WebSphere® software

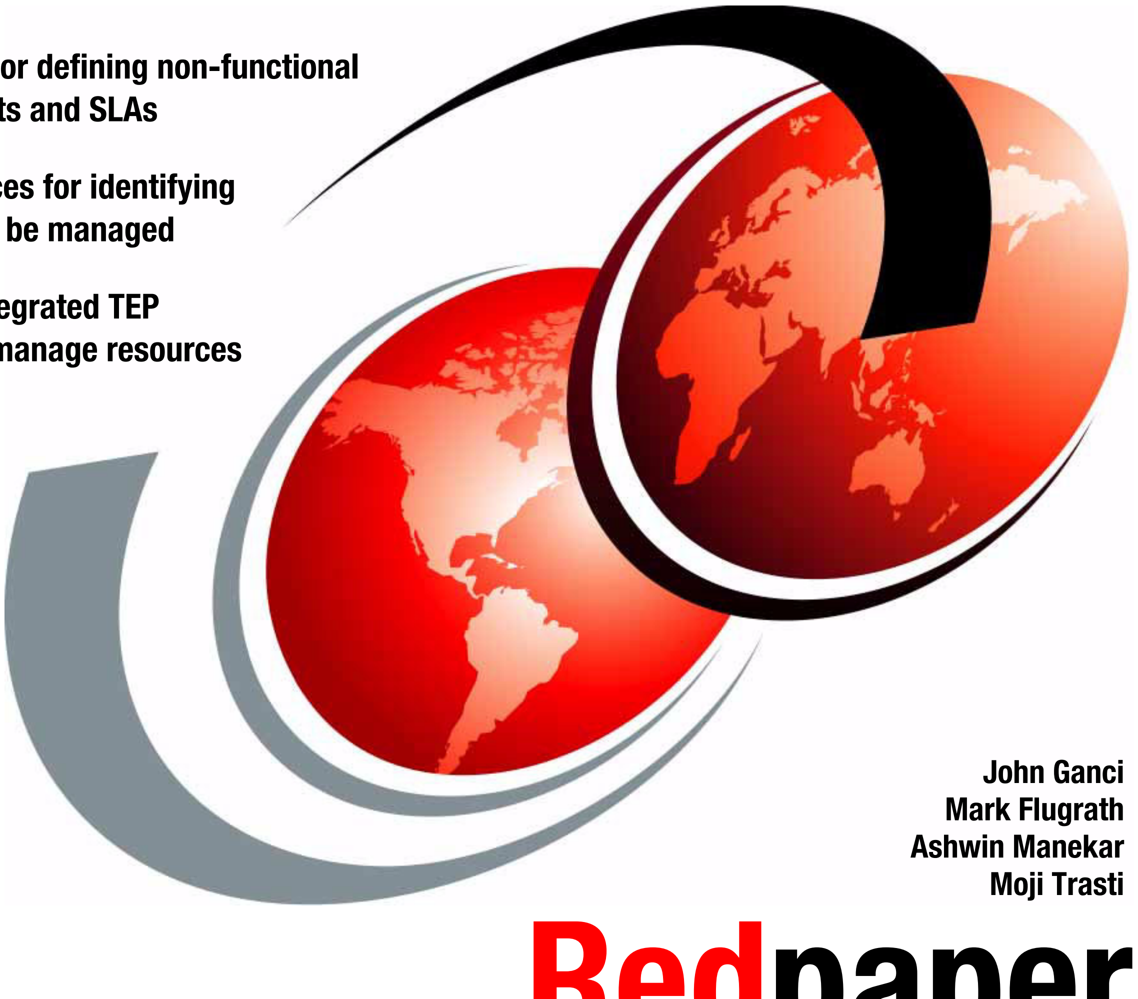
IBM

Best Practices for SOA Management

Guidelines for defining non-functional requirements and SLAs

Best practices for identifying resouces to be managed

Build an integrated TEP console to manage resources



John Ganci
Mark Flugrath
Ashwin Manekar
Moji Trasti

Redpaper

ibm.com/redbooks

International Technical Support Organization

Best Practices for SOA Management

February 2007

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

First Edition (February 2007)

This edition applies to IBM Tivoli Monitoring V6.1, ITCAM for WebSphere V6.0, ITCAM for RTT V6.1, ITCAM for SOA V6.0, ITCAM for CICS Transactions V6.0, IBM Tivoli OMEGAMON XE for CICS V3.10, IBM Tivoli Service Level Advisor V2.1.1.

Contents

Notices

Preface

SOA management provides best practices and software for managing and monitoring SOA composite applications and supporting infrastructure.

This IBM® Redpaper focuses on the following aspects of SOA management:

- Introduction to the key concepts of SOA management.
- Best practices for defining non-functional requirements and SLAs.
- Guidance on how to identify what resources should be managed at the time of the solution analysis and design, and by discovery in the runtime.
- How to build an integrated Tivoli® Monitoring Server - Enterprise Portal (TEP) console to manage and monitor resources for SOA composite applications using ITCAM products.
- How to verify the monitoring functionality by performing an application walkthrough and triggering the monitoring conditions.
- How to create an SLA and report in Tivoli Service Level Advisor.

The team that wrote this Redpaper

This Redpaper was produced by a team of specialists from around the world working at the IBM Software Group - Scenario Analysis Lab, and at the International Technical Support Organization, Raleigh Center.

John Ganci is a Solution Architect for the IBM Software Group, Scenario Analysis Lab (SAL) in Raleigh NC, USA. His areas of expertise include SOA, middleware integration, WebSphere® Application Server, WebSphere Portal, e-commerce, pervasive computing, technical team leadership, Linux® and J2EE™ programming.

Mark Flugrath is a Software Engineer for the IBM Software Group, Scenario Analysis Lab (SAL) in Raleigh NC, USA.

Ashwin Manekar is a Software Engineer for the IBM SWG Scenario Analysis Lab at the RTP Lab in Durham NC, USA. Ashwin holds a MS degree in Computer Science from the University of North Carolina, Charlotte.

Moji Trasti is a Software Engineer for the IBM Software Group, Rational® SVT Solutions Test Team in Raleigh NC, USA. She holds degrees in Chemistry and Computer Science from NCSU. She has over 10 years experience in hardware and software testing. Her areas of expertise include WebSphere Application Server, DB2®, Rational Application Developer and Tivoli monitoring.

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

- Rosalind Radcliffe, IBM Software Group, Tivoli STSM, SOA Management Architect, IBM Raleigh, USA
- Nduwuisi Emuchay, IBM Software Group, Tivoli STSM, Architecture and Technology, IBM Austin, USA
- Phil Fritz, IBM Software Group, Tivoli Program Manager, SOA Management, IBM Austin, USA
- Roger Meli, IBM Software Group, Architect for ITCAM for WebSphere, IBM Raleigh, USA
- Kaylee Thomsen, IBM Software Group, Scenario Analysis Lab Architect, IBM Raleigh, USA
- Chip Patterson, IBM Software Group, Scenario Analysis Lab Manager, IBM Raleigh, USA
- Martin Keen, ITSO Raleigh Center
- Tony Bhe, IBM Tivoli, USA
- Miguel Juarez, IBM Tivoli, USA
- Martha Yanez, IBM Tivoli, USA
- Donavan Stevens, IBM Tivoli, USA
- Jean-Jacques Heler, IBM Tivoli, USA
- Gurdeep Rahi, IBM UK

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll 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'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

  `ibm.com/redbooks`

- Send your comments in an email to:

  `redbooks@us.ibm.com`

- Mail your comments to:

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

1

Introduction to SOA management

This Redpaper starts with the premise that you have decided to adopt a service-oriented architecture and want to learn about best practices for managing SOA composite applications and supporting infrastructure.

In this chapter we first provide a brief overview of the IBM SOA Foundation scenarios to give a context for SOA management. The remaining sections of this chapter provide an introduction to the challenges, architecture, IBM products, and lifecycle used for SOA Management.

We have organized the topics into the following sections:

- SOA Foundation scenarios
- SOA management challenges
- Managing the SOA Foundation architectural layers
- IBM software for SOA management
- SOA management lifecycle

1.1 SOA Foundation scenarios

The *IBM SOA Foundation* is an integrated, open standards-based set of IBM software, best practices, and patterns designed to provide what you need to get started with SOA from an architecture perspective. A key element of the SOA Foundation is the SOA Foundation scenarios. *SOA Management* is one of the defined SOA Foundation scenarios.

1.1.1 SOA Foundation scenarios

The SOA Foundation scenarios (or simply SOA scenarios) are representative of common scenarios of use of IBM products and solutions for SOA engagements. The SOA scenarios communicate the business value, architecture, and IBM open standards-based software used to satisfy the SOA scenarios. The SOA scenarios can be used as reference material to accelerate a SOA implementation based on your customer requirement.

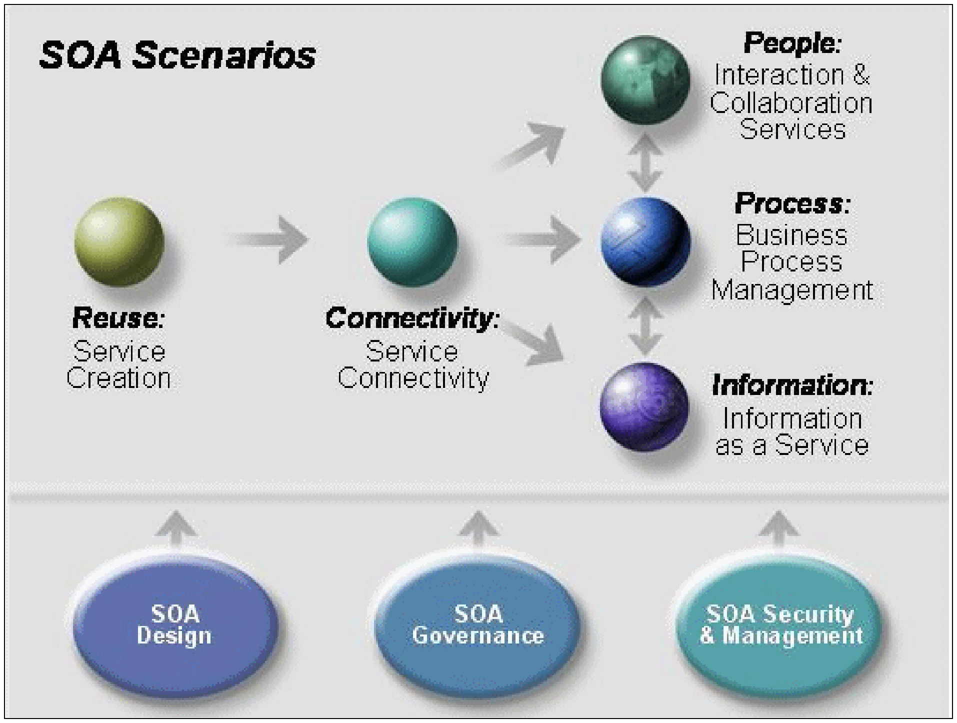

SOA Scenarios
People: Interaction & Collaboration Services
Process: Business Process Management
Information: Information as a Service
Reuse: Service Creation
Connectivity: Service Connectivity
SOA Design
SOA Governance
SOA Security & Management

Figure 1-1 SOA Scenarios and Entry Points

Figure 1-1 displays the SOA scenarios and the relationships between them. For our purposes, we have categorized the scenarios as core and supporting scenarios.

The core SOA scenarios include Service Creation, Service Connectivity, Interaction and Collaboration Services, Business Process Management, and Information as a Service. SOA Design, Governance, and Security and Management are SOA scenarios that can be applied across the core SOA scenarios. The SOA scenarios can be used together and adopted incrementally.

It is worth noting that the core and supporting scenarios are not mutually exclusive: elements and concerns of the supporting scenarios can be found in the core scenarios and vice versa. For example, Service Creation has concerns related to SOA Design, SOA Governance and SOA Security and Management. So do the other core SOA scenarios. It is also useful to note that the supporting scenarios have concerns that relate to one or more of the core scenarios.

For more information refer to the following:

- IBM SOA Foundation and SOA scenarios

  Chapter 2. IBM SOA Foundation in *Patterns: SOA Foundation - Service Creation Scenario*, SG24-7240.

  IBM SOA Foundation: An architectural introduction and overview whitepaper, available at:

  http://www.ibm.com/developerworks/webservices/library/ws-soa-whitepaper/

- SOA Security and Management scenario

  The whitepaper *Secure and manage services in a SOA Environment to achieve business objectives*, GC28-8392-00.

1.1.2 Redpaper scope for SOA management

There are three key points that we would like to highlight regarding the scope of this redpaper for SOA management.

First, within the SOA Foundation scenarios, SOA Security and Management are discussed together as a supporting scenario to the core scenarios. Our focus in this redpaper is on the management of services and supporting infrastructure. This redpaper does not include change and release management, or SOA security.

Second, although many organizations are adopting SOA, the transition is not an overnight process. In our coverage of SOA Management, we discuss how to coordinate the management of SOA service management with traditional operational management.

Third, the example business scenario discussed in this redpaper demonstrates how to apply best practices for SOA management to the Service Creation scenario, Indirect Exposure realization. The Service Creation scenario is used to demonstrate exposing application functionality of an existing application or new business logic as a service. Many of the best practices applied to the example scenario can be applied to the other core SOA scenarios as well.

For more information on the Service Creation scenario, Indirect Exposure realization, refer to the redbook *Patterns: SOA Foundation - Service Creation Scenario*, SG24-7240.

1.2 SOA management challenges

Historically, business applications were built with a monolithic purpose (silos) and were typically inflexible. While this kind of architecture can be effective, it is often very difficult to change and integrate with other applications within the enterprise and between enterprises because custom coded connections are required.

Within the SOA solution domain, business processes increasingly depend on integrated services that form multi-tier, composite applications. Composite applications often span architectural layers, including service consumers, business processes, services, service components, and operational systems.

SOA management includes solutions and software for managing and monitoring SOA composite applications and supporting infrastructure across the architecture layers. In this section, we highlight the key challenges for SOA management:

- Understand the relationship of services
- Manage services as resources
- Ensure nonfunctional requirements are achieved
- Identify the resources to manage
- Monitor the end-to-end view in an integrated console

Understand the relationship of services

It is important to understand the relationship between service consumers and providers for composite applications. SOA brings the benefits of application reuse. Although reuse of existing applications as services has many benefits, one must consider the management implications. The importance of monitoring and managing the availability and performance of the application functionality exposed as a service increases when the service is reused by service consumers that depend on this functionality.

For example, in a siloed approach each application can have its own tax calculation functionality. In a SOA composite application, the tax calculation can be exposed and consumed as a common service. This common service has the benefits of flexibility, reuse, cost savings, and so on, but also has increased dependency and must be monitored and managed accordingly.

Manage services as resources

To achieve the quality of service (QoS) defined by the business, each service endpoint should be managed as a resource. This includes the invocation of services (service consumer) as well as the application functionality exposed as a service (service provider). When speaking in general terms, management of services can refer to a range of services technologies. In the context of our example scenario, services are implemented as Web services.

Managed services should have real-time availability and performance metrics, and a defined service level agreement. Like other resources, services are deployed, configured, versioned, monitored, managed, secured, and audited. When down, the management tooling should provide a means of troubleshooting, and, better still, a method of monitoring and alerting of issues before failure.

Ensure nonfunctional requirements are achieved

SOA management is used to ensure nonfunctional requirements of the IT architecture are aligned with the business objectives. Monitoring and management of the composite application should include specific metrics to ensure service level agreements (SLAs) are meeting the business objectives. The management solution should also consider how to report on adherence to the SLAs.

Identify the resources to manage

A key challenge of SOA management is knowing how to identify what resources to manage. Knowing what should be managed is derived from analyzing the non-functional requirements and SLAs. The identification of resources to be managed is performed at the time of the solution analysis and design, and by discovery in the run-time environment.

Monitor the end-to-end view in an integrated console

In a siloed application architecture, resources are often managed by separate operators or specialists, in separate management consoles.

Composite applications require a mind shift in the management approach. There is a need for monitoring and management tooling that covers the end-to-end view of the composite application, as well as providing detailed information about performance and availability metrics for the individual resources. Also, the *operator* should have an end-to-end monitoring role.

There are several perspectives to consider for the end-to-end view. The end-to-end view includes both horizontal and vertical views. The horizontal view is the view of the transaction, for example, Web browser → Web service client → Web service → CICS®. The vertical view is the view of the service invocation through the architectural abstraction layers including consumer, business process, composite application, (Web) service endpoints, concrete application artifacts that support the service (such as mediation primitives), and operational environment.

1.3 Managing the SOA Foundation architectural layers

The SOA Foundation Reference Architecture Solution View includes architectural layers as shown in Figure 1-2. In this section we explain how the resources for these architectural layers are managed in a SOA environment. In some cases, the monitoring spans the architectural layers.

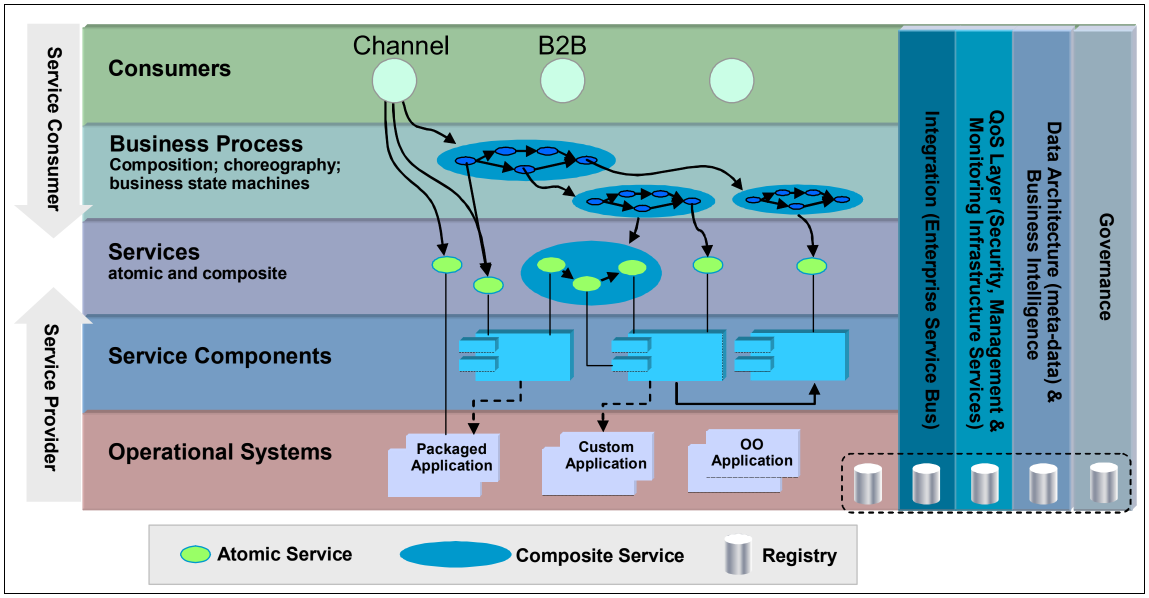


Figure 1-2 SOA Foundation Reference Architecture Solution View

Managing services

Management of services includes service consumers and service providers. Service consumers invoke services, while service providers expose application functions to be consumed. When speaking in general terms, management of services refers to a range of service technologies.

Within the redpaper example scenario, when we refer to services we are specifically referring to Web Services. For example, both a J2EE Web Services client application (service consumer) and a session EJB™ Web service (service provider) can be monitored and managed.

Some key elements of managing the services layer are:

- Understanding how services relate to each other and to the IT infrastructure, as well as how the service relates to the business process layer.
- Controlling the message flow in the service environment through management mediations such as log, filter, and route. The message flow often spans the architectural layers.
- Centralizing services management policy.
- Defining business-related IT goals.

Note: Business processes are managed indirectly in that the services that make up the business processes are managed resources. It is important to understand the relationship of service providers and consumers. This also includes understanding the impact to the business process or business service. Managing business processes is beyond the scope of the example scenario in this paper.

For more information on business process management, refer to the redbook *Patterns: SOA Foundation - Business Process Management Scenario*, SG24-7234.

Managing transaction performance

Managing and monitoring the end-to-end transaction performance is a key measurement for SLAs. Managing the transaction performance is also a very useful analysis and troubleshooting tool. For example, you can be proactive in detecting a slowdown in performance before it becomes a critical problem that stops the transaction from being completed. Also, a view of the transactional performance can be very helpful for troubleshooting to isolate the resources that are not performing or are failing.

Managing transaction performance includes the following:

- Understanding the performance of a service and the decomposition of transactions with specific metrics for individual requests
- Providing the relationship between service requests and the implementation artifacts such as J2EE beans and JDBC™ requests

Managing the supporting middleware

Many of the resources used by services are found in the middleware. For example, WebSphere Application Server application servers are used to host J2EE and Web Services applications. DB2 UDB is used to host databases. IBM WebSphere MQ is used for messaging. Each of these resources should be managed and monitored.

Managing the supporting middleware includes the following concepts:

- Understanding the health of the infrastructure that supports the services
- Correlating problems in the services to infrastructure issues such as a queue filling up or an exhausted thread pool

Managing the operational systems

SOA environments are built using real resources and those resources must also be managed. Managing the operational systems includes understanding the health of the infrastructure that supports the services.

1.4 IBM software for SOA management

This section includes a summary of best practices and IBM software products used for SOA management. The products are categorized as follows:

- IBM Service Management
- Manage services and supporting infrastructure

1.4.1 IBM Service Management

The ongoing management, enhancement, and redeployment of services involves many people, processes, and pieces of information. An organization needs ways to connect them all. IBM Service Management offers a common data model that facilitates real-time information sharing.

Note: More information on IBM Service Management can be found at:

`http://www.ibm.com/itsm/`

Some of the key capabilities provided by IBM Service Management are:

- ITIL®-aligned workflows
- Open and standards-based configuration management database
- Automated, infrastructure-aligned tasks
- Best practices and implementation support

IBM Service Management is made up of the following elements:

- IT Process Management products
- IT Service Management platform
- IT Operational Management products

We have included two examples of process flows and supporting products:

- Configuration management

  Two key components of the IBM Tivoli Change and Configuration Manager Process Manager are as follows:

  The *Change and Configuration Management Database* (CCMDB) facilitates the understanding of the relationships between services, applications, software, and hardware resources.

  The *Tivoli Application Dependency Discovery Manager* (TADDM) provides complete visibility into application complexity by automatically creating and maintaining application infrastructure maps.

- Change and release management

  During the deployment of application components, *IBM Tivoli Configuration Manager* and *IBM Tivoli Provisioning Manager* can leverage deployment descriptors created during the development process and stored in the *IBM Rational ClearCase®* repository.

1.4.2 Manage services and supporting infrastructure

This section provides a summary of the IBM software products used to manage services and supporting infrastructure.

Note: The scope of this redpaper does not include information on WebSphere Business Monitor or Tivoli Business Systems Manager.

IBM Tivoli Monitoring (ITM) products

IBM Tivoli Monitoring (ITM) is the primary product that provides the base infrastructure for management and monitoring. The technology for ITM originated and is shared with OMEGAMON®, which was obtained in the Candle® acquisition. The OMEGAMON product name is still used on the z/OS® platform.

The core components included with IBM Tivoli Monitoring V6.1 are:

- Tivoli Enterprise™ Monitoring Server (TEMS)

  The Tivoli Enterprise Monitoring Server (TEMS) is the core component of an IBM Tivoli Monitoring solution. TEMS is responsible for collecting alerts, performance, and availability data from agents. In addition, TEMS is used to track the heartbeat request interval for all configured TEMS agents. TEMS initiates and tracks all situations and policies, and stores the data in the centralized data warehouse.

When installing TEMS, the primary Monitoring Server is configured as a *Hub*, and all subsequent Monitoring Servers are configured as remote to the Hub. This type of architecture provides for great scalability and centralized collection and analysis of the data.

- Tivoli Enterprise Portal Server (TEPS)

  The Tivoli Enterprise Portal Server (TEPS) provides a presentation layer and database repository for all graphical presentation of monitoring data. The TEPS is used for retrieval, manipulation, analysis, and formatting of data. It manages this access through user workspace views. The TEPS maintains a persistent connection to the TEMS Hub, and can be considered a logical gateway between the TEMS Hub and the TEP client (desktop or browser).

  The TEPS provides the ability to customize workspace views, situations (thresholds), and workflows. A database manager such as DB2 UDB is used to host the Data Warehouse and TEPS configuration database.

- Tivoli Enterprise Portal (TEP) clients

  The Tivoli Enterprise Portal can be accessed using the following clients to view all monitoring data collection within a single window:

  – TEP Desktop client: Java™ application
  – TEP Browser client: Java applet run in a Web browser

  The following products have integrated interfaces into the Tivoli Enterprise Portal to provide consolidated view of composite application data:

  – IBM Tivoli Composite Application Manager for SOA
  – IBM Tivoli Composite Application Manager for WebSphere
  – IBM Tivoli Composite Application Manager for Response Time Tracking
  – IBM OMEGAMON XE product family
  – IBM Tivoli Monitoring product family
  – IBM Tivoli Enterprise Console® (TEC)
  – IBM NetView® for z/OS (release 5.2)

- Tivoli Enterprise Monitoring Agents

  The Tivoli Enterprise Monitoring Agents are installed on the systems or subsystems requiring data collection and monitoring. The agents are responsible for data gathering and distribution of attributes to the monitoring servers, including initiating the heartbeat status.

IBM Tivoli Composite Application Manager products

The IBM Tivoli Composite Application Manager (ITCAM) product family consists of the following products:

- **ITCAM for SOA**: Monitor and manage the SOA services layer.
- **ITCAM for Response Time Tracking**: Proactively recognize, isolate, and resolve transaction performance problems.
- **ITCAM for WebSphere**: Isolate the root cause of bottlenecks in a WebSphere application run-time environment.
- **ITCAM for CICS Transactions**: Capture data from CICS systems to be analyzed in ITCAM for Response Time Tracking and WebSphere.
- **ITCAM for IMS™ Transactions**: Capture data for IMS systems to be analyzed in ITCAM for Response Time Tracking.

IBM Tivoli OMEGAMON XE products

There are many IBM Tivoli OMEGAMON XE products. Two key products used within the SOA space are the following:

- **Tivoli OMEGAMON XE for Messaging**: Monitor MQ family run times and provide automatic corrective actions to improve performance and availability.
- **Tivoli OMEGAMON XE for CICS**: Monitor and manage CICS Transactions and resources so that, when a problem occurs, it can be quickly detected and isolated to minimize or eliminate any impact to your business.

IBM Tivoli Service Level Advisor

IBM Tivoli Service Level Advisor (TSLA) is a Service Level Management solution for providers of IT Services. It simplifies and automates the process of managing service level agreements, enabling IT organizations to proactively manage and report on service levels from across the management infrastructure.

IBM TSLA is a predictive solution for defining, analyzing and reporting on SLAs enterprise wide. Among the key features are the ability to:

- Leverage wizards to rapidly define SLAs.
- Automate SLA evaluation by enabling alerts for violations.
- Provide executive-level reports to effectively communicate SLA performance to executives across the business.
- Avoid violations by analyzing trends.
- Be integrated with IBM Tivoli Monitoring. This integration provides the ability to create TSLA reports based on data (events) mined from the TM data warehouse database. The reports can be a very useful tool to determine if SLAs are being achieved to ensure SOA governance.

Where to find more information

More information on IBM Tivoli Monitoring, ITCAM products and IBM Tivoli Service Level Advisor can be found at:

- IBM Tivoli management products InfoCenters:

  `http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=/com.ibm.tivoli.az.doc/welcome.htm`

- *IBM Tivoli Composite Application Manager V6.0 family*, SG24-7151
- *IBM Tivoli OMEGAMON XE V3.1.0 Deep Dive on z/OS*, SG24-7155

1.5 SOA management lifecycle

The objective of this section is to highlight the key tasks performed in the SOA management lifecycle (see Figure 1-3 on page 13). We describe the key inputs of the task, user roles, and artifact outputs of the tasks. Although the tasks are described in a top-down approach, many of the task are iterative.

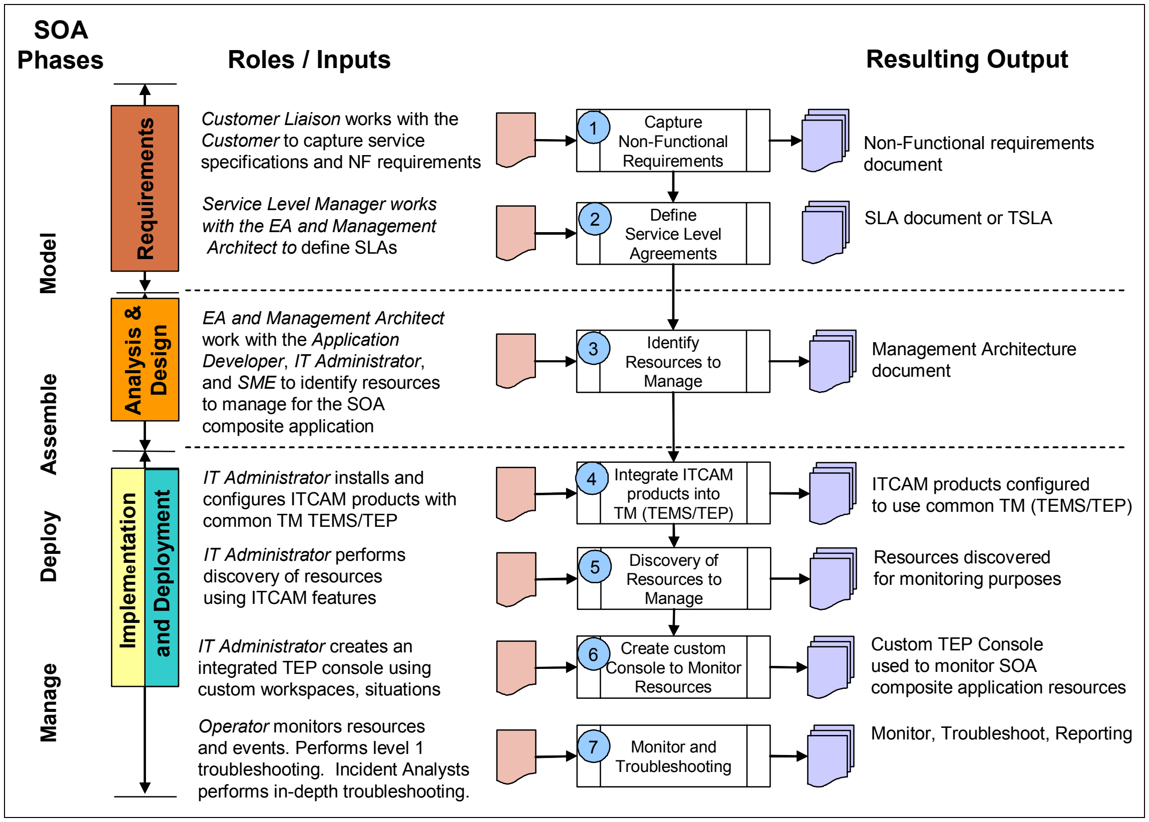

SOA Phases
Roles / Inputs
Resulting Output
Model
Assemble
Deploy
Manage
Requirements
Analysis & Design
Implementation and Deployment
Customer Liaison works with the Customer to capture service specifications and NF requirements
Service Level Manager works with the EA and Management Architect to define SLAs
EA and Management Architect work with the Application Developer, IT Administrator, and SME to identify resources to manage for the SOA composite application
IT Administrator installs and configures ITCAM products with common TM TEMS/TEP
IT Administrator performs discovery of resources using ITCAM features
IT Administrator creates an integrated TEP console using custom workspaces, situations
Operator monitors resources and events. Performs level 1 troubleshooting. Incident Analysts performs in-depth troubleshooting.
1 Capture Non-Functional Requirements
2 Define Service Level Agreements
3 Identify Resources to Manage
4 Integrate ITCAM products into TM (TEMS/TEP)
5 Discovery of Resources to Manage
6 Create custom Console to Monitor Resources
7 Monitor and Troubleshooting
Non-Functional requirements document
SLA document or TSLA
Management Architecture document
ITCAM products configured to use common TM (TEMS/TEP)
Resources discovered for monitoring purposes
Custom TEP Console used to monitor SOA composite application resources
Monitor, Troubleshoot, Reporting

Figure 1-3 SOA management lifecycle

1. Capture non-functional requirements.

   The Customer Liaison works closely with the Customer to capture service specifications and non-functional requirements.

   Note: For more information see Chapter 2, “Define non-functional requirements and SLAs” on page 17.

2. Define service level agreement (SLAs).

   The Service Level Manager works with the Enterprise Architect (EA) and Management Architect to define SLAs. The service specifications and non-functional requirements are used as a basis to define SLAs.

