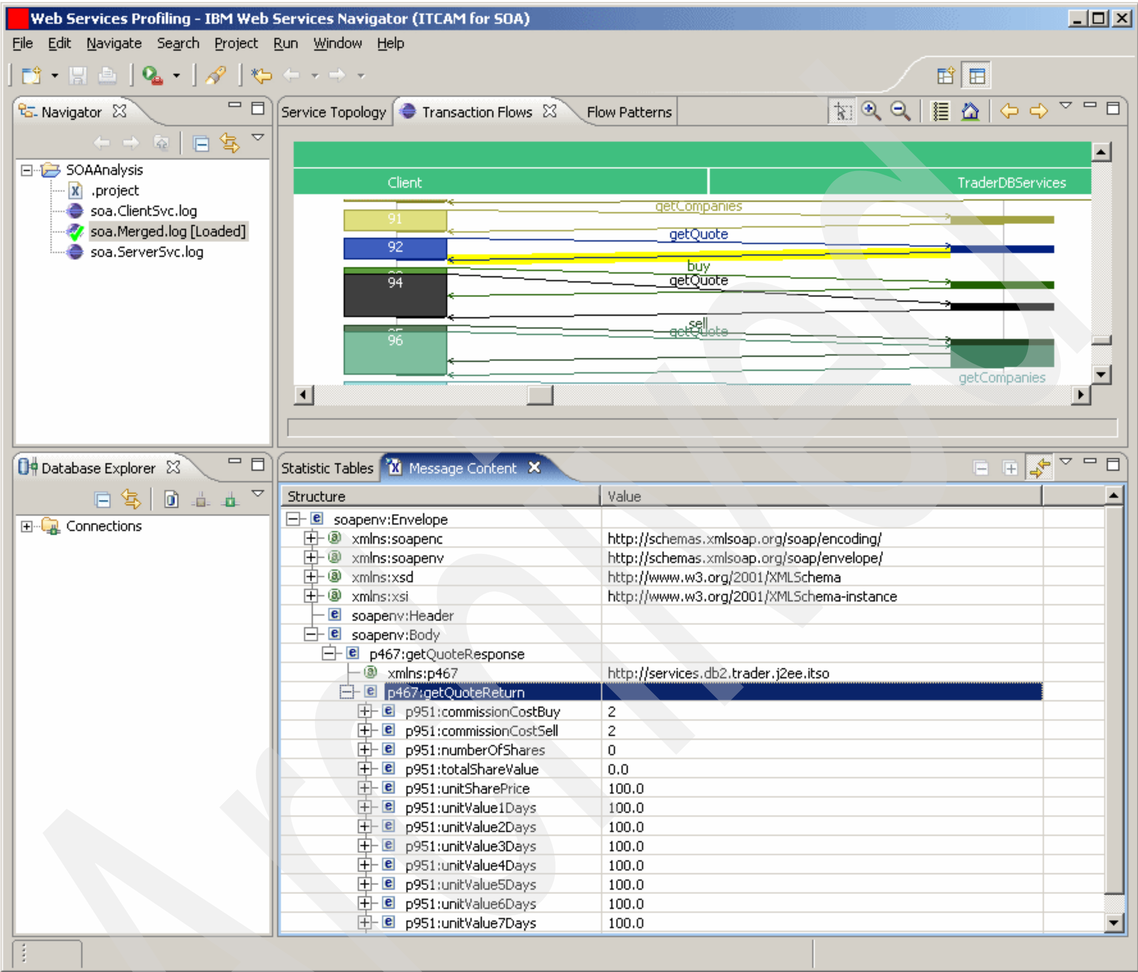


Figure 9-44 Message content

The flow pattern summarizes the transaction flows in Figure 9-43 on page 504 and collapses any similar invocation pattern. A pattern can be identified by the requestor, provider, service name, and operation name. Figure 9-45 shows the flow pattern.

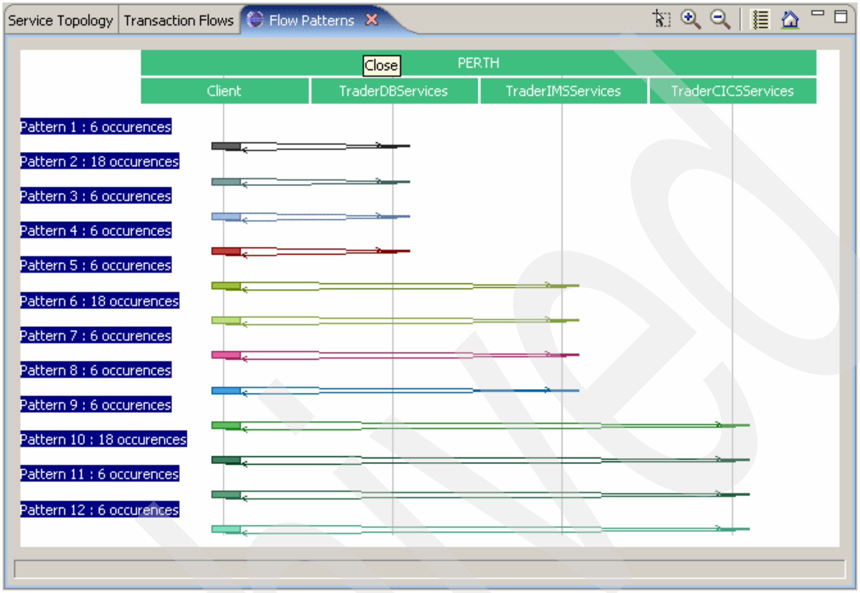


Figure 9-45 Flow patterns

9.4 Discovery Library Adapters in SOA management

Visualization can be a very powerful tool for understanding the behavior of a complex system. The Discovery Library Adapter is a software program that extracts data from a source application and writes the data into an XML file in Identity Markup Language (IdML) format, called an IDML book. ITCAM for SOA contains components that provide a data store called Tivoli Common Object Repository (TCOR) that can store and visualize data from IdML books. In this section, we demonstrate the visualization of IdML books created from WebSphere Services Registry and Repository and from ITCAM for SOA itself.

The discussion includes:

- 9.4.1, “Discovery Library Adapters setup” on page 507
- 9.4.2, “Defining Tivoli Common Object Repository” on page 507
- 9.4.3, “WebSphere Services Registry and Repository setup” on page 508
- 9.4.4, “Discovery Library Adapter” on page 511

- 9.4.5, “Displaying IdML data in Tivoli Enterprise Portal” on page 513
- 9.4.6, “ITCAM for SOA Discovery Library Adapter” on page 518

9.4.1 Discovery Library Adapters setup

The supported Discovery Library Adapters to be used with ITCAM for SOA are:

- ITCAM for SOA Discovery Library Adapter
- WebSphere Service Registry and Repository (WSRR) Discover Library Adapter
- Business Process Execution Language (BPEL) for Web Services Discovery Library Adapter

A Discovery Library Adapter (DLA) extracts data from a source application to generate an IdML book file. Data in IdML books contain managed elements and relationships between them. These books are then copied or transferred to a shared directory, called the Discovery Library File Store (DLFS).

A bulk load program script is then executed to load the books from the Discovery Library File Store into the Tivoli Common Object Repository (TCOR) database. This database is supplied with ITCAM for SOA V6.1

The IdML data books can be a full refresh or a delta from the data in the Managed Software System (MSS). It may be noted that delta books only modifies managed element data and relationships, creates, updates, and deletes from existing managed elements or relationships.

9.4.2 Defining Tivoli Common Object Repository

The Tivoli Common Object Repository is created on the IBM Tivoli Monitoring systems with DB2 Version 8.1 or later running. On our Linux system, the following procedure sets up the repository:

1. DB2 Universal Database V6.1 or later and ITCAM for SOA are installed.
2. Source the DB2 profile using the command **`source /home/db2inst1/sqllib/db2profile`**. (In a Windows system, a similar effect can be achieved by opening DB2 command window using the **`db2cmd`** command.)
3. Assuming that DB2 is started, change the directory to /opt/IBM/ITM/li6243/cq/Products/KD4/bin and run **`./kd4setuptcore.sh`**.
4. After kd4setuptcore.sh script completes, reconfigure the Tivoli Enterprise Portal server by running the command **`/opt/IBM/ITM/bin/itmcmd configure -A cq`**.

5. After the reconfigure process is complete, restart the Tivoli Enterprise Portal server.

9.4.3 WebSphere Services Registry and Repository setup

Currently, WebSphere Services Registry and Repository V6.0.1 is supported. The WebSphere Services Registry and Repository stores information regarding Web services and stores them in its repository. It provides Web services with a life-cycle governance function.

We loaded our WSDL document into WebSphere Services Registry and Repository. The following procedure loads WSDL documents into WebSphere Services Registry and Repository:

1. Open the WebSphere Services Registry and Repository console, such as:

   `http://laredo:9080/ServiceRegistry`

2. Go to **Service Documents** → **Load Document** and specify the path of the source WSDL, as shown in Figure 9-46. Click **OK**.

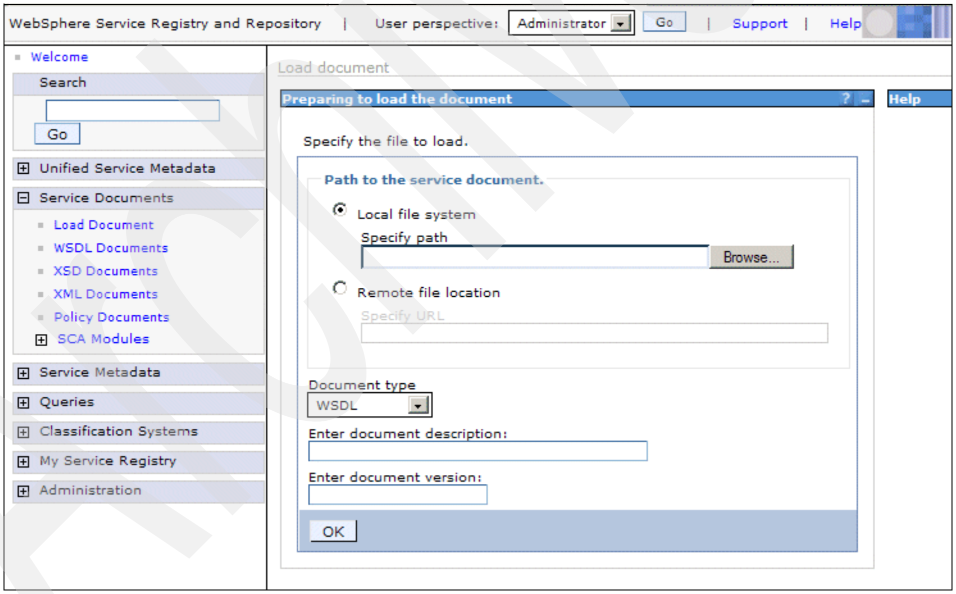


Figure 9-46 WebSphere Service Registry and Repository console

3. In our environment, we loaded several WSDL documents, as shown in Figure 9-47.

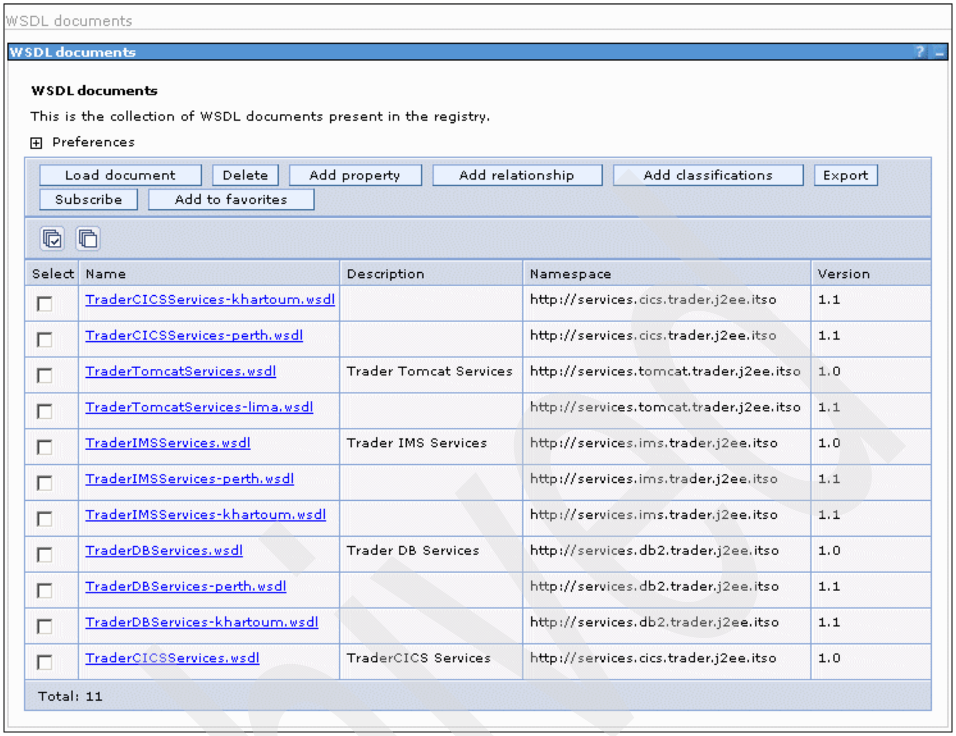


Figure 9-47 List of loaded WSDL file

4. The Discovery Library Adapter only processes the Web services definition with the life-cycle status of deploy or manage. We must add this classification to the generated Web services definition.

> **Note:** Although we load the WSDL document, the classification must be applied to the created Web Services metadata, *not* to the WSDL document.

5. Classification should be added from the **Service Metadata** → **WSDL** → **Services** menu. The resulting services window is shown in Figure 9-48.

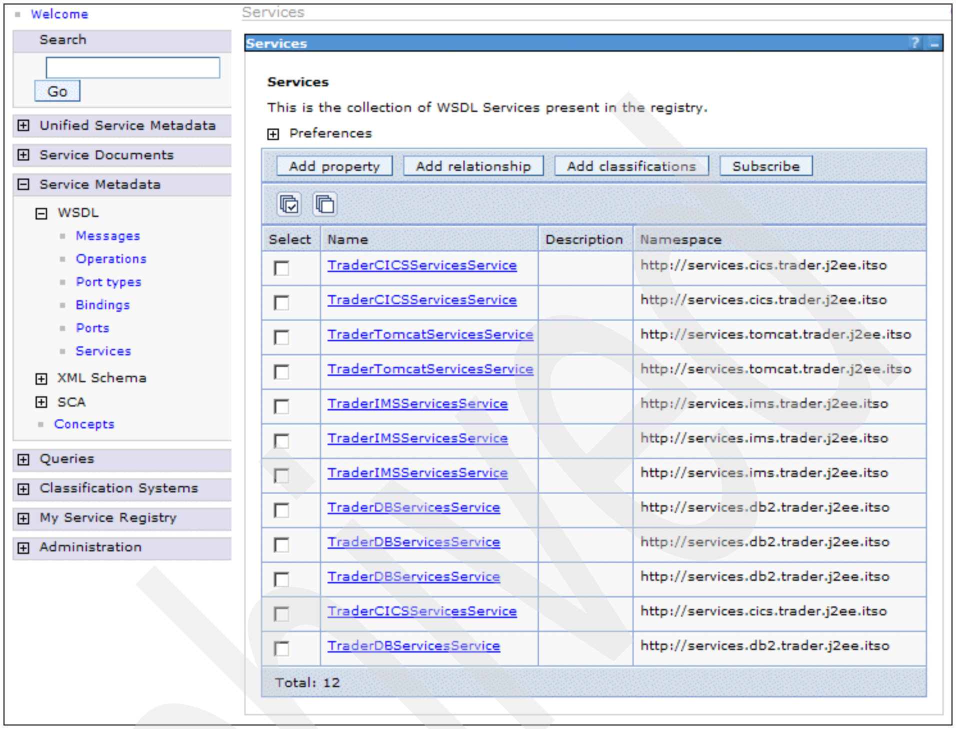


Figure 9-48 WSDL services metadata

6. From Figure 9-48 on page 510, select the appropriate entries that you want to be discovered and click **Add classifications**. The classification page is shown in Figure 9-49. Select either Deploy or Manage and click the **Add** button to add the classification. Click **OK** when done.

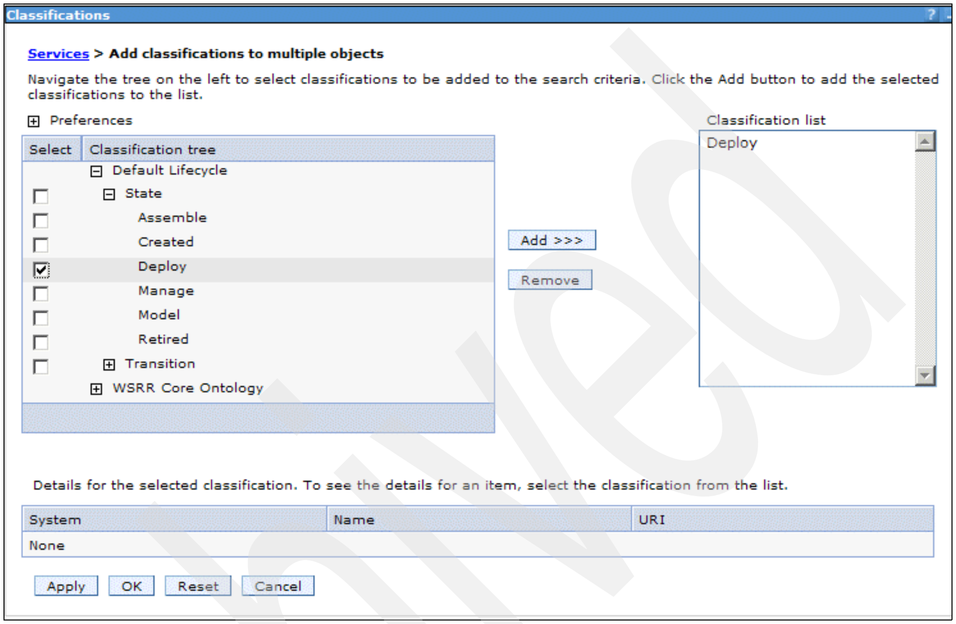


Figure 9-49 Classification setting

9.4.4 Discovery Library Adapter

WebSphere Service Registry and Repository Discovery Library Adapter must be installed on the server where WebSphere Service Registry and Repository is installed. The Discovery Library Adapter is supplied from ITCAM for SOA under the installation image KD4/DLA/ as WSRRV600_DLA.zip and WSRRV600_DLA_fixpack.zip. We unzip both files under the /opt/IBM/WSRRV6_DLA path.

Note: To use the Discovery Library Adapter with WebSphere Services Registry and Repository V6.0.2, you have to copy sdo-int.jar from <WSRR-home> to <WAS-home>/classes and restart the WebSphere Application Server that hosts the WebSphere Services Registry and Repository.

From the /opt/IBM/WSRRV6_DLA/bin directory, edit the WSRR_DLA.config.properties and DLA_FileTransfer.properties:

- For WSRR_DLA.config.properties, you must at least customize the parameters in Example 9-4. This specifies how to communicate with the WebSphere Services Registry and Repository API.

Example 9-4 Access to WebSphere Services Registry and Repository

```
com.ibm.management.soa.dla.wsrr.address=laredo
com.ibm.management.soa.dla.wsrr.port=2809
com.ibm.management.soa.dla.wsrr.domainName=itsc.austin.ibm.com
com.ibm.management.soa.dla.wsrr.websphereSecurityEnabled=false
com.ibm.management.soa.dla.wsrr.securityEnabled=false
com.ibm.management.soa.dla.wsrr.userid=
com.ibm.management.soa.dla.wsrr.password=
```

- For DLA_FileTransfer.properties, you must specify the host on which the Tivoli Common Object Repository is installed and the path to store the IdML books. These parameters are shown in Example 9-5. The ftp password will be encoded after the first invocation of the file transfer program.

Example 9-5 File transfer properties

```
com.ibm.management.soa.dla.filetransfer.host=peoria.itsc.austin.ibm.
com
com.ibm.management.soa.dla.filetransfer.targetDir=/opt/IBM/WSRRV6_DL
A/staging
com.ibm.management.soa.dla.filetransfer.stagingDir=../staging
com.ibm.management.soa.dla.filetransfer.fileFilter=*.xml
com.ibm.management.soa.dla.filetransfer.securityEnabled=true
com.ibm.management.soa.dla.filetransfer.userid=root
com.ibm.management.soa.dla.filetransfer.password=******
com.ibm.management.soa.dla.filetransfer.password.encoded=aXRzMGcwMGQ
=
```

- The logging option for both the discovery and file transfer that we use is shown in Example 9-6. The option allows us to understand the processing better.

Example 9-6 Logging option

```
com.ibm.management.soa.dla.wsrr.log.logFileName=WSRRDLALog.log
com.ibm.management.soa.dla.wsrr.log.logFileDir=../logs
com.ibm.management.soa.dla.wsrr.log.loggingLevel=DEBUG_MAX
com.ibm.management.soa.dla.wsrr.log.logFileSize=2000000
com.ibm.management.soa.dla.wsrr.log.logFileCount=3
com.ibm.management.soa.dla.wsrr.log.enableLogging=true
com.ibm.management.soa.dla.wsrr.log.logToFile=true
com.ibm.management.soa.dla.wsrr.log.logToConsole=true
```

We run the discover using the command **./WSRR_DLA.sh -r** to create a refresh XML data book. The execution of this is shown in Figure 9-50.

```
[root@laredo bin]# ./WSRR_DLA.sh -r
2007-07-06 15:32:21.793-05:00 [main] KD4SR0017I The discovery
library adapter is starting its discovery process.
2007-07-06 15:32:28.982-05:00 [P=941820:O=0:CT] KD4SR0010I The
discovery library adapter book:
WSRRv600.laredo.itsc.austin.ibm.com.2007-07-06T20.32.25.659Z.refresh
.xml was generated and stored into the following directory:
../staging/.
2007-07-06 15:32:29.020-05:00 [P=941820:O=0:CT] KD4SR0018I The
discovery library adapter is exiting its discovery process.
2007-07-06 15:32:29.591-05:00 [main] KD4FT0020I The file transfer
process is starting.
KERBEROS_V4 rejected as an authentication type
2007-07-06 15:32:31.347-05:00 [main] KD4FT0009I The file:
../staging/WSRRv600.laredo.itsc.austin.ibm.com.2007-07-06T20.32.25.6
59Z.refresh.xml was successfully transferred to host:
peoria.itsc.austin.ibm.com
2007-07-06 15:32:31.930-05:00 [main] KD4FT0021I The file transfer
process is ending.
```

Figure 9-50 Running the Discovery Library Adapter

9.4.5 Displaying IdML data in Tivoli Enterprise Portal

Loading the discovery book is performed by running the bulk load script:

`/opt/IBM/ITM/li6243/cq/Products/KD4/tcore/bin/loadidml.sh -f`

The ITCAM for SOA view can be added to the view list in the Tivoli Enterprise Portal. From Tivoli Enterprise Portal, click the administer users icon in the tool bar. The Administer Users pop-up window is shown in Figure 9-51.

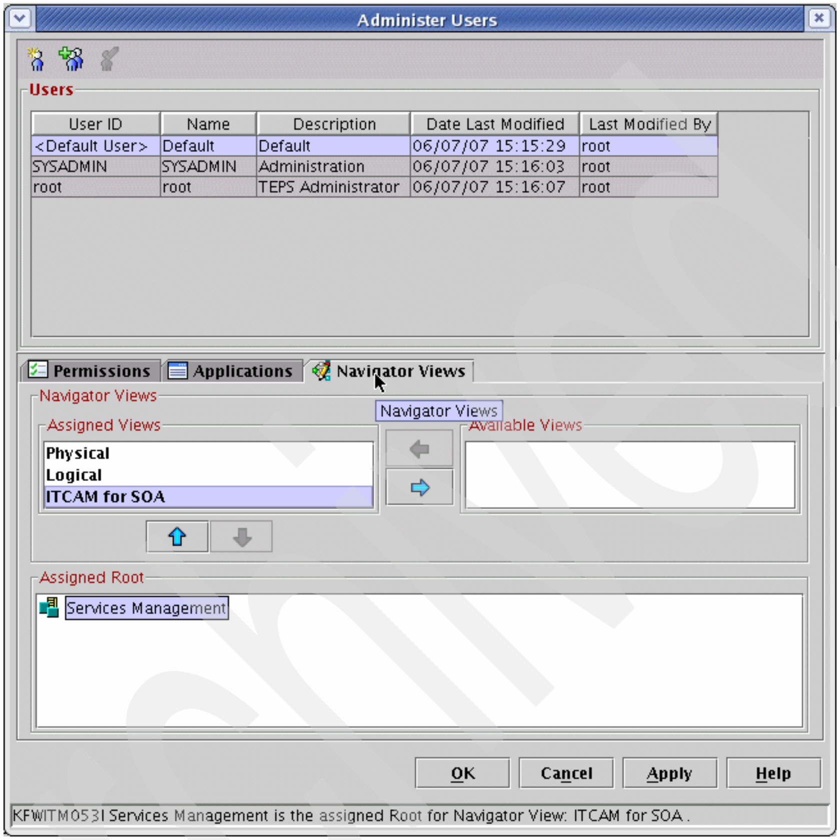


Figure 9-51 Administer Users: Navigator Views

Click **Apply** and then **OK** to add the ITCAM for SOA view to the drop-down list of the views.

The following steps work with the ITCAM for SOA view:

1. The resulting display in Tivoli Enterprise Portal for selecting the SOA view is shown in Figure 9-52. Click the **Physical** drop-down list and select **ITCAM for SOA**.

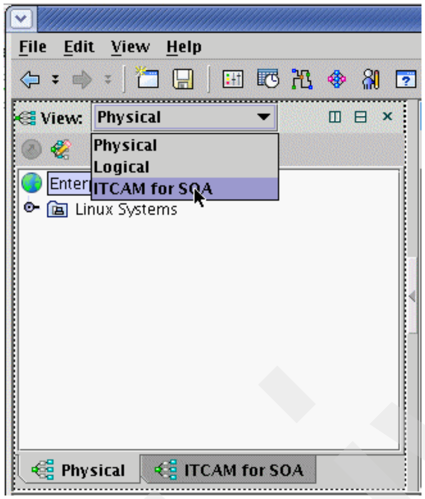


Figure 9-52 Select ITCAM for SOA view

2. The workspace lists the Web services that have been loaded from the **loadidml** command. See Figure 9-53.

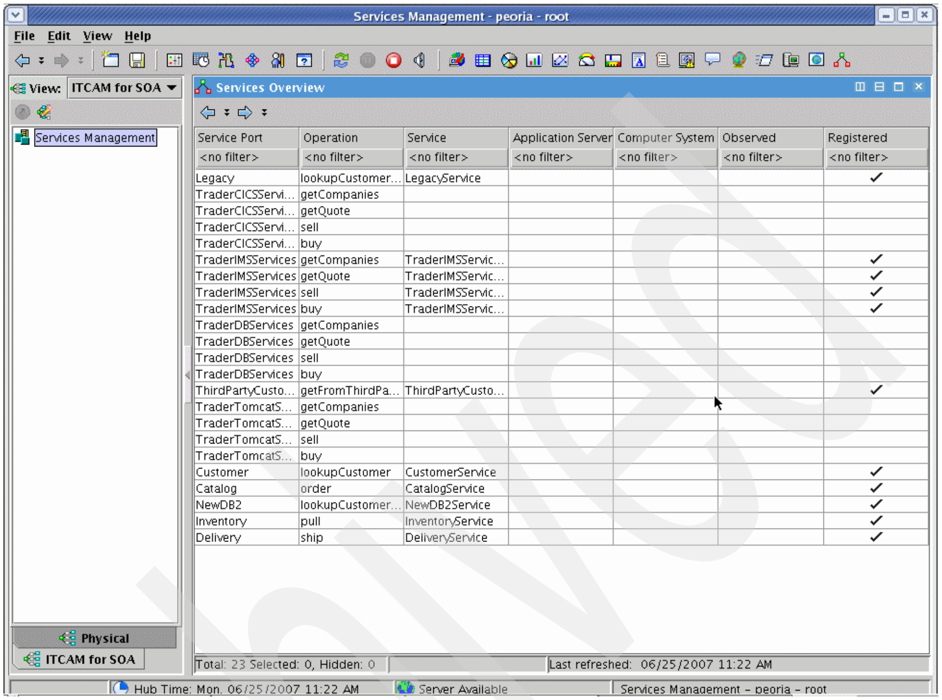


Figure 9-53 Services Management: Services Overview

3. From the Services Overview table, the data from the WebSphere Service Registry and Repository are indicated with a check box in the Registered column. Right-click an entry and select **View Service Detail**. The topology of the Web service is shown in Figure 9-54. The Topology view discovers and displays the service, service port, and the operations provided by the service.

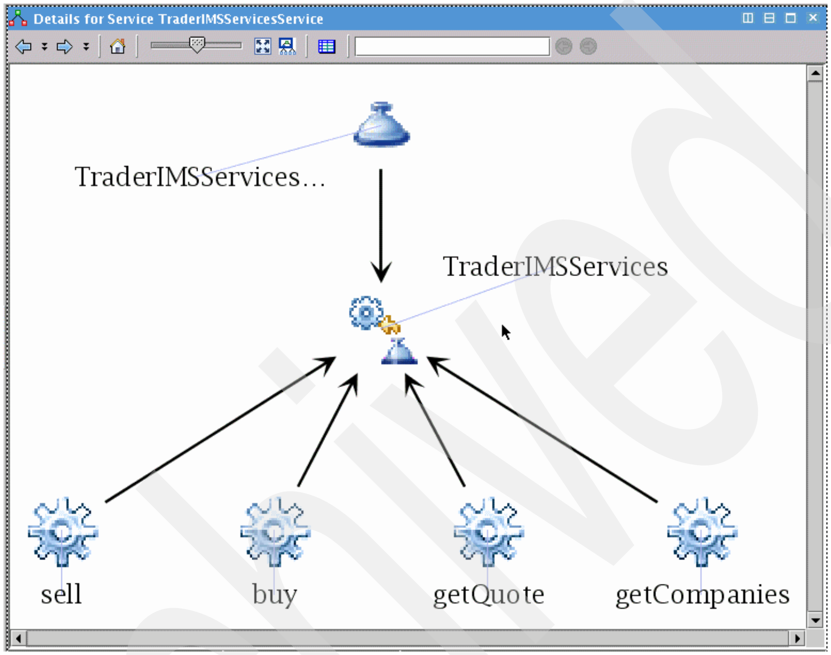


Figure 9-54 Topology view of the service selected

4. If you hover the mouse over a object icon, that object's parameters are displayed (Figure 9-55).

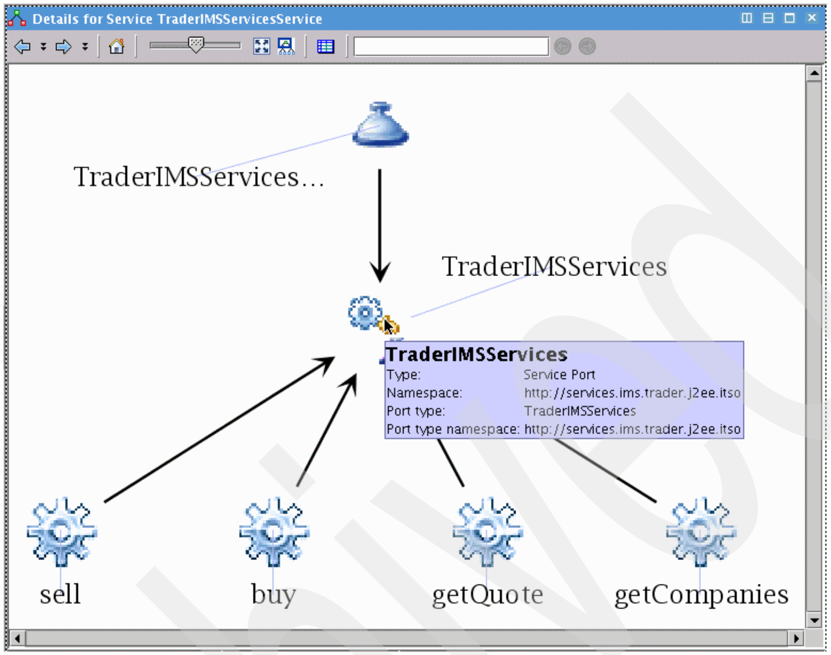


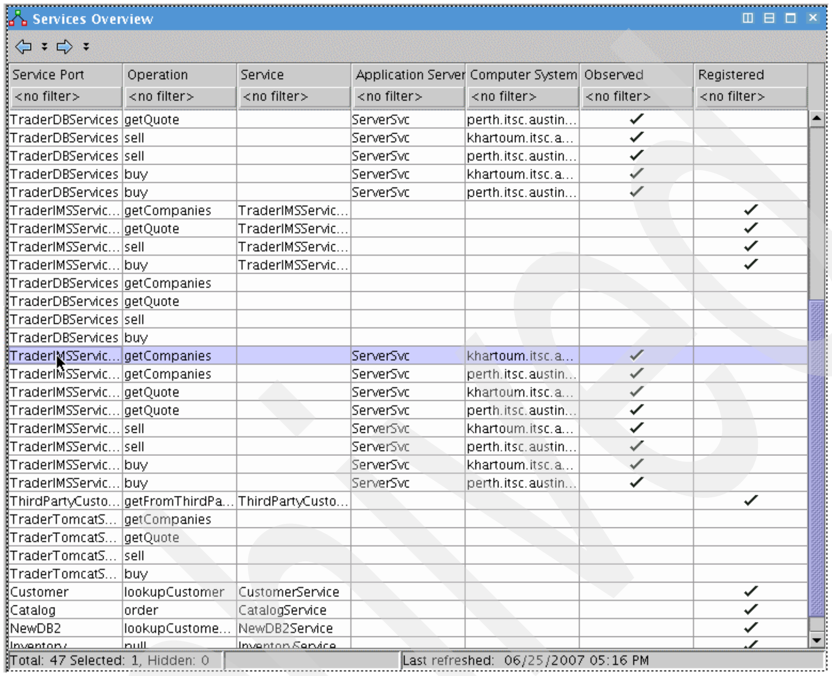
Figure 9-55 Mouse over to view the object properties

9.4.6 ITCAM for SOA Discovery Library Adapter

ITCAM for SOA Discovery Library should be installed on the machine where ITCAM for SOA is running. Extract the file ITCAMSOA61_DLA.zip from the ITCAM for SOA installation disk under KD4/DLA. We extract this file to /opt/IBM/ITCAMSOA61_DLA.

We edit KD4DiscoveryLibraryAdapter.properties in a similar in which we edited the properties file used in 9.4.4, "Discovery Library Adapter" on page 511.

Figure 9-56 displays the data discovered by the ITCAM for SOA Discovery Library Adapter. The table displays both the Web services registered with the WebSphere Service Registry and Repository and the Web services that are being observed by ITCAM for SOA.


Services Overview

Service Port	Operation	Service	Application Server	Computer System	Observed	Registered
<no filter>	<no filter>	<no filter>	<no filter>	<no filter>	<no filter>	<no filter>
TraderDBServices	getQuote		ServerSvc	perth.itsc.austin...	✓	
TraderDBServices	sell		ServerSvc	khartoum.itsc.a...	✓	
TraderDBServices	sell		ServerSvc	perth.itsc.austin...	✓	
TraderDBServices	buy		ServerSvc	khartoum.itsc.a...	✓	
TraderDBServices	buy		ServerSvc	perth.itsc.austin...	✓	
TraderIMSServic...	getCompanies	TraderIMSServic...				✓
TraderIMSServic...	getQuote	TraderIMSServic...				✓
TraderIMSServic...	sell	TraderIMSServic...				✓
TraderIMSServic...	buy	TraderIMSServic...				✓
TraderDBServices	getCompanies					
TraderDBServices	getQuote					
TraderDBServices	sell					
TraderDBServices	buy					
TraderIMSServic...	getCompanies		ServerSvc	khartoum.itsc.a...	✓	
TraderIMSServic...	getCompanies		ServerSvc	perth.itsc.austin...	✓	
TraderIMSServic...	getQuote		ServerSvc	khartoum.itsc.a...	✓	
TraderIMSServic...	getQuote		ServerSvc	perth.itsc.austin...	✓	
TraderIMSServic...	sell		ServerSvc	khartoum.itsc.a...	✓	
TraderIMSServic...	sell		ServerSvc	perth.itsc.austin...	✓	
TraderIMSServic...	buy		ServerSvc	khartoum.itsc.a...	✓	
TraderIMSServic...	buy		ServerSvc	perth.itsc.austin...	✓	
ThirdPartyCusto...	getFromThirdPa...	ThirdPartyCusto...				✓
TraderTomcatS...	getCompanies					
TraderTomcatS...	getQuote					
TraderTomcatS...	sell					
TraderTomcatS...	buy					
Customer	lookupCustomer	CustomerService				✓
Catalog	order	CatalogService				✓
NewDB2	lookupCustome...	NewDB2Service				✓
Inventory	null	InventoryService				✓

Total: 47 Selected: 1, Hidden: 0 Last refreshed: 06/25/2007 05:16 PM

Figure 9-56 Services overview with observed and registered Web services

Select **Service Detail** from an observed service to get the detail view of the service, as shown in Figure 9-57. The service port topology view displays the service port details discovered by the Discover Library Adapter, the service operations, and the application servers that are providing the service.

