IBM

Patterns: SOA Foundation Service Creation Scenario

- Key concepts and architecture of the IBM SOA Foundation
- Process for identifying SOA scenarios, patterns and services
- Service Creation scenario working examples

John Ganci
Amit Acharya
Jonathan Adams
Paula Diaz de Eusebio
Gurdeep Rahi
Diane Strachan
Kanako Utsumi
Noritoshi Washio

Redbooks

ibm.com/redbooks

International Technical Support Organization

Patterns: SOA Foundation Service Creation Scenario

September 2006

SG24-7240-00

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

First Edition (September 2006)

This edition applies to IBM Rational Application Developer V6.0.1 on Microsoft Windows XP, and IBM WebSphere Application Server Network Deployment V6.0.2 and IBM Tivoli Composite Application Manager for SOA V6.0 on Microsoft Windows 2003 Server.

Contents

Notices

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

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

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

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

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

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

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

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

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

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

Trademarks

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

AIX®
Ascendant®
BladeCenter®
Candle®
CICS®
ClearCase®
Cloudscape™
Component Business Model™
DataPower®
DataStage®
DB2 Universal Database™
DB2®
developerWorks®
Domino®
@server®
@server®
Everyplace®
IBM®
IMS™
Informix®
Iterations®
NetView®
OMEGAMON Monitoring Agent®
OMEGAMON®
RACF®
Rational Developer Network®
Rational Process Workbench®
Rational Unified Process®
Rational®
RDN™
Redbooks (logo) ™
Redbooks™
RUP®
Tivoli Enterprise Console®
Tivoli Enterprise™
Tivoli®
WebSphere®
Workplace Forms™
Workplace™
xSeries®
z/OS®
zSeries®

The following terms are trademarks of other companies:

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

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

Intel, Pentium, Xeon, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

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

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

Preface

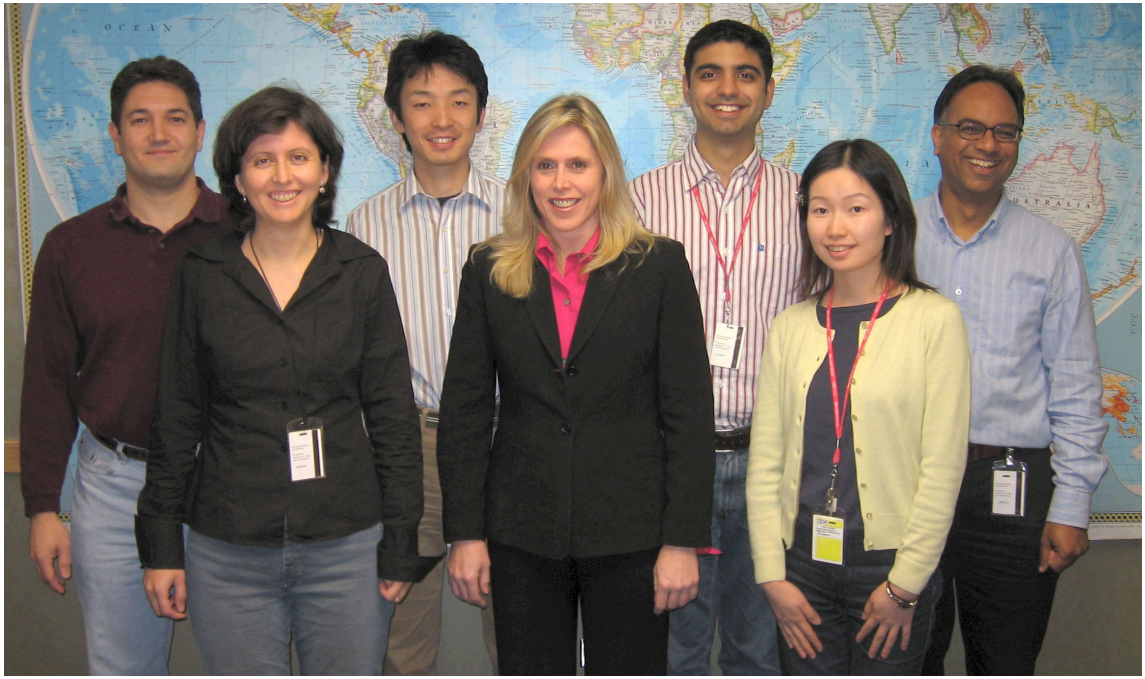
This IBM® Redbook provides an introduction to the IBM SOA Foundation, and includes a detailed implementation example for the Service Creation scenario.

Part 1, “Getting started with IBM SOA Foundation” on page 1, includes an introduction to SOA from a business and architecture perspective. We describe the key elements of the IBM SOA Foundation, including the SOA life cycle, logical architecture, and SOA scenarios. Next we describe the Service Creation scenario in more detail since this is the focus of redbook example. We describe best practices and guidelines for SOA. In addition, we include a process for applying the SOA scenarios and patterns.

Part 2, “Service Creation scenario example” on page 171, provides an end-to-end working example representative of the Service Creation scenario. We start by modeling the business requirements of a fictitious car rental company that has an existing J2EE™ based application. We demonstrate how to identify services using SOMA, and use a process for applying the SOA scenarios and reusable patterns to accelerate the creation of a solution architecture. The remaining chapters provide the implementation details to assemble using Rational® Application Developer V6.0.1, deploy using WebSphere® Application Server Network Deployment V6.0.2, and manage the solution using Tivoli® Composite Application Manager for SOA V6.0.

The team that wrote this redbook

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



The IBM Redbook team left to right: John, Paula, Nori, Diane, Amit, Kanako, Gurdeep, (Jonathan was not present for the photo)

John Ganci is a Solution Architect with the IBM SWG Scenario Analysis Lab located in IBM RTP, North Carolina. John has 16 years of experience in application design, development, testing and consulting. His areas of expertise include IBM middle ware integration, SOA, WebSphere Application Server, WebSphere Portal, e-commerce, pervasive computing, J2EE programming, and technical team leadership.

Amit Acharya is a Staff Software Engineer with the WebSphere Quality Center of Competence Organization at IBM RTP, North Carolina. He has four years of experience in Enterprise Application Development (EAD) using J2EE, Web services, Rational tools, WebSphere Studio and so forth. His areas of expertise include test strategy planning and execution of Web services for WebSphere Application Server, technical and team leadership for Web services, simulating customer environments for latest WebSphere Application Server releases. He has actively contributed to the IBM patent portfolio and is also involved in publishing technical articles and redbooks (SOA/ESB). Earlier, Amit worked with the WSES performance group in the IBM Pittsburgh Lab. He holds a Masters of Science degree in Electrical and Computer Engineering from Purdue University, Indiana USA.

Jonathan Adams is an IBM Distinguished Engineer. He has been an IT Architect with IBM for 36 years. For the past 10 years, he has focused on helping IBM deliver reuseable end-to-end middle ware solutions to its clients. Since September 1998, he has worked in the Software Group Technical Strategy organization leading the definition and development of the Patterns for e-business. These patterns have been built by teaming across all major IBM divisions. The resultant patterns are being used by IBMers, clients, and IBM Business Partners to help reduce risk and increase speed to market on many e-business solution developments.

Paula Diaz de Eusebio is an advisory IT Architect within BCS Application Services in IBM Spain. She holds an Engineering Degree in Telecommunications from the Universidad Politécnica of Madrid. She has been working at IBM for six years, with experience on the design and development of e-business solutions, and now focuses on SOA and EAI.

Gurdeep Rahi is an IBM Certified Consulting IT Specialist working in Technical Sales in IBM Software Business. Since joining IBM in 2000, he has specialized in supporting the WebSphere platform, in particular WebSphere Application Server and tools. He has worked with WebSphere customers in the UK and joined the Channel Technical team to support IBM Business Partners. Gurdeep has over 20 years of experience in the IT industry with various roles in technical capacity outside of IBM, including software development, support and presales consultancy. His background is in operating systems, programming, databases, networking and distributed technologies such as DCE, CORBA and Java™.

Diane Strachan is a Certified Senior IT Specialist with Global Business Services, SOA and Web Services practice in Canada. Diane has over 25 years of industry experience specializing in custom enterprise application development using object technology across various sectors including telecommunications, financial markets, banking, insurance, government and retail. Diane's areas of expertise include object-oriented analysis and design and application architecture. She has knowledge and experience in all phases of the software development life cycle. Her most recent focus is business transformation and application of SOMA techniques.

Kanako Utsumi is an IT Specialist at IBM Systems Engineering Co. Ltd (ISE), part of the ATS function in Japan. She has five years of experience in designing and implementing Web infrastructure, and developing Web applications associated with Java and DB2® UDB. She provides technical support for WebSphere products. Her areas of expertise include J2EE applications, WebSphere Application Server, and Edge components.

Noritoshi Washio is a Software Engineer for Hitachi Software Engineering in Tokyo, Japan, that is a business partner with IBM. He has six years of experience in J2EE technologies and WebSphere products for e-commerce,

help desk system, and bioinformatics solutions. Recently he has worked at IBM WebSphere Technologies Institute in Raleigh for a year as a WebSphere trainee. His most recent work was development of a SOA benchmark application with WebSphere Integration Developer. He holds a Master of Science in Geophysics from Tohoku University in Japan.

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

Erica Carmel
IBM WebSphere Software Platform User Experience
Kareem Yusuf
IBM Director, SOA Product Management
IBM USA

Michael Ellis
IBM Software Services for WebSphere
IBM Canada

Stephen Hurst
IBM UK

Martin Keen
Carla Sadtler
ITSO Raleigh Center
Rachel Reinitz
IBM Application and Integration Middleware Software and
Web Services Consulting
IBM USA

Adeola Allison
Ali Arsanjani
JB Baker
Don Carr
Sudhakar Chellam
Larry Deppa
Steve Graham
John Harter
Hector Hernandez
Rajesh Lalgowdar
Shweta Mathur
Sridhar Muppidi
Gareth Patterson
Franco Potepan
Rosalind Radcliffe
IBM USA

Become a published author

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

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

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

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

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

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

  ibm.com/redbooks

- Send your comments in an email to:

  redbook@us.ibm.com

- Mail your comments to:

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

Part 1

Getting started with IBM SOA Foundation

1

Introduction to service-oriented architecture

This chapter introduces service-oriented architecture (SOA) from a business and architecture perspective.

The chapter is organized into the following sections:

- Service-oriented architecture overview
- Getting started with SOA
- Web services and SOA
- Target audience of this book

1.1 Service-oriented architecture overview

This section includes an overview for a service-oriented architecture (SOA). First, we define the key terms and components used to describe an SOA. Second, we review the key challenges and drivers for SOA. Third, we highlight the reasons why SOA is the right choice now. Lastly, we describe an example scenario for building a solution using an SOA approach.

1.1.1 Definition of a service-oriented architecture

Figure 1-1 highlights the key terms used to describe a service-oriented architecture.

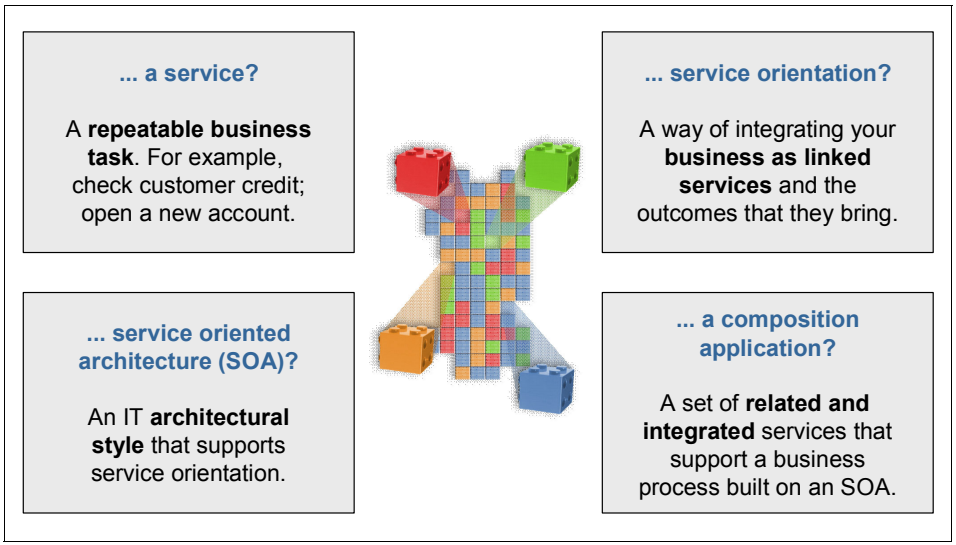


Figure 1-1 Definition of key terms for a service -oriented architecture

A *service* is representative of a repeatable business task. Services are used to encapsulate the functional units of an application by providing an interface that is well defined and implementation independent. Services can be invoked (consumed) by other services or client applications.

Service orientation defines a method of integrating business applications and processes as linked services.

Service-oriented architecture (SOA) can be different things to different people depending on the persons role and context (business, architecture, implementation, operational). From a business perspective, SOA defines a set of business services composed to capture the business design that the enterprise

wants to expose internally, as well as its customers and partners. From an architecture perspective, SOA is an architectural style that supports service orientation. At an implementation level, SOA is fulfilled using a standards based infrastructure, programming model and technologies such as Web services. From an operational perspective, SOA includes a set of agreements between service consumers and providers that specify the quality of service, as well as reporting on the key business and IT metrics.

A *composite application* is a set of related and integrated services that support a business process built on an SOA.

Basic components of an SOA

At the most basic level, an SOA consists of the following three components:

- Service provider
- Service consumer
- Service registry

Each component can also act as one of the two other components. For instance, if a service provider needs additional information that it can only acquire from another service, it acts as a service consumer. Figure 1-2 shows the operations each component can perform.

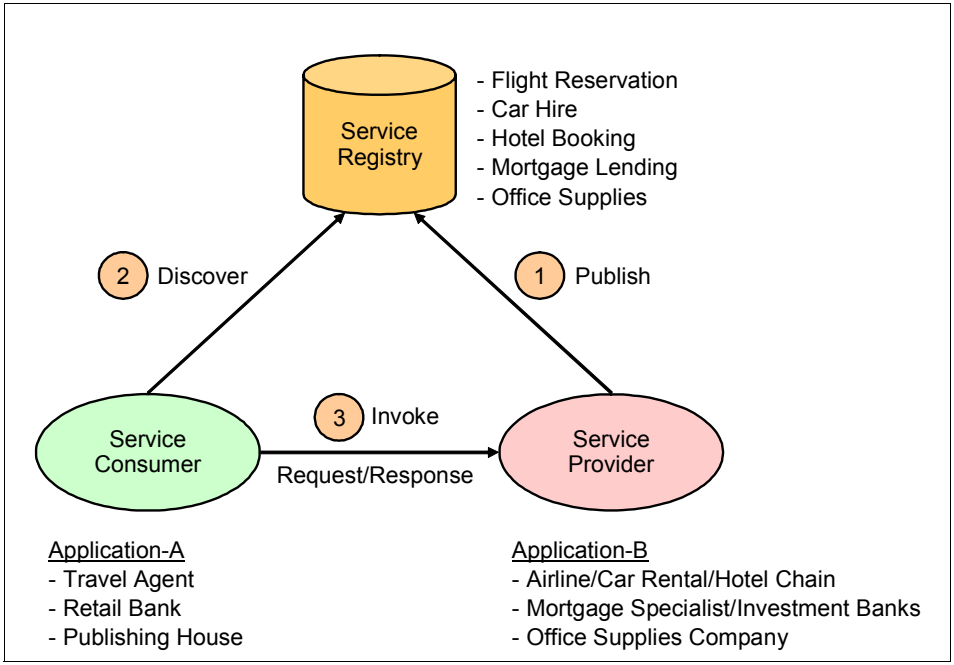


Figure 1-2 SOA components and operations

The *service provider* creates a service and in some cases publishes its interface and access information to a service registry.

Each provider must decide which services to expose, evaluate trade-offs between security and easy availability, determine how to price the services or determine how to exploit the value of the services if they are free. The provider also has to decide in which category the service should be listed, and what sort of trading partner agreements are required to use the service.

The *service registry* is responsible for making the service interface and implementation access information available to service consumers.

The implementers of a service registry must consider the scope with which the registry will be implemented. For example, there are public service registries available over the Internet to an unrestricted audience, as well as private service registries that are only accessible to users within a company-wide intranet.

The *service consumer* locates (discovers) entries in the service registry and then binds to the service provider in order to invoke the defined service.

1.1.2 Challenges and drivers for SOA

In March 2006, IBM commissioned a Global CEO survey and found that 78% of CEOs surveyed believe that integrating business and technology is fundamental for innovation. Another key finding from this survey was that only one in ten CEOs believes his or her organization has the ability to be very responsive to changing market conditions.

As noted from the survey, businesses need the ability to integrate business and technology rapidly to achieve their business objectives. Businesses also have a strong desire to leverage the investment of existing business applications and systems without a complete and costly rewrite. There are many schemes that exist today to integrate systems within and between enterprises. In most cases these solutions are proprietary, not easily adaptable, and not responsive to rapid changes needed by the business.

There is a growing demand for an architecture and technologies that support the connection or sharing of resources and data in a very flexible and industry-standard manner. There is a need to further structure large applications into building blocks that can be reused and composed into business processes.

A shift towards a service-oriented approach standardizes the interaction with applications and business processes, and allows for more flexibility in the process. By adopting an SOA approach, existing application functionality can be turned into reusable services that can be consumed by a new set of client

applications and users. SOA brings the flexibility vital to realizing innovation and desired outcomes of business.

Tip: The alignment of IT with business goals can be summarized as collaborative business and IT decision-making that ensures the following:

- IT investments are made based on business objectives.
- IT service delivery provides a business result.
- Business priorities are assessed with IT capabilities and limitations in mind.

Business requirements and drivers for SOA

Figure 1-3 highlights the common elements of a business that require flexible integration.

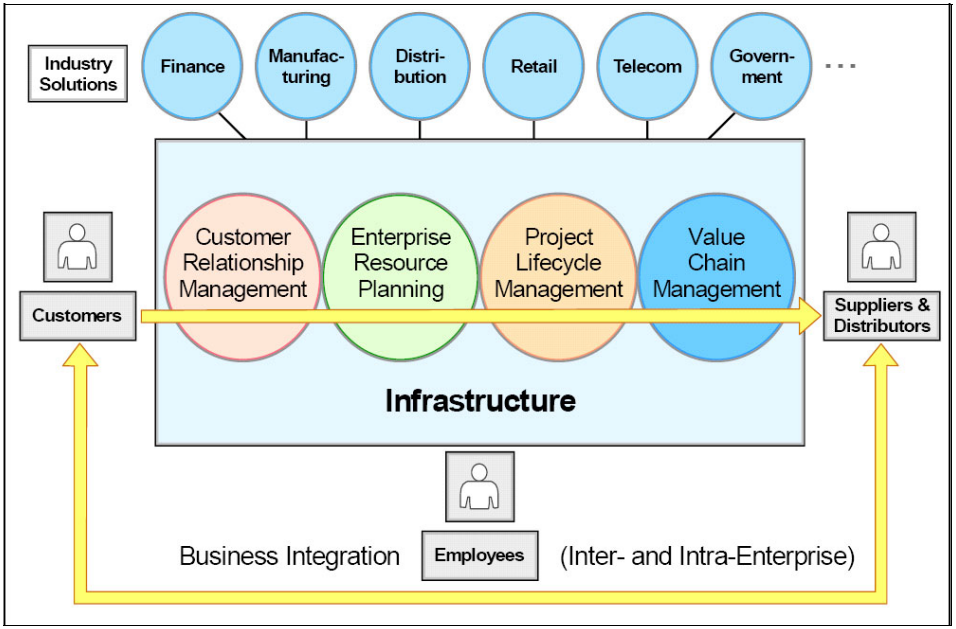


Figure 1-3 Business requirements

Here, we summarize the common business drivers that require rapid and flexible integration of IT systems:

- Support an agile business model

  The marketplace can be very dynamic and competitive. There is a great need to have a business model and IT architecture that can rapidly change to support the business model and its objectives.

- Reduce cycle time and costs

  Eliminate duplicate systems by reusing existing applications. This has the effect of reducing the time required to integrate systems, reduces cost, and simplifies the skill set required to implement the solution.

  When applying these concepts to external business processes, enterprises can move from costly manual transactions to automated transactions with suppliers.

- Simplify integration across the enterprise

  Many existing IT systems can be inhibitors to change. They are too complex and, as a result, inflexible. Also, existing integration includes multiple technologies and point-to-point integration, which is often inflexible. The need to simplify integration is essential, especially considering the challenges raised from events such as business mergers and acquisitions.

- Achieve better IT use and return on investment

  Return on investment (ROI) is a comparison of profit earned or lost for the investment with the amount invested. The investment in IT should facilitate the business objective and help the business achieve the targeted ROI.

Greater need for a flexible architecture

There are many possible reasons that a flexible business model is needed, such as business transformation, business process outsourcing, mergers, and acquisitions. SOA provides a flexible IT infrastructure and on demand operating environment to support the initiatives of a flexible business model.

For the purposes of comparison with SOA, we highlight the integration deficiencies of monolithic (silos) and component-based architectures. Next, we describe the flexibility gained by using an SOA approach.

Historically, business applications were built with a monolithic purpose (silos). 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 (custom coded connections required).

For example, a monolithic business application must periodically synchronize inventory information, as you can see in Figure 1-4. In this approach, pricing information for each Web order is inserted differently based on the application structure. Lastly, there is no common customer or inventory database to be shared across the enterprise, or flexibility in the business processes.

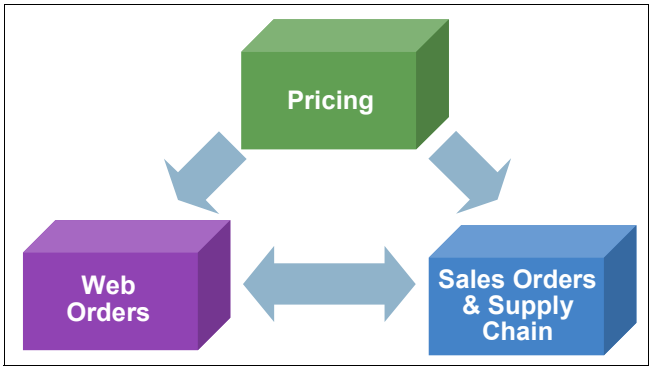


Figure 1-4 Monolithic business application (silos)

Although component-based application architecture does define services as units of business logic, there are some inherit problems with this approach. The flow of control is bound into the service logic. The transformation of data formats is also bound to the service logic. There is tight coupling between the services as seen in Figure 1-5, thus making this application integration architecture fragile, giving it a spaghetti-like appearance.

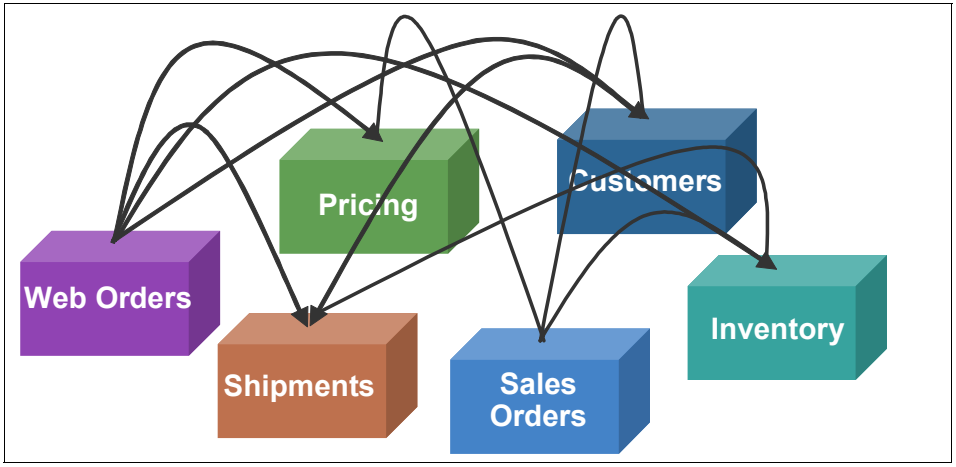


Figure 1-5 Component based application

When using an SOA approach (see Figure 1-6), the services are defined as units of business logic separated from the flow of control and routing, and the data transformation and protocol transformation. This approach provides loose coupling, thus making this approach much more flexible for integration.