   If the SLA is implemented using a product such as Tivoli Service Level Advisor (TSLA), the IT Administrator is responsible for creating the SLA and configuring the SLA with the specific resources that will be associated with it.

The IT Administrator can configure the environment to restrict access to specific built-in reports as well as create custom reports.

Note: For more information see Chapter 2, “Define non-functional requirements and SLAs” on page 17.

3. Identify resources to manage.

   The Enterprise Architect owns the end-to-end solution architecture, whereas the Management Architect is focused on the operational aspects of the architecture. The EA and Management Architect work with the Application Developer, IT Administrator, and SME to identify resources to manage for the SOA composite application. Key inputs for defining what resources to manage include non-functional requirements and SLAs.

   There are two fundamental approaches for identifying resources to be managed. First, resources to be managed are identified at the time of analysis and design of the solution based on non-functional requirements and SLAs. Second, the application is deployed to either a development test environment or a production environment, and then a discovery of key resources to manage is performed based on analysis of how the application is used. Even in cases when some of the information is available at the time of analysis and design, the use of ITCAM discovery features is still desirable in a development test environment to help automate the creation of situations or listening monitors.

   Note: For more information see Chapter 3, “Identify resources to manage” on page 43.

4. Integrate ITCAM products into a common Tivoli Monitoring Server.

   As a prerequisite to managing resources in an integrated Tivoli Enterprise Portal (TEP), each of the monitoring products needs to be configured to use a common Tivoli Monitoring Server (TMS).

   Note: For more information see Appendix A, “Deploy the environment and sample application” on page 109.

5. Discovery of resources.

   Although the discovery of resources at run time has already been mentioned, we have included this as a specific step since it is dependent on the previous step (integrate ITCAM into common TMS).

Once monitoring components are installed and configured the IT Administrator can discover application-level resources using the features provided by the ITCAM products. This information is provided to the EA and Management Architect to incorporate back into the management architecture, as well as used to create TEP situations for monitoring the resources.

Note: For more information see 4.2, “Discovery of resources” on page 71.

6. Create the custom TEP console to manage and monitor resources.

   The IT Administrator creates an integrated TEP console using custom workspaces and situations to monitor the resources defined across architecture layers using a range of ITCAM products.

   Note: For more information see Chapter 4, “Build an integrated TEP console to manage resources” on page 59.

7. Monitoring, troubleshooting, and reporting

   The Operator monitors resources and events using the custom TEP workspaces and situations, and performs the initial troubleshooting.

   For problems that are not able to be resolved by the Operator within a specified duration of time, an Incident Analyst is assigned for more in-depth problem determination using the customized TEP workspaces and in some cases, native consoles for the ITCAM products (for example, ITCAM for WebSphere and RTT - Managing Server Console).

   In addition, the reporting of events is performed to provide details on the monitoring of resources. This data can be used to evaluate whether SLAs are being fulfilled.

   Note: For more information see Chapter 4, “Build an integrated TEP console to manage resources” on page 59.

2

Define non-functional requirements and SLAs

Capturing the non-functional requirements and defining service level agreements (SLAs) provides key information for knowing what resources should be managed.

In this chapter, we first describe the business scenario used for the redpaper example. Next, we provide best practices for capturing non-functional requirements and SLAs. Finally, we demonstrate how to create an SLA and analyze reports using Tivoli Service Level Advisor.

This section is organized into the following topics:

- Business scenario
- Non-functional requirements and SLAs
- Manage SLAs using Tivoli Service Level Advisor

2.1 Business scenario

The objective of this section is to provide a description of the fictitious ITSO Bank business scenario used as an example throughout this paper. Our description of the business scenario is limited, since our focus is managing the SOA solution. We do not cover the development aspects of creating new services.

2.1.1 Business scenario initial context

The ITSO Bank provides banking services online via a Web browser (see Figure 2-1). The existing application provides support for the following business processes:

- Create account
- Get customer information
- Get account balance
- Transfer funds

The existing application is a J2EE-based Web application accessible to Web browser clients. The existing application consists of JSPs for the user interface, servlets, EJBs, DB2 database, and a backend CICS financial application.

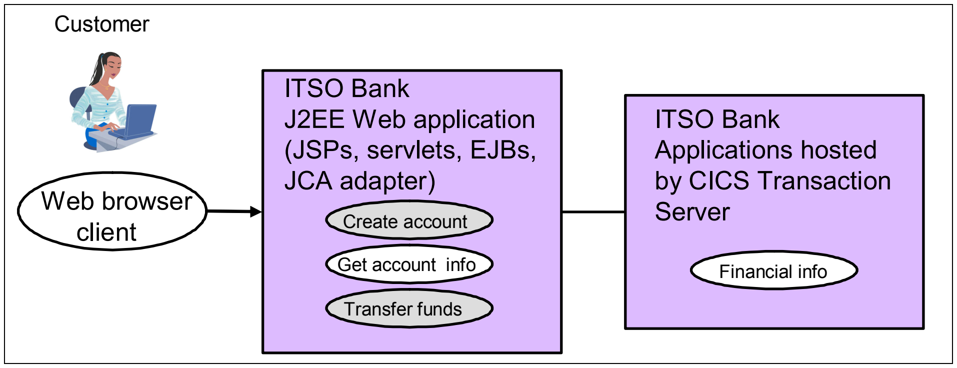


Figure 2-1 ITSO Bank initial context

Note: The ITSO Bank sample application only implements the *get customer information* transaction. We developed the sample application with the ability to retrieve customer account information from a DB2 UDB database or a CICS backend. We made the redpaper environment available to a wider audience by not requiring a CICS environment,.

The existing J2EE application uses a JCA adapter to access the backend financial application hosted by CICS Transaction Server. At a more detailed level, the session EJB uses the CICS ECI resource adapter to communicate with the CICS Transaction Gateway (CTG) to access the CICS Transaction Server. The CICS ECI resource adapter is a JCA adapter for WebSphere Application Server packaged with CTG. Within the IBM SOA Foundation scenarios, this pattern is known as the Service Creation scenario - Indirect Exposure realization.

2.1.2 Business objectives

This section outlines the key business and IT challenges of the ITSO Bank for the SOA adoption.

Business drivers

The key business objectives are as follows:

- **Enable multiple channels:** The ITSO Bank wants to expand its presence in the marketplace and increase market share. The existing online ITSO Bank has acquired a brick-and-mortar bank. This acquisition will require the merger of banking applications used by online customers and the application used by tellers at the bank branches visited by customers.
- **Lower total cost of ownership:** By making application functionality available as a service, applications can more easily consume the common application functionality as a service. This has the effect of lowering development and maintenance costs by driving down redundancy.
- **Leverage existing core assets and skills:** The ITSO Bank wants to maximize the reuse of existing application assets, business processes, and skills of the organization.

IT challenges

The ITSO Bank faces many IT challenges. An assessment of the existing ITSO architecture has revealed the following pain points:

- **Integrate application architecture:** IT services need to be identified from a business point of view. The existing applications are not designed to fully support the exposure and reuse of services via component-based development.

  Integration tools and middleware are not all-encompassing to support SOA integration and component placement. The current application design approach does not consistently apply SOA or component design principles.
- **Management architecture:** The existing management architecture focuses on the management of resources as individual resources. The management

architecture needs to be transformed to manage the end-to-end transactions of the SOA composite application and supporting infrastructure.

2.1.3 Functional requirements

This section highlights the functional requirements for phase one for the ITSO Bank example. The key functional requirements are as follows:

- FR1: Reuse existing application functionality as a Web service.

  Reuse existing ITSO Bank application functionality as Web services such that other applications can consume these services.

- FR2: Create a Web service client application to consume services.

  Modify the existing Web application used by online bank customers to invoke services to access bank application functionality that is now service-enabled.

  Modify the .NET client application used by bank tellers to consume services to access bank application functionality that is now service-enabled. Note, the sample application does not include a .NET Web services client application.

2.1.4 System context diagram

Now that we have more specific technical requirements, we can continue to evolve the picture of the solution in the system context diagram. Figure 2-2 shows the inputs and outputs of the system, as well as the context in which they are used.

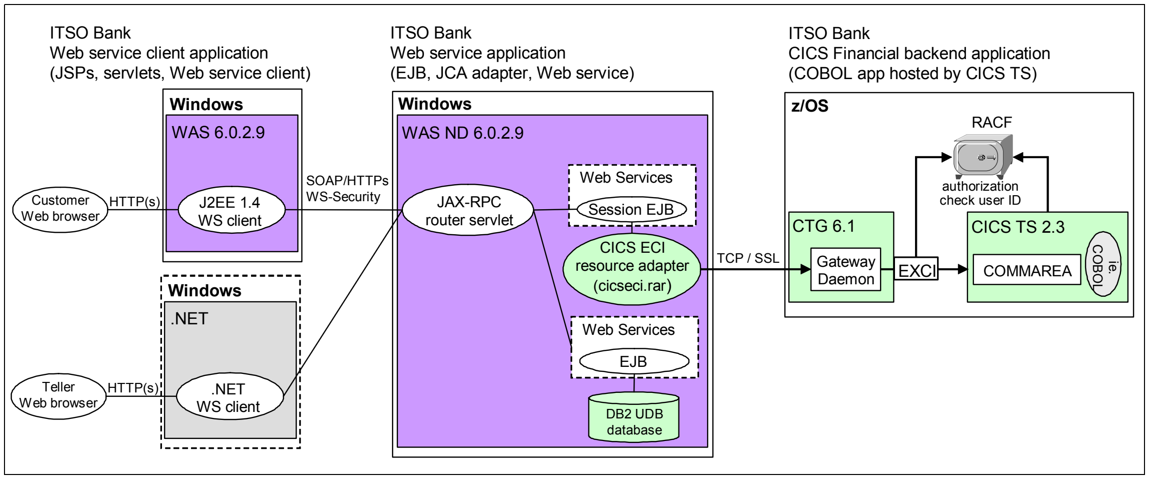


Figure 2-2 System context diagram

As seen in Figure 2-2, the existing ITSO Bank application functionality is now exposed as Web services. The client applications have been modified to consume the exposed services.

The sample application includes an application that can perform the get customer information transaction by retrieving data from a DB2 UDB database, and an application hosted by CICS.

- Service that retrieves data from a DB2 UDB database

  Customer interacts with the ITSO Bank J2EE-based Web services client application via a Web browser. The Web services client application is hosted by WebSphere Application Server. The Web services client application consumes the get customer information Web service that wraps an EJB (hosted by a WebSphere Application Server) to retrieve data from the DB2 UDB database.

- Service that retrieves data from an application hosted by CICS

  The customer interacts with the ITSO Bank Web services client application via a Web browser. The Web services client application is hosted by WebSphere Application Server. The Web services client application consumes the get customer information service component hosted by WebSphere Application Server. The service component is a Web service that wraps a session EJB, which uses the CICS ECI resource adapter to communicate with the CICS Transaction Gateway to access the application hosted by the CICS Transaction Server.

Note: The remaining sections of this chapter and subsequent chapters focus on the end-to-end solution for SOA management.

2.2 Non-functional requirements and SLAs

This section is organized as follows:

- Key concepts of non-functional requirements and SLAs
- Entry points for non-functional requirements and SLAs
- Common categories of non-functional requirements and SLAs

2.2.1 Key concepts of non-functional requirements and SLAs

To clarify what we mean by non-functional requirements and service level agreements, we start by providing a definition for each. Next, we describe the relationship between non-functional requirements and SLAs. Finally, we describe tooling, such as IBM Tivoli Service Level Advisor, that can be used to effectively manage SLAs.

Non-functional requirements

Functional requirements describe the functional behavior of the application in support of user roles, tasks, or activities. Non-functional requirements are used to define requirements of the supporting infrastructure used to host the application. Some common categories of non-functional requirements are availability, performance (response time), usage, and security.

Non-functional requirements include properties or characteristics of the system that its stakeholders care about and hence will affect their degree of satisfaction with the system. The non-functional requirements can be used as a basis to define internal and external service level agreements, and to help determine what resources should be managed.

The Customer Liaison role is responsible for interfacing with the customer (external or internal) to capture the non-functional requirements.

Service level agreements

Service Level Management is the process of negotiating, defining, and managing the levels of IT service that are required and cost-justified. A key goal of Service Level Management is to establish specific metrics for evaluating the quality of service. SLAs are used to define, monitor and report the quality of service provided by an internal IT organization or an external service provider.

A Service Level Agreement (SLA) is an agreement or contract between a service provider and a customer of that service, which sets expectations for the level of service with respect to availability, performance, and other measurable objectives.

There are three key activities of an SLA we would like to highlight. First, the SLA should be defined with specific metrics. Second, there needs to be a method of monitoring the service level objective. Third, a method of analyzing the adherence to the service level objective should be established, such as the use of reports and trend analysis. If the IT department detects a trend towards violation of an SLA, it can take proactive measures to avoid violating the SLA. In practice, it is common to use tooling to more effectively create, monitor, and analyze SLAs.

The Service Level Manager role is responsible for the quality and integrity of the Service Level Management process. The definition of service level agreements and verification of compliance is a key activity of the Service Level Management process.

Relationship of non-functional requirements and SLAs

Both non-functional requirements and SLAs are key sources of information to determine what resources should be managed. There are two key points to consider regarding the relationship between non-functional requirements and SLAs.

First, SLAs can be derived from non-functional requirements with a goal of providing more specific quality of service (QoS) metrics. Note, non-functional requirements do not always have to be mapped to an SLA.

Second, we would like to introduce the concept of OLAs versus SLAs. SLAs define agreements between the customer and the service provider. The customer may be internal to the organization (for example, a customer service department) or an external customer or partner (for example, a supplier or an end customer). Conversely the service provider may be internal or external.

Operational Level Agreements (OLAs) might be used by the IT department to monitor the internal operations, which enables them to provide service to their customers on a reliable basis. In some cases, IT departments will use OLAs to map non-functional requirements. For example, an SLA may define specific response time metrics of a Web service, whereas an OLA may define the need to monitor the availability of supporting infrastructure hosting the Web service. OLAs are intended to be evaluated by the IT department, not customers. OLAs are referred as “Internal SLAs” in IBM Tivoli Service Level Advisor.

Tooling used to manage SLAs

There are many possible tools that can be used to manage SLAs. Our focus in this section is on the capabilities provided by IBM Tivoli Service Level Advisor (TSLA).

TSLA provides the capability to create SLAs in a Web browser interface and store them in a database. In addition, TSLA can be integrated with IBM Tivoli Monitoring to analyze the data collected and report the adherence to the SLA.

TSLA simplifies and automates the process of managing service level agreements, enabling IT organizations to proactively manage and report on service levels from across the management infrastructure. TSLA is a predictive solution for defining, analyzing and reporting on SLAs enterprise wide.

The predictive features of TSLA can be used to create internal SLAs that monitor operational metrics such as CPU utilization, rather than requiring an Operator to visually monitor a variety of consoles to spot a trend or problem. Instead, the Operator can be automatically notified when a disruptive trend has been detected and can work to address the problem before it becomes critical.

The subset of metrics collected by Tivoli Monitoring can be used in conjunction with TSLA to produce best practices SLAs designed to meet the needs of the IT organization. The best practices are a filtered list of metrics that TSLA deems useful in creating SLAs and are categorized into: availability, performance, and utilization. Some metric data collected by Tivoli Monitoring can also be used to derive other metric analysis. In particular, TSLA provides the ability to map service states to well-known availability states so that availability focused data can be derived from state transition data. For example, the running and stopped state associated with the service are transformed to available and unavailable periods and the metric provides a percentage of time spent in states considered available.

Note: Section 2.3, “Manage SLAs using Tivoli Service Level Advisor” on page 32 demonstrates how to use TSLA to create, monitor and analyze an SLA for the redpaper example.

2.2.2 Entry points for non-functional requirements and SLAs

The entry point for capturing non-functional requirements and SLAs can be influenced by the development methodology used. There are many possible methodologies that can be used for customer engagement. We have listed a couple of common methodologies to explain how the methodology might influence how the non-functional requirements and SLAs are captured with the objective of ultimately identifying what resources to manage.

SOMA goal-to-service model

Service-Oriented Model and Architecture (SOMA) is a methodology pioneered by the IBM Software Services for WebSphere (ISSW) organization to address the activities needed to analyze and design a SOA. SOMA includes techniques for the identification, specification, and realization of services, their flows and composition, as well as the enterprise-scale components needed to realize and ensure the quality of services required of a SOA.

A key component of SOMA as it relates to SOA management is the *goal-to-service model*. The key objective of the goal-to-service model is to demonstrate traceability and alignment of services with your enterprise’s business goals.

The goal-to-service model is a middle-out approach used to validate completeness of the list of candidate services identified through the domain decomposition and existing asset analysis techniques. In addition, the goal-to-service model is used to reveal new candidate services that were not identified through the top-down and bottom-up approaches.

When developing the goal-to-service model, you typically work closely with business executives, analysts, and subject matter experts to identify the goals of the business within the scope and phase of the project. For each goal and subgoal, you identify key performance indicators (KPIs) and metrics that will be used to assess success of the business or performance of the organization.

The SOMA techniques are incorporated into the RUP® for SOA plug-in found in the Rational Method Composer V7.0.1.

IBM Process Reference Model for IT

The IBM Process Reference Model for IT (PRM-IT) is an IBM instantiation and extension of the Information Technology Infrastructure Library (ITIL). PRM-IT includes processes for Change and Configuration Management Database (CCMDB).

PRM-IT is used by Tivoli Services and IBM Global Services. IBM Tivoli Unified Process (ITUP) Composer V2.1.1 provides a Rational Method Composer (RMC) plug-in for ITUP content used to implement PRM-IT. ITUP includes processes that are used within the scope of identifying non-functional requirements. The key processes are Service Level Management, Solution Requirements, and Solution Analysis and Design.

2.2.3 Common categories of non-functional requirements and SLAs

Although there are many unique possibilities for non-functional requirements and SLAs, they typically can be categorized into the following four types:

- Availability
- Performance
- Usage
- Security

This section describes best practices for each of the defined categories and provides examples within the context of the redpaper business scenario.

Availability

Availability is defined as a system resource being accessible in a timely fashion. Availability is usually expressed as a percentage of time the resource is available for service.

High availability refers to a specific type of availability defined as the ability for an application or supporting infrastructure to run for an extended period of time with no (or minimal) unplanned outage. High availability is often measured as a percentage of absolute availability. For example, 100% means the resource is always available. Although striving for 100% operational high availability may be a goal, it is very difficult to achieve in practice.

Table 2-1 provides a summary of the industry standard method of measuring high availability, known as the *Five 9s*. The percentage of availability is calculated as follows:

```
Availability % = ((total time - total downtime) / total time)
```

Table 2-1 Five 9s of high availability

Measure	Outage per year
99.999 %	5 minutes
99.99 %	53 minutes
99.9 %	8.8 hours
99.0	87 hours
90.0 %	876 hours

We describe the Five 9s method of high availability to reinforce the point that you need a method for measuring (calculating) availability and communicating what this means in practical terms to the customer in the SLA. Consider that scheduled outages are not included in the total time. Only unplanned outages are accounted for in the calculation of availability.

In order to achieve high availability, various hardware and software solutions can be used. Some common examples are:

- RAID 5 Disk Arrays
- High Availability Cluster Multi-Processing (HACMP™)
- WebSphere Application Server Network Deployment - Application Server and Web Server Clusters
- DB2 UDB - High Availability Disaster Recovery (HADR)

From a SOA management perspective, monitoring availability is a very common non-functional requirement and SLA type. At the stage of defining non-functional requirements and SLAs, you may not have a detailed picture of the solution architecture. The details of what resources should be monitored for availability may not be known. For this reason, the approach used to define external and internal SLAs may be different.

External SLAs for availability typically are defined at the business level. In our banking example, it is required that the customer transaction service is 99.99% available during peak banking hours. In this case, the SLA does not specify the operational resources that need to be available to achieve this metric. When using Tivoli Service Level Advisor, you need to associate the SLA offering with a specific resource; however, there may be many other resources that are associated with the transaction that are not specifically defined in the external SLA.

Internal SLAs often do require an understanding of the solution architecture. For this reason, internal SLAs may need to be defined in an iterative manner as the Management Architecture analysis and design progresses. For example, after decomposing the service transaction across the architecture layers, you may determine that the application server hosting the Web service client application and Web service application need to be monitored for availability. In this case, an internal SLA can be defined to monitor and analyze the availability of the application servers.

Note: In “Identify resources in SOA Foundation Reference Architecture” on page 47 we describe how to identify resources in the architecture layers.

ITSO Bank example

The ITSO Bank example includes the definition of a schedule for monitoring resources for SLAs, and describes the external and internal availability SLAs for the this example.

The ITSO Bank’s Customer Service Department requires the Web service used to retrieve customer information for online customers to be available 24 hours a day, 7 days a week, with the exception of scheduled maintenance. The maintenance schedule is as follows:

- Nightly account processing from 12 am - 3 am

  The nightly account update process runs from 12 am to 3 am. While the ITSO bank is open, it is absolutely critical that service be available. The nightly processing job is also important but can tolerate limited unavailability times since it normally runs for only 2.5 hours and is scheduled for a 3 hour period.

- System maintenance 1st Saturday of month from 5 am to 10 am

  There is a scheduled maintenance period every first Saturday of the month from 5 am to 10 am to allow for hardware or software changes.

When using Tivoli Service Level Advisor, the IT Administrator can create a Business Schedule that matches the ITSO Bank constraints. The Business Schedule will reflect the maintenance schedule listed.

Table 2-2 provides a summary of availability related external and internal SLAs for the redpaper example. We use the following convention to reference the SLAs:

- ESLA1xx: External SLAs, where E denotes external, 1 denotes availability, and XX is the specific SLA number.
- ISLA1xx: Internal SLA, where I denotes internal, 1 denotes availability, and XX is the specific SLA number.

> **Note:** As stated in “Relationship of non-functional requirements and SLAs” on page 23, internal SLAs can be used to track non-functional requirements of the IT organization. Internal SLAs are often derived iteratively, and with more detail once the Management Architecture analysis and design is performed.

- Base environment (DB2), and Enterprise environment (CICS)

Table 2-2 External and internal availability SLAs

SLA title	Description	Environment
ESLA101: Get customer information service 99.9% available.	Monitor and report on the availability of the get customer information service with a QoS metric of 99.9%. **Note:** ITCAM for SOA does not provide a specific attribute to monitor availability of a service. The Tivoli Monitoring - Universal Agent can be used to track the availability of the URL of the Web service.	* Base * Enterprise
ISLA102: Monitor WAS - application server availability in support of ESLA101.	Monitor the availability of the WebSphere Application Server - application servers hosting the Web services client application, and Web service application.	* Base * Enterprise
ISLA103: Monitor WAS - application server logs for exceptions in support of ESLA101.	Monitor the WebSphere Application Server - application server logs for any exceptions.	* Base * Enterprise
ISLA104: Monitor WAS logs for CICS ECI resource adapter connection exceptions in support of ESLA101.	Monitor the WebSphere Application Server - application server logs for CICS ECI resource adapter connection exceptions.	* Enterprise
ISLA105: Monitor availability of CTG in support of ESLA101.	Monitor the availability of the CICS Transaction Gateway (CTG) connection.	* Enterprise

SLA title	Description	Environment
ISLA106: Monitor availability of CICS TS in support of ESLA101.	Monitor the availability of the CICS Transaction Server.	* Enterprise
ISLA107: Monitor availability of the operating system for each node in support of ESLA101.	Monitor availability of the operating system for each node of the service. **Note:** We did not implement the Tivoli Monitoring Operating System Agent.	* Base * Enterprise

The Tivoli Service Level Advisor product includes a best practices file containing a list of attributes for the following monitoring agents for Tivoli Monitoring registered by default:

- Operating System Agents
  - Linux
  - Windows®
  - UNIX®
  - i5/OS®
- ITCAM for SOA Agent

Tivoli Service Level Advisor requires the monitoring data being captured in the warehouse database. This is accomplished by enabling historical collection for the agent. The best practice file attributes are based on the monitoring data captured in the warehouse database.

Additional monitoring agents can be configured in Tivoli Service Level Advisor. For details refer to the specific monitoring product and the Tivoli Service Level Advisor InfoCenters.

Note: We did not implement an availability SLA in Tivoli Service Level Advisor for the redpaper example.

Performance

Understanding the performance requirements and metrics can be very important in the success of the solution and how happy the end user is with the solution. A wonderfully architected application will not have customers coming back if it does not perform well.

One of the most powerful metrics for any resource is the simple monitoring of its responsiveness. For example, is it fast enough or has the performance deteriorated, resulting in a long wait? The performance characteristics of the SOA management solution should be robust enough to handle the load of transactions. The stakeholder will specify the satisfactory metric for the responsiveness of the service.

Redpaper example

The ITSO Bank customer service department requires the daily average response time to be less than 10 seconds during the critical banking period.

Table 2-3 provides a summary of *performance* related external and internal SLAs for the redpaper example. We will use the following conventions to reference the SLAs:

- ESLA2xx: External SLAs, where E denotes external, 2 denotes performance, and XX is the specific SLA number.
- ISLA2xx: Internal SLA, where I denotes internal, 2 denotes performance, and XX is the specific SLA number.
- Base environment (DB2), and Enterprise environment (CICS)

Note: Section 2.3, "Manage SLAs using Tivoli Service Level Advisor" on page 32 describes how to create an SLA in TSLA for the redpaper example ESLA202 listed in Table 2-3.

Table 2-3 External and internal performance SLAs

SLA title	Description	Environment
ESLA201: Monitor end-to-end round trip response time of the get customer information transaction to CICS	Monitor end-to-end round trip response time of the get customer information transaction to the CICS backend application. This requires the use of ITCAM for RTT on each of the nodes of the transaction.	Enterprise
ESLA202: Monitor the round trip response time of the get customer information service.	Monitor end-to-end round trip response time of the get customer information service (data retrieved from either DB2 or CICS). This SLA requires use of ITCAM for SOA.	* Base * Enterprise
ISLA203: Monitor the size of message returned by service in support of ESLA202.	Monitor the size of the message returned by the service. ► DB2: less 600 bytes or over 1000 bytes for *AccountListingSQL* service and *retrieveAccounts* operation. ► * CICS: less 900 bytes or over 1500 bytes for the getCustInfo3 service and callCustomer3 operation	* Base * Enterprise

Usage

Understanding the usage requirement is the key to understanding how customers can remain satisfied with the services offered. Some common questions to gather for the usage requirements are:

- Does the customer have specific usage requirement?
- Do you want to offer personalized service based on the customer's profile?

The usage characteristics can distinguish between the customers requiring the service. One can offer the predetermined usage of service to customers based on their privilege.

For example, consider a requirement for an IT organization of a company to restrict the access to their application with an objective to optimize the utilization of the hardware resources. Based on the available infrastructure, the IT organization might want to limit the access to their application to up to 'x' number of times within a specific interval.

The following SLA can be created to manage the usage requirements:

- Monitor the number of calls made to a service within a specific interval.

Table 2-4 Internal usage SLA

SLA Title	Description	Environment
ISLA301: Monitor the number of calls made to the service within the specified interval	Monitor the message count returned by the service within the specified interval of 1 minute.	* Base

Security

Virtually every SOA implementation encompasses the need for a security architecture that enables secure business transactions across and within enterprises. The *SOA Security* reference architecture includes security services such as identity services, authentication services, authorization and privacy services, message protection services, and audit services. Other important facets of this reference architecture include security policy and security management infrastructures.

Although security is a required dimension of the architecture, the caveat is that enabling security can have an impact on the performance of the infrastructure and application. The enablement of security can impact performance-related SLAs. Also consider that SLAs may be used to ensure security components are being monitored.

2.3 Manage SLAs using Tivoli Service Level Advisor

This section demonstrates how to create an SLA using Tivoli Service Level Advisor (TSLA) for the redpaper example scenario. In addition, we describe how to analyze reports for the defined SLAs. The reports contain data on the results of the Tivoli Monitoring - TEP situation associated with the SLA.

The section is organized as follows:

- Prerequisites for Tivoli Service Level Advisor
- Create an SLA in Tivoli Service Level Advisor
- Analyze a Tivoli Service Level Advisor report

2.3.1 Prerequisites for Tivoli Service Level Advisor

Prior to creating Tivoli Service Level Advisor (TSLA) reports, ensure the following tasks have been completed:

- Install Tivoli Service Level Advisor
- Configure the history collection
- Create an SLA in Tivoli Service Level Advisor
- Trigger the situation

Install Tivoli Service Level Advisor

For details on installing Tivoli Service Level Advisor (TSLA), refer to the *Quick Start Installation Example using IBM Tivoli Monitoring V6.10* appendix in the *Getting Started, IBM Tivoli Service Level Advisor V2.1.1*, SC32-0834-04 product guide.

Important: During TSLA installation, we chose to configure data retrieval from IBM Tivoli Monitoring 6.10 data warehouse database. After the TSLA installation, we recommend registering the best practices data only for the component types to be evaluated in an SLA.

Configure the history collection

TSLA is dependent on data being captured in the Tivoli Monitoring Server - Warehouse database. Complete the following steps to configure history collection:

1. Tivoli Monitoring - Warehouse proxy.

   Ensure the Warehouse Proxy is configured and started from the Manage Tivoli Enterprise Monitoring Services Console.

2. History Collection.

   Ensure the History Collection is configured to 5 minute Collection interval, and 1 hour Warehouse interval for the ITCAM for SOA Services_Inventory and Services_Message_Metric Group.

   For details on configuring the history collection, refer to the "Completing the configuration" chapter in the *Installation and User's Guide, ITCAM for SOA*, GC32-9492-00.

   Note: In the high volume production environment, it is recommended to avoid collecting data for the Service_Message_Metric group.

   Note: To verify that measurements are being recorded in the Tivoli Monitoring data warehouse, you might need to wait for up to two warehouse collection intervals to expire (2 hours if hourly was selected).

2.3.2 Create an SLA in Tivoli Service Level Advisor

To create an SLA in Tivoli Service Level Advisor, complete the following steps:

- Access the TSLA Administration Console
- Create a customer
- Create a schedule
- Create an offering
- Create an SLA

Access the TSLA Administration Console

To launch the TSLA Administration Console, enter the following URL:

```
http://<tsla_hostname>:9080/SLMAdmin
```

Create a customer

A customer is a person, department, or company, that enters into an SLA with the provider of a particular service. Customers are grouped together by realms or customer groups.

1. From the TSLA Administration Console, click **Create Customer**.

   Note: Each Create function has a Welcome page. We found this information to be useful in self-documenting the task. Click **Next** on the Welcome page to continue.

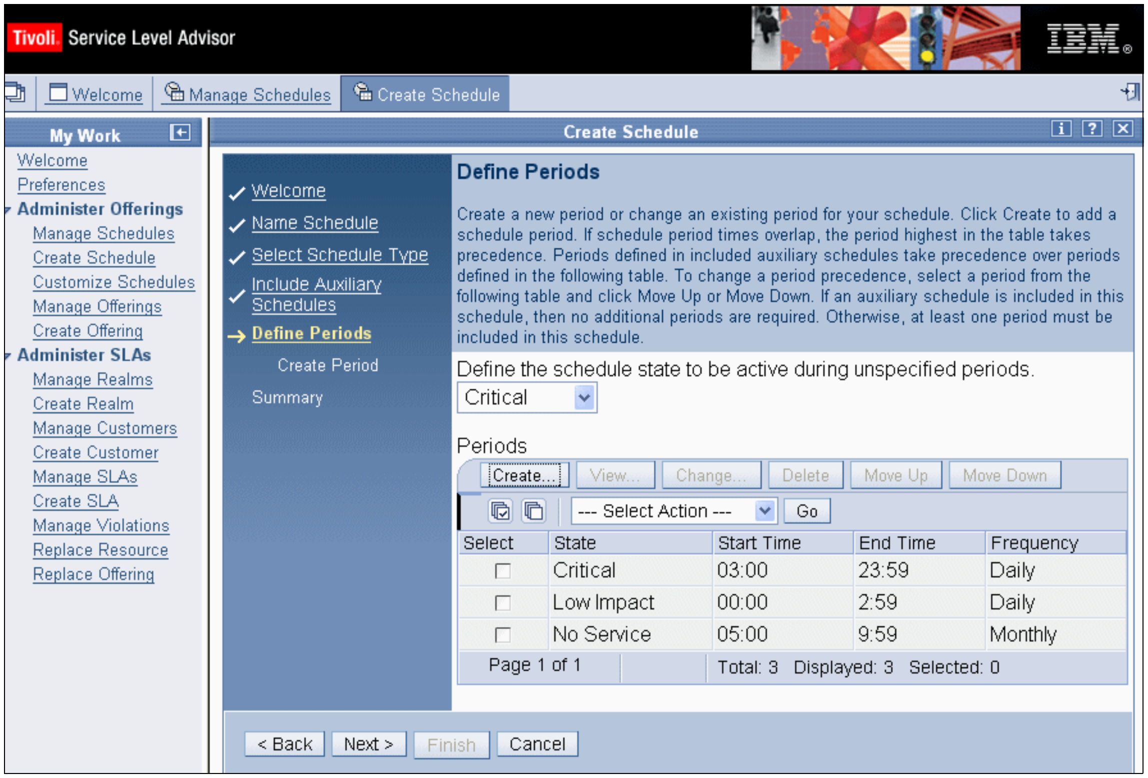
2. When the Name Customer page appears, enter `ITSO Bank` in the Customer Name field and click **Next**.
3. When the Include Realms page appears, click **Add**.
4. When the Select Realms page appears, select **Create a new realm** and click **OK**.
5. When the Name Realm page appears, enter `East` in the Realm Name field and click **Next**.
6. Click **Next** again and the summary will be displayed.
7. Click **Finish** and the customer will be created.