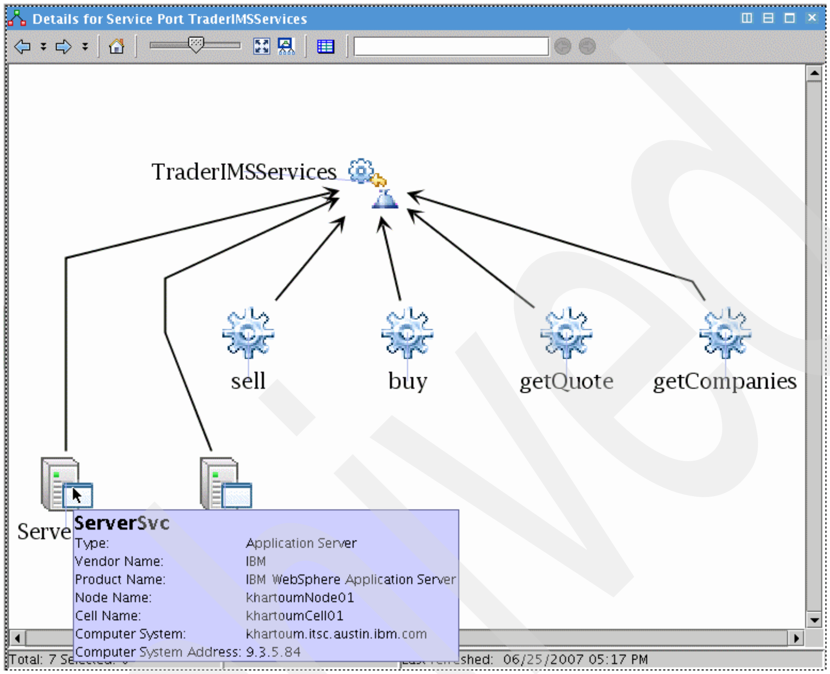


Figure 9-57 Topology view showing the servers providing the service

Hover the mouse over an object in the topology view to view the properties associated with that object as discovered by the Discovery Library Adapter. Figure 9-57 displays the properties associated with one of the application servers providing the trader IMS Service.

10

Implementation of ITCAM products on z/OS

This chapter describes the implementation specifics for z/OS of IBM Tivoli Composite Application Manager. The z/OS implementation tasks are typically performed separately from the distributed components. Therefore, this chapter is for the z/OS implementer. The discussion includes:

10.1 Implementation overview

This chapter contains the overall implementation procedure for all IBM Tivoli Composite Application Manager family products for z/OS.

This section provides an overview of the components of ITCAM products that are installed in z/OS. Figure 10-1 shows the complete configuration of agents and components in each subsystem and started task.

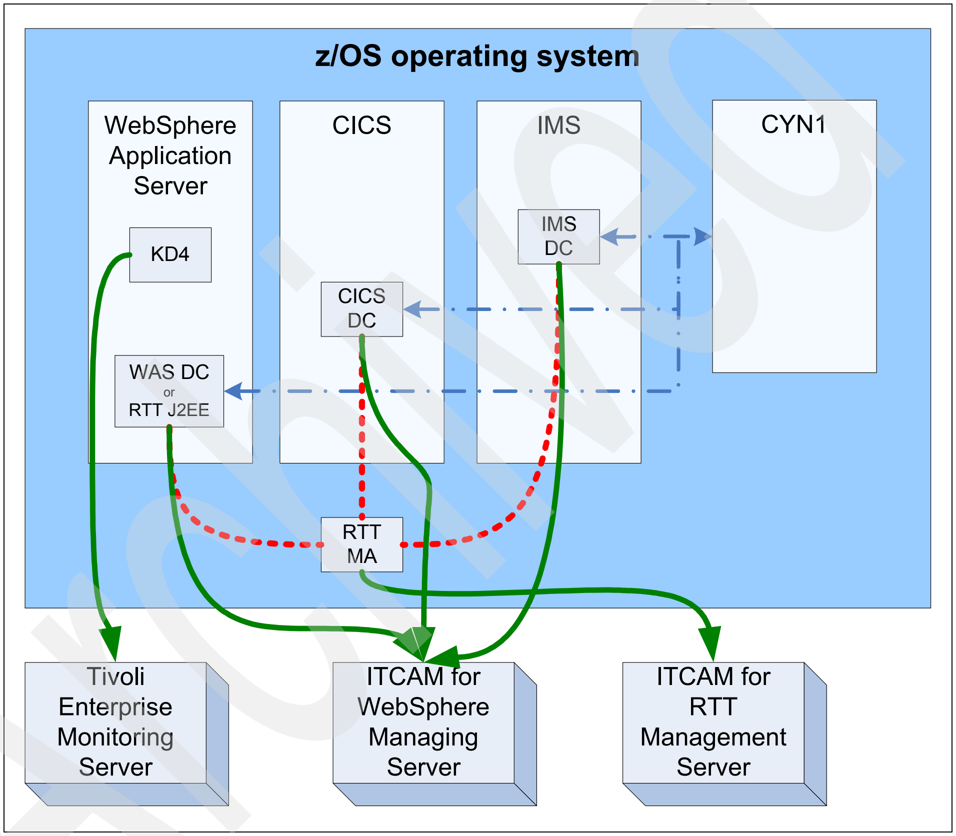


Figure 10-1 z/OS overview

From Figure 10-1, the interaction agents are as follows:

- The CYN1 subsystem runs to collect information related to Workload Manager (WLM) and the System Measurement Facility (SMF). ITCAM for WebSphere, ITCAM for CICS Transactions, and ITCAM for IMS Transactions data collectors retrieve information from the CYN1 subsystem. We discuss this implementation process in 10.2, “The CYN1 subsystem” on page 523.

- The WebSphere data collector from ITCAM for WebSphere provides the appropriate mechanism to collect WebSphere Application Server on z/OS transaction performance data. We discuss the WebSphere Application Server data collector in 10.3, “ITCAM for WebSphere data collector” on page 526.
- ITCAM for Response Time Tracking provides a management agent for z/OS. This management agent primarily resides in UNIX System Services. The agent consists of the ARM agent and the management agent process, similar to the distributed implementation. WebSphere monitoring is acquired from the J2EE monitoring component. We provide installation and configuration information in 10.4, “ITCAM for Response Time Tracking agent” on page 537.
- ITCAM for CICS Transactions and ITCAM for IMS Transactions provide data collector mechanisms for both ITCAM for Response Time Tracking and ITCAM for WebSphere. They communicate directly to the ITCAM for WebSphere managing server, while for ITCAM for Response Time Tracking, they feed the management agent for performance data. We discuss ITCAM for CICS Transactions and ITCAM for IMS Transactions in 10.5, “ITCAM for CICS Transactions” on page 550, and 10.6, “ITCAM for IMS Transactions” on page 554.
- The ITCAM for SOA data collector, KD4, resides in the WebSphere Application Server JAX-RPC handler. It collects Web services performance and statistics information. We discuss this in 10.7, “ITCAM for SOA management agent for z/OS” on page 558.

10.2 The CYN1 subsystem

The CYN1 subsystem can be used for ITCAM for WebSphere, ITCAM for CICS Transactions, and ITCAM for IMS Transactions on z/OS. It provides the interface to SMF and z/OS-related information, such as CPU and memory usage. The CYN1 subsystem uses address space storage to store SMF information collected from the various WebSphere for z/OS Application Server instances running in the same LPAR. SMF statistics consist of server and container data, bean data, bean method data, Web application data, and servlet data. This data is collected from SMF record 120 and resides in the CYN1 address space according to a hierarchical structure with the server instance as the top level. Memory for such a structure is pre-allocated by the CYN1 address space when it comes up according to a predefined parameter template, which specifies the maximum number of server instances and the sizes of the bean section, bean method section, Web application section, and servlet section pools.

The ITCAM for CICS Transactions, ITCAM for IMS Transactions, and ITCAM for WebSphere data collector on z/OS use for each runtime instance a unique composite identification token for tracking transaction correlation. The CYN1 subsystem is useful for collecting additional information related to Workload Manager and System Measurement Facility. Therefore, implement the CYN1 subsystem for z/OS systems running a data collector, as it provides a composite application token for the z/OS application.

To define the CYN1 subsystem on z/OS, you must perform several tasks, which we describe in the following sections:

- 10.2.1, “CYN1 subsystem definition” on page 524
- 10.2.2, “Assigning a RACF user ID for CYN1PROC” on page 524
- 10.2.3, “Started task creation” on page 525
- 10.2.4, “Authorizing SCYNAUTH” on page 525
- 10.2.5, “SMF customization” on page 525

10.2.1 CYN1 subsystem definition

Provide the subsystem definition by updating SYS1.PARMLIB(IEFSSNxx). You must include the following statement:

```
SUBSYS SUBNAME(CYN1)
```

The update in IEFSSNxx takes effect with the next IPL of the system and is permanent. You can dynamically add a subsystem using this console command:

```
SETSSI ADD,SUBNAME=CYN1
```

10.2.2 Assigning a RACF user ID for CYN1PROC

Installations have a significant number of options to choose to enable started tasks and authorize them with RACF®. A started task must have a RACF association with a user ID or a group in the started task table ICHRIN03 or using the STARTED class. To associate the CYN1 subsystem procedure (CYN1PROC) to the appropriate user ID STC, you can issue the following RACF command:

```
RDEF STARTED (CYN1*.*) STDATA(USER(STCRACF))
```

Because the STARTED class is typically cached using the RACLIST option, you might have to refresh the cached started task information using the command:

```
SETROPTS REFRESH RACLIST(STARTED)
```

The user ID associated with the started tasks must have RACF access to the ITCAM for WebSphere libraries.

10.2.3 Started task creation

Copy the member CYN1PROC from the library data set SCYNPROC into the system procedure library, typically SYS1.PROCLIB. You can apply this member to any data set listed in the PROC00 DD statement of the JES2 started task.

10.2.4 Authorizing SCYNAUTH

The SCYNAUTH data set must be APF authorized and concatenated to the system link list. A link list is a cached system search area for executable modules on z/OS.

To authorize a library, you can update either the IEAAPFxx or the PROGxx data set according to your installation requirements. You can also dynamically assign a data set to be APF authorized using a console command similar to this:

```
SETPROG APF,ADD,DSN=ITCAM.V6R1.SCYNAUTH,VOL=ITCAM1
```

To add SCYNAUTH to the link list, you can update the LNKLSTxx or PROGxx and include the data set. You can also issue the following commands to dynamically add the data sets to the link list concatenation:

```
SETPROG LNKLST,DEFINE,NAME=ITCAMLST,COPYFROM=CURRENT
SETPROG LNKLST,ADD,NAME=ITCAMLST,DSN=ITCAM.V6R1.SCYNAUTH
SETPROG LNKLST,ACTIVATE,NAME=ITCAMLST
```

10.2.5 SMF customization

The SMF exit IEFU83 of ITCAM for WebSphere must be installed as a system exit module. This can be placed in the PROGxx member using the definition shown in Example 10-1.

Example 10-1 Defining IEFU83

```
EXIT ADD EXITNAME(SYS.IEFU83)
   MODNAME(CYN1FU83)
   DSNAME(ITCAM.V6R1.SCYNAUTH)
   FIRST
EXIT ADD EXITNAME(SYSSTC.IEFU83)
   MODNAME(CYN1FU83)
   DSNAME(ITCAM.V6R1.SCYNAUTH)
   FIRST
```

These can also be performed dynamically using console commands:

```
SETPROG EXIT,ADD,EX=SYS.IEFU83,MOD=CYN1FU83,DSN=ITCAM.V6R1.SCYNAUTH,FIRST
SETPROG EXIT,ADD,EX=SYSSTC.IEFU83,MOD=CYN1FU83,DSN=ITCAM.V6R1.SCYNAUTH,FIRST
```

The IEFU83 exit must be then associated with the SMF record 120 and record it. This is performed from the SYS1.PARMLIB(SMFPRMxx). Example 10-2 shows an example of an entry to enable an IEFU83 exit and recording of SMF type 120.

Note: The SMF exit definition shown in Example 10-2 on page 526 is the minimum definition. Your installation most likely will have much more of the SMF record type being captured and more exits being defined. Consult the SMF documentation for details about how to update the SMF parameter.

Example 10-2 Defining SMF exit

```
SYS(TYPE(120), EXITS(IEFU83))
SUBSYS(STC,EXITS(IEFU83))
```

10.3 ITCAM for WebSphere data collector

This section describes the installation and configuration of the ITCAM for WebSphere data collector for WebSphere Application Server on z/OS. The steps documented in this section take place after the System Modification Program/Extended (SMP/E) installation. We assume that you finished SMP/E processing for those components successfully.

The installation of ITCAM for WebSphere on z/OS is performed using SMP/E with FMID HAAD610. The following UNIX System Services file systems need to be available:

- /usr/lpp/itcam (main product file system, SMP/E maintained)
- /etc/ibm/tivoli/cfg/ (configuration files)
- /var/itcam61 (runtime configuration directory)
- /var/ibm/tivoli/common/CYN (Tivoli common log files)

We configured and customized the ITCAM for WebSphere data collector on WebSphere Application Server on z/OS based on the instructions in *IBM Tivoli Composite Application Manager for WebSphere Installation and Customization Guide*, GC32-9506.

The implementation tasks includes:

1. Define the CYN1 subsystem to z/OS, which is an optional task described in 10.2, “The CYN1 subsystem” on page 523.
2. Generate the runtime environment in UNIX System Services. You need to modify some files here to further tailor the data collector execution. See 10.3.1, “Modifying WebSphere started tasks” on page 527.

3. Modify WebSphere Application Server parameters using the administrative console in case of failures. See 10.3.2, “Defining the data collector to WebSphere Application Server” on page 527.

If you have any problems, see 10.3.3, “Problem determination and log files” on page 533.

10.3.1 Modifying WebSphere started tasks

The WebSphere Application Server on z/OS consists of multiple address spaces, including:

- WebSphere daemon address space
- WebSphere manager address space
- Control address space
- Servant address space

Each application server environment typically consists of a control address space and a dynamic number of servant address spaces that can be started on demand as needed. ITCAM for WebSphere is implemented in each application server environment. It requires modification to the servant address space by adding the statement HEAP(,,,FREE) in the started task, as shown for our environment in Example 10-3.

Example 10-3 Adding the HEAP parameter for servant address space

```
//BBO5ASR PROC ENV = ADCDPL.P390.SERVER1,Z=BBO5ASRZ
// SET ROOT='/wasitconfig/itcell/itnodea'
//BBOSR EXEC PGM=BBOSR,REGION=OM,TIME=NOLIMIT,
//PARM='TRAP(ON,NOSPIE),HEAP(,,,FREE),ENVAR("_EDC_UMASK_DFLT=007")/"
//BBOENV DD PATH='&ROOT/&ENV/was.env'
// INCLUDE MEMBER=&Z
```

10.3.2 Defining the data collector to WebSphere Application Server

The data collector is installed as a WebSphere Application Server custom service. It is started in the Java Virtual Machine (JVM).

You might have multiple distinct server instances under a common node, or you might have multiple cells in a single z/OS image, or you might share the binary for all the images in the SYSPLEX. In Version 6.1, all runtime configurations are placed in a user-maintained directory, such as /var/itcam61. Multiple application server instances create different subdirectories in this path.

All ITCAM V6.1 data collectors can share the same configuration home.

- Run the createcfg.sh script in the data collector base home directory /usr/lpp/itcam/WebSphere/DC/bin. The script creates a runtime configuration directory, separating your typical SMP/E-maintained libraries. You only need to run this script once. Our runtime environment is created in the directory /var/itcam61/websphere/DC/runtime.
- Generate runtime environments in UNIX System Services. ITCAM for WebSphere generates a runtime directory based on the WebSphere Application Server name. This can be performed using the **setupwas.sh** command. Example 10-4 shows our setup. This script is in the data collector base home directory /usr/lpp/itcam/WebSphere/DC/bin. The configuration program automatically updates datacollector.properties and environment files.

  The script optionally integrates the support for WebSphere MQ and IMS. If requested, the appropriate resource adapters must already be installed in the WebSphere Application Server environment. In our test case, we do not use them.

Example 10-4 Running setupwas.sh

```
TIVO01:/usr/lpp/itcam/WebSphere/DC/bin: >./setupwas.sh

ITCAM 6.1 for WebSphere Application Server
   Data Collector Configuration

Enter the path of the WAS user install root [/u/WAS510]:
/SC67/wasitconfig/itcell/itnodea/AppServer/bin
Searching for wsadmin.sh. Please wait...
Found wsadmin.sh in the following locations:
   1) /SC67/wasitconfig/itcell/itnodea/AppServer/bin/wsadmin.sh

Using wsadmin: /SC67/wasitconfig/itcell/itnodea/AppServer/bin/wsadmin.sh
Checking Websphere Application Server version...
The history xml file does not exist.
Found WAS Version 6.1.0.2 ND
Searching for Application Servers
This may take some time. Please wait...
WASX7209I: Connected to process 'dmgr' on node itdmnode using SOAP connector;
The type of process is: Deployment Manager

WASX7303I: The following options are passed to the scripting environment and
are available as arguments that are stored in the argv variable:
'[/tmp/am_setupwas67961453.a]'
Found the following servers:
```

```
1)itsr01a(cells/itcell/nodes/itnodea/servers/itsr01a|server.xml#Server_11613722
25374)

ITCAM configuration /usr/lpp/itcam/WebSphere/DC/runtime/was61.itnodea.itsr01a
already exists. Overwrite? (y|n):
y
+-----------------------------------------------------------------+
+Which data collection agents(s) would you like to configure?
+Select all that apply.
+-----------------------------------------------------------------+


+-----------------------------------------------------------------+
|Data collection for ITCAM for WebSphere and J2EE's Tivoli Enterprise
|Portal Interface requires a separate installation of ITCAM
|for WebSphere's Tivoli Enterprise Monitoring Agents
+-----------------------------------------------------------------+
Configure data collection for ITCAM for WebSphere and J2EE's
Tivoli Enterprise Portal Interface?(y|n):
y
Enter the IP address of the TEMA agent[127.0.0.1]:
127.0.0.1
Enter the port number of the TEMA agent[63335]:
63335
+-----------------------------------------------------------------+
|Data collection for ITCAM for WebSphere and J2EE's Application Monitor
|Interface requires a separate installation of ITCAM for
|WebSphere's Managing Server.
+-----------------------------------------------------------------+
Configure data collection for ITCAM for WebSphere and J2EE Application Monitor
Interface? (y|n):
y
Enter the IP address of the ITCAM Managing Server:
peoria.itsc.austin.ibm.com
Enter ITCAM Managing Server install directory: [/opt/IBM/itcam/WebSphere/MS]:
/opt/IBM/itcam/WebSphere/MS
Enter the MS Codebase Port [9122]:
9122
Do you wish to configure data collection for IMS Connect for Java? (y|n):
n
Do you wish to configure data collection for WebSphere MQ JMS Provider? (y|n):
n
Setup will create an ITCAM runtime with the following parameters:

   1) WAS server name: itsr01a
   2) ITCAM for Tivoli Enterprise Portal Interface: Yes
      TEMA Host        : 127.0.0.1
      TEMA Port        : 63335
      Monitor GC       :
   3) ITCAM for Tivoli Application Monitor Interface: Yes
```

```
        ITCAM MS host   : peoria.itsc.austin.ibm.com
        ITCAM MS home   : /opt/IBM/itcam/WebSphere/MS
        Codebase port   :9122
        Collect SMF data: No
     4) IMS Connect for Java support    :No
     5) WebSphere MQ                    :No

Enter item number to modify, 'y' to accept, or 'n' to cancel:
y
ITCAM configuration for itsr01a created in
/usr/lpp/itcam/WebSphere/DC/runtime/was61.itnodea.itsr01a
Creating plugin directory
/SC67/wasitconfig/itcell/itnodea/AppServer/plugins/itcam_6.1.0
Creating plugin directory
/SC67/wasitconfig/itcell/itnodea/AppServer/plugins/itcamsib_6.1.0
Creating plugin directory
/SC67/wasitconfig/itcell/itnodea/AppServer/plugins/itcammqjms_6.1.0
Reinitializing plugins...
Configure WebSphere Application Server itsr01a? (y|n):
y
Configuring Application Server. This may take some time. please wait...

WASX7209I: Connected to process 'dmgr' on node itdmnode using SOAP connector;
The type of process is: DeploymentManager
WASX7303I: The following options are passed to the scripting environment and
are available as arguments that are stored in the argv variable
:'[/usr/lpp/itcam/WebSphere/DC/runtime/was61.itnodea.itsr01a/input.txt]'
server.platform=z/OS
server.id=itsr01a(cells/itcell/nodes/itnodea/servers/itsr01a|server.xml#Server_
1161372225374)

   .  .  .

Server major version is 61.
INFO: Configuring for z/OS
INFO:  Configuring AdminService

   .  .  .

INFO: Start setting attributes for
itsr01a(cells/itcell/nodes/itnodea/servers/itsr01a|server.xml#Server_1161372225
374)
INFO: Configuring JVM Args and SystemProperties
Modify JVM command line arguments

   .  .  .

INFO:  Configuring Environment Variables
```

```
Create/Modify java process environments
create new environment
LIBPATH=/usr/lpp/itcam/WebSphere/DC/runtime/was61.itnodea.itsr01a/lib:${ITCAM61
HOME}/toolkit/lib
create new environment NLSPATH=${ITCAM61HOME}/toolkit/msg/%L/%N.cat
create new environment JITC_COMPILEOPT=NALL
INFO:  Configuring CustomService
ignore externalConfigURL=am
Configure custom service (am)
INFO:  Configuring PmiService
modify PMI service to {enable true} {statisticSet extended}

   .  .  .

INFO: Configuring variable protocol_iiop_propagate_unknown_service_ctxs
create new variable protocol_iiop_propagate_unknown_service_ctxs
INFO: Configuring variable AM_HOME
create new variable AM_HOME
INFO: Configuring variable MS_AM_HOME
create new variable MS_AM_HOME
INFO: Configuring variable ITCAM61HOME
create new variable ITCAM61HOME
INFO: Dsabling SMF data collection
INFO: Configuring Control Region system properties
Create/Modify JVM system properties
newProperty={name protocol_http_timeout_output_recovery} {value SESSION}
create new property protocol_http_timeout_output_recovery
WAS_PROPS_DIR=/wasitconfig/itcell/itdmnode/DeploymentManager/profiles/default/p
roperties
Synchronizing with node itnodea
Synchronization completed successfully on itnodea
Successfully configured data collector for server
itsr01a(cells/itcell/nodes/itnodea/servers/itsr01a|server.xml#Server_1161372225
374)
```

The setupwas.sh script defines a separate directory structure beneath the runtime configuration path. Our servant application server is called itsr01a. Therefore, the runtime environment is created under /var/itcam61/websphere/DC/runtime/was61.itnodea.itsr01a, as shown in Figure 10-2.

```
TIVO01:/SC67/var/itcam61/websphere/DC/runtime/was61.itnodea.itsr01a: >ls -l
total 2208
drwxr-xr-x 2 ASSR1 WSSR1 Oct 23 custom
-rw-r--r-- 1 ASSR1 WSSR1 Oct 23 cyn-cclog.properties
-rwxr-xr-x 1 ASSR1 WSSR1 Oct 23 cynlogging.properties
-rw-r--r-- 1 ASSR1 WSSR1 Oct 23 datacollector.policy
-rw-r--r-- 1 ASSR1 WSSR1 Oct 23 datacollector.policy.asc
-rw-r----- 1 ASSR1 WSSR1 Oct 24 datacollector.properties
-rw-r--r-- 1 ASSR1 WSSR1 Oct 23 input.properties
-rw-r--r-- 1 ASSR1 WSSR1 Oct 23 input.txt
-rw-rw---- 1 ASSR1 WSSR1 Nov  1 itnodea.itsr01a.0.gls.properties
-rw-rw---- 1 ASSR1 WSSR1 Nov 1 itnodea.itsr01a.0.parser.xml
-rw-r----- 1 ASSR1 WSSR1 Oct 24 itnodea.itsr01a.cyaneaGpsCounter.txt
-rw-r----- 1 ASSR1 WSSR1 Oct 23 jiti.properties
drwxr-xr-x 2 ASSR1 WSSR1 Oct 23 lib
drwxr-xr-x 2 ASSR1 WSSR1 Oct 23 logs
-rw-r----- 1 ASSR1 WSSR1 Nov  1 was61.itnodea.itsr01a.classinfo.data
-rw-r----- 1 ASSR1 WSSR1 Oct 24
was61.itnodea.itsr01a.datacollector.properties
-rw-rw---- 1 ASSR1 WSSR1 Nov  1 was61.itnodea.itsr01a.id
-rw-r----- 1 ASSR1 WSSR1 Oct 24 was61.itnodea.itsr01a.kwjdc.properties
-rw-r----- 1 ASSR1 WSSR1 Nov  1 was61.itnodea.itsr01a.requestaggregator.xml
-rw-r----- 1 ASSR1 WSSR1 Nov  1 was61.itnodea.itsr01a.toolkit.properties
-rw-r----- 1 ASSR1 WSSR1 Nov  1 was61.itnodea.itsr01a.toolkit.xml
```

Figure 10-2 J2EE data collector runtime configuration directory

When the setup and configuration is done, you must restart the application server servant started tasks. The data collector then appears in the Unconfigured Data Collectors in the ITCAM for WebSphere Web console, as displayed in Figure 10-3.

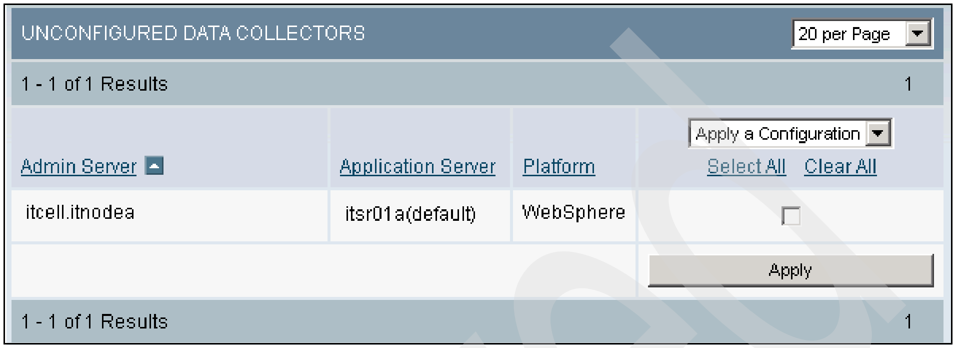


Figure 10-3 ITCAM for WebSphere Unconfigured Data Collectors

Further configuration and usage is similar to non-z/OS WebSphere Application Server steps, as discussed in Chapter 2, “ITCAM for WebSphere and ITCAM for J2EE concepts and installation” on page 15, and Chapter 3, “ITCAM for WebSphere and ITCAM for J2EE usage” on page 65.

10.3.3 Problem determination and log files

Apart from the main runtime directory, ITCAM for WebSphere uses the Tivoli common directory to store its logs. You must allocate a generous amount of space for this common directory because it potentially grows quite large for problem determination. We create a separate Hierarchical File System (HFS) under /var/ibm with the primary allocation size of 500 MB and 100 MB secondary. This Tivoli common directory is used to store logs and First Failure Data Collector (FFDC) information.

For problem determination, there are several locations to check:

- The datacollector.properties file for the application server itsr01a provides the necessary configuration data for the connection to the management server. For problem determination, check the settings in datacollector.properties. Remember that there is a datacollector.properties prefixed by the WebSphere instance name.
- Because most of the WebSphere Application Server configuration is performed automatically using setupwas.sh, perform some verification from

the administrative console to ensure that all updates are in the WebSphere environment. These updates include:

- Definition of the WebSphere Application Server JVM parameters
- Definition of the custom service itself

WebSphere Application Server JVM parameters

JVM parameters must be defined for the servant region. From the administrative console menu, select **Servers** → **Application Servers** and select the server on which you are installing the data collector. Note the following values:

- Server Infrastructure: **Java and Process Management** → **Process Definition: Servant**

  Additional properties: Java Virtual Machine:

  - Boot classpath: ${ITCAM61HOME}/toolkit/lib/bcm-bootstrap.jar: $%ITCAM61HOME}/itcamdc/lib/ppe.probe-bootstrap.jar
  - Generic JVM arguments:
    -Djlog.propertyFileDir.CYN=${ITCAM61HOME}/toolkit /etc
    -Dcom.ibm.tivoli.jiti.config=${ITCAM61HOME}/toolkit/etc/config.properties
  - Initial heap size: 256
  - Maximum heap size: At least 512

- Custom properties for the servant process

  For the custom service, some customer properties are required (Figure 10-4 on page 535):

  - java.security.policy:

    ```
    /usr/lpp/itcam/WebSphere/DC/runtime/was61.itnodea.itsr01a/datacol
    lector.policy
    ```

  - am.home:

    ```
    /usr/lpp/itcam/WebSphere/DC/itcamdc
    ```

  - com.ibm.tivoli.jiti.injector.ProbeInjectorManagerChain.primaryInjectorFile:

    ```
    /usr/lpp/itcam/WebSphere/DC/runtime/was61.itnodea.itsr01a/jiti.pr
    operties
    ```

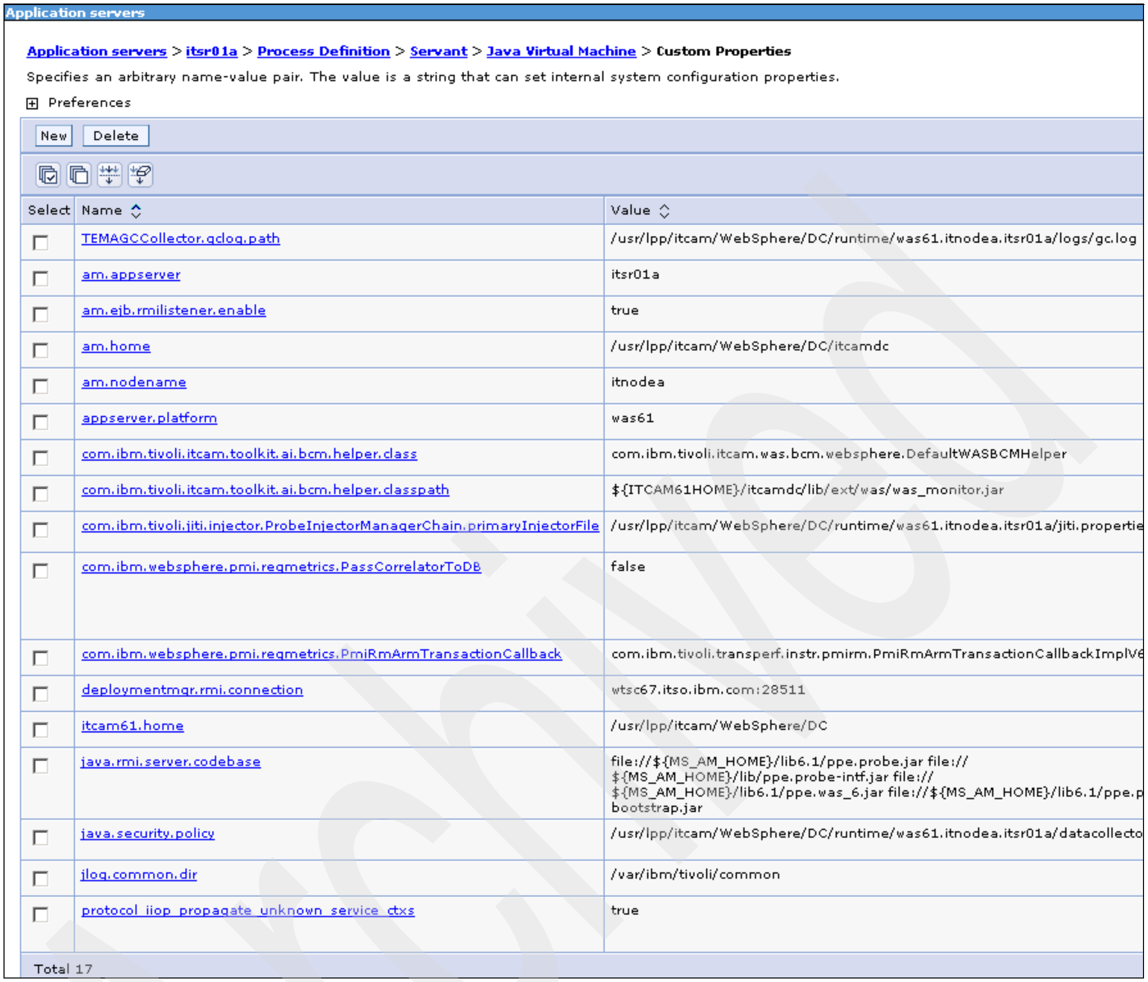


Figure 10-4 Custom properties for servant address space JVM

- Environment entry for the servant address space: LIBPATH:

  `/var/itcam61/WebSphere/DC/runtime/was61.itnodea.itsr01a/lib:${ITCAM61HOME}/toolkit/lib`

- Message catalog for the servant process: NLSPATH:

  `/usr/lib/nls/msg/%L/%N:${ITCAM61HOME}/toolkit/msg/%L/%N.cat`

Probe services definition

From the administrative console menu, select **Servers** → **Application Servers** and choose the server on which you are installing the data collector. From **Server Infrastructure** → **Administration**, select **Custom Services.** One entry for the ITCAM for WebSphere data collector must exist. Otherwise, click **New** to create a new custom service. The new service has the display name of *am*, as shown in Figure 10-5.

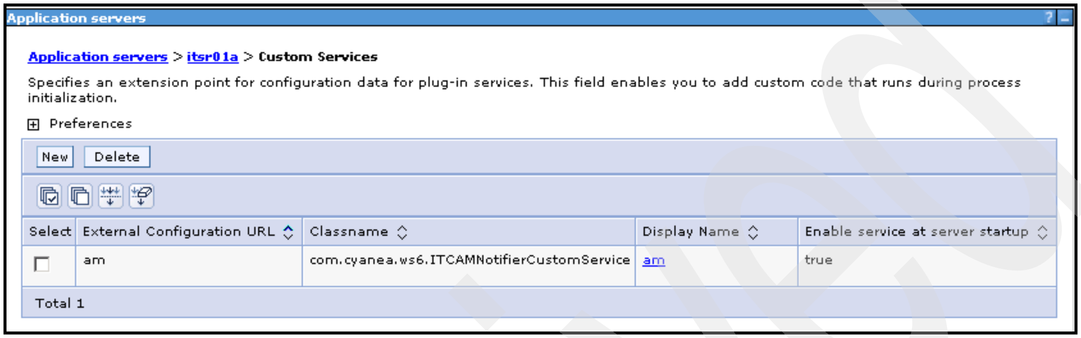


Figure 10-5 Custom services list

For the custom service values:

- Enable service at server startup: Activated
- Classname: com.cynea.ws6.ITCAMNotifierCustomService
- Display name: am
- Description: Probe Service
- Classpath: ${ITCAM61HOME}/itcamdc/lib/ppe.probe-bootstrap.jar

Figure 10-6 shows a configuration example.

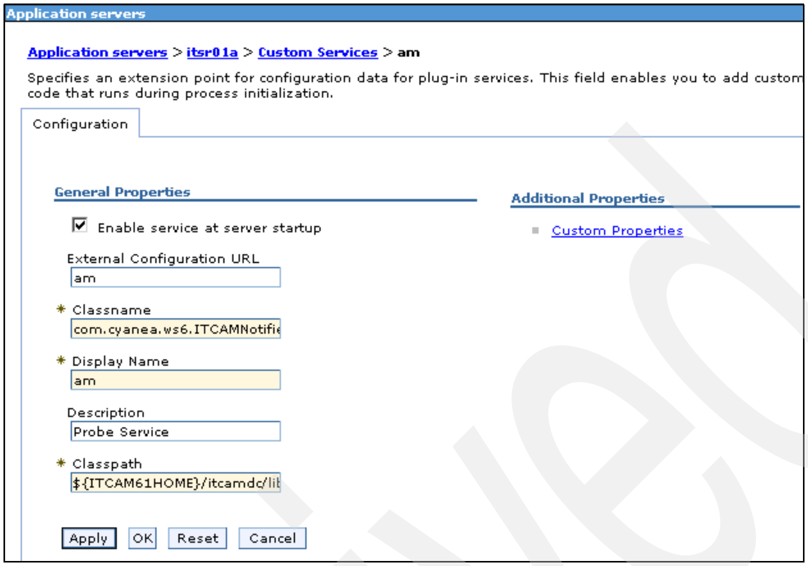


Figure 10-6 Probe service setting

Save the settings by clicking the **Save** link and **Save to master configuration** with the **Synchronize with node** option selected.