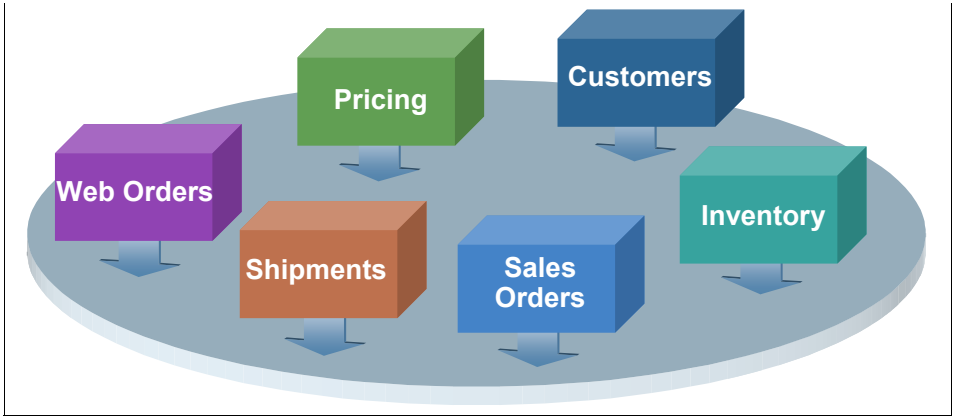


Figure 1-6 SOA-based application

1.1.3 Why SOA now?

In the previous section, we explained how a service-oriented architecture provides the flexibility to align you IT with your business goals. In this section, we explain why SOA is the right choice now. When there is a shift in architecture, it is important to understand why a shift is needed, and evaluate the maturity level of the architecture that supports adoption.

We highlight the following key reasons why SOA is the right choice now:

- Business driving a shift in IT
- Enables flexibility of both IT and business
- Open standards and platforms
- Best practices
- Software for SOA

Business driving a shift in IT

Table 1-1 provides a summary of business needs that are driving a shift in IT from function-oriented to process- and service-oriented to achieve flexibility.

Table 1-1 Shift in IT driven by business

From function-oriented	To process and service-oriented
Build for permanence	Build to change
One long development cycle	Incremental development cycle
Application silos	Orchestrated solutions that work together
Tightly coupled	Loosely coupled

From function-oriented	To process and service-oriented
Structure applications using components and objects	Structure applications using services
Known implementation	Implementation abstraction

Enables flexibility of both IT and business

SOA enables flexibility of both IT and business through flexible connectivity of business services:

- Represent applications or data as a service with a standardized interface.
- Enable applications as services to exchange structured information (messages, documents, and other business objects).
- Mediate the message exchange through an Enterprise Service Bus (ESB).
- Provide on-ramps to the bus for existing applications and systems.

Open standards and platforms

Another key reason that SOA is the right choice for your enterprise, is that it is based on open standards and platforms as summarized in Figure 1-7. These open standards are widely adopted across the industry.

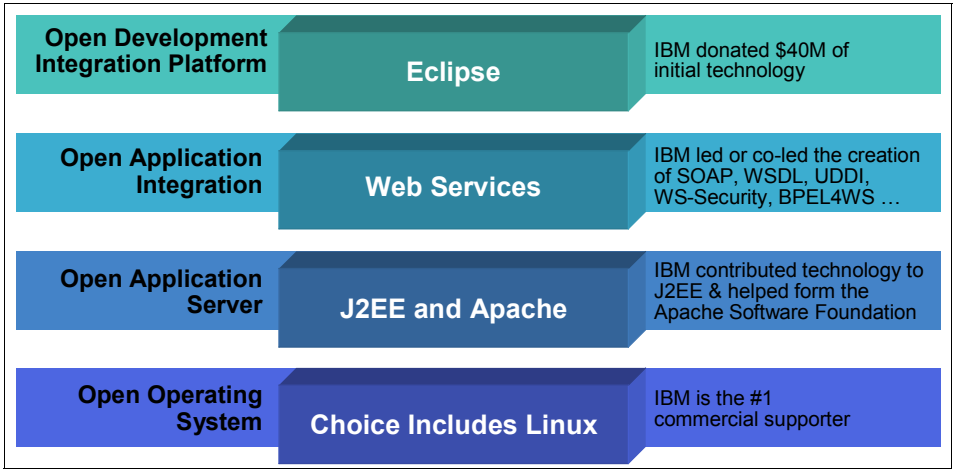


Figure 1-7 Summary of SOA open standards and platforms

IBM continues to be a leader in SOA based technologies, products and solutions. IBM is a key partner in helping define the specifications and technologies used to implement an SOA, such as Web Services, Service Component Architecture (SCA) and Service Data Objects (SDO).

Best practices

Best practices are used to deliver a particular outcome by leveraging the knowledge learned from experience. Best practices include methodologies, techniques, guidelines and patterns. By leveraging the knowledge captured in best practices listed here, your project can be run with less problems and be deployed more rapidly.

Note: Many of the best practices listed in this section are described in greater detail in Chapter 4, “Best practices for SOA” on page 83.

Here is a list of best practices in use today:

- SOA Adoption

  The SOA adoption process provides guidelines that assist in developing a roadmap towards a successful migration to an SOA.

- SOA Governance

  SOA Governance helps clients extend the planned SOA across the enterprise in a controlled manner.

- Methodology

  A well-established set of methodologies can help to break down complex problems into smaller and more manageable pieces that are easier to analyze and, therefore, develop solutions. Example methodologies include the Component Business Model™ (CBM), IBM Service Integration Maturity Model (SIMM), Rational Unified Process® (RUP®), and Service-Oriented Modeling and Architecture (SOMA).

- Process modeling

  Process modeling is used to define business processes. A processes flow is a sequence of tasks and decision elements with multiple branches, linked by connectors.

- Model-driven development

  Model-driven development is a style of software development where the primary software artifacts are models from which code and other artifacts are generated. A model is a description of a system from a particular perspective, omitting irrelevant detail so that the characteristics of interest are more clear.

- Reference architecture

  A reference architecture provides the underlying architecture components used to overcome the initial problems of finding an architecture with which to begin. The most notable reference architecture for SOA is the IBM SOA Foundation.

- Patterns

  As a general principle, starting from the beginning each time should be avoided. The use of *patterns* is one specific form of capturing and reusing reoccurring design elements. For example, the Patterns for e-business include reusable architecture and implementation assets used to accelerate the creation of a solution design and implementation.

Software for SOA

The marketplace offers many software choices for SOA. IBM is the market leader in providing mature software and solutions for SOA. For detailed information on the SOA software and solutions provided by IBM, refer to 2.4, “SOA Foundation scenarios” on page 34.

1.1.4 SOA approach for building a solution

This section includes an example SOA approach for building a solution. In this example, the company wants to implement a new business process to support customers who are placing orders from an Internet Web site.

The company has existing retail, warehouse, and billing systems, as seen in Figure 1-8. The company would like to build new business processes by reusing the functionality provided by the existing systems rather than having to write new applications or new proprietary interfaces to the existing systems.

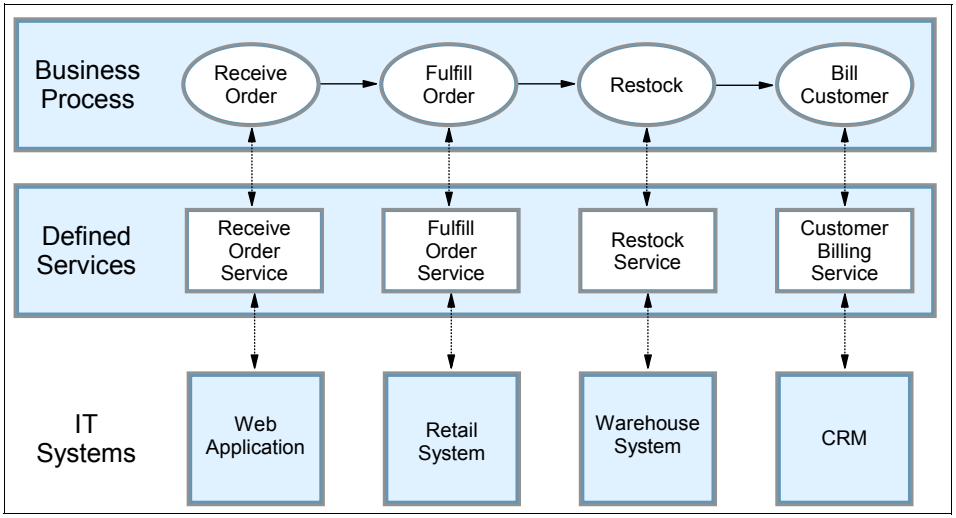


Figure 1-8 Service-oriented approach to building systems

If the company has already adopted an SOA approach, it will have defined the interfaces to its existing systems in terms of the functions or services that they offer in support of building business processes. The defined interfaces make building the new system Web front end very simple. The company simply needs to develop an application that invokes (consumes) services to complete the new business process.

By using an SOA approach, companies are able to build horizontal business processes that integrate systems, people, and processes from across the enterprise quickly and easily in response to changing business needs.

1.2 Getting started with SOA

In this section, we explore the question of how to get started with SOA from both a business and an architectural perspective.

1.2.1 SOA adoption

SOA adoption provides an iterative and incremental process, and guidelines that assist in developing a road map towards a successful migration to SOA. As seen in Figure 1-9, the SOA adoption process begins by defining the scope of possible projects that fit the criteria for being a good fit for a service-oriented architecture.

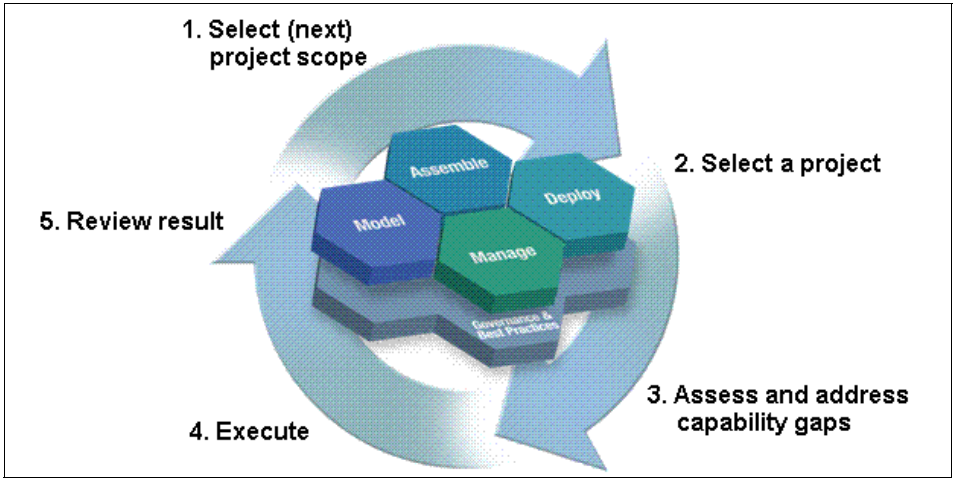


Figure 1-9 SOA adoption process

There are two primary perspectives, including strategic vision and project plan. The *strategic vision* perspective describes the business and IT statement of direction, which can be used as a guideline for decision making, organizational

buy-in, and standards adoption. The *project plan* perspective (or *tactical* perspective) refers to implementation projects to meet immediate needs of the current business drivers.

Defining the strategic vision starts with assessing the business current maturity across multiple dimensions including business, methodology and technical. The IBM Service Integration Maturity Model (SIMM) can be used to help in this assessment. If you are more comfortable with starting with a self assessment, you can use the IBM online SOA Assessment Tool:

http://www.ibm.com/software/solutions/soa/soaassessment/index.html

After the assessment has been performed, the business must establish targets for where they want to be. This includes documenting important goals and metrics for transition across the maturity dimensions. In addition, it is important to have regular checkpoints to reassess the vision.

Note: Refer to 4.2, “SOA Adoption” on page 97 for more detailed information.

1.2.2 IBM SOA entry points

As seen in Figure 1-10, SOA connects people, processes, and information. To help customers get started with SOA, IBM has defined three core business-centric starting points (people, processes, and information), and two IT-centric starting points (connectivity and reuse). These are known as the SOA entry points.

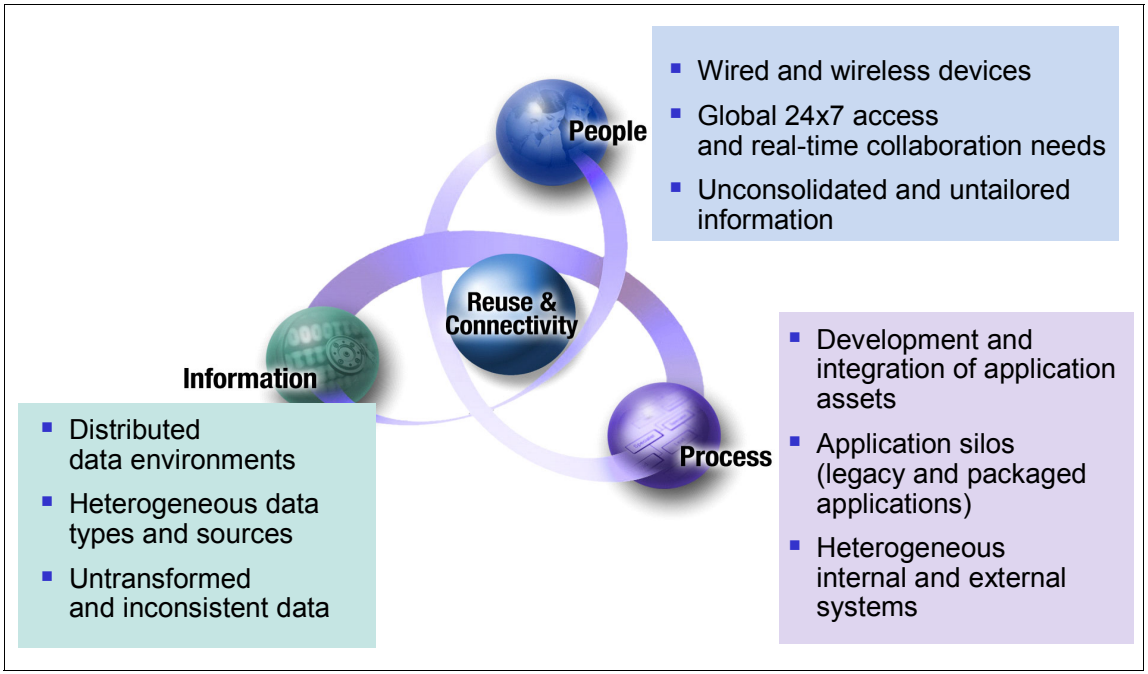


Figure 1-10 SOA connects people, processes, and information

Through business-centric SOA, companies can tie IT projects to their business needs directly by addressing the companies immediate pain points.

- People: productivity through collaboration

  Improve people productivity by aggregating views that deliver information and interaction in the context of a business process. This enables human and process interaction with consistent levels of service.

  Start by building a view of a key business process by aggregating information to help people make better decisions. The next steps are tighter management of performance with alert-driven dashboards that link to more processes.

- Process: business process management for continuos innovation

  Deploy innovative business models quickly with reusable and optimized processes to adapt the enterprise to changing opportunities and threats.

  Start by modeling an under-performing process, remove bottlenecks, then simulate and deploy the optimized process. Next, create flexible linkages between multiple processes across the enterprise and outside the firewall to suppliers and partners. Then, monitor the process to measure and track performance.

- Information: delivering information as a service

  Improve business insight and reduce risk with trusted information services delivered in-line and in context.

  Start by discovering and understanding information sources, relationships and the business context. Next steps are to expand the volume and scope of the information delivered as a service across internal and external processes.

The IT-centric entry points to help the enterprise integrate the business-centric SOA entry points are as follows:

- Connectivity: underlying connectivity to enable business centric SOA

  Connectivity has always been a key requirement. SOA brings new levels of flexibility. As well as acting as a building block for additional SOA initiatives, connectivity provided through SOA has distinct, standalone value.

- Reuse: create flexible, service base business applications

  Cut costs, reduce cycle times and expand access to core applications through reuse. Analysts estimate it is up to five times less expensive to reuse existing applications than to write new applications.

  Use portfolio management to consider which assets you need to run your company. Identify high-value existing IT assets and service-enable them for reuse. Satisfy remaining business needs by creating new services. Finally, create a service registry and repository to provide centralized access and control of these reusable services.

1.2.3 IBM SOA Foundation

The IBM SOA Foundation is an integrated, open standards based set IBM software, best practices and patterns to provide you with the architecture knowledge to get started with SOA. The key elements of the IBM SOA Foundation are the SOA life cycle (model, assemble, deploy, manage), reference architecture, programming model, and SOA 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 quickly communicate the business value, architecture, and IBM open standards-based software used within the SOA scenario. The concept of realizations are used to provide more specific solution patterns and IBM product mappings within the SOA scenarios.

The SOA scenarios can be used as a reference architecture implementation (starting point) to accelerate the SOA architecture and implementation of your scenario. The SOA scenarios can be implemented using an incremental SOA

adoption approach, whereby a customer can incrementally add elements of other SOA scenarios to the environment to achieve their business objectives.

Note: Refer to Chapter 2., “IBM SOA Foundation” on page 25 for more detailed information.

1.3 Web services and SOA

This section describes the core technologies of Web services, as well as how Web services are used to implement an SOA.

Note: For more detailed information on Web services, we recommend you read the *WebSphere Version 6 Web Services Handbook, Development and Deployment*, SG24-6461 redbook.

1.3.1 Web services technologies

Web services technology is a collection of standards (or emerging standards) that can be used to implement an SOA. Web services technology is vendor- and platform-neutral, interoperable, and supported by many vendors today.

Web services are self-contained, modular applications, that can be described, published, located, and invoked over networks. Web services encapsulate business functions, ranging from a simple request-reply to full business process interactions. The services can be new or wrap around existing applications.

Core elements

The following are the core technologies used for Web services.

- Extensible Markup Language (XML)

  XML is the markup language that underlies most of the specifications used for Web services. XML is a generic language that can be used to describe the content in a structured way, separated from its presentation to a specific device.

- Simple Object Access Protocol (SOAP)

  SOAP is a specification for the exchange of structured XML-based messages between the service provider, service consumer, and service registry, consisting of three parts:

  – The format of a SOAP message is an envelope containing zero or more headers and exactly one body. The *envelope* is the top element of the XML document, providing a container for control information, the

addressee of a message, and the message itself. *Headers* contain control information such as quality-of-service attributes. The *body* contains the message identification and its parameters.

- Encoding rules are used for expressing instances of application-defined data types. SOAP defines a programming language independent data type schema based on an XML Schema Descriptor (XSD), plus encoding rules for all data types defined to this model.
- RPC representation is the convention for representing remote procedure calls (RPC) and responses.

SOAP, in principle, is a protocol-independent transport. Consequently, it can potentially be used in combination with a variety of protocols such as HTTP, JMS, SMTP, or FTP. Currently, the most common way of exchanging SOAP messages is through HTTP, which is also the only protocol supported by WS-I Basic Profile 1.0.

- Web Services Description Language (WSDL)

  WSDL is an XML-based interface and implementation description language. A WSDL document contains the following main elements:

- *Types* is the container for data type definitions using a type system such as an XML Schema.
- An abstract, typed definition of the data being communicated, a *message* can have one or more typed parts.
- A *port type* is an abstract set of one or more operations supported by one or more ports.
- An *operation* is an abstract description of an action supported by the service that define the input and output message and optional fault message.
- The *binding* is a concrete protocol and data format specification for a particular port type. The binding information contains the protocol name, the invocation style, a service ID, and the encoding for each operation.
- The *service* as a collection of related ports.
- The *port* is a single endpoint, an aggregation of a binding and a network address.
- Universal Description, Discovery, and Integration (UDDI)

  UDDI is both a client-side API and a SOAP-based server implementation that can be used to store and retrieve information on service providers and Web services.

Standards

Figure 1-11 displays a stacked view of Web services technologies. Most of the technologies displayed have been standardized. Since interoperability is a key goal of Web services, an open industry organization known as the *Web services Interoperability Organization* (WS-I)) has been created to allow interested parties such as IBM and Microsoft® to work together to maximize interoperability between Web services implementations. For more information about WS-I, visit their Web site:

`http://ws-i.org`

Note: Web services standards are evolving at a rapid pace. For the most current information, we recommend that you reference the Web services standards information online at sites such as IBM developerWorks®:

`http://www.ibm.com/developerworks/webservices/standards/`

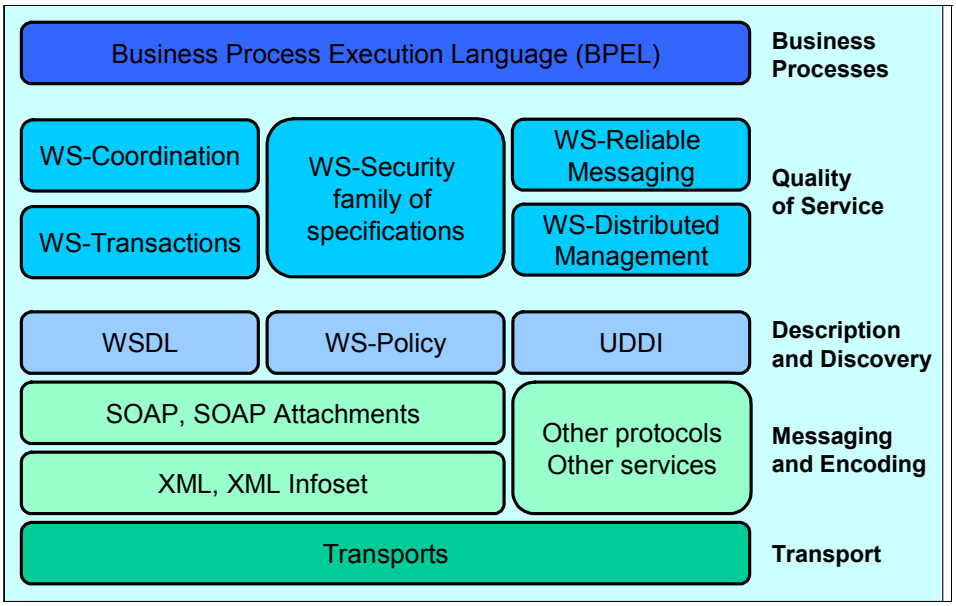


Figure 1-11 Stack view of Web services technology

Web services for J2EE

Web services for J2EE V1.1 (WSEE) support is included in the J2EE V1.4 specification which is used by WebSphere Application Server V6. The Java API for XML-based RPC (JAX-RPC) provides the programming model for SOAP-based applications by abstracting the runtime details and providing mapping services between Java and WSDL.

Exposing Web services

The port component is fundamental part of a Web service used to define the programming model artifacts that make the Web service portable. The programming model includes:

- A *WSDL definition* provides the description of a Web service.
- The *service endpoint interface* (SEI) defines the operations of the Web service.
- A *service implementation bean* implements the SEI methods to provide the business logic of the Web service.
- The *security role references* provide instance-level security checks across different modules.

From a server programming model perspective, there are primarily two types of J2EE application artifacts exposed as Web services (service provider):

- Stateless session EJB™ (EJB container)
- JAX-RPC servlet-based service that invokes a JavaBean, known as a service endpoint (Web container)

There are three principal approaches to generating a Web service, depending on the elements that are used to start the creation of the service:

- An existing application is used to generate the Web service, which includes a service wrapper used to expose application functionality. This is known as the *bottom-up* approach.
- An existing service definition WSDL is used to generate a new application for a specific programming language and model. This is known as the *top-down* approach.
- An existing group of already generated Web services provides a new combination of functionality (multiple services). Composing a new Web service in this way might include the use of workflow technologies.

Invoking Web services

The J2EE client container provides the WSEE runtime used by a Web services client application, to access and invoke Web service methods. The J2EE client container is responsible for the JNDI name to service implementation mapping.

From a client application programming perspective, there are three mechanisms used to invoke a Web service (service consumer) from the Web service client application:

- A *static stub* is created before being deployed to the runtime environment. This requires complete knowledge of the WSDL.

- A *dynamic proxy* class is created during runtime. Only a partial WSDL definition is required (port type and bindings).
- A *dynamic invocation* interface does not require WSDL knowledge. The signature or service name are unknown until runtime.

The task to build or generate a Web service client (service consumer) depends on the methods of how the client is binding to a Web service server. The client uses a local service stub or proxy to access the remote server and service. The WSDL document is used to generate or set up the particular stub or proxy. The stub or proxy knows at request time how to invoke the Web service based on the binding information.

1.3.2 Web services and SOA

Web services technology is a collection of standards (or emerging standards) that can be used to implement a service-oriented architecture (SOA). That is not to say that Web services and SOA are intrinsically linked, because they can be implemented separately.

In fact, many significant SOAs are proprietary or customized implementations that are based on reliable messaging and Enterprise Application Integration middle ware (for example, IBM WebSphere MQ and IBM WebSphere Business Integration Message Broker) do not use Web services technologies. Also, most existing Web services implementations consist of point-to-point integrations that address a limited set of business functions between a defined set of cooperating partners.

The logical links between Web services and SOA are:

- Web services provide an open-standard model for creating explicit, implementation-independent descriptions of service interfaces.
- Web services provide communication mechanisms that are location-transparent and interoperable.
- Web services are evolving, through Business Process Execution Language for Web Services (WS-BPEL), document-style SOAP, Web services Definition Language (WSDL), to support the implementation of well-designed services that encapsulate and model reusable function in a flexible manner.