Create a schedule

To create a schedule in TSLA, do the following:

1. Click **Create Schedule** and click **Next** on the welcome screen.
2. In the Name Schedule page, enter `Business Schedule for ITSO Bank` in the Schedule Name field, and click **Next**.
3. In the Select Schedule Type page, select **Business Schedule** and click **Next**.
4. Click **Next** in the Include Auxiliary Schedules page.
5. In the Define Periods page, click **Create**.
6. In the Create Period page, enter the following and click **OK**:
   - State: select **Critical**
   - Time Zone: select **GMT-05:00 America/New York**
   - Start Time: select **03:00**
   - End Time: select **23:59**
   - Frequency: select **Daily**
7. In the Define Periods page, click **Create**.
8. In the Create Period page, enter the following and click **OK**:
   - State: select **Low Impact**
   - Time Zone: select **GMT-05:00 America/New York**
   - Start Time: select **00:00**
   - End Time: select **2:59**
   - Frequency: select **Daily**
9. In the Define Periods page, click **Create**.

10. In the Create Period page, enter the following and click **OK**:
    - State: select **No Service**
    - Time Zone: select **GMT-05:00 America/New York**
    - Start Time: select **05:00**
    - End Time: select **9:59**
    - Frequency: select **Monthly**
    - Select **Day Interval**
    - Interval: select **1st**
    - Day of Week: select **Saturday**
11. When the Define Periods page reappears, verify that the periods are created as shown in Figure 2-3 and click **Next.**



Define Periods

Create a new period or change an existing period for your schedule. Click Create to add a schedule period. If schedule period times overlap, the period highest in the table takes precedence. Periods defined in included auxiliary schedules take precedence over periods defined in the following table. To change a period precedence, select a period from the following table and click Move Up or Move Down. If an auxiliary schedule is included in this schedule, then no additional periods are required. Otherwise, at least one period must be included in this schedule.

Define the schedule state to be active during unspecified periods.

Critical

Periods

Select	State	Start Time	End Time	Frequency
☐	Critical	03:00	23:59	Daily
☐	Low Impact	00:00	2:59	Daily
☐	No Service	05:00	9:59	Monthly

Page 1 of 1 Total: 3 Displayed: 3 Selected: 0

Figure 2-3 Create Period

12. When the Summary page appears, click **Finish**.

Create an offering

To create a service level offering, complete the following steps:

1. Click **Create Offering**.
2. When the Name Offering page appears, enter `Ext_Monitoring_svc_Response_time` in the Offering Name field and click **Next**.
3. In the Select SLA Type page, select **External** and click **Next**.
4. Since we have not created an SLA yet, click **Next** in the Included SLAs page.
5. In the Select Business Schedule page, select **Use an existing business schedule** and select **Business Schedule for ITSO Bank** schedule. Click **Next**.
6. In the Included Offering Components page, click **Add**.
7. In the Resource Type Tree, expand **IBM Tivoli Monitoring V6 Agents** and select **ITCAM for SOA**.
8. In the Resource Types, select **Service_Inventory** and click **Next**.
9. In the Included Metrics page, click **Add**.
10. In the Select Metrics page, select **Avg_Elapsed_Time** and click **Next**.
11. In the Define Breach Values page, enter `10000` in the Average field for Critical state and enter `30000` in the Average field for Low Impact state and click **Next**.

    Important: The average elapsed time is measured in milliseconds.

12. In the Evaluation Frequency page, select **Every Two hours** for the evaluation frequency and check **Configure advanced metric settings**.
13. In the Advanced Metric Settings page, select **Every hour** and **Continuous trend evaluation** for the trend analysis and then click **Finish**.
14. In the Included Metrics page, Click **Next**.
15. In the Name Offering Component page, enter `Services_Inventory` in the Offering Component Name field and click **Next**.
16. Click **Next** in the Include Offering Components page.
17. In the Summary screen, select **Publish the Offering** and click **Finish**.

Create an SLA

To create an SLA, complete the following steps:

1. Click **Create SLA**.

2. In the welcome page, click **Next**.
3. In the Name SLA page, enter `ESLA202_monitor_svc_response_time` in the SLA Name field and click **Next**.
4. In the Select Customer page, select **Use an existing customer**, select **ITSOBank**, and click **Next**.
5. In the Select Offering page, select **Ext_Monitoring_svc_Response_time** and click **Next**.
6. In the Add Resources to Service_Inventory page, click **Add**.
7. In the Select Resource Type List page, select **Static Resource List** and click **Next**.
8. In the Select Resources page, do the following and then click **Next**:

   If using the DB2 UDB environment, select the following:

   ```
   /D4:4b7c211a:soa1was1-server1/server1//AccountListingSQL:retrieve
   Accounts:P
   ```
9. In the Add Resources to Service_Inventory page, click **Next**.
10. In the Select SLA Start Date page, enter the appropriate values and click **Next**.
11. In the Summary screen, click **Finish**.

2.3.3 Analyze a Tivoli Service Level Advisor report

Tivoli Service Level Advisor offers a rich reporting interface, which summarizes the results of SLA analysis, providing information in both table and graph format.

Launch TSLA Reporting console using the following URL:

```
http://<tsla_hostname>:9080/SLMReport
```

This will display a colorful high-level summary report in tabular form showing trends and violations across the reporting period, grouped by realms and customers. The cells in the table can be clicked to view additional details about trends and violations.

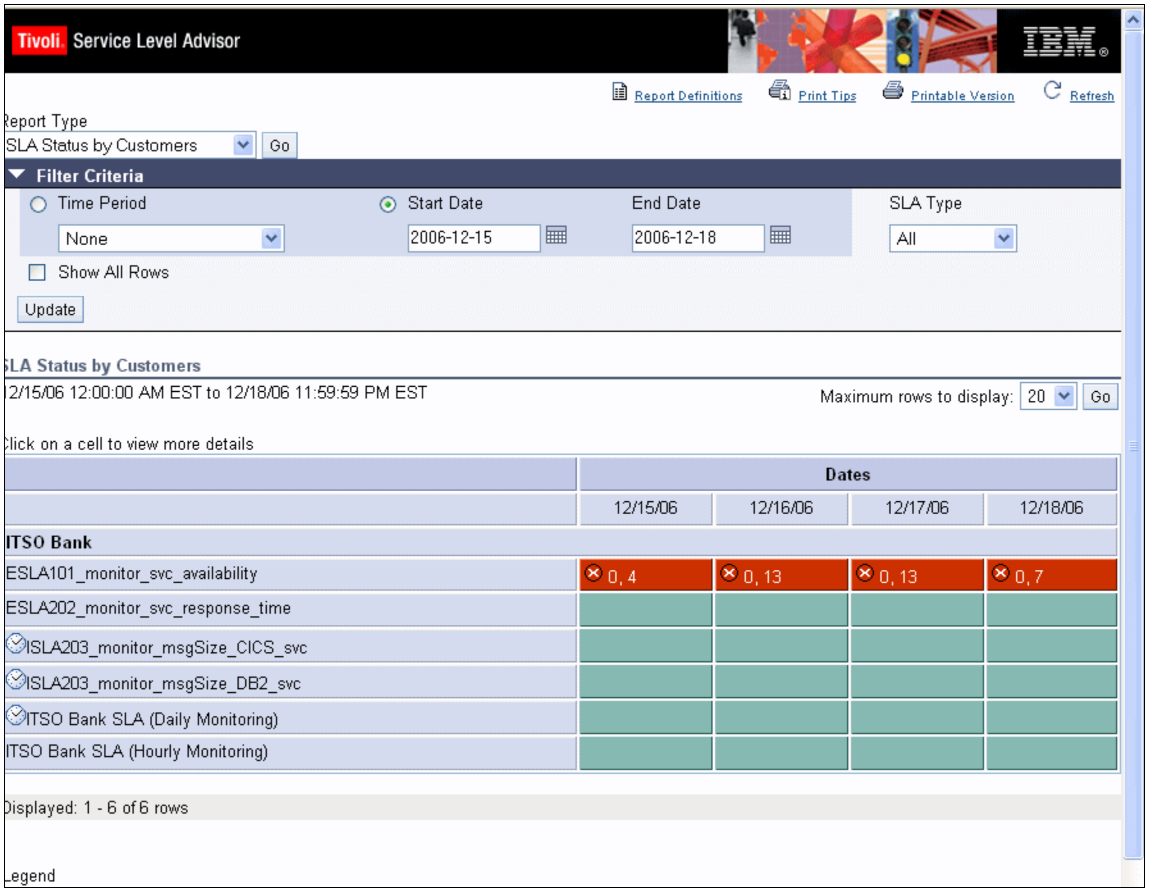
Depending on the view authorization for your user name, you can see the following types of reports:

- Customer Status by Realms
- SLA Status by Customers
- Customer Ranking
- SLA Ranking
- Offering Component Ranking

- Resource Ranking
- SLA Type Ranking
- Realm Ranking
- Overall Details
- SLO Results
- Trends
- Violations

High-level status report

The first report that you see is the high-level status report, which looks like a dashboard and which gives an overall summary of the SLA status. This report provides the user with a broad snapshot view of the SLAs over the indicated time period. Figure 2-4 displays an example of a high-level status report.



Report Type
SLA Status by Customers Go

Filter Criteria

Time Period: None
[✓ Start Date]: 2006-12-15
End Date: 2006-12-18
SLA Type: All
Show All Rows
Update

SLA Status by Customers

2/15/06 12:00:00 AM EST to 12/18/06 11:59:59 PM EST

Maximum rows to display: 20 Go

lick on a cell to view more details

	Dates			
	12/15/06	12/16/06	12/17/06	12/18/06
ITSO Bank				
ESLA101_monitor_svc_availability	🟥 0, 4	🟥 0, 13	🟥 0, 13	🟥 0, 7
ESLA202_monitor_svc_response_time	🟩	🟩	🟩	🟩
ISLA203_monitor_msgSize_CICS_svc	🟩	🟩	🟩	🟩
ISLA203_monitor_msgSize_DB2_svc	🟩	🟩	🟩	🟩
ITSO Bank SLA (Daily Monitoring)	🟩	🟩	🟩	🟩
ITSO Bank SLA (Hourly Monitoring)	🟩	🟩	🟩	🟩

isplayed: 1 - 6 of 6 rows

egend

Figure 2-4 Tivoli Service Level Advisor - High Level Report

Detailed report

The user can click each of the cells in the high-level report to view the associated lower level of operational data. The detail data shows violations, trends, and service level objectives. Figure 2-5 displays an example of a detailed report.

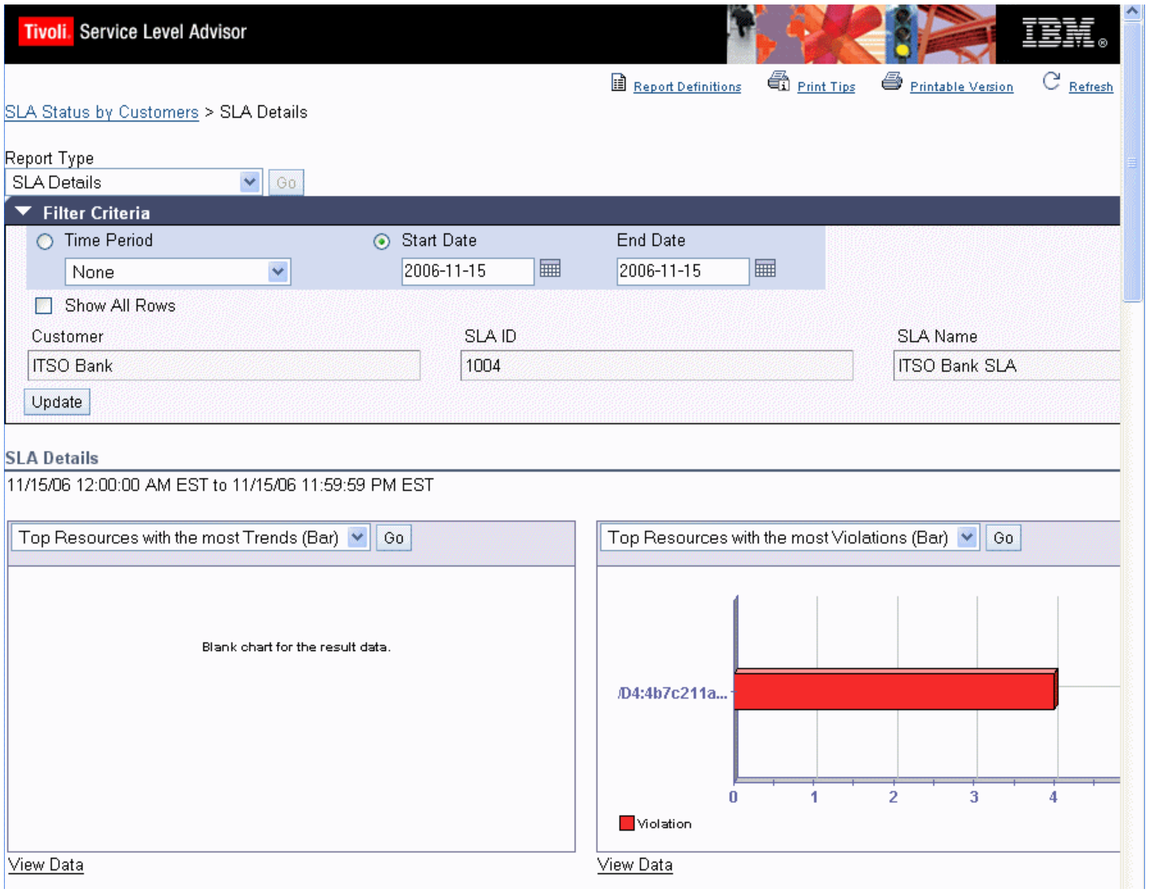


Figure 2-5 Detailed SLA Report for ITSO Bank SLA

Utilizing SLA reporting capabilities within TEP

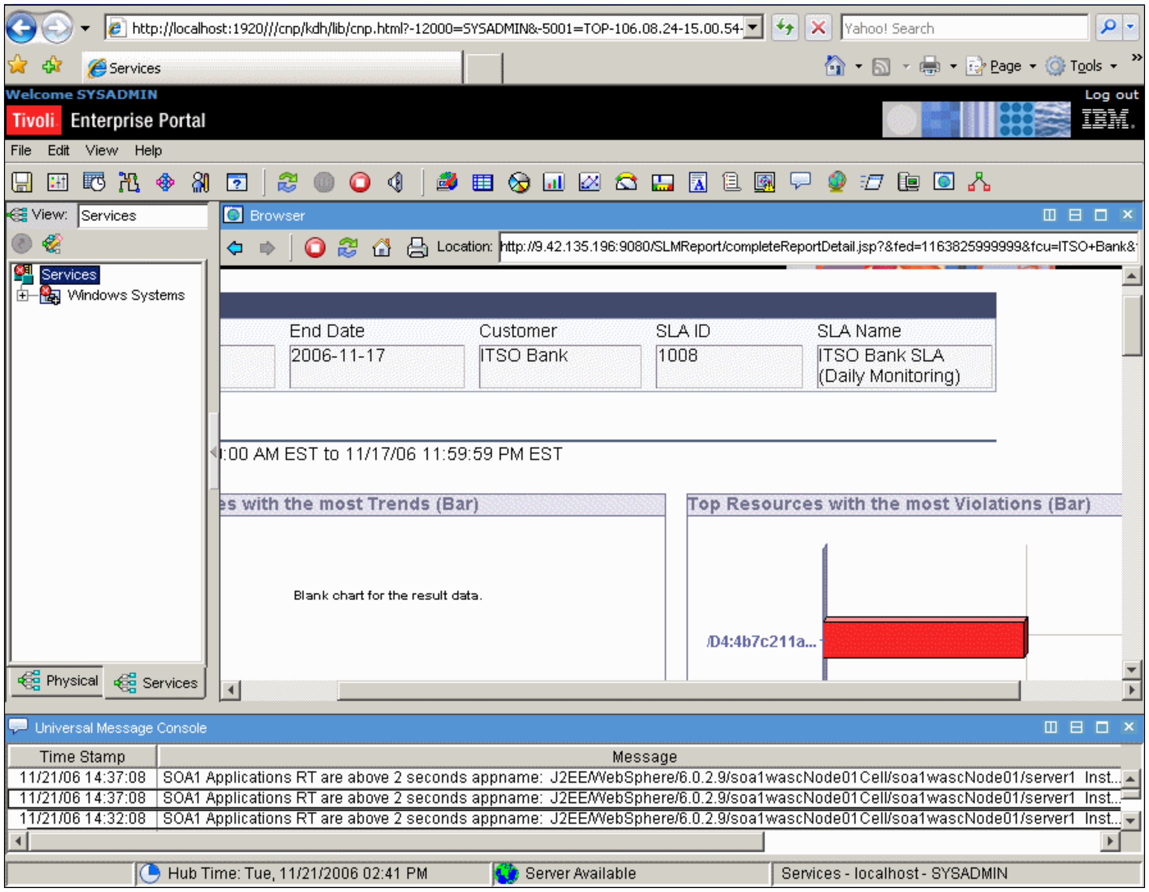
Tivoli Enterprise Portal supports a browser control that can be leveraged to easily integrate TSLA reports into the TEP console.

To configure the TEP console to access Tivoli Service Level Advisor reports, complete the following steps:

1. Launch the Tivoli Service Level Advisor Report Console.

```
http://<tsla_hostname>:9080/SLMReport
```

2. From Tivoli Service Level Advisor Reporting console, click the **Printable Version** link of the report.
3. Copy the URL displayed to the clipboard.
4. Log on to TEP using either the Desktop or Browser client. If you use a browser client, use a separate browser window.
5. From TEP, select the desired workspace and click the browser icon in the toolbar, and then click the empty view within the workspace. This will launch a browser session in the desired workspace.
6. Copy the URL from the TSLA Reporting Console into the browser control inside Tivoli Enterprise Portal. This will integrate the TSLA report into the Tivoli Enterprise Portal workspace.
7. Save this workspace so that each time this workspace is accessed, the latest SLA details are shown. Figure 2-6 displays an example of the integrated TSLA report with the existing Tivoli Enterprise Portal workspace.


http://localhost:1920///cnp/kdh/lib/cnp.html?-12000=SYSADMIN&-5001=TOP-106.08.24-15.00.54-
Yahoo! Search
Services
Page
Tools
Welcome SYSADMIN
Log out
Tivoli Enterprise Portal
IBM.
File Edit View Help
View: Services
Services
Windows Systems
Physical
Services
Browser
Location: http://9.42.135.196:9080/SLMReport/completeReportDetail.jsp?&fed=1163825999999&fcu=ITSO+Bank&
es with the most Trends (Bar)
Blank chart for the result data.
Top Resources with the most Violations (Bar)
/D4:4b7c211a...
Universal Message Console

End Date	Customer	SLA ID	SLA Name
2006-11-17	ITSO Bank	1008	ITSO Bank SLA (Daily Monitoring)

:00 AM EST to 11/17/06 11:59:59 PM EST

Time Stamp	Message
11/21/06 14:37:08	SOA1 Applications RT are above 2 seconds appname: J2EE/WebSphere/6.0.2.9/soa1wascNode01Cell/soa1wascNode01/server1 Inst...
11/21/06 14:37:08	SOA1 Applications RT are above 2 seconds appname: J2EE/WebSphere/6.0.2.9/soa1wascNode01Cell/soa1wascNode01/server1 Inst...
11/21/06 14:32:08	SOA1 Applications RT are above 2 seconds appname: J2EE/WebSphere/6.0.2.9/soa1wascNode01Cell/soa1wascNode01/server1 Inst...

Hub Time: Tue, 11/21/2006 02:41 PM | Server Available | Services - localhost - SYSADMIN

Figure 2-6 View TSLA Report in the TEP console

3

Identify resources to manage

One of the key challenges of SOA management is determining what resources should be managed.

This chapter includes best practices for identifying resources to be managed. In addition, it provides guidance on capturing the management architecture and operation model so that the IT administrator will have the information needed to implement the management environment.

The chapter is organized into the following sections:

- Overview of identifying resources to manage
- Identify resources during analysis and design
- Design the management architecture
- Design the operational model

3.1 Overview of identifying resources to manage

This section provides an overview of approaches for identifying resources, and factors that influence which approach to use.

3.1.1 Approaches for identifying resources to manage

There are two fundamental approaches for identifying resources to be managed: design and discovery. These approaches can be used together or separately.

From a roles perspective, the Enterprise Architect and Management Architect are responsible for defining what resources should be managed to fulfill the requirements of the non-functional requirements, SLAs, and overall solution. The Application Developer, IT Administrator, and Incident Analyst provide input on what resources should be managed based on knowledge of what is possible with the management software.

Design

In this approach, resources to be managed are identified at the time of analysis and design based on non-functional requirements and SLAs. The identification of resources at design time requires an understanding the application architecture and supporting infrastructure.

The key high-level tasks are as follows:

- Identify services and end-to-end transactions.

  The Enterprise Architect analyzes the non-functional requirements and SLAs to identify specific service transactions that need to be managed. For example, an SLA might define a specific service metric, such as average response time no greater than five seconds.

  The Enterprise Architect works with the Application Developer to better understand the specific application components and supporting infrastructure for the application. For example, the Application Developer can provide the specific service name and operation name of the service.

- Identify resources in SOA Foundation reference architecture.

  The Enterprise Architect decomposes the service transaction across the SOA Foundation reference architecture layers. This helps determine the specific resource types of the application and supporting infrastructure.

  The Enterprise Architect and Management Architect then map the appropriate management products for each layer and resource type identified for the transaction. For example, the ITCAM for SOA product will be used to monitor service consumers and providers.

- Define criteria to monitor the resources identified.

  Now that the service transaction and resource types are identified, the Enterprise Architect and Management Architect can work with the IT Administrator and other subject matter experts (SMEs) to define specific criteria to monitor the resources. For example, the Management Architect may document the need to create a TEP situation to monitor the get customer information request service average round trip response time, with a threshold of 5 seconds to achieve the defined SLA.

Note: For more information refer to 3.2, “Identify resources during analysis and design” on page 46.

Discovery

In this approach, the application is deployed to either a development test environment or a production environment and then a discovery of key resources to manage is performed based on analysis of how the application is used. Discovery can be used to identify the application and supporting infrastructure details based on use, as well as to automate the creation of situations (thresholds) for monitoring the resource.

Discovery can be used in addition to identifying resources in the analysis and design approach. For example, consider the get customer information service request identified with the need to measure round trip response time. The ITCAM for SOA discovery feature (services inventory) can be used to *discover* the service name and operation name, both of which are needed to create the TEP situation for monitoring. Also, the use of the discovery feature is desirable to help automate the creation of situations or listening monitors.

In production environments where the application is already deployed, the application details may not be known by the operations team. In cases such as this, once the Tivoli Monitoring and ITCAM products are deployed, the application can be exercised by performing a workload analysis of the common uses of the application. For example, a baseline of the performance metrics can be established. The infrastructure and application can be tuned and new performance data can be analyzed after rerunning the application.

Note: For more information refer to 4.2, “Discovery of resources” on page 71.

3.1.2 Factors that influence the approach for identifying resources

There are several factors that can influence the approach an Enterprise Architect and Management Architect will take when identifying resources to manage for SOA composite applications.

New or existing

Is this a green field environment started from requirements and design, or is this an existing environment? In situations where infrastructure is already in place, there is a level of discovery that can be performed using features provided with the Tivoli Monitoring and ITCAM products to obtain information about the relationships of the resources.

Scope of the project

If the scope is relatively small, then a manual approach for identifying resources to manage may be feasible. If on the other hand, if the number of services or scope of the resources to manage is very large, it may be necessary to use a more automated method of identifying resources via discovery.

Methodology

The best practices we describe can be applied to a variety of methodologies. Different methodologies, such as RUP/ITUP, IBM Method, and SOMA, might lead to different entry points for establishing non-functional requirements and SLAs of what should be managed.

Tooling

From a design perspective, the identification of resources to manage does not depend upon a specific tool. With that said, there are several IBM products that can be found within this solution domain, including Rational Method Composer, Tivoli Unified Process tool, Tivoli Service Level Advisor, and Tivoli Change and Configuration Management Database (CCMDB).

3.2 Identify resources during analysis and design

This section demonstrates how to identify resources at the time of the solution analysis and design by following the high-level steps defined in “Approaches for identifying resources to manage” on page 44 for the redpaper example scenario.

3.2.1 Identify services and end-to-end transactions

When looking at a SOA composite application, it is important to understand the relationships of service consumers and providers for composite applications and supporting infrastructure. The Enterprise Architect analyzes the non-functional requirements and SLAs to identify specific transactions that need to be managed.

In our example business scenario we defined the following services:

- Create account
- Get customer information
- Get account balance
- Transfer funds

We focus on the get customer information service in this discussion because this is the only service transaction implemented in the sample application. When analyzing the non-functional requirements and SLAs, it was determined that monitoring the end-to-end response time is needed.

As described in the example system context diagram in 2.1.4, “System context diagram” on page 20, the sample application includes an application that can perform the get customer information transaction by retrieving data from a DB2 UDB database or an application hosted by CICS.

3.2.2 Identify resources in SOA Foundation Reference Architecture

This section demonstrates how the Enterprise Architect decomposes the service transaction for the redpaper example across the SOA Foundation Reference Architecture layers. The Enterprise Architect works with the Management Architect to identify the specific resource types for each layer of the application and supporting infrastructure. This information is then used to map the appropriate management products to manage the resources.

As noted previously, our sample application includes the ability to get customer information from a DB2 UDB database or a CICS application, depending on your deployment environment (see Figure 3-1 on page 49). We have included a decomposition of the get customer information service transaction for both DB2 UDB and CICS application samples.

- Data retrieved from DB2 UDB:
  - Monitor services:

    A Web service exposes the application functionality to retrieve the customer information from a DB2 UDB database. The Web service can be consumed by a Web service client application. ITCAM for SOA is used to monitor the services provider and consumer.
  - Monitor transactional performance:

    In this case, transactional performance between the Web services client application and Web service can be monitored using ITCAM for SOA.

  - Monitor middleware:

    The Web services client application, Web service and business logic are dependent on the application server hosting these applications. ITCAM for WebSphere is used to monitor the availability of these resources.

- Data retrieved from CICS Transaction Server:
  - Monitor services:

    A Web service exposes the application functionality of the session bean and CICS ECI resource adapter used to communicate with the CICS Transaction Gateway to access the backend COBOL application hosted by CICS Transaction Server. The Web service can be consumed by a Web service client application. ITCAM for SOA is used to monitor the services provider and consumer. The service response time reported by ITCAM for SOA includes the time in the EJB and CICS backend.

  - Monitor transactional performance

    In this case, transactional performance needs to go end-to-end from the Web services client application, to the Web service, which then communicates with CICS. In order to get the end-to-end transaction performance, ITCAM for RTT is needed in this environment to correlate and provide a breakdown of each segment of the transaction (Figure 3-1 on page 49).

  - Monitor middleware

    The Web services client application, Web service and business logic are dependent on the application server hosting these applications. ITCAM for WebSphere is used to monitor the availability of these resources.

  - Monitor operational systems

    In this case, the customer information is being retrieved from the backend COBOL application hosted by CICS Transaction Server. Both the CICS Transaction Server and CICS Transaction Gateway must be running properly for the transaction to be successful. OMEGAMON XE for CICS is used to monitor CICS Transaction Server. ITCAM for WebSphere and ITCAM for CICS Transaction can be used to monitor the success or exceptions of connecting to CTG from the application using the CICS ECI resource adapter.

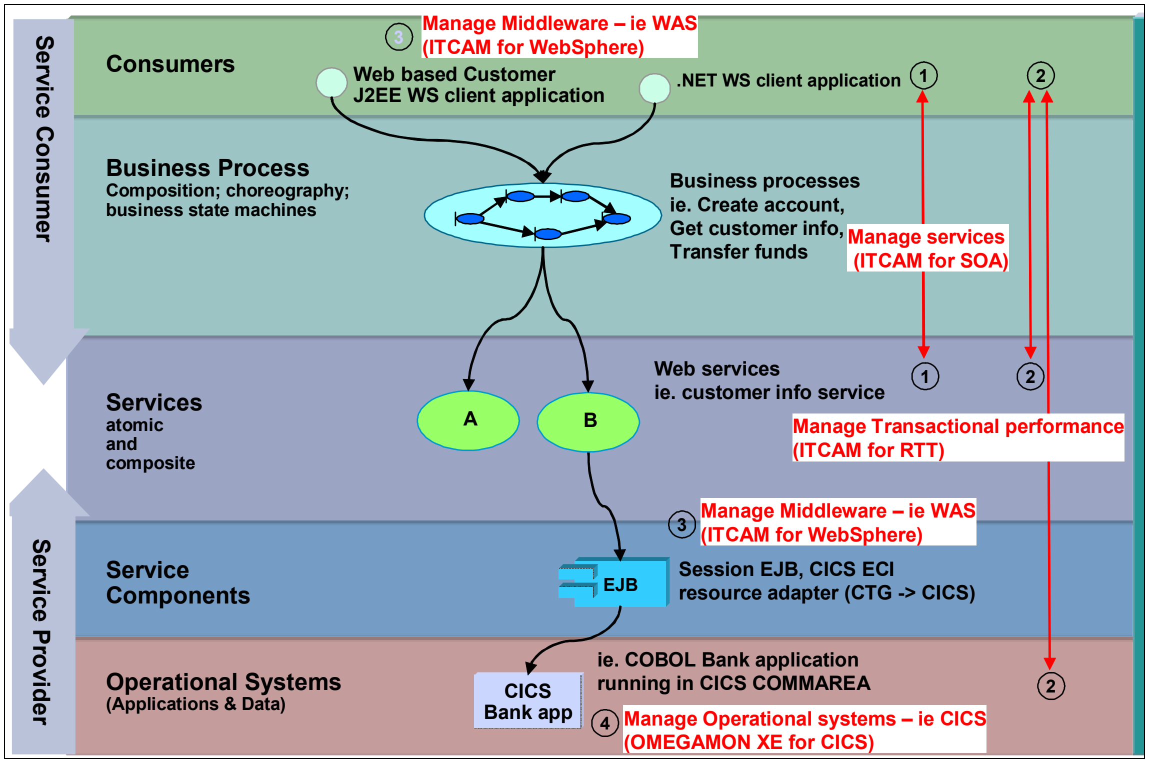

Service Consumer
Service Provider
Consumers
3
Manage Middleware – ie WAS (ITCAM for WebSphere)
Web based Customer J2EE WS client application
.NET WS client application
1
2
Business Process
Composition; choreography; business state machines
Business processes ie. Create account, Get customer info, Transfer funds
Manage services (ITCAM for SOA)
Services
atomic and composite
A
B
Web services ie. customer info service
1
2
Manage Transactional performance (ITCAM for RTT)
3
Manage Middleware – ie WAS (ITCAM for WebSphere)
Service Components
EJB
Session EJB, CICS ECI resource adapter (CTG -> CICS)
Operational Systems
(Applications & Data)
CICS Bank app
ie. COBOL Bank application running in CICS COMMAREA
2
4
Manage Operational systems – ie CICS (OMEGAMON XE for CICS)

Figure 3-1 Identify resources to manage in the SOA Foundation architecture layers

3.2.3 Define criteria to monitor the resources identified

Once the resource types for the service transaction and management products are identified, the Enterprise Architect and Management Architect can work with the IT Administrator and SMEs to define specific criteria to monitor the resources.

The details for this step are in 3.4, “Design the operational model” on page 54.

3.3 Design the management architecture

By decomposing the application across the SOA Foundation Reference Architecture layers, we were able to map the appropriate management software to manage the resource types defined. This section includes logical and physical architecture diagrams for the base environment and enterprise environment where the ITSO Bank application is deployed.

To make our sample application and monitoring environment available to a wider audience, our sample can be deployed to either a base or an enterprise environment.