10.4 ITCAM for Response Time Tracking agent

This section describes how to install and configure ITCAM for Response Time Tracking agents on z/OS, which can communicate with the ITCAM for CICS Transactions data collector and the ITCAM for IMS Transactions data collector. We discuss some specific implementation issues with those components.

10.4.1 Implementation

The installation for ITCAM for Response Time Tracking on z/OS is performed using SMP/E with FMID HWTP610 and HWTP61A. The following file systems need to be available:

- /usr/lpp/tmtp (main product file system, SMP/E maintained)
- /etc/tmtp (configuration files)
- /var/tmtp (data files)
- /var/ibm/tivoli/common/BWM (Tivoli common log files)

We configured the components based on the instructions in the corresponding guides, including *IBM Tivoli Composite Application Manager for Response Time Tracking Installation and Configuration Guide*, GC32-1907. The general configuration tasks are:

1. Prepare RACF authorizations:
   a. Define the user TMTPAGNT and group TMTPGRP with the necessary permissions:
      - Allow access to UNIX System Services.
      - Specify the JVM maximum address space size on the user TMTPAGNT OMVS segment. Use the RACF ASSIZEMAX configuration parameter. A minimum value of 268435456 bytes is required.
      - Allow access to the group for the WebSphere servant address space.
      - Connect the WebSphere servant user to the TMTPGRP.
      - Allow access to the group for WebSphere runtime configuration files in UNIX System Services.
      - Allow access to the Tivoli configuration file /var/ibm/tivoli/common/cfg.
      - Assign the STARTED profile to start and stop the agent.

   b. Prepare a certificate configuration for SSL communication. In z/OS, certificates are managed from RACF.
2. Generate runtime environments in UNIX System Services:
   – Ensure that /var/tmtp and /etc/tmtp are owned by TMTPAGNT and TMTPGRP with an access mode of 750 or more.
   – Copy prepConfig.sh and setupEnv.sh from /usr/lpp/tmtp/V6R1M0/MA/config1 to a temporary directory, and modify setupEnv.sh to provide BASEDIR, the management agent installation directory, and the JAVA_HOME path, as shown in Example 10-5.

Example 10-5 The setupEnv.sh

```
#!/bin/sh
BASEDIR=/usr/lpp/tmtp/V6R1M0/MA
export BASEDIR
JAVAHOME=/usr/lpp/java/J5.0
export JAVAHOME

# Do not change this
INTERP=zos
export INTERP
```

- Run prepConfig.sh to generate runtime files. This creates a new set of files in /etc/tmtp/MA/config.

– Update zos.properties from /etc/tmtp/MA/config to include the management server parameters. Example 10-6 shows our zos.properties.

Example 10-6 Our zos.properties

```
#
# ITMTP MA for z/OS Configurable Parameters
#
zos.msHostName=khartoum.itsc.austin.ibm.com
zos.msUserName=itcrttadm
zos.msUserPassword=XXXXXXX
zos.sslEnabled=false
zos.msProtocol=http
zos.msPort=9081
zos.maPort=1976
zos.offline=false
zos.proxyProtocol=noproxy
zos.proxyHostName=
zos.proxyPortNumber=
zos.epKeyStore=/etc/tmtp/MA/config/agent.jks
zos.epKeyPass=changeit
# uncomment to set this to full local hostname
# it is discovered automatically if not set
zos.maHostName=wtsc67.itso.ibm.com
```

3. Configure the management agent. From /etc/tmtp/MA/config:

   a. Run configMa.sh install, as shown in Example 10-7.

Example 10-7 Configuring the management agent

```
TIV001:/SC67/etc/tmtp/MA/config # ./configMa.sh install
Retrieving document at
'/usr/lpp/tmtp/V6R1M0/MA/wsdl//EndpointAdmin.wsdl'.
Retrieving document at
'/usr/lpp/tmtp/V6R1M0/MA/wsdl//AgentStatus.wsdl'.
Retrieving document at
'/usr/lpp/tmtp/V6R1M0/MA/wsdl//PolicyManager.wsdl'.
Retrieving document at
'/usr/lpp/tmtp/V6R1M0/MA/wsdl//EventService.wsdl'.
Retrieving document at
'/usr/lpp/tmtp/V6R1M0/MA/wsdl//TransferController.wsdl'.
Retrieving document at '/usr/lpp/tmtp/V6R1M0/MA/wsdl//BDH.wsdl'.
```

```
Retrieving document at
'/usr/lpp/tmtp/V6R1MO/MA/wsdl//ManagementServer.wsdl'.
Retrieving document at
'/usr/lpp/tmtp/V6R1MO/MA/wsdl//EndpointGroup.wsdl'.
Retrieving document at '/usr/lpp/tmtp/V6R1MO/MA/wsdl//Upgrade.wsdl'.
Retrieving document at '/usr/lpp/tmtp/V6R1MO/MA/wsdl//RCA.wsdl'.
Configured the management agent successfully.
If you have a IBM Tivoli Composite Application Manager Data Collector
installed,
 datacollector.properties
needs to be updated for Am.tt.enable key. This key can either be edited
manually
 or
this configuration script can do it for you.
Would you like this script to update the key (y/n) ?
y
```

b. Run createArmLinks.sh, as shown in Example 10-8.

Example 10-8 Creating links for ARM libraries

```
TIVO01:/SC67/etc/tmtp/MA/config # ./createArmLinks.sh
This script will create links for the ARM shared libraries in /usr/lib
directory.
Do you want to continue ? [y/n]
y
A link was found for the common ARM library used by both ITCAMfRTT and eWLM,
/usr/lib/libarm4_31.so. If eWLM is installed on the system, do not replace it,
choose to replace otherwise. Replace ? [y/n]
y
Done. Links were created successfully in /usr/lib directory.
```

c. Start the management agent by running start_tmtpd.sh.

Our completed agent installation is shown in the ITCAM for Response Time Tracking agent list, as shown in Figure 10-7.

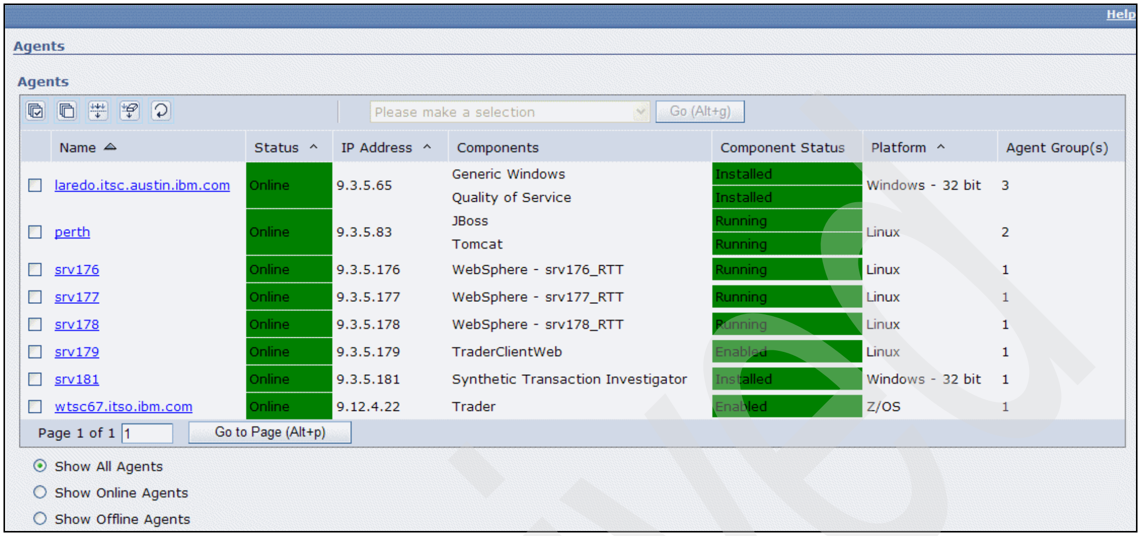


Figure 10-7 All agents installed

10.4.2 Problem determination and log files

The tmtp management agent writes information to the following files:

- Initial startup of the JVM and failure to start the JVM is shown here:
  - /etc/tmtp/MA/config/ma-stdout.log
  - /etc/tmtp/MA/config/ma-stderr.log
- Message and trace files in the Tivoli common directory, typically under /var/ibm/tivoli/common/BWM/logs:
  - msg-ma.log
  - msg-instr.log
  - trace-ma.log
  - trace-instr.log
  - trace-tapmagent.log
  - SystemErr.log
  - SystemOut.log
  - wsadmin.traceout
  - wsadmin.valout
- Check the setting for the variables BASEDIR and JAVAHOME in setupEnv.sh in the configuration directory /etc/tmtp/MA/config/.

- Check the setting for variables zos.msHostName, zos.msUserName, zos.msUserPassword, zos.msProtocol, and zos.msPort in zos.properties in the configuration directory /etc/tmtp/MA/config/.
- Check the NETACCESS profiles in RACF to grant access for the tmtp agent listener port.
- Export the variable DEBUG_TMTP to any value to debug events on the script start_tmtpd.sh.

10.4.3 J2EE monitoring component on z/OS

The J2EE monitoring component is used by ITCAM for Response Time Tracking to collect response time information from a WebSphere Application Server environment. The configuration is done by deploying it using the ITCAM for Response Time Tracking Web console.

> **Important:** Back up the configuration directory of the WebSphere Application Server on which you are deploying the ITCAM for Response Time Tracking J2EE agent. Deployment defines the JVM arguments needed for ITCAM for Response Time Tracking J2EE arguments, which might affect some settings for your other JVMPI agents, such as the ITCAM for WebSphere data collector.

To deploy the J2EE component, select **System Administration → Monitoring Components**. Select **J2EE** from the monitoring components table, and choose **Deploy Monitoring Component** from the menu. Click **Go** (Figure 10-8).

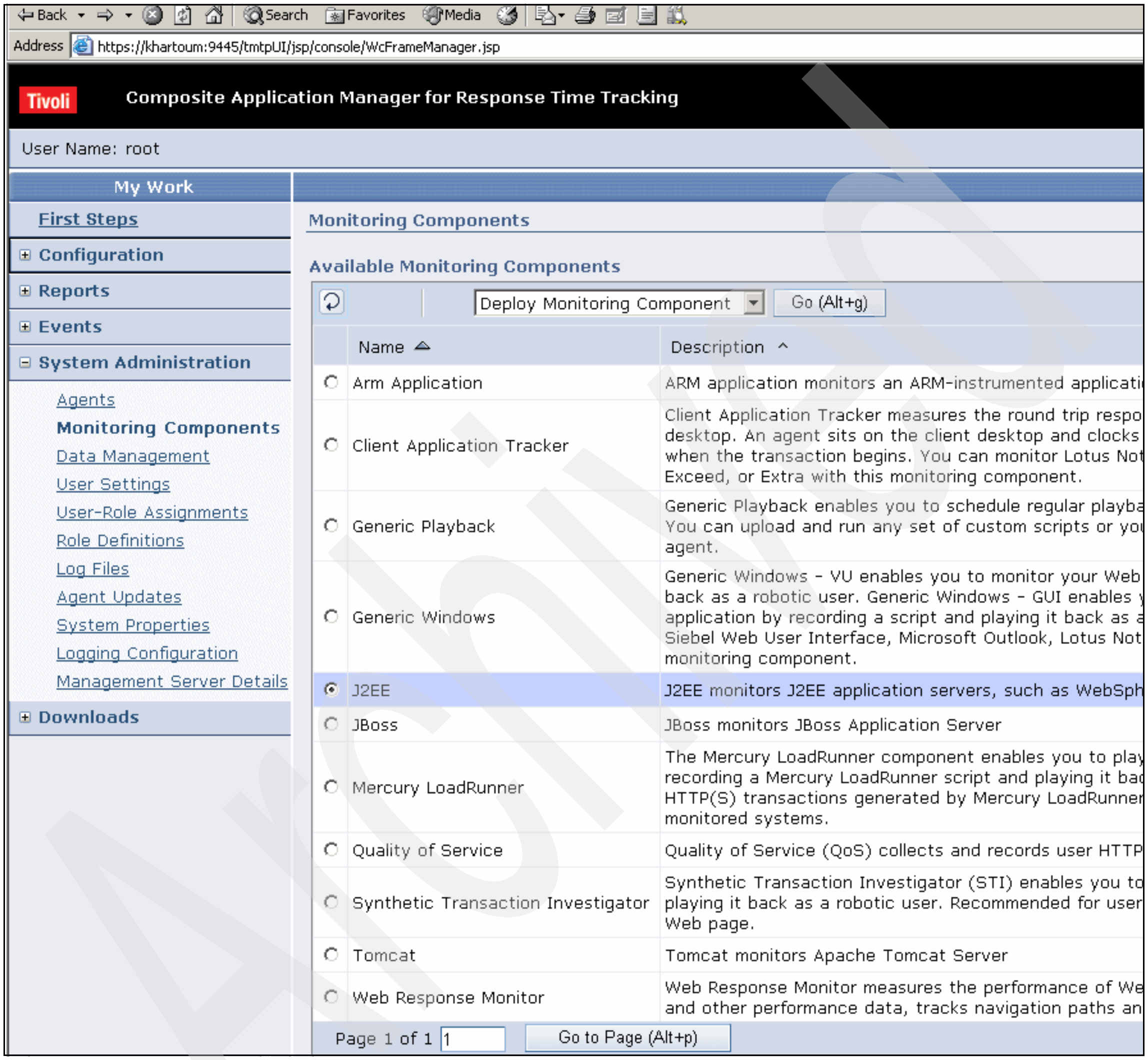


Figure 10-8 Deploy J2EE Agent on z/OS

The deployment process can discover and deploy the J2EE server automatically, or you can specify it manually. See Figure 10-9 to choose the deployment method. We use the discover and deploy method.

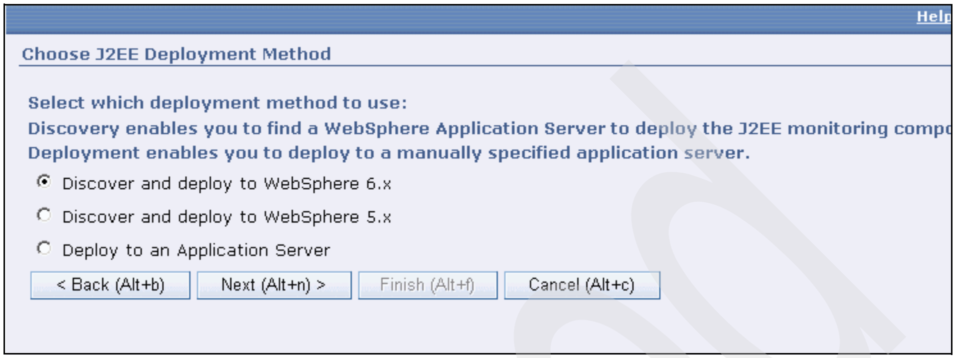


Figure 10-9 Agent deployment: choose J2EE deployment method

You select the management agent that maintains the J2EE server and supplies the application server path, profile name (for WebSphere Application Server V6.x), and the SOAP port of the server. See Figure 10-10 to specify the discovery criteria.

Note: The SOAP port can be found in JESMSGLOG of the WebSphere Control region by searching for SOAP.

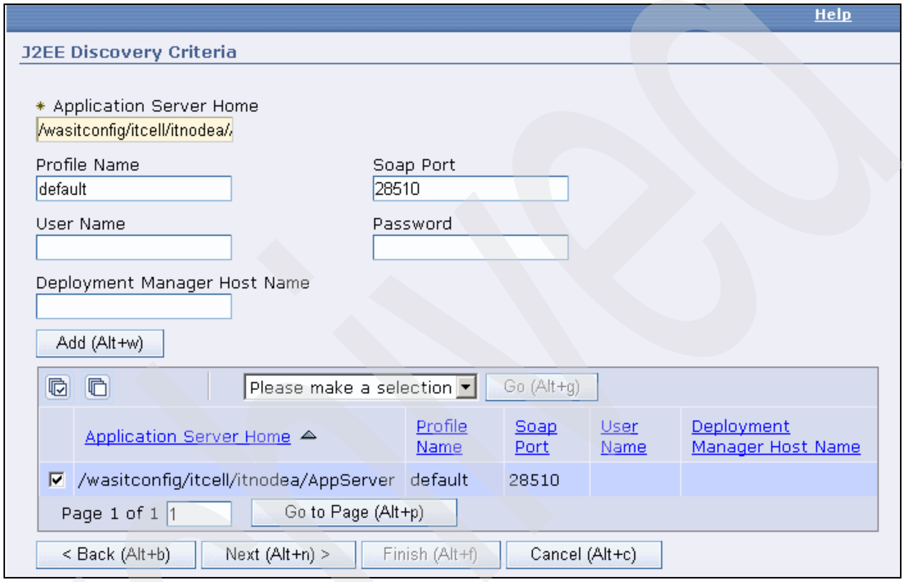


Figure 10-10 Agent deployment: J2EE Discovery Criteria

The J2EE deployment process enables you to pre-discover your instances or just deploy your monitoring component. The deployment generates a runtime environment for every selected server instance. The deployment process creates several directories under /var/tmtp/MA/app/instrument/6101, as shown in Figure 10-11. Subsequent agent deployments create a separate directory in /var/tmtp/MA/app/instrument/6101/appServers/*server_name*.

```
TIVO01:/SC67/var/tmtp/MA/app/instrument/6101: >ls -l
total 88
-rw-rw-r--   1 TMTPAGNT TMTPGRP      100 Oct 31 12:44 TMTPJ2EEJ050300.sys
drwxrwxr-x   3 TMTPAGNT TMTPGRP     8192 Oct 31 12:44 appServers
drwxrwxr-x   2 TMTPAGNT TMTPGRP     8192 Oct 31 14:57 bin
drwxrwxr-x   3 TMTPAGNT TMTPGRP     8192 Oct 31 12:44 cli
drwxrwxr-x   3 TMTPAGNT TMTPGRP     8192 Oct 31 12:44 ic
drwxrwxr-x   5 TMTPAGNT TMTPGRP     8192 Oct 31 12:44 lib
```

Figure 10-11 J2EE component directories

Our configuration parameters for the itsr01a WebSphere Application Server process are in the /var/tmtp/MA/app/instrument/6101/appServers/itsr01a_213/config directory.

Note: If you remove the last J2EE agent from the system, all runtime directories under /var/tmtp/MA/app/instrument/6101 are removed.

Problem executing wsadmin scripts

In this case, check the message and trace log file trace-ma.log in /var/ibm/tivoli/common/BWM/logs.

The J2EE deployment process might not have enough address space for the executing task user for the tmtp management agent defined, as shown in Figure 10-12. In other cases, the process is not be able to communicate with the WebSphere Deployment Server because of improper settings in the scripting language.

```
STDERR from jacl script, /SC67/var/ibm/tivoli/common/BWM/logs/discoverServers.jacl:
STDERR:Error: unable to allocate 268435456 bytes for GC in j9vmem_reserve_memory.
JVMJ9VM015W Initialization error for library j9gc23(2): Failed to instantiate heap.
256M requested
Could not create the Java virtual machine.
```

Figure 10-12 JVM initialization error with tmtp management agent

Ensure the maximum JVM address space size of user TMTPAGNT. Inspect the configuration with the arguments shown in Example 10-9.

Example 10-9 Checking the Java virtual machine setting

```
TMTPAGNT:/SC67/etc/tmtp/MA/config: >/usr/lpp/java/J5.0/bin/java -version -Xmx256m
java version'1.5.0'
Java(TM) 2 Runtime Environment, Standard Edition (build pmz31devifx-20060524 (SR2))
IBM J9 VM (build 2.3, J2RE 1.5.0 IBM J9 2.3 z/OS s390-31 j9vmmz3123-20060505a (JIT
enabled)
J9VM - 20060501_06428_bHdSMr
JIT - 20060428_1800_r8
GC   - 20060501_AA)
JCL - 20060524a
```

Check the setting for variable com.ibm.ws.scripting.defaultLang in wsadmin.properties in the properties directory of the appropriate WebSphere Application Server profile name. The value of jacl is required.

Coexistence with other JVMPI agents

The ITCAM for Response Time Tracking J2EE data collector uses JVMPI to gather information. It can coexist with other products using this interface. Tivoli maintains a list of tested products for coexistence with the ITCAM for Response Time Tracking J2EE component. During deployment, a jvmpiagents.properties file is created containing the list of tested products. All products defined as supported are fully deployed, meaning that the runtime files are generated and the JVM arguments are defined.

The deployment process checks whether other JVMPI agents are defined in the generic JVM arguments and compares them with the allowed entries in jvmpiagents.properties. This includes instrumentation for ITCAM for WebSphere. Figure 10-13 shows an excerpt of our jvmpiagents.properties file from /var/tmtp/MA/app/instrument/6101/lib.

```
#
# Supported JVMPI Profiler Agents
#
# Cyanea WSAM 2.1.4/3.1
supported.wsam31=-Xruncyanea
#
# Cyanea WSAM 3.2/ITCAMfJ2EE 6.0+
supported.wsam32=-Xrunam
#
## Cyanea ITCAMfWAS 6.0 (OS400 only)
supported.itcamfwas.os400.v5=-Xrunam5
supported.itcamfwas.os400.v6=-Xrunam6
#
# Rational Performance Analyst 6.1
supported.rpa=-XrunpiAgent
#
# ITM/WebSphere Performance Monitor
supported.wpm=-XrunpmiJvmpiProfiler
#
# CAM Toolkit
supported.camtoolkit=-Xrunam_*
#
# Unsupported JVMPI Profiler Agents
#
# Borland OptimizeIt
unsupported.BorlandOptimizeIt1=-Xrunoii
unsupported.BorlandOptimizeIt2=-Xrunpri
unsupported.BorlandOptimizeIt3=-Xruncci
unsupported.BorlandOptimizeIt4=-Xruntdi
```

Figure 10-13 Example of jvmpiagents.properties

Note: If the phrase `Installed - Restart of the application server needed` shows up under component status after server restart (as shown in Figure 10-14), it might indicate that an unsupported JVMPI agent was found.

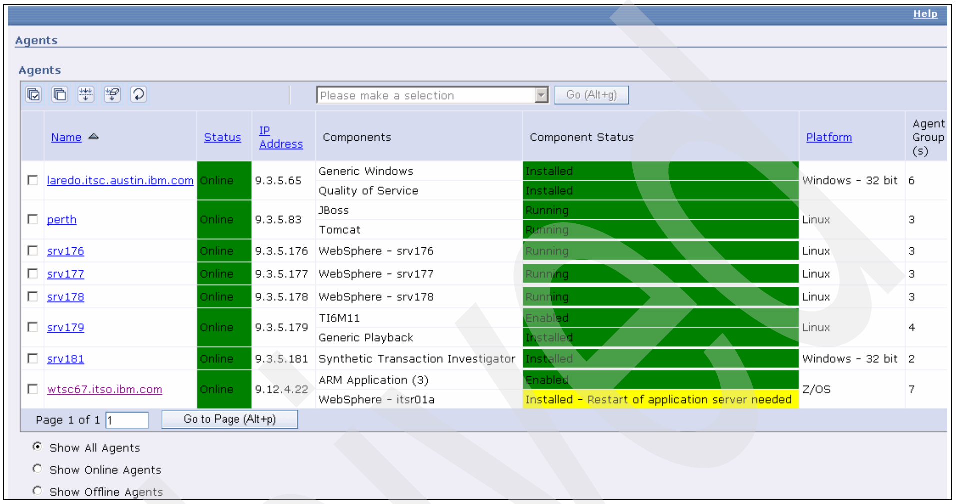


Figure 10-14 Indicator for existence of an unsupported JVMPI agent

In this case, check the log file msg-ma.log in the Tivoli logs directory (in our lab, in /var/ibm/tivoli/common/BWM/logs). Message BWMN0052E in msg-ma.log informs you that an unsupported JVMPI agent was found for the J2EE server in question.

We added the line shown in bold in Figure 10-13 on page 548 to facilitate ITCAM for WebSphere data collector coexistence. To resolve the message in Figure 10-14, we run **enableprobes** (from the /var/tmtp/MA/app/instrument/6101/bin directory) with the following arguments:

```
./enableprobes.sh -serverHome /wasitconfig/itcell/itnodea -serverName
itsr01a -profileName default -enable -netdeploy
```

ITCAM for Response Time Tracking writes information to SYSOUT and SYSPRINT of the J2EE server. Product logs and trace information are in the common Tivoli log directory, for example, /var/ibm/tivoli/common/BWM/logs.

Problem finding libraries

WebSphere Application Server could not find some requested libraries. These libraries must be allocated in the LIBPATH. For example:

libarmjni.so /usr/lpp/tmtp/V6R1M0/MA/bin/zos/USRLIB

libvirt.so /var/tmtp/MA/app/instrument/6101/lib/os390

File not found: cynlogging.properties

The logging configuration file was not found in the created runtime environment: /var/tmtp/MA/app/instrument/6101/appServers/itsr01a_213/config/cynlogging.properties. You can copy this file from the ITCAM for WebSphere data collector.

WebSphere Application Server cannot write to common log

ITCAM for Response Time Tracking uses the Tivoli common path to write log information. This path is typically /var/ibm/tivoli/common/BWM/logs. The WebSphere Application Server user ID needs access to this directory to create its log files. You can either change the owner of the path or connect the WebSphere Application Server user ID to the ITCAM for Response Time Tracking user.

10.5 ITCAM for CICS Transactions

ITCAM for CICS Transactions is a data collector installed on CICS systems to provide data for ITCAM for Response Time Tracking or ITCAM for WebSphere. ITCAM for CICS Transactions does not provide an in-depth application analysis or CICS system analysis.

ITCAM for CICS Transactions is not a stand-alone product. It needs ITCAM for Response Time Tracking, ITCAM for WebSphere, or both. We discuss the ITCAM for CICS Transactions integration in their respective chapters.

We base the discussion here on *IBM Tivoli Composite Application Manager IMS Transactions Product Guide*, SC32-9511, and *IBM Tivoli Composite Application Manager CICS Transactions Product Guide*, SC32-9510.

10.5.1 Concepts

The CICS data collector uses standard CICS calls that run in CICS to monitor transactions. It also installs the CICS global user exit (GLUE) and task-related user exit (TRUE). The global user exit gets control for every EXEC CICS execution, and the task-related user exit gets control for task invocation and completion.

The command agent installs itself as a CICS transaction SAMC that must be running for the extent of the data collector. The event agent runs in a JVM in CICS to bridge the communication from the CICS data collector to the managing server. Figure 10-15 shows the CICS data collector concept.

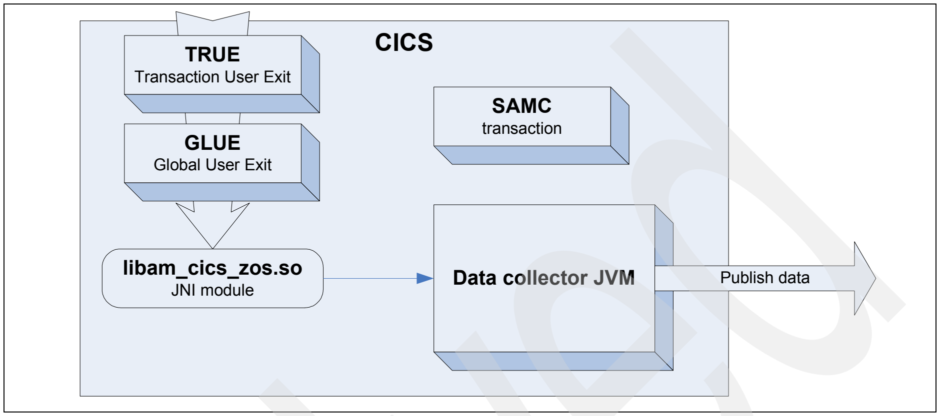


Figure 10-15 CICS data collector

10.5.2 Implementation

The configuration of the CICS data collector is according to *IBM Tivoli Composite Application Manager CICS Transactions Product Guide*, SC32-9510. Configuration tasks for ITCAM for CICS Transactions include:

1. (Optional) Define the CYN1 subsystem. See 10.2, "The CYN1 subsystem" on page 523.
2. Assign the ITCAM for CICS Transactions library as APF authorized. (The library is SCYNAUT2.) APF authorization can be given using either the IEAAPFxx or the PROGxx member from SYS1.PARMLIB, or using the SETPROG command similar to:

```
SETPROG APF,ADD,DSN=ITCAM.V6R1.SCYNAUT2,VOL=ITCAM1
```

3. Update the CICS startup JCL to include the library SCYNAUT2 in the STEPLIB and DFHRPL concatenation.
4. We recommend the JVM Support for the CICS Region.

5. Generate runtime environments in UNIX System Services. ITCAM for CICS Transactions comes with a default template directory in /usr/lpp/itcam/cics/CICS1. This template directory must be replicated to a runtime directory based on the CICS application ID. This can be performed using the **setupcics.sh** command (Example 10-10). The configuration program automatically updates the datacollector.properties and datacollector.env files.

Example 10-10 Running setupcics.sh

```
TIVO01:/usr/lpp/itcam # ./cics/CICS1/bin/setupcics.sh
+--------------------------------------------------------+
|  (C) Copyright IBM Corp. 2005,2006 All Rights Reserved.    |
|    ITCAM for CICS Transactions Data Collector V6.1.0       |
|               Runtime configuration.                       |
+--------------------------------------------------------+
Enter Java home directory [/usr/lpp/java/J1.4]:
/usr/lpp/java/J1.4
Enter ITCAM runtime directory name [/u/itcam]:
/var/itcam61
Enter the CICS region name (APPLID) to monitor [cics]:
SCSCBUD1
Do you want to enable ITCAM for WebSphere support [y or n]:
y
Enter the IP address of the ITCAM Managing Server:
peoria.itsc.austin.ibm.com
Enter ITCAM Managing Server install directory [/opt/IBM/itcam/WebSphere/MS]:
/opt/IBM/itcam/WebSphere/MS
Do you want to enable ITCAM for RTT Support [y or n]:
y
What is the RTT port number [32323]:
32323
Enter the name of the TMTP MA install directory [/etc/tmtp]:
/etc/tmtp/MA
Do you want to enable RTT MQ Monitoring Support [y or n]: n
Setup will create an ITCAM runtime with the following parameters:

 1)    JAVA runtime:      /usr/lpp/java/J1.4
 2)    ITCAM CICS runtime: /var/itcam61/cics/SCSCBUD1
 3)    CICS APPLID:       SCSCBUD1
 4)    ITCAM enabled:     yes
       ITCAM MS host:     peoria.itsc.austin.ibm.com
       ITCAM MS home:     /opt/IBM/itcam/WebSphere/MS
 5)    RTT enabled:       yes
       RTT MA port:       32323
       RTT MA dir:        /etc/tmtp/MA
 6)    RTT MQ enabled:    no
```

```
Create ITCAM configuration for CICS SCSCBUD1?
Enter item number to modify, 'y' to accept, or 'n' to cancel: y
ITCAM configuration for SCSCBUD1 created in /var/itcam61/cics/SCSCBUD1
```

6. Update the CICS PLT definitions. ITCAM for CICS Transactions supplies a sample PLT definition in SCYNSAM2(CYN$PTCY) for assembling. If you already have a PLT definition, you can copy the DFHPLT definition from CYN$PTCY to your PLT definition to include CYNCINI1, DFHDELIM, and CYNCINI2. For the example, we use the data set member CICSLOAD(DFHPLTCY).
7. Update CICS SYSIN to include the following parameter:

   ```
   PLTPI=CY
   INITPARM=(CYNCINI1=’/var/itcam61/’)
   INITPARM=(CYNCSQLP=DB2.DBRMLIB)
   ```

8. Update CICS definitions using the SCYNSAM2(CYN$INST) job. This job creates a new group called CYNCICS. This CYNCICS must be added to a group list that will be auto-started when CICS is started.
9. Restart your CICS region.

10.5.3 Problem determination and log files

The CICS data collector writes information to:

- DD name SYSOUT: initial startup of the JVM. Failure to start the JVM is shown here.
- DD name SYS00001: Provide common system output for the data collector Java process.
- Message and trace files in the Tivoli common directory, typically under /var/ibm/tivoli/common/CYN/logs:
  – msg-<applid>-cicsprobe.log
  – trace-<applid>-cicsprobe.log
  – msg-<applid>-cicsprobe-native.log
  – trace-<applid>-cicsprobe-native.log
- Check the setting for variables am.classpath, java.home, am.path, and am.libpath in datacollector.env in your appropriate CICS runtime directory, for example, /var/itcam61/cics/<applid>/.

10.6 ITCAM for IMS Transactions

ITCAM for IMS Transactions is a data collector installed on IMS systems to provide data for ITCAM for Response Time Tracking or ITCAM for WebSphere. ITCAM for IMS Transactions does not provide an in-depth application analysis or IMS system analysis.

ITCAM for IMS Transactions provides the following information:

- IMS transaction response time
- IMS processor and memory usage

ITCAM for IMS Transactions is not a stand-alone product. It needs ITCAM for Response Time Tracking or ITCAM for WebSphere, or both. We discuss more about ITCAM for IMS Transactions integration in the ITCAM for Response Time Tracking and ITCAM for WebSphere chapters.

We base the discussion here on *IBM Tivoli Composite Application Manager IMS Transactions Product Guide*, SC32-9511, and *IBM Tivoli Composite Application Manager CICS Transactions Product Guide*, SC32-9510.

10.6.1 Concepts

The IMS data collector installs itself as a standard IMS exit that taps transaction information. There are two IMS exits that are used:

- DFSYIOE0: OTMA message exit
- DFSMSCE0: terminal message exit

These exits then turn the processing to the data collector JVM using Java Native Interface (JNI) routines. The data collector forwards the information to the managing server. Figure 10-16 illustrates this process.

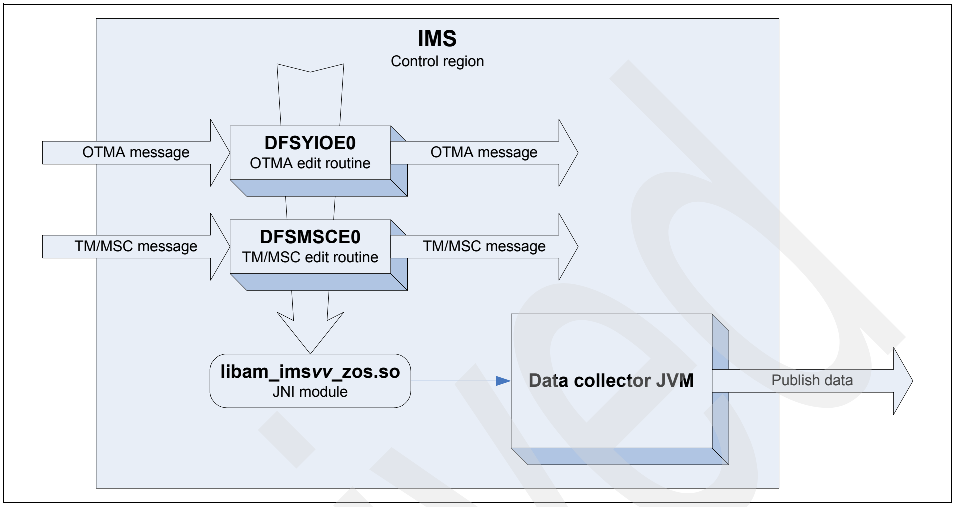


Figure 10-16 IMS data collector

The MSC edit routine takes control when a message is passed around. It can collect the time stamp when:

- A transaction message is inserted from the terminal (TR start).
- A transaction message is returned to the terminal (TR end).
- A program gets the message from IO PCB (PR start).
- A program inserts a message to IO PCB (PR end).

10.6.2 Implementation

Perform the configuration of the ITCAM for IMS Transactions data collector according to the instructions in *IBM Tivoli Composite Application Manager IMS Transactions Product Guide*, SC32-9511. The configuration tasks are:

1. (Optional) Set up the CYN1 subsystem, as discussed in 10.2, “The CYN1 subsystem” on page 523.
2. Assign the ITCAM for IMS Transactions library as APF authorized. The library is SCYNAUT3. APF authorization can be given using either the IEAAPFxx or PROGxx member from SYS1.PARMLIB or using the SETPROG command similar to:

```
SETPROG APF,ADD,DSN=ITCAM.V6R1.SCYNAUT3,VOL=ITCAM1
```

3. Update the IMS control region startup JCL to:
   – Add SCYNAUT3 to STEPLIB concatenation.
   – Set the IMS region size to 0 or any appropriate number. The data collector is a Java process. Therefore, it requires additional virtual memory.
   – Add a new DD called CYNIMSIN. The content is an 80-byte record with the path of the IMS data collector product. You can refer to the SCYNSAMP(CYN$PATH). For the runtime environment in Example 10-11, this must refer to /var/itcam61.
4. Depending on your IMS version, link-edit the appropriate IMS exit to use. You can use the CYNILINK job from SCYNSAM3. The default exit is link-edited for IMS V7.1. IMS is also supported for IMS V8.1 and IMS V9.1.
5. Generate runtime environments in UNIX System Services. ITCAM for IMS Transactions comes with a default template directory in /usr/lpp/itcam/ims/IMS1. This template directory must be replicated to a runtime directory, such as /var/itcam61/ims/<applid>, that represents the IMS data collector in the specific application ID. This can be performed using setupims.sh. See Example 10-11.