1.4 Target audience of this book

This book includes topics for architecture, design, development, deployment, and administration. The target audience of this book can be best matched by role

to the topic of interest within the book. Table 1-2 provides a summary of the book topics by chapter and appendix for the following roles:

- Business Analyst
- Integration Specialist
- Software Architect
- Application Developer
- Enterprise Developer
- Enterprise Architect

Note: Refer to Appendix A, “Role descriptions” on page 473 for details.

Table 1-2 Matching redbook topics to roles and skills

	Chapter/appendix	Primary role	Secondary role
Part 1, “Getting started with IBM SOA Foundation”			
	Chapter 1, “Introduction to service- oriented architecture” on page 3	All	
	Chapter 2, “IBM SOA Foundation” on page 25	All	
	Chapter 3, “Service Creation scenario” on page 51	Business Analyst Enterprise Architect Software Architect	Application Developer Enterprise Developer
	Chapter 4, “Best practices for SOA” on page 83	Business Analyst Enterprise Architect	Application Developer Enterprise Developer
	Chapter 5, “Process for applying SOA scenarios and patterns” on page 131	Business Analyst Enterprise Architect	Software Architect Application Developer Enterprise Developer
	Chapter 6, “Patterns for the Service Creation scenario” on page 155	Business Analyst Enterprise Architect	Software Architect Application Developer Enterprise Developer
Part 2, “Service Creation scenario example”			
	Chapter 7, “Business scenario and solution architecture” on page 173	Business Analyst Software Architect Enterprise Architect	Application Developer Enterprise Developer
	Chapter 8, “Assemble an application with Rational Application Developer” on page 203	Application Developer Enterprise Developer	Software Architect Enterprise Architect
	Chapter 9, “Implement the runtime environment” on page 281	Integration Specialist	Enterprise Architect Software Architect Application Developer Enterprise Developer

	Chapter/appendix	Primary role	Secondary role
	Chapter 10, “Deploy application to WebSphere Application Server” on page 351	Integration Specialist Application Developer Enterprise Developer	Enterprise Architect Software Architect
	Chapter 11, “Walk through the application” on page 381	Business Analyst Integration Specialist	Application Developer Enterprise Developer Software Architect
	Chapter 12, “Manage and monitor services with ITCAM for SOA” on page 405	Integration Specialist	Enterprise Architect
Part 3, “Appendixes”			
	Appendix A, “Role descriptions” on page 473	Business Analyst Enterprise Architect	Software Architect Application Developer Enterprise Developer
	Appendix B, “IBM product descriptions” on page 475	Business Analyst Enterprise Architect	Software Architect Application Developer Enterprise Developer
	Appendix C, “Troubleshooting tips” on page 491	Integration Specialist	Software Architect Application Developer Enterprise Developer
	Appendix D, “Web Services Security” on page 505	Integration Specialist Application Developer Enterprise Developer	Enterprise Architect Software Architect
	Appendix E, “Additional material” on page 515	Software Developer Enterprise Developer	Integration Specialist

2

IBM SOA Foundation

If you are not familiar with the concepts of SOA, read Chapter 1, “Introduction to service- oriented architecture” on page 3 before reading this chapter.

This chapter discusses the *IBM SOA Foundation*, 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.

The key elements that we discuss in this chapter are are:

- SOA life cycle (model, assemble, deploy, manage)
- Reference architecture
- SOA scenarios overviews for the rest of the book

2.1 SOA Foundation overview

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 within the SOA scenario.

The SOA scenarios can be used as a reference architecture (starting point) to accelerate the SOA architecture and implementation of your customer scenario. The SOA scenarios can be implemented using an incremental SOA adoption approach, whereby a customer can incrementally add elements of other SOA scenarios to the environment to achieve their business objectives.

To gain a a better understanding of the IBM SOA Foundation, we explore the following defining elements:

- SOA Foundation life cycle
- SOA Foundation Reference Architecture
- SOA Foundation scenarios

Note: For a more detailed explanation of the SOA Foundation, refer to *IBM SOA Foundation, An Architectural Introduction and Overview V1.0* found here:

`http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf`

2.2 SOA Foundation life cycle

IBM customers have indicated that they think of SOA in terms of a life cycle. As seen in Figure 2-1, the IBM SOA Foundation includes the following life cycle phases:

- Model
- Assemble
- Deploy
- Manage

There are a couple of key points to consider about the SOA life cycle:

First, the SOA life cycle phases apply to all SOA projects. Second, the activities in any part of the SOA life cycle can vary in scale and the level of tooling used depending on the stage of adoption.

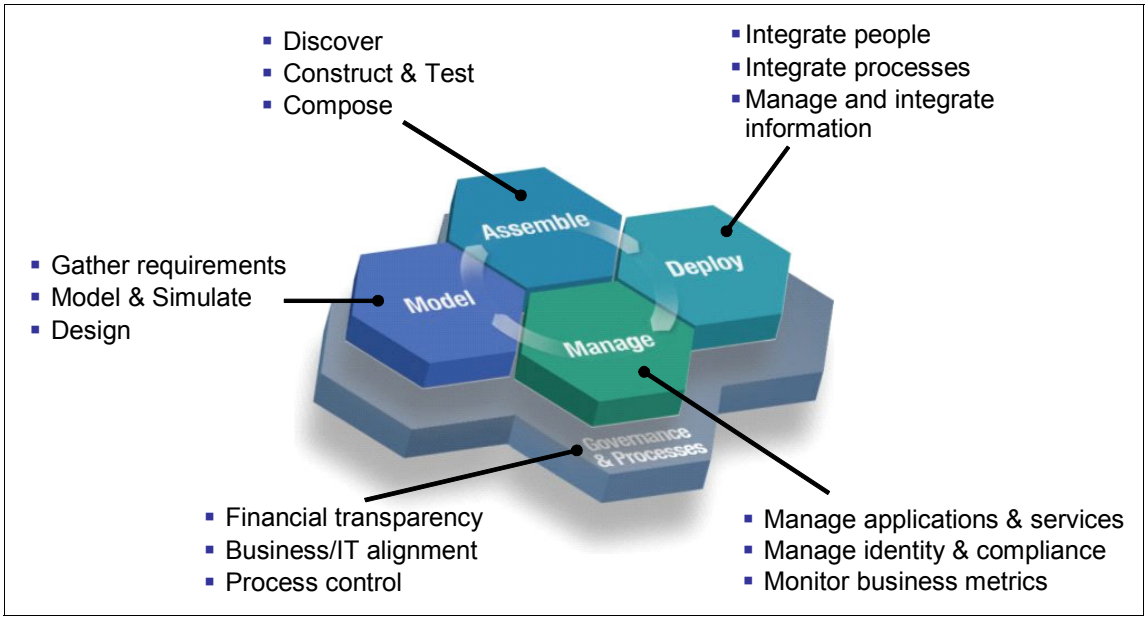


Figure 2-1 IBM SOA Foundation life cycle

2.2.1 Model

Modeling is the process of capturing the business design from an understanding of business requirements and objectives. The business requirements are translated into a specification of business processes, goals and assumptions for creating a model of the business. Many businesses do not go through a formal modeling exercise. In some case, businesses that do perform modeling use primitive techniques such as drawing the design in Visio® or using text documents.

Capturing the business design using a sophisticated approach that includes the use of specialized tooling lets you perform what-if scenarios with various parameters the business might experience. The process can then be simulated using those parameters to predict the effect that process will have on the business and IT systems. If the achieved results do not match the business objectives, then the process definition can be refined.

The model also captures key performance indicators that are important measurements of your business, such as business metrics. For example, these measurements could include a measure of the new accounts that you have opened in a given month. These key performance indicators are built into the

assembly of the application. In addition, you can monitor the indicators in production, capturing critical data to measure if the objectives are being met.

2.2.2 Assemble

The business design is used to communicate the business objectives to the IT organization that will assemble the information system to implement the design. The enterprise architect works closely with the business analyst to convert the business design into a set of business process definitions, as well as activities used to derive the required services from the activity definitions. The enterprise architect and business analyst work with the software architect to fully develop the design of the services.

While you are resolving the design and implementation of the modeled business processes and services, you should perform a search of existing artifacts and applications to find components that meet the design requirements. Some applications will fit perfectly. Some applications will have to be refactored, and some will have to be augmented to meet the design requirements.

These existing assets should be rendered as services for assembly into composite applications. Any new services required by the business design must be created. Software developers should use the SOA programming model to create these new services.

Lastly, the assemble phase includes applying a set of policies and conditions to control how your applications operate in the production runtime environment. For example, these policies and conditions include business and government regulations. In addition, the assemble phase includes critical operational characteristics such as packaging deployment artifacts, localization constraints, resource dependency, integrity control, and access protection.

2.2.3 Deploy

The deploy phase of the life cycle includes a combination of creating the hosting environment for the applications and the deployment tasks of those applications. This includes resolving the application's resource dependencies, operational conditions, capacity requirements, and integrity and access constraints.

A number of concerns are relevant to construction of the hosting environment, including the presence of the already existing hosting infrastructure supporting applications and pre-existing services. Beyond that, you must consider appropriate platform offerings for hosting the user interaction logic, business process flows, business-services, access services, and information logic.

2.2.4 Manage

The manage phase includes the tasks, technology and software used to manage and monitor the services and business processes that are deployed to the production runtime environment.

Monitoring is a critical part of ensuring the underlying IT systems and application are up and running to maintain service availability requirements. Monitoring includes these activities:

- Monitoring performance of service requests and timeliness of service responses.
- Maintaining problem logs to detect failures in various services and system components, as well as localizing failures and restoring the operational state of the system.

Managing the system also involves performing routine maintenance, administering and securing applications, resources and users, and predicting future capacity growth to ensure that resources are available when the demands of the business warrant using them. The security domain includes such topics as authentication, single sign-on, authorization, federated identity management, and user provisioning.

The manage phase also includes managing the business model, and tuning the operational environment to meet the business objectives expressed in the business design, and measuring success or failure to meet those objectives. SOA is distinguished from other styles of enterprise architecture by its correlation between the business design and the software that implements that design, and its use of policy to express the operational requirements of the business services and processes that codify the business design. The manage phase of the life cycle is directly responsible for ensuring those policies are being enforced, and for relating issues with that enforcement back to the business design.

2.2.5 Governance

SOA Governance is critical to the success of any SOA project. Governance helps clients extend the planned SOA across the enterprise in a controlled manner. SOA Governance has four core objectives or challenges:

- Establish decision rights.
- Define high value business services.
- Manage the life cycle of your assets.
- Measure effectiveness.

Note: For more detailed information on SOA Governance, refer to 4.3, “SOA Governance” on page 107.

2.3 SOA Foundation Reference Architecture

This section describes the SOA Foundation Reference Architecture, which includes the components and middle ware services used by applications in the runtime environment.

Figure 2-2 depicts the SOA Foundation Reference Architecture solution view used to decompose an SOA design. SOA puts a premium on the role of the Enterprise Architect, who is responsible for spanning between the business design and the information system that codifies that design.

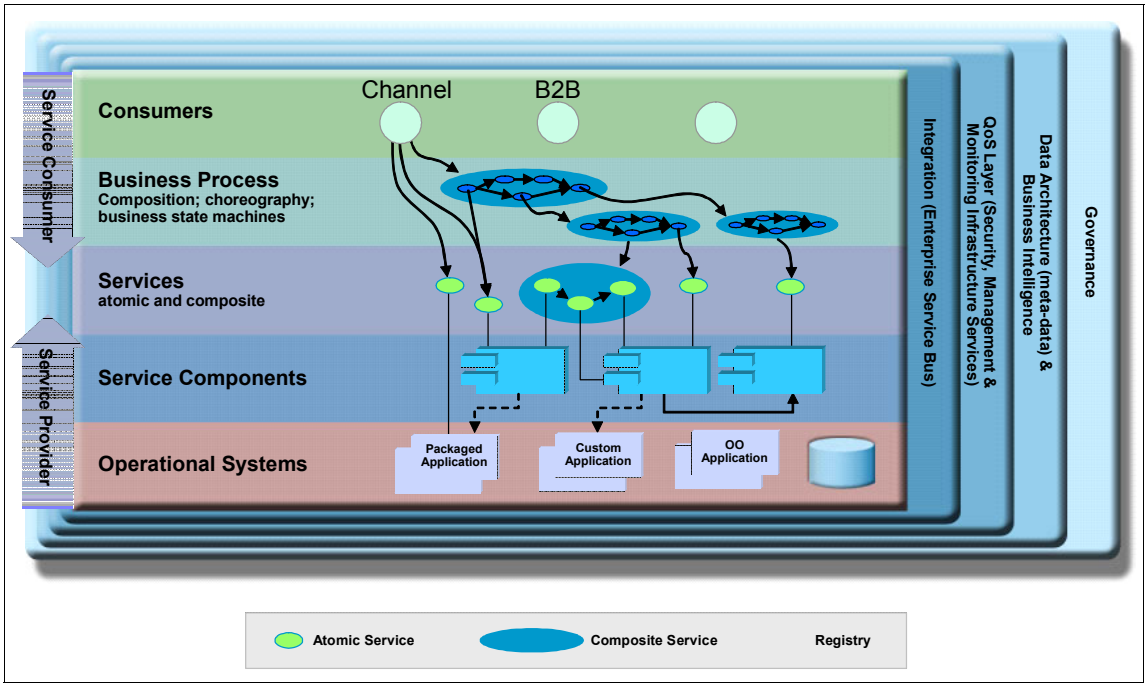


Figure 2-2 SOA Foundation Reference Architecture: Solution view

While this flow describes a top-down approach, flow variations include a bottom-up approach, and the more common meet-in-the-middle approach. When taking a top-down approach, the enterprise architect starts by identifying the business processes and business services used by the business users. The business users are consumers of the processes and services. Business processes should be treated as compositions of other business processes and services, and therefore should be separated into their subordinate subprocesses and services.

Services and business processes are then detailed into service components. Service components include a detailed set of definition metadata used to describe the service to the information system. Services can be aggregated into module assemblies. The module assemblies are used to establish related design concerns, and begin the planning to determine what teams will collaborate to implement the related services to be deployed as a single unit.

The resulting set of business process definitions, services, and schemas make up the logical architecture of the application. The enterprise architect then needs to map that logical architecture to a physical architecture.

We have included a summary description for each of the services found in the logical architecture displayed in Figure 2-3. The services found in the center (Interaction, Process, Information, Partner, Business Application, Access) are the core set of services used by application within the runtime environment when deployed. The other outer services displayed are used in support of the core services.

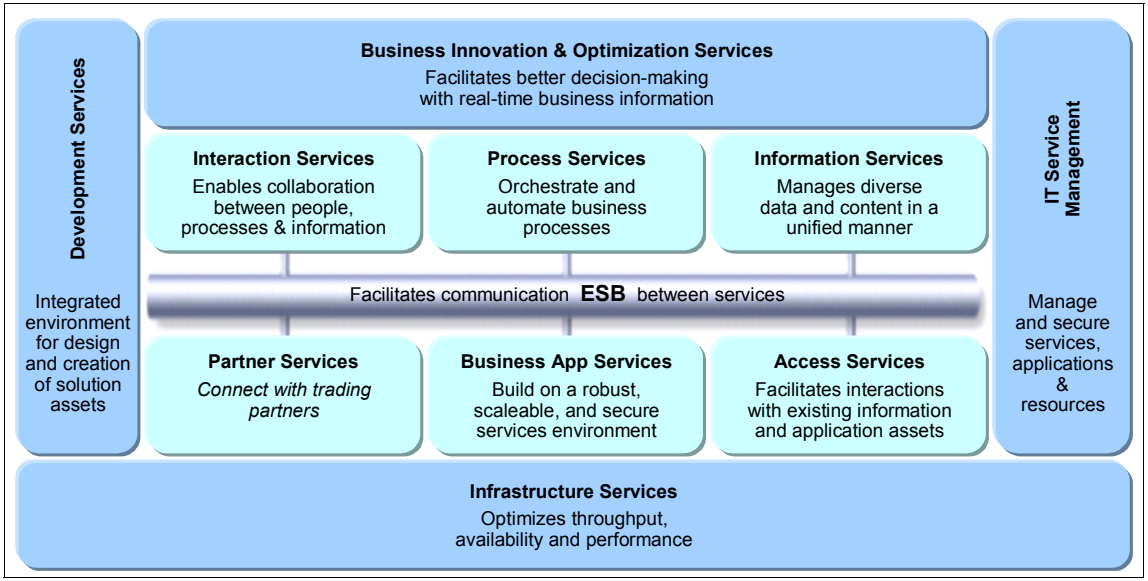


Figure 2-3 SOA Foundation Reference Architecture: Middleware Services view

Core components of the logical architecture

This section includes a brief description on the following core components of the logical architecture:

- Interaction services
- Process services
- Business application services

- Information services
- Access services
- Partner services

Interaction services

Interaction services provide the capabilities required to deliver IT functions and data to users, meeting their specific preferences.

Process services

Process services provide the control capabilities required to manage the flow and interactions of multiple services in ways that implement business processes.

Business application services

Business application services are called by service consumers. Service consumers include other components in the logical architecture such as portal or a business processes.

Information services

Information services provide the capabilities necessary to federate, replicate and transform disparate data sources.

Access services

Access services provide bridging capabilities between core applications, prepackaged applications, enterprise data stores and the ESB to incorporate services that are delivered through existing applications into an SOA.

Partner services

Partner services provide the document, protocol, and partner management capabilities for business processes that involve interactions with outside partners and suppliers.

Supporting components of the logical architecture

This section includes a brief description of the supporting components of the SOA Foundation logical architecture used in support of the core components:

- Enterprise Service Bus
- Business innovation and optimization services
- Development services
- IT service management
- Infrastructure services

Enterprise Service Bus

The Enterprise Service Bus (ESB) or simply bus, provides an infrastructure that removes the direct connection dependency between service consumers and

providers. Consumers connect to the bus and not the provider that actually implements the service. This type of connection further decouples the consumer from the provider. A bus also implements further value add capabilities, such as security and delivery assurance. It is preferred to implement these capabilities centrally within the bus at an infrastructure level rather than within the application. The primary driver for an ESB, however, is that it increases decoupling between service consumers and providers.

Although it is relatively straight forward to build a direct link between a consumer and provider, these links can lead to an interaction pattern that consists of building multiple point-to-point links that perform specific interactions. With a large number of interfaces this quickly leads to the build up of a complex spaghetti of links with multiple security and transaction models. When routing control is distributed throughout the infrastructure, there is typically no consistent approach to logging, monitoring, or systems management. This type of environment is difficult to manage or maintain and inhibits change.

Note: An ESB can be thought of as an architectural pattern, with an implementation to match the deployment needs. There are two IBM ESB products:

- IBM WebSphere Enterprise Service Bus
- IBM WebSphere Message Broker

In addition, there are a number of products that extend the capabilities of these ESBs, including DataPower® XML Security Gateway XS40.

Business innovation and optimization services

Business innovation and optimization services are primarily used to represent the tools and the metadata structures for encoding the business design, including the business policies and objectives.

Business innovation and optimization services exist in the architecture to help capture, encode, analyze and iteratively refine the business design. The services also include tools to help simulate the business design. The results are used to predict the effect of the design, including the changes the design will have on the business.

Development services

Development services encompass the entire suite of architecture tools, development tools, visual composition tools, assembly tools, methodologies, debugging aids, instrumentation tools, asset repositories, discovery agents, and publishing mechanisms needed to construct an SOA based application.

IT service management

After the application has been deployed to the runtime environment, it must be managed along with the IT infrastructure on which it is hosted. IT service management represents the set of management tools used to monitor your service flows, the health of the underlying system, the utilization of resources, the identification of outages and bottlenecks, the attainment of service goals, the enforcement of administrative policies, and recovery from failures.

Infrastructure services

Infrastructure services form the core of the information technology runtime environment used for hosting SOA applications. These services provide the ability to optimize throughput, availability, performance and management.

2.4 SOA Foundation scenarios

The SOA Foundation scenarios (simply SOA scenarios) are representative of common scenarios of IBM products and solutions for SOA engagements. The SOA scenarios quickly communicate the business value, architecture, and IBM open standards-based software used within the SOA scenario. The SOA scenarios can be implemented as part of an incremental adoption of SOA growing from one scenario to using elements of multiple scenarios together. The concept of realizations are used to provide more specific solution patterns and IBM product mappings within the SOA scenarios.

The SOA scenarios can be used as a reference architecture implementation (starting point) to accelerate the SOA architecture and implementation of your customer scenario.

Figure 2-4 displays the SOA scenarios (Service Creation, Service Connectivity, Interaction and Collaboration Services, Business Process Management, Information as a Service), and the relationship between the scenarios.

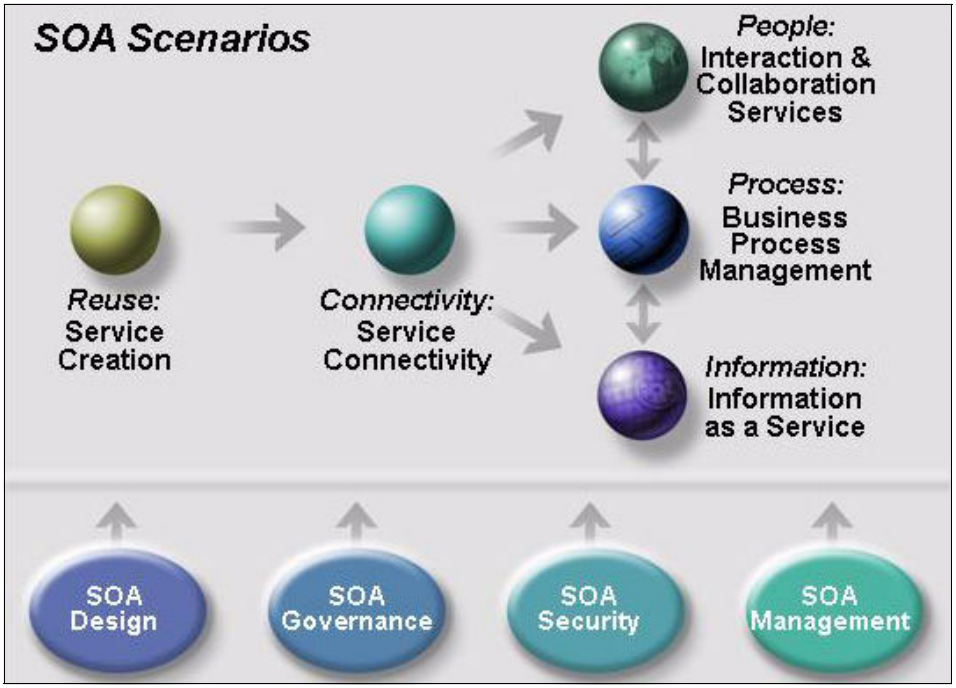


Figure 2-4 SOA Scenarios and Entry Points

We have included examples of how the scenarios can be used together and adopted incrementally. For example, it is common that the other scenarios will include service creation and often want connectivity. In addition, the scenarios can be used together such as a portal accessing a business process or a portal accessing an information service through an ESB from a service consumer.

SOA Design, SOA Governance, SOA Security and SOA Management are used in each of the SOA scenarios based on customer requirements.

SOA Design is focused on service modeling and service design. The service model is an abstraction of the IT services implemented within an enterprise and supporting the development of one or more service-oriented solutions. It is used to conceive and document the design of the software services. It is a comprehensive, composite work product encompassing all services, providers, specifications, messages, collaborations, and relationships between them.

The service design model extends the standard Rational Unified Process (RUP) design model by adding the service components. The RUP for SOA service design includes service identification and modeling techniques derived from IBM Service-Oriented Modeling and Architecture (SOMA). The service components artifact is intended for use in describing the realization of a service specification.

A service component can provide the realization for one or more services by the realization of multiple service specifications. The set of model elements on the inside of the component represent the concrete realization of the structural, behavioral, and policy contract described by these service specifications.

SOA Governance is critical to the success of any SOA project. Governance helps clients extend the planned SOA across the enterprise in a controlled manner. SOA Governance has four core objectives or challenges:

- Establish decision rights
- Define high value business services
- Manage the life cycle of assets
- Measure effectiveness

Note: For more detailed information on SOA Governance, refer to 4.3, “SOA Governance” on page 107.

Every SOA implementation encompasses the need of 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 factors of this reference architecture include security policy and security management infrastructures. For more detailed information about the SOA security reference architecture, refer to the redbook *Understanding SOA Security Design and Implementation*, SG24-7310.

SOA-based composite applications span the architectural layers of the SOA Foundation Reference Architecture Solution View (see Figure 2-2 on page 30). SOA composite applications require a mind shift from a silos-based, application-management approach for two key reasons:

- There is a need for a management approach and tooling that covers the end-to-end view of transactional performance and availability of the composite application.
- Second, it is important to understand the relationship of service consumers and providers for composite applications. The importance of monitoring the availability and performance of common services increases when reused by service consumers that depend on this functionality.