- Base environment

  The base environment requires fewer resources and nodes on a distributed platform (for example, Windows). In this environment, the Web service retrieves customer information from a DB2 UDB database.

- Enterprise environment

  The enterprise environment is more advanced. It include the base environment plus CICS Transaction Gateway, CICS Transaction Server, and supporting management software on z/OS. In this environment, the Web service retrieves customer data from a COBOL application hosted by CICS Transaction Server.

3.3.1 Management logical architecture

This section includes the logical architecture diagrams and descriptions for the base and enterprise environments. In addition, it highlights the management components added to the logical architecture specific to managing the SOA composite application.

Base environment logical architecture

Figure 3-2 depicts the logical architecture of the base environment used to deploy the ITSO Bank application.

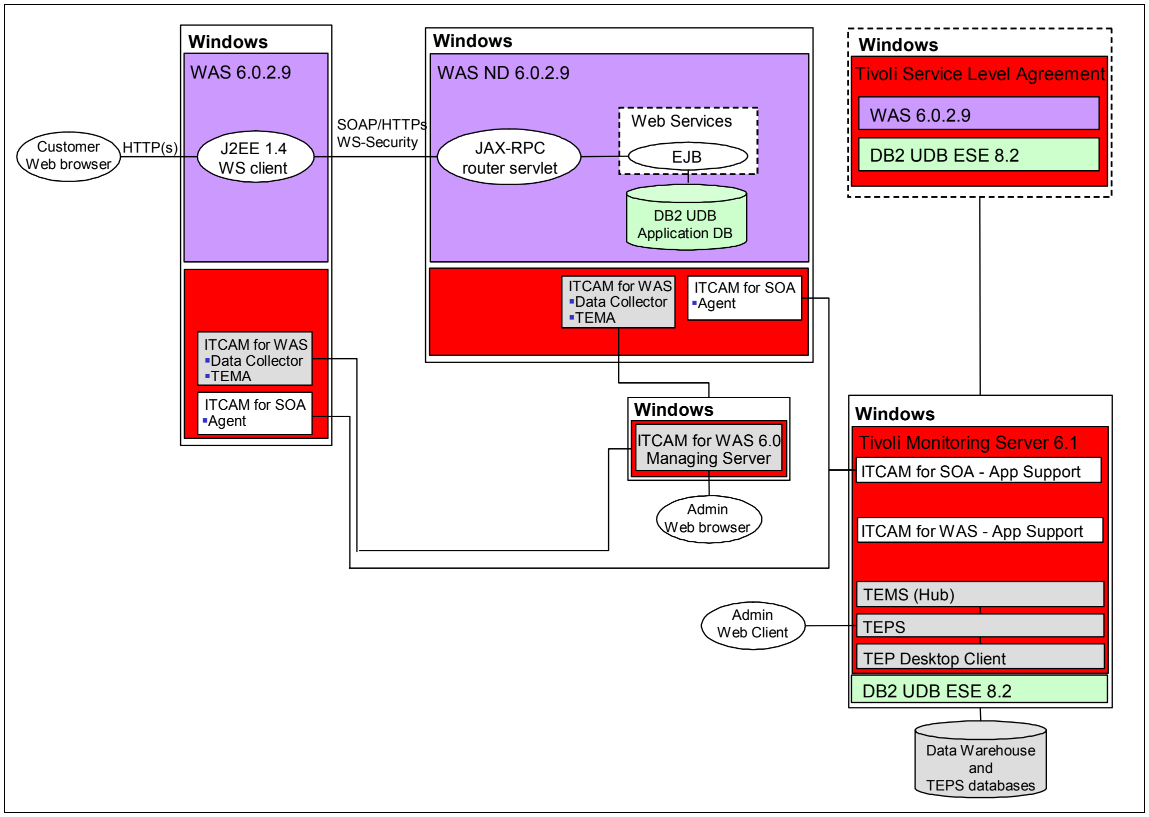


Figure 3-2 Base environment to-be logical architecture

There are two primary additions made to the *to-be* environment when compared to the *as-is* environment. First, in order to manage and monitor service consumers and providers, we added ITCAM for SOA. Second, in order to monitor the resources in a centralized console, we configured ITCAM for WebSphere with the Tivoli Monitoring Server (TEMS / TEP). This provides the capability to create customized workspaces and situations to monitor each resource type.

Enterprise environment logical architecture

Figure 3-3 depicts the logical architecture of the enterprise environment used to deploy the ITSO Bank application.

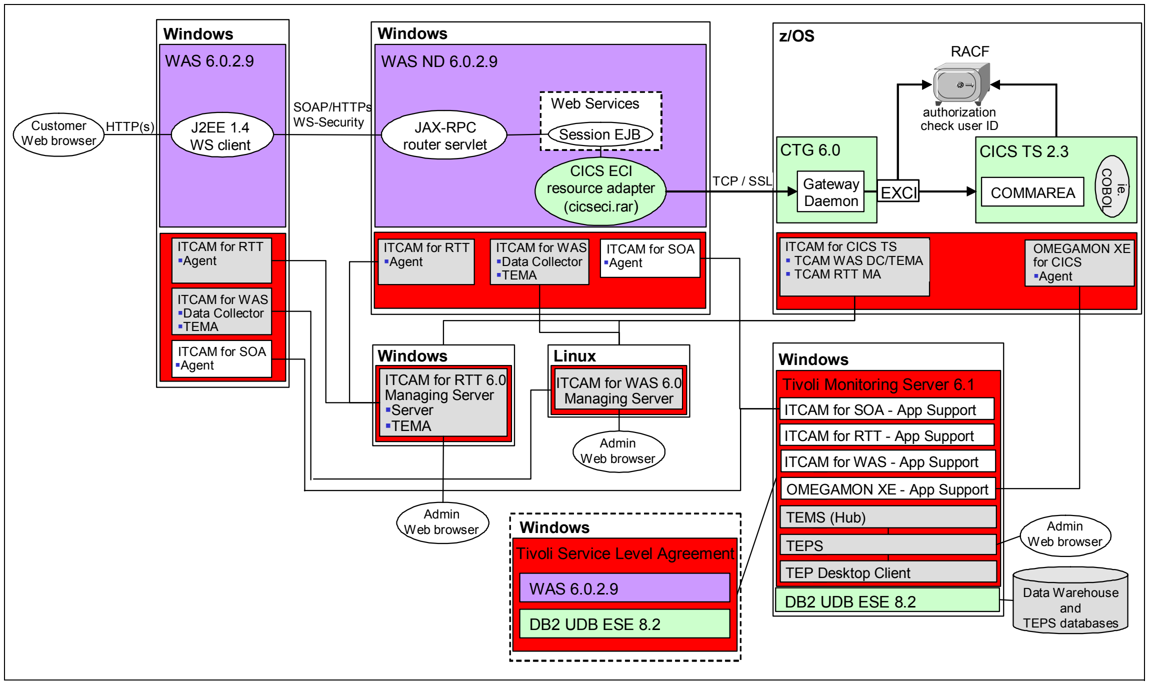

Windows
WAS 6.0.2.9
Customer Web browser
HTTP(s)
J2EE 1.4 WS client
SOAP/HTTPs WS-Security
Windows
WAS ND 6.0.2.9
JAX-RPC router servlet
Web Services
Session EJB
CICS ECI resource adapter (cicseci.rar)
TCP / SSL
z/OS
RACF
authorization check user ID
CTG 6.0
Gateway Daemon
EXCI
CICS TS 2.3
COMMAREA
ie. COBOL
ITCAM for RTT ▪Agent
ITCAM for WAS ▪Data Collector ▪TEMA
ITCAM for SOA ▪Agent
ITCAM for CICS TS ▪ TCAM WAS DC/TEMA ▪ TCAM RTT MA
OMEGAMON XE for CICS ▪Agent
Windows
ITCAM for RTT 6.0 Managing Server ▪Server ▪TEMA
Linux
ITCAM for WAS 6.0 Managing Server
Admin Web browser
Windows
Tivoli Monitoring Server 6.1
ITCAM for SOA - App Support
ITCAM for RTT - App Support
ITCAM for WAS - App Support
OMEGAMON XE - App Support
TEMS (Hub)
TEPS
TEP Desktop Client
DB2 UDB ESE 8.2
Data Warehouse and TEPS databases
Windows
Tivoli Service Level Agreement
WAS 6.0.2.9
DB2 UDB ESE 8.2

Figure 3-3 Enterprise environment to-be logical architecture

There are several additions made to the *to-be* enterprise environment when compared to the *as-is* environment. The changes made were the following:

- In order to manage and monitor service consumers and providers, we added ITCAM for SOA.
- We added ITCAM for RTT to manage the end-to-end transactional performance from the service all the way to the backend CICS application.
- We added the ITCAM for CICS Transactions agent, and ITCAM for WebSphere Data Collector TEMA support, to provide integration with the Tivoli Monitoring Server.
- In order to monitor the resources in a centralized console, we configured ITCAM for WebSphere - Managing Server, and ITCAM for RTT - Managing Server with the Tivoli Monitoring Server (TEMS/TEP). This provides the capability to create customized workspaces and situations to monitor each resource type.

3.3.2 Management physical architecture

This section includes the physical architecture diagrams and descriptions for the to-be base and enterprise environments. The physical architecture includes the node names, hostname, and product mappings.

Base environment physical architecture

Figure 3-4 depicts the physical architecture of the base environment used to deploy the ITSO Bank application.

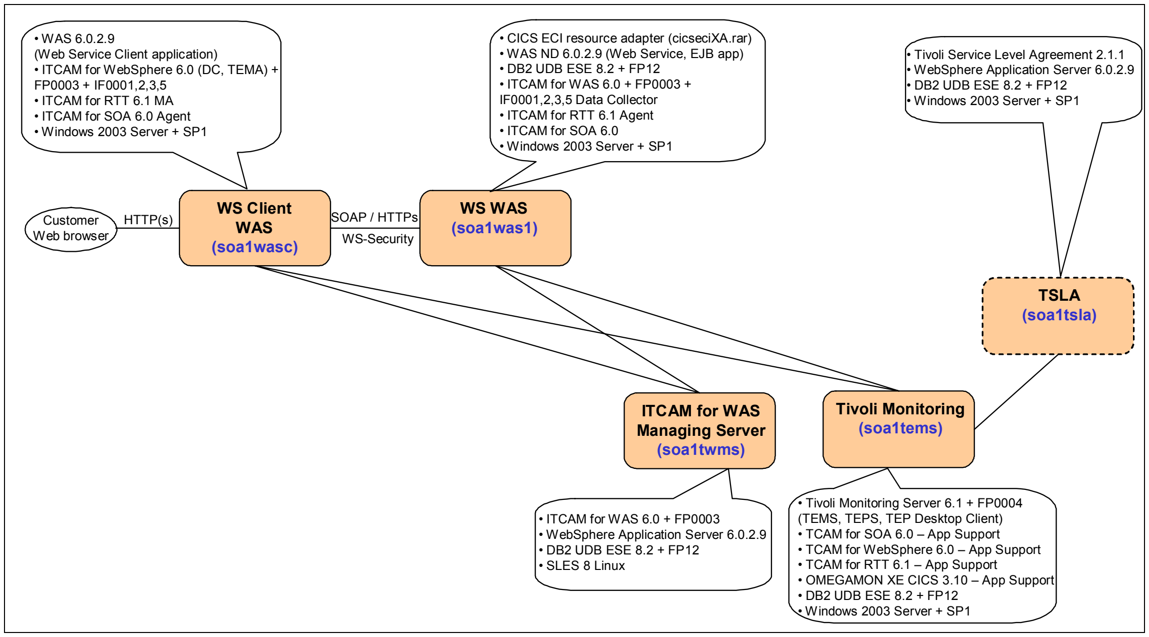
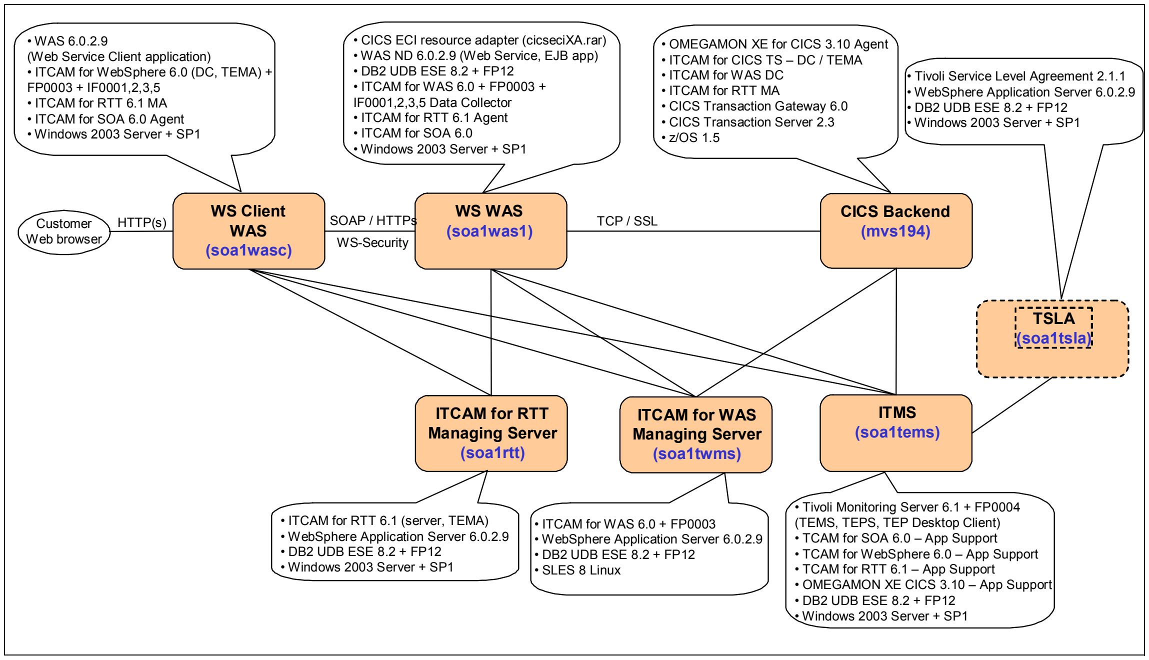

• WAS 6.0.2.9 (Web Service Client application) • ITCAM for WebSphere 6.0 (DC, TEMA) + FP0003 + IF0001,2,3,5 • ITCAM for RTT 6.1 MA • ITCAM for SOA 6.0 Agent • Windows 2003 Server + SP1
• CICS ECI resource adapter (cicseciXA.rar) • WAS ND 6.0.2.9 (Web Service, EJB app) • DB2 UDB ESE 8.2 + FP12 • ITCAM for WAS 6.0 + FP0003 + IF0001,2,3,5 Data Collector • ITCAM for RTT 6.1 Agent • ITCAM for SOA 6.0 • Windows 2003 Server + SP1
• Tivoli Service Level Agreement 2.1.1 • WebSphere Application Server 6.0.2.9 • DB2 UDB ESE 8.2 + FP12 • Windows 2003 Server + SP1
Customer Web browser
HTTP(s)
WS Client WAS (soa1wasc)
SOAP / HTTPs
WS-Security
WS WAS (soa1was1)
TSLA (soa1tsla)
ITCAM for WAS Managing Server (soa1twms)
Tivoli Monitoring (soa1tems)
• ITCAM for WAS 6.0 + FP0003 • WebSphere Application Server 6.0.2.9 • DB2 UDB ESE 8.2 + FP12 • SLES 8 Linux
• Tivoli Monitoring Server 6.1 + FP0004 (TEMS, TEPS, TEP Desktop Client) • TCAM for SOA 6.0 – App Support • TCAM for WebSphere 6.0 – App Support • TCAM for RTT 6.1 – App Support • OMEGAMON XE CICS 3.10 – App Support • DB2 UDB ESE 8.2 + FP12 • Windows 2003 Server + SP1

Figure 3-4 Base environment to-be physical architecture

Note: Refer to "Deploy base environment" on page 110 for more information on the functionality of the nodes and specific versions used for our scenario.

Enterprise environment

Figure 3-5 depicts the physical architecture of the enterprise environment used to deploy the ITSO Bank application.



• WAS 6.0.2.9 (Web Service Client application) • ITCAM for WebSphere 6.0 (DC, TEMA) + FP0003 + IF0001,2,3,5 • ITCAM for RTT 6.1 MA • ITCAM for SOA 6.0 Agent • Windows 2003 Server + SP1
• CICS ECI resource adapter (cicseciXA.rar) • WAS ND 6.0.2.9 (Web Service, EJB app) • DB2 UDB ESE 8.2 + FP12 • ITCAM for WAS 6.0 + FP0003 + IF0001,2,3,5 Data Collector • ITCAM for RTT 6.1 Agent • ITCAM for SOA 6.0 • Windows 2003 Server + SP1
• OMEGAMON XE for CICS 3.10 Agent • ITCAM for CICS TS – DC / TEMA • ITCAM for WAS DC • ITCAM for RTT MA • CICS Transaction Gateway 6.0 • CICS Transaction Server 2.3 • z/OS 1.5
• Tivoli Service Level Agreement 2.1.1 • WebSphere Application Server 6.0.2.9 • DB2 UDB ESE 8.2 + FP12 • Windows 2003 Server + SP1
Customer Web browser
HTTP(s)
WS Client WAS (soa1wasc)
SOAP / HTTPs
WS-Security
WS WAS (soa1was1)
TCP / SSL
CICS Backend (mvs194)
TSLA (soa1tsla)
ITCAM for RTT Managing Server (soa1rtt)
ITCAM for WAS Managing Server (soa1twms)
ITMS (soa1tems)
• ITCAM for RTT 6.1 (server, TEMA) • WebSphere Application Server 6.0.2.9 • DB2 UDB ESE 8.2 + FP12 • Windows 2003 Server + SP1
• ITCAM for WAS 6.0 + FP0003 • WebSphere Application Server 6.0.2.9 • DB2 UDB ESE 8.2 + FP12 • SLES 8 Linux
• Tivoli Monitoring Server 6.1 + FP0004 (TEMS, TEPS, TEP Desktop Client) • TCAM for SOA 6.0 – App Support • TCAM for WebSphere 6.0 – App Support • TCAM for RTT 6.1 – App Support • OMEGAMON XE CICS 3.10 – App Support • DB2 UDB ESE 8.2 + FP12 • Windows 2003 Server + SP1

Figure 3-5 Enterprise environment to-be physical architecture

Note: Refer to “Deploy enterprise environment” on page 113 for more information on the functionality of the nodes and specific versions used for our scenario.

3.4 Design the operational model

The operational model is used to communicate the following operational implementation details from the Management Architect to the IT Administrator to take action in setting up the monitoring environment:

- Define management deployment environment
- Define the TEP users and roles
- Define the TEP workspaces and views used to monitor
- Define the TEP situations to monitor the resources identified
- Define how SLAs will be reported

3.4.1 Define management deployment environment

The Management Architect works with the IT Administrator to communicate the management environment that needs to be deployed (installed and configured). This includes providing a documented definition of the logical and physical environments.

Note: For details on the ITSO Bank example refer to:

- 3.3, “Design the management architecture” on page 49
- Appendix A, “Deploy the environment and sample application” on page 109

3.4.2 Define the TEP users and roles

In the operational environment, there are several roles that will need to have a user created within Tivoli Monitoring - TEP for monitoring and troubleshooting purposes. Table 3-1 provides a summary of the roles, user IDs, and descriptions that must be created within the TEP. The users are assigned roles with specific permissions from the TEP.

Table 3-1 Roles and user IDs for TEP

Role	User ID	Description
IT Administrator	sysadmin	* Create the TEP user IDs and assign roles. * Create situations and workspaces for monitoring.
Operator	operator1	* Monitor events in TEP. * Perform initial troubleshooting of events and if not resolved in 15 minutes, assign to SME for more in-depth problem determination.
Incident Analyst	analyst1	* Perform in-depth problem determination.

Note: For details on how to create users, and assign roles and permissions, refer to the IBM Tivoli Monitoring product InfoCenter.

3.4.3 Define the TEP workspaces and views used to monitor

Workspaces within TEP provide the ability to create a set of customized views of resources being monitored. Some of the views needed are supplied in pre-defined workspaces included with the SOA management products. The existing workspaces can be used as a template or starting point to create new workspaces.

Table 3-2 provides a summary of the TEP workspaces for the ITSO Bank example. In addition, we highlight which environment the workspace can be used in for the example application.

Table 3-2 Summary of workspaces

TEP workspace	Description	Environment
Main	Main workspace and views used to monitor resources for each SOA architectural layer.	* Base * Enterprise
Services	Workspace and views to manage services.	* Base * Enterprise
Middleware	Workspace and views to manage middleware resources.	* Base * Enterprise
Transactional	Workspace and views to manage end-to-end transactional performance.	* Enterprise
Operational	Workspace and views to manage operational resources.	* Enterprise

Note: For details on the ITSO Bank example refer to 4.1, "Create TEP workspaces" on page 60.

3.4.4 Define the TEP situations to monitor the resources identified

A TEP situation is a condition where a set of attributes are tested against a threshold. The situation evaluates the conditions at predefined intervals and invokes the appropriate automated responses and notification methods within the TEP, such as event message or take action. There are many predefined situations, which can be used as-is or as a template to create customized situations.

The Management Architect needs to communicate to the IT Administrator the TEP situations that need to be created to monitor the resources identified. At this stage the data includes the information needed to create the TEP situation, but it is still in English form, not TEP syntax. The IT Administrator will need to have the technical knowledge of how to implement the situations within the TEP.

Table 3-3 provides a summary of the TEP situations that need to be created to monitor the resource types identified in support of the non-functional requirements and SLAs. In addition, we highlight which environment the situation can be used in for the example application.

Table 3-3 TEP situations used to monitor resources

TEP situation description	SLA	Environment
Monitor services using ITCAM for SOA		
Trigger warning event if avg round trip response time > 10 seconds: * DB2: AccountListingSQL service and retrieveAccounts operation. * CICS: getCustInfo3 service and callCustomer3 operation.	ESLA202 ESLA201	* Base * Enterprise
Trigger critical event if message size out of range: * DB2: less than 600 bytes or over 1000 bytes for AccountListingSQL service and retrieveAccounts operation. * CICS: less 900 bytes or over 1500 bytes for the getCustInfo3 service & callCustomer3 operation. **Note:** Actual message size can be seen in the TEP Message Summary view.	ISLA203	* Base * Enterprise
Trigger a critical event service fault if the getCustInfo3 service callCustomer3 operation is not able to connect to CICS TS from CTG.	ISLA104	* Enterprise
Trigger a critical event service fault if the AccountListingSQL service retrieveAccounts operation is accessed more than 5 times within the interval of 1 minute.	ISLA301	* Base
Monitor middleware using ITCAM for WebSphere		
Trigger a warning event if the application server hosting the Web service consumer or provider is stopped. If WAS is stopped, take action to start.	ISLA102	* Base * Enterprise
Trigger an informational event when an exception event is encountered in the WAS logs.	ISLA103	* Base * Enterprise
Trigger a critical event when J2CA message is generated in WAS logs for CTG connection.	ISLA105	* Enterprise
Monitor transactional performance using ITCAM for RTT		
Monitor end-to-end response time of Web service client > Web service > application hosted by CICS.	ESLA201	* Enterprise
Monitor operational systems using OMEGAMON and ITCAM for CICS		
Monitor channel between CTG and CICS is available.	ISLA105	* Enterprise
Monitor CICS application dumps.	ISLA106	* Enterprise

Note: For details on the ITSO Bank example refer to 4.3, “Create TEP situations” on page 78.

3.4.5 Define how SLAs will be reported

To ensure that SLAs are being monitored and results reported, implement Tivoli Service Level Advisor. The SLAs are entered into Tivoli Service Level Advisor and associated with specific TEP situations to monitor the metrics defined in the SLA. Use the Tivoli Service Level Advisor reports to analyze the metrics of the system for the defined SLAs.

Note: For details on the ITSO Bank example refer to 2.3, “Manage SLAs using Tivoli Service Level Advisor” on page 32.

4

Build an integrated TEP console to manage resources

Once you have prepared the environment and deployed the sample application, you are ready to customize the TEP to build an integrated console for managing resources. This chapter demonstrates how to perform discovery of resources, create custom integrated workspaces, and create situations to monitor the resources.

The chapter is organized into the following topics:

- Create TEP workspaces
- Discovery of resources
- Create TEP situations

Note: This chapter requires that you have prepared the environment as described in Appendix A, “Deploy the environment and sample application” on page 109.

4.1 Create TEP workspaces

This section describes how to create custom TEP workspaces with views of resources specific to the SOA management scenario.

Note: When moving from a development to a production environment, it is possible to export the workspaces and import them. In order for the views of the workspaces to be migrated, the development and production TEP must be configured to use the same Tivoli Enterprise Monitoring Server. Refer to the IBM Tivoli Monitoring InfoCenter for more information.

Table 4-1 provides a summary of the TEP workspaces for the redpaper example scenario.

Table 4-1 Summary of workspaces

TEP workspace	Description
Main	Main workspace and views used to monitor resources for each SOA architectural layer.
Services	Workspace and views to manage services.
Transactional	Workspace and views to manage end-to-end transactional performance.
Middleware	Workspace and views to manage middleware resources.
Operational	Workspace and views to manage operational resources.

In our example scenario, we created five different workspaces to monitor resources in each layer of the architecture for our sample application. This section describes the steps required to create each of the workspaces defined in Table 4-1.

The section is organized as follows:

- Create the Services workspace
- Create the Middleware workspace
- Create the Transactional Performance workspace
- Create the Operational Systems workspace
- Create the Main workspace
- Link workspaces and views

4.1.1 Create the Services workspace

The Services workspace is intended to be used by the Operator to monitor services and by the Incident Analyst for problem determination. We have included detailed steps that can be used to create a customized view of the monitoring information in the workspace.

Create the Services Navigator view

Complete the following steps to create the Services navigator view:

1. In the top left of the Navigator window, click the **Edit Navigator View** icon.
2. When the Edit Navigator View window appears, click the **Create New Navigator View** icon within the Target View.
3. When the Create New Navigator View window appears, enter `Services` in the Name field, enter `Customized View for Services` in the Description field, and click **OK**.
4. From the Source View, select **Physical**.
5. Select Windows System nodes (for example, soa1wasc and soa1was1), drag the nodes onto **Services** in the Target View.

   Repeat this process for each desired operating system type of node.
6. Expand soa1wasc and remove every agent except Services Management Agent, since you only want to see services in this workspace view.

   Repeat this step for each node for which you wish to only display services information.
7. Click **Close**.
8. In the Navigator, if you should see a message `KFWITM024I 1 Navigator update pending`, click the green arrow icon to update the Navigator (that is, to make the changes active).
9. At the top of the Navigator, select **Services** from the View drop-down list.
10. Select the **Situation Event Console** icon from the toolbar and drop it in the upper right frame.
11. Select **Universal Message Console** from the toolbar and drop it in the bottom frame.
12. Select **File** → **Save Workspace**.

Customize the Performance Summary view

Complete the following steps to create a customized Performance Summary view:

1. Expand the soa1was1 node.

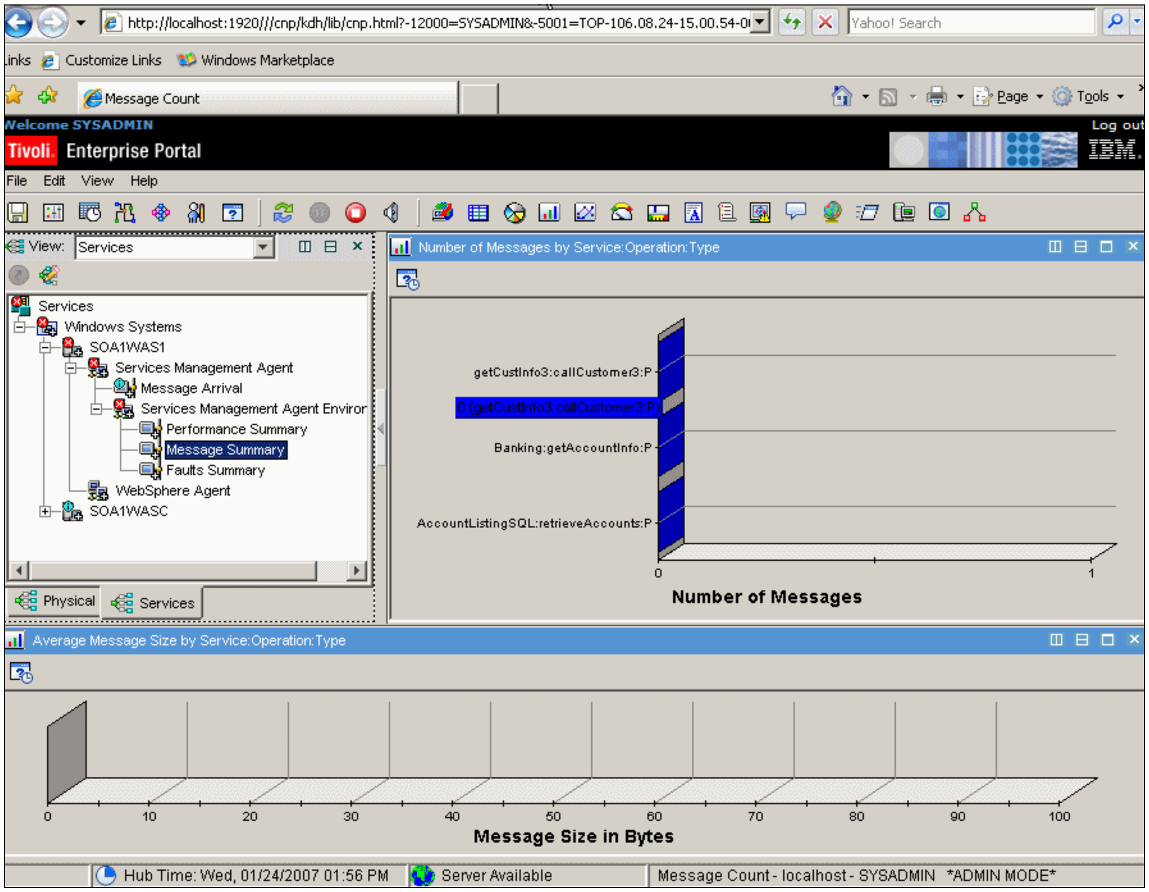
2. Expand **Services Management Agent** → **Services Management Agent Environment**.
3. Select **Performance Summary**.
4. In the Service Inventory view there are four icons. The icons allow you to split the screen either vertically or horizontally. Click **Split Horizontally.** Notice that it has created a new table called Service Inventory:1.
5. Right-click **Properties** from within the table.
6. Click the **Filters** tab.
7. By default many columns will be displayed. Ensure only the following columns are checked (uncheck the others):
   - **Service Type**
   - **Service Name**
   - **Operation Name**
   - **Average Elapsed Message Round Trip Time**
   - **Local Hostname**
8. Click the **Style** tab.
9. Enter `Services Inventory Customized` in the Title text field.
10. Click **OK**.
11. Generate some traffic and then view the customized workspace for the specific application.
12. Select **File** → **Save Workspace As**.
13. Enter `SOA1 Performance Summary` in the Name field, check **Assign as default for this Navigator Item**, and click **OK**.

Customize the Message Summary view

Complete the following steps to create a customized Message Summary view:

1. Expand the soa1was1 node.
2. Expand **Services Management Agent** → **Services Management Agent Environment**.
3. Select **Message Summary**.
4. Select **Average Message Size by Service:Operation:Type** and split it vertically.
5. Select **Number of Messages by Service:Operation:Type** and drag it to the newly created table to swap the tables.
6. Select the **Table** icon from the toolbar and drag it to the table (Average Message Size by Service:Operation:Type:1) to display the data in the table format.