Example 10-11 Running setupims.sh

```
TIVO01:/usr/lpp/itcam # ./ims/IMS1/bin/setupims.sh
+------------------------------------------------------------+
|(C) Copyright IBM Corp. 2005,2006 All Rights Reserved.       |
|  ITCAM for IMS Transactions Data Collector V6.1.0           |
|                Runtime configuration.                       |
+------------------------------------------------------------+
Enter Java home directory [/usr/lpp/java/J1.4]:
/usr/lpp/java/J1.4
Enter ITCAM runtime directory name [/u/itcam]:
/var/itcam61
Enter the IMS region ID to monitor [IMS]:
IMSJ
Enter version of IMS [71, 81, 91]:
91
Enter IMS network name [network1]:
wtsc67.itso.ibm.com
Do you want to enable ITCAM for WebSphere support [y or n]:
y
Enter the IP address of the ITCAM Managing Server:
peoria.itsc.austin.ibm.com
Enter ITCAM Managing Server install directory [/opt/IBM/itcam/WebSphere/MS]:
/opt/IBM/itcam/WebSphere/MS
Do you want to enable ITCAM for RTT Support [y or n]:
y
What is the RTT port number [32323]:
32323
```

```
Enter the name of the TMTP MA install directory [/etc/tmtp]:
/etc/tmtp/MA
Do you want to enable RTT MQ Monitoring Support [y or n]:
n
Setup will create an ITCAM runtime with the following parameters:

 1)    JAVA runtime:      /usr/lpp/java/J1.4
 2)    ITCAM IMS runtime: /var/itcam61/ims/IMSJ
 3)    IMS Region Information
       IMS APPLID:        IMSJ
       IMS network name:  wtsc67.itso.ibm.com
 4)    ITCAM enabled:     yes
       ITCAM MS host:     peoria.itsc.austin.ibm.com
       ITCAM MS home:     /opt/IBM/itcam/WebSphere/MS
 5)    RTT enabled:       yes
       RTT MA port:       32323
       RTT MA dir:        /etc/tmtp/MA
 6)    RTT MQ enabled:    no

Create ITCAM configuration for IMS IMSJ?
Enter item number to modify, 'y' to accept, or 'n' to cancel: y
ITCAM configuration for IMSJ created in /var/itcam61/ims/IMSJ
```

The setupims.sh script updates the datacollector.properties and datacollector.env files for the IMSJ subsystem that we have.

6. APF authorize data collector native modules:

```
extattr +a /usr/lpp/itcam/ims/IMS1/bin/imsprobe
extattr +a /usr/lpp/itcam/ims/scripts/imsprobe
extattr +a /usr/lpp/itcam/ims/IMS1/lib/libam_ims*_zos.so
```

You can check the extended attribute using the command **ls -lE**.

> **Note:** There is one libam_ims*_zos.so for each IMS version that is supported. You only have to authorize the module for the IMS version that you are implementing.

7. Restart the IMS region.

10.6.3 Problem determination and log files

The CICS data collector writes information to:

- DD name SYSOUT: initial startup of the JVM. A failure to start the JVM is shown here.
- DD name SYS00001: provides common system output for the data collector Java process.

- Message and trace files in the Tivoli common directory, typically under /var/ibm/tivoli/common/CYN/logs:
  - msg-<applid>-imsprobe.log
  - trace-<applid>-imsprobe.log
  - msg-<applid>-imsprobe-native.log
  - trace-<applid>-imsprobe-native.log
- Check the setting for variables am.classpath, java.home, am.path, and am.libpath in datacollector.env in your appropriate IMS runtime directory, for example. /var/itcam61/ims/<applid>/.
- Check for the correct IMS version for variable probe.library.name in <instance-id>.datacollector.properties.

If you do not see any transaction-related data in the ITCAM for WebSphere console, check whether you are running the link job CYNILINK from SCYNSAM3 for the correct IMS version.

10.7 ITCAM for SOA management agent for z/OS

We discuss the overall implementation procedure for the ITCAM for SOA management agent for z/OS in the following sections:

- 10.7.1, “Considerations for the z/OS environment” on page 558
- 10.7.2, “ITCAM for SOA management agent for z/OS” on page 559
- 10.7.3, “Enabling the CICS data collector” on page 570

10.7.1 Considerations for the z/OS environment

As you plan for the deployment of ITCAM for SOA monitoring into your z/OS environment, consider the following additional deployment and configuration options:

- Identify which z/OS systems to monitor with the ITCAM for SOA monitoring agent and other monitoring agents. ITCAM for SOA uses the OMEGAMON monitoring agent architecture, so you might need to consider the OMEGAMON environment. See *IBM Tivoli OMEGAMON XE V3.1.0 Deep Dive on z/OS*, SG24-7155.
- The ITCAM for SOA monitoring agent can be run within the same address space as the Tivoli Enterprise Monitoring Server on z/OS.

Note: Before installing the ITCAM for SOA monitoring agent, review the space requirements and considerations for an SMP/E installed environment to make sure that your z/OS system has sufficient DASD storage available.

Be aware that the DASD space estimates assume that these products were installed in a separate CSI environment. When multiple products are installed into a shared CSI environment, DASD requirements for each product should be less. Refer to *Configuring IBM Tivoli Composite Application Manager on z/OS*, SC32-9493, for additional information.

- By default, the data collector does not have write privileges to write to its log file. You must grant write privileges for the $ITCAMSOA_HOME/KD4 directory to the user ID that runs your WebSphere Application Server, such as ASSR1 in WSCFG1 group.
- Ensure that TCP/IP has been configured correctly in the z/OS environment in which the ITCAM for SOA monitoring agent is running. TCP/IP must have the connection services running with a minimum lifetime of 0. This can be verified using the D TCPIP,NETSTAT,CONFIG console command. The output should be similar to Example 10-12.

Example 10-12 TCP/IP configuration

```
   . . .
NETWORK MONITOR CONFIGURATION INFORMATION:
PKTTRCSRV: NO TCPCNNSRV: YES MINLIFTIM: 0 SMFSRV: YES
```

Otherwise, modify TCP/IP with the **obey** command. Run **V TCPIP,,OBEY,*data set*** from the system console with the input data set containing the line:

```
NETMON SMFS TCPCONNS MINLIFET 0
```

10.7.2 ITCAM for SOA management agent for z/OS

ITCAM for SOA supplies a complete SMP/E distribution for installing on z/OS systems. The ITCAM for SOA product tape includes all of the z/OS components required for installation, which are:

- Configuration Tool V3.1.0
- Tivoli Enterprise Monitoring Server on z/OS
- ITCAM for SOA V6.0 Monitoring Agent

This is an outline of the installation procedure for ITCAM for SOA on z/OS:

1. We perform the standard SMP/E installation. Run this by using batch jobs or by using dialog boxes under the Interactive System Productivity Activity/Program Development Facility (ISPF/PDF). For more information about SMP/E processing, see *IBM Tivoli Composite Application Manager for SOA Program Directory*, GI11-4087. The installed FMIDs are:

   – HKCI310: Configuration Tool V3.1
   – HKDS360: Tivoli Enterprise Monitoring Server Version 3.6.0
   – HKD4600: ITCAM for SOA Agent V6
   – HKLV190: CT Engine

   > **Note:** If you have an existing OMEGAMON XE V3.1.0 monitoring agent product, you do not have to install the configuration tool again. If you have an earlier version of the Candle Installation and Configuration Assistance Tool (CICAT) in the same CSI on your z/OS system, it is overwritten automatically by IBM Configuration Tool V3.1.0 during the SMP/E installation.

2. When ITCAM for SOA has been installed in the ACCEPT stage, we can start the installation configuration tool. The configuration tool is executed from a copy of the KCIINST target library that we call INSTLIB. The high-level qualifier that we use is ITCAMSOA. Note that this configuration tool can be executed by only one TSO user at a time.

   We describe the configuration process for installing an ITCAM for SOA agent and connecting it to a Tivoli Enterprise Monitoring Server running on a distributed workstation. We do not configure the Tivoli Enterprise Monitoring Server on z/OS. We also create a full runtime environment (RTE) instead of splitting the environment for a sysplex environment. See *IBM Tivoli OMEGAMON XE V3.1.0 Deep Dive on z/OS*, SG24-7155, for more information about RTE. Figure 10-17 shows the overall configuration process for a new installation.

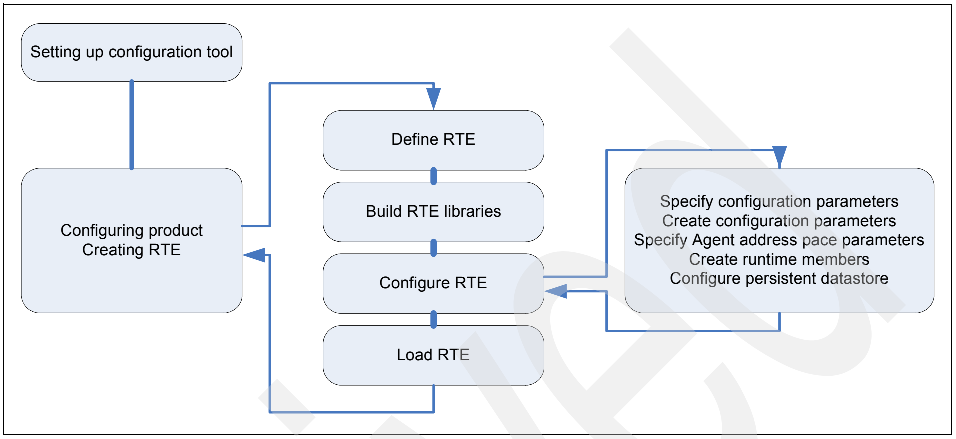


Figure 10-17 Configuration tool processing

The ITCAM for SOA configuration tool window uses Candle terminology. Changes in this book and the IBM Tivoli Monitoring V6.1 documentation include:

– The Candle Management Server is called Tivoli Enterprise Monitoring Server.

– The CandleNet Portal Server is called the Tivoli Enterprise Portal Server.

– The CandleNet Portal is called the Tivoli Enterprise Portal.

3. On the Configuration Tool initial page, shown in Figure 10-18, select option 1 to set up your Configuration Tool.

```
--------------------------------- MAIN MENU ---------------------------------
OPTION ===>

 Enter the number to select an option:

    1   Set up work environment

    2   Install products or maintenance

    3   Configure products

    I   Installation information                <=== Revised
    S   Services and utilities




        Installation and Configuration Assistance Tool Version 310.01
                       (C) Copyright IBM Corp. 1992-2004
                  Licensed Material - Program Property of IBM

 F1=Help  F3=Back
```

Figure 10-18 Configuration tool: main menu

4. The setup involves defining the JOB card and high-level qualifiers and allocates work libraries.

   Select option 2 to configure a product.

   ITCAM for SOA should be available, as shown in Figure 10-19.

```
--------------------------- PRODUCT SELECTION MENU ---------------------------
COMMAND ===>

 Actions: S Select product

S   IBM Tivoli Composite Application Manager for SOA V6.0.0
F1=Help   F3=Back   F5=Refresh   F7=Up   F8=Down
```

Figure 10-19 Configuration tool: product selection

5. When we configure the product, we define the RTE. We only define a full RTE, instead of split SHARING and BASE RTEs. The split RTE is useful, especially if you are running on multiple systems in a sysplex environment. Figure 10-20 shows the definition.

```
------------------------ RUNTIME ENVIRONMENTS (RTEs) -------------------------
COMMAND ===>

Actions: A Add RTE, B Build libraries, C Configure,
         L Load all product libraries after SMP/E,
         D Delete, U Update, V View values, Z Utilities

Action Name     Type    Sharing  Description
  a    SOA      FULL             ITCAM for SOA full RTE
F1=Help F3=Back  (No RTEs defined, use Action A to Add)
```

Figure 10-20 Configuration tool: creating a full RTE

6. As shown in the process overview in Figure 10-21, we need to add (Figure 10-14 on page 549), build, configure, and load the RTE. We do not create a CMS, because we use our Tivoli Enterprise Monitoring Server in our distributed server.

```
---------------------- ADD RUNTIME ENVIRONMENT (1 of 2) ----------------------
COMMAND ===>

RTE: SOA      Type: FULL    Desc: ITCAM for SOA full RTE

  Libraries  High-level Qualifier       Volser Unit     Storclas Mgmtclas PDSE
  Non-VSAM   ITCAMSOA                   TAROM2 3390                       N
  VSAM       ITCAMSOA                   TAROM2
  Mid-level qualifier ==> SOA

  JCL suffix         ==> SOA
  STC prefix         ==> CANS
  SYSOUT class       ==> X           Diagnostic SYSOUT class  ==> X
  Load optimization  ==> N     (Y, N)

  Will this RTE have a Candle Management Server               ==> N  (Y, N)
     If Y, CMS name     ==> SOA:CMS                         (Case sensitive)

  Copy configuration values from RTE ==>          (Optional)

 Enter=Next  F1=Help  F3=Back
```

Figure 10-21 Configuration tool: defining a RTE

7. Press Enter to go to the second RTE definition window, as shown in Figure 10-22.

```
----------------------- ADD RUNTIME ENVIRONMENT (2 of 2) ----------------------
COMMAND ===>

Use OS/390 system variables?    ==> N (Y, N)
   RTE name specification       ==> &SYSNAME
   RTE base alias specification ==>                      n/a
   Applid prefix specification  ==> K&SYSCLONE.
   Use VTAM model applids?      ==> N (Y, N)

Security system        ==> NONE  (RACF, ACF2, TSS, NAM, None)
  ACF2 macro library   ==>

VTAM communications values:
   Applid prefix       ==> CTD        Network ID     ==> USIBMSC
   Logmode table       ==> KDSMTAB1   LU6.2 logmode  ==> CANCTDCS

If you require TCP/IP communications for this RTE, complete these values:
  Hostname             ==> WTSC58.ITSO.IBM.COM
  Address              ==> 9.3.4.129
  Started task         ==> TCPIP
  Port number          ==> 1918
  Interlink subsystem  ==>        (if applicable)
 Enter=Next  F1=Help  F3=Back
```

Figure 10-22 Configuration tool: defining a RTE (second window)

8. In the configuration window for the RTE (Figure 10-23), we configure only the ITCAM for SOA agent. We do not configure the Tivoli Enterprise Monitoring Server because we will connect to Tivoli Enterprise Monitoring Server on a distributed system.

```
---CONFIGURE IBM TIVOLI COMPOSITE APPLICATION MANAGER FOR SOA RTE: SOA --------
OPTION ===>
                                                            Last selected
 Perform the appropriate configuration steps in order:      Date     Time

 If you have defined a TEMS in this RTE that this Agent
 will communicate with, select option 1.
   1  Register with local TEMS

   2  Specify configuration parameters
   3  Create configuration parameters

   4  Specify Agent address space parameters
   5  Create runtime members
   6  Configure persistent datastore

   7  Create HFS directories and copy files on USS
   8  Complete the configuration


 F1=Help  F3=Back
```

Figure 10-23 Configuration tool: ITCAM for SOA steps

9. The configuration parameters require a UNIX System Services path. We use the path /usr/lpp/Candle, as shown in Figure 10-24.

```
----------------------- SPECIFY CONFIGURATION PARAMETERS ----------------------
OPTION ===>

   HFS CandleHome directory  (case sensitive):
    /usr/lpp/Candle_______________________
    ______________________________________
    ______________________________________
    ______________________________________
    ______________________________________


 USS CLIST library (Required) ==> SYS1.SBPXEXEC
   HFS migration directory (case sensitive):


 Enter=Next  F1=Help  F3=Back
```

Figure 10-24   Configuration tool: configuration parameters

10. The ITCAM for SOA agent starts in its own address space. Figure 10-25 shows the address space configuration.

```
-------------------- SPECIFY AGENT ADDRESS SPACE PARAMETERS -------------------
COMMAND ===>


 The following information is needed to define the Agent address space:

  Agent started task             ==> CANSD4
  Connect to CMS in this RTE     ==> N          (Y, N)
     Name of Primary CMS         ==> BEIJING

  Specify communication protocols in priority sequence:
     Protocol 1                  ==> IPPIPE  (SNA, IP, IPPIPE)
     Protocol 2 (optional)       ==>         (SNA, IP, IPPIPE)
     Protocol 3 (optional)       ==>         (SNA, IP, IPPIPE)


 Enter=Next  F1=Help  F3=Back  F5=Advanced  F10=CMS List
```

Figure 10-25   Configuration tool: agent address space

11. We configure the connection to the Tivoli Enterprise Monitoring Server, as shown in Figure 10-26.

```
--------------------- SPECIFY AGENT PRIMARY CMS VALUES ------------------------
COMMAND ===>

 CMS name (case sensitive)        ==> BEIJING

 Complete this section if the primary CMS requires SNA support:
   LU6.2 logmode                  ==>
   Logmode table name             ==>
   Local location broker applid   ==>
   Network ID                     ==>

 Complete this section if the primary CMS requires TCP support:
   Hostname                       ==> BEIJING.ITSC.AUSTIN.IBM.COM
   Address                        ==> 9.3.5.35
   Network interface card (NIC)   ==>

   IP port number                 ==>

   IP.PIPE port number            ==> 1918
   Address translation            ==> N     (Y, N)
       Partition name             ==>

 Enter=Next  F1=Help  F3=Back
```

Figure 10-26   Configuration tool: CMS values

Figure 10-27 shows the communication parameter for the ITCAM for SOA agent.

```
------------------ SPECIFY AGENT IP.PIPE CONFIGURATION VALUES -----------------
COMMAND ===>


 Specify the TCP communication values for this Agent:
   Hostname                       ==> WTSC58.ITSO.IBM.COM
   Address                        ==> 9.3.4.129
   Started task                   ==> TCPIP


   If applicable
     Network interface card (NIC) ==>
     Interlink subsystem          ==>
     IUCV interface in use?       ==> N     (Y, N)

 Specify Agent IP.PIPE configuration:
   Address translation            ==> N     (Y, N)
      Partition name              ==>


 Enter=Next  F1=Help  F3=Back
```

Figure 10-27 Configuration tool: IP.PIPE configuration

12. The persistent data store is used to store short-term historical data for the ITCAM for SOA agent. We generally use the default values for the configuration menu.

13.Load and populate the runtime environment. You have the option to populate the UNIX System Services directory with this step or defer it to be executed manually from RKANSAM(KD4USSJB). See Figure 10-28.

```
------------------------ LOAD JOB - INCLUDE USS STEPS -----------------------
COMMAND ===>

The IBM Tivoli Composite Application Manager for SOA component is
configured in this RTE. In order to include information for this RTE on
z/OS UNIX System Services (USS), you must meet the following conditions:
-  the job to create directories and copy files to HFS must be submitted
   on a machine that has access to the USS directories.
-  the job to create directories and copy files to HFS must be submitted
   by a TSO userid that has write access to the HFS directories specified
   in the Specify configuration parameters panel from the IBM Tivoli
   Composite Application Manager for SOA menu.

If the above conditions are met, then IBM recommends that you specify Y below
to ensure that all maintenance is synchronized.  If not, then specify N and
the configuration job will create a KD4USSJB job for later submission. Submit
the hilev.RKANSAMU(KD4USSJB) job from an appropriate machine with a TSO
userid that satisfies these conditions.


Do you want to include the USS steps in the configuration job? ==> Y (Y, N) Y

 Enter=Next  F1=Help
```

Figure 10-28 Configuration tool: load job

14.Copy startup JCLs to the system procedure library, such as SYS1.PROCLIB. These JCLs reside in the RKANSAM data set:

CANSD4	ITCAM for SOA agent
KDSPROC1	Persistent data store maintenance
KDSPROC2	Persistent data store backup and initialization

15.Authorize the RKANMOD, RKANMODU, and RKANMODL from your RTE. You can use the SETPROG console command to perform this. Our setup uses the following commands:

```
SETPROG APF,ADD,DSN=ITCAMSOA.SOA.RKANMOD,VOL=TAROM2
SETPROG APF,ADD,DSN=ITCAMSOA.SOA.RKANMODU,VOL=TAROM2
SETPROG APF,ADD,DSN=ITCAMSOA.SOA.RKANMODL,VOL=TAROM2
```

Activating the ITCAM for SOA to the WebSphere Application Server environment on z/OS is similar to the distributed environment. See 9.2.4, “Enabling the monitoring agent” on page 473.

10.7.3 Enabling the CICS data collector

ITCAM for SOA V6.1 has a new feature to collect Web services information from the CICS Transaction Server. This feature enables Web services information to be collected similar to the WebSphere Application Server collection.

There are several configuration steps that must be performed:

- The CICS Started Task user ID must have read/write access to the agent directory in UNIX System Services. Our setup is /usr/lpp/Candle.
- The CICS started task needs access to the TKANMOD data set in the DFHRPL concatenation.
- Data collector programs and transactions must be defined using the CICS System Definition program. A sample JCL is provided in TKANSAM(KD4CSD). Some of these transactions might need to be secured (KD4, KD4O, and KD4C).
- Update the CICS program load table (PLT) to include THE KD4INIT module to be loaded and unloaded with startup and shutdown of CICS.
- The Web services through CICS are handled using the CICS pipelines. These pipelines are configured using an XML definition. The pipeline that you want to monitor must be handled by the KD4HAND program. This program is basically the Web services interceptor. KD4HAND can be defined similarly for the requestor pipeline and provider pipeline. Example 10-13 shows the handler definition excerpt.

Example 10-13 CICS handler definition

```
<service_handler_list>
   <handler>
      <program>KD4HAND</program>
      <handler_parameter_list><soap_1.1/><handler_parameter_list>
   </handler>
</service_handler_list>
```

11

Integration scenarios with ITCAM products

This chapter describes the features and capabilities of the IBM Tivoli Composite Application Manager family of products. The discussion includes:

11.1 ITCAM family integration scenarios

As the products develop, cross-product integration is needed to see the bigger picture and have a better understanding of the environment. We discuss several integration topics here:

- Section 11.2, "Deep dive and in-context launch" on page 572, discusses this visible first level of integration from ITCAM for Response Time Tracking to ITCAM for WebSphere.
- Section 11.3, "Tivoli Enterprise Portal workspace" on page 576, discusses a sample session in which we customize Tivoli Enterprise Portal workspace to monitor our trader application servers with information from ITCAM for Response Time Tracking, ITCAM for SOA, and ITCAM for WebSphere.
- Section 11.4, "Workflows and situations" on page 602, provides an overview of a sample workflow that can be used to raise a situation based on the analysis from multiple situations that are customized from various ITCAM products.

11.2 Deep dive and in-context launch

The ITCAM for Response Time Tracking topology view allows deep dive to Launch In Context ITCAM for WebSphere. This feature enables operational and support areas to view the application at a high level using ITCAM for Response Time Tracking and, where needed, quickly link to the relevant detailed data in the same user interface. Using the ITCAM for Response Time Tracking topology view, you can quickly launch ITCAM for WebSphere by right-clicking a J2EE object in the topology. This discussion includes:

- 11.2.1, "Configuring Launch In Context" on page 573
- 11.2.2, "Using Launch In Context" on page 575

11.2.1 Configuring Launch In Context

This functionality must be preconfigured before use by using the ITCAM for Response Time Tracking management browser. From the Web interface, navigate to **System Administration** → **System Properties** and choose **Launch In Context** from the menu, as shown in Figure 11-1. Click **Show Properties**.

Figure 11-1 Launch In Context configuration menu

This opens a window similar to the one shown in Figure 11-2.

Provide the ITCAM for Response Time Tracking server host and ITCAM for WebSphere managing server host details, as we have done for our example environment, and click **Apply**.

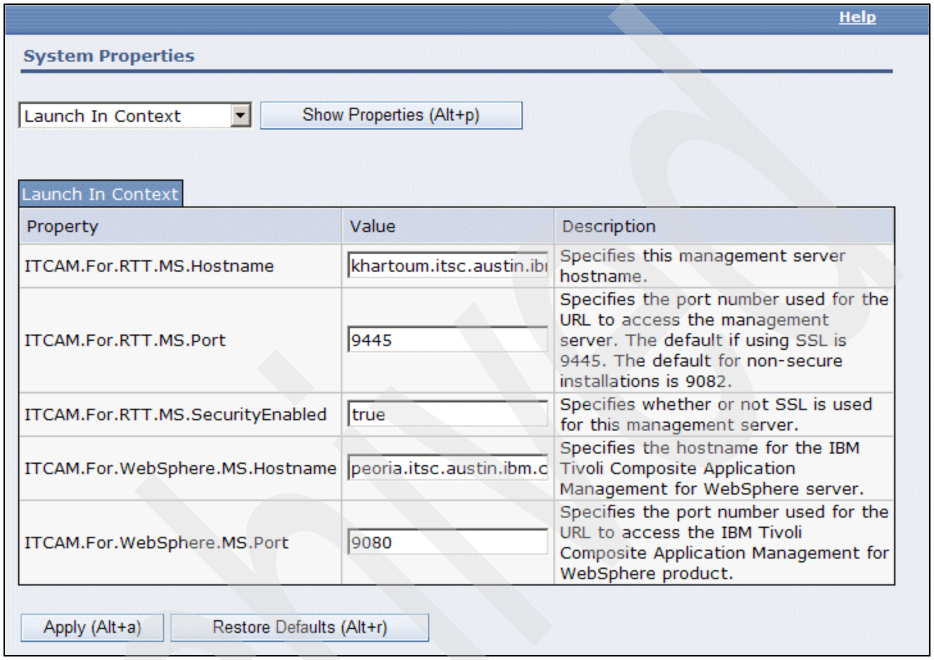


Figure 11-2 Launch In Context configuration

11.2.2 Using Launch In Context

The benefit of performing this integration is the reduction in diagnosis or resolution time of problems because all relevant information is right there. This is highlighted in our example application in Figure 11-3. On a relevant object in the topology view, right-click and select **ITCAM for WebSphere** from the pop-up menu.

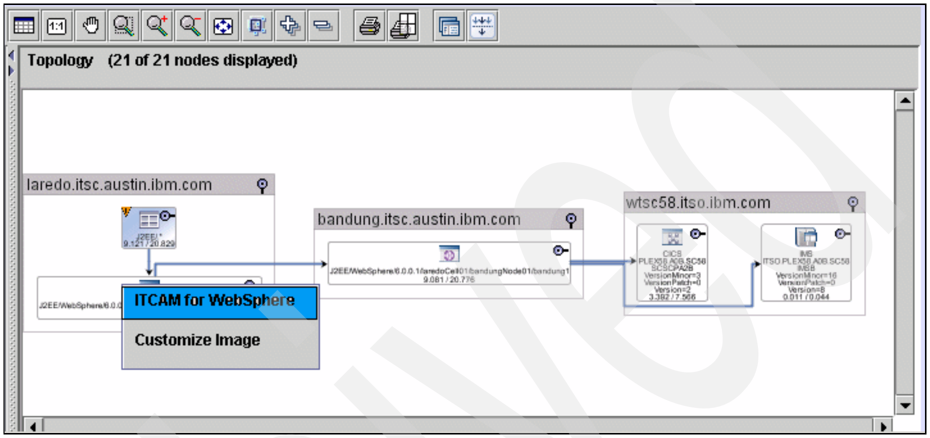


Figure 11-3 Launch In Context pop-up menu

The browser for ITCAM for WebSphere opens. A valid user name and password might be needed. Figure 11-4 shows an example.

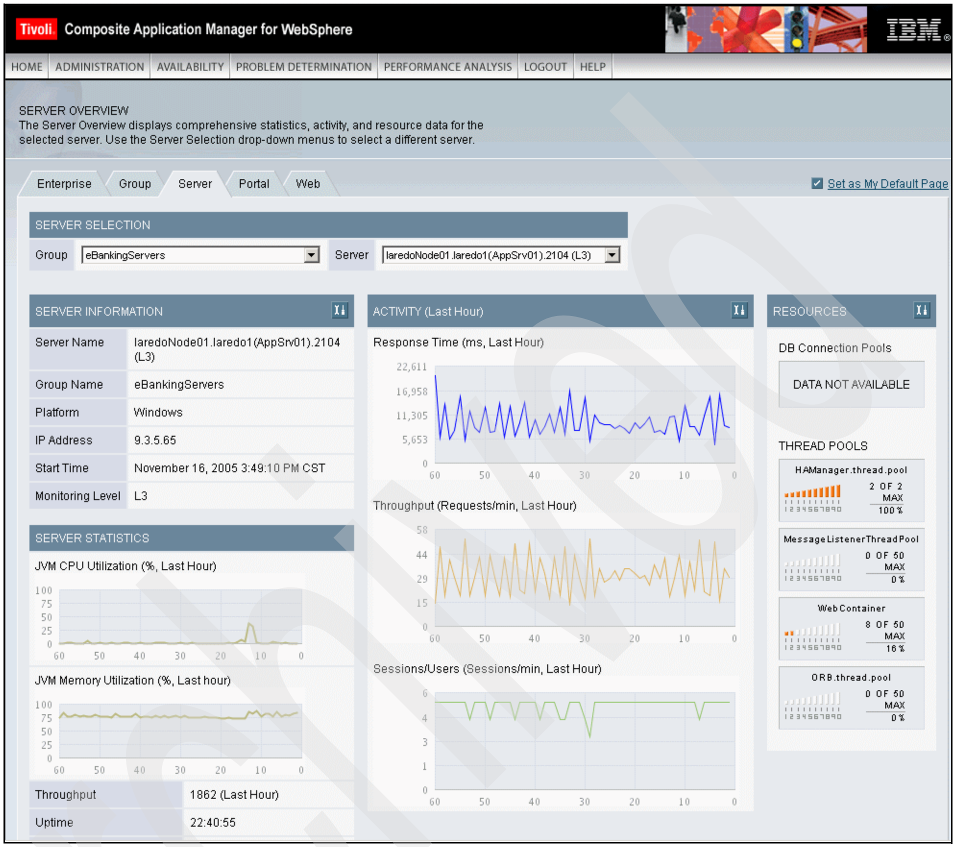


Figure 11-4 Launch In Context WebSphere window

We selected to Launch In Context from the J2EE object on laredo in the topology view. As you see, it has directly placed the view into context, showing details of laredo.

11.3 Tivoli Enterprise Portal workspace

Tivoli Enterprise Portal is the common user interface for real-time information and historical data collected by Tivoli Enterprise Monitoring Server. Products from the various Tivoli families integrate with it to provide a flexible, customizable repository and linked information. This section shows an example Tivoli Enterprise Portal customization for workspaces that can be used to monitor our trader application.

The discussion includes:

11.3.1 Designing the workspaces

As discussed in Chapter 1, “IBM Tivoli Composite Application Manager overview” on page 1, the application configuration of the trader application goes through two levels of WebSphere Application Servers. This enables us to configure the structure of our navigation tree to show the application servers.

We assume that the Tivoli Enterprise Portal is operational and all data collection from ITCAM for Response Time Tracking, ITCAM for SOA, and ITCAM for WebSphere are running. Refer to *IBM Tivoli Monitoring User's Guide*, SC32-9409, for further details about defining queries and workspaces. We use a basic workspace that does not includes links.

You might need to be familiar with the information provided for each product. We discuss them in the following topics:

- ITCAM for Response Time Tracking: 6.10, “Working with Tivoli Enterprise Portal” on page 326
- ITCAM for WebSphere: 3.11, “Using Tivoli Enterprise Portal” on page 143
- ITCAM for SOA: 9.3, “ITCAM for SOA usage scenarios” on page 481

Because we are using an IBM Tivoli Monitoring platform, we can create new workspaces with the combined content of several monitoring sources. This is similar in OMEGAMON Dashboard Edition (DE).

We create a new navigation tree and workspaces for the trader application and the two servers that it uses. Important performance considerations for workspaces include:

- The queries in the workspace retrieve data from the agents through the Tivoli Enterprise Monitoring Server. They use some processing power on the agent.
- You can share queries for several portlets in the workspace to save some processing.
- Limit the rows and columns of data that you retrieve from the query as much as possible from the query definition, not the filter definition of the portlet.

11.3.2 Building the navigation tree

As management agents are configured, they are automatically added to the Physical view workspaces in the navigator panel. As you would expect in our environment, it shows the various physical servers that we have worked with so far (Figure 11-5).

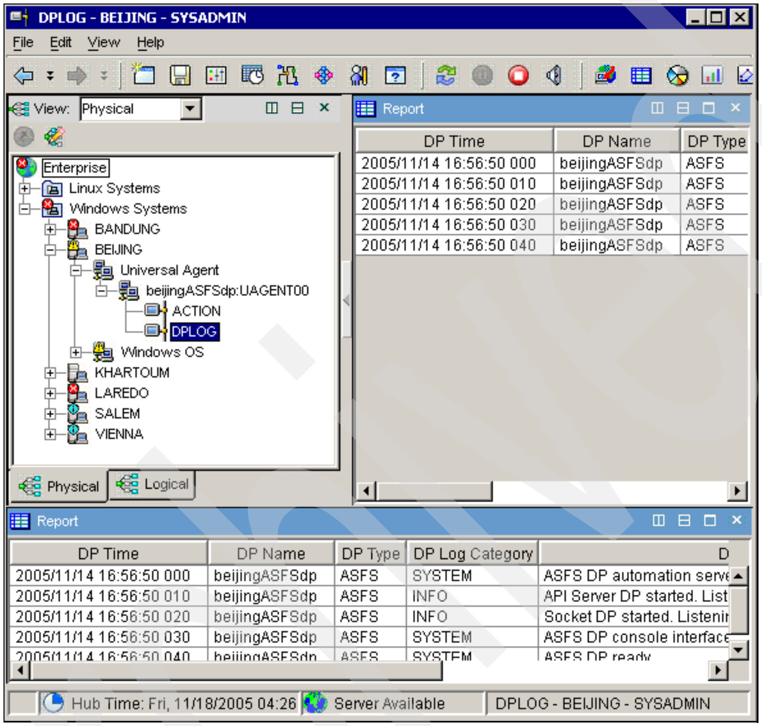


Figure 11-5 Tivoli Enterprise Portal Physical view example

We develop a new navigation tree called trader for a user who must monitor the trader application only. We can do this using a logical view in a new navigator. We perform these procedures:

1. Click the Edit Navigator View icon to open the Edit Navigator View window shown in Figure 11-6.

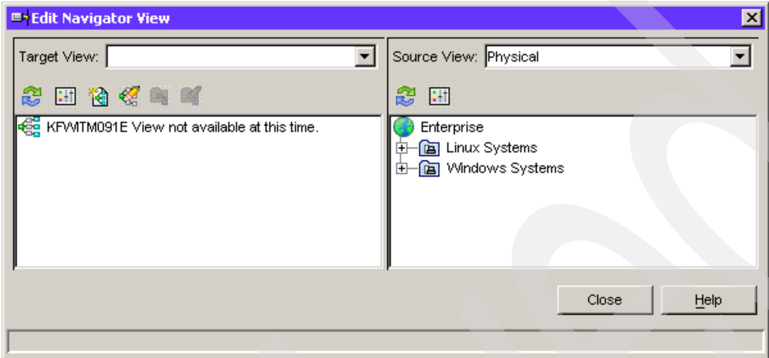


Figure 11-6 Edit Navigator View

2. Click the **Create New Navigator View** icon and enter a name and description, as we did in Figure 11-7. Click **OK**.

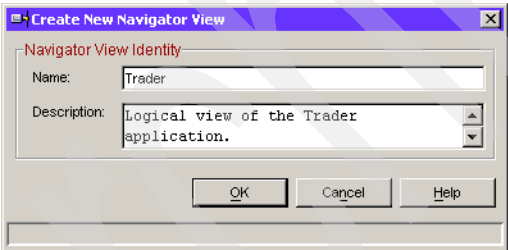


Figure 11-7 Create New Logical View

3. In the new trader navigator item, we create two more navigator items. Right-click and select **Create Child Item**. This represents the WebSphere Application Servers that we use: bandung1 and laredo1.

4. The managed systems represent the monitoring agents that provide the information display for the appropriate navigation tree. Select this carefully because this represents available data for your portlets. Figure 11-8 shows the property of the bandung1 navigator view.

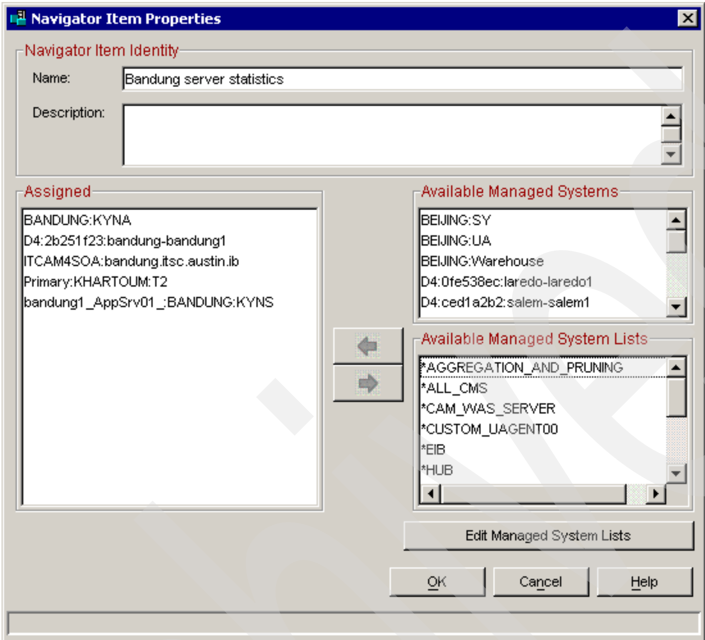


Figure 11-8 Bandung1 Navigator managed systems

Figure 11-9 shows the display for the laredo navigator item.