SOA Management provides best practices and software for managing and monitoring composite applications and supporting infrastructure across the architectural layers. This includes managing and monitoring the services layer (providers and consumers), transactional performance, middle ware, and operational systems with specific metrics to ensure Service Level Agreements (SLAs) meet the defined business objectives.

Note: For more detailed information on SOA management, refer to 12.1, “Introduction to SOA management” on page 406.

We have provided a summary for each of the following SOA scenarios:

- Service Creation scenario
- Service Connectivity scenario
- Interaction and Collaboration Services scenario
- Business Process Management scenario
- Information as a Service scenario

2.4.1 Service Creation scenario

The Service Creation scenario (service provider and service consumer) is used to demonstrate exposing application functionality of an existing application or new business logic as a service. The services can then be consumed by other services or client applications within and between enterprises. The key driver for this scenario is the reuse of existing or new application functions as services.

Note: For a more detailed explanation of the Service Creation scenario, refer to Chapter 3, “Service Creation scenario” on page 51.

There are many possible examples to illustrate the Service Creation scenario. We have included a summary of the following four common realizations of the Service Creation scenario:

- Directly expose existing applications as services
- Indirectly expose existing applications with service components
- Create an EJB Web service from WSDL
- Consume services from 3rd party service providers

One of the goals of enterprise transformation is to enable access to Enterprise Information System (EIS) applications as services, with the objective of leveraging the investment of existing business applications and systems. By adopting an SOA approach, the EIS application functionality can be turned into reusable services that can be consumed by a new set of client applications and users.

The Service Creation scenario includes two methods of exposing EIS applications as services, known as direct and indirect exposure. We use CICS® as an example of an existing EIS application for both the direct and indirect exposure realization examples.

Directly expose existing applications as services

In this realization, we expose existing applications directly as services. The existing application can be a wide range of application types such as an EIS (for example, CICS or IMS™), J2EE application, SAP, and so forth. A key distinction for this realization is that the service interface to be exposed is defined by the existing application. This approach to creating a service is known as *bottom-up*.

We use CICS as an example of an existing EIS application. In this realization example, we access applications hosted by CICS Transaction Server *directly* as Web services.

The core IBM products used for this example realization are:

- Assemble: WebSphere Developer for zSeries® V6.0.1
- Deploy:
  - CICS Transaction Server V3.1
  - WebSphere MQ V6

Additionally, the following IBM products can be considered, depending on specific customer requirements:

- Manage: Tivoli OMEGAMON® XE for CICS V3.10
- Governance: WebSphere Service Registry and Repository

WebSphere Service Registry and Repository: IBM currently intends to make available in the second half of 2006 a WebSphere service registry and repository capability that will allow customers, to securely register business services for finding, publishing and notifying changes to SOA infrastructure components such as enterprise service bus and process servers. Customers will also be able to house the metadata about business services in managing the life cycle of a service in SOA. The new capabilities will also include a model for governance to provide guidance and oversight for a SOA project.

While this is our intention, all statements regarding IBM's plans, directions, and intent are subject to change or withdrawal without notice.

Indirectly expose existing applications with service components

For this realization, we expose existing application functionality indirectly with service components. In this realization, business alignment is achieved by defining the service interface (WSDL), independently of existing assets. This approach to creating a service is known as top-down. In practice, the implementation of the service might be more completely described as a combination of top-down and meet-in-the-middle.

We use CICS as an example of an existing EIS. In this realization example, we expose existing COMMAREA applications hosted by CICS Transaction Server *indirectly* by creating middle-tier Web services to access CICS. The middle-tier Web service wraps a session EJB that uses the CICS ECI resource adapter to communicate with the CICS Transaction Gateway (CTG) to access CICS Transaction Server. The CICS ECI resource adapter is a JCA adapter for WebSphere Application Server packaged with the CTG.

The core IBM products used for this example realization are as follows:

- Assemble: Rational Application Developer V6.0.1
- Deploy:
  - WebSphere Application Server V6 (zSeries V6 or distributed platform depending on the selected runtime topology).
  - CICS Transaction Gateway (TG) V6.1
    (CICS ECI resource adapter for WebSphere is included with the CICS TG)
  - CICS V2.x

Additionally, the following IBM products might be considered, depending on specific customer requirements:

- Model: Rational Software Architect V6
- Manage:
  - Tivoli OMEGAMON-XE for CICS V3.1
- Governance: WebSphere Service Registry and Repository

Create an EJB Web service from WSDL

In this realization, we create a new session EJB Web service from an existing service WSDL. This realization is also known as *create from scratch.* This realization uses a top-down approach to creating a Web Service.

The core IBM products used for this realization are:

- Assemble: Rational Application Developer V6
- Deploy: WebSphere Application Server Network Deployment V6
- Manage: Tivoli Composite Application Manager for SOA V6

Additionally, the following IBM products might be considered, depending on specific customer requirements:

- Model: Rational Software Architect V6
- Deploy: WebSphere Service Registry and Repository
- Manage: Tivoli Composite Application Manager for WebSphere V6
- Governance: WebSphere Service Registry and Repository

Consume services from 3rd party service providers

In this realization, client applications consume services from third-party service providers. This realization represents the view of a consumer that is using one or more third-party services. The service consumer only sees the service interfaces. The endpoints and any security or transport constraints are imposed by the service provider.

For example, a Web services client application can invoke an address verification Web service from a third-party service provider.

This realization assumes that the WSDL is WS-I compliant and JAX-RPC compatible. The service provider is outside the enterprise firewall and security is implemented between the consumer and provider using mutual SSL authentication.

The core IBM products used for this realization as follows:

- Assemble: Rational Application Developer V6
- Deploy: WebSphere Application Server V6

Additionally, the following IBM products might be considered, depending on specific customer requirements:

- Manage:
  - Tivoli Composite Application Manager for SOA V6
  - Tivoli Composite Application Manager for WebSphere V6
- Governance: WebSphere Service Registry and Repository

2.4.2 Service Connectivity scenario

The Service Connectivity scenario is used to demonstrate the integration of service providers and consumers, allowing for the reuse of existing and new services across multiple channels. This scenario is appropriate for an enterprise that has a set of core services or systems which are to be made available as services to internal and external clients. Flexibility to make changes to service providers and service clients independent from each other is a requirement.

The focus of this scenario is on the underlying connectivity used to support business-centric SOA. An enterprise service bus provides decoupling between clients and providers, providing the flexibility to implement applications more quickly. In circumstances where services are provided to or consumed from a third party, an ESB gateway can be used in conjunction with the ESB to add security measures. An ESB gateway alone can be sufficient if all of your service interactions are with third parties and you have basic requirements to mediate between service consumers and providers.

Note: A more detailed description of the Service Connectivity scenario can be found in the *Patterns: SOA Foundation - Service Connectivity Scenario*, SG24-7228 redbook.

Implementations of this scenario have the following features:

- Enables changes to the implementation of a service without affecting clients.
- Registers services to a service registry.
- Uses an enterprise service bus as the integration point between service providers and service consumers.
- Enables clients to access a service with a different interface and protocol than what the service consumer supports.
- Uses an ESB gateway to isolates and protect services.
- Enables management and monitoring of services to insure Service Level Agreements.
- Provides security and credential mapping (where needed) to insure proper use of the services.

Specific connectivity and integration requirements for an enterprise will ultimately drive product selection of the ESB and supporting products. The choice of runtime products can include one or more of the following:

- WebSphere Message Broker
- WebSphere Enterprise Service Bus
- IBM DataPower SOA Appliances
- Web Services Gateway (WebSphere Application server Network Deployment V6)
- WebSphere Adapters
- WebSphere Service Registry and Repository (available 2H06)

The choice of SOA life cycle products will depend largely on the runtime products selected. The following products can be used to support the runtime environment:

- WebSphere Message Broker Toolkit
- Rational Application Developer
- WebSphere Integration Developer
- IBM Tivoli Composite Application Manager for SOA (ITCAM for SOA)
- IBM Tivoli Composite Application Manager for WebSphere (ITCAM for WebSphere)
- IBM Tivoli Composite Application Manager for RTT V6.0 (TCAM for RTT)

- OMEGAMON for Messaging
- Tivoli Access Manager
- Tivoli Federated Identity Manager

Realizations have been developed that will help you understand how the scenario can be used and how products are selected. A *realization* is an example business case that describes a customer situation and the solution. We have included a summary of the following common realizations of the Service Connectivity scenario:

- Gateway: For use when interactions with third parties are present and mediation requirements are basic.
- Local integration: For use with standards-based interactions that require routing capabilities.
- Web services access to Enterprise Information Systems: For use when requiring access to EIS systems.
- Expose existing systems to heterogeneous clients: For use in a diverse, nonstandards-based environment.

Gateway

An ESB gateway can be used alone or in conjunction with an ESB to provide controlled and secure service interaction between internal or external domain boundaries. In this realization, its primary function is to provide secure access to resources when interacting with 3rd parties. An ESB gateway can also provide basic functionality such as protocol switching and message switching to enable interaction between service consumers and service providers.

This realization assumes the customer has adopted standards-based technology, has an existing infrastructure, and has the following business requirements:

- Standards-based requestors/providers will use SOAP/HTTP for transport.
- Dynamically add new providers and requestors at runtime.
- Support a defined, high response time with a moderate load.
- SOA security for interaction with requestors and providers. Security may need to be adapted between the requestors and providers.
- Requests and responses must be logged to a file

The following technical requirements have been identified:

- Many services deployed will require the same mediation flow. An ESB gateway will minimize administration and streamline the process for making new services available.

- Services must be monitored for performance and usage.
- Monitoring for all components must be integrated into existing management infrastructure.

The IBM products used for this realization are as follows:

- Deploy: IBM DataPower XML Security Gateway XS40

  The customer chooses the DataPower XS40 model for the runtime. The XS40 is designed specifically to provide XML acceleration and SOA security and can provide the basic mediation functions required. Because the requirements for mediating the interaction between consumers and providers is met, the XS40 as an ESB gateway is sufficient and no ESB product is required.
- Assemble: DataPower Toolkit
- Manage: IBM Tivoli Composite Application Manager for SOA

  ITCAM for SOA will be used to monitor Web services flowing through the DataPower appliance.
- Manage: Tivoli Access Manager

  The XS40 can be integrated with Tivoli Access Manager to secure applications.

Local integration

A local integration solution provides multi-channel access for clients to an existing service with a range of connectivity options for standards-based clients and services, message routing with or without the use of external data, message transformation and augmentation, protocol switching, and security.

With this type of connectivity, a client can request a secure service without knowledge of its location. Transparent to the client, requests can be routed to the service which can best handle the request. Also transparent to the client is the message format and transport protocol required to access the provider. The response could be immediate or delayed.

This realization assumes the customer has adopted standards-based technology, has an existing WebSphere Application Server infrastructure, and has the following business requirements:

- Provide integration of multiple client channels to service providers.
- Provide routing of client requests to the appropriate service provider.
- Intranet environment which does not require WS-Security or other complex security considerations.
- Support moderate volume of requests

The following technical requirements have been identified:

- Message data from clients must be examined in order to determine the service provider to route the request to.
- Clients and service providers will use JMS, SOAP/JMS, or SOAP/HTTP.
- Data transformation will be required. This should be done with XSLT.

The core IBM products used for this realization are as follows:

- Deploy: WebSphere Enterprise Service Bus

  WebSphere ESB provides the transport flexibility to support the transports required by the customer. WebSphere ESB also has the mediation capabilities required to perform the message routing and transformation required.

- Assemble: WebSphere Integration Developer

  WebSphere Integration Developer is the development and assembly tool for building WebSphere ESB mediations.

- Manage: IBM Tivoli Composite Application Manager for SOA

  ITCAM for SOA will be used to monitor Web services requests as they arrive at WebSphere ESB.

Web services access to Enterprise Information Systems

An ESB can be used to provide access to EIS systems through the use of adapters. Mediations in the ESB are used to adapt the client request to a form understood by the adapter, and then to adapt the response to the client's format.

This realization assumes the customer has adopted standards-based technology, has an existing WebSphere Application Server infrastructure, and has the following business requirements:

- Provide Web service access to functionality in an Enterprise Information System such as SAP R/3, PeopleSoft, or Oracle Financials.
- Intranet environment which does not require WS-Security or other complex security considerations.
- The integration is based on message exchange/data replication scenarios - there is no business process or data synchronization between clients and EIS systems
- Support moderate volume of requests

The following technical requirements have been identified:

- The targeted integration is point-to-point, although multiple EISs can be exposed as Web services at the same time.

- Data transformation will be required. This should be done with XSLT.
- Log the messages as they flow through the hub – want to log asynchronously to a file.

The IBM products used for this realization are:

- Deploy: WebSphere Enterprise Service Bus and WebSphere Adapters

  WebSphere ESB supports the SOAP/HTTP transport required by the customer. WebSphere Adapters provide the EIS adapters required. WebSphere ESB also provides the mediation capability required to do XSLT transformation on the data and includes a logging function to log messages as they flow through the mediation.

- Assemble: WebSphere Integration Developer

  WebSphere Integration Developer is the development and assembly tool for building WebSphere ESB mediations. It includes the enterprise discovery capabilities needed to incorporate the WebSphere Adapters into the mediation applications.

- Manage: IBM Tivoli Composite Application Manager for SOA

  ITCAM for SOA will be used to monitor Web services requests as they arrive at WebSphere ESB.

Expose existing systems to heterogeneous clients

An integration solution that includes a range of diverse business applications must provide connectivity for a wide range of service consumers and service providers as well as advanced options for message mediation, including message augmentation, message routing, and the ability to decompose messages into multiple requests and to recompose the responses.

This type of connectivity would provide the most advanced options for integrating dissimilar and wide-spread service consumers and service providers. Clients can request a secure service that may be provided by one or more service providers with the service composition occurring within the ESB. Services and clients also have a wide range of connectivity options. Connectivity to legacy applications as well as standards-based applications are managed by the ESB.

This realization assumes the customer has extensive legacy systems as well as some newer Web services based systems and has the following business requirements:

- Providers use a variety of heterogeneous protocols
- Any provider must be accessible through basic Web services which will be used by a variety of clients
- Support moderate volume of requests

- Intranet environment does not require SOA security or other complex security considerations
- Global transactions across multiple heterogeneous transaction managers for some providers

The following technical requirements have been identified:

- The ESB must support communication protocol conversion.
- The ESB must support flexible data model conversion, with acceptable performance and adequate tooling
- Enterprise class persistent messaging backbone
- Global transactions management
- The ESB must adapt the service definitions between the requestors and providers

The IBM products used for this realization are as follows:

- Deploy: WebSphere Message Broker, WebSphere MQ, and WebSphere Adapters

  WebSphere Message Broker is selected to provide the ESB capabilities, including mediation support. WebSphere MQ will be used to provide an enterprise class persistent messaging backbone. This combination will support the wide variety of transport protocols and conversions required for the integration solution. WebSphere Adapters provide connectivity to legacy systems.
- Assemble: Message Brokers Toolkit

  The Message Brokers Toolkit is the development tool for building mediation message flows in WebSphere Message Broker and provides the runtime configuration and management tools.

2.4.3 Interaction and Collaboration Services scenario

The Interaction and Collaboration Services scenario features single sign-on and a role based portal used to consolidate access to information and application within the enterprise and between enterprises.

The key drivers for this scenario are to improve people productivity and consumability of applications and content. The content can be personalized in the aggregated portal page based on the user role.

The core IBM products used for the Interaction and Collaboration Services scenario are as follows:

- Assemble:
  - Rational Application Developer V6
  - Bowstreet Portlet Factory
- Deploy: WebSphere Portal V5.1
- Manage: Tivoli Composite Application Manager for SOA V6

Additionally, the following IBM products may be considered by a customer depending on specific requirements:

- Assemble: WebSphere Integration Developer
- Deploy:
  - WebSphere Process Server
  - WebSphere Everyplace® Deployment Server
- Manage:
  - Tivoli Access Manager
  - Tivoli Federated Identity Manager

2.4.4 Business Process Management scenario

Business Process Management leads to business innovation and optimization by implementing business strategy through modeling, developing, deploying and managing business processes throughout the entire life cycle. Business Process Management acts as an enabler for businesses in defining and implementing strategic business goals and then measuring and managing a company's financial and operational performance against these goals.

Note: A more detailed description of the Business Process Management scenario can be found in the *Patterns: SOA Foundation - Business Process Management Scenario*, SG24-7234 redbook.

Business Process Management combines business processes, information, and IT resources, aligning the organization's core assets, people, information, technology, and processes to create a single integrated view, with real-time intelligence, of both its business measurements and IT system performance.

The IBM process integration portfolio provides capabilities required for the delivery of the comprehensive enterprise wide Business Process Management strategies and solution. It offers a holistic approach to transform and manage a business by aligning strategic and operational objectives with business activities and supporting IT services. The IBM Business Process Management solution includes development tools used to implement custom artifacts that leverage the infrastructure capabilities, and business performance management tools, used to

monitor and manage the runtime implementations at both the IT and business process levels.

The core IBM products used for the Business Process Management scenario are:

- Model: WebSphere Business Modeler V6
- Assemble: WebSphere Integration Developer V6
- Deploy: WebSphere Process Server V6
- Manage:
  - WebSphere Business Monitor V6
  - Tivoli Composite Application Manager for SOA V6

The additional IBM products used for this scenario, depending on customer requirements, are as follows:

- Model: Rational Software Architect V6
- Deploy:
  - WebSphere Portal
  - WebSphere Adapters
- Manage:
  - Tivoli Access Manager
  - Tivoli Federated Identity Manager
- Governance: WebSphere Service Registry and Repository

2.4.5 Information as a Service scenario

The Information as a Service scenario is used to demonstrate unified and trusted information available as services from multiple data sources. This scenario includes content management, e-forms, security, business intelligence, information integration through *just-in-time* integration, and ETL (extract, transform and load).

The key drivers for this scenario are to facilitate better decision making and better information sharing between business operations.

The core IBM products used in the Information as a Service scenario are as follows:

- Model:
  - WebSphere Business Modeler
  - Rational Data Architect
- Assemble:
  - WebSphere Integration Developer
  - WebSphere DataStage® Designer
- Deploy:
  - WebSphere Information Server

  - WebSphere DataStage Integration Suite
- Manage:
  - WebSphere Business Monitor
  - Tivoli CAM for SOA

Additionally, the following IBM products can be considered by a customer, depending on specific requirements:

- Deploy:
  - WebSphere Portal
  - WebSphere Adapters
  - Workplace™ Forms View / Server
- Manage:
  - Tivoli Access Manager
  - Tivoli Federated Identity Manager

3

Service Creation scenario

This chapter provides detailed information on the Service Creation scenario. We describe the key business drivers, define common realizations of the scenario, examine the integration architecture and components, and highlight the key tasks performed using IBM software in the SOA life cycle phases.

This chapter is organized into the following sections:

- Service Creation scenario overview
- Directly expose existing applications as services
- Indirectly expose existing applications with service components
- Create an EJB Web service from WSDL
- Consume services from third-party service providers

3.1 Service Creation scenario overview

The Service Creation scenario (service provider and service consumer) is used to demonstrate exposing application functionality of an existing application or new business logic as a service. The services can then be consumed by other services or client applications within an enterprise and between enterprises.

Business context

For example, a customer has a core set of IT or business functions which offer value to a variety of internal and external clients. The customer wants to enable a wider use of these functions without incurring the complexity of point-to-point integration with each of its customers.

By exposing the application functionality as services, client applications internal and external to the enterprise can consume these common services. This SOA approach simplifies the integration challenges and leverages the business value of existing systems.

Business value

The Service Creation scenario solution domain includes the following defining elements that provide business value:

- Ease integration challenges.
- Provide a modern. industry standards-based posture to EIS (for example, CICS or IMS) systems.
- Leverage the business value of existing applications.
- Enhance responsiveness to business demands.
- Increase reliability through reuse and fewer connections.

Service Creation scenario realizations

There are many possible examples to illustrate the Service Creation scenario. This chapter includes a description of the following four realizations that highlight the key elements of the Service Creation scenario:

- Directly expose existing applications as services
- Indirectly expose existing applications with service components
- Create an EJB Web service from WSDL
- Consume services from third-party service providers

The realizations are representative of the service architecture and the method in which the service is created. The service architecture describes whether the application functionality will be exposed directly as a service, or indirectly using a middle-tier service component. The realization examples include different

approaches to creating services such as bottom-up and top-down. There is a wide range of existing application types such as EIS (for example, CICS and IMS), J2EE, SAP, and so forth.

EIS application access patterns

One of the goals of enterprise transformation it to enable access to Enterprise Information System (EIS) applications as services, with the objective of leveraging the investment of existing business applications and systems. By adopting an SOA approach, EIS application functionality can be turned into reusable services that can be consumed by a new set of client applications and users.

We use CICS as a common example of an EIS to illustrate the concepts of the Service Creation scenario for direct and indirect exposure of EIS applications. There are primarily two types of CICS applications, including COMMAREA and 3270 terminal-based applications. CICS applications can be implemented with several programming languages (COBOL, PL/I, C, C++). Our focus is on COMMAREA-based CICS applications in which there is a separation of presentation and business logic.

There are several possible architecture patterns found for accessing applications hosted by the CICS Transaction Server (TS). We explore the following two common access patterns for exposing applications hosted by CICS as services:

- Directly expose existing applications as services

  In this realization, we provide an example for exposing existing COMMAREA applications hosted by CICS TS *directly* as Web services. For more information refer to 3.2, “Directly expose existing applications as services” on page 53.

- Indirectly expose existing applications with service components

  In this realization, we expose the application functionality hosted by CICS TS *indirectly* as Web services by creating middle-tier Web services to access CICS. The middle-tier Web service wraps a session EJB that uses CICS ECI Resource Adapter and CICS Transaction Gateway to access CICS TS. For more information refer to 3.3, “Indirectly expose existing applications with service components” on page 61.

3.2 Directly expose existing applications as services

In this realization, we directly expose existing applications as services. The existing application can be a wide range of application types such as an EIS (for example, CICS or IMS), J2EE application, SAP, etc. A key distinction for this

realization is that the service interface to be exposed is defined by the existing application. This approach to creating a service is known as *bottom-up*.

In this example realization, we use CICS as an example EIS. When using the CICS Transaction Server V3.1, COMMAREA applications (for example, COBOL) hosted by the CICS Transaction Server can be exposed directly as Web Services. The service can be invoked by a service consumer application within the enterprise or in some cases between enterprises. The service interface is defined largely by the existing application, thus little or no analysis is required to determine what the interface should be.

Key assumptions

The key assumptions for this realization are:

- Simple case of deploying a (provider) Web service
- Service being implemented relies on the existing application infrastructure (for example, service is implemented using CICS Transaction Server V3.1 Web services)
- WS-I Basic Profile 1.0 (WSDL, SOAP) compliant
- No development time or runtime use of Service Registry is needed

Benefits

The primary benefits for this realization are:

- The service interface is defined by that of the exposed legacy asset, thus no required analysis to determine service interface.
- Typically shorter deployment cycle
- No knowledge of other run times is necessary
- Fewer platforms or other components needed

Considerations

The key considerations for this realization are:

- Consumers become coupled to the definition of a legacy asset.
- Poor implementation substitutability because changing the implementation results in a change in the interface which will impact the consumer.
- This realization pattern requires that the runtime environment of the existing application (for example, CICS) have support for service invocation.
- This realization places an XML processing burden on systems that are often paid for on a MIPS consumed basis.

Summary of IBM products

The core IBM products used for this realization example are as follows:

- Assemble:
  - WebSphere Developer for zSeries V6
  - CICS Web Services Assistant (included with CICS TS V3.1)
- Deploy:
  - CICS Transaction Server V3.1
  - WebSphere MQ V6

Additionally, the following IBM products can be considered, depending on specific customer requirements:

- Manage: Tivoli OMEGAMON-XE for CICS V3.10
- Governance: WebSphere Service Registry and Repository

> **Note:** IBM currently intends to make available in the second half of 2006 a WebSphere service registry and repository capability that will allow customers, to securely register business services for finding, publishing and notifying changes to SOA infrastructure components such as enterprise service bus and process servers. Customers will also be able to house the metadata about business services in managing the life cycle of a service in SOA. The new capabilities will also include a model for governance to provide guidance and oversight for a SOA project.
>
> While this is the current plan, all statements regarding IBM's plans, directions, and intent are subject to change or withdrawal without notice.

Figure 3-1 on page 56 highlights the logical architecture components of the SOA Foundation and IBM products used for this realization example.