7. Right-click **Properties** from within the table.
8. Select the **Styles** tab.
9. Enter `Average Complete Message Size by Service:Operation:Type` in the Title Text field and click **OK**.
10. Split the new table horizontally.
11. Right-click **Properties** from within the table.
12. Select the **Styles** tab.
13. Enter `Average Incomplete Message Size by Service:Operation:Type` in the Title Text field.
14. Select the **Query** tab.
15. Click **Click here to assign a query**.
16. Select **Incomplete** from the Interval Status column and click **OK**.
17. Click **OK**.
18. In Average Incomplete Message Size by Service:Operation:Type click **Split Horizontally**.
19. Right-click **Properties** from within the table.
20. Click the **Filters** tab.
21. Ensure only the following columns are checked (uncheck others):
    - **Interval Status**
    - **Average Message Length**
    - **Operation Name**
    - **Service Name**
    - **Service Type**
    - **Local Hostname**
22. Click the **Style** tab.
23. Enter `Message Average Customized` in the Title Text field, and click **OK**.
24. Click **OK**.
25. Generate some traffic and then view the customized workspace for the specific application.
26. Select **File** → **Save Workspace As**.
27. Enter `SOA1 Message Summary` in the Name field, check **Assign as default for this Navigator Item**, and click **OK**.



http://localhost:1920///cnp/kdh/lib/cnp.html?-12000=SYSADMIN&-5001=TOP-106.08.24-15.00.54-0
Yahoo! Search
Links Customize Links Windows Marketplace
Message Count
Page Tools
Welcome SYSADMIN
Log out
Tivoli. Enterprise Portal
IBM.
File Edit View Help
View: Services
Services
Windows Systems
SOA1WAS1
Services Management Agent
Message Arrival
Services Management Agent Enviror
Performance Summary
Message Summary
Faults Summary
WebSphere Agent
SOA1WASC
Physical
Services
Number of Messages by Service:Operation:Type
getCustInfo3:callCustomer3:P
0 (getCustInfo3:callCustomer3:P)
Banking:getAccountInfo:P
AccountListingSQL:retrieveAccounts:P
0
1
Number of Messages
Average Message Size by Service:Operation:Type
0
10
20
30
40
50
60
70
80
90
100
Message Size in Bytes
Hub Time: Wed, 01/24/2007 01:56 PM
Server Available
Message Count - localhost - SYSADMIN *ADMIN MODE*

Figure 4-1 Message Summary View

4.1.2 Create the Middleware workspace

The Middleware workspace is intended to be used by the Operator to monitor middleware resources and by the Incident Analyst for problem determination.

1. In the top left of the Navigator window, click the **Edit Navigator View** icon.
2. When the Edit Navigator View window appears, click the **Create New Navigator View** icon within the Target View.
3. When the Create New Navigator View window appears, enter `Middleware` in the Name field, and `Customized View for Middleware` in the Description field, and click **OK**.
4. From the Source View, select **Physical**.

5. Select the desired Windows Systems node (for example, soa1wasc, soa1was1), and drag and drop the nodes onto Middleware in the Target View.

   Repeat this process for each desired operating system or node.

6. Expand soa1wasc and delete the agent types you do not wish to be displayed (for example, Services Management Agent). In this case, we want to keep the WebSphere Agent.

   Repeat this step for each node for which you wish to only display WebSphere Agent information.

7. Click **Close**.
8. At the top of the Navigator, select **Middleware** from the in View drop-down list.
9. Select the **Situation Event Console** icon from the toolbar and drop it into the upper right frame. When prompted to save, click **Yes**.
10. Select **Universal Message Console** from the toolbar and drop it in the bottom frame.
11. Select **File** → **Save Workspace**.

4.1.3 Create the Transactional Performance workspace

The Transactional Performance workspace is intended to be used by the Operator to monitor the end-to-end transactional performance and by the Incident Analyst for problem determination.

1. In the top left of the Navigator window, click the **Edit Navigator View** icon.
2. When the Edit Navigator View window appears, click the **Create New Navigator View** icon within the Target View.
3. When the Create New Navigator View window appears, enter `Transactional Performance` in the Name field, and `Customized View for Transactional Performance` in the Description field, and click **OK**.
4. From the Source View, select **Physical**.
5. Select the ITCAM for RTT - Managing Server node (soa1rtt) under Windows Systems node, and drag and drop the node onto Transaction Performance in the Target View.
6. Click **Close**.
7. At the top of the Navigator, select **Transactional Performance** from the in View drop-down list.
8. Select the **Situation Event Console** icon from the toolbar and drop it in the upper right frame. When prompted to save, click **Yes**.

9. Select **Universal Message Console** from the toolbar and drop it in the bottom frame.
10. Select **File** → **Save Workspace**.

4.1.4 Create the Operational Systems workspace

The Operational Systems workspace is intended to be used by the Operator to monitor the end-to-end Operational Systems (CICS Transaction Server, CICS Transaction Gateway) and by the Incident Analyst for problem determination.

1. In the top left of the Navigator window, click the **Edit Navigator View** icon.
2. When the Edit Navigator View window appears, click the **Create New Navigator View** icon within the Target View.
3. When the Create New Navigator View window appears, enter `Operational Systems` in the Name field, and `Customized View for Operational System` in the Description field, and click **OK**.
4. From the Source View, select **Physical**.
5. Select the OMEGAMON XE for CICS node (mvs194) under z/OS Systems, and drag and drop the node onto Operational Systems in the Target View.
6. Click **Close**.
7. At the top of the Navigator, select **Operational Systems** from the in View drop-down list.
8. Select the **Situation Event Console** icon from the toolbar and drop it in the upper right frame. When prompted to save, click **Yes**.
9. Select **Universal Message Console** from the toolbar and drop it in the bottom frame.
10. Select **File** → **Save Workspace**.

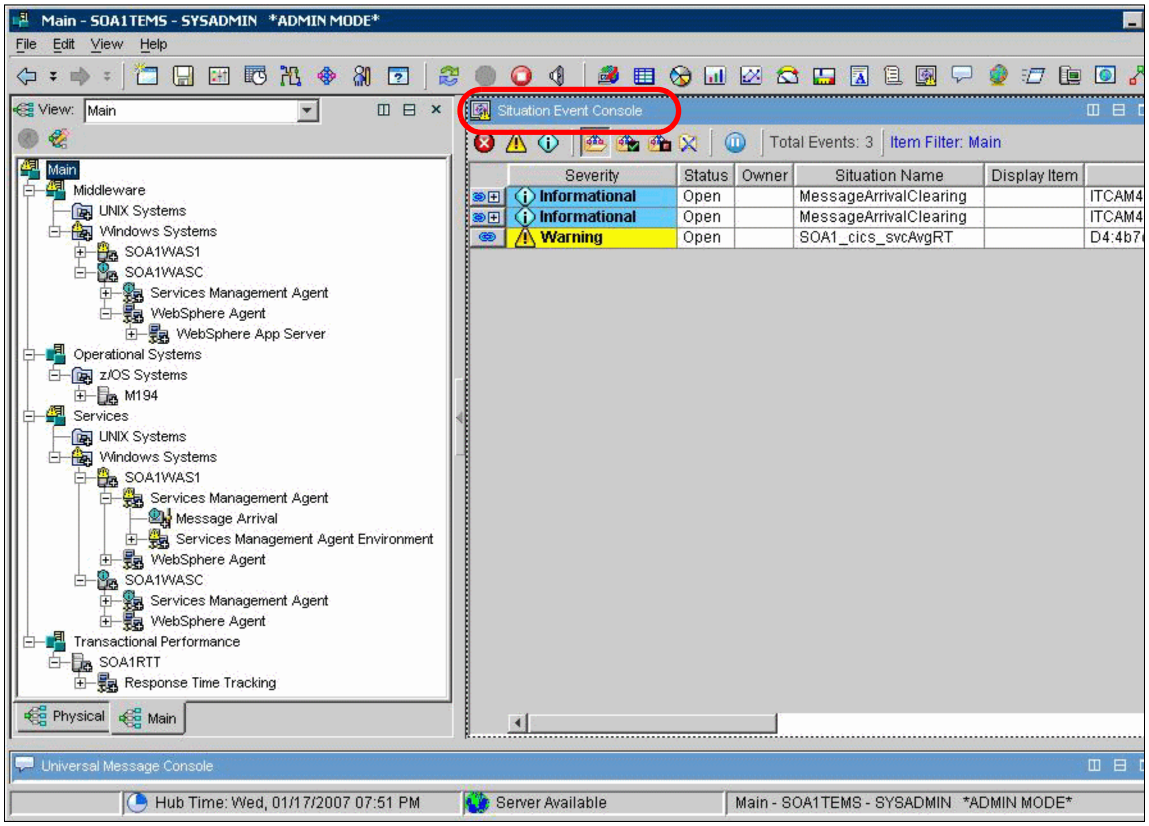
4.1.5 Create the Main workspace

The Main workspace is intended to be used by the Operator to monitor the end-to-end transaction and supporting infrastructure, and by the Incident Analyst for problem determination. The Main workspace provides a consolidated view of all resource types and will be used as the primary view the Operator uses to monitor resources via events driven from the TEP situations.

1. In the top left of the Navigator window, click the **Edit Navigator View** icon.
2. When the Edit Navigator View window appears, click the **Create New Navigator View** icon within the Target View.

3. When the Create New Navigator View window appears, enter `Main` in the Name field, and `Customized View for Main` in the Description field, and click **OK**.
4. Add Service to the Main workspace.
    a. From the Source View, select the **Services** view.
    b. Select **Services** from the right pane, and drag and drop it onto Main in the Target View.
5. Add Middleware to the Main workspace.
    a. From the Source View, select the **Middleware** view.
    b. Select **Middleware** from the right pane, and drag and drop it onto Main in the Target View.
6. Add Transactional Performance to the Main workspace.
    a. From the Source View, select the **Transactional Performance** view.
    b. Select **Transactional Performance** from the right pane, and drag and drop it onto Main in the Target View.
7. Add Operation Systems to the Main workspace.
    a. From the Source View, select the **Operational Systems** view.
    b. Select **Operational Systems** from the right pane, and drag and drop it onto Main in the Target View.
8. Click **Close**.
9. At the top of the Navigator, select **Main** from the in View drop-down list.
10. Select **Main** from the top-level tree.
11. Select the **Situation Event Console** icon from the toolbar and drop it in the upper right frame. When prompted to save, click **Yes**.
12. Select **Universal Message Console** from the toolbar and drop it in the bottom frame.
13. Select **File** → **Save Workspace**.

When complete, the Main workspace will look like Figure 4-2, and will provide an integrated view of the resources being managed. From the integrated Main workspace, you can monitor resources for Middleware, Operational Systems, Services, and Transactional Performance. When situations are triggered, the events will be displayed in the Situation Event Console.


Main - SOA1TEMS - SYSADMIN *ADMIN MODE*

File Edit View Help

View: Main

- Main
  - Middleware
    - UNIX Systems
    - Windows Systems
      - SOA1WAS1
      - SOA1WASC
        - Services Management Agent
        - WebSphere Agent
          - WebSphere App Server
  - Operational Systems
    - z/OS Systems
      - M194
  - Services
    - UNIX Systems
    - Windows Systems
      - SOA1WAS1
        - Services Management Agent
          - Message Arrival
          - Services Management Agent Environment
        - WebSphere Agent
      - SOA1WASC
        - Services Management Agent
        - WebSphere Agent
  - Transactional Performance
    - SOA1RTT
      - Response Time Tracking

Physical Main

Situation Event Console

Total Events: 3 Item Filter: Main

Severity	Status	Owner	Situation Name	Display Item	
Informational	Open		MessageArrivalClearing		ITCAM4
Informational	Open		MessageArrivalClearing		ITCAM4
Warning	Open		SOA1_cics_svcAvgRT		D4:4b7

Universal Message Console

Hub Time: Wed, 01/17/2007 07:51 PM Server Available Main - SOA1TEMS - SYSADMIN *ADMIN MODE*

Figure 4-2 Main workspace: Integrated view

4.1.6 Link workspaces and views

Workspaces and views can be linked to one another. This is useful when you want to drill down from a higher level workspace to another workspace that provides more details. We outline two examples of linking TEP resources for our example environment.

Link service response time situation to WebSphere log

This example applies to both the base and enterprise environments.

Consider an event where the Web service response time exceeds a monitored situation. In this example, the root cause is due to the WebSphere Application Server - server1 application server being stopped that hosts the Web service. We demonstrate how we can link the workspaces to drill down from the service response time event in the Situations Event Console to the WebSphere Application Server - Log Analysis view situation.

Complete the following steps to link the SOA1_db2_svcAvgRTT situation to the ITCAM for WebSphere - Log Analysis.

1. Run the application and trigger the situation.

   Refer to 5.1.1, “Verify the SOA1_db2_svcAvgRT situation” on page 96 for details on how to run the ITSO Bank to trigger the situation.

2. Log on to the TEP.
3. From the Situations Event Console, right-click the **SOA1_db2_svcAvgRTT** warning event, and select **Link Wizard**.
4. When the Workspace Link Wizard dialog opens, select **Create a new link** and click **Next**.
5. Enter `WASLog` in the Name field, `WAS log analysis link` in the Description field, and click **Next**.
6. When the Type of Link dialog opens, select **Dynamic** and click **Next**.
7. From the Navigator, drill down to the desired workspace view. In our example, we selected **Log Analysis** under WebSphere App Server for soa1was1 (node where service is deployed). Select **Log Analysis** from the Workspace, and then click **Next**.
8. When the Assign Expressions dialog appears, accept the defaults and click **Next**.
9. When the Summary dialog appears, review the selections and click **Finish**.
10. To verify the Link from the service response time situation (ITCAM for SOA) to the WebSphere Log Analysis (ITCAM for WebSphere), do the following:
    a. To verify, stop the Web Service - WebSphere Application Server node (soa1was1) server1 application server.
    b. Run the application. An error will be displayed in the browser when the service attempts to retrieve data from the database via the EJB.
    c. The TEP SOA1_db2_svcAvgRTT situation will be triggered in the Situation Event Console.
    d. The Operator can right-click the event and select the link (for example, WASLog) and drill down to the linked workspace view. In this case, the WebSphere Application Server - server1 application server was stopped, thus causing the service response time issue.

Web Service to CICS response time slow

This example applies to the enterprise environment.

Consider an event where the Web service that connects to CICS via CTG has a response time is too slow and a situation is triggered. More detailed information may be required to get to the root problem. The situation can be linked to a specific workspace or view that would provide the Operator or Incident Analyst with more detailed information to troubleshoot the event.

To link the SOA1_cics_svcAvgRT situation to Reporting Groups in the Navigator, complete the following steps:

1. Run the application and trigger the situation to monitor end-to-end response time.

   Refer to 5.2.5, “Verify the SOA1_e2eRT situation” on page 106 for details on how to run the ITSO Bank to trigger the situation. This example has a 60 second delay to force the monitoring situation to be triggered.
2. Log on to the TEP.
3. From the Situations Event Console, right-click the **SOA1_e2eRT** warning event, and select **Link Wizard**.
4. When the Workspace Link Wizard dialog opens, select **Create a new link** and click **Next**.
5. Enter `RTTlink` in the Name field, `end-2-end response time` in the Description field, and click **Next**.
6. When the Type of Link dialog opens, select **Absolute** and click **Next**.
7. From the Navigator, drill down to the desired workspace view. In our example, we selected **Applications** (type of report used), **Response Time Tracking Report Group Application**, and then clicked **Next**.
8. When the Assign Expressions dialog appears, accept the defaults and click **Next**.
9. When the Summary dialog appears, review the selections and click **Finish**.
10. When the situation that has been linked is triggered, the user can right-click the event, select the link (for example, RTTlink), and drill down to the linked workspace view.

4.1.7 Importing and exporting workspaces

IBM Tivoli Monitoring V6.1 - Fixpack 00004 provides the capability to export and import workspaces.

For more detailed information on importing and exporting workspace, refer to the Tivoli Monitoring V6.1 - Fixpack 0004 documentation.

4.2 Discovery of resources

In “Approaches for identifying resources to manage” on page 44 we described the discovery of resources in a development test environment and in a production environment. In this section, we demonstrate how to use the discovery features of the ITCAM for SOA and ITCAM for RTT products at runtime.

We have included the following two discovery examples:

- Discovery of service using ITCAM for SOA
- Discovery of ARM Transactions using ITCAM for RTT

4.2.1 Discovery of service using ITCAM for SOA

For the purposes of illustrating how to perform discovery of a service, suppose that we do not know the name of the services we want to manage. In this example, we run the ITSO Bank application with the ITCAM for SOA agent started and listen for service requests. We can then look at the service inventory populated by ITCAM for SOA within the TEP workspace. The services inventory captures the service name and operation name. This information will later be used to create a TEP situation to monitor the service average round trip response time.

The high-level tasks are as follows:

- Launch the Tivoli Enterprise Portal Desktop Client.
- Ensure the Services Management Agent is started.
- Generate traffic by running the Web services application.
- Review the services inventory.

Launch the Tivoli Enterprise Portal Desktop Client

To launch the TEP Desktop Client, perform the following steps on the Tivoli Monitor Server node:

1. Click **Start** → **Programs** → **IBM Tivoli Monitoring** → **Tivoli Enterprise Portal**.

   > **Note:** Alternatively, start TEP by double-clicking the **Tivoli Enterprise Portal** icon added to the desktop during the installation. You can also use the TEP browser-based client.

2. When the Logon dialog box opens, enter the following and click **OK**:
   – Logon ID: `sysadmin`

Note: The default Logon ID is sysadmin.

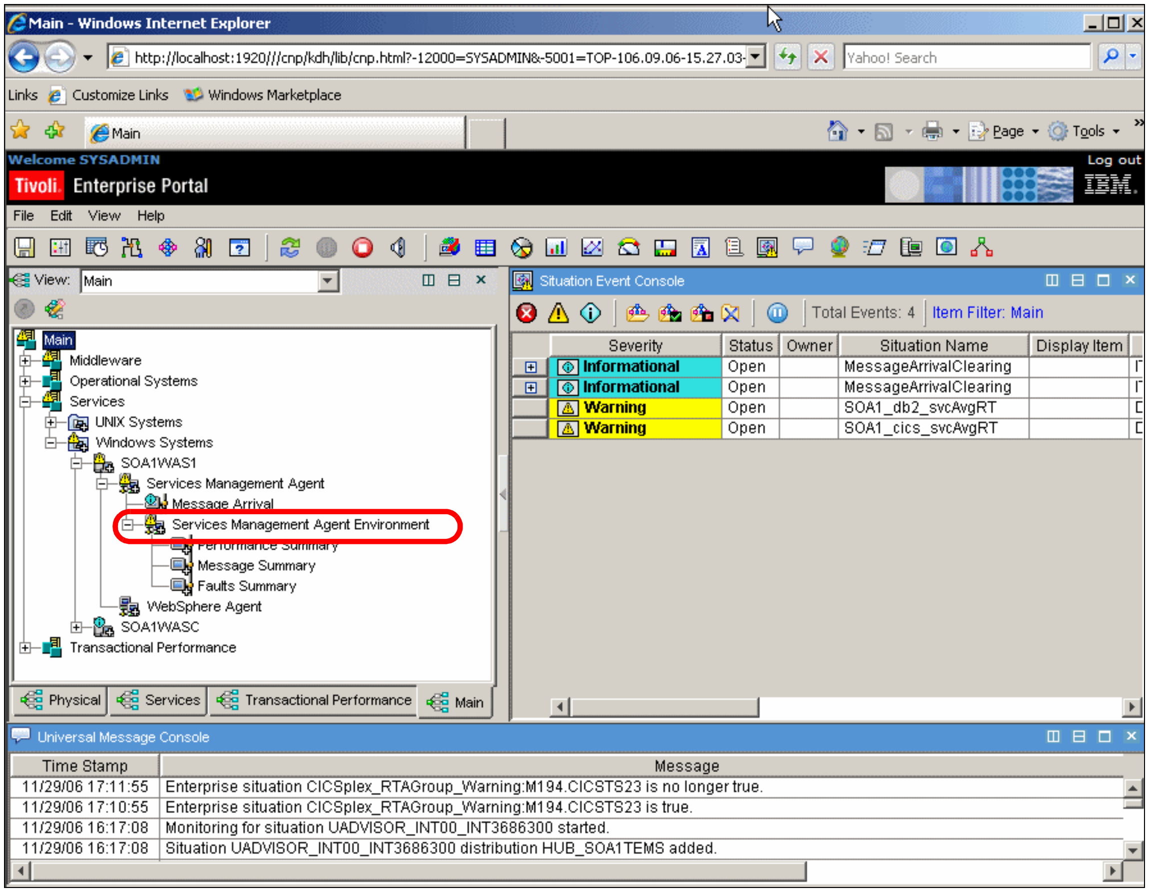
– Password: <*password*>

3. If you see a Security Alert dialog box, click **Accept**.

Ensure the Services Management Agent is started

Do the following to ensure the ITCAM for SOA - Service Management Agent is started:

After you have logged in to the Tivoli Enterprise Portal Desktop, expand **Enterprise** → **Windows Systems** → **soa1was1** → **Services Management Agent** (see Figure 4-3), where soa1was1 is hosting Web service application.



Severity	Status	Owner	Situation Name	Display Item
Informational	Open		MessageArrivalClearing	
Informational	Open		MessageArrivalClearing	
Warning	Open		SOA1_db2_svcAvgRT	
Warning	Open		SOA1_cics_svcAvgRT	

Time Stamp	Message
11/29/06 17:11:55	Enterprise situation CICSplex_RTAGroup_Warning:M194.CICSTS23 is no longer true.
11/29/06 17:10:55	Enterprise situation CICSplex_RTAGroup_Warning:M194.CICSTS23 is true.
11/29/06 16:17:08	Monitoring for situation UADVISOR_INT00_INT3686300 started.
11/29/06 16:17:08	Situation UADVISOR_INT00_INT3686300 distribution HUB_SOA1TEMS added.

Figure 4-3 Tivoli Enterprise Portal - Services Management Agent

Note: Troubleshooting tip: If you do not see the Services Management Agent displayed in the workspace as shown in Figure 4-3, the ITCAM for SOA agent may not be started or may be mis-configured.

The Services Management Agent Environment will be greyed out until the Web service has been invoked on the remote application server.

Generate traffic by running the Web services application

To generate Web services traffic, you must invoke Web services in the application. In our example, we used the ITSO Bank application that retrieves data from a DB2 database via a Web service.

Note: For more detailed information on using the ITSO Bank application, refer to Chapter 5, "Monitor resources from the integrated TEP console" on page 95.

1. Enter the following URL in a Web browser to launch the ITSO Bank application:

   ```
   http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
   ```

2. Select **DB2** under Select Source of Data, and click **Submit**.
3. When the ITSO Bank Welcome page appears, select **1** from the CustomerID drop-down list, and click **Submit**.

   This will invoke the service to retrieve the customer account information from the DB2 UDB application database.

Review the services inventory

After the Web services traffic is generated, it can be reviewed within the TEP in the services inventory.

1. Now that the Web services traffic has been generated, click the **Refresh Now (F5)** button found in the toolbar of the TEP.
2. Select **Enterprise** in the Navigator.

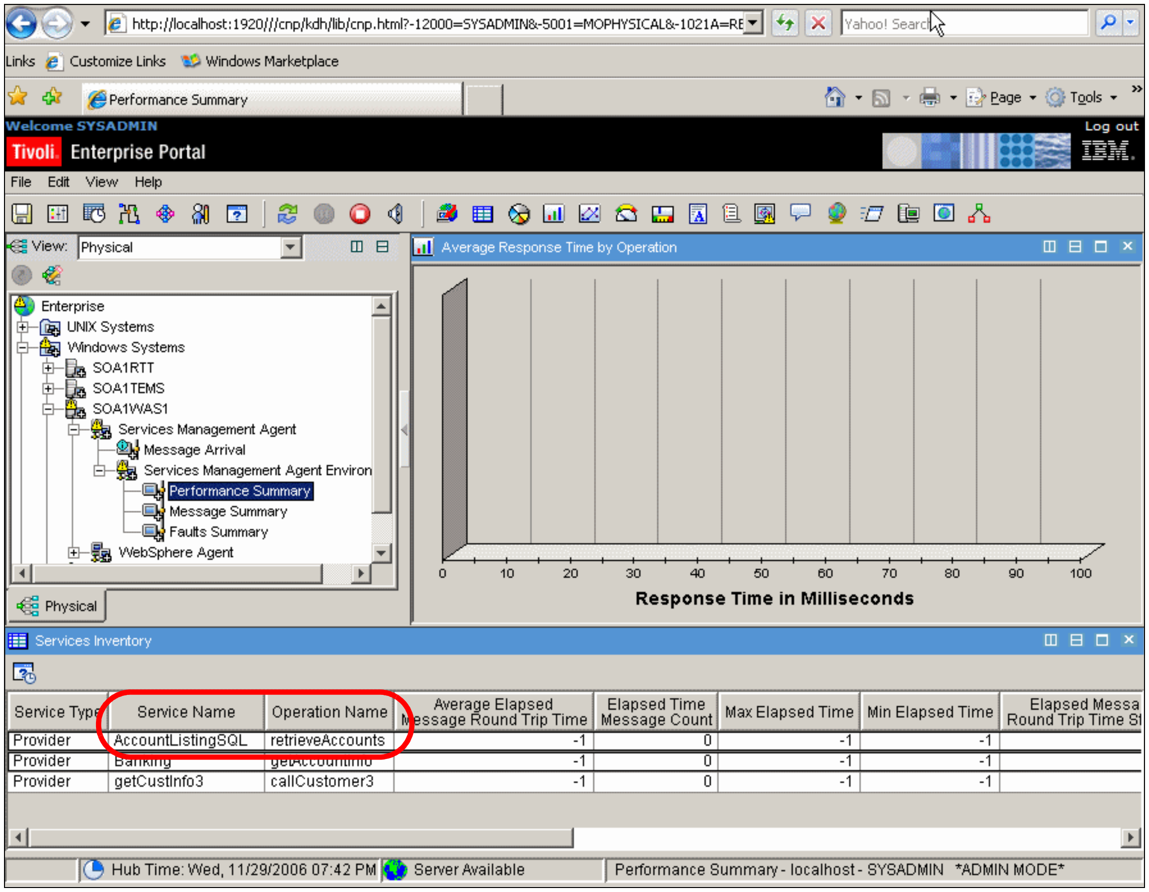
   Notice in the Situation Event Console, there are color-coded icons to Filter Critical, Filter Warning, and Filter Information. When a particular situation becomes true, then a color-code icon is displayed.

Note: The predefined Situations for `MessageArrivalCritical` and `MessageClearing` will become true depending on how much and when the Web services traffic is generated.

If you run the test application several times in succession to generate enough messages that it reaches the defined threshold, the `MessageArrivalCritical` situation will become True. Similarly when the number of messages drops below the threshold, the `MessageArrivalClearing` situation will become True.

3. In TEP, the icon with an arrow in the top left corner above the **Navigator** is green. Click it to update the workspace.
4. Expand **Services Management Agent Environment** and click **Performance Summary**.

   As shown in Figure 4-4 on page 75, the Services_Inventory view displays the service name AccountListingSQL, and operation name retrieveAccounts. The invocation of the Web service (provider) has been discovered by ITCAM for SOA. The service name and operation name are needed when creating a TEP situation to monitor a service.



Service Type	Service Name	Operation Name	Average Elapsed Message Round Trip Time	Elapsed Time Message Count	Max Elapsed Time	Min Elapsed Time	Elapsed Message Round Trip Time St
Provider	AccountListingSQL	retrieveAccounts	-1	0	-1	-1	
Provider	Banking	getAccountInfo	-1	0	-1	-1	
Provider	getCustInfo3	callCustomer3	-1	0	-1	-1	

Hub Time: Wed, 11/29/2006 07:42 PM Server Available Performance Summary - localhost - SYSADMIN *ADMIN MODE*

Figure 4-4 Services_Inventory: Discover service name and operation name

4.2.2 Discovery of ARM Transactions using ITCAM for RTT

In our enterprise example environment, the end-to-end transaction flows from the Web services client application to a Web service that then accesses CICS. ITCAM for RTT Management Agents are required on each node of the transaction. A listening monitor is used as the entry point of the transaction. The end-to-end transaction is automatically correlated by ITCAM for RTT to calculate the end-to-end response time.

Note: In our example, we used the ARM feature to facilitate the correlation. Refer to Appendix B, “Using the ITCAM for RTT ARM feature” on page 131 for implementation details.

In the following steps, we describe how to use the ITCAM for RTT - Managing Server Console to discover the ARM and then create a listening monitor, which is used to correlate and monitor the end-to-end response time.

1. Prerequisites for ITCAM for RTT discovery are as follows:
   - ITCAM for RTT V6.1 - Managing Server installed
   - ITCAM for RTT V6.1 - Management Agent installed on each node of the transaction:
     - soa1wasc - Web Services client application
     - soa1was1 - Web Service (wraps session EJB / CICS ECI resource adapter)
     - mvs194 - CTG / CICS TS
   - Agent Group in ITCAM for RTT has been created (for example, winagents)
   - Schedule created (for example, 24x7)
   - Reporting Group created (for example, report1)
   - Management Agent is online
   - J2EE Monitoring Component deployed to the Agent
2. Perform Discovery setup.
   a. Click **Configuration** → **Discovery** from the ITCAM for RTT - Managing Server Console.

   b. When the Configuration window appears, expand the drop-down list and click **Create New**. **Arm Application**.

   c. On the ARM Settings tab, accept defaults for all settings (Match all - view details after creating), check **Enable additional data uploads every 5 minutes** (for test purposes - default upload is 1 hour). Click **Next**.

   d. On the Schedules tab, select **24x7** and click **Next**.

   e. On the Agents Group tab, select **winagents** and click **Next**.

   f. On the Reporting Groups tab, check **report1** and click **Next**.

   g. On the Name tab, enter **arm_discovery1** and click **Finish**.

   Note: Only have one discovery filter of the same pattern (for example, .*) enabled at a time; otherwise, this will cause the discovery to fail.

3. Generate traffic with the target application so that the management agent can discover application resources.
4. Create the listening monitor from the discovered Arm Application.

a. Click **Configuration** → **Discovery**.

b. Check **arm_discovery1**, select **View Discovered Transactions** from the drop-down list, and click **Go**.

c. Click the **Database** icon to retrieve discovered transactions from the management agent.

d. When prompted with the “Retrieval may take several minutes” dialog, click **OK**.

e. A dialog will display the progress of the retrieval. At completion, it should say `Completed successfully`. Expand **Details**. It should say `No failure`. If it say `No data available`, then it did not capture data from the MA.

f. Click **Update** and close. You can also click the Refresh icon to show the servlets retrieved.

g. Select the desired ARM Application (ITSOBankClientWeb), select **Create Listening Monitor** from the drop-down list, and click **Go**.

h. On the ARM Settings tab. Under Data Collection select **Collect All Instance Records** (or **Collect Instance records after failure**) and check **Enable additional data uploads every 5 minutes** (test purposes). Set Sample Rate to **100** (measured in %; the default is 5). Click **Next**.

i. Under ARM Thresholds, select **Performance from the Threshold Type** drop-down list and click **Create**. Set the desired values. The Threshold by default will be set to value when discovered. Set accordingly, click **Apply**, and then click **Next**.

j. Click **Schedule**. Select **24x7** and click **Next**.

k. Under Agent Group, select **winagents** and click **Next**.

l. Under Reporting Groups, check **report1** and click **Next**.

m. Under Name, enter `armmonitor1` and click **Finish**.

n. Generate traffic by running the application.

Note: At the time of writing, we found that if more than one listening monitor has the same monitoring filter, the data capture does not work properly.

Once the listening monitor is configured and the application is run, the end-to-end response time reporting data can be accessed from the ITCAM for RTT - Managing Server Console. In addition, if ITCAM for RTT has been configured with Tivoli Monitoring Server, the reporting data will also be displayed in the TEP.

4.3 Create TEP situations

In this section we first outline the critical information for each of the situations we created to monitor the resources of our scenario. Next we explain how to import the custom TEP situations we created for our scenario. In addition, we provide an example of how to create a custom situation in the TEP.

Table 4-4 provides a summary of the TEP situations created for the redpaper example.

Sample code: Refer to Appendix C, "Additional material" on page 139 for details on downloading the 4233code.zip containing the sample application and TEP situations.

The sample TEP situations can be found in the C:\4233code\tep directory of the unpacked 4233code.zip.

Table 4-2 TEP situations used to monitor resources

TEP situation description	TEP situation filename	Environment
Monitor services using ITCAM for SOA		
Trigger warning event if avg round trip response time > 10 sec: * DB2: AccountListingSQL service and retrieveAccounts operation. * CICS: getCustInfo3 service and callCustomer3 operation.	SOA1_db2_svcAvgRT.xml SOA1_cics_svcAvgRT.xml	* Base * Enterprise
Trigger critical event if message size out of range: * DB2: less 600 bytes or over 1000 bytes for AccountListingSQL service and retrieveAccounts operation * CICS: less 900 bytes or over 1500 bytes for the getCustInfo3 service & callCustomer3 operation **Note:** Actual message size can be seen in the TEP Message Summary view.	SOA1_db2_svcMsgSize.xml SOA1_cics_svcMsgSize.xml	* Base * Enterprise
Trigger a critical event service fault if the getCustInfo3 service callCustomer3 operation is not able to connect to CICS TS from CTG.	SOA1_cics_svcFault.xml	* Enterprise
Trigger a critical event service fault if the AccountListingSQL service retrieveAccounts operation is accessed more than 5 times within the interval of 1 minute.	SOA1_db2_svcUsage.xml	* Base

	TEP situation description	TEP situation filename	Environment
Monitor middleware using ITCAM for WebSphere			
	Trigger a warning event if the application server hosting the Web service consumer or provider is stopped. If WAS stopped, take action to start.	SOA1_wasAvailability.xml SOA1_wasStart.xml	* Base * Enterprise
	Trigger an informational event when an exception event is encountered in the WAS logs.	SOA1_wasLogException.xml	* Base * Enterprise
	Trigger an critical event when J2CA message is generated in WAS logs for CTG connection.	SOA1_ctgJ2CConnection.xml	* Enterprise
Monitor transactional performance using ITCAM for RTT			
	Monitor end-to-end response time of Web service client > Web service > application hosted by CICS.	SOA1_e2eRT.xml	* Enterprise
Monitor operational systems using OMEGAMON and ITCAM for CICS			
	Monitor channel between CTG and CICS is available.	SOA1_ctg2CICSChannel	* Enterprise
	Monitor CICS application dumps.	SOA1_cicsAppDump.xml	* Enterprise

4.3.1 Configure historical collection

In our example, we enabled historical collection for the specific monitoring product (for example, ITCAM for SOA, ITCAM for WebSphere, ITCAM for RTT) integrated with the Tivoli Monitoring Server.