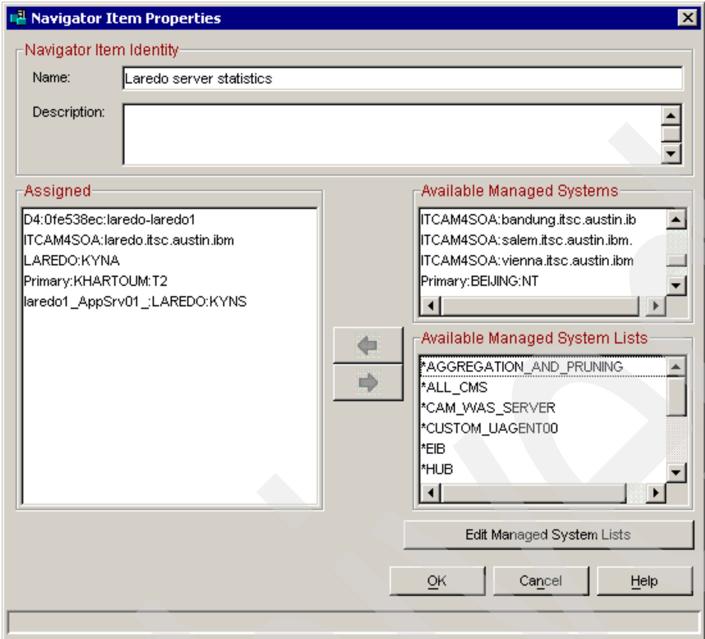


Figure 11-9 Laredo1 Navigator managed systems

5. In Figure 11-8 on page 580 and Figure 11-9, you can see that we use an individual agent for ITCAM for SOA and ITCAM for WebSphere, but for ITCAM for Response Time Tracking, we use a shared object from the management server.

D4	Represents ITCAM for SOA agent data
ITCAMSOA	Represents ITCAM for SOA agent status
KYNA	Represents ITCAM for WebSphere agent status
KYNS	Represents ITCAM for WebSphere agent data
T2	Represents ITCAM for Response Time Tracking agent data and status

6. You can also add views from the physical view into this new navigator view. To drag and drop, select a physical view object. (When selected, a box surrounds it.) Drop it directly onto the trader object on the left, ensuring that it has the surrounding outline box. We do not perform this step. Figure 11-10 shows the final navigator tree for our example.

   Now that the navigator tree is defined, we can close the navigator edit window by clicking **Close**.

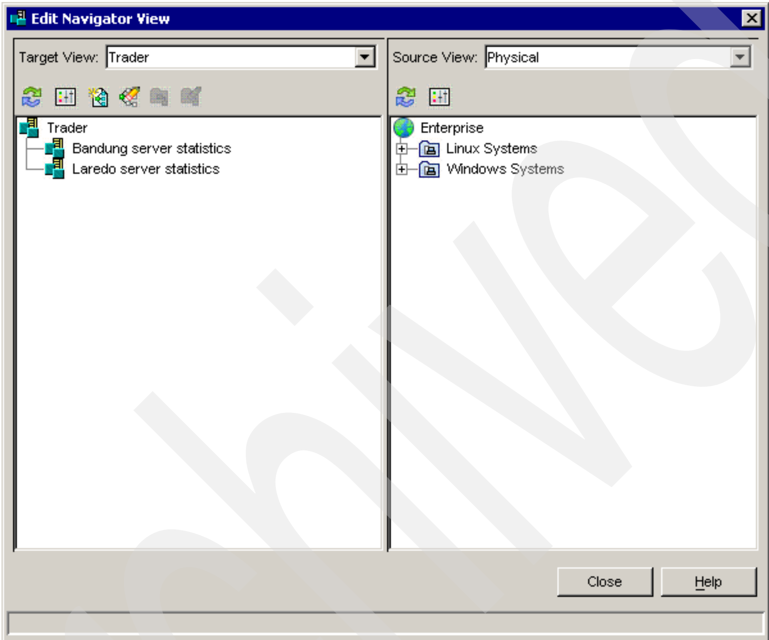


Figure 11-10 Final Navigator window

11.3.3 Defining the workspace

The initial workspace for the new navigator item is an empty workspace with a notepad and a browser pointing to a generic page showing `Workspace not defined`, as shown in Figure 11-11.

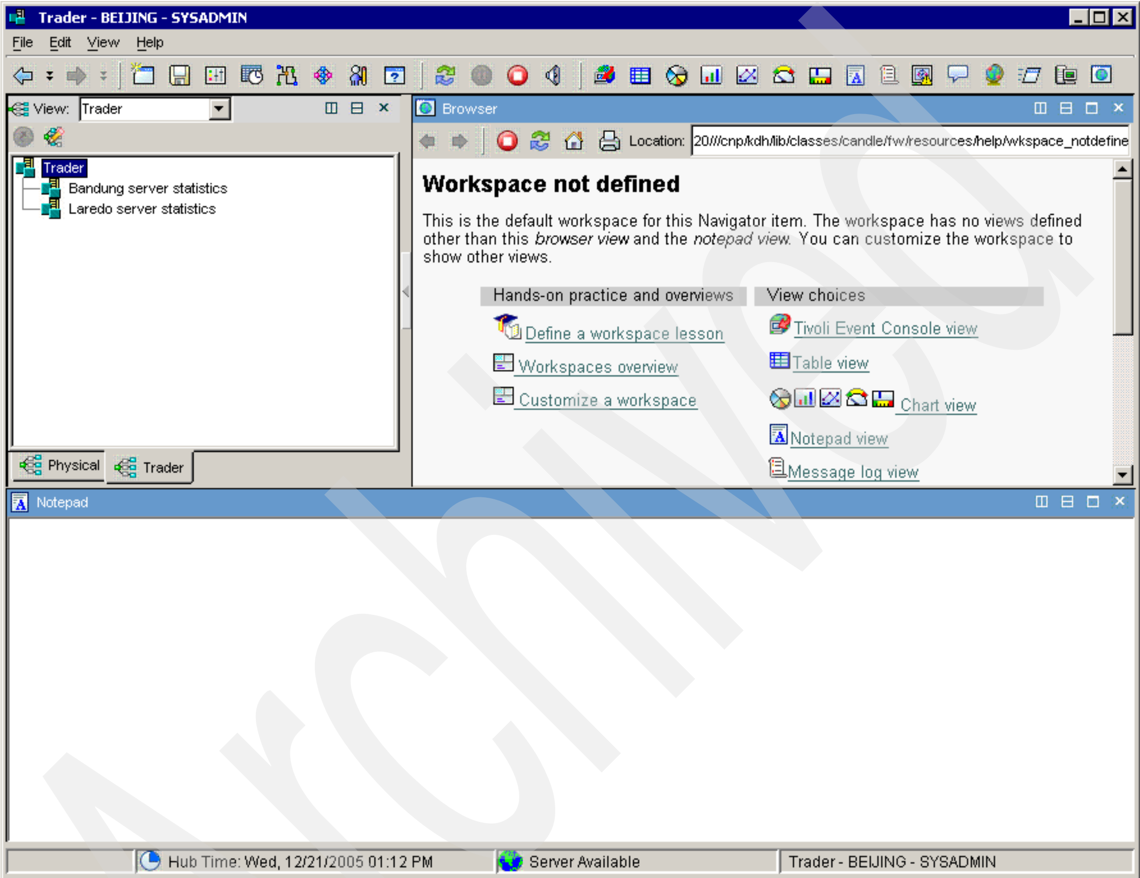


Figure 11-11 Workspace not defined

We have to define these workspaces for the navigator objects:

- The main trader workspace must be the initial display window for each operator and must consume the least amount of processing. Keeping that in mind, we use the alert view for the individual objects as the initial display for the trader workspace.

  Alerts are generated by background collectors called situations. This has additional processing for alerts only involved in transporting the alert to and from the Tivoli Enterprise Portal Server. When an operator sees an alert, drilling down to the appropriate application server exposes information used in analyzing the problem. Figure 11-12 shows the completed trader workspace.

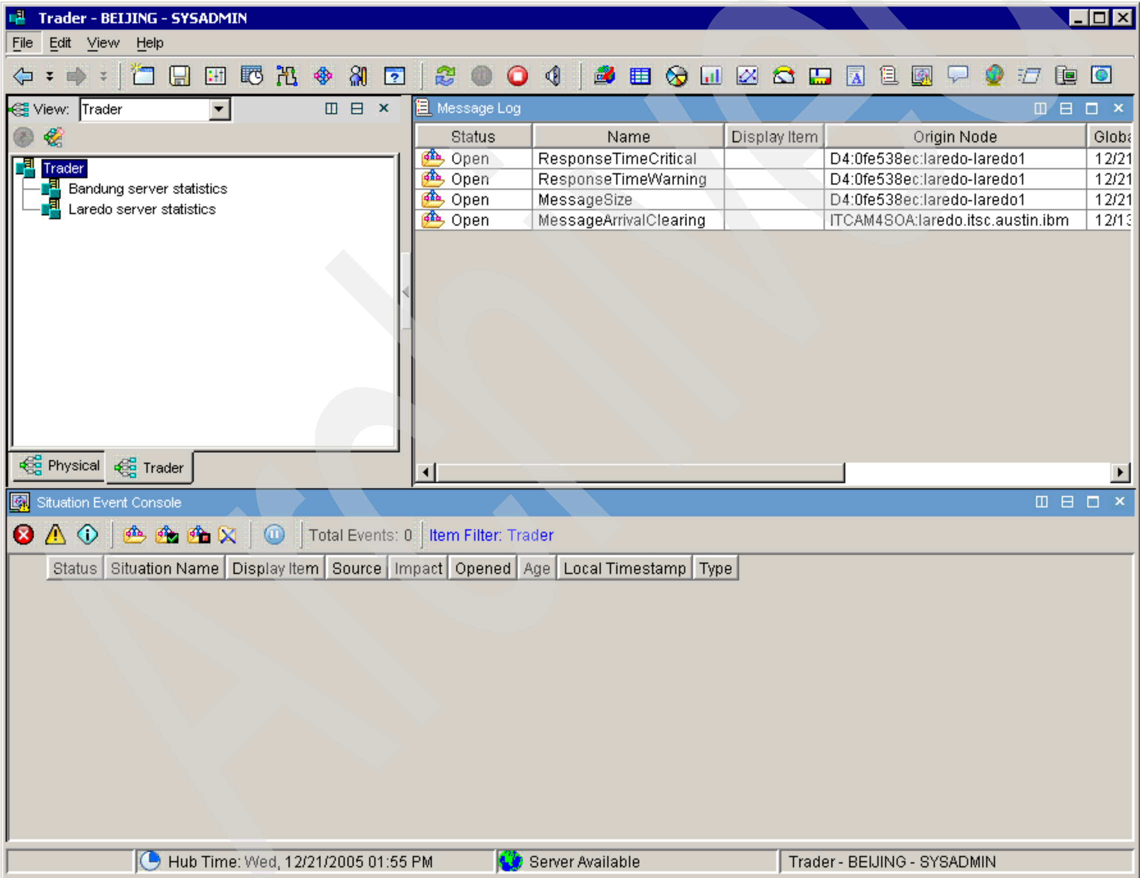


Figure 11-12 Trader workspace

- The laredo workspace contains the information for laredo. The final workspace contains:
  - WebSphere Application Server statistics from ITCAM for WebSphere: CPU usage, memory usage, transaction rate, transaction response time
  - Response time information from ITCAM for Response Time Tracking that shows the performance of the trader servlets that run on the TraderClientWeb application
  - Web services client information from ITCAM for SOA: message rate and response time

  Figure 11-13 shows the completed laredo workspace.

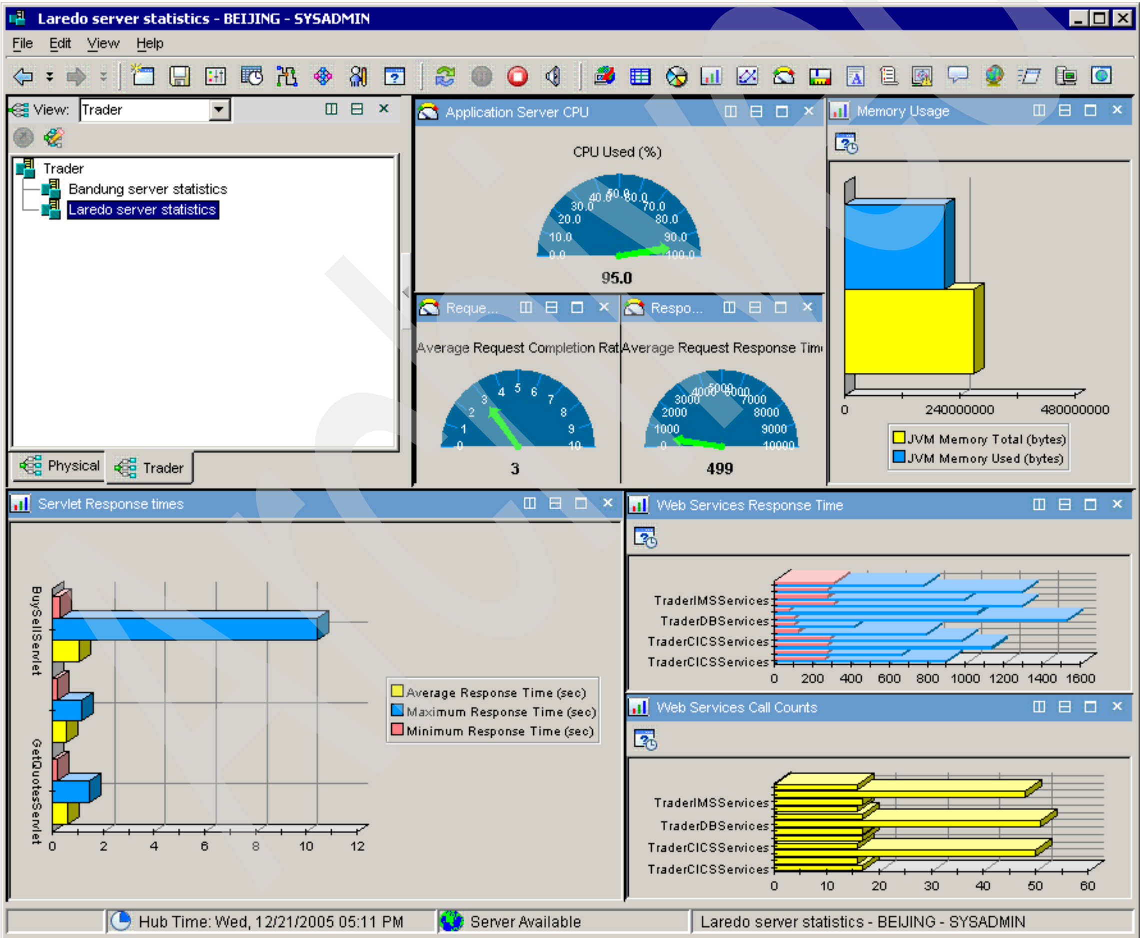


Figure 11-13 Laredo workspace

- The final bandung workspace contains this information for bandung:
  - WebSphere Application Server statistics from ITCAM for WebSphere: CPU usage, memory usage, transaction rate, transaction response time.
  - Response time information from ITCAM for Response Time Tracking that shows the performance of the trader servlets that run on the Trader*Services applications. This is typically called directly from the Java application because requests from laredo have been correlated to the calling servlets. Also, we monitor requests from Trader*Web applications.
  - Web services client information from ITCAM for SOA: message rate and response time.

Figure 11-14 shows the completed bandung workspace.

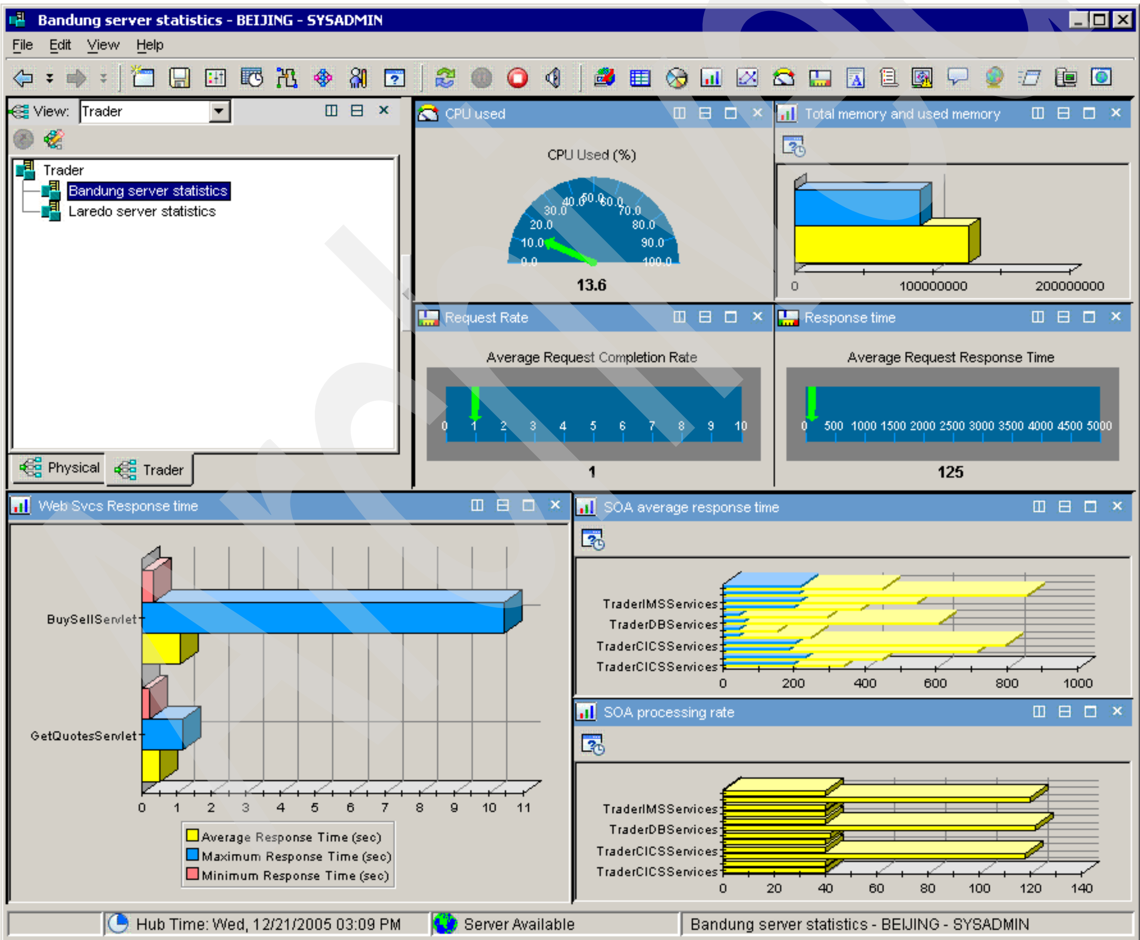


Figure 11-14 Bandung workspace

The workspace is defined by dissecting the workspace area using the split vertical button or split horizontal button. For each area, we then populate it with the appropriate type of portlet that we wanted to fit in. Figure 11-15 indicates the available components.

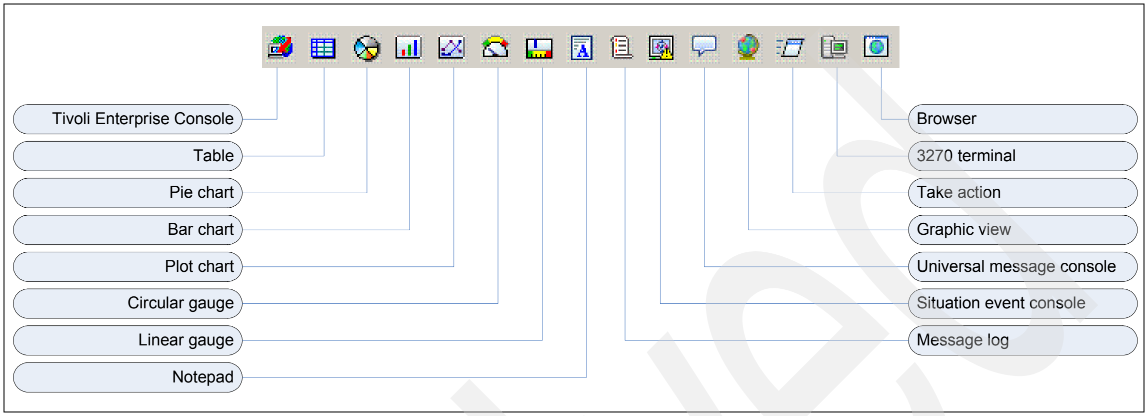


Figure 11-15 Portlet components

We describe the building of some of the workspace portlets in the following sections.

11.3.4 Defining the trader main view

The main view of the trader workspace is similar to the Enterprise workspace in the physical view. It contains the situation event console and message log. This display is adequate if you have already tuned your system and have the appropriate situations defined with the appropriate thresholds. When most of the definitions are valid, you should not get any false alarms or silent problems.

Both the message log and the situation event console are inserted into the area by clicking the appropriate icon and then clicking the area that you want to assign them. There is no real customization for these types of portlets.

11.3.5 Defining the bandung and laredo workspaces

The bandung and laredo workspaces shown in Figure 11-13 on page 585 and Figure 11-14 on page 586 are similar, so we discuss them together.

Data portlets are built from queries. However, the more queries that we submit in a single page, the more processing it introduces to the system. In designing the portlets, we take into consideration the number of queries that we use and the possibilities of using an IBM-supplied query.

Figure 11-16 shows our workspace with the areas identified.

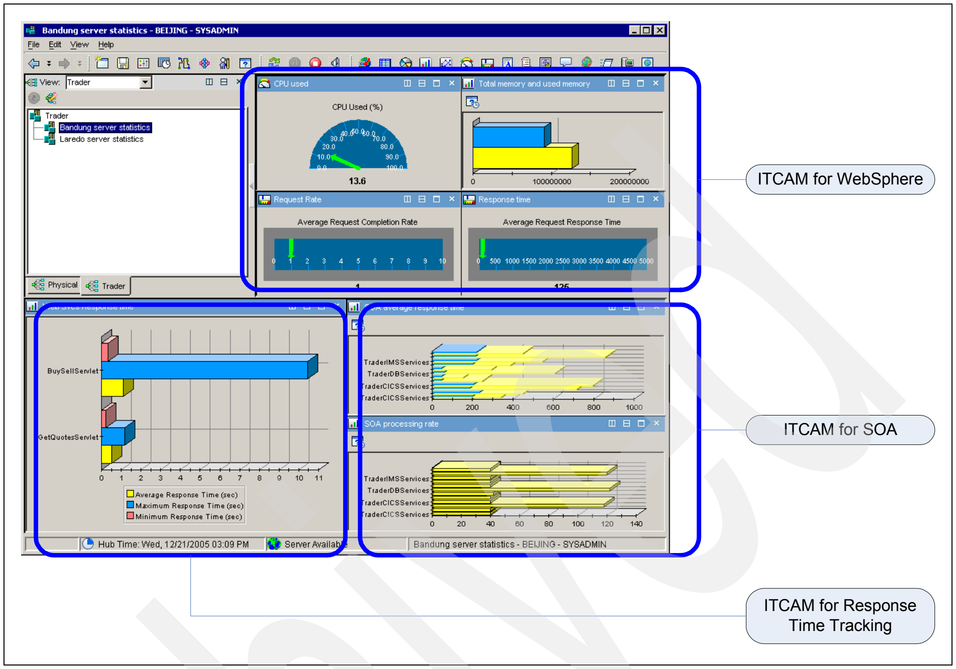


Figure 11-16   Workspace areas

We used these queries for our bandung and laredo workspaces:

- ITCAM for WebSphere information

  From the WebSphere Application Server queries, we use the following queries:

  – Application_Server attribute group, with the existing application server query. This query provides CPU usage percentage and memory usage information (total, used, and free memory). The CPU usage is shown as a circular gauge, and the total memory and memory used in a bar chart.

  – Request_Times_and_Rates attribute group, with the existing request time and rates query that provides average response time and request rate information. These are displayed as linear gauges.

- ITCAM for Response Time Tracking information

  From the response time tracking queries, we cannot use the available queries because the original workspaces are mostly accessed through links. (See 6.10, “Working with Tivoli Enterprise Portal” on page 326.) Links collect information from previous stages to present information. We create new queries, one for bandung and one for laredo, to present the information from the ITCAM_TT_Policy_Status attribute group. We copy the response time agent policy status query to our own query. See “Creating a new query” on page 596.

- ITCAM for SOA information

  From the service management agent environment, under the Services_Inventory attribute group, we can retrieve Web services information. We want to show the response time and invocation rate of Web services. We can either use two existing queries for the same attribute group, which means that data collection will be performed twice, or create a new query that selects the information that we need. We decided to create our own query to collect the information that we use.

With this design, the workspace for bandung and laredo uses only four queries to retrieve information. One goes to ITCAM for Response Time Tracking agent, and three go to the application server machine for execution by the ITCAM for SOA agent and ITCAM for WebSphere agent.

Setting a query portlet

This is the procedure for setting a query portlet:

1. Select the appropriate portlet type from the toolbar and click the workspace area that you want to customize.
2. A prompt asks whether to assign a query (Figure 11-17). Click **Yes**.

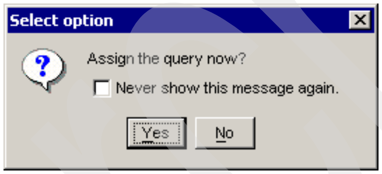


Figure 11-17 Query assignment confirmation

3. On the portlet setting page, click **Click here to assign a query** (Figure 11-18).

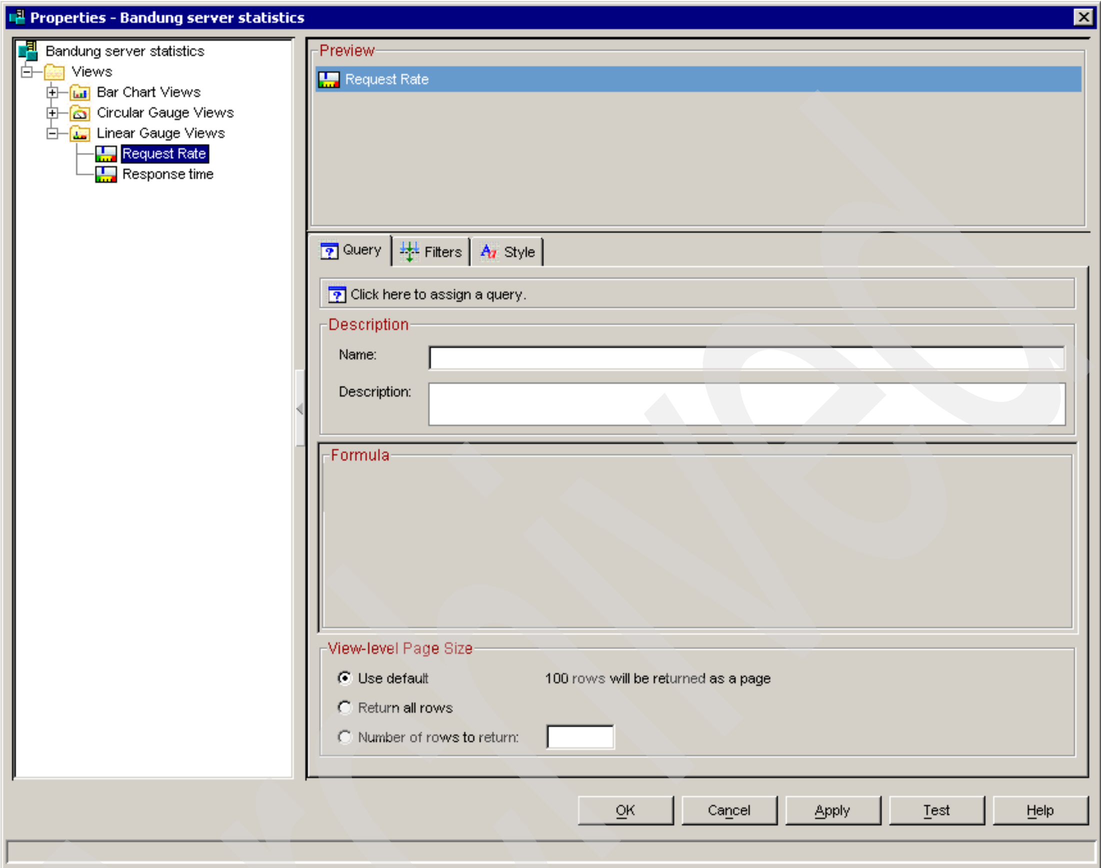


Figure 11-18 Empty portlet property page

4. When you reach the Query Editor page, select the query that you want to assign. Or you can create a new query, as discussed in “Creating a new query” on page 596. Figure 11-19 shows the example query for request times and rates. Click **OK** to select the query.

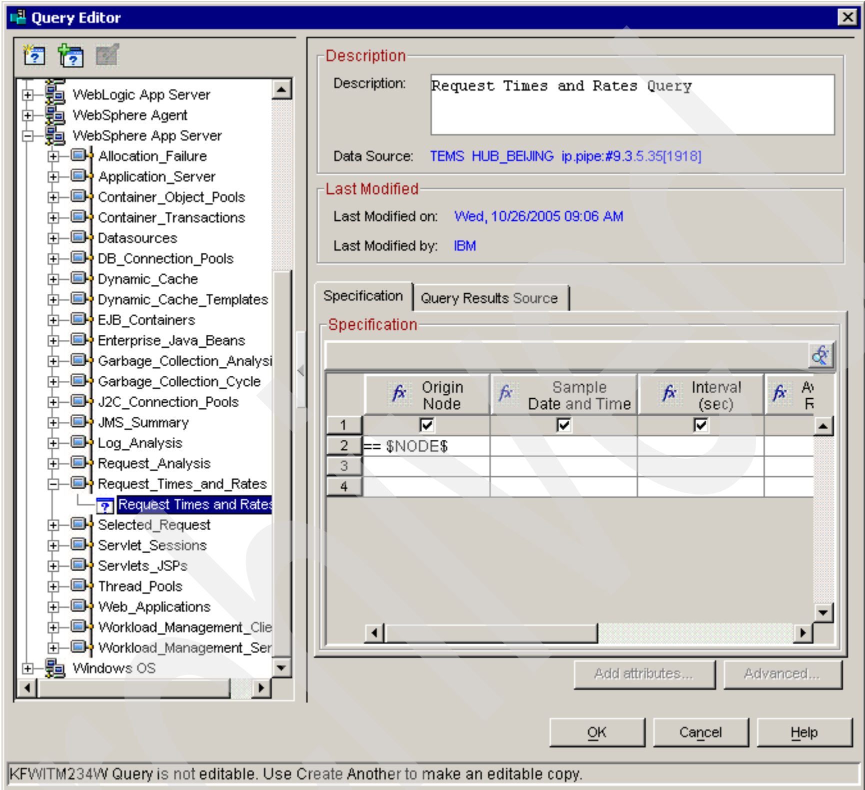


Figure 11-19 Query editor

5. Back on the portlet property page, select the **Filter** tab, which enables you to select the columns (attributes) that you want to be displayed on the portlet. Some portlets support multiple columns, such as table, bar, and plot charts, but gauges support only a single column. Select the column by selecting the check box, as shown in Figure 11-20. If your query is valid and there is an appropriate provider for data, you will see a snapshot of data for the query for your reference.

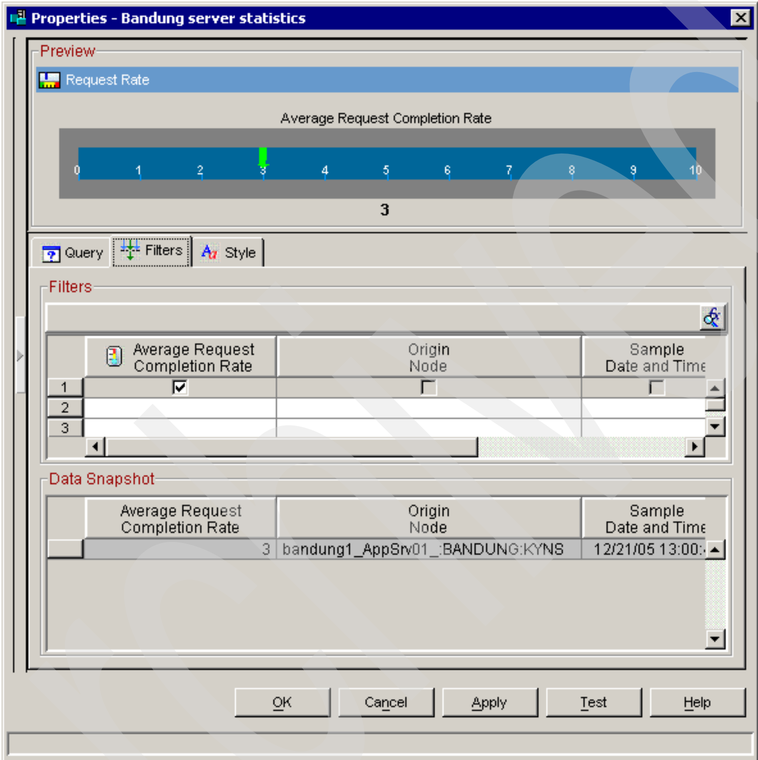


Figure 11-20 Portlet filter

6. You can customize the appearance of the portlet using the Style tab. First, change the heading text, which is provided on the initial page, as shown in Figure 11-21.

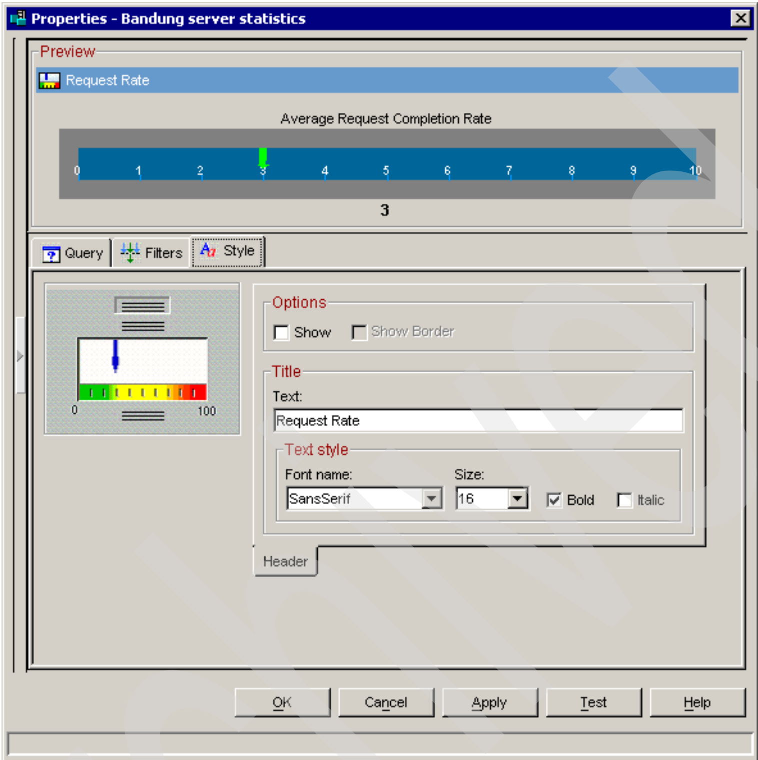


Figure 11-21 Heading text

7. Customize chart-specific attributes, such as:

   – For the circular gauge, customize the shape and value range of the data, as shown in Figure 11-22.