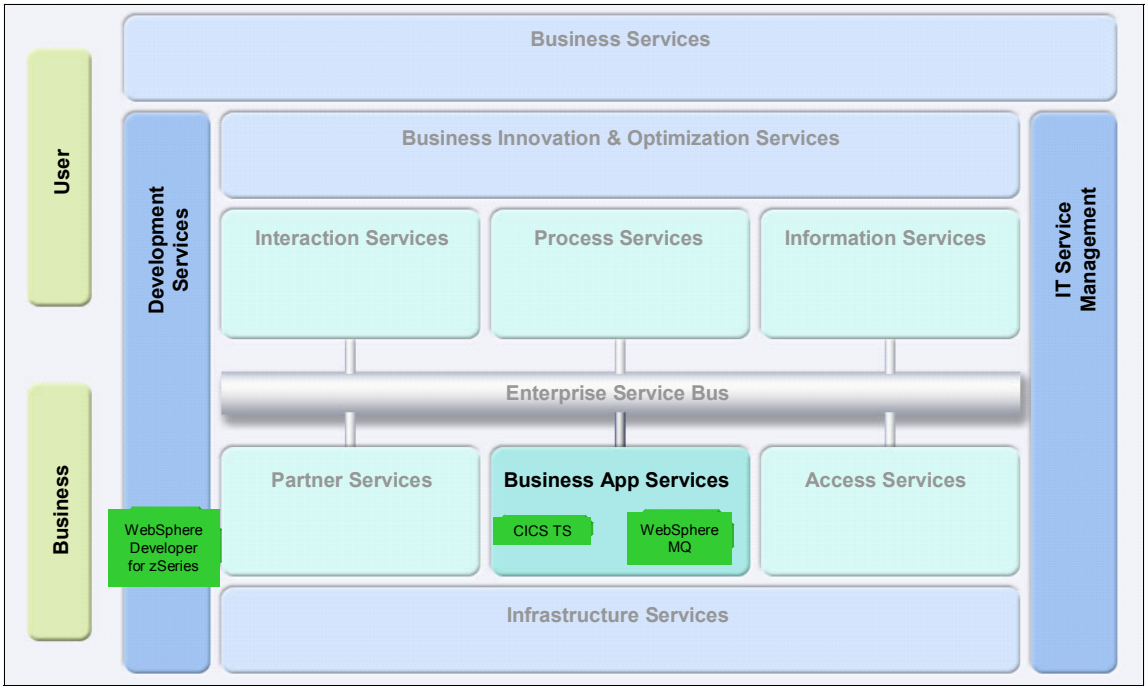


Figure 3-1 SOA Foundation logical architecture components and product mapping: Directly expose existing applications as services realization example

3.2.1 Component and integration architecture

Figure 3-2 on page 57 displays the layers of the component architecture as well as the relationships between components. In this realization, we focus on directly exposing CICS applications at the operational systems layer. This is achieved by creating an atomic service to access CICS. The implementation of an atomic service is self-contained, in that does not invoke other services. The atomic service is a Web Service hosted by the CICS Transaction Server, which can be invoked by a Web Services client application.

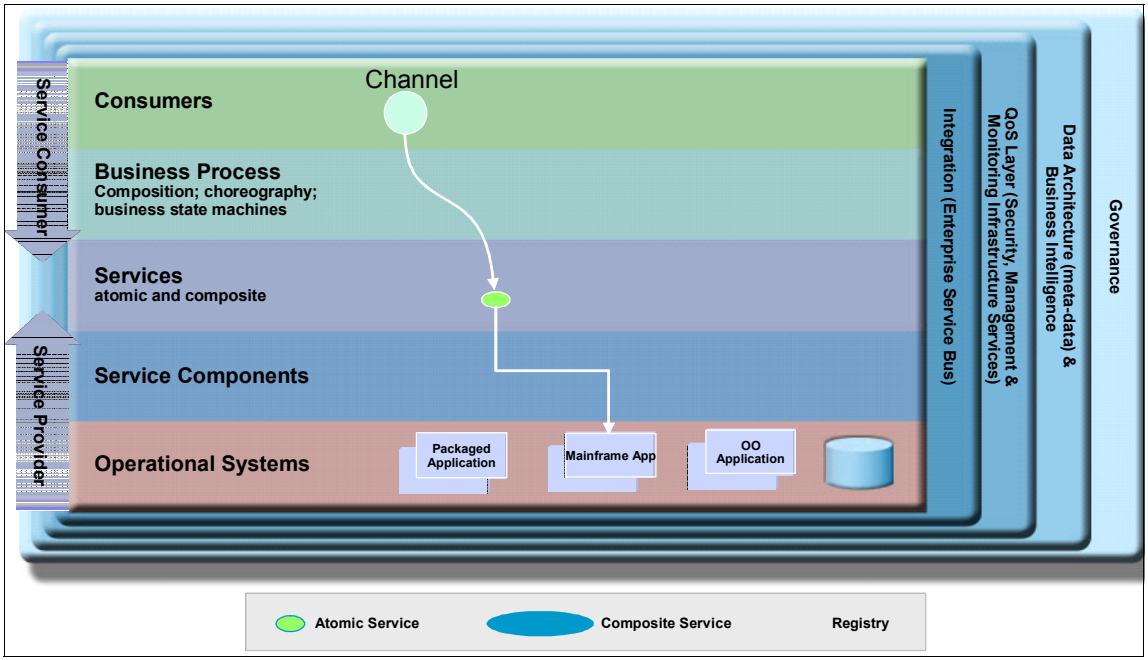


Figure 3-2 Directly expose existing applications as services realization: Solution view

Figure 3-3 on page 58 depicts the direct exposure of applications hosted by CICS Transaction Server (TS) V3.1 as Web Services. Service consumers can invoke the Web services using SOAP over HTTP(s) or MQ protocols, provided WebSphere MQ is installed on z/OS®.

CICS TS V3.1 supports Web Services Security (WS-Security) used to secure the SOAP message (authentication, integrity, confidentiality), and transport level security using SSL.

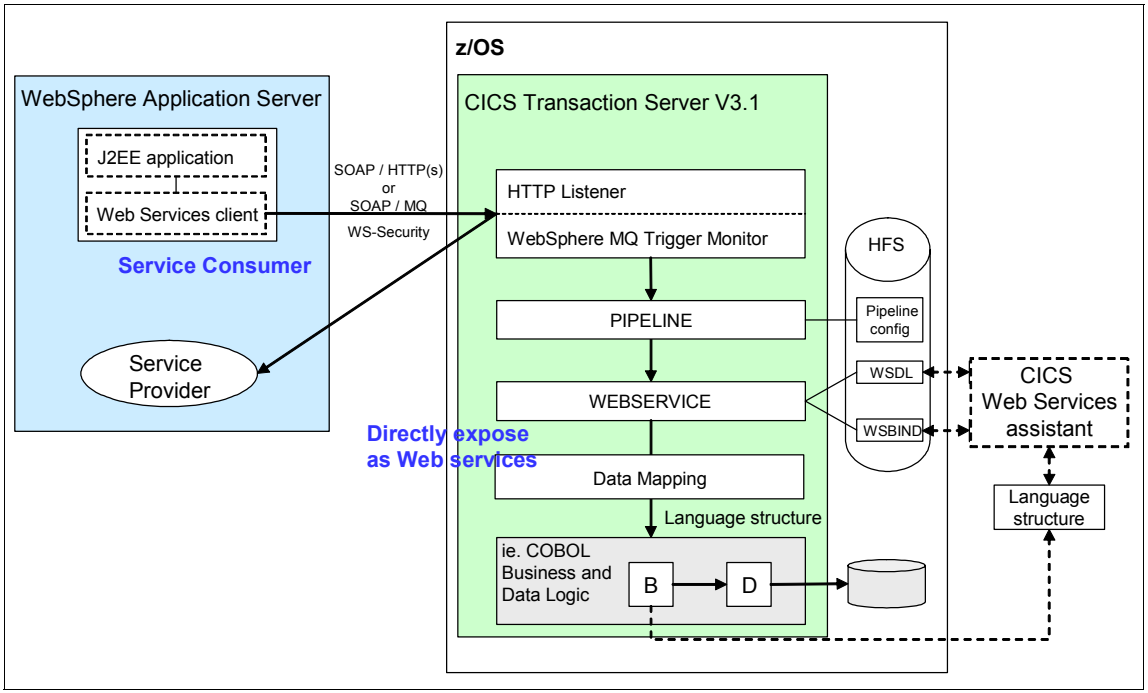


Figure 3-3 Directly expose existing applications hosted by CICS TS as Web services

3.2.2 Key tasks and IBM products for the SOA life cycle

This section describes the key tasks performed in the SOA life cycle phases using IBM products defined for this example realization.

Model

The Service Creation scenario does not require a specific IBM product for the model phase of the SOA life cycle. Methodologies can be used for modeling the business and requirements, such as the Rational Unified Process (RUP) for SOA or IBM Method. This also includes the use of the Patterns for e-business, and Service-Oriented Modeling and Architecture (SOMA), which provides techniques to identify possible service candidates.

If the customer has specific requirements for business modeling that require tooling, IBM Rational Software Architect is the recommended IBM product.

Assemble

There are two key tools available to assemble Web Services for CICS for the direct exposure realization, including the CICS Web Services Assistant

packaged with the CICS Transaction Server V3.1, and WebSphere Developer for zSeries V6.

Note: For more information on the CICS Web Services Assistant and WebSphere Developer for zSeries refer to the following:

- Summary information:
  - “CICS Web Services Assistant” on page 479
  - “WebSphere Developer for zSeries” on page 479
- Detailed information can be found in the redbook *Application Development for CICS Web Services*, SG24-7126.

There are three approaches for developing Web Services for CICS including, bottom-up, top-down, and meet-in-the-middle. For the direct exposure of existing CICS applications, we use a bottom-up approach.

We highlight the use of the CICS Web Services Assistant and WebSphere Developer for zSeries tooling for the bottom-up approach. We use the CICS Web Services Assistant to generate the service binding and WSDL, and use WebSphere Developer for zSeries to perform unit testing. Note, the WebSphere Developer for zSeries V6.0.1 includes a CICS Transaction Server test environment for Windows®.

The high level steps are as follows:

1. Retrieve the COBOL artifacts (copybook, COBOL program) from the source code converter (SCC).
2. Import the artifacts into the appropriate tooling. For example, import into a WebSphere Developer for zSeries project.

   Start with the language structures in the existing target CICS application. There might be a need to simplify these structures to suit the tooling.
3. Generate the following in WebSphere Developer for zSeries:
   - Input and output XML converters
   - Driver program
   - Service WSDL

   Alternatively input the existing, or refined, language structures to the DFHLS2WS utility included with the CICS Web Services Assistant to generate a new WSDL file that describes the Web service.

   The binding file will be used at deployment/runtime to handle the mapping of the COMMAREA application.
4. The Web Service for the CICS application can be deployed to either the CICS TS test environment or CICS TS on z/OS. The deployment includes

publishing the generated WSDL, copying the generated COBOL source to z/OS, compiling and preparing the generated source code to be invoked as a Web Service.

To create the a J2EE based Web services client application that invokes the service:

1. Import the WSDL service definition into WebSphere Developer for zSeries.
2. Generate a client proxy from the WSDL.
3. Develop the Web service client application which is used to invoke the service.
4. The client application in this case is deployed to WebSphere Application Server.

Deploy

In this realization example, CICS Transaction Server V3.1 is used to provide the Web Services support to host directly exposed Web Services. CICS TS V3.1 includes many important features for Web Services such as support for Web Services Security. If you plan to use the MQ protocol between the Web services client application and the Web services, you must install WebSphere MQ on z/OS.

The Web Services application deployment step was described at the end of the assembly tasks previously.

Manage

The manage life cycle phase includes management and security. Depending on the customer specific requirements, you may wish to use Tivoli OMEGAMON-XE for CICS V3.10 to manage and monitor CICS TS on z/OS.

Note: For more information refer to the *IBM Tivoli OMEGAMON XE V3.1 Deep Dive on z/OS*, SG24-7155 redbook.

Security is a very important dimension of the scenario, but is largely out of the scope of this book. CICS TS V3.1 includes support for Web Services Security, which provides message level security for authentication, integrity and confidentiality. In addition, SSL can be used to secure the transport when using HTTPS. CICS TS in this case is likely to be configured to use RACF® for authentication and authorization purposes. Note, there are several Tivoli products that can be used to bolster the security and integration such as Tivoli Federated Identity, Tivoli Access Manager, and Tivoli Identity Manager.

Note: For more information on CICS security architectures refer to the IBM Redbook *Revealed! Architecting e-business Access to CICS*, SG24-5466.

3.3 Indirectly expose existing applications with service components

For this realization, we expose existing application functionality *indirectly* using service components. In this realization, business alignment is achieved by defining the service interface (WSDL), independently of existing assets. This approach to creating a service is known as top-down. In practice, the implementation of the service may be a combination of top-down and meet in the middle.

When using the techniques found in Service-Oriented Modeling and Architecture (SOMA), top-down analysis is used to define a service independently of existing assets. This approach helps to define better what a service should do or how it should look, based on business domain analysis and decomposition. A second phase of service identification called *existing asset analysis* is used to identify opportunities for reuse that might exist among existing programs and so forth.

Service components provide the linkage between the domain-driven service definition and the reuse of existing assets. The service component interface is derived from the business driven service description while the implementation of the service component takes advantage of any existing assets that have been identified for reuse. The implementation typically aggregates existing behavior using adapters or connectors. This usually involves some refactoring of the existing assets.

Here, we use CICS as an example of an existing EIS. In this realization example, we expose existing COMMAREA applications hosted by CICS Transaction Server *indirectly* by creating middle-tier Web services to access CICS. The middle-tier Web service wraps a session EJB that uses the CICS ECI resource adapter to communicate with the CICS Transaction Gateway (CTG) to access CICS Transaction Server. The CICS ECI resource adapter is a JCA adapter for WebSphere Application Server packaged with the CTG.

Key assumptions

The key assumptions for this realization are:

- Simple case of deploying a Web service
- Service implementation is J2EE V1.4, JSR 109
- Solution uses CICS TS 2.x and CTG

- Conforms to WS-I Basic Profile 1.0 (WSDL, SOAP)
- No requirement for using a service registry
- Message payload size maximum of 32K when using CTG

Benefits

The primary benefits for this realization are as follows:

- Business alignment is maintained by defining a service interface that suits or aligns with the business view and not with existing application assets. The service component maps between the two worlds.
- Supports implementation substitutability because service component may be replaced without impact on the consumer.
- Off loads the XML processing burden from systems that are often paid for on a MIPS consumed basis. (assuming the service component runtime is not co-located with the backend system)
- A service may be implemented using behavior from more than one system, the service component aggregates the behavior to realize the service.

Considerations

The key considerations for this realization are as follows:

- Longer deployment cycle than the direct exposure realization because thought must be given to the definition of the service interface and time must be spent developing the service component.
- More complex than the direct exposure realization in that it often involves the use of an adapter or connector technology between the service component and the existing backend systems.

IBM products

The core IBM products used for this realization are:

- Assemble: Rational Application Developer V6
- Deploy:
  - WebSphere Application Server V6 (z/OS or distributed platform depending on selected topology)
  - CICS Transaction Gateway (V6.1 on z/OS, V6.0.1 on distributed depending on selected topology)
    Note: CICS ECI resource adapter for WebSphere is included with CTG
  - CICS TS V2.x (assumes this is already installed)

> **Note:** The realization example assumes CICS TS V2.x is used, however the example can also be applied to CICS TS V3.1.

Additionally, the following IBM products can be considered, depending on specific customer requirements:

- Model: Rational Software Architect V6
- Manage:
  - Tivoli OMEGAMON XE for CICS V3.10
  - Tivoli Composite Application Manager for SOA V6
  - Tivoli Composite Application Manager for WebSphere V6
- Governance: WebSphere Service Registry and Repository

Figure 3-4 highlights the logical architecture components of the SOA Foundation and IBM products used by the indirect exposure realization.

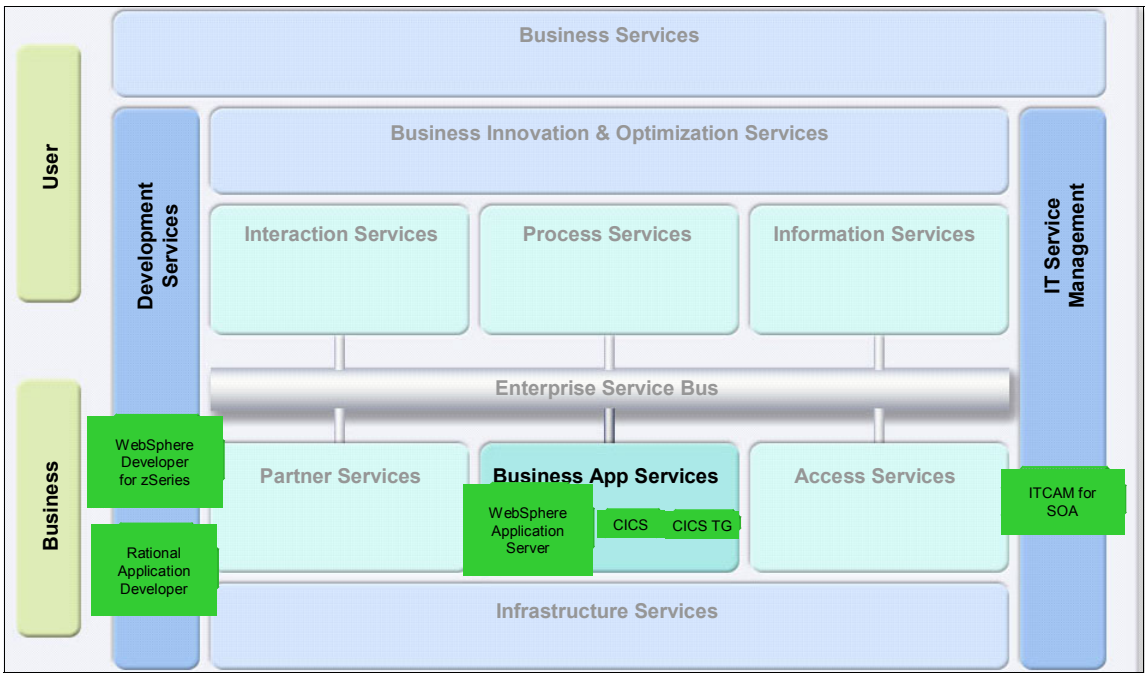


Figure 3-4 SOA Foundation logical architecture components and product mapping: Indirectly expose existing applications with service components realization example

3.3.1 Component and integration architecture

As depicted in Figure 3-5 on page 64, the indirect exposure of existing applications via service component decouples the service interface from implementation details. In the indirect exposure realization, an atomic service is created to access a Web service in the service components layer, which wraps a session EJB that uses CICS ECI resource adapter (JCA adapter) and CTG to access CICS TS in the operational systems layer.

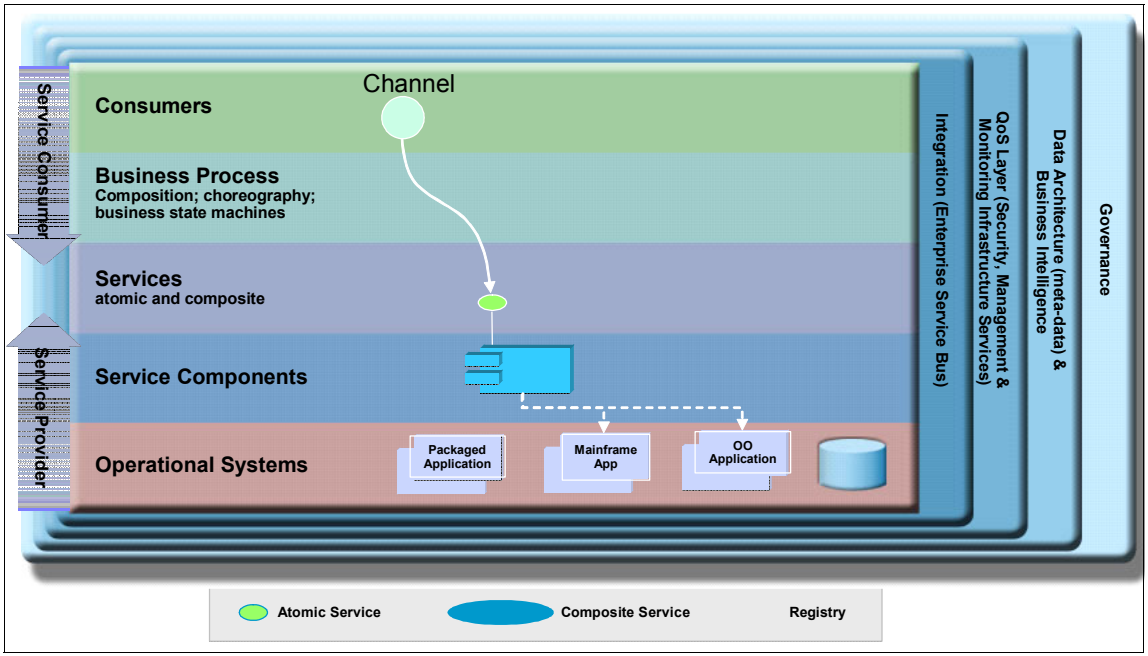


Figure 3-5 Indirectly expose existing applications with service components realization: Solution view

There are three possible runtime topologies that may be used to implement the indirect exposure access pattern for CICS TS:

- Topology 1: WebSphere Application Server and CTG on distributed
- Topology 2: WebSphere Application Server on distributed and CTG on z/OS
- Topology 3: WebSphere Application Server on z/OS

From an application development standpoint, the middle-tier Web service is assembled in Rational Application Developer (or WebSphere Developer for zSeries) the same manner for each of the three topologies. We will highlight some of the differences in the IBM products used in the runtime environment the application is deployed and managed.

The term distributed is used to describe the distributed platform supported by WebSphere Application Server and CTG such as AIX®, Windows, Linux®, etc. This example realization is focused on AIX as the distributed platform.

Topology 1: WebSphere Application Server and CTG on distributed

This topology can be use if the goal is to off load processing to an external node from the z/OS system where CICS TS is installed. This topology is not as common in production as topology 2 and 3.

- Node A:
  - AIX, WebSphere Application Server V6, Web Services application (session EJB / ECI resource adapter), CICS Transaction Gateway V6.0.1
- Node B:
  - z/OS, CICS Transaction Server V2.x

Topology 2: WebSphere Application Server on distributed and CTG on z/OS

When installing CTG on the same z/OS node as CICS TS, the CTG can communicate directly with the CICS TS using EXCI for faster processing. CICS TG V6.1 for z/OS includes a ECI resource adapter (ECIciceciXA.rar) that supports 2-phase transactional commit.

- Node A:
  - AIX, WebSphere Application Server V6, Web Services application (session EJB / CICS ECI resource adapter)
- Node B:
  - z/OS, CICS Transaction Gateway V6.1, CICS Transaction Server V2.x

Figure 3-6 displays topology 2 with the addition of the management components.

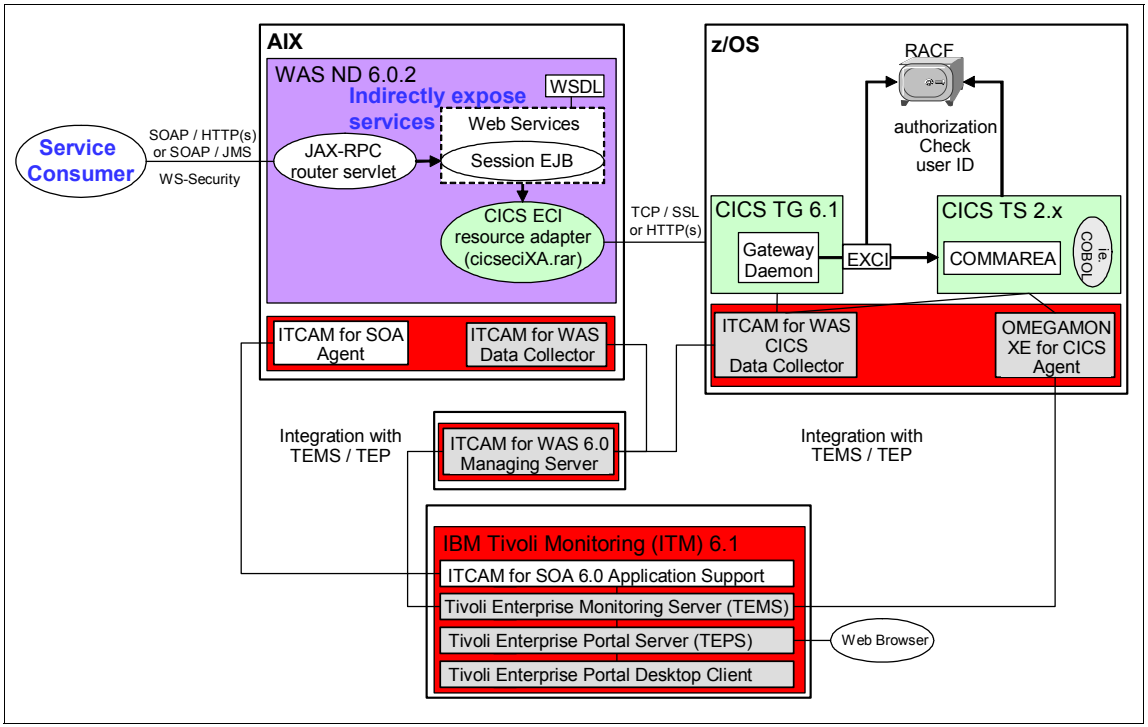


Figure 3-6 Topology 2: WebSphere Application Server on distributed and CTG on z/OS for the indirect exposure realization

Topology 3: WebSphere Application Server on z/OS

Topology 3 includes the benefits of topology 2, and also has the added benefit of centralized systems management, and optimized performance. In topology 3, the middle-tier Web services application and middleware reside on z/OS.