Note: As a prerequisite, the Tivoli Monitoring Warehouse Proxy must be configured. Refer to the *Tivoli Monitoring Server InfoCenter* for details.

In this example, we modify the History Collection Configuration to change the collection interval to 5 minutes (from the default of 15) for the services being monitored by ITCAM for SOA.

1. Log on to the TEP as sysadmin.
2. Select **Edit** → **History Configuration**.
3. When the History Collection Configuration window opens, follow these steps:
   a. Select **ITCAM for SOA** from the Select a product drop-down list.
   b. Click **Services_Inventory** under Select Attribute Groups.
   c. Select **5 minutes** (default is 15 minutes) for the Collection Interval.
   d. Select **TEMA** from the Collection Location drop-down list.

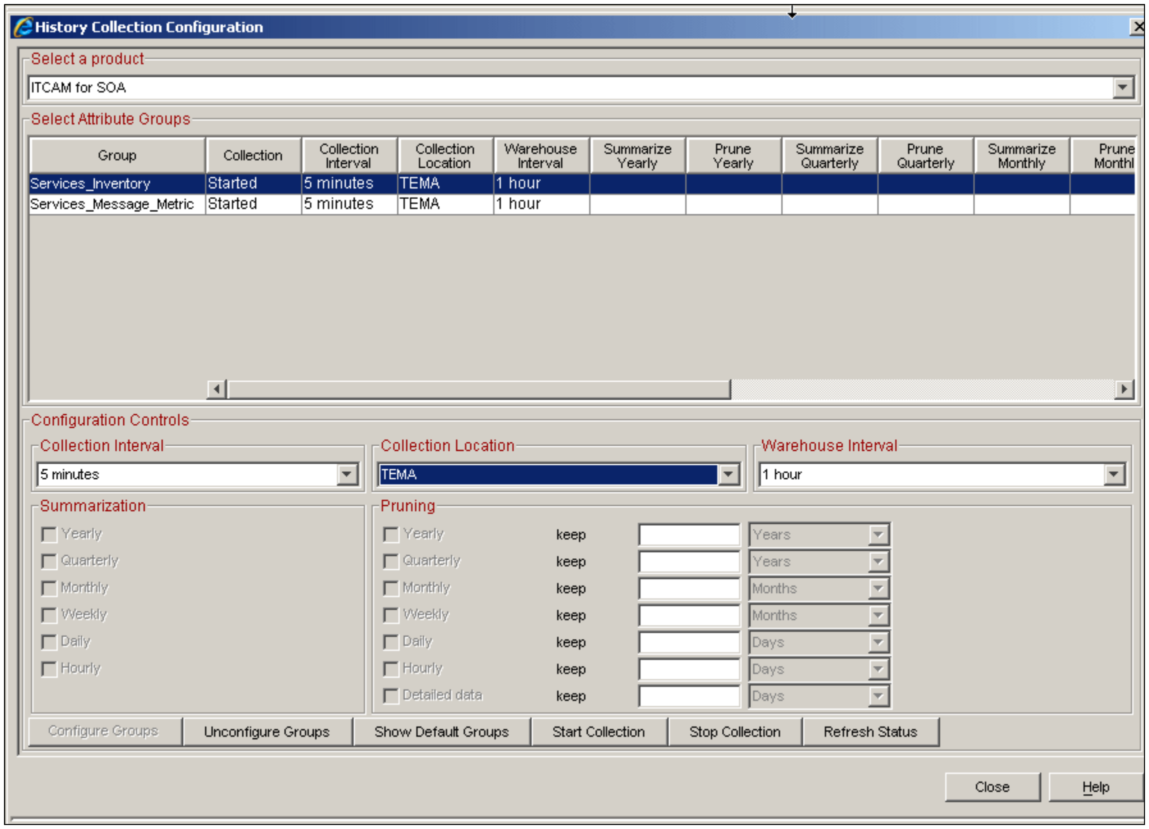
Note: When using TEMA, minimize the performance on the Tivoli Monitoring Server by performing the data collection on the agent node.

e. Select **1 hour** for Warehouse interval.

f. Click **Configure Groups**.

g. Click **Services_Inventory** from the Select Attribute Groups.

h. Click **Start Collection** (Figure 4-5).



Select a product: ITCAM for SOA

Select Attribute Groups

Group	Collection	Collection Interval	Collection Location	Warehouse Interval	Summarize Yearly	Prune Yearly	Summarize Quarterly	Prune Quarterly	Summarize Monthly	Prune Monthl
Services_Inventory	Started	5 minutes	TEMA	1 hour						
Services_Message_Metric	Started	5 minutes	TEMA	1 hour						

Collection Interval: 5 minutes

Collection Location: TEMA

Warehouse Interval: 1 hour

Figure 4-5 History Collection Configuration: Set collection interval

i. Repeat this process for Services_Message_Metric.

j. Click **Close**.

This sets the collection interval time to five minutes for real-time data and one hour for historical data stored in the Warehouse database.

Note: In the high volume production environment, it is recommended to avoid collecting data for the Service_Message_Metric group.

Note: You will need to configure historical collection for each product you have integrated with Tivoli Monitoring (for example, ITCAM for SOA, ITCAM for WebSphere, ITCAM for RTT).

4.3.2 Import situations

This section describes how to import TEP situations from a file. As noted, Table 4-2 on page 78 lists the file names of the situations we created for the redpaper example.

Note: Chapter 5, “Monitor resources from the integrated TEP console” on page 95 describes how to monitor resources and verify each of the situations provided for import.

Import situation

The following example describes how to import the db2_svcAvgRT.xml situation, used to monitor the service average round trip response time.

1. Open a Windows command window.
2. Enter the following to log in to TEP:

   ```
   tacmd login -s soa1tems -u sysadmin -p password
   ```

   Syntax:

   ```
   tacmd login -s <tms_hostname> -u sysadmin -p <password>
   ```

3. Navigate to the `c:\4233code\tep\situations` directory, and enter the following command to create the situation from a file:

   ```
   tacmd createSit -i SOA1_db2_svcAvgRT.xml
   ```

Note: To automate the import of all TEP situations for the redpaper example, do the following:

1. Open a Windows command window.
2. Enter the following to log in to TEP:

   ```
   tacmd login -s soa1tems -u sysadmin -p password
   ```

3. Navigate to the `c:\4233code\tep\situations` directory, and enter the following command:

   ```
   4233_createSituations.bat
   ```

Tip: When moving from a development test environment to production, it is very useful to export the situations to a file that can then be imported.

To export a situation to a file, where SOA1_db2_svcAvgRT is the situation name, enter the following command:

```
tacmd viewSit -s SOA1_db2_svcAvgRT -e SOA1_db2_svcAvgRT.xml
```

To get a listing of situations, enter the following command:

```
tacmd listSit
```

Note: After exporting the situation to a file, you need to account for the hostname of the nodes from the export and import environments. After exporting the situation to a file, do the following:

1. Open the situation XML file in an editor.
2. Search for the <Distribution> tag.
3. Remove the hostname entries under the <Distribution> tag.
4. You can now use this version of the situation file to import.
5. The procedure found in the following section describes how to complete the process after importing the situation, in order to associate (distribute) the situation with the node.

Post import configuration

After importing the situation, you need to associate the situation with the nodes you wish to monitor.

1. After creating the situation, you must distribute the situation so that it can run on the target node. To determine the target node, view the system or systemlist using the appropriate command. Refer to Table 4-3 on page 84 to determine the proper command for the given situation.

2. We have included an example of listing the target nodes and used the `SOA1_db2_svcAvgRT.xml` in our example procedure.

   ```
   tacmd listSystems -t D4
   ```

   The output looked like the following:

   ```
   Managed System Name              Product Code Version     Status
   D4:4b7c211a:soa1was1-server1     D4           01.01.00.XX Y
   D4:f6d49bbd:soa1wasc-server1     D4           01.01.00.XX Y
   ITCAM4SOA:soa1wasc.rtp.raleigh.i D4           01.01.00.XX Y
   ITCAM4SOA:soa1was1.rtp.raleigh.i D4           01.01.00.XX Y
   ```

   This command displays the list of managed systems for the product code `D4`, which corresponds to ITCAM for SOA. In our case, the entry that we need to select is: `D4:4b7c211a:soa1was1-server1`.

3. Edit the situation to update the distribution property with this node information using the command:

   ```
   tacmd editSit -s SOA1_db2_svcAvgRT -p
   Distribution=D4:4b7c211a:soa1was1-server1

   Are you sure you want to update the SOA1_db2_svcAvgRT situation? (Y
   - yes or N - no:): y

   KUICES005I: The situation SOA1_db2_svcAvgRT was updated on server
   https://soa1tems:1117
   ```

   Note: Refer to Table 4-3 on page 84 to verify the value of the distribution attribute for the other situations listed in the table.

4. After the situation is updated, associate the situation with the required Navigator Item so that the situation is displayed in the Situation Event Console. This particular situation example needs to be associated with: Services Management Agent Environment. From the TEPS console, perform the following steps:

   a. Select **Services Management Agent Environment** for the `SOA1_db2_svcAvgRT` example situation.

      Note: Refer to Table 4-3 on page 84 to identify the exact Navigator Item for the other situations listed in the table.

   b. Right-click to select **Situations**.

   c. Click on the left icon **Set situation filter criteria**.

   d. Select **Eligible for association** and click **OK**.

   e. Select **SOA1_db2_svcAvgRT** and right-click to select **Associate**.

After the situation is associated, future triggered events will be displayed in the Situation Event Console.

Table 4-3 Mapping table for associating situations with the Navigator items

Situation	Navigator item	Distribution	Command to view the system or system list
SOA1_cicsAppDump .xml	CICS → M194.CICSTS23 → Dump Details	*MVS_CICS	`tacmd listsystemlist`
SOA1_cics_svcAvgR T.xml	Services Management Agent Environment	D4:4b7c211a:soa1was1-serv er1	`tacmd listSystems -t D4`
SOA1_cics_svcFault .xml	Services Management Agent Environment	*SERVICES_MANAGEMENT _AGENT_ENVIR	`tacmd listsystemlist`
SOA1_cics_svcMsg Size.xml	Services Management Agent Environment	D4:4b7c211a:soa1was1-serv er1	`tacmd listSystems -t D4`
SOA1_ctg2CICSCha nnel.xml	CICS → M194.CICSTS23 → Connections Analysis	M194.CICSTS23	`tacmd listSystems -t CP`
SOA1_ctgJ2CConne ction.xml	WebSphere Agent → WebSphere App Server → Log Analysis	server1_default:SOA1WAS1: KYNS server1_default:SOA1WASC: KYNS	`tacmd listSystems -t YN`
SOA1_db2_svcAvgR T.xml	Services Management Agent Environment	D4:4b7c211a:soa1was1-serv er1	`tacmd listSystems -t D4`
SOA1_db2_svcMsg Size.xml	Services Management Agent Environment	D4:4b7c211a:soa1was1-serv er1	`tacmd listSystems -t D4`
SOA1_db2_svcUsag e.xml	Services Management Agent → Message Arrival	*SERVICES_MANAGEMENT _AGENT	`tacmd listsystemlist`
SOA1_e2eRT.xml	SOA1RTT	Primary:SOA1RTT:T2	`tacmd listSystems -t T2`
SOA1_wasAvailabilit y.xml	WebSphere Agent	SOA1WAS1:KYNA, SOA1WASC:KYNA	`tacmd listSystems -t YN`
SOA1_wasLogExcep tion.xml	WebSphere Agent → WebSphere App Server	server1_default:SOA1WAS1: KYNS server1_default:SOA1WASC: KYNS	`tacmd listSystems -t YN`
SOA1_wasStart.xml	WebSphere Agent	SOA1WAS1:KYNA, SOA1WASC:KYNA	`tacmd listSystems -t YN`

4.3.3 Create a custom TEP situation

In our example scenario, we created situations to monitor resources in each layer of the architecture for our sample application. This section describes the steps required to create a custom TEP situation.

Note: The Service and Operation names can either be determined from application design, application WSDL file, or from the Services Inventory workspace in TEP using discovery.

Create service average response time situation

Complete the following steps to create a situation that triggers a warning event if the Average Elapsed Message Round Trip Time is more than 10 seconds (10000 milliseconds) or equal to -1 for the AccountListingSQL service and retrieveAccounts operation. This service is used to retrieve customer information from the DB2 UDB application database.

1. Navigate to **Services Management Agent** for the node where the ITSO Bank Web service application is deployed (for example, soa1was1).
2. Right-click **Services Management Agent Environment**, and select **Situations**.

   Note: By creating the situation using the process we describe, the situation will automatically be associated with the node; when the situation is triggered using this method, it will automatically be displayed in the Situation Event Console.

   If you create a situation by launching the Situation Editor from the toolbar, the situation is not associated with a node by default, and as a result the message will not be displayed in the Situation Event Console.

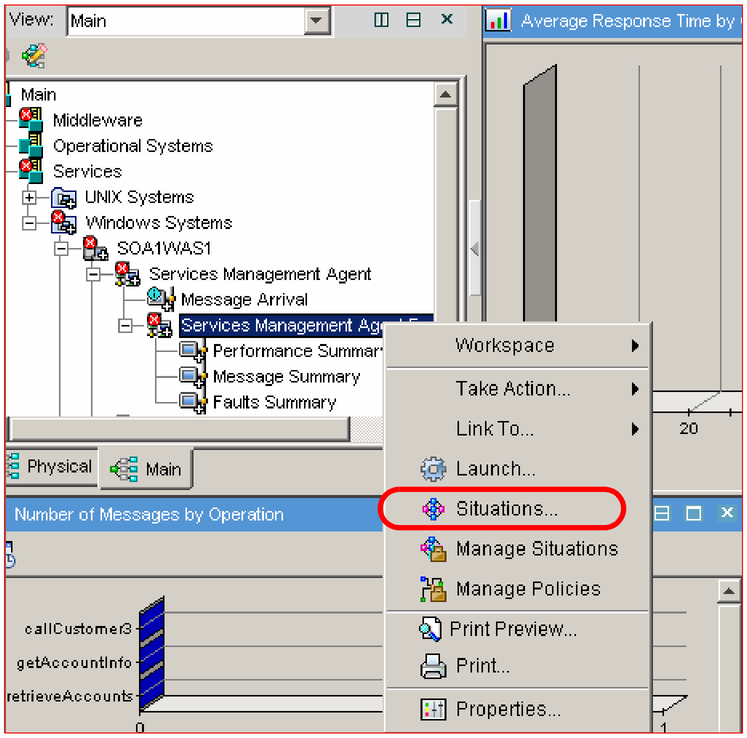


Figure 4-6 Launch Situation Editor from Service Management Agent Environment

3. From the Situation Editor, click the icon **Create new Situation** (second icon in the Situation Editor toolbar).
4. When the Create Situation windows appears, enter the following and click **OK**:
   – Name: `SOA1_db2_svcAvgRT`

   > **Tip:** We recommend that you establish a naming convention for your situations. In our example, we have included SOA1_ as the prefix to our situations to distinguish them from the template situations provided with the products. We also included a descriptive short name for the situation. When exporting situations, we used the situation name in the name of the exported XML file.

   – Description: `Average Roundtrip Response Time >10 seconds or = -1`
5. When the Select Conditions window appears, select **Attribute Comparison** and select **Services_Inventory** from the Attribute Group list.

6. While holding the Ctrl key down, select **Average Elapsed Message Round Trip Time**, **Operation Name**, and **Service Name**. Click **OK**.
7. Click the **Formula** tab and do the following:
   a. Click the cell in column Average Elapsed Message Round Trip Time and row 2. Change the operator to equal to (`=`) and enter value `-1` in the Average Elapsed Message Round Trip Time field.
   b. Click the cell in column Operation Name and row 1, and enter the value `retrieveAccounts`.
   c. Click the cell in column Service Name and row 1, and enter the value `AccountListingSQL`.
   d. Click the cell in column Average Elapsed Message Round Trip Time and row 1. Change the operator to greater than (`>`) and enter value `10000` (measured in milliseconds) in the Average Elapsed Message Round Trip Time field.
   e. Click the cell in column Operation Name and row 2, and enter the value `retrieveAccounts`.
   f. Click the cell in column Service Name and row 2, and enter the value `AccountListingSQL`.
8. In the panel Formula, click the blue icon with symbol **fx** in the top right corner to display the formula as follows:

```
(Average Elapsed Message Round Trip Time ==-1 AND Operation Name ==
retrieveAccounts AND Service Name = AccountListingSQL OR Average
Elapsed Message Round Trip Time > 10000 AND Operation Name ==
retrieveAccounts AND Service Name = AccountListingSQL)
```

9. Check the **Show detailed formula** field at the bottom of the panel. Notice it shows the table name and the attribute names used in the formula. Click **OK**.
10. Click the **State** drop-down list, and select **Warning (yellow)**.
11. Change the Sampling interval to **1** min.
12. Check **Run at startup** and click **Apply**.
13. Click the **Distribution** tab.

    Ensure the soa1was1 node (the node where Web service is deployed) is listed in the Assigned column. Since we launched the Situation Editor from the desired node, it was automatically added to the Assigned column.

Note: If you have a situation that may apply to several nodes, then you can assign individual nodes from the Available Managed Systems column, or select SERVICES_MANAGEMENT_AGENT_ENVIR from the Available Managed Systems list (represents a group of systems).

For example, consider a situation that monitors the availability of an application server or a Web service that is deployed to a cluster or application servers.

14. Click the **Expert Advice** tab.
15. In Text or Advice Location, select the current text and overwrite with the following new text:

```
The average response time exceeded the 10 second threshold. Take the
following steps to troubleshoot:

* Verify the application server is started and running properly.

* Verify the DB2 Server is started and running properly.

* Verify the service message length.
```

Note: The Expert Advice is intended to provide guidance to the Operator to perform the initial troubleshooting. We recommend that you include text to highlight other resources being monitored in support of this transaction to facilitate troubleshooting.

16. Click **Preview** and then click **Close**.
17. Click the **Action** tab.
    a. Select **Universal Message**.
    b. Enter `Services` in the Category field.
    c. Enter `Warning` in the Severity field.
    d. Enter the following in the Message field:

```
SOA1_db2_svcAvgRT: Service & {Services_Inventory.Service_Name}  &
Operation {Services_Inventory.Operation_Name} on  &
{Services_Inventory.Local_Hostname} average response time is not
within the accepted threshold.
```

Tip: We have the following recommendations for the text to be entered in the Message field:

- This message will be logged in the Universal Message Console. We recommend that you prefix the text with the name of the situation (for example, `SOA1_db2_svcAvgRT`).
- Use attribute substitution to create the message by clicking the **Attribute Substitution** button and selecting the service name, operation name, and local hostname. In addition, include descriptive text.
- You have the option of running the system command at the Agent node or at the TEMS server node. Accept the default (TEMS).

18. Click the **Until** tab.

Note: You have the option to stop this situation when another situation becomes True or when a time interval expires.

19. Click **Apply** and click **OK**.

Note: To verify the situation is triggered when the threshold is met, refer to “Verify the SOA1_db2_svcAvgRT situation” on page 96.

Control flow of Web services traffic

By default, ITCAM for SOA monitors all Web services traffic that flows through the Data Collector. You can restrict this by using the `AddMntrCntrl` action command. AddMntrCntrl has variations, which are prefixed with SI- and SM-. When AddMntrCntrl is prefixed with either SI- or SM-, most of the arguments are pre-filled to reduce key strokes for editing the argument for your needs.

Using AddMntrCntrl, you can specify data collection based on a specific service name (WSDL port name) and operation name. You can also specify how much data is logged. Depending on the setting (which could be none, header information only, body information only, or both), the command will log the appropriate level of a SOAP message (for example, SOAP header, SOAP body, both SOAP header and body, or none). Each message is evaluated according to the criteria specified in the AddMntrlCntrl, and if the message matches the criteria, the monitoring data for that message is collected and logged.

If there is no specific criteria for a particular operation, then message data for all services and operations is logged by default (** indicates monitor everything).

For our example scenario, we demonstrate how to limit the data collection to the `showAllVehicles()` operation of the Web service.

1. Run the ITSO Car Rental application to generate Web services traffic.
2. From TEP, click **Message Summary**.

   Ensure messages are being monitored. In our example, we set the collection interval value to **5 minutes** in the History Collection configuration. The message arrival data will appear in the workspace based on the interval set.
3. Click **Services Management Agent**.
4. From the Data Collector Global Configuration window, right-click **WebSphere_Application_Server** and select **Take Action** → **Select**.
5. When the Take Action dialog box opens, do the following:
   a. Select **AddMntrCtrl** from the Name drop-down list.
   b. Click **Arguments**. Enter the values listed in Table 4-4, then click **OK**.

   Table 4-4 Argument details for SI-AddMntrCntrl Take Action Command

Argument Name	Value
Services_Inventory.Application_ServerName	`server1`
Services_Inventory.Application_ServerEnv	`WebSphere_Application_Server`
Services_Inventory.Local_Hostname	`itsoapp1.itso.ral.ibm.com`
Services_Inventory.Service_Name	`CarRentalManager`
Services_Inventory.Operation_Name	`showAllVehicles`
DataCollectorMessageLoggingLevel	`Full`

   c. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm** from Destinations Systems, and click **OK**.
6. When the Action Status dialog appears, you should see `Return Code: 0`, to indicate the command was successful. Click **OK**.
7. Select **View** → **Refresh Now (F5)**.

   You should see an extra line added to the Data Collector Monitor Control Configuration table view. Each row in the table represents a unique definition of monitor criteria.
8. Run the ITSO Car Rental application to generate Web services traffic.
9. Wait for at least five minutes, then select **View** → **Refresh Now**.

10. Click **Message Summary**.

    If there is no traffic being generated or displayed, carefully check for spelling mistakes for the values entered (see Table 4-4 on page 90).

Important: You can specify service and operation names using an asterisk (wild card value), which will match all values for that criteria. You can also specify multiple criteria for a specific application server environment and use a wild card in the criteria. Multiple criteria are evaluated in a specific order, from most specific to least specific. Table 4-5 lists the order in which the criteria are evaluated.

For more detail on the order of evaluation for multiple criteria refer to the "Take Action Commands" chapter of the *Installation and User's Guide, IBM Tivoli Composite Application Manager for SOA*, GC32-9492 product guide*.*

When the Data Collector Monitor Control Configuration is added to or modified, a log file like the following will be created:

```
KD4.1..itsoCell1.itsoNode1.server1.content.log
```

The log file can be imported into IBM Web Services Navigator for offline analysis. For more information on the IBM Web Service Navigator included with ITCAM for SOA, refer to *Installing and Troubleshooting IBM Web Services Navigator*, GC32-9494.

Table 4-5 Precedence order for evaluating multiple monitor criteria

Service name	**Operation name**	**Comments**
CarRentalManager	showAllVehicles	This is most specific and is evaluated first.
CarRentalManager	*	This is less specific than the previous one and is evaluated next.
*	showAllVehicles	This is not a valid combination.
*	*	This is least specific and applies to all; therefore it is evaluated last.

11. Verify the new log file exists.

    The new log file in the C:/CANDLE/CMA/KD4/logs directory on the node the ITCAM for SOA Agent is installed (ITSO Application Server node, itsoapp1). In our example, the log file is named as follows:

    KD4.1..itsoCell1.itsoNode1.server1.content.log

12. Open the ..content.log file in an editor, and verify that log messages for operation `showAllVehicles()` exists.

Filtering a Web service call

Filtering of Web service calls can be very useful for maintaining a service level agreement (SLA). Filtering can also be used to identify availability and performance issues. Common examples of setting filters on Web service calls are as follows:

- Monitor a client application making successive calls for a particular operation.
- Reject calls after a predefined threshold has been reached for a give period.
- Allow only 50 calls to createReservations() within a five-minute interval.

Set filtering on a Web service call

You can use the `AddFltrCntrl` and its variant commands to filter Web service calls and reject those calls. In our example, we demonstrate how to set the filtering to reject messages for `Provider.CarRentalManager.showAllVehicles()`.

1. Navigate to **Performance Summary** under Services Management Agent Environment.
2. In the Service Inventory table, identify an entry that matches the first three column values with Provider, CarRentalManager, showAllVehicles. Select the entry and right-click **Take Action** → **Select**.
3. Select **SI-AddFltCntrl** from the Name action drop-down list. This pre-fills the arguments.
4. Click **Arguments**. Enter the RemoteIPAddress value and click **OK**.

   Note: In our example, we updated the hosts file found in the c:\Windows\system32\drivers\etc directory with the IP address and hostname as follows:

   ```
   192.168.0.100 itsoapp1.itso.ral.ibm.com itsoapp1
   ```

   We were then able to enter the hostname `itsoapp1.itso.ral.ibm.com` as the RemoteIPAddress value.

5. In Destination Systems, click the entry for the application server and click **OK**.
6. When the Action Status dialog box opens, you should see `Return Code: 0`, to indicate the command was successful. Click **OK**.
7. Click the **Services Management Agent** item in the navigator.

   The Data Collector Filter Control Configuration table should show an additional entry with details about the filter that has just been defined.

8. Rerun the ITSO Car Rental application to generate traffic.
9. Select **View** → **Refresh Now**.
10. Click **Services Management Agent Environment**.

    Notice that the messages for `showAllVehicles()` were not rejected.

    Important: The Data Collector failed to reject the message because it did not recognize the remote IP address since both the service requester and provider are on the same host. In our example, the Web service client application and Web services application are deployed on the same WebSphere Application Server node.

    For our WebSphere Application Server, the default_host maps to a node name (for example, itsoapp1.itso.ral.ibm.com); however, TCP/IP is using the local loopback driver to make the inter process communication call.

    Our Data Collector was programmed to look for a message from a remote host with the name itsoapp1.itso.ral.ibm.com, which maps to an IP address like 192.168.0.100. Instead it found a message from an IP address 127.0.0.1, which did not match the evaluation criteria specified in the filter.

11. Click **Services Management Agent**.
12. In the Data Collector Filter Control Configuration view, select the corresponding entry for this filter, right-click and select **Take Action** → **Select**.
13. Select **DelFltrCntrl**.
14. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm.com** from Destinations Systems.
15. Click **OK**.
16. When the Action Status dialog box opens, you should see `Return Code: 0` to indicate the command was successful. Click **OK.**
17. Repeat steps 2 through 7. In step 4, make sure the IP address is set to 127.0.0.1.
18. Rerun the ITSO Car Rental application to generate Web services traffic. In the application, when you click **Submit** after selecting itinerary, you will notice an application error where the Web page failed to display and issued the error: `The page can not be displayed`. The URL in the Web browser should be as follows:

    ```
    http://itsoapp1.itso.ral.ibm.com:9080/ITSOCarRentalWSWeb/SearchServlet
    ```

19. Select **View** → **Refresh Now (F5)**.
20. In the Services Management Agent Environment, click **Faults Summary.**

21. In Faults Summary check **Fault Details** and **Number of Faults by Operation**. Notice the faults raised for `CarRentalManager.showAllVehicles()`.

Delete filtering on a Web services call

In the previous section a filter was defined on a Web service call to reject a message. This may be because we only want to allow a certain number of Web services calls within a time period from an external client. At some point we may want to delete the filter so that normal service can start. Perform the following steps to delete filtering:

1. Click **Performance Summary** under Service Management Agent Environment.
2. In Service Inventory, select the entry with the values Provide, CarRentalManager, showAllVehicles.
3. Right-click **Take Action** → **Select**.
4. Select **SI-DelFltrCntrl**.
5. Specify the remote IP address.

   In our case, we entered `127.0.0.1` because both the Requester and the Provider are running on the same system.
6. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm.com** from Destination Systems and click **OK**.
7. When the Action Status dialog box opens, you should see `Return Code: 0` to indicate the command was successful. Click **OK**.
8. Rerun the ITSO Car Rental application.

   Notice that this time you are able to select the car and reserve a vehicle (you do not see an application error on `searchServlet`).
9. Click **Faults Summary** to ensure that the fault on `showAllVehicles()` is not raised.

5

Monitor resources from the integrated TEP console

Once the integrated TEP console has been configured, you can run the application and verify the monitoring of resources. The redpaper sample application includes error conditions built into the application that can be triggered by entering specific employee IDs while running the application.

This chapter is organized into the following sections:

- ► Monitor the ITSO Bank base environment
- ► Monitor the ITSO Bank enterprise environment

Note: In our redpaper example, we have configured the events to be displayed in the TEP. The events can be configured to be sent to the Tivoli Enterprise Console (TEC).

The TEP situations used to monitor the application are defined in Chapter 4, “Build an integrated TEP console to manage resources” on page 59.

5.1 Monitor the ITSO Bank base environment

This section provides an application walkthrough scenario to verify monitoring functionality for the ITSO Bank when deployed to the base environment. In the base environment, the get customer information service retrieves data from a DB2 UDB application database.

We describe how to verify the following monitoring situations:

- Verify the SOA1_db2_svcAvgRT situation
- Verify the SOA1_db2_svcMsgSize situation
- Verify the SOA1_wasAvailability situation
- Verify the SOA1_wasStart situation
- Verify the SOA1_wasLogException situation
- Verify the SOA1_db2_svcUsage situation

5.1.1 Verify the SOA1_db2_svcAvgRT situation

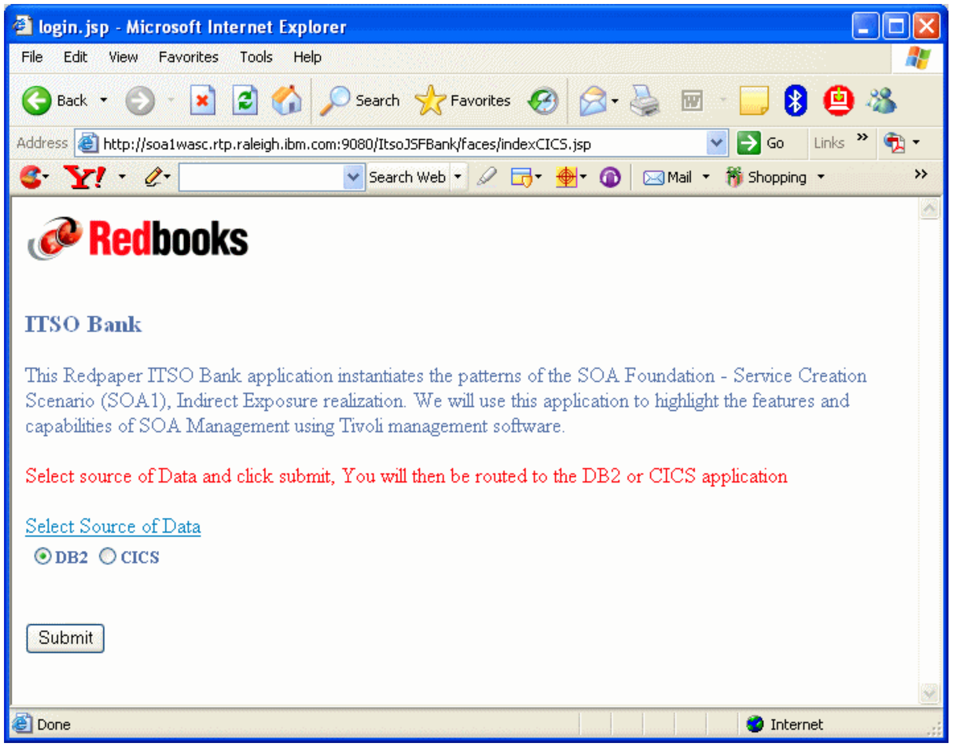
The SOA1_db2_svcAvgRT situation triggers a warning event in the TEP Situations Event Console if the AccountListingSQL service retrieveAccounts operation average round trip response time exceeds 10 seconds.