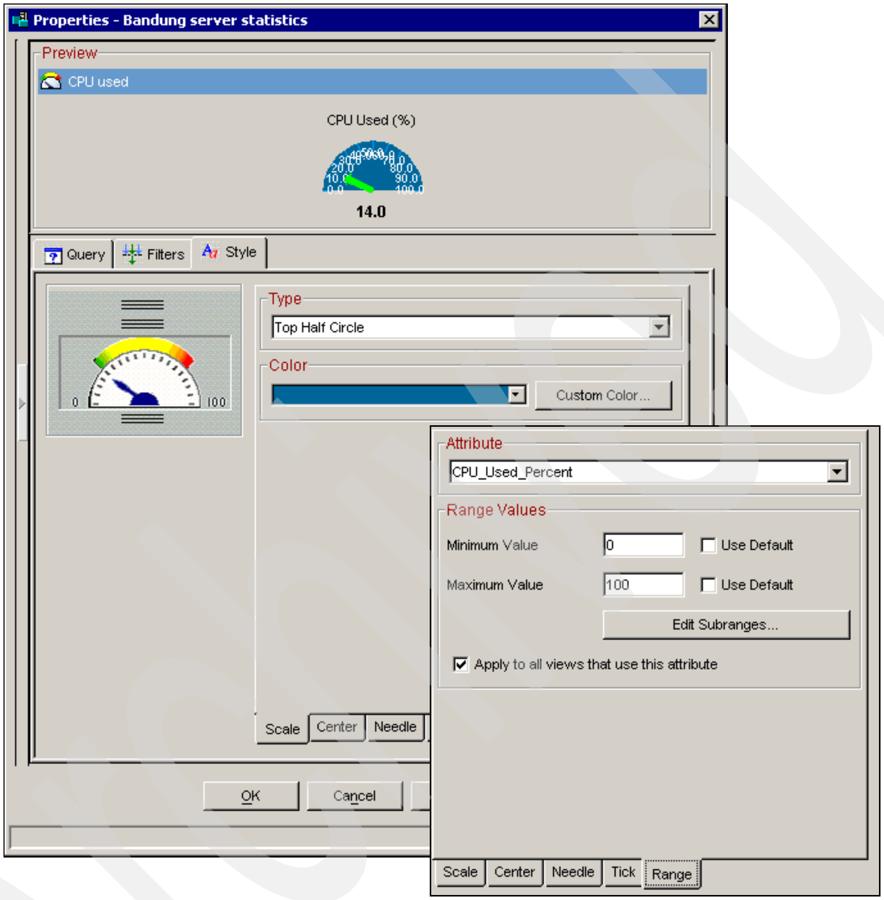


Figure 11-22 Circular gauge setting

– For the linear gauge, customize the orientation and range of data, as shown in Figure 11-23.

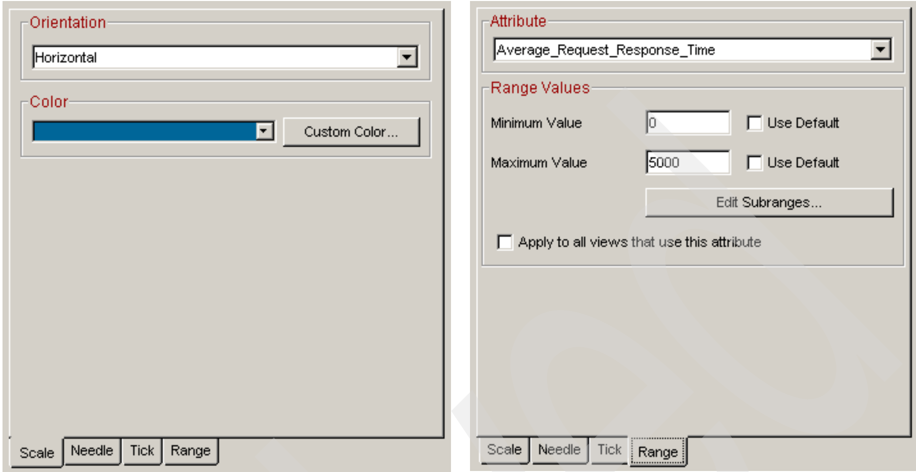


Figure 11-23 Linear gauge settings

– For the bar chart, customize the orientation and axis labels, as shown in Figure 11-24.

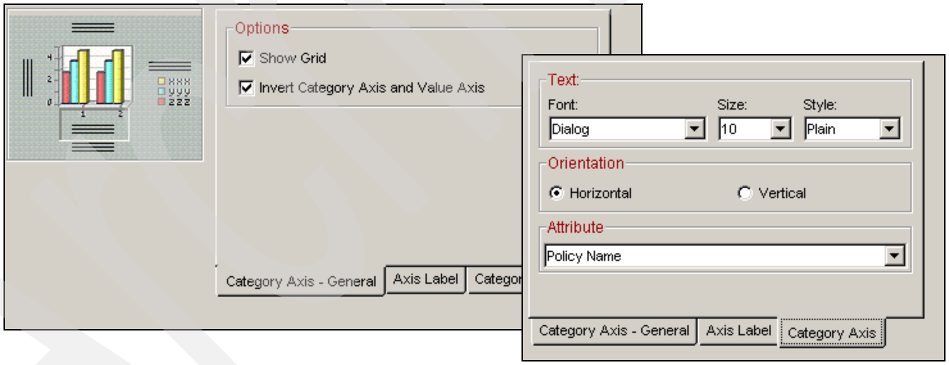


Figure 11-24 Bar chart settings

You can also change the legend text and position, as shown in Figure 11-25.

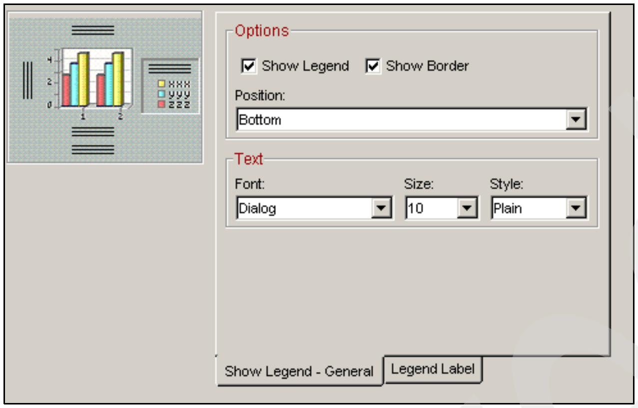


Figure 11-25 Legend customization for bar chart

8. Click **OK** to save the portlet property and select **File** → **Save Workspace** to save the workspace.

Creating a new query

As discussed in 11.3.5, “Defining the bandung and laredo workspaces” on page 587, we create a new query to optimize the workspace and provide data for our portlet.

From the query editor, you can either create a completely new query or copy an exiting query . You must assign a name for the new query, as shown in Figure 11-26.

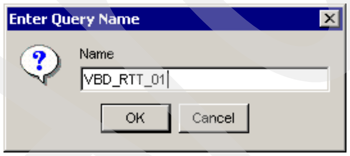


Figure 11-26 Name for new query

The query specification defines both the selected attributes and the row selection conditions. Some conditions are mandatory, and typically they are selected from a variable. A variable is specified by enclosing it with dollar signs ($). You can substitute these variables with a fixed value. Conditions specified in the same row represent an AND operation, and conditions specified on different rows represent an OR operation. See a sample specification in Figure 11-27.

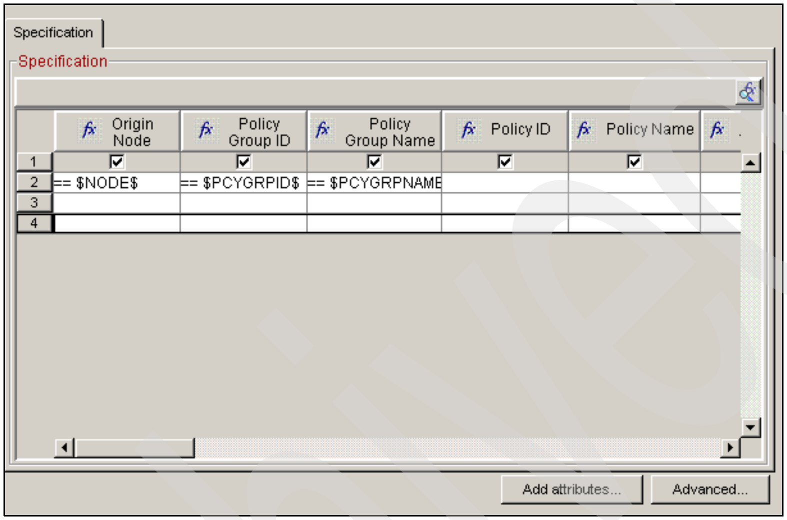


Figure 11-27 Query specification

The query for ITCAM for Response Time Tracking that we create copies from the existing query shown in Figure 11-27 on page 597. We modify the query because the PCYGRPID and PCYGRPNAME variables will not be available from a simple workspace. They can only be retrieved from a linked workspace. For laredo, we retrieve all policy response time information for the policy inside the Trader_Web_appl policy group, as shown in Figure 11-28.

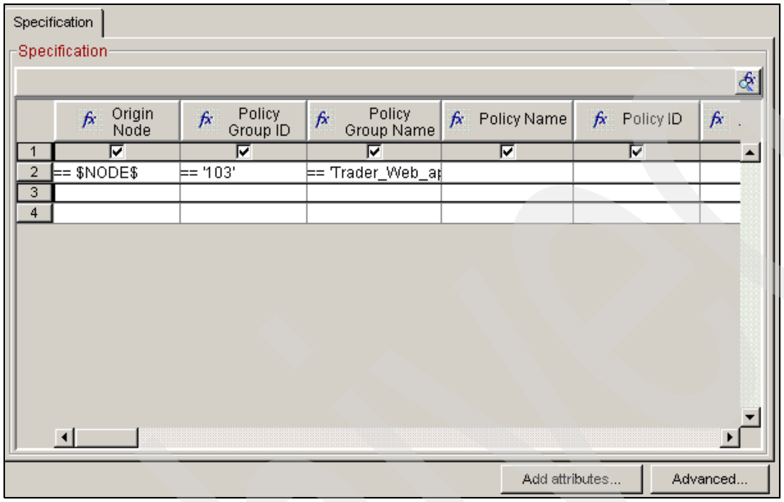


Figure 11-28 Laredo policies

Retrieve the policy group ID number from the ITCAM for Response Time Tracking dashboard report by hovering the cursor over the policy group to see the status bar, as shown in Figure 11-29.

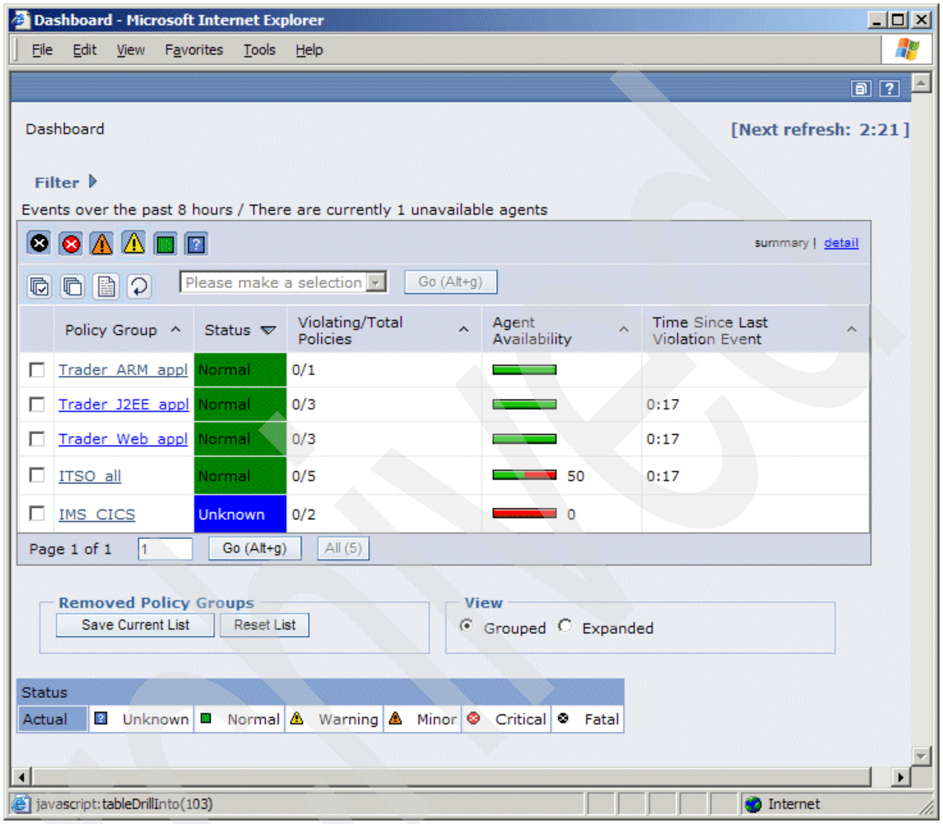


Figure 11-29 Getting policy group ID

The ITCAM for SOA query that we create selects only specific columns from scratch from the Services_Inventory attribute group. Use the following procedure:

1. Create a new query using the **Create Query** button. Assign the name and category of the query, as shown in Figure 11-30.

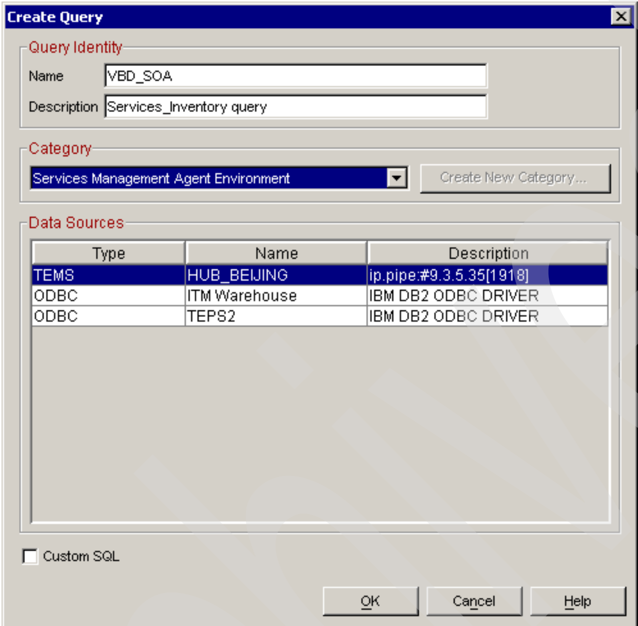


Figure 11-30 Query name and category

2. Select the attributes that you want to collect, as shown in Figure 11-31.

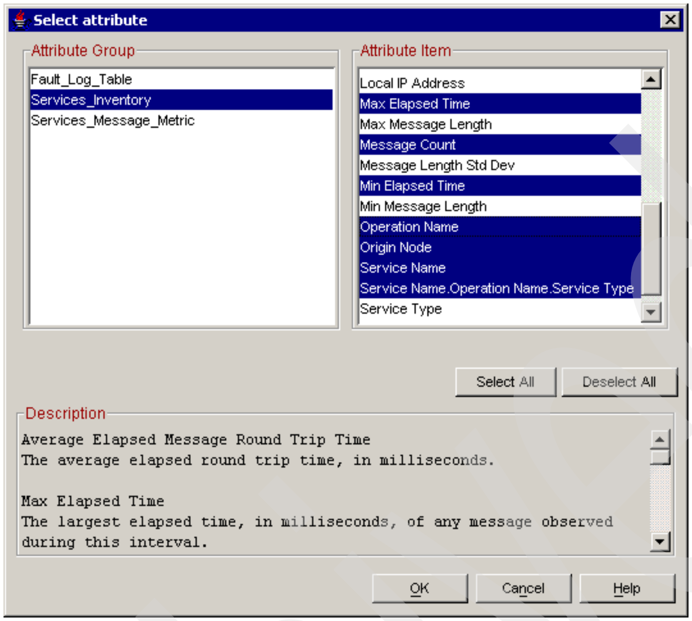


Figure 11-31 Query attributes

3. The query appears in the Query Editor window, where we provide the selection conditions. For the Services_Inventory attribute group, specify at least the origin node to be **$NODE$**. Figure 11-32 shows the condition that we use.

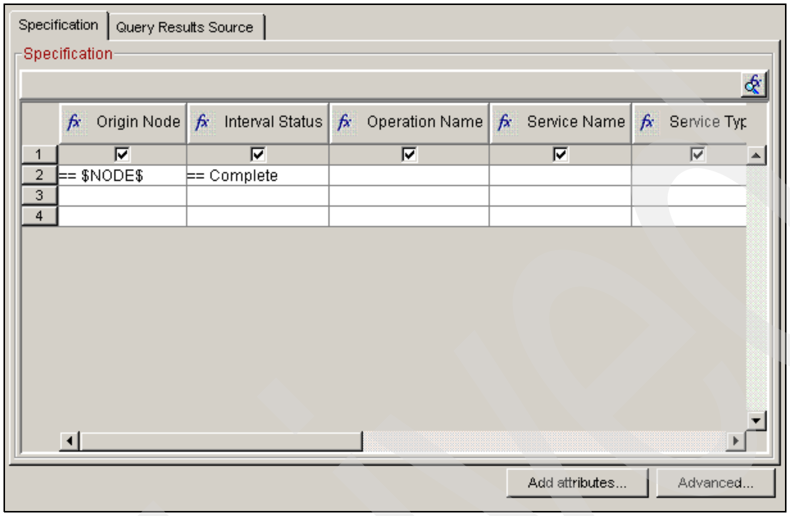


Figure 11-32 Setting condition

4. You can further select the attributes to be included in the query by selecting or deselecting the attributes. For attributes that you deselected in a previous session, you can add them by clicking the **Add attributes** button.

11.4 Workflows and situations

A situation is an automatic monitoring of the system that can be raised based on a certain condition. A situation runs in the background based on a predefined interval. It is useful for getting basic health information from a Tivoli Enterprise Monitoring Agent.

Workflow is also called *policy*. The terms are used interchangeably on the Tivoli Enterprise Portal. Workflows enable some level of automation for IBM Tivoli Monitoring V6.1.

This section discusses the creation of custom situations and incorporates them into a workflow for monitoring our trader application environment. Because the situation will be used in a workflow, these situations will not be auto-started. Only the workflow has to be auto-started, and this starts the situation.

The discussion in this section includes:

- 11.4.1, “Situation basics” on page 603
- 11.4.2, “Workflow basics” on page 603
- 11.4.3, “Trader workflow design” on page 605

11.4.1 Situation basics

A situation is a conditional expression that is evaluated at certain intervals. The situation can evaluate to true. This is considered a situation change event.

A situation evaluates the attributes in an agent from an attribute group. Because an attribute group is considered a table and attributes are its columns, the situation definition contains the following components:

- The name of the situation
- The category of the situation
- The attribute group that will be evaluated
- The row selection condition from the attribute group

The selection condition can contain multiple expressions:

- Filtering for certain types of data, such as servlet name, policy name, or other attributes.
- Checking data value for a selected row. This checking can be considered a threshold for the data in the table. Some checking uses an aggregation function such as count, maximum, or minimum. This type of checking is performed at the Tivoli Enterprise Monitoring Server.

11.4.2 Workflow basics

Workflows can automate and correlate several situations. A situation can evaluate attributes from only a single attribute group. Correlation from several situations is possible using a workflow. This allows a more intelligent approach to event analysis.

Table 11-1 shows workflow components.

Table 11-1 Workflow components

Component icon	Description
	Fire the situation collection when the policy executes. This does not create an additional collection of the situation.
	Schedules the situation for collection. In IBM Tivoli Monitoring V6.1, the situations are removed from the synchronization mechanism, but if the original situation is collecting, this will not generate an additional collection.
	Perform a predefined action for the situation or write a message to the message log.
	Suspend the execution of the workflow for a certain time range.
	Start or stop a workflow. This can be used to trigger another workflow to start (for example, to change the workflow from workdays to a weekend workflow).
	Start or stop a new situation to be executed. The new situation can be used to indicate a time-dependent threshold.
	Decision point that communicates a message to a group of users. When this is executed, someone has to answer the prompt to determine whether the workflow will continue. This is available only for Candle Management Workstation users.
	This is an extension component to wait for a situation to become false.
	Send an SNMP event.

A workflow can also be a huge source of additional work in situation processing. This can be more than double the work of collecting data for situations because they duplicate the collection for the situation to drive alerts to the Tivoli Enterprise Portal Server and might not be eligible for situation synchronization.

When three situations that are eligible for synchronization in one synchronized data collection are added to a workflow, the situation will not be collected in a single synchronized collection. In IBM Tivoli Monitoring V6.1, these collections are performed once for all instances of a situation.

In a workflow, the *evaluate a situation now* activity does not start the situation, it just performs a one-time sampling of data. However, the *wait until a situation is true* activity does.

11.4.3 Trader workflow design

The workflow that is built for the trader application is constructed based on the diagram in Figure 11-33. For a problem in response time, we want to pinpoint the source of the problem, whether the front end, middle tier, or back end. This problem is because the front end uses the same servlet to access different back-end systems.

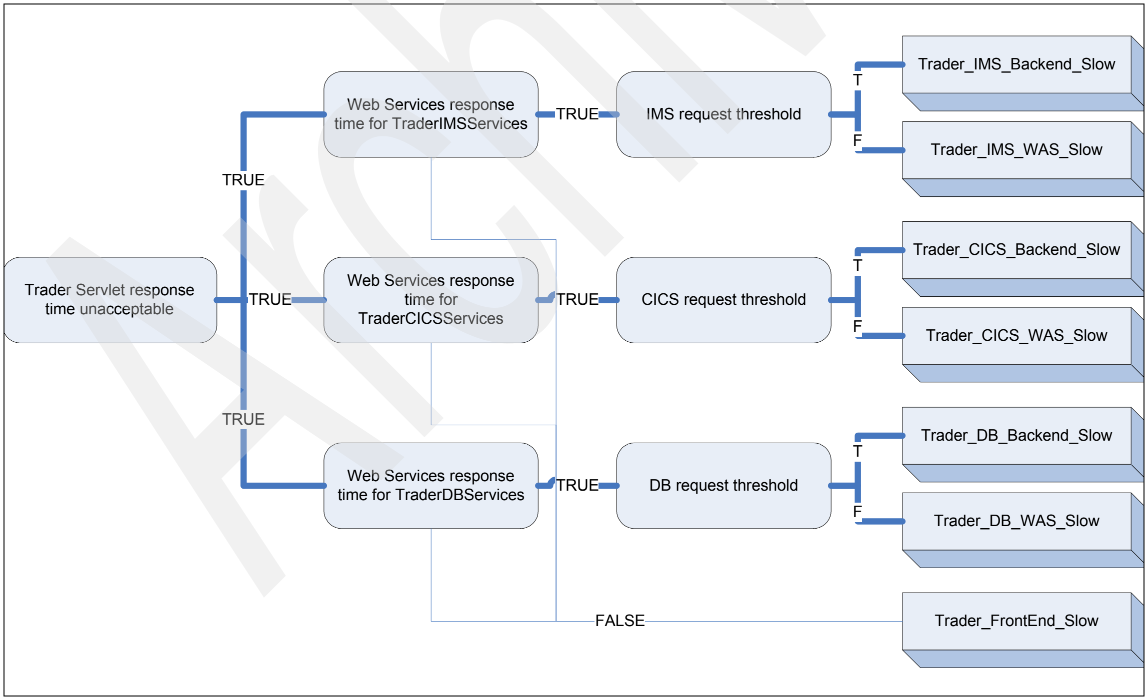


Figure 11-33 Trader workflow

To accomplish this workflow, we need to define 21 new situations for our sample and a workflow. The rounded rectangles represent conditional situations. We need two situations for each condition, true and false. The raised rectangles represent new events. Each represents one new situation. Table 11-2 lists the new situations.

Table 11-2 New situations list

Name	Category	Attribute group	Condition
TraderServlet_slow	Response Time Tracking	ITCAM_TT_Policy_Status	Policy Group Name == Trader_Web_appl Policy Name == SCAN(Servlet) Average Response Time >= 5
TraderIMS_WS_slow	Service Management Agent Environment	Service_Inventory	Service Name == TraderIMSServices Average Elapsed Message Round Trip Time >=4000
TraderCICS_WS_slow	Service Management Agent Environment	Service_Inventory	Service Name == TraderCICSServices Average Elapsed Message Round Trip Time >=4000
TraderDB_WS_slow	Service Management Agent Environment	Service_Inventory	Service Name == TraderDBServices Average Elapsed Message Round Trip Time >=4000
TraderIMS_req_pct	WebSphere App Server	Request_Analysis	Request Detail = STR(2,TraderIMSServices) JCA Time (%) >= 70.0
TraderCICS_req_pct	WebSphere App Server	Request_Analysis	Request Detail = STR(2,TraderCICSServices) JCA Time (%) >= 70.0
TraderDB_req_pct	WebSphere App Server	Request_Analysis	Request Detail = STR(2,TraderDBServices) SQL Connect Time (%) >= 30.0 SQL Query Time (%) >= 30.0 SQL Update Time (%) >= 30.0
TraderServlet_recover	Response Time Tracking	ITCAM_TT_Policy_Status	Policy Group Name == Trader_Web_appl Policy Name == SCAN(Servlet) Average Response Time < 5
TraderIMS_WS_recover	Service Management Agent Environment	Service_Inventory	Service Name == TraderIMSServices Average Elapsed Message Round Trip Time <4000

Name	Category	Attribute group	Condition
TraderCICS_WS_recover	Service Management Agent Environment	Service_Inventory	Service Name == TraderCICSServices Average Elapsed Message Round Trip Time <4000
TraderDB_WS_recover	Service Management Agent Environment	Service_Inventory	Service Name == TraderDBServices Average Elapsed Message Round Trip Time <4000
TraderIMS_req_pct_recover	WebSphere App Server	Request_Analysis	Request Detail = STR(2,TraderIMSServices) JCA Time (%) < 70.0
TraderCICS_req_pct_recover	WebSphere App Server	Request_Analysis	Request Detail = STR(2,TraderCICSServices) JCA Time (%) < 70.0
TraderDB_req_pct_recover	WebSphere App Server	Request_Analysis	Request Detail = STR(2,TraderDBServices) SQL Connect Time (%) < 30.0 SQL Query Time (%) < 30.0 SQL Update Time (%) < 30.0
TraderIMS_backend_slow	Universal Agent	Universal_Time	Hour>=0
TraderIMS_WAS_slow	Universal Agent	Universal_Time	Hour>=0
TraderCICS_backend_slow	Universal Agent	Universal_Time	Hour>=0
TraderCICS_WAS_slow	Universal Agent	Universal_Time	Hour>=0
TraderDB_backend_slow	Universal Agent	Universal_Time	Hour>=0
TraderDB_WAS_slow	Universal Agent	Universal_Time	Hour>=0
TraderWAS_frontend_slow	Universal Agent	Universal_Time	Hour>=0

Defining situations

Situations are defined using the situation editor that is invoked using the button. From the situation editor main window shown in Figure 11-34, we create a new situation by right-clicking the appropriate category and selecting **Create New**.

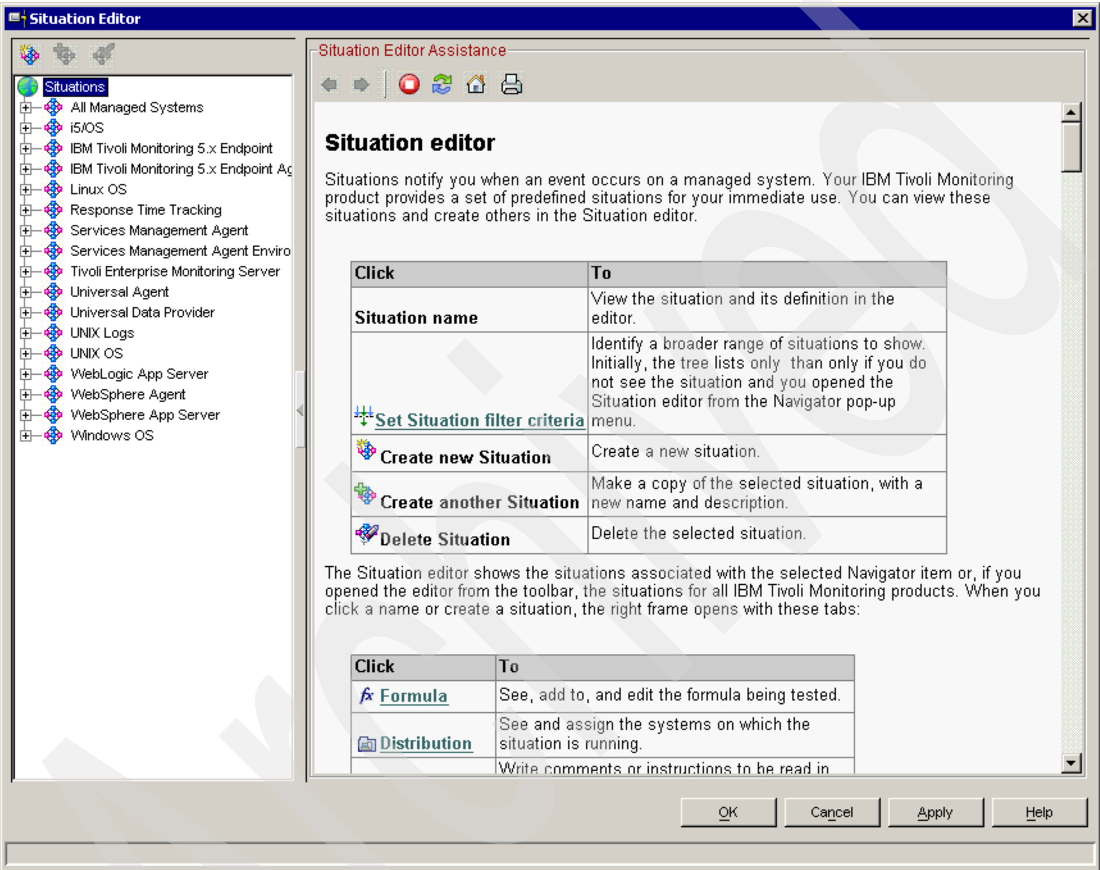


Figure 11-34 Situation editor

When the Create Situation window opens, as shown in Figure 11-35, enter the name and description of the situation. Click **OK**.

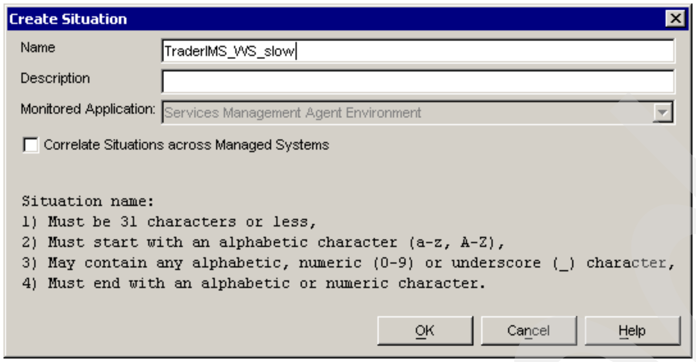


Figure 11-35 New situation

The next window enables you to select the attribute group and attributes that you want to use for the situation (Figure 11-36). Just select the attributes that you want to evaluate.

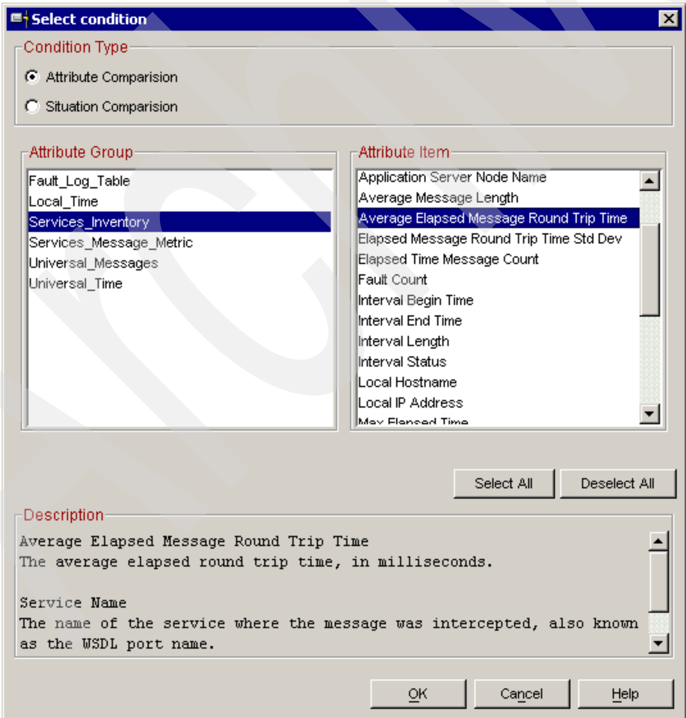


Figure 11-36 Selecting attribute group and attributes

When the attributes have been selected, click **OK** to see the situation formula definition. You must specify your condition here and select the distribution list on the next tab. The distribution list represents the Tivoli Enterprise Monitoring Agent that the situation will run on.

- Figure 11-37 shows the ITCAM for Response Time Tracking situation condition. Trader_Web_appl is the policy group where we collect the servlet response time listening policies. The target is the Primary:KHARTOUM:T2 node.

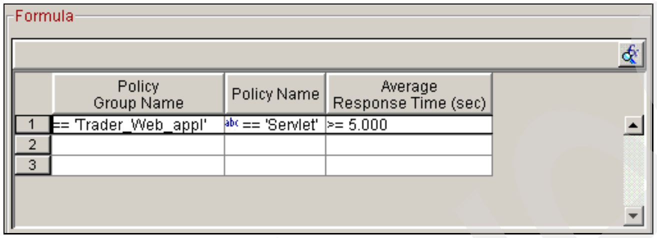


Figure 11-37 ITCAM for Response Time Tracking condition

- Figure 11-38 shows the ITCAM for SOA situation condition. This example shows the TraderIMSServices as the filter with a threshold of 4 seconds (4000 milliseconds). The distribution is to D4:2b251f23:bandung-bandung1, which is the ITCAM for SOA agent.

Figure 11-38 ITCAM for SOA condition

- Figure 11-39 shows the ITCAM for WebSphere situation condition. We use a substring to locate the TraderIMSServices and have the JCA percentage greater than 70%. The distribution target is bandung1_AppSrv01_:BANDUNG:KYNS.

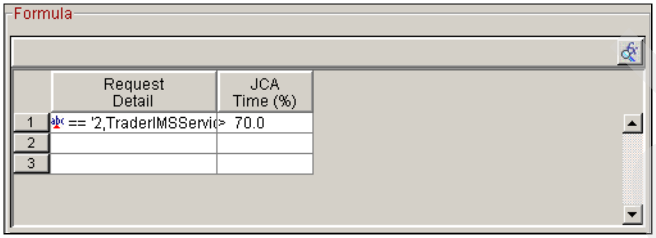


Figure 11-39 ITCAM for WebSphere condition

- Figure 11-40 shows the ITCAM for WebSphere situation condition for a SQL request with TraderDBServices.

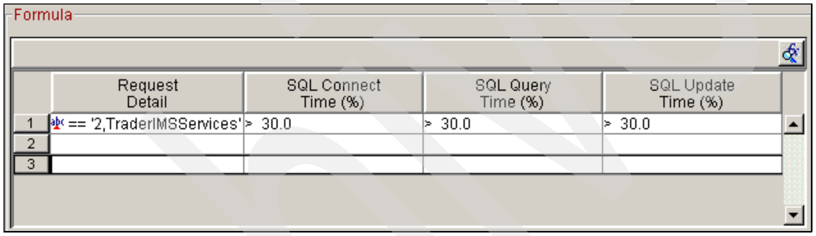


Figure 11-40 ITCAM for WebSphere for JDBC call condition

- The Universal Message situation that we provide just to get an event uses an always true condition with an hour for the evaluation time. We use *hour >= 0* as the condition.

11.4.4 Defining the workflow

Now that we have defined all of the new situations, we are ready to define the workflow. You invoke the workflow editor using the workflow editor button . Figure 11-41 shows the initial workflow editor window.

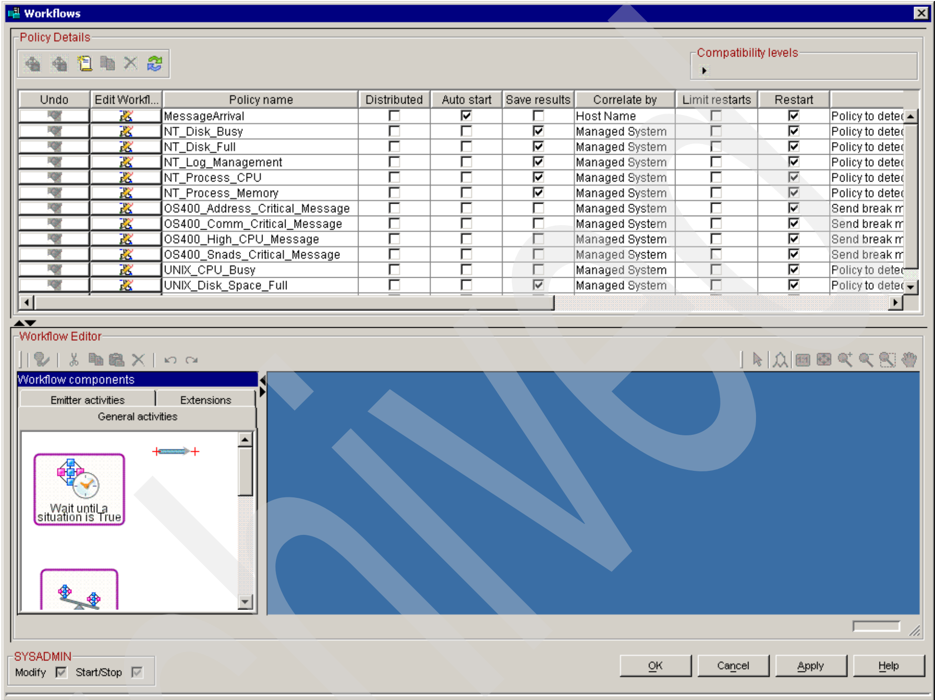


Figure 11-41 Workflow editor initial window

Click the create new workflow button and specify the workflow name. You are presented with the edit display, as shown in Figure 11-42. If you need to get out and come back to the workflow editor, click the edit workflow button to start editing your workflow.