Figure 3-7 displays topology 3 with the addition of the management components.

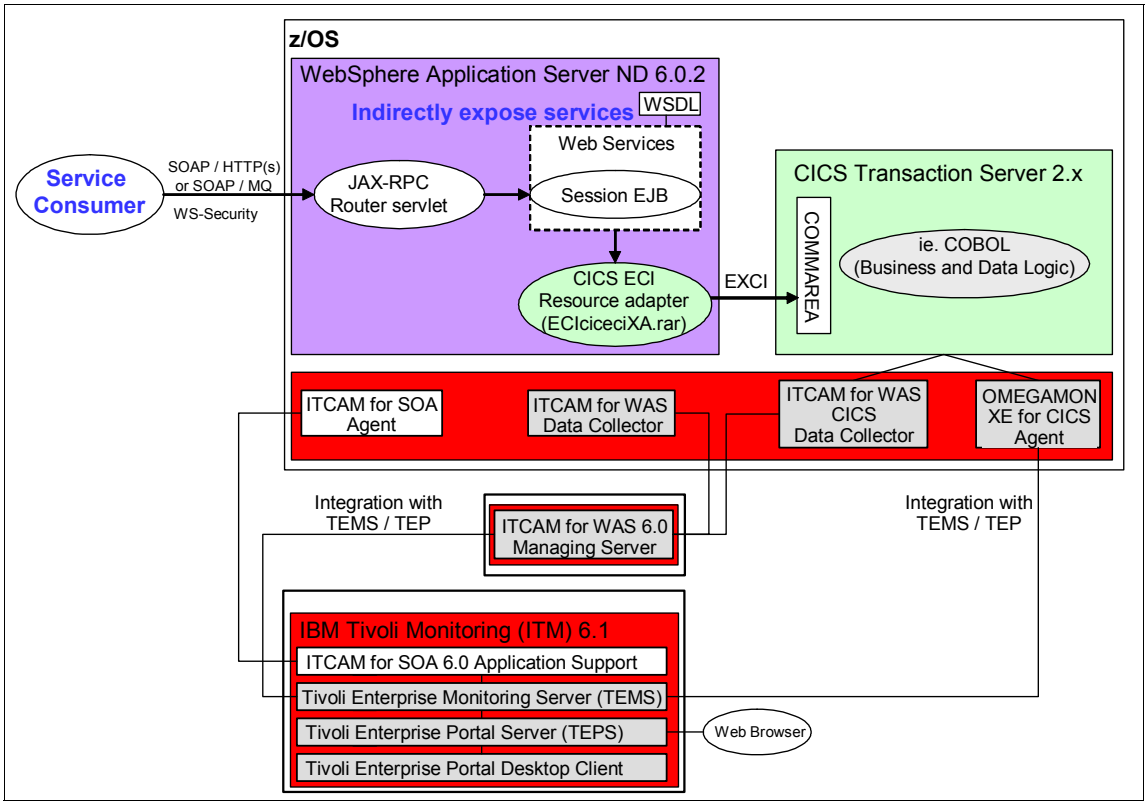


Figure 3-7 Topology 3: WebSphere Application Server on z/OS for the indirect exposure realization

3.3.2 Key tasks and IBM products for the SOA life cycle

This section describes the key tasks performed in the SOA life cycle phases using IBM products defined for this example realization. For the most part, the assemble tasks and tooling are the same for topology 2 and 3. We highlight the differences between topology 2 and 3 for the deploy and manage phases.

Assemble

The indirect access pattern assumes that the service definition (WSDL) does not mirror the interface exposed by the CICS asset being reused. The WSDL determines what the service looks like and is either designed from scratch, or preexists and is used to generate the service component skeleton.

Rational Application Developer V6.0.1 is the core IBM product used to assemble the application artifacts for the indirect exposure realization.

Tip: The combination of WebSphere Developer for zSeries V6.0.1 and the bundled CICS Transaction Server for WebSphere Developer test environment, and the CICS Transaction Gateway V6.0.2, provide an incredibly powerful environment for developing and uniting testing indirect exposure Web services for CICS application. For this reason, WebSphere Developer for zSeries may also be considered for the assemble tasks depending on your requirements.

The key assemble tasks for indirect exposure are:

1. Create a dynamic Web project in Rational Application Developer.
2. Define the service.

   The identification and modeling of the service can be a lengthy process.

   – If the service is not defined, create the WSDL within Rational Application Developer.

   – If the service is already defined and a WSDL exists, import the service WSDL into Rational Application Developer.
3. Generate the session EJB Web Service skeleton from the service WSDL file.

   Using the skeleton, complete all the methods with the appropriate logic in Rational Application Developer. The service definition should have a mapping of the Java data binding (CICS COBOL to Java).

   This includes the use of the CICS ECI resource adapter.

   – Use the Create J2C JavaBean wizard to create a bean, which means you do not have to write any code yourself to use the CCI interface to the CICS Transaction Gateway to call the program on CICS.

   – Modify session EJB code to invoke the resource adapter to access CICS through CTP.

Package tasks to prepare artifacts for deployment:

- Export enterprise application to an EAR file for deployment
- Obtain and prepare WSDL file(s) to be used by Service Registry (optional)

Deploy

We consider both topology 2 and 3 when describing the IBM products used in the deploy life cycle phase.

Topology 2: WebSphere Application Server on distributed and CTG on z/OS

In this topology, WebSphere Application Server Network Deployment Edition V6 and the CICS ECI resource adapter are installed on a distributed platform such

as AIX. The CICS ECI resource adapter (ECIciceciXA.rar) must be added to the application server classpath. CTG and CICS TS are installed on z/OS.

From an application deployment standpoint, the middle-tier Web service is deployed on WebSphere Application Server installed on AIX.

Topology 3: WebSphere Application Server on z/OS

In this topology, WebSphere Application Server is installed on z/OS. The CTG daemon is not needed in this case. The CICS ECI resource adapter can directly communicate with CICS TS using EXCI.

From an application deployment perspective, the Web service and session EJB application are deployed on WebSphere Application Server installed on z/OS.

Manage

The manage life cycle phase includes management and security. Depending on the customer specific requirements, you may wish to use following Tivoli products to manage and monitor the indirect exposure solution:

- Tivoli OMEGAMON-XE for CICS V3.1 to manage and monitor CICS TS on z/OS.
- Tivoli Composite Application Manager for SOA to manage and monitor services.
- Tivoli Composite Application Manager for WebSphere to manage and monitor J2EE applications.

Ensuring that the CICS transaction is secure is a critical dimension to the indirect exposure realization. The security domain includes such security topics as authentication, authorization, auditing, identity management, and user provisioning. WS-Security can be employed between the client application and the middle-tier Web service. In addition, SSL can be used to secure the transport.

Depending on customer requirements, you might want to use the following Tivoli products to bolster the security infrastructure for the realization:

- Tivoli Federated Identity Manager provides a simple, loosely-coupled model for managing identity and access to resources that span companies or security domains.
- Tivoli Access Manager is used to provide single sign-on (SSO) authentication and authorization solutions.
- Tivoli Identity Manager can be used to provision users to different systems across the enterprise. For example, provision users to both an LDAP directory on AIX and RACF on z/OS.

We have provided further details on flowing the user identity from end-to-end for both topology 2 and 3. This is a common requirement for businesses that require the ability to audit the users identity through the transaction (for example, financial transaction).

Topology 2: WebSphere Application Server on distributed and CTG on z/OS

In topology 2, WebSphere Application Server is likely to be configured with WebSphere security and an LDAP directory such as Tivoli Directory Server. From an authentication standpoint, WebSphere Application Server supports basic authentication (user ID, password), LTPAtoken, and certificates. In an enterprise environment, it is likely that authentication will occur using the Tivoli Access Manager WebSEAL or a DataPower appliance.

The high-level authentication steps within WebSphere Application Server are:

1. Authentication information such as basic authentication (user ID and password) or credential token (LTPAtoken) are received either the EJB authentication module or the Web authentication module.
2. The EJB or Web authentication modules pass the information to the Java Authentication and Authorization Service (JAAS) login module.
3. Depending on the configuration, the JAAS login module uses either the Lightweight Third-Party Authentication (LTPA) or Simple WebSphere Authentication Mechanism.

   It is possible to create a custom JAAS login module to suit you login processing needs.
4. The authentication module uses the configured registry (LocalOS, LDAP, or custom registry) to validate the authentication information.
5. Once authenticated, the login module create the JAAS subject and stores the Common Object Request Broker Credential (CORBA) credential derived from the authentication data in the public credentials list of the subject.

   This credential will be used by the authorization service to perform further access to resources.

When user identities exist in more than one system, the issue of mapping user identities arises. If you have the ability to provision the users with a common user ID you can partially alleviate some of the challenges.

There are two fundamental approaches to configuring security between distributed WebSphere Application Server and CICS using the ECI resource adapter.

- Component managed authentication

  This approach requires that the user ID and password be supplied as parameters for the getConnection method within the application code.

  The upside of this approach is that the user ID can be flowed to CICS for transaction auditing purposes. If the user IDs are different between the LDAP used by WebSphere Application Server and RACF used by CICS, then this option will not work. Also, this option requires that security be implemented at the application level rather than at the infrastructure level.

- Container managed authentication

  This approach removes the requirement that the application programatically supply the credentials for accessing CICS. The getConnection method is called with no arguments. The credentials used to connect to CICS are supplied the JAAS - J2C authentication alias (user ID and password) at the container level.

  The advantage to this approach is that the application does not need to be modified. The disadvantage is that one user (JAAS - J2C authentication alias) is used to establish the connection to CICS. This does not lend itself to auditability for a transaction for the distributed topology.

  Note: The principle mapping module as shipped by WebSphere will flow a J2EE identity as a connection identity to a local CTG connection to CICS but not to a remote CTG connection.

  In topology 2 CTG is installed on z/OS. To address the issue mentioned in the above notebox, you can create a customized principle mapping module which flows a J2EE identity as the connection identity for both local and remote CTG connections.

  Tivoli Access Manager, Tivoli Federated Identity Manager, and Tivoli Identify Manager can also be used to provide a more complete and manageable security solution.

Note: For more information on WebSphere Application Server V6 security refer to the following resources:

- *WebSphere Application Server V6 Security Handbook*, SG24-6316 redbook
- WebSphere Application Server V6 InfoCenter at:

  http://www.ibm.com/software/webservers/appserv/was/library/library60.html

Topology 3: WebSphere Application Server on z/OS

In topology 3, all components are installed on z/OS. It is likely for this topology that both WebSphere Application Server and CICS are configured to use RACF.

When using container-managed authentication, a z/OS system-specific functionality known as *thread identity* support is provided by WebSphere Application Server. Thread identity allows the application server to automatically pass the user ID of the thread (caller's identity) to CICS when using the ECI resource adapter. The combination of features on z/OS simplify the authentication, authorization and identity security implementation.

Note: If WebSphere Application Server is configured to use a different registry than CICS, then the identity issues discussed in the distributed topology would also apply to this topology.

3.4 Create an EJB Web service from WSDL

In this realization, we create a new session EJB Web service from an existing service WSDL. This realization is also known as *create from scratch*. This realization uses a top-down approach to creating a Web Service.

Key assumptions

The key assumptions for this realization are as follows:

- Service implementation is J2EE V1.4, JSR 109
- Conforms to WS-I Basic Profile 1.0 (WSDL, SOAP)
- No requirement for using a service registry
- XML document message size averages 20 K in-bound, and 5 K out-bound
- Relatively low transaction rate (for example, one transaction per second)

IBM products

The core IBM products used for this realization are as follows:

- Assemble: Rational Application Developer

- Deploy: WebSphere Application Server Network Deployment
- Manage: Tivoli Composite Application Manager for SOA

Additionally, the following IBM products may be considered depending on specific customer requirements:

- Model: Rational Software Architect V6
- Manage:
  - Tivoli Composite Application Manager for WebSphere

  For security purposes, the following Tivoli products can be used:

  - Tivoli Access Manager
  - Tivoli Directory Server
- Governance: WebSphere Service Registry and Repository

Figure 3-8 highlights the logical architecture components of the SOA Foundation and IBM products used for this realization example.

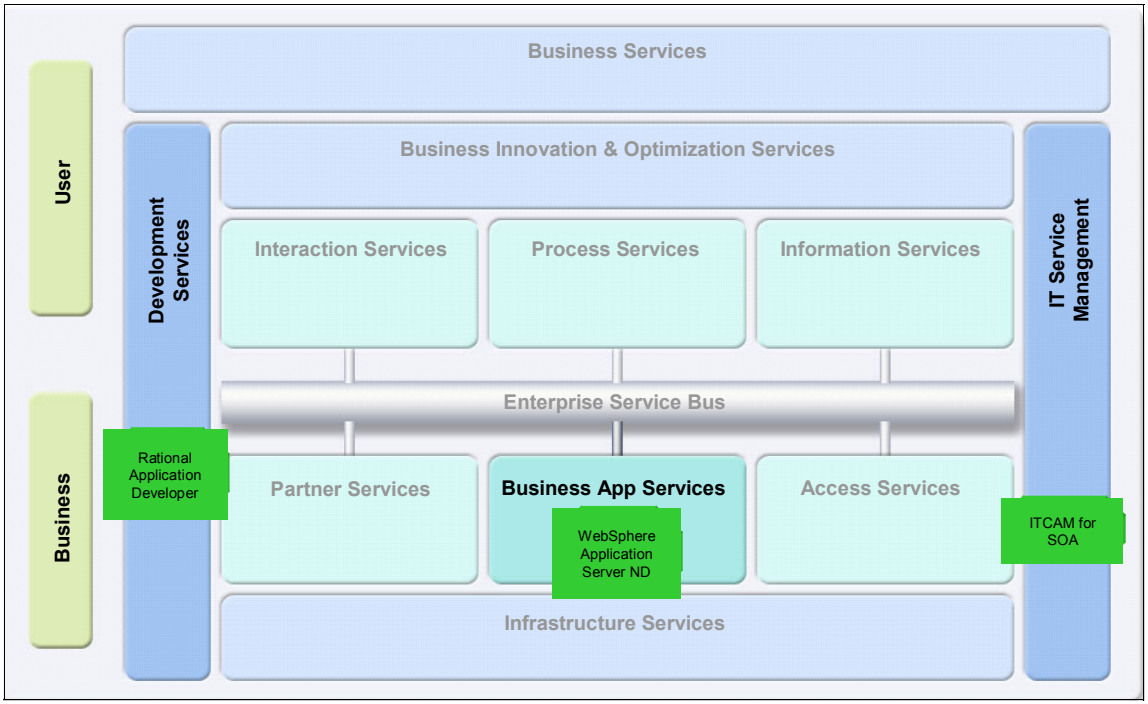


Figure 3-8 SOA Foundation logical architecture components and product mapping: Create an EJB Web service from WSDL realization example

3.4.1 Component and integration architecture

This section describes the key components and integration architecture for this realization example. In this realization, we start with a WSDL service definition and create a new session EJB based Web service from scratch.

Figure 3-9 on page 74 depicts the architecture layers. This realization relies upon a J2EE application server (for example, WebSphere Application Server) to support for service invocation.

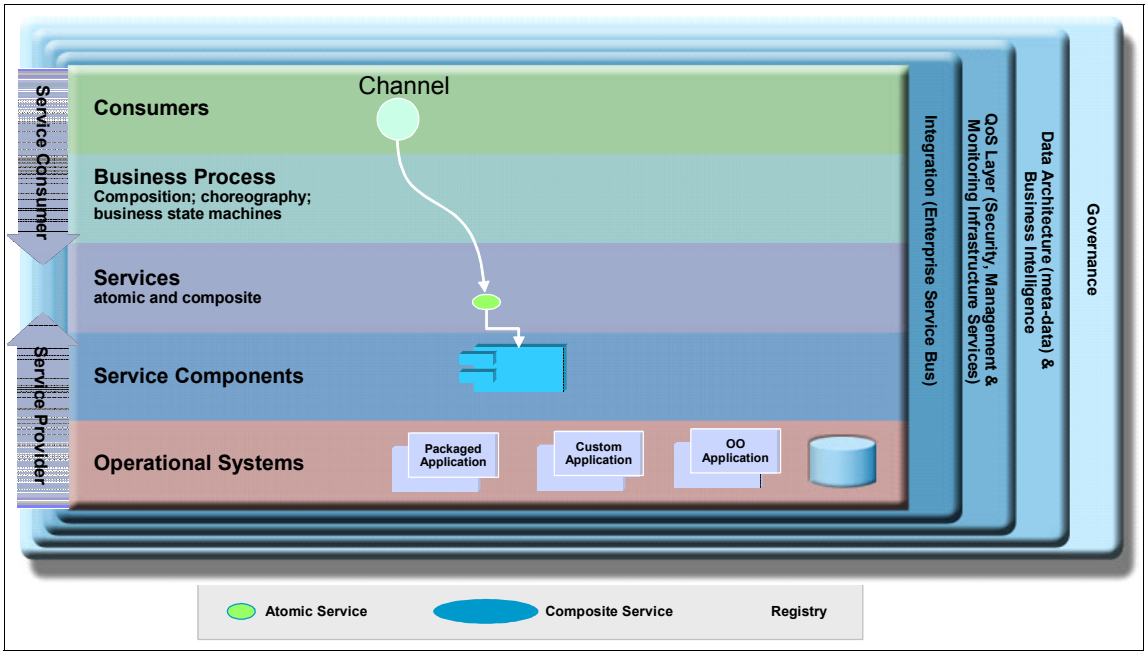


Figure 3-9 Create an EJB Web service from WSDL realization: Solution view

Figure 3-10 on page 75 depicts the component and integration architecture for the create a new session EJB Web Service from a WSDL definition realization.

Notice in Figure 3-10 on page 75, the internal J2EE client can invoke services using SOAP over HTTP, and SOAP over JMS (provided the corresponding Web service wraps a message driven bean). Although technically possible, SOAP over JMS is not typically used between enterprises.

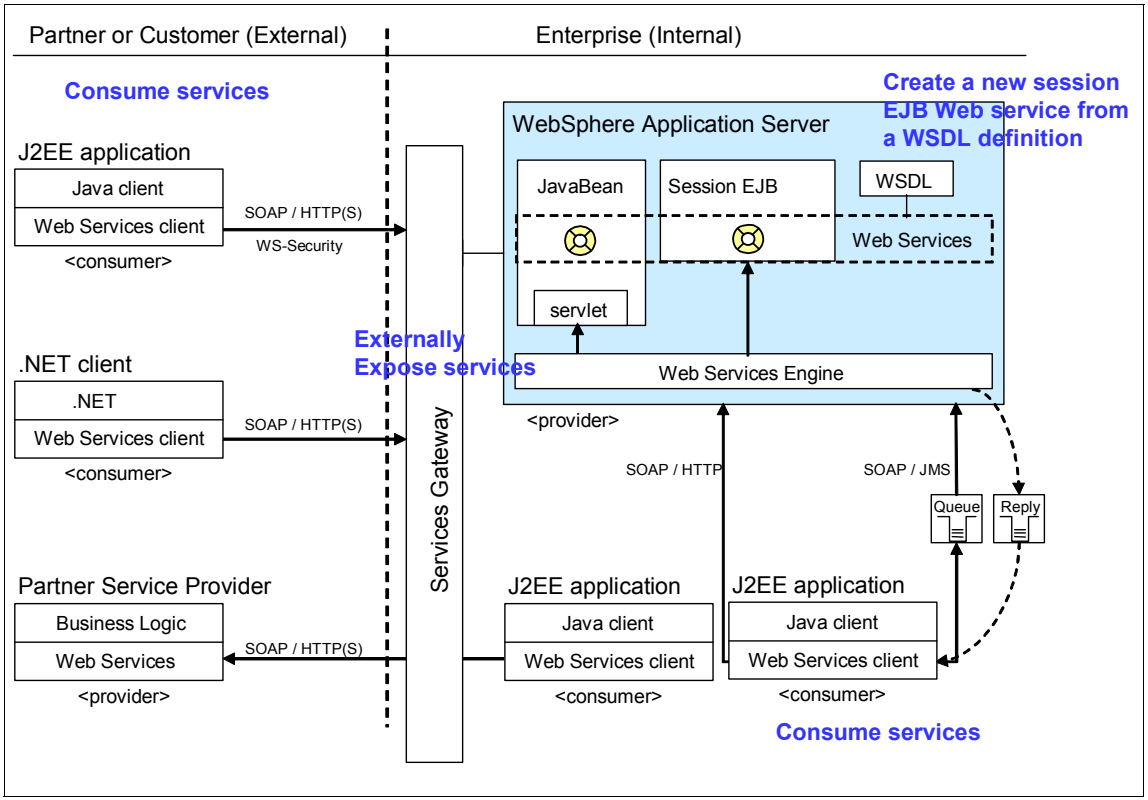


Figure 3-10 Integration architecture for the Create an EJB Web service from WSDL realization

3.4.2 Key tasks and IBM products for the SOA life cycle

This section describes the key tasks performed in the SOA life cycle phases using the core IBM products defined for this example realization.

Model

The Service Creation scenario does not require a specific IBM product for the model phase of the SOA life cycle. Refer to “Model” on page 58 for more information.

Assemble

Web services for J2EE based applications are *assembled* with IBM Rational Application Developer V6. This includes using the Application Developer tooling to create Web services, J2EE application development, and package the application for deployment. In this realization, we start with a WSDL document

used to define the service. This approach to creating a Web service is known as top-down.

Note: For more detailed information on creating Web Services for J2EE based applications refer to the following:

- *WebSphere Version 6 Web Services Handbook, Development and Deployment*, SG24-6461
- *Rational Application Developer V6 Programming Guide*, SG24-6449

The WSDL file, is an XML based interface and implementation description language. The service provider uses the WSDL file to specify the operations the Web service provides, as well as the parameters and data types for the operations.

There are primarily two types of J2EE artifacts exposed as Web services:

- Java classes such as JavaBeans™ and servlets (Web container)
- EJB session beans and message driven beans (EJB container)

Note: Our focus in this realization example is on creating session EJB Web services from an existing WSDL.

In a top-down approach, a skeleton implementation of a Web service is created from an existing WSDL file. Rational Application Developer can generate the skeleton session EJB (or JavaBean).

Note: A potentially lengthy process might be required for service identification and validation.

The high-level steps used to create a new session EJB Web service from scratch using an existing WSDL file are:

1. Create a project in Rational Application Developer (for example, enterprise application project).

   The project will be used to import the WSDL and later create the session EJB Web service.

2. Import the existing service WSDL file into the project.

   Alternatively, create the WSDL file.

3. Generate the session EJB Web Service skeleton.

4. Using the skeleton, complete all the methods with the appropriate logic in Rational Application Developer.

The Web services client application is assembled with Rational Application Developer. This includes generating the Web service client from the Web Service WSDL, and adding the code to invoke the service within the Java code.

Deploy

The primary software used as the runtime where the Web Services application is deployed for this realization example is IBM WebSphere Application Server Network Deployment Edition V6.

Note: For information on using WebSphere Application Server Network Deployment Edition to deploy Web Service provider and consumer applications refer to:

- Chapter 9, “Implement the runtime environment” on page 281.
- Chapter 10, “Deploy application to WebSphere Application Server” on page 351

Once the development task of creating the Web service is complete, the enterprise application can be exported from Rational Application Developer to an EAR file used for deployment to WebSphere Application Server. The WSDL can optionally be published to a service registry. At this stage, the service is available to be consumed.

Once the Web services client application is complete, it can be exported (for example, EAR or WAR) and deployed to WebSphere Application Server. The client application can now be run to consume services.

If your solution requires that partners and customers have access to services, then you will need to expose the services in a secure manner. Within WebSphere Application Server Network Deployment Edition, the Web Services Gateway can be used to selectively expose services. Other services gateways can be used as well such as WebSphere ESB or DataPower. For more information refer to *Patterns: SOA Foundation - Service Connectivity Scenario*, SG24-7228 redbook.

Web Services security can be enabled (provider and consumer) to secure the authentication, integrity and confidentiality of the SOAP messages sent between the enterprise and the partner. Also, SSL can be enabled to secure the transport used between the enterprise and partner.

After the services have been exposed externally, authorized partners or customers can consume the services. From a technology stand point, Web services client application outside the enterprise is virtually identical to Web services clients within the enterprise.

When exposing services externally for multiple partners or customers to consume the services, it is more likely different Web services clients will be implemented, such as J2EE client and .NET client.