To verify the SOA1_db2_svcAvgRT situation, do the following:

1. Enter the following URL to access the ITSO Bank:

   ```
   http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
   ```

2. Select **DB2** under Select Source of Data (see Figure 5-1 on page 97), and click **Submit**.



http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/indexCICS.jsp

ITSO Bank

This Redpaper ITSO Bank application instantiates the patterns of the SOA Foundation - Service Creation Scenario (SOA1), Indirect Exposure realization. We will use this application to highlight the features and capabilities of SOA Management using Tivoli management software.

Select source of Data and click submit, You will then be routed to the DB2 or CICS application

Select Source of Data

[✓ DB2] CICS

Figure 5-1 ITSO Bank home page

3. When the ITSO Bank Welcome page appears, select **Delay 60 Seconds** from the Customer ID drop-down list, and click **Submit**. The customer account information should be retrieved.

   Note: In this case, we have inserted code in our application to delay the service for 60 seconds to ensure the 10 second threshold is triggered. Alternatively, set the threshold to a very low value for testing.

4. Verify the monitoring situation event in TEP.

   a. Log on to the TEP.

   b. Launch the Main workspace.

      - All events are logged to the Universal Message Console.
      - Informational, Warning, and Critical events are logged to the Situations Event Console.

        We have included the situation name in the text of the message to make it easier to identify for testing purposes.

c. The operator should take action on events displayed in the Situations Event Console that are in Open status. For example, right-click the warning event from the Situations Event Console, and select **Acknowledge Event** → **Acknowledge**.

d. When the Acknowledge window appears, enter the action performed to analyze and address the event in the Add Notes text field. For example, this event was explained by the application server being stopped. The server has been restarted and the service is running properly.

   If the operator is not able to resolve this issue within a specified period of time (for example, 15 minutes), the problem can be assigned to the Incident Analyst for more in-depth problem determination.

   Click **OK** when done.

e. Notice the Status is now set to Acknowledged, and the user ID of the person that acknowledged the event is displayed in the Situation Events Console window.

5.1.2 Verify the SOA1_db2_svcMsgSize situation

The SOA1_db2_svcMsgSize situation will trigger a critical event in the TEP Situations Event Console if the AccountListingSQL service retrieveAccounts operation message size is less than 600 bytes or over 1000 bytes.

To verify the SOA1_db2_svcMsgSize situation, do the following:

1. Stop DB2 UDB to simulate a problem with the message.

   In this case, the message size will be null since the Web service that wraps the EJB attempting to retrieve data from the database will not be able to connect.

   a. Open a DB2 command window on soa1was1 (node where the Web service application is deployed).

   b. Enter the following DB2 commands:

```
db2 force applications all
db2 terminate
db2 stop
```

2. Enter the following URL to access the ITSO Bank:

```
http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
```

3. Select **DB2** under Select Source of Data, and click **Submit**.

4. When the ITSO Bank Welcome page appears, select **101** from the Customer ID drop-down list, and click **Submit**.

You will see an error 500 message in the Web browser. The customer account information will not be displayed.

5. Verify the monitoring situation event in TEP.
   a. Log on to the TEP.
   b. Launch the Main workspace.
   c. Right-click the message size critical error in the Situation Event Console, select **Acknowledge Event** → **Acknowledge**, and take the appropriate action.

      In this example, more in-depth problem determination may be needed to understand that the connection to DB2 has been terminated due to DB2 UDB being stopped.

5.1.3 Verify the SOA1_wasAvailability situation

The SOA1_wasAvailability situation will trigger a critical event in the TEP Situations Event Console if the WebSphere Application Server - server1 application server is stopped. This situation can be distributed both to the node hosting the Web service client application, and to the node hosting the Web service application.

Note: This situation applies to both the base and enterprise example environments.

To verify the SOA1_wasAvailability situation, do the following:

1. Stop DB2 UDB to simulate a problem.

   The service message size will be null since the Web service that wraps the EJB attempting to retrieve data from the database will not be able to connect.

   a. Open a DB2 command window on soa1was1 (node where the Web service application is deployed).
   b. Enter the following DB2 commands:

```
db2 force applications all
db2 terminate
db2 stop
```

2. Verify the monitoring situation event in TEP.
   a. Log on to the TEP.
   b. Launch the Main workspace.

   c. Right-click the critical error alerting in the Situation Event Console, and select **Acknowledge Event** → **Acknowledge** and take the appropriate action.

5.1.4 Verify the SOA1_wasStart situation

The SOA1_wasStart situation will attempt to start the WebSphere Application Server - server1 application server if the server has been stopped. This is performed using the Take Action option when creating the situation. This situation has also been configured to trigger an information event in the TEP Situations Event Console, indicating that the application server restart has been attempted.

> **Note:** This situation applies to both the base and enterprise example environments.

To verify the SOA1_wasStart situation, do the following:

1. Stop the WebSphere Application Server - server1 application to simulate the problem.

   This step should be performed on each application server node on which you have distributed the situation that you would like to verify.

   a. Open a command window and navigate to the following directory:

   ```
   c:\IBM\WebSphere\AppServer\profiles\default\bin
   ```

   b. Enter the following command:

   ```
   stopServer server1
   ```

2. Verify the monitoring situation event in TEP.

   a. Log on to the TEP.

   b. Launch the Main workspace.

   c. Right-click the information error alert in the Situation Event Console, and select **Acknowledge Event** → **Acknowledge**. The server1 application should be started.

3. Verify the application server has been started.

   This step is only needed for an internal check that the Take Action command to start the application server is working properly.

   a. Open a command window and navigate to the following directory:

   ```
   c:\IBM\WebSphere\AppServer\profiles\default\bin
   ```

   b. Enter the following command:

      ```
      serverStatus server1
      ```

4. Run the ITSO Bank application to verify it is working properly.

   ```
   http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
   ```

5.1.5 Verify the SOA1_wasLogException situation

The SOA1_wasLogException situation will trigger a warning event in the TEP Situations Event Console indicating that the application server restart has been attempted.

> **Note:** This situation applies to both the base and enterprise example environments.

To verify the SOA1_wasLogException situation, do the following:

1. Stop the DB2 UDB to simulate the problem.

   The service message size will be null since the Web service that wraps the EJB attempting to retrieve data from the database will not be able to connect. This will result in an exception in the WebSphere Application Server logs and will trigger the situation.

   a. Open a DB2 command window on soa1was1 (the node where the Web service application is deployed).

   b. Enter the following DB2 commands:

      ```
      db2 force applications all
      db2 terminate
      db2 stop
      ```

2. Enter the following URL to access the ITSO Bank:

   ```
   http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
   ```

3. Select **DB2** under Select Source of Data, and click **Submit**.
4. When the ITSO Bank Welcome page appears, select **101** from the Customer ID drop-down list, and click **Submit**.

   You will see an error 500 message in the Web browser.

5. Verify the monitoring situation event in TEP.

   a. Log on to the TEP.

   b. Launch the Main workspace.

   c. Right-click the warning message alert in the Situation Event Console, and select **Acknowledge Event** → **Acknowledge**.

5.1.6 Verify the SOA1_db2_svcUsage situation

The SOA1_db2_svcUsage situation will trigger a critical event in the TEP Situations Event Console if the AccountListingSQL service retrieveAccounts operation is accessed more than 5 times within the specified interval of 1 minute.

To verify the SOA1_db2_svcMsgSize situation, do the following:

1. Enter the following URL, to access the ITSO Bank:

   `http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp`

2. Select **DB2** under Select Source of Data, and click **Submit**.
3. When the ITSO Bank Welcome page appears, select **101** from the Customer ID drop-down list, and click **Submit**.
4. Repeat the previous steps in 6 separate browser sessions.
5. You will see an error 500 message in the Web browser for one of the sessions. The customer account information will not be displayed.
6. Verify the monitoring situation event in TEP.
   a. Log on to the TEP.
   b. Launch the Main workspace.
   c. Right-click the service usage critical error in the Situation Event Console, select **Acknowledge Event** → **Acknowledge**, and take the appropriate action.

5.2 Monitor the ITSO Bank enterprise environment

This section describes how to verify each of the TEP situations implemented in the application walkthrough scenario to verify monitoring functionality for the ITSO Bank when deployed to the enterprise environment. In the base environment, the get customer information service retrieves data from a COBOL application hosted by the CICS Transaction Server.

We describe how to test the following monitoring conditions:

- Verify the SOA1_cics_svcAvgRT situation
- Verify the SOA1_cics_svcMsgSize situation
- Verify the SOA1_cics_svcFault situation
- Verify the SOA1_ctgJ2CConnection situation
- Verify the SOA1_e2eRT situation
- Verify the SOA1_ctg2CICSChannel situation
- Verify the SOA1_cicsAppDump situation

5.2.1 Verify the SOA1_cics_svcAvgRT situation

The SOA1_cics_svcAvgRT situation will trigger a warning event in the TEP Situations Event Console if the getCustInfo3 service callCustomer3 operation average round trip response time exceeds 10 seconds.

To verify the SOA1_cics_svcAvgRT situation, do the following:

1. Enter the following URL, to access the ITSO Bank:

   `http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp`

2. Select **CICS** under Select Source of Data, and click **Submit**.
3. When the ITSO Bank Welcome page appears, select **60 Sec Delay** from the Customer ID drop-down list, and click **Submit**. The customer account information should be retrieved.

   Note: In this case, we have inserted code in our application to delay the service for 60 seconds to ensure the 10 second threshold is triggered. Alternatively, set the threshold to a very low value for testing.

4. Verify the monitoring situation event in TEP.
   a. Log on to the TEP.
   b. Launch the Main workspace.
   c. Right-click the message size critical error in the Situation Event Console, select **Acknowledge Event** → **Acknowledge**, and take the appropriate action.

5.2.2 Verify the SOA1_cics_svcMsgSize situation

The SOA1_cics_svcMsgSize situation will trigger a critical event in the TEP Situation Event Console if the getCustInfo3 service callCustomer3 operation message size is less than 900 bytes or over 1500 bytes.

To verify the SOA1_cics_svcMsgSize situation, do the following:

1. Stop the CICS Transaction Gateway (CTG) to simulate a problem.

   The Web service wraps a session EJB that uses the CICS ECI resource adapter to retrieve data from CICS via CTG. In this case the message size will be null since CTG is stopped.

2. Enter the following URL, to access the ITSO Bank:

   `http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp`

3. Select **CICS** under Select Source of Data, and click **Submit**.

4. When the ITSO Bank Welcome page appears, select **1** from the Customer ID drop-down list, and click **Submit**.

   You will see an error 500 message in the Web browser. The customer ID information will not be displayed.

5. Verify the monitoring situation event in TEP.

   a. Log on to the TEP.

   b. Launch the Main workspace.

   c. Right-click the message size critical error in the Situation Event Console, select **Acknowledge Event** → **Acknowledge**, and take the appropriate action.

More in-depth problem determination might be needed to understand if CTG and CICS TS are available. In our redpaper examples, we have another situation that monitors the connection and availability of CTG. For details refer to 5.2.4, “Verify the SOA1_ctgJ2CConnection situation” on page 105.

5.2.3 Verify the SOA1_cics_svcFault situation

The SOA1_cics_svcFault situation will trigger a critical event service fault in the TEP Situations Event Console if the getCustInfo3 service callCustomer3 operation is not able to connect to CICS Transaction Server from the CICS Transaction Gateway.

To verify the SOA1_cics_svcFault situation, do the following:

1. Stop the CICS Transaction Server to simulate a problem.

   The Web service wraps a session EJB that uses the CICS ECI resource adapter to retrieve data from CICS via CTG. In this case, ITCAM for SOA will detect the Web service fault when CTG fails to connect to CICS.

2. Enter the following URL to access the ITSO Bank:

   `http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp`

3. Select **CICS** under Select Source of Data, and click **Submit**.

4. When the ITSO Bank Welcome page appears, select **1** from the Customer ID drop-down list, and click **Submit**.

   You will see an error 500 message in the Web browser.

5. Verify the monitoring situation event in TEP.

   a. Log on to the TEP.

   b. Launch the Main workspace.

   c. Right-click the service fault critical error in the Situation Event Console, select **Acknowledge Event** → **Acknowledge**, and take the appropriate action.

More in-depth problem determination may be needed to understand if CTG and CICS TS are available. In our redpaper examples, we have another situation that monitors the connection and availability of CTG. For details refer to the next section.

5.2.4 Verify the SOA1_ctgJ2CConnection situation

The SOA1_ctgJ2CConnection situation will trigger a critical event in the TEP Situations Event Console if the CICS ECI resource adapter called by the session EJB is not able to connect to the CICS Transaction Gateway (CTG). We will monitor the availability of the CTG by detecting connection exceptions in the WebSphere Application Server SystemOut.log.

To verify the SOA1_ctgJ2CConnection situation, do the following:

1. Stop the CICS Transaction Gateway to simulate a problem.

   The Web service wraps a session EJB that uses the CICS ECI resource adapter to retrieve data from CICS via CTG. If CTG is not available, the connection from the application will write an exception to the WebSphere Application Server SystemOut.log. We have created a TEP situation that uses ITCAM for WebSphere to detect this exception.

2. Enter the following URL to access the ITSO Bank:

   ```
   http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
   ```

3. Select **CICS** under Select Source of Data, and click **Submit**.
4. When the ITSO Bank Welcome page appears, select **1** from the Customer ID drop-down list, and click **Submit**. You will see a CICS connection exception error message.
5. Verify the monitoring situation event in TEP.
   a. Log on to the TEP.
   b. Launch the Main workspace.
   c. Right-click the critical error in the Situation Event Console, select **Acknowledge Event** → **Acknowledge**, and take the appropriate action.

Note: In addition to displaying the critical error message, the situation could be modified to include a Take Action command that will attempt to automatically start the CICS Transaction Gateway.

5.2.5 Verify the SOA1_e2eRT situation

The SOA1_e2eRT situation will trigger a warning event in the TEP Situations Event Console if the total end-to-end response time duration of the entire CICS Transaction as defined in RTT is exceeded. For example, Web service client calls Web service, which calls CTG to communicate with CICS Transaction Server. In our example enterprise environment, we defined the total transaction time in ITCAM for RTT as 20 seconds.

To verify the SOA1_e2eRTT situation, do the following:

1. Enter the following URL, to access the ITSO Bank:

   `http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp`

2. Select **CICS** under Select Source of Data, and click **Submit**.
3. When the ITSO Bank Welcome page appears, select **60 Sec Delay** from the Customer ID drop-down list, and click **Submit**. The customer account information should be retrieved.
4. Verify the monitoring situation event in TEP.

   When monitoring the end-to-end transaction using ITCAM for RTT, there are a couple of key points to understand that are unique to ITCAM for RTT:

   – The threshold of monitoring is defined in native ITCAM for RTT - Managing Server Console. This includes defining a report where the data will be displayed.

   – Summaries of reports are listed in the TEP, under the hostname of the ITCAM for RTT - Managing Server.

   – The reports are categorized based on the type of report (for example, All Reporting Groups, Applications, Customers, Locations).

   – More detailed information on the transaction performance can be obtained in the native ITCAM for RTT - Managing Server Console.

   The steps to acknowledge the event are the same.

5.2.6 Verify the SOA1_ctg2CICSChannel situation

The SOA1_ctg2CICSChannel situation will trigger a critical event in the TEP Situations Event Console if the channel between CTG and CICS is not available. CTG communicates with CICS through the EXCI. ITCAM for CICS Transaction is used to monitor the channel.

To verify the SOA1_ctg2CICSChannel situation, do the following:

1. To simulate the CICS channel not being available, use the following CICS command:

   ```
   CP:SET CON(GR1G)Out
   ```

2. Enter the following URL to access the ITSO Bank:

   ```
   http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
   ```

3. Select **CICS** under Select Source of Data, and click **Submit**.
4. When the ITSO Bank Welcome page appears, select **1** from the Customer ID drop-down list, and click **Submit**.
5. Verify the monitoring situation event in TEP.
   a. Log on to the TEP.
   b. Launch the Main workspace.
   c. Right-click the critical error in the Situation Event Console, select **Acknowledge Event** → **Acknowledge**, and take the appropriate action.

5.2.7 Verify the SOA1_cicsAppDump situation

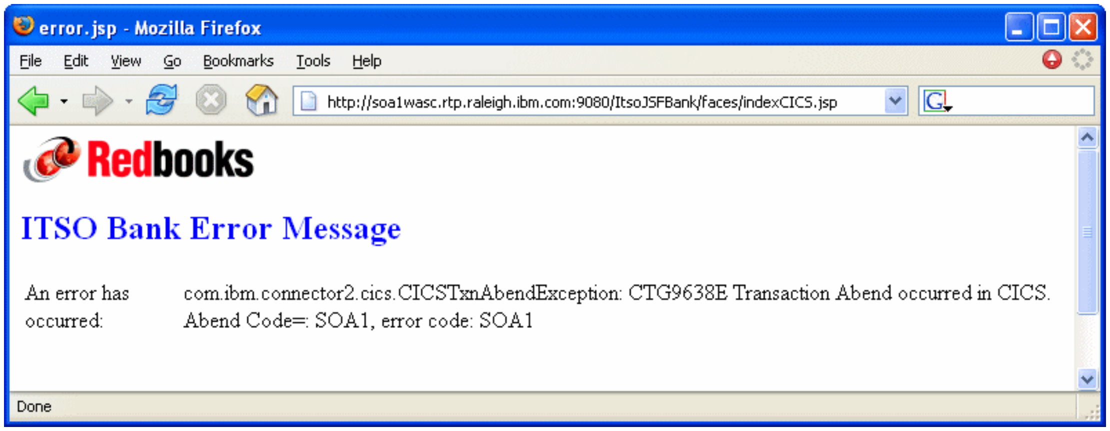
The SOA1_cicsAppDump fault situation will trigger a warning event in the TEP Situations Event Console if a CICS abend is triggered using the sample application. This event is monitored by OMEGAMON XE for CICS.

To verify the SOA1_cicsAppDump situation, our test CICS program can simulate an abend. To test this, perform the following steps:

1. Enter the following URL to access the ITSO Bank:

   ```
   http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
   ```

2. Select **CICS** under Select Source of Data, and click **Submit**.
3. When the ITSO Bank Welcome page appears, select **Trigger Abend** from the Customer ID drop-down list, and click **Submit**. The customer account information will not be received, since the application intentionally abends so that this fault can be simulated. You will see an error message like the one in Figure 5-2 on page 108.


error.jsp - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/indexCICS.jsp
Redbooks
Done

ITSO Bank Error Message

An error has occurred: com.ibm.connector2.cics.CICSTxnAbendException: CTG9638E Transaction Abend occurred in CICS. Abend Code=: SOA1, error code: SOA1

Figure 5-2 Example of CICS Abend

4. Verify the monitoring situation event in TEP.
   a. Log on to the TEP.
   b. Launch the Main workspace.
   c. Right-click the service fault critical error in the Situation Event Console, select **Acknowledge Event** → **Acknowledge**, and take the appropriate action.

A

Deploy the environment and sample application

This appendix describes the steps required to deploy the sample application to both the base and enterprise environments. In addition, we have included a procedure to import the sample application source code, which has been packaged in a project interchange file, into a clean IBM Rational Application Developer workspace.

The appendix is organized into the following sections:

- Prepare the environment
- Deploy the application to the base environment
- Deploy the application to the enterprise environment
- Start servers and services for monitoring
- Import the project interchange file

Prepare the environment

This section outlines the steps to prepare the environment as a prerequisite to deploying the application and building an integrated console for monitoring resources.

- Base environment

  The base environment requires fewer resources and nodes on a distributed platform (for example, Windows). In this environment, the Web service retrieves customer information from a DB2 UDB database.

- Enterprise environment

  The enterprise environment is more advanced. It includes the base environment plus CICS Transaction Server and supporting management software on z/OS. In this environment, the Web service retrieves customer data from a COBOL application hosted by CICS Transaction Server.

Note: The logical and physical architectures for the base and enterprise environments can be found in 3.3, "Design the management architecture" on page 49.

Deploy base environment

The base environment includes the following nodes:

- Web Service Client - WebSphere Application Server node
  This WebSphere Application Server node hosts the Web Services client sample application. The node is managed by ITCAM for SOA Agent, WebSphere Data Collector and TEMA, and RTT Management Agent.
- Web Service - WebSphere Application Server node
  This WebSphere Application Server node hosts the Web Service, EJB, and DB2 database for the sample application. The node is managed by ITCAM for SOA Agent, WebSphere Data Collector and TEMA, and RTT Management Agent.

  Note: The Web Services Client application and Web Service applications can be installed on the same node to reduce the number of nodes.

- Tivoli Monitor Server node
  This node contains Tivoli Monitoring (TEMS, TEPS, TEP Desktop Client), and Application Support for ITCAM for SOA, WebSphere, and RTT (server component).

- ITCAM for WebSphere - Managing Server node
  This node is the Managing Server for ITCAM for WebSphere, which has been integrated with the Tivoli Monitoring Server.
- ITCAM for RTT - Managing Server
  This node is the Managing Server for ITCAM for RTT, which has been integrated with the Tivoli Monitoring Server.
- Tivoli Service Level Advisor (TSLA) node
  The TSLA node is integrated with the Tivoli Monitoring Server node.

The following six tables summarize these software requirements, including version and service level details.

Table 5-1 Web Services Client - WebSphere Application Server node

Software	**Version**	**Service level**
Microsoft® Windows Server®	2003 Standard Edition	Service Pack 1 + Critical Fixes
IBM WebSphere Application Server	6.0	Refresh Pack 2 + Fixpack 9
ITCAM for WebSphere * Data Collector (DC) * TEMA	6.0	DC - FP0003, Interim Fixes 1, 2, 3, 5
ITCAM for Response Time Tracking (RTT) * Management Agent (MA)	6.1	
ITCAM for SOA	6.0	

Table 5-2 Web Service - WebSphere Application Server node

Software	**Version**	**Service level**
Microsoft Windows Server	2003 Standard Edition	Service Pack 1 + Critical Fixes
IBM WebSphere Application Server Network Deployment **Note:** Base WebSphere Application Server can be used in place of ND.	6.0	Refresh Pack 2 + Fixpack 9
IBM DB2 UDB Enterprise Server Edition	8.2	Fixpack 12
ITCAM for WebSphere * Data Collector (DC) * TEMA	6.0	DC - FP0003, Interim Fixes 1, 2, 3, 5
ITCAM for Response Time Tracking (RTT) * Management Agent (MA)	6.1	
ITCAM for SOA	6.0	

Table 5-3 Tivoli Monitoring Server node

Software	**Version**	**Service level**
Microsoft Windows Server	2003 Standard Edition	Service Pack 1 + Critical Fixes
IBM DB2 UDB Enterprise Server Edition	8.2	Fixpack 12
IBM Tivoli Monitoring (ITM) * Tivoli Enterprise Monitor Server (TEMS) * Tivoli Enterprise Portal Server (TEPS) * Tivoli Enterprise Portal Desktop Client	6.1	FP0004
ITCAM for WebSphere - Application Support **Note:** Server component to integrate with ITM	6.0	FP0003
ITCAM for RTT - Application Support **Note:** Server component to integrate with ITM	6.1	
ITCAM for SOA - Application Support **Note:** Server component to integrate with ITM	6.0	

Table 5-4 ITCAM for WebSphere - Managing Server node

Software	**Version**	**Service level**
Microsoft Windows Server	2003 Standard Edition	Service Pack 1 + Critical Fixes
Microsoft Windows Service for UNIX	3.5	Hotfix 899522
IBM WebSphere Application Server	6.0	Refresh Pack 2 + Fixpack 9
IBM DB2 UDB Enterprise Server Edition	8.2	Fixpack 12
ITCAM for WebSphere - Managing Server	6.0	FP0003

Table 5-5 ITCAM for RTT - Managing Server node

Software	**Version**	**Service level**
Microsoft Windows Server	2003 Standard Edition	Service Pack 1 + Critical Fixes
IBM WebSphere Application Server	6.0	Refresh Pack 2 + Fixpack 9
IBM DB2 UDB Enterprise Server Edition	8.2	Fixpack 12
ITCAM for RTT * Managing Server * TEMA	6.1	

Table 5-6 Tivoli Server Level Advisor node

Software	Version	Service level
Microsoft Windows Server	2003 Standard Edition	Service Pack 1 + Critical Fixes
IBM WebSphere Application Server	6.0	Refresh Pack 2 + Fixpack 9
IBM DB2 UDB Enterprise Server Edition	8.2	Fixpack 12
IBM Tivoli Service Level Advisor	2.1.1	

Deploy enterprise environment

The enterprise environment includes everything in the base environment plus the addition of CICS Transaction Server and supporting management components:

- Web Service Client - WebSphere Application Server node

  Same as base environment.

- Web Service - WebSphere Application Server node

  Same as base environment, plus the addition of the CICS ECI Resource Adapter to communicate with the CICS Transaction Gateway.

  Note: The Web Services Client application and Web Service applications can be installed on the same node to reduce the number of nodes.

- Tivoli Monitor Server node

  Same as base environment, plus the addition of OMEGAMON XE for CICS - Application Support.

- ITCAM for WebSphere - Managing Server node

  Same as base environment. At the time of writing, we found that deploying the ITCAM for WebSphere - Managing Server on Linux was more robust for CICS Data Collector integration.

- ITCAM for RTT - Managing Server node (same as base environment)
- Tivoli Service Level Advisor node (same as base environment)
- CICS Transaction Server node

  This node hosts the back-end Cobol application. The node includes CICS Transaction Server (TS) and CICS Transaction Gateway (CTG). OMEGAMOM XE for CICS Agent is used to manage CICS TS. ITCAM for CICS Transactions, ITCAM for WebSphere - Data Collector, and RTT Management Agent are used to manage application and CTG.

Table 5-7 CICS backend node

Software	Version	Service level
IBM z/OS	1.5	
CICS Transaction Server (TS)	2.3	
CICS Transaction Gateway (CTG)	6.0	
OMEGAMON XE for CICS Agent **Note:** Manage CICS TS	3.10	
ITCAM for CICS Data Collector	6.0	
ITCAM for WebSphere - Data Collector	6.0	
ITCAM for RTT - Management Agent	6.1	

Where to find information on how to install and configure

For details on installing the software for each of these nodes found in the base and enterprise environments, refer to the following sources of information:

- InfoCenters for Tivoli Monitoring products:

  ```
  http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=/com
  .ibm.tivoli.az.doc/welcome.htm
  ```

- IBM Redbooks:
  - *IBM Tivoli Composite Application Manager V6.0 family*, SG24-7151
  - *IBM Tivoli OMEGAMON XE V3.1.0 Deep Dive on z/OS*, SG24-7155

Tip: To avoid integration and coexistence issues, we recommend that you install ITCAM for WebSphere - Data Collector before ITCAM for RTT - Management Agent, and apply the prescribed fixpacks and interim fixes.

Download the sample code for the redpaper

Refer to Appendix C, "Additional material" on page 139 for details on downloading the 4233code.zip containing the sample application and TEP situations.

Deploy the application to the base environment

This section outlines the steps required to deploy the sample application to the base environment.

Prerequisites to deploy application

Ensure the following prerequisites to deploy the sample application.

1. Ensure the environment is installed and configured.

   This appendix assumes that you have deployed the base environment defined in “Deploy base environment” on page 110.

2. Start the application server where you wish to deploy the sample application.

   For example, start the server1 application server on the following nodes:

   – Web Services Client - WebSphere Application Server node
   – Web Service - WebSphere Application Server node

3. Start DB2 UDB on the Web Server - WebSphere Application Server node.

Deploy the ITSO Bank DB2 UDB database

To create the ITSO Bank DB2 UDB application database on the Web Service - WebSphere Application Server node (soa1was1), follow these steps:

1. Open a DB2 UDB command window by selecting **Start** → **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**.

2. Create the database.

   From the DB2 command window, enter the following command to create the itsodb database:

   ```
   db2 create db itsobank
   ```

3. Connect to the itosdb database:

   ```
   db2 connect to itsobank
   ```

4. Create tables.

   Enter the following command to run the ITSOBank.ddl to create database tables:

   ```
   db2 -vf C:\4233code\deploy\db2\ITSOBank.ddl
   ```

   Note: Do not enter **`-tvf`** (only **`-vf`** since delimited with spaces).

5. Populate the tables with sample data.

   Enter the following commands to run the ITSOBank.sql to populate the database with sample data:

   ```
   db2 -vf C:\4233code\deploy\db2\ITSOBank.sql
   ```

   Note: Do not enter **`-tvf`** (only **`-vf`** since delimited with spaces).

6. Disconnect from the itsodb database:

   ```
   db2 disconnect itsobank
   ```

Create the ITSO Bank application datasource

This section describes how to configure the ITSO Bank application database datasource on the Web Service - WebSphere Application Server node (soa1was1).

Complete the following tasks:

1. Create a J2C authentication alias
2. Create a JDBC provider for DB2 UDB
3. Create the DB2 data source
4. Test the connection

Create a J2C authentication alias

To create a J2C authentication alias required to access a DB2 UDB database, follow these steps:

1. Click **Security** → **Global security** from the Administrative Console.
2. Under Authentication, expand **JAAS Configuration** and click **J2C Authentication data**. Click **New**.
3. When the Configuration dialog box opens, enter the following values, then click **OK**:
   - Alias: `itsobank`
   - User ID: `admin`

   Note: In our example, we created the admin user ID as an administrative user. By default, the user logged into Windows when the database is created is used as the database owner.

   - Password: `<password>`
   - Description: `ITSO Bank database authentication alias`
4. Click **Save**. Click **Save** again to save to master configuration.

Create a JDBC provider for DB2 UDB

To create a JDBC provider for DB2 UDB at the node level, follow these steps:

1. Click **Resources** → **JDBC Providers** from the Administrative Console.
2. Click **Browse Nodes**, select **soa1was1Node1**, and click **OK**.
3. With the node scope applied, click **New** to create the JDBC provider.

4. When the General Properties dialog box opens, select these options, then click **Next**:
   - Select **DB2** from the Step 1: Select the database type drop-down list.
   - Select **DB2 Universal JDBC Driver Provider** from the Step 2: Select the provider type drop-down list.
   - Select **Connection pool data source** from the Step 3: Select the implementation type drop-down list.
5. When the variables dialog box opens, change the variables displayed in the classpath text box as follows, then click **OK**:
   - Classpath:

     ```
     C:/IBM/SQLLIB/java/db2jcc.jar
     C:/IBM/SQLLIB/java/db2jcc_license_cu.jar
     C:/IBM/SQLLIB/java/db2jcc_license_cisuz.jar
     ```

   - Native library path: delete variables
6. Click **Save**. Click **Save** again to save to master configuration.

Create the DB2 data source

To create the DB2 UDB data source for the ITSO Car Rental application database, do the following:

1. Click **Resources** → **JDBC Providers**.
2. Click **DB2 Universal JDBC Driver Provider**.
3. Click **Data sources** under Additional Properties.
4. Click **New**.
5. Enter the following:
   - Name: `itsobankDS`
   - JNDI name: `jdbc/ejbbank`
   - Component-managed authentication alias: `soa1was1Node01/itsobank`
   - Database name: `itsobank`
   - Driver type: `4`

     Note: The default is type 2. Server name and port number are only required if you use driver type 4. Type 4 JDBC drivers are direct-to-database pure Java drivers (*thin* drivers).

   - Server name: `soa1was1.rtp.raleigh.ibm.com`

     This is the hostname of the server where DB2 UDB is installed.

   - Port number: `50000`

     This is the port defined in the services file for the DB2 UDB connection port.

   - Leave all other values as default. Click **OK**.

6. Click **Save**. Click **Save** again to save to master configuration.
7. Click **OK**.

Test the connection

To test the connection, follow these steps:

1. Select **Resources** → **JDBC Providers**.
2. Click **DB2 Universal JDBC Driver Provider**.
3. Click **Data sources**.
4. Check the **itsobankDS** data source and click **Test connection**. Look for a success message.

   Note: If the connection test does not work, restart the server1 and node agent and try again.