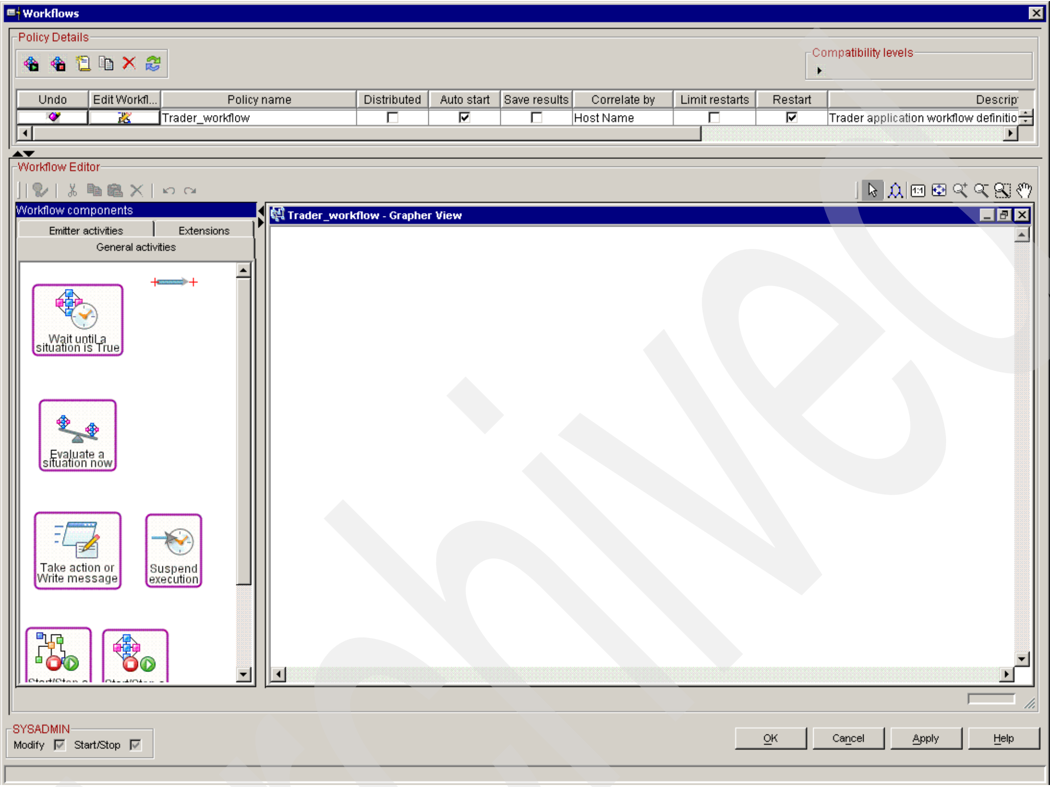


Figure 11-42 New empty trader workflow

As indicated in Figure 11-33 on page 605, the workflow checks the existing situations that we define. First, we analyze the definition for the IMS application. Figure 11-43 shows the workflow with only the IMS components.

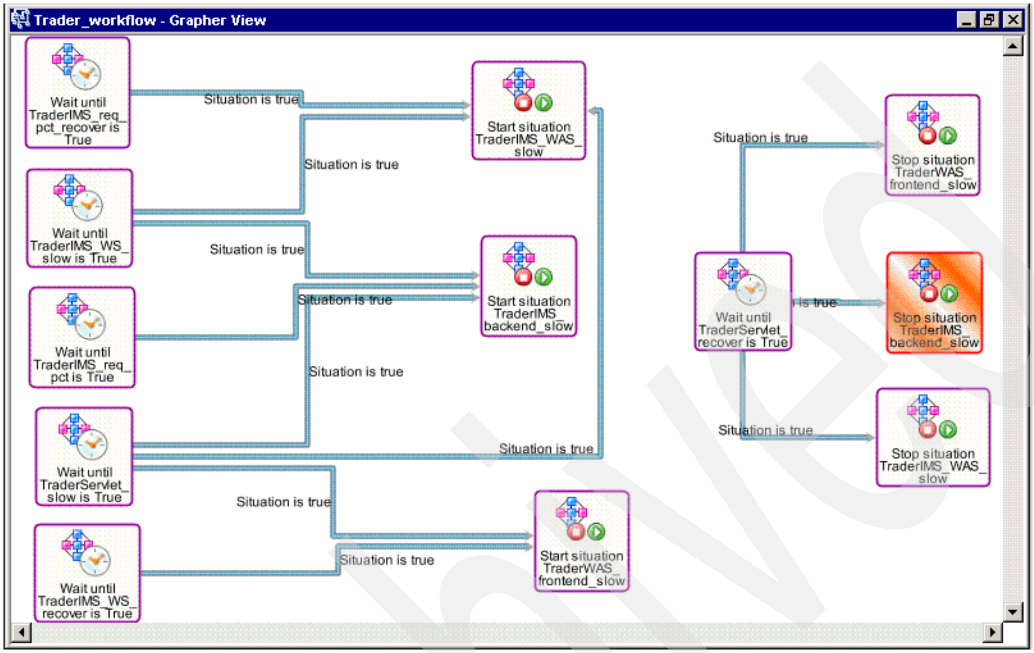


Figure 11-43 Workflow with only IMS components

As shown in Figure 11-43, we want to start and stop TraderIMS_WAS_slow, TraderIMS_backend_slow, or TraderWAS_frontend_slow, depending on the condition. The primary indication is the TraderServlet situation. When the servlet response time exceeds the threshold, it starts the trigger of events. When the TraderServlet response time drops below the threshold as indicated by TraderServlet_recover situation, all generated situations are stopped.

The decision about which component is slow — whether it is the front end, back-end WebSphere Application Server, or the back-end server — is based on Table 11-3.

Table 11-3 Conditions for problem sources

Problem source	Servlet is slow	Web services is slow	JCA call percentage high
Front end	Yes	No	-
WebSphere Application Server back end	Yes	Yes	No
Back-end server	Yes	Yes	Yes

We then added the components for the CICS and DB services. Figure 11-44 shows the recovery part for stopping the situations.

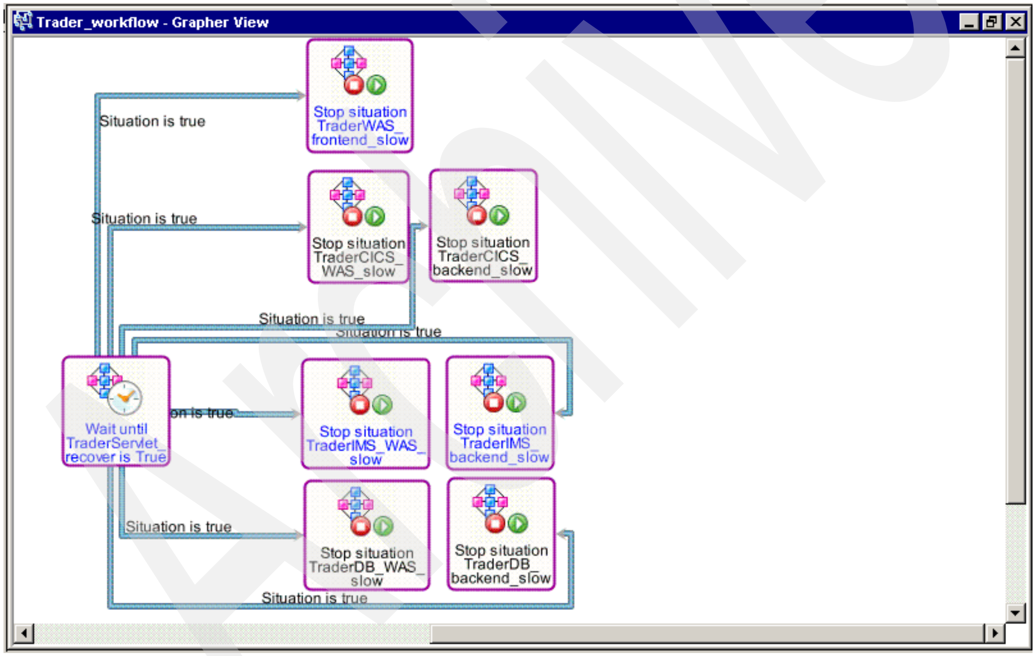


Figure 11-44 Completed workflow: recovery

Starting the situation is much more complex. We show the completed pieces separately. Figure 11-45 shows the database portion. The external links are from the TraderServlet_slow situation.

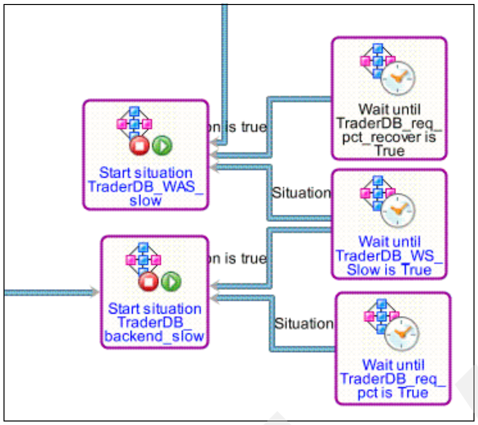


Figure 11-45 Completed workflow: failure for database

Figure 11-46 shows the CICS portion. Again, the external links are from the TraderServlet_slow situation.

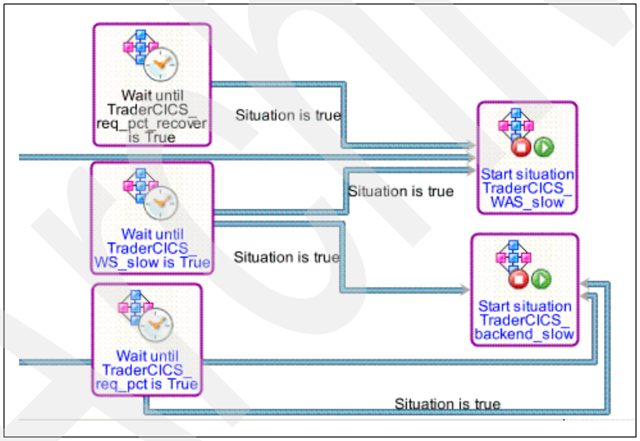


Figure 11-46 Completed workflow: failure for CICS

For the front-end failure, the CICS and DB Web services recover situations are used, as shown in Figure 11-47.

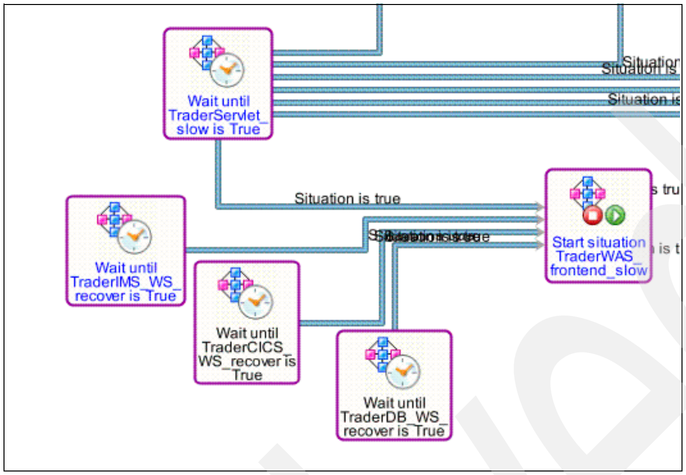


Figure 11-47 Completed workflow: failure for front end

Figure 11-48 shows the overall workflow. This has been rearranged for readability and display.

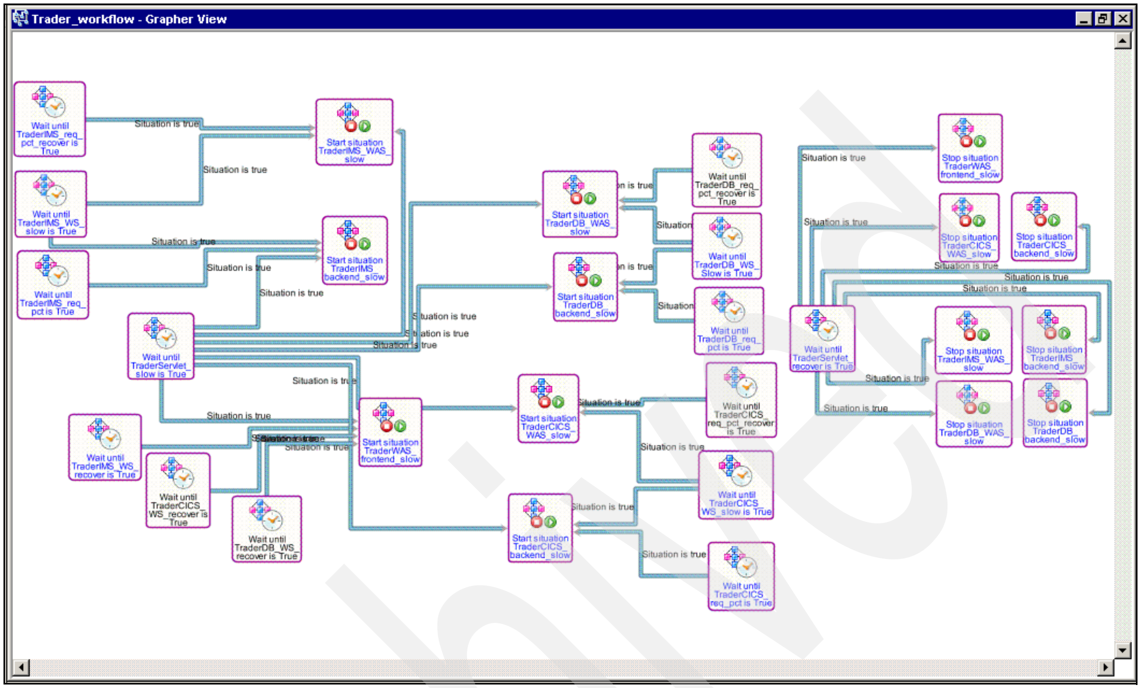


Figure 11-48 Completed workflow

A

Trader application description

This chapter explains the trader application. The discussion is divided into:

- "Application components" on page 620
- "Software requirements" on page 633
- "Installation procedure" on page 636

Application components

The trader application is a multi components composite application that runs on heterogeneous platforms and in an execution environment. It is a simple stock trading application that allows the user to list companies, get a quote, and trade stocks of the listed companies. Figure A-1 shows the trader application conceptual interfaces.

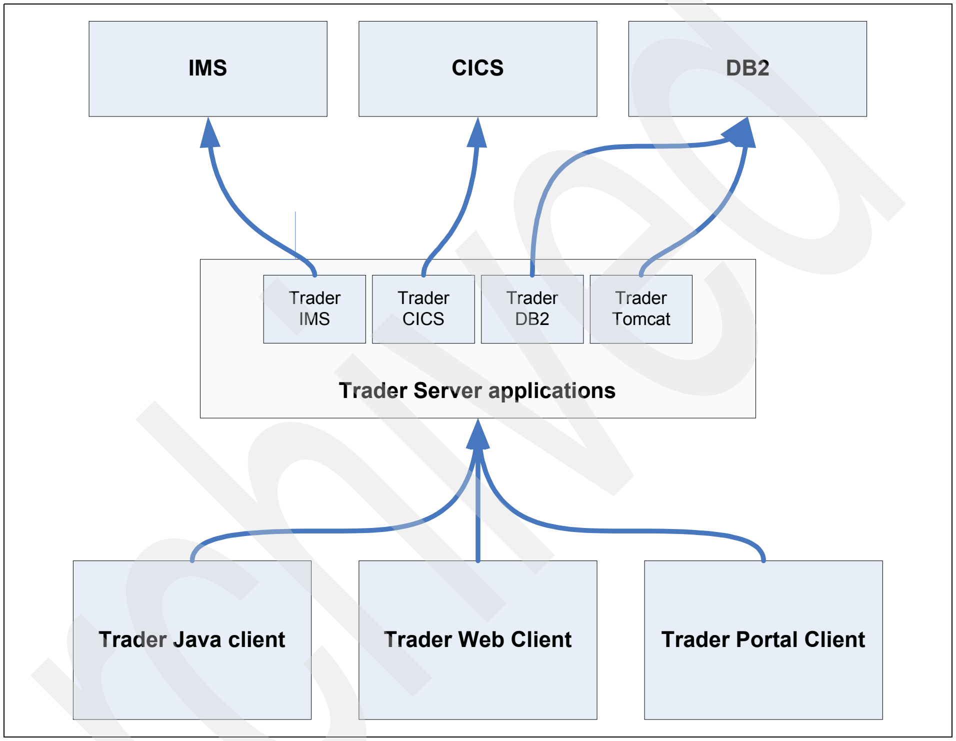


Figure A-1 The trader application

The trader application can be viewed as having a three-layer (three-tier) structure:

- The trader application has three types of front-end interfaces, which are a native Java client, a Web interface, and a Portal-based interface. Each of these interfaces has the ability to connect to the server application that provides the business logic. The connection to the server applications is based on Web service calls.
- The server applications are differentiated based on their different access methods to the underlying data, or the platform it resides on. These are J2EE-based applications that serve as Web Services providers.

- The back-end data storage can be IMS or CICS on z/OS or a DB2 database. The DB2 database can reside on z/OS or a distributed server. In our environment, we deploy the DB2 database in a distributed environment.

We discuss the components in the following sections:

- “Portal interface” on page 621
- “Front-end J2EE Web application” on page 623
- “Java desktop application” on page 626

Portal interface

The trader portal interface consists of three portlets that communicate with each other. Figure A-2 shows the portal client.

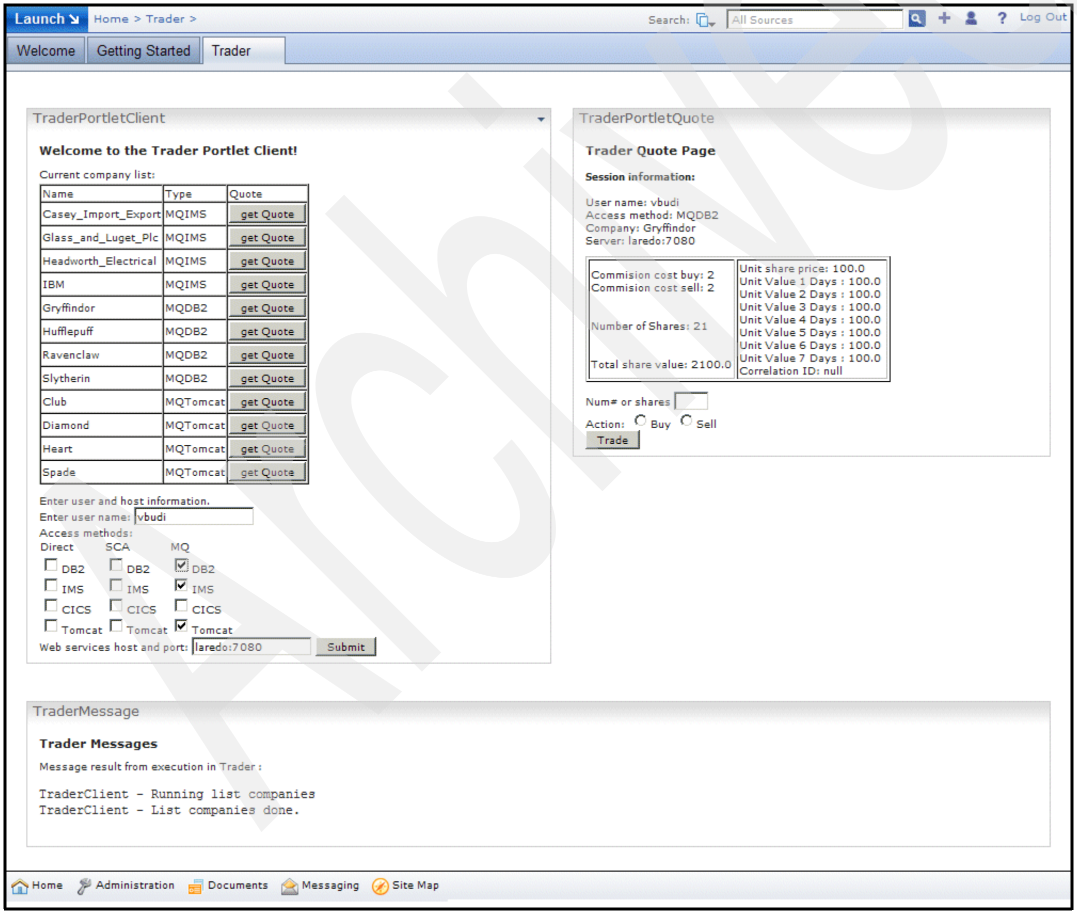


Figure A-2 The trader portal client

The trader client portlet is the main portlet that lets you query the available companies. If you request a quote for the company, the quote information is shown in the trader quote portlet. The trader quote portlet allows you to buy or sell stocks of the currently displayed company. The trader message portlet just collects and displays the messages from the application.

The trader client portlet requires you to specify the host and port of the back-end server that you connect to. This means that the connection only allows you to connect to the servers that resides on the same J2EE servers. This introduces the need to have a Web services mediation to pool all the connections. Then the mediation would be able to connect to the appropriate Web services provider.

The connections in the SCA column are used for connecting to a WebSphere Enterprise Service Bus mediation, and the connections in the MQ column are used for connecting to a WebSphere Message Broker mediation.

The portal application is distributed as a single Web archive file TraderPortletClient1.war.

Front-end J2EE Web application

The front-end Web application is developed using the Web services client wizard Trader*Services projects. The application consists of:

- Initial login page in login.html (Figure A-3).

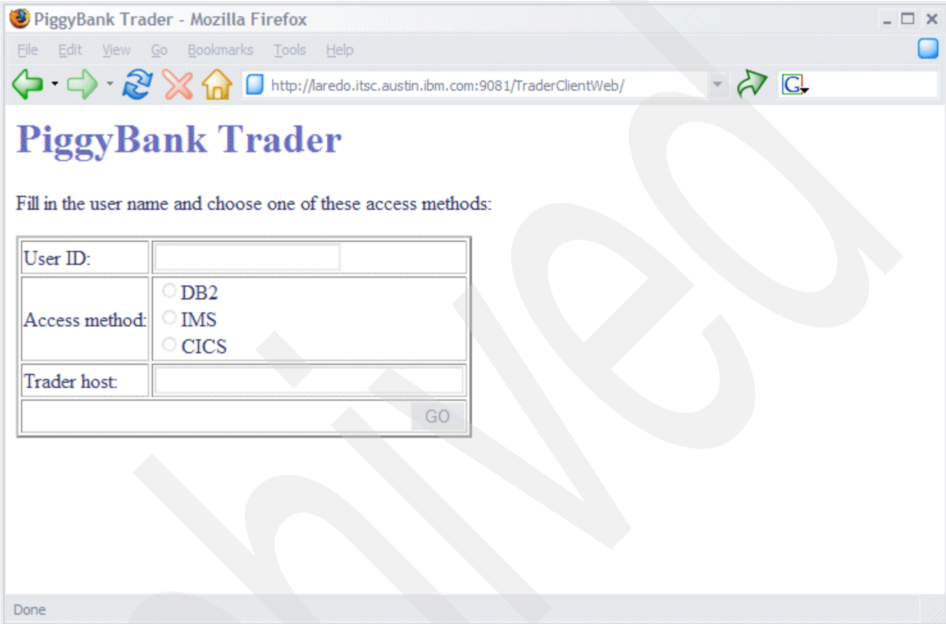


Figure A-3 Login page

> **Note:** The DB2, IMS, and CICS radio check boxes shown in Figure A-3 are not normally available to an end user. They are included on our sample application purely to highlight the possible back-end system. Similarly, a typical application would not select a target host, but this is shown here as part of our lab environment.

- ListCompanyServlet (Figure A-4): invokes the back-end ListCompany Web services.

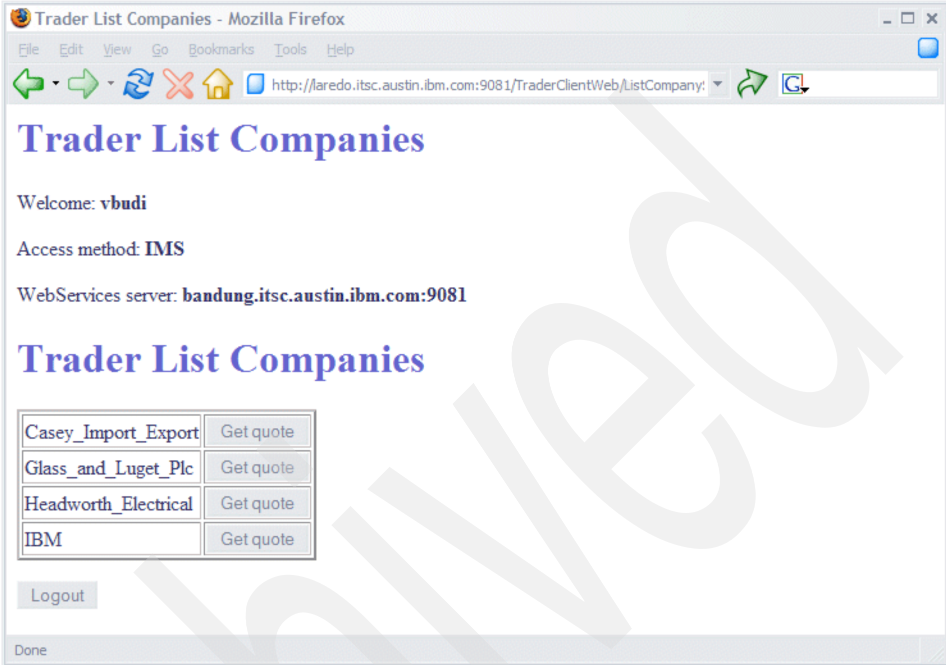


Figure A-4 List company page

- GetQuotesServlet (Figure A-5): invokes the back-end GetQuote Web services.

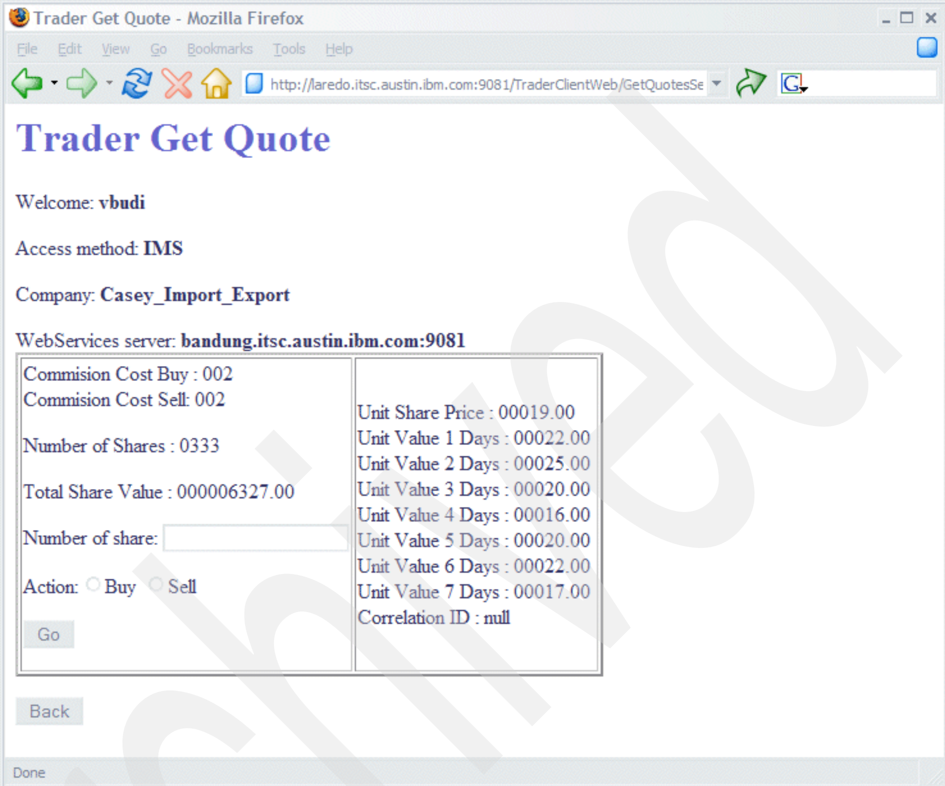


Figure A-5 Quotes window

- BuySellServlet: invokes either the buy or sell Web service.
- LogoutServlet: clears up the session bean.

We provide three types of enterprise application archive (ear) files for the client interface:

- TraderClientEAR: This ear file runs the TraderClientWeb application that provides the basic trader application functionality.
- TraderClientMemEAR: This ear file runs the TraderClientMem application that has a memory leak in the logic for testing a memory leak situation.
- TraderClientLckEAR: This ear file runs the TraderClientLck application that has a lock problem injected for testing dead lock situation.

Java desktop application

The desktop application is also developed from the Web services client wizard from WebSphere Studio Application Development. It is made of the following Java classes:

- TraderClientLogin.java: JDialog extension that provides the initial parameter (Figure A-6)

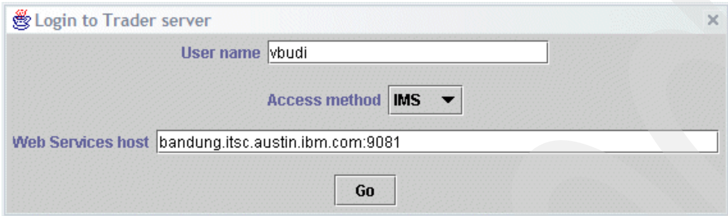


Figure A-6 Login dialog

- TraderClientMain.java: main JApplet that provides the list of companies (Figure A-7)

Figure A-7 Company listing

- TraderClientQuote.java: company quote window that allows invocation of buying or selling stock (Figure A-8)

Figure A-8 Quote window

Back-end implementation

The back-end systems consist of two entities: the company and the customer. The company has the quotes definitions, and the customer database has the customer's name and its stock ownership. Figure A-9 shows the conceptual data structure.

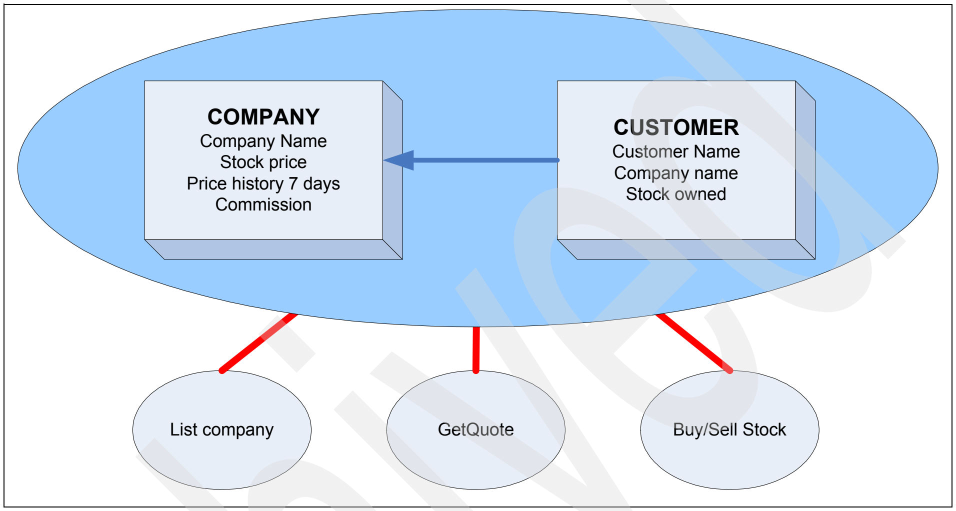


Figure A-9 Entity diagram

The back-end system is implemented in the following platforms:

- IMS: implemented in a single IMS transaction called TRADERBL
- CICS: implemented in a single CICS transaction called TRADERBL
- DB2: implemented as two tables, the customer and company tables

IMS implementation

A single transaction called TRADERBL represents the IMS implementation. This transaction, which is written in COBOL, reads the message that contains the appropriate command and arguments. The TRADERBL transaction accesses two databases: company and customer.

CICS implementation

The CICS implementation consists of two programs:

- TRADERPL: the presentation logic for a 3270 interface that can be invoked using the TRAD transaction.
- TRADERBL: the business logic for the trader application that will be invoked from either the TRAD transaction or from a distributed CSMI transaction. This transaction uses two databases: CUSTFILE and COMPFILE.

DB2 implementation

The DB2 implementation is represented in two tables, the CUSTOMER table and the COMPANY table.

Back-end J2EE servers

The back-end J2EE servers run on a WebSphere server for IMS, DB2, and CICS access and run on an Apache Tomcat server for DB2 access.

WebSphere-based Web services

The WebSphere-based Web services server has the following modules deployed into it:

- TraderDBServices.ear: This program accesses DB2 data. It consists of the following modules:
  - TraderDBWeb: direct front end for the trader DB. This is useful for validating that the trader DB application is running.
  - TraderDBServices: Web module that provides Web services provider implementation.
  - Trader_DB: contains the database access module.
- TraderIMSServices.ear: This program accesses an IMS server using the IMS Connect for Java J2C connection. It consists of the following modules:
  - TraderIMSWeb: direct front end for the trader IMS. This is useful for validating that the trader IMS application is running.
  - TraderIMSServices: Web module that provides Web services provider implementation.
  - TraderIMSEJB and TraderIMSEJB2: EJB implementations for accessing the J2C interface.
- TraderCICSServices.ear: This program accesses the CICS server using the CICS Transaction Gateway J2C connection. It consists of the following modules:
  - TraderCICSWeb: direct front end for the trader CICS. This is useful for validating that the trader CICS application is running.
  - TraderCICSServices: Web module that provides Web services provider implementation.
  - TraderCICSEJB and TraderCICSEJB2: EJB implementations for accessing the J2C interface.

Apache Tomcat server

The Tomcat server runs the TraderTomcat Web application that acts as the Web services provider. The Web services are implemented as JDBC calls to a DB2 database. We used a different DB2 database from the TraderDB application to show a different set of companies.

WebSphere Enterprise Service Bus mediation

The WebSphere Enterprise Service Bus mediation that we use is simple mediation logic that queries the WebSphere Services Registry and Repository (WSRR) server for the location of the service and invokes them. This allows the injection of logic to check which Web services servers are available and redirect the call if necessary.

The mediation is deployed in four modules:

- TraderDBMediation
- TraderCICSMediation
- TraderIMSMediation
- TraderTomcatMediation

The WebSphere Enterprise Service Bus mediation implementation for the trader application is shown in Figure A-10.

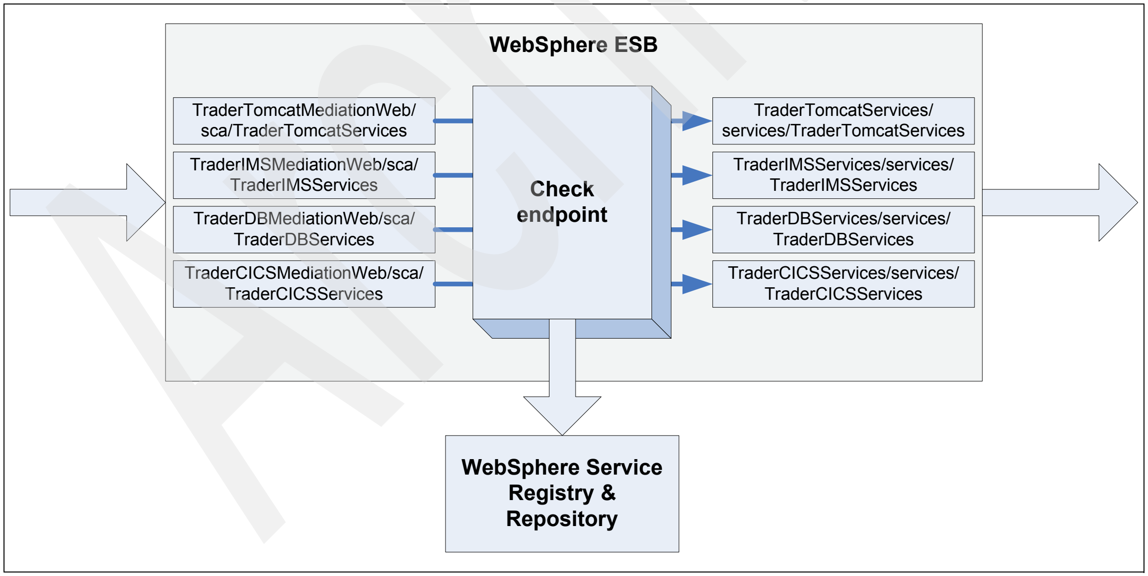


Figure A-10 Trader mediation with WebSphere ESB

All the mediation has similar logic, as shown in Figure A-11.