Lastly, Web services clients within the enterprise can consume services outside the enterprise as depicted in Figure 3-10 on page 75.

Manage

The primary IBM product used to manage and monitor services in this example realization is Tivoli Composite Application Manager for SOA. In addition, Tivoli Composite Application Manager for WebSphere can be used to manage and monitor the J2EE application server and application.

From a security perspective, WS-Security can be used to secure the SOAP messages over HTTP(s), and SSL can be used to secure the HTTP transport. In addition, there are several Tivoli products that can be used to bolster the security and integration such as Tivoli Federated Identity, Tivoli Access Manager, and Tivoli Identity Manager.

Note: For information on managing and monitoring using IBM Tivoli Composite Application Manager for SOA, refer to the following:

- Chapter 12, “Manage and monitor services with ITCAM for SOA” on page 405
- *IBM Tivoli Composite Application Manager V6.0 Family: Installation, Configuration, and Basic Usage*, SG24-7151

3.5 Consume services from third-party service providers

In this realization, client applications consume services from third-party service providers. This realization represents the view of a consumer that is using one or more third-party services. The service consumer only sees the service interfaces. The endpoints and any security or transport constraints are imposed by the service provider.

For example, a Web services client application may invoke an address verification Web service from a third party service provider.

Key assumptions

The key assumptions for this realization are:

- WSDL is WS-I compliant and JAX-RPC compatible.
- Service provider is outside the enterprise firewall and security is implemented between the consumer and provider using mutual SSL authentication.

Considerations

The key considerations for this realization are:

- Providers of third-party services strive to minimize churn in the service interface in that changes often breaks the consumer. They need to have a plan for service evolution that is well communicated to the consumer in advance, particularly when that evolution changes features of the service on which consumers depend.
- To reach the broadest market, the provider should expose their services using open standards and, where applicable, industry standard interfaces and message formats. For some industry standard formats, this might pose a problem because XML Schema mapping technology is still evolving.

IBM products

The core IBM products used for this realization are:

- Assemble: Rational Application Developer V6
- Deploy: WebSphere Application Server V6

The additional products used for the realization are as follows, depending on customer requirements:

- Manage:
  - Tivoli Composite Application Manager for SOA V6
  - Tivoli Composite Application Manager for WebSphere V6
- Governance: WebSphere Service Registry and Repository

3.5.1 Component and integration architecture

The logical architecture layers for Consume services from 3rd party service providers are depicted in Figure 3-11 on page 80. Service consumers create Web services client applications and use a Service Registry to discover the services from the third-party service providers. The implementation of the services from the third-party service provider could be direct using Web services or using service components.

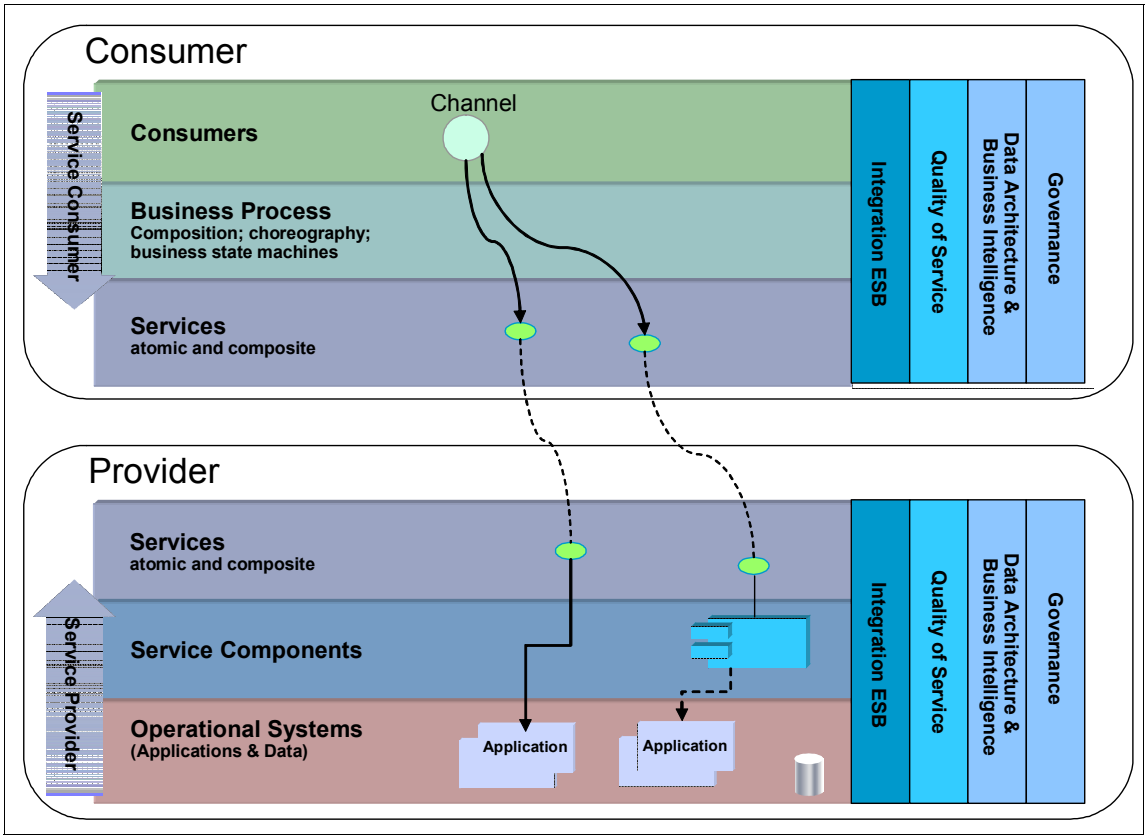


Figure 3-11 Consume services from 3rd party service providers realization: Solution view

3.5.2 Key tasks and IBM products for the SOA life cycle

This realization is focused on consuming services. We have covered much of the information regarding the life cycle phases where the client application would be deployed and managed. This section will highlight the tasks required to assemble a Web services client application.

Web services for J2EE specifies three different types of JAX-RPC clients:

- Static stub
- Dynamic proxy
- Dynamic invocation interface

The task to build or generate a Web service client (also known as a service requestor) depends on the methods of how the client is binding to a Web service server. The client uses a local service stub or proxy to access the remote server

and service. The WSDL document is used to generate or set up this particular stub or proxy. The stub or proxy knows at request time how to invoke the Web service based on the binding information.

The methods of binding to a service are defined by the time when the various elements of binding information are available and used, namely at build time versus at runtime. This results in three client types:

- Static client
- Dynamic client with known service type
- Dynamic client with unknown service type

The high level steps required to assemble the Web service client application (consume services) are as follows:

1. Get WSDL file from the service provider.
2. Validate the WSDL.
   - If the WSDL is not valid, work with the service provider to fix the WSDL.
   - If the WSDL is valid, continue to the next step.
3. Create an enterprise application project in Rational Application Developer.
4. Import the WSDL into the project.
5. Generate the Web services client application.

   Right-click WSDL file, select Web Services -> Generate Client within Rational Application Developer.

   If the XML schema is too complex for JAX-RPC, then use nodatabinding option and third-party binding framework (for example, JAXB).
6. Develop the client application. This includes developing the client application to invoke the Web service.

Note: For detailed information on creating a Web services client application refer to the following:

- 8.6, “Create the internal Web Service client application” on page 254
- *WebSphere Version 6 Web Services Handbook, Development and Deployment*, SG24-6461 redbook

When the Web services client application is complete, it can be exported (for example, EAR or WAR) and deployed to WebSphere Application Server. The client application can now be run to consume services.

4

Best practices for SOA

Best practices are used to deliver a particular outcome by leveraging the knowledge learned from previous engagements. Best practices include methodologies, techniques, guidelines and patterns. By leveraging the knowledge captured in the best practices, the project can be run with less problems and can be deployed more rapidly.

The IBM SOA Foundation scenarios described in 2.4, “SOA Foundation scenarios” on page 34 span a very wide range of solution offerings using IBM products. The best practices described in this chapter are broad in their scope and can be applied in most cases to each of the SOA scenarios.

The chapter is organized into the following sections:

- Rational Unified Process (RUP) and SOA
- SOA Adoption
- SOA Governance
- Service identification and modeling
- Patterns

4.1 Rational Unified Process (RUP) and SOA

The two most common software development methodologies from an IBM perspective are the Rational Unified Process (RUP) and IBM Method (used by IBM Global Services). When reviewing these methodologies, you will find that they share many of the same concepts. We focus on Rational Unified Process (RUP), because it is a commercial software development process framework available to customers, business partners, and IBMers.

RUP includes a library of best practices for software development. In addition, RUP provides a framework and context in which other best practices can be used. For example, SOA adoption guidelines can be used in the Requirements discipline of RUP, and the Patterns for e-business can be used in the Analysis and Design disciplines of RUP to accelerate the creation of artifacts and work products.

This section is organized into the following topics to provide an overview of RUP, the Rational Method Composer tooling and plugins for SOA:

- RUP key elements
- RUP software development best practices
- RUP architecture
- Rational Method Composer
- RUP for SOA plugin
- RUP for SOA Governance plugin

4.1.1 RUP key elements

RUP is comprised of best practices, process delivery tools, configuration tools, process authoring tools, and a community for architects and software developers to share their process extensions.

Best practices

RUP includes a library of best practices for software engineering, spanning a wide range of disciplines including project management, business modeling, capturing requirements, analysis and design, implementation (development), test, deployment, configuration and change management, and the environment.

Process delivery tools

RUP is delivered using Web technology, allowing it be integrated with other software development tools and making it easily accessible to developers.

Configuration tools

RUP is made up of components and plug-ins that can be selected and configured to meet the needs of different projects.

Process authoring tools

An organization can extend or modify the RUP by creating its own plug-ins using the Rational Process Workbench® product.

Community and marketplace

The Rational Developer Network® (RDN™) provides a place for process engineers in the software development community to share their process extensions.

4.1.2 RUP software development best practices

RUP is a software development process framework that provides a disciplined approach to software development. Its goal is to ensure the production of high-quality software that meets the needs of its end users within a predictable schedule and budget. Central to meeting this goal is promoting commercially proven best practices.

Develop iteratively

System functionality should be delivered in a successive series of releases, addressing critical risks and getting feedback in the early releases.

Manage requirements

Requirements management is a systematic approach to eliciting, organizing, communicating, and managing the changing requirements of a software-intensive system or application. Effective requirements management enables better control over customer satisfaction, budget, and schedule.

Use component-based architecture

RUP provides a methodical, systematic way to design, develop, and validate an architecture. Architectures based on components are more flexible, reusable, and understandable than architectures based on other approaches.

Visually model software

Models are simplifications of reality. They help us to understand and shape both a problem and its solution and to comprehend large, complex systems that we could not otherwise understand as a whole. RUP uses the Unified Modeling Language (UML), a standard graphical language for visualizing, specifying, constructing, and documenting software-intensive systems.

Continuously verify quality

Quality is the responsibility of every member of the development organization. It should not be looked at as something a specialized staff looks after at the end of a project. Quality is managed throughout the life cycle by implementing and assessing both process and product quality.

4.1.3 RUP architecture

Figure 4-1 shows the overall architecture of the Rational Unified Process. The process has two dimensions:

- The horizontal dimension represents time and shows the *phase* and *iteration* milestones that occur over the life of the project.
- The vertical dimension represents content in logical groups called *disciplines*.

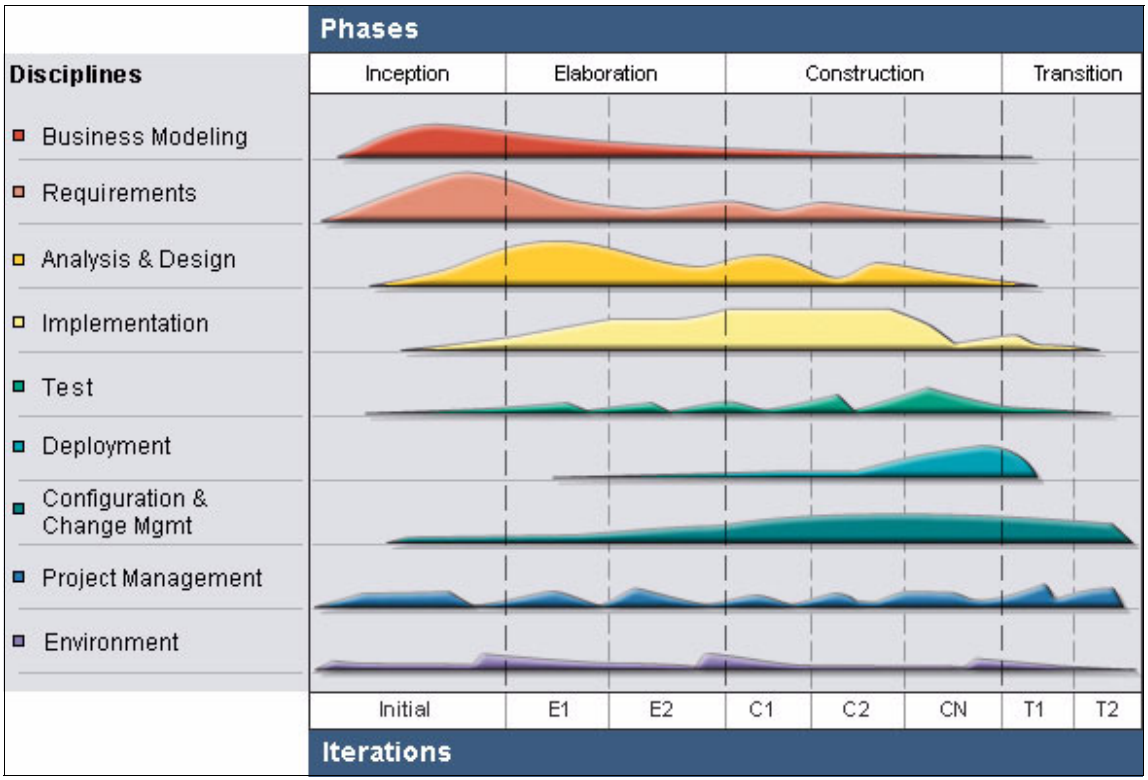


Figure 4-1 Rational Unified Process (RUP) disciplines and phases

The bulges in the Figure 4-1 show the relative emphasis of the disciplines over time. Iterative development is shown in that we see all disciplines represented in every iteration. What changes is the emphasis of the iterations. For example,

more requirements in the early stages of the project and more testing in the later stages.

RUP follows a standard meta-model for describing software processes of who is doing what, how and when:

- Role: *Who* performs the work?
- Artifacts: *What* is produced?
- Activities: *How* is the work performed?
- Workflows: *When* is the the activity performed?

For more information about meta-models, See the OMG standard Software Process Engineering Meta-Model, available from:

http://www.omg.org

Roles

A *role* defines the behavior and responsibilities of an individual, or a set of individuals working together as a team, within the context of a software engineering organization. Note that roles are not individuals, but instead describe responsibilities. An individual will typically take on several roles at one time and frequently will change roles over the duration of the project.

Artifacts

Artifacts can take various shapes or forms, such as:

- A model, such as the use-case model or the design model. These contain model elements (sub-artifacts) such as design classes, use cases, and design subsystems.
- Databases or other types of tabular information repositories such as spreadsheets.
- Source code and executables.
- Various types of documents. For example, a specification document such as the requirements specification, or a plan document such as the software development plan.

Activities

An *activity* represents the tasks performed by a role. It is usually defined as a series of steps that involve creating or updating one or more artifacts.

Some examples of activities are:

- Find actors and use cases: An activity performed by the system analyst role to identify high-level functional requirements in terms of actors and use cases.

- Describe distribution: An activity performed by the software architect role to describe software distribution across multiple processors.

Workflows

A *workflow* describes a sequence of activities that shows how the work is ordered. The workflows and interaction plans include the concept of iterative phases. Phases provide project milestones that ensure that iterations make progress and converge on a solution. The RUP phases are Inception, Elaboration, Construction, and Transition.

The workflows within RUP have been categorized as follows:

- Core workflows: The workflows represent the activities performed by role, and are known as *disciplines* (see Figure 4-1 on page 86).
- Workflow details: The workflow details represent closely related activities, including information flows and work product inputs and outputs for the activities.
- Iteration plans: The iteration plans represent the selection of an activity with one or more iteration through the phases of the corresponding discipline.

4.1.4 Rational Method Composer

The Rational Unified Process is instantiated by tooling and plug-ins used to extend RUP for specific project types. Within the SOA solution domain, our focus is on the following RUP tooling and plug-ins:

- IBM Rational Method Composer
- RUP for SOA plugin
- RUP for SOA Governance plugin

The IBM Rational Method Composer is an Eclipse-based method authoring and publishing tool, with multiple process content libraries, including RUP and RUP plug-ins, SUMMIT Ascendant® methods, and other processes for program, and portfolio management.

IBM Rational Method Composer V7.0.1 includes plug-ins for the base RUP features, as well as some important extensions for SOA such as the RUP for SOA plug-in. Since the Rational Method Composer is an Eclipse based tool, the plug-ins are accessible from the IBM Rational Software Development Platform products based on Eclipse, such as IBM Rational Software Architect.

More detailed information on the IBM Rational Method Composer can be found at this Web site:

`http://www.ibm.com/software/awdtools/rmc/index.html`

4.1.5 RUP for SOA plugin

The RUP for SOA Plug-in provides extensions to RUP specifically for SOA. The RUP for SOA Plug-in V2.0.1 is included in IBM Rational Method Composer V7.0.1.

RUP for SOA provides guidance for the Software Architect and Designer in developing a Service Model, which is a model representing a portfolio of services that can be used as the basis for implementation tasks already in the RUP. Many SOA projects use business-process models in understanding their business, functional requirements, and the services required to support a process. RUP for SOA describes the connection between business modeling and the services model.

We have included a following topics to provide a summary description of the content provided with the RUP for SOA plugin:

- Scope of updates in RUP for SOA
- Roadmap through RUP life cycle for SOA
- Roles and tasks
- Work products

Note: Providing a detailed description of all the features provided by the RUP for SOA plugin is beyond the scope of this redbook. For a full description of the RUP for SOA plugin features, refer to the information provided with the plugin.

Scope of updates in RUP for SOA

The scope of the updates made in RUP for SOA are as follows:

- Leverage the existing RUP: Where possible describe new tasks and work products in relation to existing ones in the RUP and not unnecessarily add new concepts. Also, new elements should be added such that they fit into the overall flow of the RUP.
- Integration of other IBM experience in SOA: Leverage the knowledge of other groups within IBM that have experience that can be harvested; and added to the concepts, guidelines, and practice we are introduced.
- Scope changes to Analysis & Design: The focus in this version was given to architecture and design issues.
- Deliver a foundation: This iteration provides a foundation upon which to build. Additional guidance can be added to the framework presented, such that there will be a connection made between this content and the rest of the RUP in subsequent iterations.
- SOMA: A number of key ideas have been incorporated from the IBM Global Services Service-Oriented Modeling and Architecture (SOMA).

Roadmap through RUP life cycle for SOA

This section describes a roadmap through the RUP life cycle when developing service-oriented solutions.

Inception phase activities

The basic workflow for the Inception Phase applies, with the following extensions or variations:

- Project Management
  - Activity - Conceive new project: The focus on the *Task: Develop Business Case* is adjusted to take into account that reusing existing internal and external services change the cost structure of development. Specifically, the cost of developing services decreases, but more effort is spent identifying services and validating that selected services meet their requirements.
  - Activity - Evaluate Project Scope and Risk: Developing a service-oriented solution changes the nature of certain risks and introduces new risks.
    - Externally sourced services increase risk because they introduce critical elements not under the project team's direct control.
    - Externally sourced services can reduce development time, reducing resource risk
    - Externally sourced services may introduce security and quality of service policies that are not in line with internal architectural guidelines
  - Activity - Plan the Project: In the *Task: Plan Phases and Iterations®*, the plan for the Construction phase may show the project splitting into three different but parallel tracks: one which develops the application-specific services, one which develops reusable domain-specific services, and the reusable infrastructure services. The decision to introduce parallel tracks is largely a staffing and resource issue introduced by a desire to manage services as reusable assets independent of the applications in which they are deployed.
- Business modeling
  - Activity - Identify Business Processes and Explore Process Automation: In the *Task: Business Architectural Analysis* and the *Task: Identify Business Goals* the key forces that affect the business and the business strategy are outlined.

    It is important that these tasks be aligned to services being developed to support the business. For this reason the *Task: Identify Services*, the traceability from Business Goals through to Services is explored in more detail. It is also important that, as service-oriented solutions are more

aligned with the business and therefore the business architecture, the key elements of the IT services become reflected through to the business

- Requirements
  - Activity - Manage the Scope of the System: In the *Task: Manage Dependencies*, the solution's requirements should be managed with the autonomy of services in mind. Services with fewer dependencies are more autonomous and therefore more reusable.
- Analysis & Design
  - Activity - Perform Architectural Synthesis: In the *Task: Architectural Analysis*, a number of the identified steps are affected by the development of service-oriented solutions. The survey of available assets can reuse the Service Portfolio models to gather information on existing services and their capabilities.

    Also, the development of the deployment overview has to take into account the middleware platforms to be used for service implementation and deployment.
- Test
  - Activity - Define Evaluation Mission: In the *Task: Identify Targets of Test,* the test strategy should take into account the autonomous nature of services and ensure adequate stand-alone tests for each service and tests for the collaboration of services when they interact with each other.
- Environment
  - Activity - Prepare Environment for Project: When collecting and preparing guidelines for the project, take into account the specific set of technologies and standards to be used in developing services. Guidelines should include how the particular set of standards for service definition, policy identification and security. They should also provide testing guidance on how to verify conformance with both the standards themselves and the interfaces defined between components.

Elaboration phase activities

The basic workflow for the Elaboration Phase applies, with the following extensions or variations:

- Requirements
  - Activity - Refine the System Definition: The *Task: Detail the Software Requirements* additionally focuses on the technical and nonfunctional requirements and constraints imposed on the services that are either built or purchased. Specific non-functional requirements to consider are

performance, security, infrastructure requirements, service dependencies, run-time licensing issues, and similar constraints that will influence service selection or construction.

- Analysis & Design
  - Activity - Define a Candidate Architecture: The *Task: Architectural Analysis* assumes a service-oriented architectural style and uses the technical and non-functional requirements to define the detailed architecture, including initial partitioning of the solution and a default set of and services represented using the *Work Product: Service Model*.
  - Activity - Design Services: In the *Task: Identify Services*, the services are identified, using a number of techniques to develop an initial service model. In the *Task: Service Design* these candidate services are modeled in detail and their operations are specified with message input and output requirements.
  - Activity - Design Components: The *Task: subsystem_design_real-time_design* further refines the design of the service, identifying subsystems within the service which provide the real behavior of the service. In the early stages of the elaboration phase, there is likely to be a single component which acts as a stub to simulate the behavior of the service for architectural prototyping purposes. Later, the behavior of this component is distributed to a collaboration of classes contained within the subsystem.
  - Activity - Design the Database: The focus in elaboration is on ensuring that the service encapsulates its data requirements completely and that dependencies between services do not arise due to the introduction of services that should be using the same data source.
- Test
  - Activity - Define Evaluation Mission, Verify Test Approach, Test and Evaluate, Achieve Acceptable Mission, Improve Test Assets

    The testing tasks in elaboration focus on validating the individual services. For a service-oriented solution, this focus translates to:

    - Exercising the interfaces between services, to ensure that service boundaries are appropriate.
    - A focus on the complete service specification, to ensure that service implementations fulfill their obligations.
    - A greater focus on performance testing, especially performance scaling tests, to ensure that anticipated transaction volumes can be sustained.

Construction phase activities

The basic workflow for the Construction Phase applies, with the following extensions or variations:

- Analysis & Design
  - Activity - Refine the Architecture and Design Components: The focus in construction is on analyzing the remainder of the business-use cases, system-use cases, or business-process models and identifying appropriate services and service collaborations that realize these.

    The existing architecture is expanded and completed and the internal behaviors of the component are completely designed, implemented and tested against their specifications. Specifically the *Task: identify_design_elements_real-time_design, Task: Identify Design Mechanisms* and *Task: Incorporate Existing Design Elements* are extended to look specifically for services and related elements and mechanisms appropriate to the overall SOA.
- Implementation
  - Activity - Implement Components: It is important to realize that the move toward development of service-oriented solutions does not invalidate lessons learned from developing component-based solutions. In fact, components are used as the implementation artifacts for services. This can be done in two ways, as described below. In either case the approach taken to the development of components will depend on the runtime platform chosen.
    - From the service model, generate Web service artifacts, such as WSDL, which act as a description of a service interface and generate the information about the location and protocols used to access an instance of a service. From these artifacts, most design tools can generate server stubs and client accessors for the specified service.
    - From the service model, transform into either a system Use Case Model, or a Design Model that can be further refined to describe the implementation before generating an Implementation Model or programming language artifacts that represent the service implementation.