Enable performance monitoring instrumentation on WebSphere Application Server

Enable the following PMI setting to allow WebServices ARM instrumentation:

1. Start the Administrative Console by entering the following URL in a Web browser for the Web Service - WebSphere Application Server node (soa1was1):

   ```
   http://soa1was1.itso.ral.ibm.com:9060/ibm/console
   ```

2. When prompted, enter the login user ID, then click **Login**. At this time, WebSphere security is not enabled.
3. Click **Monitoring and Tuning**, then select **PMI**.
4. Select the **All** radio button. This will enable all statistics.
5. Click **Save**. Click **Save** again to save to master configuration.
6. Click **OK**.
7. You will need to restart the server for this change to be effective

Deploy the Web Service application

To install the ITSO Bank Web Service enterprise application on the Web Service - WebSphere Application Server node (soa1was1), follow these steps:

1. Start the Deployment Manager Administrative Console by entering the following URL in a Web browser for the Web Service - WebSphere Application Server node (soa1was1):

   ```
   http://soa1was1.rtp.raleigh.ibm.com:9060/ibm/console
   ```

2. When prompted, enter the login user ID, then click **Login**. At this time, WebSphere security is not enabled.
3. Expand the **Applications** tab; click **Install New Application**.
4. When the Preparing for the application installation dialog box opens, complete these tasks and click **Next**:
   - Select **Local file system**
   - Specify path: `C:\4233code\deploy\ITSOBankDB2WS.ear`
5. When the Generate default bindings dialog box opens, accept the defaults and click **Next**.
6. When the Step 1: Select installation options dialog box opens, accept default settings and click **Next**.
7. When the Step 2: Map modules to servers dialog box opens, accept default options and click **Next**.
8. When the Step 3: Map resource references to resources dialog opens, accept default and click **Next**.
9. When the Step 4: Map virtual hosts for Web modules dialog opens, accept the default (default_host) and click **Next**.
10. When the Step 5: Provide default data source mapping for modules containing 2.x entity beans dialog box opens, accept the defaults, then click **Next**. In our example, the JNDI name is jdbc/ejbbank and is defined in the application deployment descriptor.
11. Accept defaults for the remaining steps. When on the Summary page, click **Finish**.

    You should see a message similar to the following if you are successful:

    ```
    Application ITSOBankDB2WS installed successfully
    ```

12. Click **Save to Master Configuration**, and click **Save**.
13. Expand **Applications** → **Enterprise Applications**.
14. Check **ITSOBankDB2WS** and click **Start**.

Deploy the Web Services Client application

Follow these steps to install the ITSO Bank Web Services client application on the Web Services Client - WebSphere Application Server node (soa1wasc):

1. Expand the **Applications** tab, then click **Install New Application** from the Administrative Console.
2. In the Preparing for the application installation dialog box, complete the following items, then click **Next**:
   - Select **Local file system**.
   - Specify path: `C:\4233code\deploy\ITSOBankWSWeb.ear`
3. On the Generate default bindings dialog box, accept the defaults and click **Next**.
4. On the Step 1: Select installation options dialog box, accept the defaults and click **Next**.
5. Accept defaults for the remaining steps. When the Summary dialog box opens, review the settings and click **Finish**.

   You should see a message similar to the following if you are successful:

   ```
   Application ITSOBankWSWeb installed successfully
   ```
6. Click **Save to Master Configuration**, and click **Save**.
7. Expand **Applications** → **Enterprise Applications**.
8. Check **ITSOBankWebService** and click **Start**.
9. Expand **Applications** → **Enterprise Applications**.
10. Check **ITSOBankWSWeb** and click **Start**.
11. Modify the Endpoint hostname for the Web Service client binding.
    a. Expand **Applications** → **Enterprise Applications**.
    b. Click **ITSOBankWSWeb**.
    c. Click **Web modules** under Related Items.
    d. Click **ITSOBankWebPROJ.war**.
    e. Click **Web services client bindings**.
    f. Click **Edit** under Port Information for the AccountListingSQLService.
    g. Enter the following URL in the Overridden Endpoint URL (hostname of node where Web Service application is deployed) and click **OK**:

    ```
    http://soa1was1.rtp.raleigh.ibm.com:9080/ItsoBankingWebService/servi
    ces/AccountListingSQL
    ```
    h. Click **Edit** under Port Information for the getCustInfo3Service.

   i. Enter the following URL in the Overridden Endpoint URL (hostname of node where Web Service application is deployed) and click **OK**:

      ```
      http://soa1was1.rtp.raleigh.ibm.com:9080/routerProject/services/getC
      ustInfo3
      ```

12. Click **Save**, and click **Save**.
13. Launch the ITSO Bank Web Service Client application by entering the following URL in a Web browser:

    ```
    http://soa1wasc.rtp.raleigh.ibm.com:9080/ItsoJSFBank/faces/ItsoBank.jsp
    ```

Deploy the application to the enterprise environment

This section outlines the steps required to deploy the sample application to the enterprise environment. The enterprise environment includes the deployment of the base environment, plus the addition of the CICS Resource Adapter and backend COBOL application hosted by CICS TS.

Prerequisites to deploy application

Ensure the following prerequisites are met before you deploy the sample application.

1. Ensure the environment is installed and configured.

   This following procedures assume that you have deployed the enterprise environment defined in "Deploy enterprise environment" on page 113.
2. Start the application server on which you wish to deploy the sample application.

   For example, start the server1 application server on the following nodes:

   – Web Services Client - WebSphere Application Server node
   – Web Service - WebSphere Application Server node
3. Start the CICS Transaction Gateway.
4. Start the CICS Transaction Server.

Deploy the Cobol application to CICS TS

The sample code packaged with this redbook includes a COBOL application that needs to be installed to your CICS Transaction Server.

The COBOL application is a simple application that stores customer-related information in memory, with no database required. This application is also easily modified to fit the requirements for your SOA management environment. Given a customer number, WBCSCUST will return information about that customer in the CICS DFHCOMMAREA. This application is called from WebSphere via a Web service that utilizes the CICS Transaction gateway, which obtains the results from CICS.

The COBOL application source can be found in the following file once the 4233code.zip is unzipped:

```
c:\4233code\src\itsobankcobol4cics.cbl
```

1. Use ftp or a similar program to copy the COBOL program from c:\4233code\src\itsobankcobol4cics.cbl to a z/OS data set.
2. Compile and link-edit the conversion program to create the load module to be deployed to CICS. You can use the z/OS Application Development tools in WebSphere Developer, or native z/OS tools, to generate and submit the needed JCL.
3. After you create the load module, make it available to CICS by copying to a PDS or library that is part of the DFHRPL concatenation in the CICS start-up job. If CICS program auto-install is not active, then you need to define a program resource for WBCSCUST.

For more detailed instructions for installing this application, refer to the CICS Transaction Server product documentation.

Modify the code page conversion table

The ITCAM for WebSphere V6 - Data Collector installed on the first node of the transaction (for example, Web Service Client application server node) inserts an 11 byte token to the beginning of the message sent to CICS TS (in the applications DFHCOMMAREA), so that it can correlate the end-to-end transaction. The ITCAM for WebSphere - Data Collector installed on z/OS strips this 11 byte token out before the message is received by CICS TS.

The code page conversion table (DFHCNV) is used to convert messages from an ASCII (distributed platform) code page to EBCDIC (z/OS platform) and vice versa. The conversion table has to account for the 11 bytes inserted by the ITCAM for WebSphere - Data Collector.

Example A-1 on page 123 includes a sample of the code page conversion table we modified for our environment for use by the WBCSCUST application.

For more detailed information, refer to Appendix B, “DFHCNV and CICS data conversion,” in the redbook *CICS Transaction Gateway for z/OS Version 6.1*, SG24-7161.

Attention: If you are using ITCAM for WebSphere - Data Collector, and your application requires a DFHCNV conversion table, you must modify the conversion table for the application to account for the 11 byte correlation token added by ITCAM for WebSphere. Failure to perform this might cause your applications to fail or produce inaccurate results.

Example: A-1 IBM Redbook sample conversion table

```
//TABLEASM EXEC DFHAUPLE
//ASSEM.SYSUT1 DD *
         DFHCNV TYPE=INITIAL
         DFHCNV TYPE=ENTRY,RTYPE=PC,RNAME=WBCSCUST,USREXIT=NO,         X
               SRVERCP=037,CLINTCP=850
         DFHCNV TYPE=SELECT,OPTION=DEFAULT
* The first field at OFFSET=0 originally had a DATALEN=4, but
* has been adjusted by 11 bytes to account for the ITCAM for WebSphere
* correlation token
*
         DFHCNV TYPE=FIELD,OFFSET=0,DATATYP=BINARY,DATALEN=15
         DFHCNV TYPE=FIELD,OFFSET=15,DATATYP=CHARACTER,DATALEN=100
         DFHCNV TYPE=FIELD,OFFSET=100,DATATYP=BINARY,DATALEN=2,        X
               LAST=YES
         DFHCNV TYPE=FINAL
         END    DFHCNVBA
/*
```

Deploy the JCA Web Service for the CICS application

To install the ITSO Bank JCA Web Service for the CICS enterprise application, follow these steps:

1. Start the WebSphere Administrative Console by entering the following URL in a Web browser for the Web Service - WebSphere Application Server node (soa1was1):

   `http://soa1was1.itso.ral.ibm.com:9060/ibm/console`

2. When prompted, enter the login user ID, then click **Login**. At this time, WebSphere security is not enabled.
3. Expand the **Applications** tab; click **Install New Application**.
4. When the Preparing for the application installation dialog box opens, complete these tasks and click **Next**:
   – Select **Local file system**

– Specify path: `C:\4233code\deploy\ITSOBankJCAWS.ear`

5. When the Generate default bindings dialog box opens, accept the defaults and click **Next**.
6. When the Step 1: Select installation options dialog box opens, accept default settings and click **Next**.
7. Accept the default settings for the remaining steps. When the Summary page appears, click **Finish**.

   You should see a message similar to the following if you are successful:

   ```
   Application ITSOBankJCAWS installed successfully
   ```
8. Click **Save to Master Configuration**, and click **Save**.
9. Expand **Applications** → **Enterprise Applications**.
10. Check **ITSOBankJCAWS** and click **Start**.

Deploy CICS ECI Resource Adapter

To deploy the CICS ECI Resource Adapter to the WebSphere Application Server where the application is deployed, complete the following steps:

1. Copy the cicseci.rar installed with the CICS Transaction Gateway (CTG) to the c:\temp directory of the application server on which the file will be deployed.
   – From the z/OS node where CTG is installed: <ctg_root>/deployable
   – To application server node: c:\temp

   Note: The CICS ECI Resource Adapter (cicseci.rar) is also packaged with IBM Rational Application Developer. Due to version dependency issues with CTG, we recommend you use the cicseci.rar included with CTG.

2. Start the WebSphere Administrative Console by entering the following URL in a Web browser for the Web Service - WebSphere Application Server node (soa1was1):

   ```
   http://soa1was1.itso.ral.ibm.com:9060/ibm/console
   ```
3. When prompted, enter the login user ID, then click **Login**. At this time, WebSphere security is not enabled.
4. Expand the **Resources** tab; click **Resource Adapters**.
5. Click **Install RAR**.
6. When the Install RAR file dialog box opens, do the following and click **Next**:
   – Select **Local file system**
   – Specify path: `c:\temp\cicseci.rar`

7. When the Resource Adapters page opens, enter `CICSECI` in the Name field and click **OK**.
8. Click **Save**, and click **Save** again to save to master configuration.
9. Create the connection factory:
   a. Expand the **Resources** tab, and click **Resource Adapters**.
   b. Click the newly created adapter (CICSECI).
   c. Click **J2C Connection Factories** under Additional Properties.
   d. Click **New**.
   e. Enter `CICSTR` in the Name field, enter `CICSTR` in the JNDI Name field, and click **Apply**.
   f. Click **Custom Properties**.
   g. Enter the following and click **OK**:
      - ConnectionURL: tcp://<cics_hostname>
      - PortNumber: <CTG_defined_port> (for example, 13101)

      Note: The PortNumber is defined in the CTG initialization file (location written to log file, which is visible at startup).

      - ServerName: `CICSTR`
10. Click **Save**, and click **Save** again to save to master configuration.

Start servers and services for monitoring

This section outlines the monitoring servers and services, application servers, data collectors, and agents that need to be started by node to perform the discovery, monitoring, and management of resources.

We have listed the servers and services that need to be started by node:

- Web Service Client - WebSphere Application Server
  - Start the application server (for example, server1) where the Web Service Client application is deployed.
- Web Service - WebSphere Application Server
  - Start the DB2 UDB Windows services.
  - Start the application server (for example, server1) where the Web Service and EJB application are deployed. This will also start the ITCAM for WebSphere Data Collector and RTT Management Agent, since they are deployed to the application server.

  - Start the Tivoli Monitoring Services:
    - Start the ITCAM for SOA service from the Tivoli Monitoring Service Console.
    - Start the ITCAM for WebSphere Agent (TEMA) service from the Tivoli Monitoring Service Console.
- Monitoring Server
  - Start the DB2 UDB Windows services.
  - Start the following Tivoli Monitoring Services (for example, TEMS, TEPS).
- ITCAM for WebSphere - Managing Server
  - Start the DB2 UDB Windows services.
  - Start the application server (for example, server1) where the ITCAM for WebSphere - Managing Server is deployed.
  - Run am-start.sh.
- ITCAM for RTT - Managing Server
- If you are using the enterprise environment, you will also need start servers on the CICS backend node as follows:
  - Start CICS Transaction Server.
  - Start CICS Transaction Gateway.
  - Start the ITCAM for CICS Transaction Agent.
  - Start the ITCAM for RTT Management Agent.
  - Start the ITCAM for WebSphere Data Collector.
  - Start the OMEGAMON for CICS Agent.

Import the project interchange file

The importing of the sample application project interchange file into IBM Rational Application Developer is optional. This section describes the steps required to import the 4233PI.zip project interchange file into a clean IBM Rational Application Developer workspace if you want to run the application within the test environment or modify the application for your needs.

Development environment prerequisites

We used IBM Rational Application Developer V6.0.1.1 to develop and package the sample application.

Import project interchange file

To import the ITSOBank.zip project interchange file into a clean Rational Application Developer workspace, do the following:

1. Start Rational Application Developer and open the J2EE perspective.
2. Select **File** → **Import**.
3. Select **Project Interchange** and click **Next**.
4. When the Import Projects dialog appears, enter `c:\4233code\src\4233PI.zip` in the From zip file field and click **Select All**.
5. Click **Finish**.

Add the CICS ECI Resource Adapter

If you plan on using the part of the application that connects to CICS, you will need to add the cicseci.rar to the project workspace.

1. Create a new J2EE connector project named cicseci.
   a. From the J2EE perspective, select **File** → **New** → **Other**.
   b. Click **Show all wizards**.
   c. Expand **J2EE**. Select **Connector Project** and click **Next**.
   d. Enter `cicseci` in the Name field.
   e. Uncheck **Add module to an EAR project** and click **Finish**.
2. Import the cicseci.rar to the cicseci project.
   a. Right-click the **cicseci** project, and select **Import** → **RAR file**.
   b. When the Connector Import dialog appears, do the following:
      - Connector file: `C:\Program Files\IBM\Rational\SDP\6.0\Resource Adapters\cics15\cicseci602.rar`.
      - Connector project: select **cicseci**.
      - Check **Overwrite existing resources without warning**.
      - Click **Finish**.
   c. Right-click the **ITSOBankJCAEJB** project and select **Properties**.
   d. Click **Java Build Path**.
   e. Check **cicseci** on the **Projects** tab, and click **OK**.
   f. Perform a clean build by selecting **Project** → **Clean**. Select **Clean all projects** and click **OK**.

Add the ARM libraries

Copy the ARM libraries from the ITCAM for RTT node; armjini4.jar, armjini.jar from your itcam/RTT/MS/scripts/cli/lib directory to your RAD development machine.

1. Right-click project **ITSOBankWebPROJ** and select **Properties**.
2. Click **Java Build Path** and then click the **Libraries** tab.
3. Click **Add external jars** and select the location to include the jar files: armjini4.jar; armjini.jar. Click **OK.**
4. Right-click project **ITSOBankJCAEJB** and select **Properties**.
5. Click **Java Build Path** and then click the **Libraries** tab.
6. Click **Add external jars** and select the location to include the jar file armjini4.jar. Click **OK.**

Note: At the time of writing, we found that we had to restart Rational Application Developer, and perform a Clean build for the errors to go away.

Build EJB deployment code

After importing the project, use the following steps to build the generated EJB deployment code:

1. Expand **EJB Projects**, right-click **ItsoBank5SessionEJB** and select **Deploy**.
2. Repeat the previous step for **ItsoBankCmpEJB**, and **ITSOBankJCAEJB**.

Type mismatch and serializable errors work around

To address the type mismatch errors after import, do the following:

1. Right-click the red error icon, and select **Quick Fi**x for each of the errors.
2. Select **Change code to return long instead of string**. Save and close the opened source file. You will need to do this 3 times.
3. Modify serializable error to select cast argument `arg0` to `java.io.Serializable`.
4. Select **Project** → **Clean** to rebuild the project to resolve the errors.

Add projects to run on server

In order to run the project within the Rational Application Developer - WebSphere Application Server Test Environment, you will need to add the projects to run on the server.

1. From the J2EE Perspective, select the **Servers** view.
2. Right-click **WebSphere Application Server v6.0** and select **Add and remove projects.**
3. Click **Add All** and click **Finish**.
4. Restart the Unit test server.

Create application database

Refer to “Deploy the ITSO Bank DB2 UDB database” on page 115.

Create the datasource

To create the application datasource, do the following:

1. Right-click **WebSphere Application Server v6.0** from the Servers view, and click **Start**.
2. Right-click **WebSphere Application Server v6.0** and select **Run administrative console**.
3. Refer to “Create the ITSO Bank application datasource” on page 116 for the remaining steps to create the datasource.

B

Using the ITCAM for RTT ARM feature

At the time of writing this redpaper, WebSphere Application Server does not support Performance Monitoring for the Java Server Faces (JSF) applications. Since the sample Web service client application code is written using JSF, this chapter summarizes the changes we added to allow ITCAM for RTT to report performance statistics for our JSF-based Web service client application.

Introduction to ARM

ITCAM for RTT and WebSphere Application servers support the ARM open standard. As a result, we added ARM instrumentation code to the IndexCICS.java filename. Classes defined in this file are invoked from the indexCICS.jsp. Since sufficient documentation exists on ARM, we do not provide details.

For additional information on ARM, refer to the following URL:

http://publib.boulder.ibm.com/tividd/td/ITMFTP/SC32-9412-00/en_US/HTML/arm14.htm

Implementation

In the sample application included with this redpaper, we wanted the end-to-end response time correlated across multiple servers for the transactions using the CICS data source. In this section we provide a brief overview of what happens behind the scenes during a Customer lookup using CICS:

1. The IndexCICS.jsp file is displayed from WebSphere Application Server when a user selects the CICS source.
2. A user selects a customer ID and then clicks the submit button. Clicking the submit button invokes the “doButton1Action” contained in IndexCICS.java. This is the method to which we added the ARM functions listed in Example B-1.
3. The following are the important keys to our modifications:
   - In line #1, the correlator is generated and retrieved via the ARM classes
   - In line #2, a singleton type class is created (only 1 instance of the object is allowed). This code is in the Project: ITSOBankWebPROJ, in com.soa1.data.SingleCorrelatorObject.java. This class is used to pass the correlator to the SOAP Handler, which modifies the SOAP Header, when the WebService is called from the Client.
   - In line #3, the ARM token is saved in the SingleCorrelatorObject.

Example B-1: Example snippet from the doButton1Action method

```
This is an example:
public String doButton1Action() {
int armStatus = ArmConstants.STATUS_GOOD;
byte[] corr = new byte[0];

Correlator newCorrelator;
armOn = true;
System.out.println("ARM tranFactory name: " + armTranFactory);
      System.out.println("ARM enabled: " + armOn);

         // ARM instrumentation init environment

try {
         if (armOn) {

            System.out.println("Detected ARM Instrumentation Enabled
in: " + this.getClass());

            armAppDef = armTranFactory.newArmApplicationDefinition(
                 "ITSOBankClientWeb",null,null);

            armTranDef =
armTranFactory.newArmTransactionDefinition(armAppDef,
                 "IndexCICS_Class",null,null);


         armClient =
armTranFactory.newArmApplication(armAppDef,"ITSO","Raleigh",null);
         armTran =
armTranFactory.newArmTransaction(armClient,armTranDef);
         armTran.start();
1) soapCorr = armTran.getCorrelator();
            /*
         System.out.println("Correlator error: " +
soapCorr.getErrorCode());
         System.out.println("Correlator message: " +
soapCorr.getErrorMessage(soapCorr.getErrorCode()));
         System.out.println("Correlator length is: " +
soapCorr.getLength());
         */
         // newCorrelator = new Correlator();
         corrHexString  = Correlator.toHexString(soapCorr.getBytes());
         System.out.println("Transaction started with
correlatorid:(HEX) " + corrHexString);

         2) SingleCorrelatorObject scoorobject =
SingleCorrelatorObject.getSingleCorrelatorObject();
         3) scoorobject.setCorrelator(corrHexString);
         System.out.println("Correlator stored in Cache: " +
scoorobject.getCorrelator());
         }
if (armOn) {
armTran.stop(armStatus);
armClient.end();

}
      } catch (Exception e) {
         armStatus = ArmConstants.STATUS_FAILED;
         System.out.println(this.getClass() + " :ARM Initialization
Failed");
         logException(e);
         return (e.getMessage());
      }
```

SOAP Client Handler usage

A SOAP Client Handler is used to modify the SOAP header with the ARM correlator token. Passing the ARM token in this manner will propagate the token into the WebSphere Application Server Web Services container, resulting in a seamless total end-to-end correlation when using ITCAM for RTT for transaction reporting.

Example B-2 lists a code snippet from SOAPHandler.java. We have added code to the handleRequest method. The code is self-explanatory. It retrieves the correlator stored in the singleton class, and modifies the SOAP header prior to invoking the Web Service with the modified header.

Example B-2: Example snippet from SOAPHandler.java

```
   public boolean handleRequest(MessageContext mc) {

      SingleCorrelatorObject scoorobject
SingleCorrelatorObject.getSingleCorrelatorObject();
      corrHexString = scoorobject.getCorrelator();
      System.out.println("SOAPClientHandler::SOAP Correlator: " + corrHexString);

      addCorrelator((SOAPMessageContext)mc);
      return true;
   }
```

Deployment of SOAP handler

Our handler is contained in Project: ITSOBankWebPROJ; in java package: itso.test.handler; which is in filename: SOAPclientHandler.java. You specify the handler class file in the Web Services client descriptor file (Figure 5-3).

1. To open up the handlers view, click the **Handlers** view under project ITSOWebPROJ. This will open the Web Deployment descriptor for this project.
2. Verify your settings are similar to ours, which are in Figure 5-3.
   a. In Service reference box, select **service/getCustInfo3Service**.
   b. Enter any Description name and Display name: `TestHandler`.
   c. Add Handler Name and class information.
   d. Save deployment descriptor.

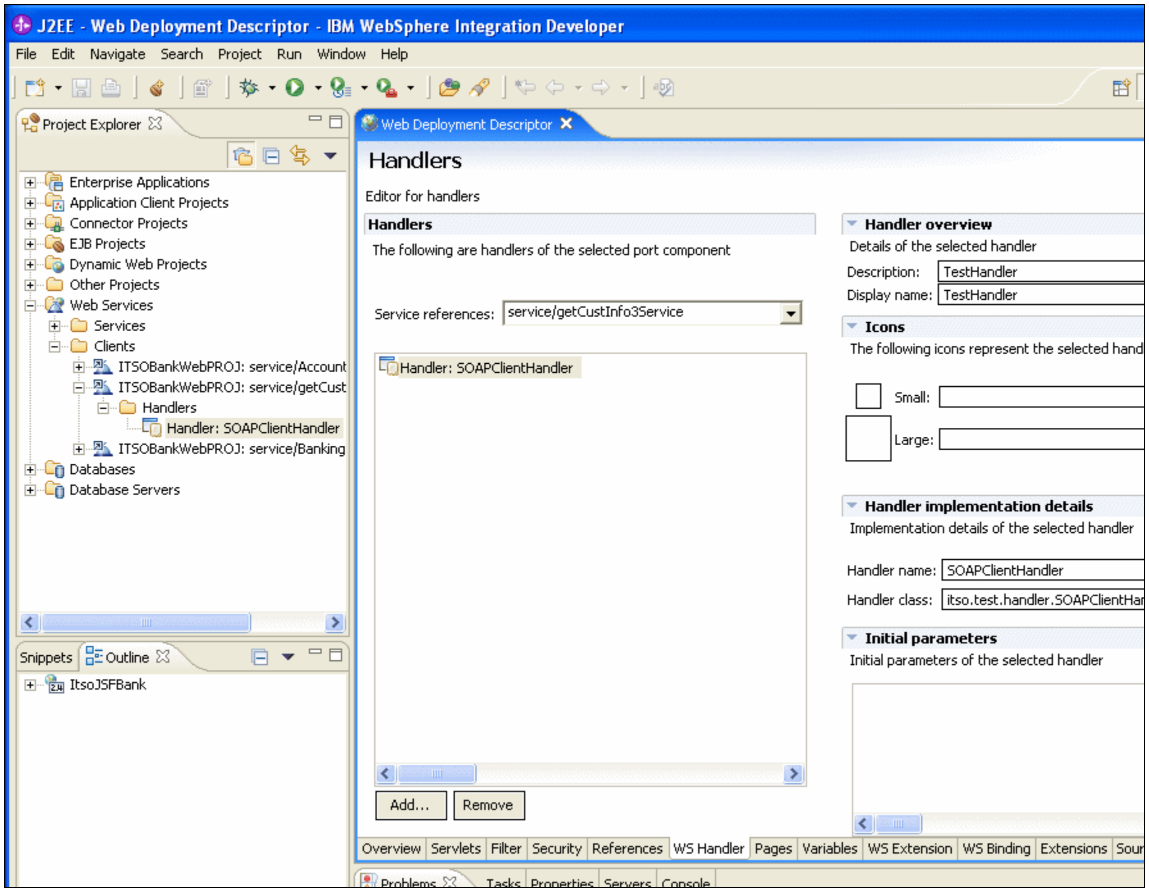

J2EE - Web Deployment Descriptor - IBM WebSphere Integration Developer
File Edit Navigate Search Project Run Window Help
Project Explorer
Enterprise Applications
Application Client Projects
Connector Projects
EJB Projects
Dynamic Web Projects
Other Projects
Web Services
Services
Clients
ITSOBankWebPROJ: service/Account
ITSOBankWebPROJ: service/getCust
Handlers
Handler: SOAPClientHandler
ITSOBankWebPROJ: service/Banking
Databases
Database Servers
Snippets
Outline
ItsoJSFBank
Web Deployment Descriptor
Handlers
Editor for handlers
Handlers
The following are handlers of the selected port component
Service references: service/getCustInfo3Service
Handler: SOAPClientHandler
Add...
Remove
Handler overview
Details of the selected handler
Description: TestHandler
Display name: TestHandler
Icons
The following icons represent the selected hand
Small:
Large:
Handler implementation details
Implementation details of the selected handler
Handler name: SOAPClientHandler
Handler class: itso.test.handler.SOAPClientHar
Initial parameters
Initial parameters of the selected handler
Overview Servlets Filter Security References WS Handler Pages Variables WS Extension WS Binding Extensions Sour
Problems Tasks Properties Servers Console

Figure 5-3 Example of setting a Web Services client handler

Testing SOAP handler

To test the handler, deploy the application ear file: ITSOBankWSEAR from the admin console, and open a browser to point to the following URL:

```
http://<hostname>:9080/ItsoJSFBank/faces/ItsoBank.jsp
```

1. Select **CICS** as the source of data, and click **Submit**.
2. Click **Submit**.
3. Select a Customer ID, and click **Submit**.
4. Data should be returned. Verify that you receive the messages in the WebSphere Application Server SystemOut.log (see Example B-3).

Example B-3: Example WebSphere Application Server SystemOut.log

```
ARM tranFactory name:
com.ibm.tivoli.transperf.arm4.transaction.Arm40TransactionFactory@5226806c
[1/17/07 17:15:43:445 EST] 00000031 SystemOut     O ARM enabled: true
[1/17/07 17:15:43:445 EST] 00000031 SystemOut     O Detected ARM Instrumentation
Enabled in: class pagecode.IndexCICS
[1/17/07 17:15:43:523 EST] 00000031 SystemOut     O Transaction started with
correlatorid:(HEX)
0072CC20030000000000000000000000111111111111111000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000100000000
[1/17/07 17:15:43:539 EST] 00000031 SystemOut     O Correlator stored in Cache:
0072CC20030000000000000000000000111111111111111000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000100000000
[1/17/07 17:15:44:945 EST] 00000031 SystemOut     O loading arm4 transaction class
[1/17/07 17:15:44:945 EST] 00000031 SystemOut     O errCode = 0
[1/17/07 17:15:45:586 EST] 00000031 SystemOut     O SOAPClientHandler::SOAP
Correlator:
0072CC20030000000000000000000000111111111111111000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000100000000
[1/17/07 17:15:45:617 EST] 00000031 SystemOut     O SOAPClientHandler::Start of SOAP
Header.
[1/17/07 17:15:45:632 EST] 00000031 SystemOut     O <soapenv:Header
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><reqmetrics:correlator
soapenv:actor="reqmetricsURI"
xmlns:reqmetrics="http://websphere.ibm.com">0072CC20030000000000000000000000111111111
1111110000000000000000000000000000000000000000000000000000000000000000000000000000000
```

```
0000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000100000000</reqmetrics:correlator></soapenv:Header>
[1/17/07 17:15:45:632 EST] 00000031 SystemOut     O SOAPClientHandler::End of SOAP
Header.
```

5. Following the verification of the correlator generation, you may want to remove these statements in your code.

C

Additional material

This redpaper refers to additional material that can be downloaded from the Internet as described here.

Locating the Web material

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

`ftp://www.redbooks.ibm.com/redbooks/REDP4233`

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

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redpaper form number, REDP4233.

Using the Web material

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

File name	*Description*
4233code.zip	Zipped sample application, and TEP workspaces and situations.

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	10 MB minimum
Operating System:	Windows or Linux
Processor:	1 GHz PIII or higher
Memory:	512 MB

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder. After unzipping 4233code.zip, you will have a directory structure as seen in Table C-1.

Table C-1 Directory structure of unpacked 4233code.zip

Directory	Description
c:\4233code	Root directory
c:\4233code\deploy	Contains deployable EAR files: * Web Service Client application (ITSOBankWSWeb.ear) * Web Service DB2 application (ITSOBankDB2WS.ear) * Web Service for CICS application (ITSOBankJCAWS.ear)
c:\4233code\deploy\db2	DB2 UDB ITSO Bank application database scripts.
c:\4233code\src	4233PI.zip Project Interchange file containing source code.
c:\4233code\tep\situations	Contains the sample TEP situations.

Related publications

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

IBM Redbooks

For information about ordering these publications, see "How to get IBM Redbooks" on page 142. Note that some of the documents referenced here may be available in softcopy only.

- *Patterns: SOA Foundation - Service Creation Scenario*, SG24-7240
- *Patterns: SOA Foundation - Business Process Management Scenario*, SG24-7234
- *IBM Tivoli Composite Application Manager V6.0 family*, SG24-7151
- *IBM Tivoli OMEGAMON XE V3.1.0 Deep Dive on z/OS*, SG24-7155
- *CICS Transaction Gateway for z/OS Version 6.1*, SG24-7161

Other publications

These publications are also relevant as further information sources:

- *Install Guide, IBM Tivoli Composite Application Manager V6.0 for SOA*, GC01-4444
- *Getting Started, IBM Tivoli Service Level Advisor V2.1.1*, SC32-0834-04
- *Secure and manage services in a SOA Environment to achieve business objectives*, GC28-8392-00
- *IBM Tivoli Monitoring for Transaction Performance, Application Response Measurement (ARM) Instrumentation Guide*, SC32-9412-00

Online resources

These Web sites are also relevant as further information sources:

- *IBM Tivoli Composite Application Manager InfoCenter* found at:

  http://publib.boulder.ibm.com/infocenter/tivihelp/v3r1/index.jsp?topic=/com.ibm.tivoli.az.doc/welcome.htm

- IBM IT Service Management:

  http://www.ibm.com/software/tivoli/features/it-serv-mgmt/

- *IBM Tivoli Monitoring for Transaction Performance, Application Response Measurement (ARM) Instrumentation Guide*, SC32-9412-00 found at:

  http://publib.boulder.ibm.com/tividd/td/ITMFTP/SC32-9412-00/en_US/HTML/arm.htm

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

Best Practices for SOA Management

Redpaper

Guidelines for defining non-functional requirements and SLAs

Best practices for identifying resouces to be managed

Build an integrated TEP console to manage resources

SOA management provides best practices and software for managing and monitoring SOA composite applications and supporting infrastructure.

This IBM Redpaper focuses on the following aspects of SOA management:

- Introduction to the key concepts of SOA management.
- Best practices for defining non-functional requirements and SLAs.
- Guidance on how to identify what resources should be managed at the time of the solution analysis and design, and by discovery in the runtime.
- How to build an integrated Tivoli Monitoring Server - Enterprise Portal (TEP) console to manage and monitor resources for SOA composite applications using ITCAM products.
- How to verify the monitoring functionality by performing an application walkthrough and triggering the monitoring conditions.
- How to create an SLA and report in Tivoli Service Level Advisor.