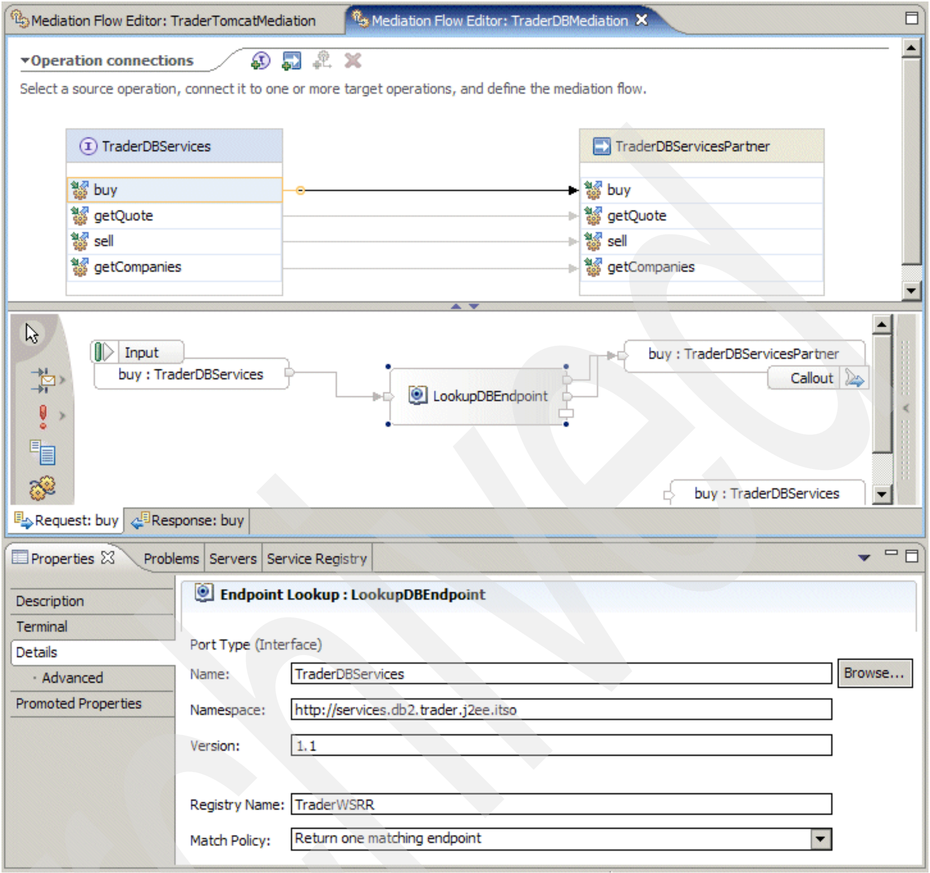


Figure A-11 Mediation logic

Figure A-11 only shows the request logic. It uses the endpoint lookup mediation to query based on the port, namespace, and version. We add another field, which is called *active*, that we can turn on or off based on which Web service provider that we want to activate.

WebSphere Message Broker mediation

The message broker implementation is currently using a hardcoded target. Further enhancement is possible to connect WebSphere Message Broker to WebSphere Services Registry and Repository to query for Web services endpoint. We do not do that in this project.

The broker logic uses four message flow definition stored in a broker archive (bar) file. The bar file is then deployed to the message broker execution group for processing the Web services calls. Figure A-12 shows the conceptual structure of the broker.

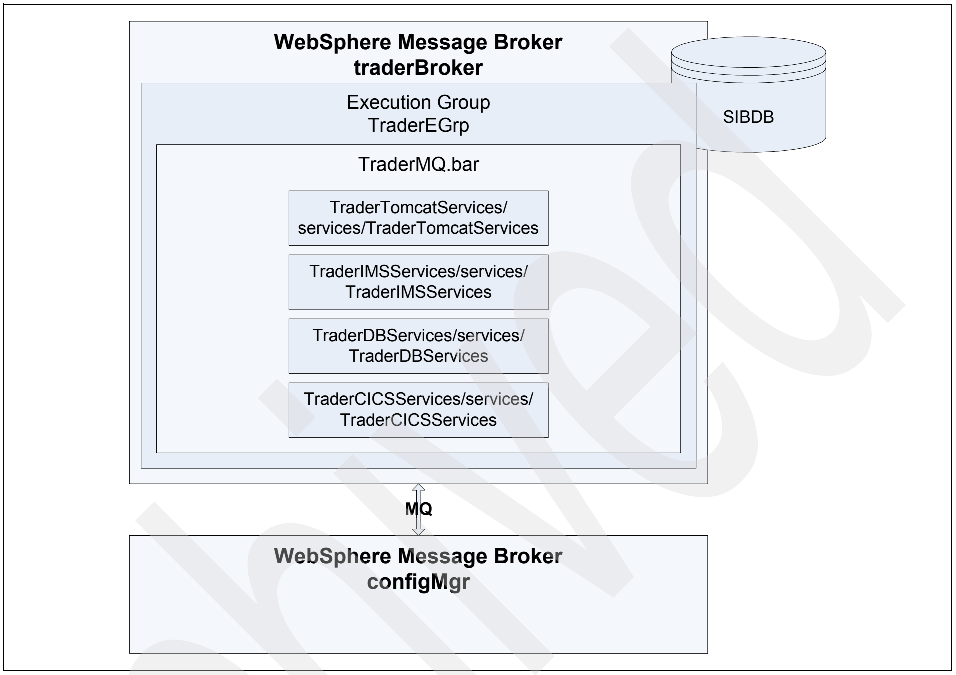


Figure A-12 Broker structure

The message flow logic uses the following:

- An HTTP input node that gets the Web services requests.
- The input node forwards the request to the HTTP request node that performs the actual Web services call. The endpoint address is hardcoded here.
- The HTTP reply node that sends the result back to the requester.

Figure A-13 shows the message flow implementation.

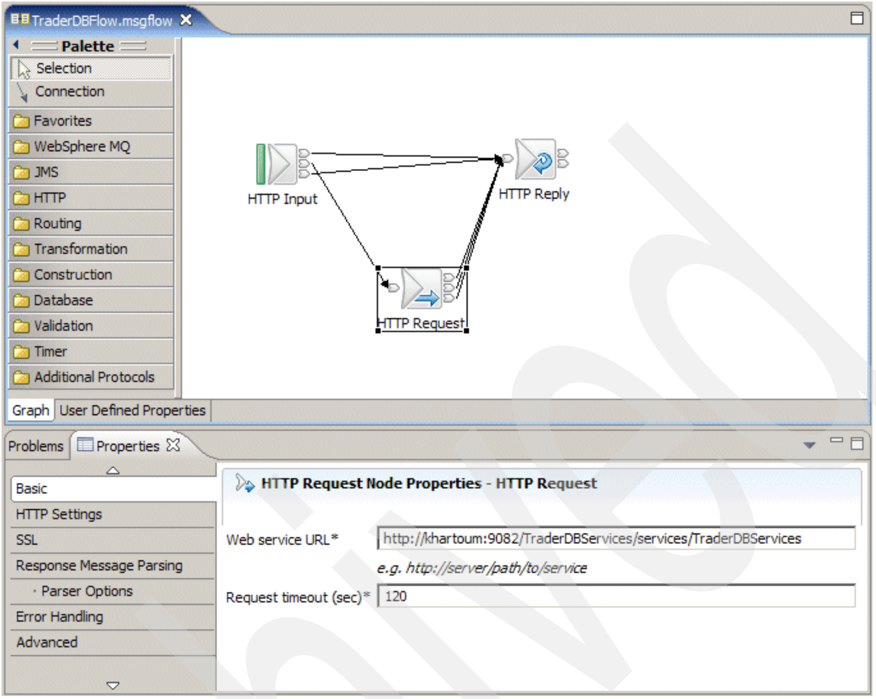


Figure A-13 Message flow implementation

Software requirements

The software required for running the trader application is discussed here. We divide the discussion into:

- Runtime environment
- Development environment

Runtime environment

We use the following software for running the trader components. Other versions of the software we run may be acceptable.

Figure A-14 shows the detailed configuration of our trader application.

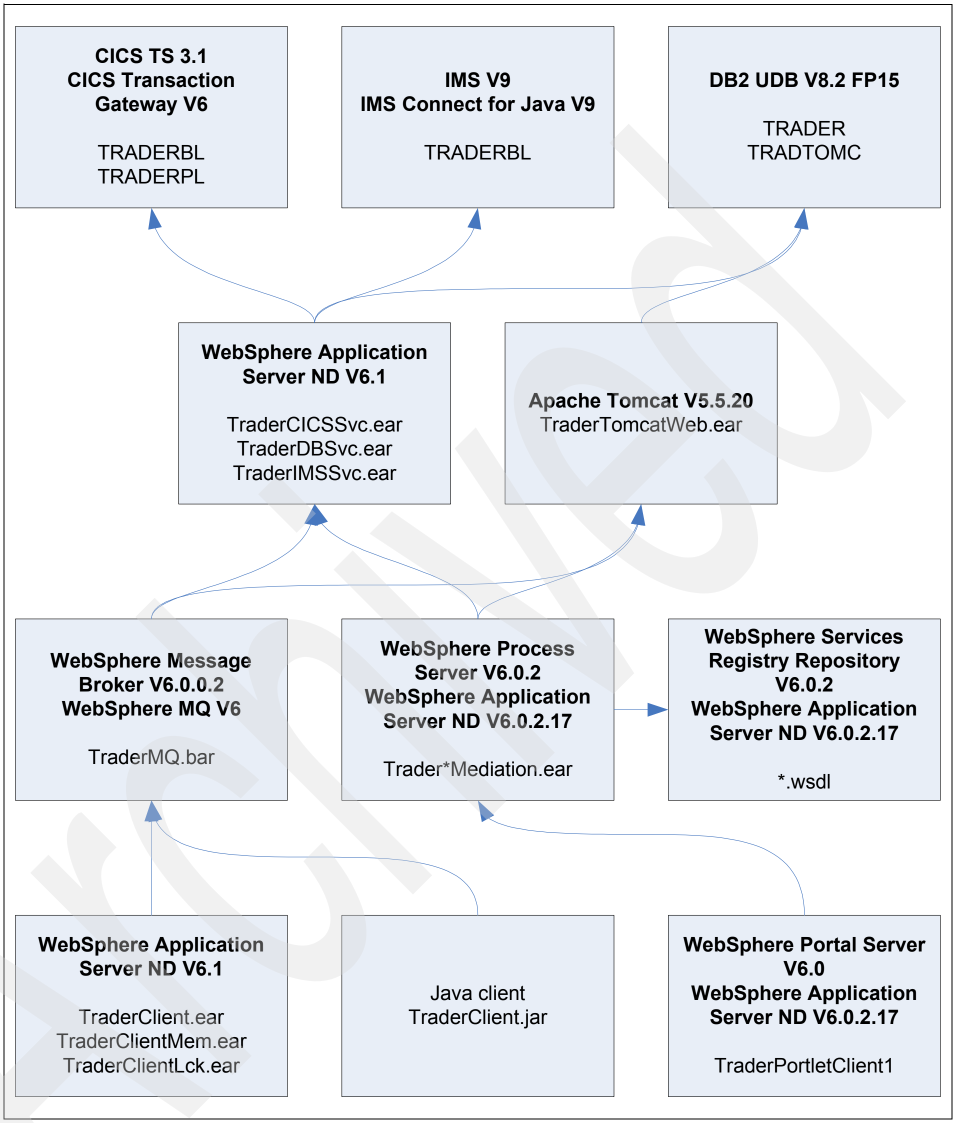


Figure A-14 Trader application detail

The detailed software level of each component is:

- Trader Java client (TraderClient.jar)
- TraderPortletClient1.war
- TraderClient.ear, TraderClientMem.ear, TraderClientLck.ear
- TraderCICSMediation.ear, TraderIMSMediation.ear, TraderDBMediation.ear, TraderTomcatMediation.ear
- TraderMQ.bar
- TraderDBSvc.ear, TraderCICSSvc.ear, TraderIMSSvc.ear
- TraderTomcatServices.war
- DB2 databases
- Trader IMS
- Trader CICS

Development environment

We use various development tools for developing the trader application. You may need to use these tools to customize or change the trader information to suit your needs. The development software that we use is:

- Rational Application Development V7: This is a general-purpose programming environment that is based on Eclipse. We use this tool for developing the J2EE applications, the portal application, and the Java client.
- WebSphere Integration Developer V6.0.2: This is the tool for defining mediation and process modelling for WebSphere Enterprise Service Bus and WebSphere Process Server. This is based on Eclipse also. We use WebSphere Integration Developer for developing the mediation modules to be deployed to the WebSphere Enterprise Service Bus.
- WebSphere Message Broker Toolkit V6.0.2: This is the tool for administering and developing the application for the WebSphere Message Broker. This tool is based on Eclipse, too. If you install it on the same machine as WebSphere Integration Developer, it uses the same Eclipse environment. We specifically uses a different workspace directory for WebSphere Integration Developer and the WebSphere Message Broker Toolkit to prevent any side effects the different products may cause.
- COBOL compiler for IMS and CICS programs in z/OS. Also, various IMS and CICS compilers to translate the necessary definition in IMS and CICS.

Installation procedure

This section provides a detailed installation guideline for installing the trader application in our environment. The installation is discussed in the following sections:

- “IMS installation” on page 636
- “CICS installation” on page 637

IMS installation

The trader IMS environment is supplied in the file TRADIMS.bin. This file should be uploaded to the z/OS sequential data set with the attribute of fixed block 80 records (FB 80). This data set is used to create the IMS.TRADER partitioned data set. Follow this procedure:

1. Upload the data set to z/OS. You must use the binary option.
2. The TRADIMS partitioned data set must be created using the command RECEIVE INDSN(TRADIMS) and specifying the option dsn(IMS.TRADER). This creates a data set called userid.IMS.TRADER.
3. Compile the TRADERBL program. The COBOL program is supplied as a member in IMS.TRADER(TRADERBL). Use the appropriate COBOL compile and link edit JCL to compile it.
4. Create the physical databases for trader IMS. Sample JCL for creating these databases is supplied in IMS.TRADER(DBDEF).
5. Create a database description (DBD) of the trader IMS databases, the CUSTFILE and COMPFILE databases, and their indexes. The sample DBD definition is supplied as member IMS.TRADER(TRADDBD).
6. Create the program specification block (PSB) definition of the TRADERBL that access its databases. The sample PSB definition is supplied as member IMS.TRADER(TRADPSB).
7. Define an application control block (ACB) definition for IMS from the generated DBD and PSB in the previous steps.
8. Define the dynamic allocation records for trader IMS databases using the sample JCL in IMS.TRADER(DFSMDA).
9. Define the appropriate resources in the IMS generation deck. IMS generation should be updated similar to the member IMS.TRADER(IMSSTG1). This would define:
   - Program resource TRADERBL
   - Transaction TRADERBL
   - Databases: COMPFILE, COMPFILX, CUSTFILE, CUSTFILX

– Interregion communication (This is not required for Trader, but ITCAM for IMS Transactions requires this definition.)

10. Run IMS system definition generation:

   a. Stage 1 compilation that generates the JCL for stage 2.
   b. Run stage 2 JCL.
   c. Perform security modification.
   d. Activate the online libraries.

11. Start IMS to verify that IMS can be started properly.

CICS installation

The CICS side of the installation is performed from the TRADCICS.bin file. Similar to the IMS installation, it can be exploded into the CICS.TRADER partitioned data set.

1. Upload the data set to z/OS. You must use the binary option.
2. The TRADCICS partitioned data set must be created using the command RECEIVE INDSN(TRADCICS) and specifying the option dsn(CICS.TRADER). This creates a data set called userid.CICS.TRADER.
3. Compile the TRADERBL and TRADERPL programs. The COBOL program is supplied as a member in CICS.TRADER. Use the appropriate COBOL compile JCL with link edit it with an CICS library.
4. Generate the TRADER map definition to run the 3270 TRAD transaction.
5. Define CICS VSAM files that would be used as trader CICS databases.
6. Modify the CICS resource definition using the DFHCSDUP program. The supplied sample member in CICS.TRADER(CSDUP) modifies the following resources:
   – Programs: TRADERPL, TRADERBL
   – Files: CUSTFILE, COMPFILE
   – Transaction: TRAD
   – Group: TRADER
7. Test the application.

DB2 database creation

The DB2 databases are created from the trader.zip file in the database machine:

1. Extract the file to a temporary directory.
2. Create an empty TRADER database:

```
db2 create database TRADER
```

3. Run the DB2 command:

```
db2move TRADER import
```

WebSphere server installation

The server part of the Trader application is installed within the WebSphere Application Server instance. Use the WebSphere Administration Console to prepare the application server as follows:

1. Define a new Java Connector for IMS connector and the CICS Transaction Gateway.
2. Create a new Java Connection Factory for each IMS and CICS connector with the appropriate parameters for your installation.
3. Define a new JDBC provider for DB2 and create a data source for TRADER database. You must also create an authentication entry for the database.
4. Install the enterprise application (ear) files:
   - TraderIMSServices.ear
   - TraderCICSServices.ear
   - TraderDBServices.ear

WebSphere client installation

The client part is installed within a WebSphere Application Server instance. Install the TraderClientWeb.ear file using the WebSphere Administration Console.

Java client installation

The Java client is provided using the TraderJavaClient.jar. It can be run using the command:

```
java -jar TraderJavaClient.jar
```

WebSphere Service Registry and Repository installation

To customize the WebSphere Services Registry and Repository:

1. Create a WebSphere instance.
2. Install the WebSphere Service Registry and Repository.
3. Install the appropriate DB2 level on the machine.
4. Run the script install.sh from /opt/IBM/WebSphereServices/install and supply all the necessary option.

5. Load the wsdl files for the trader application using the WebSphere Services Registry and Repository Web application.

WebSphere Process Server installation

A WebSphere Process Server or WebSphere Enterprise Service Bus instance must be installed and have a profile created. Install the mediation module (TraderDBMediationApp.ear) using the WebSphere Administration Console.

Update the WebSphere Services Registry and Repository entry from the administration console menu **Service Integration** → **WSRR definition**.

B

Additional material

This book refers to additional material that can be downloaded from the Internet as described here.

Locating the Web material

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

`ftp://www.redbooks.ibm.com/redbooks/SG247151`

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

ibm.com/redbooks

Select **Additional materials** and open the directory that corresponds with the book form number, SG247151.

Using the Web material

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

File name	*Description*
TradersWksp.zip	Workspace of the original application
TradersARMWksp.zip	Workspace of the instrumented application
TradersEARs.zip	Zipped application archives and Java archives

System requirements for downloading the Web material

Figure B-1 shows the system requirements.

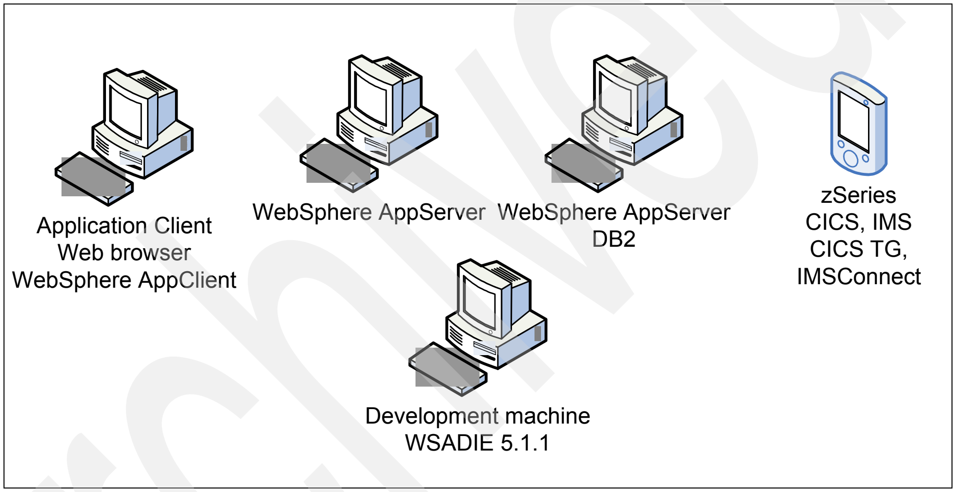


Figure B-1 Application environment

We recommend the following system configuration for the development environment:

Hard disk space	5 GB free space
Operating system	As required by WebSphere Studio Application Development Integrated Edition V5.1.1
Processor	Pentium® IV
Memory	1 GB of RAM
Software	WebSphere Studio Application Development Integrated Edition V5.1.1

The configuration is meant to be run on two WebSphere Application Server environments with access to the back-end DB2, CICS, and IMS applications. We recommend the following system configuration for running the WebSphere Application Server application environment:

Hard disk space	2 GB free space
Operating system	As required by WebSphere Application Server
Processor	As required by WebSphere Application Server
Memory	1 GB of RAM
Software	WebSphere Application Server Version 5.1 or later DB2 Enterprise Server Edition V8.2

The back-end system runs z/OS Version 1.5 or later with the following software running:

- IMS Transaction Manager V8.1
- CICS Transaction Server V2.3
- IMS Connect V2.3
- CICS Transaction Gateway V5.1

We recommend the following system configuration for the front-end Java client:

Hard disk space	1 GB free space
Operating system	Microsoft Windows 2000 SP4, Windows XP SP1, or Windows 2003
Processor	Pentium class
Memory	512 MB of RAM
Software	WebSphere Application Server Application Client Version 5.1 or later

We recommend he following system configuration for the Web browser system:

Operating system	Windows 2000 SP4, Windows XP SP1, or Windows 2003
Processor	Pentium class
Memory	512 MB of RAM
Software	Microsoft Internet Explorer V6 or Mozilla Firefox V1

How to use the Web material

To work with the development environment, follow this procedure:

1. Install the WebSphere Studio Application Development Integrated Edition V5.1.1 on your workstation.
2. Create a subdirectory and unzip the contents of either TradersWksp.zip or TradersARMWksp.zip into this folder.

3. Open WebSphere Studio Application Development Integrated Edition V5.1.1 and specify the workspace directory of the subdirectory that you created in step 2.
4. Import the projects from the directory by selecting **File** → **Import**, and specify **Import existing project into workspace**. Import the following directories:
   - ECIConnector
   - IMSConnector
   - TraderLib
   - Trader_DB
   - TraderCICSCommand
   - TraderCICSEJB
   - TraderCICSEJB2
   - TraderCICSServices
   - TraderCICSWeb
   - TraderCICSEAR
   - TraderDBClientWeb
   - TraderDBEAR
   - TraderDBServices
   - TraderDBWeb
   - TraderDBClientEAR
   - TraderIMSCommand
   - TraderIMSEJB
   - TraderIMSEJB2
   - TraderIMSServices
   - TraderIMSWeb
   - TraderIMSEAR
   - TraderClientWeb
   - TraderClientEAR
   - TraderJavaClient
5. Rebuild all projects using **Project** → **Rebuild All**. You will see several warning messages.
6. Export the projects into a staging directory using **File** → **Export**, and select **EAR file** to export the following projects:
   - TraderWebClient
   - TraderIMSServices
   - TraderCICSServices
   - TraderDBServices
   - TraderIMSWeb
   - TraderCICSWeb
   - TraderDB

7. WebExport the Java application into a staging directory using **File** → **Export** and select **Java archive file** to export the TraderJavaClient project. Make sure to include itso.j2ee.trader.client.TraderClientMain as the main class.

To work with the sample application environment:

1. Use the staging directory, either from the files in the development environment or from the TradersEAR.zip file.
2. From the second WebSphere server:
   a. Install the IMS connector: imsico.rar.
   b. Install the CICS connector.
   c. Install DB2 database JDBC connectivity.
   d. Install TraderIMSSvc.ear or TraderIMSArmSvc.ear.
   e. Install TraderCICSSvc.ear or TraderCICSArmSvc.ear.
   f. Install TraderDBSvc.ear or TraderDBArmSvc.ear.
   g. Restart the application server.
3. From the first WebSphere server, perform the following tasks:
   a. Install TraderClient.ear or TraderClientArm.ear.
   b. Restart the application server.
4. At the back-end z/OS system, perform the following tasks:
   a. Set up the TraderIMS application.
   b. Set up IMS Connect.
   c. Set up the TraderCICS application.
   d. Set up the CICS Transaction Gateway.
5. For the Java application front-end setup:
   a. Install the WebSphere Application Server Application Client.
   b. Run TraderJavaClient.jar.

Abbreviations and acronyms

APF	Authorized Program Facility
API	Application Programming Interface
ARM	Application Response Measurement
ASM	Application Service Monitor
BCM	Byte Code Modification
CAT	Client Application Tracker
CD-ROM	Compact Disk - Read-Only Memory
CICS	Customer Information Control System
CLI	Command Line Interface
CPU	Central Processing Unit
DASD	Direct Access Storage Device
EJB	Enterprise JavaBeans™
ETE	End-to-End
FFDC	First Failure Data Capture
FMID	Function Modification Identifier
GLUE	Global User Exit
GUI	Graphical User Interface
HFS	Hierarchical File System
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP Secure
IBM	International Business Machine Corp.
IMS	Information Management System
IPL	Initial Program Load
ISM	Internet Service Monitoring
ITCAM	IBM Tivoli Composite Application Monitor
ITIL	Information Technology Infrastructure Library
ITSO	International Technical Support Organization
J2C	Java 2 Connector
J2EE	Java 2, Enterprise Edition
JAR	Java Archive
JAX-RPC	Java API for XML-based Remote Procedure Call
JCL	Job Control Language
JDBC	Java Database Connectivity
JES2	Job Entry Subsystem 2
JMX	Java Management Extension
JNI	Java Native Interface
JRE™	Java Runtime Environment
JVM	Java Virtual Machine
JVMPI	Java Virtual Machine Profiler Interface
JVMTI	Java Virtual Machine Tool Interface
LDAP	Lightweight Directory Access Protocol
MSC	Multiple Systems Coupling
MVS	Multiple Virtual Storage
OTMA	Open Transaction Manager Access
PCB	Program Control Block
PDF	Portable Document Format
PLT	Program List Table
PMI	Performance Management Interface
RACF	Resource Access Control Facility
RMI	Remote Method Invocation

RTE	Runtime Environment
SCA	Service Component Architecture
SMF	System Measurement Facility
SMP/E	System Modification Program/Extended
SNMP	Simple Network Management Protocol
SOA	Service-Oriented Architecture
SQL	Structured Query Language
SSH	Secure Shell
SSL	Secure Sockets Layer
SSM	System Service Monitor
STI	Synthetic Transaction Investigator
SYSPLEX	System Complex
TCP/IP	Transmission Control Protocol/Internet Protocol
TRUE	Transaction User Exit
TSO	Time Sharing Option
UDB	Universal Database
UML	Universal Markup Language
URI	Universal Resource Identifier
URL	Universal Resource Locator
WLM	Workload Manager
WRM	Web Response Monitor
WSDL	Web Services Definition Language
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 book.

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 654. Note that some of the documents referenced here may be available in softcopy only.

- *Deployment Guide Series: IBM Tivoli Composite Application Manager for WebSphere V6.0*, SG24-7252
- *Getting Started with IBM Tivoli Monitoring 6.1 on Distributed Environments*, SG24-7143
- *IBM Tivoli OMEGAMON XE V3.1.0 Deep Dive on z/OS*, SG24-7155
- *Implementing OMEGAMON XE for Messaging V6.0*, SG24-7357
- *Installing WebSphere Studio Application Monitor V3.1*, SG24-6491
- *Large-Scale Implementation of IBM Tivoli Composite Application Manager for WebSphere and Response Time Tracking*, REDP-4162
- *Migrating to Netcool/Precision for IP Networks --Best Practices for Migrating from IBM Tivoli NetView*, SG24-7375
- *Solution Deployment Guide for IBM Tivoli Composite Application Manager for WebSphere*, SG24-7293
- *Unveil Your e-business Transaction Performance with IBM TMTP 5.1*, SG24-6912
- *WebSphere Studio Application Monitor V3.2 Advanced Usage Guide*, SG24-6764

Other publications

These publications are also relevant as further information sources:

- IBM Tivoli Composite Application Manager for Response Time Tracking publications
  - *IBM Tivoli Composite Application Manager for Response Time Tracking Administrator's Guide*, SC32-1905
  - *IBM Tivoli Composite Application Manager for Response Time Tracking V6.1 Administrator's Guide*, SC32-9483
  - *IBM Tivoli Composite Application Manager for Response Time Tracking V6.1 Checking Performance and Availability*, SC32-9484
  - *IBM Tivoli Composite Application Manager for Response Time Tracking V6.1 Command Reference Guide*, SC32-9485
  - *IBM Tivoli Composite Application Manager for Response Time Tracking V6.1 Prerequisites*, SC32-9486
  - *IBM Tivoli Composite Application Manager for Response Time Tracking V6.1 Problem Determination Guide*, SC32-9513
  - *IBM Tivoli Composite Application Manager for Response Time Tracking V6.1 Program Directory for z/OS*, GI11-4099
  - *IBM Tivoli Composite Application Manager for Response Time Tracking V6.1: Installing a Management Server in a WebSphere Cluster Environment*, SC32-1804
  - *IBM Tivoli Composite Application Manger for Response Time Tracking V6.1 Installation and Configuration Guide*, GC32-9482
  - *IBM Tivoli Composite Application Manger for Response Time Tracking: Installation and Configuration Guide*, GC32-1907
- IBM Tivoli Composite Application Manager for CICS and IMS transaction publications
  - *IBM Tivoli Composite Application Manager for CICS Transactions V6.1 Product Guide*, SC32-9510
  - *IBM Tivoli Composite Application Manager for IMS Transactions V6.1 Product Guide*, SC32-9511
- 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 Operator's Guide*, SC32-9508

- *IBM Tivoli Composite Application Manager for WebSphere Problem Determination Guide*, SC32-9509
- *IBM Tivoli Composite Application Manager for WebSphere Usage Guide*, GC32-1934
- *IBM Tivoli Composite Application Manager for WebSphere User's Guide*, SC32-9507
- *IBM Tivoli Composite Application Manager for WebSphere: Installing and Configuring the Tivoli Enterprise Monitoring Agent*, SC32-1801
- *IBM Tivoli Composite Application Manager for WebSphere: Tivoli Enterprise Monitoring Agent Problem Determination Guide*, SC32-1800

- IBM Tivoli Composite Application Manager for SOA publications
  - *Configuring IBM Tivoli Composite Application Manager for SOA on z/OS*, SC32-9493
  - *IBM Tivoli Composite Application Manager for SOA Installation and User's Guide*, GC32-9492
  - *IBM Tivoli Composite Application Manager for SOA Program Directory*, GI11-4087
  - *IBM Tivoli Composite Application Manager for SOA Release Notes*, GI11-4096
  - *IBM Tivoli Composite Application Manager for SOA: Installing and Troubleshooting IBM Web Services Navigator*, GC32-9494
- IBM Tivoli Composite Application Manager for Response Time publications:
  - *IBM Tivoli Composite Application Manager for Client Response Time User's Guide Version 6.2*, SC23-6332
  - *IBM Tivoli Composite Application Manager for Web Response Time User's Guide Version 6.2*, SC23-6333
  - *IBM Tivoli Composite Application Manager for Robotic Response Time User's Guide Version 6.2*, SC23-6334
  - *IBM Tivoli Composite Application Manager for End User Response Time Dashboard User's Guide Version 6.2*, SC23-6335
  - *IBM Tivoli Composite Application Manager for Response Time Problem Determination Guide Version 6.2*, GI11-8061
- IBM Tivoli Composite Application Manager for Web Resources publications:
  - *IBM Tivoli Composite Application Manager for Web Resources: J2EE Data Collector Installation Guide*, GC23-6179
  - *IBM Tivoli Composite Application Manager for Web Resources: WebSphere Distributed Data Collector Installation Guide*, GC23-6180

- IBM Tivoli Composite Application Manager for Web Resources: J2EE Agent Installation Guide*, GC23-6181
  - *IBM Tivoli Composite Application Manager for Web Resources: WebSphere Agent Installation Guide*, GC23-6182
  - *IBM Tivoli Composite Application Manager for Web Resources: Web Servers Agent Installation Guide*, GC23-6183
  - *IBM Tivoli Composite Application Manager for Web Resources: Community Edition Data Collector Installation Guide*, GC23-6184
  - *IBM Tivoli Composite Application Manager for Web Resources: Quick Start Guide*, GC23-6185
  - *IBM Tivoli Composite Application Manager for Web Resources: J2EE Agent Problem Determination Guide*, GI11-8160
  - *IBM Tivoli Composite Application Manager for Web Resources: WebSphere Agent Problem Determination Guide*, GI11-8161
  - *IBM Tivoli Composite Application Manager for Web Resources: Web Servers Agent Problem Determination Guide*, GI11-8162
- IBM Tivoli Monitoring publications
  - *Exploring IBM Tivoli Monitoring*, SC32-1803
  - *IBM Tivoli Monitoring Administrator's Guide*, SC32-9408
  - *IBM Tivoli Monitoring: Configuring IBM Tivoli Enterprise Monitoring Server on z/OS*, SC32-9463
  - *IBM Tivoli Monitoring Installation and Setup Guide*, GC32-9407
  - *IBM Tivoli Monitoring Problem Determination Guide*, GC32-9458
  - *IBM Tivoli Monitoring User's Guide*, SC32-9409
  - *IBM Tivoli Monitoring: Upgrading from Tivoli Distributed Monitoring*, GC32-9462
  - *IBM Tivoli Universal Agent API and Command Programming Reference Guide*, SC32-9461
  - *IBM Tivoli Monitoring Universal Agent User's Guide*, SC32-9459
  - *Introducing IBM Tivoli Monitoring*, GI11-4071
- *CandleNet ETEWatch User's Guide*, GC32-9178

Online resources

These Web sites are also relevant as further information sources:

- IBM Tivoli

  http://www.ibm.com/tivoli

- System requirements and prerequisites for IBM Tivoli Composite Application Manager for Response Time Tracking, Version 6.1

  http://publib.boulder.ibm.com/tividd/td/ITCAMRTT/prereq61/en_US/HTML/Version61.html

- Prerequisites for Data Collectors and Monitoring Agents for IBM Tivoli Composite Application Manager for WebSphere 6.1

  http://publib.boulder.ibm.com/tividd/td/ITCAMWAS/prereq61/en_US/HTML/itcam6.html

- IBM Tivoli Composite Application Manager for Response Time Tracking product page

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

- IBM Tivoli Composite Application Manager for WebSphere product page

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

- IBM Tivoli Composite Application Manager for SOA product page

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

- IBM Tivoli Composite Application Manager for J2EE product page

  http://www.ibm.com/software/tivoli/products/composite-application-mgr-itcam-j2ee/

- IBM Tivoli Composite Application Manager for Internet Service Monitoring product page

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

- DB2 UDB Version 8 FixPaks and clients

  http://www.ibm.com/software/data/db2/udb/support/downloadv8.html

- ITCAM for Response Time Tracking Fix Pack 1

  http://www3.software.ibm.com/ibmdl/pub/software/tivoli_support/patches/

- IBM Tivoli Monitoring Workspace Package for IBM Tivoli Composite Application Manager for Internet Service Monitoring

  http://catalog.lotus.com/topal?NavCode=1TW10TM31

- Technote FAQ "Is there a relatively easy way to estimate the amount of disk space needed for your database?"

  http://www.ibm.com/support/docview.wss?rs=2344&context=SS3PGL&dc=DB520&q1=database&uid=swg21250907&loc=en_US&cs=utf-8&lang=en

- Open Group Web site for Application Response Management (ARM)

  http://www.opengroup.org/arm

- Microsoft link for InstallShield error

  http://support.microsoft.com/default.aspx?scid=kb;en-us;295278

- Microsoft Windows Services for UNIX

  http://www.microsoft.com/technet/interopmigration/unix/sfu/default.mspx

- Java specification for JAX-RPC: JSR-000109 Implementing Enterprise Web Services

  http://www.jcp.org/aboutJava/communityprocess/final/jsr109/

- BEA WebLogic interceptor information

  http://e-docs.bea.com/wls/docs81/webserv/interceptors.html

- Eclipse Web site

  http://www.eclipse.org

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

Z

IBM Tivoli Composite Application Manager Family: Installation, Configuration, and Basic Usage

IBM Tivoli Composite Application Manager Family

Installation, Configuration, and Basic Usage

Managing response time and transactions

Monitoring J2EE application servers

Collecting Web services information

This IBM Redbooks publication describes the IBM Tivoli Composite Application Manager family of products. The Composite Application Manager family currently consists of:

- IBM Tivoli Composite Application Manager for WebSphere
- IBM Tivoli Composite Application Manager for Web Resources
- IBM Tivoli Composite Application Manager for Response Time Tracking
- IBM Tivoli Composite Application Manager for Response Time
- IBM Tivoli Composite Application Manager for SOA
- IBM Tivoli Composite Application Manager for Internet Service Monitoring
- IBM Tivoli Composite Application Manager for J2EE
- IBM Tivoli Composite Application Manager for CICS Transactions
- IBM Tivoli Composite Application Manager for IMS Transactions

The aim of IBM Tivoli Composite Application Manager (ITCAM) is to simplify and enhance distributed application management. Application components can reside on multiple servers, across different platforms and J2EE environments, even through mainframes. The complexity of understanding and solving application-related problems, typically around performance issues, requires a cohesive set of tools to provide an end-to-end view of the application.

We provide usage scenarios in this book as a demonstration of using these products in our sample environment. We use a single application environment that can be managed by all of the products.

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

SG24-7151-02 ISBN 0738486590