Transition phase activities

The basic workflow for the Transition Phase applies, with the following extensions or variations:

- Deployment
  - Activity -Plan Deployment: When developing service-oriented solutions it is important to remember that, for the most part, services are single-instance components of the solution. All applications reusing a

service are using the same instance of a service. By comparison, in a component-based application, components are generally copied into an application and new instances created for each application. This leads to the following concerns:

- Generally, services do not have to be packaged for distribution; they are deployed in a single location, except where distribution or replication concerns require more than one instance.
- Changes to interfaces have immediate effect on all consumers. We cannot plan to roll out new versions of a service in a phased manner as we can with components.
- Requirements for infrastructure services such as security or management must be in place before business or application services.
- Services generally require additional meditate for deployment as dictated by the chosen runtime platform

Roles and tasks

There are three new roles added to RUP for SOA, including SOA Database Designer, SOA Designer, and SOA Software Architect. We have included the key tasks performed by role.

SOA Database Designer

This role contributes to the design of persistent data storage structure to be used by the system for SOA specific considerations.

SOA Designer

This role contributes to the design of a part of the system, within the constraints of the requirements, architecture, and development process for the project SOA specific considerations.

The SOA Designer performs the following tasks:

- Service design
- Class design
- Design testability elements
- Subsystem design
- Use case design

SOA Designer is responsible for the following work products:

- Service model
- Service components

SOA Software Architect

This role contributes to the development of the system's software architecture, including promoting and creating support for the key technical decisions that constrain the overall design and implementation for the project SOA specific considerations.

The SOA Software Architect performs the following tasks:

- Identify design elements
- Identify design mechanisms
- Incorporate design elements
- Identify services

> **Note:** Many techniques for service identification and modeling have been derived from IBM Service-Oriented Modeling and Architecture (SOMA).
>
> For information on SOMA, refer to “Service identification and modeling” on page 121.

- Describe distribution

SOA Software Architect contributes to the following work products:

- Service model
- Service components

Work products

When using RUP for SOA, there are two key work products we want to highlight.

Service model

The *service model* is an abstraction of the IT services implemented within an enterprise and supporting the development of one or more service-oriented solutions. It is used to conceive and document the design of the software services. It is a comprehensive, composite work product encompassing all services, providers, specifications, partitions, messages, collaborations, and the relationships between them.

Service design model

The service *design* model extends the standard RUP design model by adding the service components. The service components artifact is intended for use in describing the realization of a service specification. A service component may provide the realization for one or more services by the realization of multiple service specifications. The set of model elements on the inside of the component represent the concrete realization of the structural, behavioral, and policy contract described by these service specifications.

Tooling used to perform tasks and create work product

The RUP for SOA Plug-in includes a wide array of tooling to perform tasks and create work products. Figure 4-2 displays the tasks and work products available in the plugin tooling. The plug-in also includes guidelines, white papers, standard and custom categories, and processes to accelerate the creation of the creation of architecture and design assets.

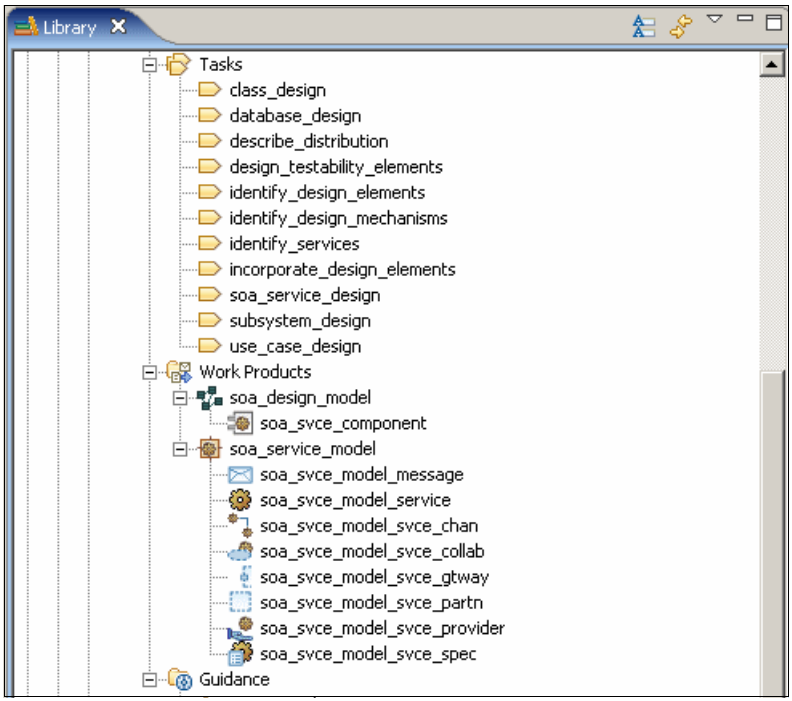


Figure 4-2 RUP for SOA plugin tasks and work products

4.1.6 RUP for SOA Governance plugin

The RUP for SOA Governance plug-in helps identify appropriate best practices, merged with your existing IT processes, to provide proper governance of the capabilities introduced with SOA. The end result is a project plan to create your organization's unique governance framework. The RUP for SOA Governance plug-in V1.0 prerequires IBM Rational Method Composer V7.0.1, however is not dependent on RUP content.

Details on obtaining and installing the RUP for SOA Governance plug-in in the Rational Method Composer can be found at this Web site:

`http://www.ibm.com/developerworks/rational/downloads/06/plugins/rmc_soa_gov/soa_plugin.html#download`

Note: SOA Governance best practices can be found in 4.3, “SOA Governance” on page 107.

Providing a description of all the features provided by the RUP for SOA Governance plugin is beyond the scope of this redbook. For a full description of the RUP for SOA Governance plugin features, refer to the information provided with the plugin.

4.2 SOA Adoption

SOA Adoption provides an iterative and incremental process, and guidelines that assist in developing a roadmap towards a successful migration to SOA. As seen in Figure 4-3, the SOA adoption process begins by defining the scope of possible projects that fit the criteria for being a good fit for a service-oriented architecture.

We explore the following topics for SOA adoption:

- Establish an SOA vision
- Determine the project scope
- Assess and address capability gaps
- Select a pilot project
- IBM can help you get started

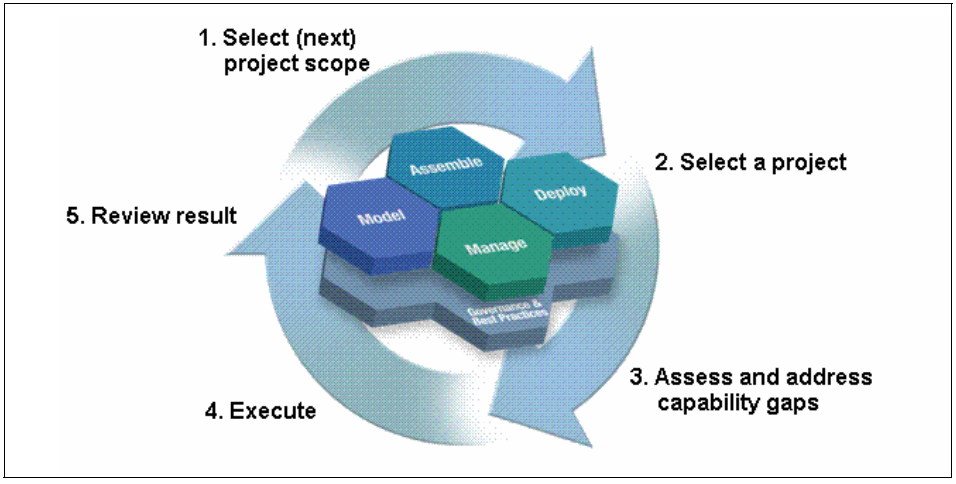


Figure 4-3 SOA adoption process

Tip: The SOA adoption process seen in Figure 4-3 and subsequent sections describe scenarios when SOA adoption is appropriate.

We would like to highlight two articles on IBM developerWorks that provide insight into knowing when SOA is appropriate, as well as learn common anti-patterns that are inhibitors to a successful migration to SOA.

- *Insight and outlook, Part 1: Why and when should you choose SOA?* article found at:

  http://www.ibm.com/developerworks/library/ar-itio1/

- *SOA antipatterns* found at:

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

4.2.1 Establish an SOA vision

Getting started with SOA requires a vision. A vision can be realized by creating and executing a roadmap or plan.

As seen in Figure 4-4 on page 99, there are two primary roadmap perspectives, including *strategic vision* and *project plan*. The strategic vision perspective describes the business and IT statement of direction, which can be used as a guideline for decision making, organizational buy-in, and standards adoption. The project plan perspective (aka. tactical perspective) refers to implementation projects to meet immediate needs of the current business drivers.

Defining the strategic vision starts with assessing the business current maturity across multiple dimensions including business, methodology and technical. The IBM Service Integration Maturity Model (SIMM) can be used to help in this assessment.

After the assessment has been performed, the business needs to establish target goals. This includes documenting important goals and metrics for transition across the maturity dimensions. In addition, it is important to recognize that aspects of the vision can shift with experiences gained. It is a good practice to have regular checkpoints to reassess the vision.

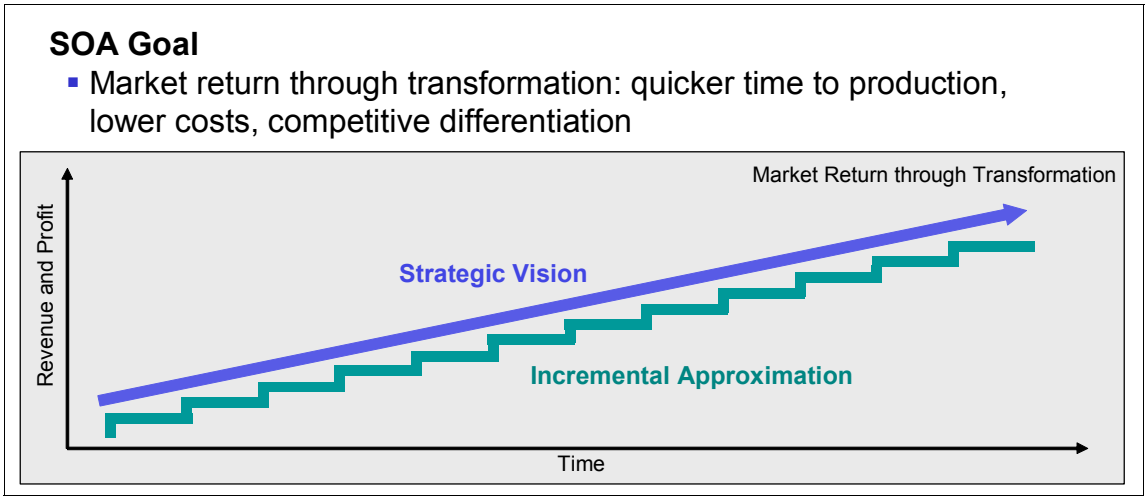


Figure 4-4 SOA adoption: Tactical and Strategic action combined

4.2.2 Determine the project scope

Determining the scope and related business challenges is a key step in gaining an understanding of vision. For example, what are the pain points in current and planned IT systems. Understand what the company wants to achieve with SOA. Do they just want to add a wrapper around existing systems for interoperability or do they want full on-demand capability.

Figure 4-5 on page 100 shows four common requirements based on business objectives. Based on our understanding of the company's objectives we can establish the proper scope.

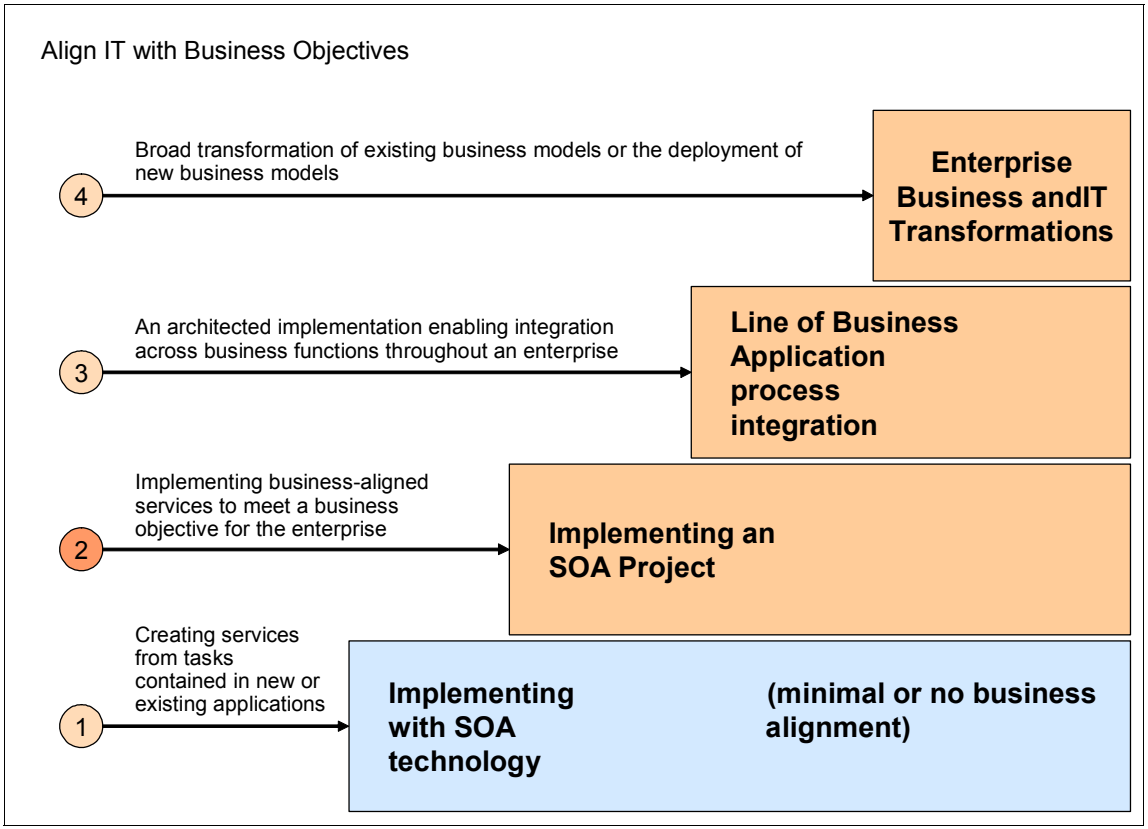


Figure 4-5 Common SOA requirements

Initial entry

In many cases, adoption begins at this level which typically involves a readiness assessment, as described in the next section, and initial validation of the technology. SOA technologies introduced at this level can assist in solving some of the integration problems with the existing infrastructure. Because this entry point is focused mainly on the technology and non-functional requirements, there is minimal or no business alignment.

Business alignment

At this level, we begin to explore transformation of existing assets into business-aligned services to realize some of the benefits of adopting a SOA. The organization might have already identified and prioritized specific business processes that could be potential candidates for a SOA pilot project.

During this stage we engage in activities to establish SOA governance within the organization such as developing best practices and guidelines for development of services. We continue to build new technology and develop skills within the organization.

Integration across business functions

At this level, we shift into enterprise adoption and begin to integrate across various lines of business. A key aspect of this level is the identification and categorization of common services. SOA governance continues to evolve as service domains and ownership are established and key decisions are made involving the service life cycle; defining, monitoring and changing services. The architecture also continues to evolve perhaps with the introduction of an ESB.

Enterprise business and IT transformation

At this level there is a broad transformation of existing business models, through general decomposition of businesses, into processes and business services. This transformation may extend beyond the enterprise across the value net to customers, business partners and suppliers increasing business value.

4.2.3 Assess and address capability gaps

In this section, we describe the IBM Service Integration Maturity Model (SIMM), which includes techniques used to assess the *as-is* business maturity, and define targets for the *to-be* maturity level. In addition, we will highlight some techniques used to address the capability gaps to achieve the target to-be maturity level.

SIMM evaluates the business integration across multiple dimensions including business view, organization, methods, applications, architecture, and infrastructure. SIMM defines seven levels of maturity:

- Level 1: The organization starts from proprietary and ad-hoc integration, rendering the architecture rigid in the face of change.
- Level 2: The organization moves toward some form of EAI (Enterprise Application Integration), albeit with proprietary connections and integration points. The approaches it uses are tailored to use legacy systems and attempt to dissect and re-factor through data integration.
- Level 3: At this level, the organization componentizes and modularizes major or critical parts of its application portfolio. It uses legacy transformation and renovation methods to re-factor legacy J2EE or .NET-based systems with clear component boundaries and scope, exposing functionality in a more modular fashion. The integration between components is through their interfaces and the contracts between them.

- Level 4: The organization begins the early phases of SOA by defining and exposing services for consumption internally or externally for business partners.
- Level 5: Now the organization extends its influence into the value chain and into the service ecosystem. Services form a contract among suppliers, consumers, and brokers who can build their own ecosystem for on-demand interaction.
- Level 6: The organization now creates a virtualized infrastructure to run applications. It achieves this level after decoupling the application, its services, components, and flows. The infrastructure is more finely tuned, and the notions of the grid and grid service render it more agile. It externalizes its monitoring, management, and events (common event infrastructure).
- Level 7: The organization has a dynamically reconfigurable software architecture. It can compose services at runtime using externalized policy descriptions, management, and monitoring.

Figure 4-6 on page 103 provides an example assessment using SIMM. The stars represent the current maturity level (as-is), and the bulls eyes show the target maturity level (to-be).

The bubbles in Figure 4-6 highlight the techniques used to address the capability gaps to achieve the target to-be maturity level. For example:

- Perform business service decomposition.
- Reengineer development process.
- Adopt process choreography assembly model.
- Focus architecture on service orientation.
- Form an SOA Center of Excellence to build skill competency.
- Introduce open standards.

When you have a good understanding of where the company is today and the company vision for the future, you can begin to develop a roadmap towards full adoption of SOA. It is crucial to create a roadmap that is gradual, incremental, iterative, and evolutionary. Plan to attain incremental value from incremental investment.

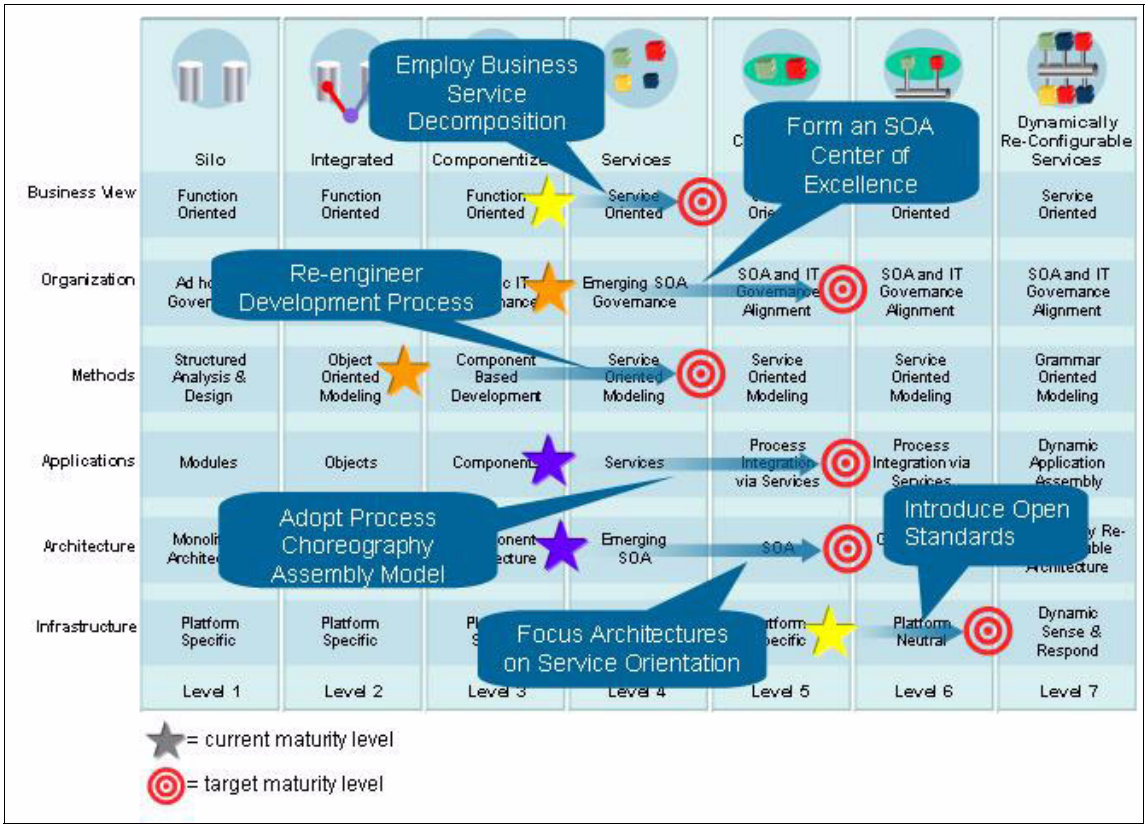


Figure 4-6 IBM Service Integrated Maturity Model (SIMM)

4.2.4 Select a pilot project

The selection of a pilot project is an important step in a successful adoption of SOA. The pilot project should not be considered a proof-of-concept, but rather and incremental move towards implementing the SOA vision.

Be careful not to take on more than you can handle and avoid the *big bang* approach, attempting to solve each and every problem of the enterprise through a single project. Begin with a small, manageable pilot project that demonstrates early success. There is significant value in demonstrating component reuse while minimizing the inherent risks of adopting a new architecture. It is paramount to be able to demonstrate incremental successes with each phase of adoption.

This section includes some criteria that can be used in the selection of SOA pilot projects. In addition, we expand on leveraging the SOA entry points to accelerate SOA adoption.

We have listed key criteria to be used in the selection of a pilot SOA project:

- Address a well understood business problem.

  The pilot project should illustrate the potential for (relatively) immediate the benefits with SOA. In depth business analysis such as a Component Business Model (CBM) effort will be used to help uncover the less obvious problems later in the adoption process.

- Incorporate aspects of governance.

  Governance is an issue at each stage of adoption. However the different adoption levels have different Governance concerns. In the initial stages, Governance will be more concerned with identifying and implementing technical best practices (XML Schema usage, WSDL best practices, etc.). As the number of services grows the issues to address become more coarse grained such as service catalog management.

- Include line-of-business (LOB) objectives and IT objectives.
- Require an achievable stretch beyond current capabilities to address gaps (skills, processes, and so forth)
- Create something that will be put into production and have value.

  Specifically, the pilot project should not be a proof of concept (PoC). Issues of deployment and management are very much a part of SOA and tend to be glossed over in PoC efforts.

- Leverage the SOA entry points.

  Refer to 1.2.2, “IBM SOA entry points” on page 15 for more information.

4.2.5 IBM can help you get started

In this section, we highlight the following IBM resource to help you get started with SOA:

- IBM SOA Assessment tool

IBM SOA Assessment tool

The IBM SOA Assessment tool can be used to evaluate your SOA maturity and provide you valuable information and actionable recommendations. The IBM SOA Assessment tool can be accessed online at the following URL for customers to perform a self-assessment of there SOA maturity:

http://www.ibm.com/software/solutions/soa/soaassessment/index.html

Conduct and IBM SOA workshop

Here are some IBM SOA workshop opportunities of which to be aware:

- IBM SOA Industry teams

  The IBM SOA industry team collects data for your infrastructure, integrates it with applications, best practices, and intellectual property and then delivers completed business processes to you. This utilizes the best-of-breed set of capabilities that IBM has accumulated around process knowledge, industry knowledge, and technology knowledge to provide you with a high-value set of services.

- Client Architectural Readiness Evaluation (CARE)

  IBM reviews your business, along with you IT initiatives and architecture alignment. IBM evaluates your enterprise architecture's SOA support readiness, and analyzes your governance maturity to ensure that you have an environment that encourages SOA success. This results in an SOA-based solution adoption roadmap.

- SOA Jumpstart

  SOA Jumpstart is a multi-day on-site session offered world-wide free of charge. IBM brings architects and specialists to your site to perform skills development and integration architecture workshops that results in actionable next steps for your service orientation.

IBM SOA scenarios

For details refer to 2.4, "SOA Foundation scenarios" on page 34.

SOA skills development for SOA architect role

Figure 4-7 on page 106 displays the SOA classes for architects and the relationship between the classes.

SOA Architect (Web based):

- Code SW717: Introduction to the Value and Governance Model of Service-Oriented Architecture found at:

  http://www.ibm.com/developerworks/websphere/education/enablement/wbt/sw717.html

- Code SW718: Design SOA Solutions and Apply Project, Technical, and Operational Governance found at (SW717 prereq):

  http://www.ibm.com/developerworks/websphere/education/enablement/wbt/sw718.html

- Code SW719: Technologies and Standards for Service-Oriented Architecture Project Implementation found at (SW717 prereq):

  http://www.ibm.com/developerworks/websphere/education/enablement/wbt/sw719.html

Enterprise Service Bus:

- Code SW340: Implementing ESB Solutions using IBM WebSphere products (classroom based)

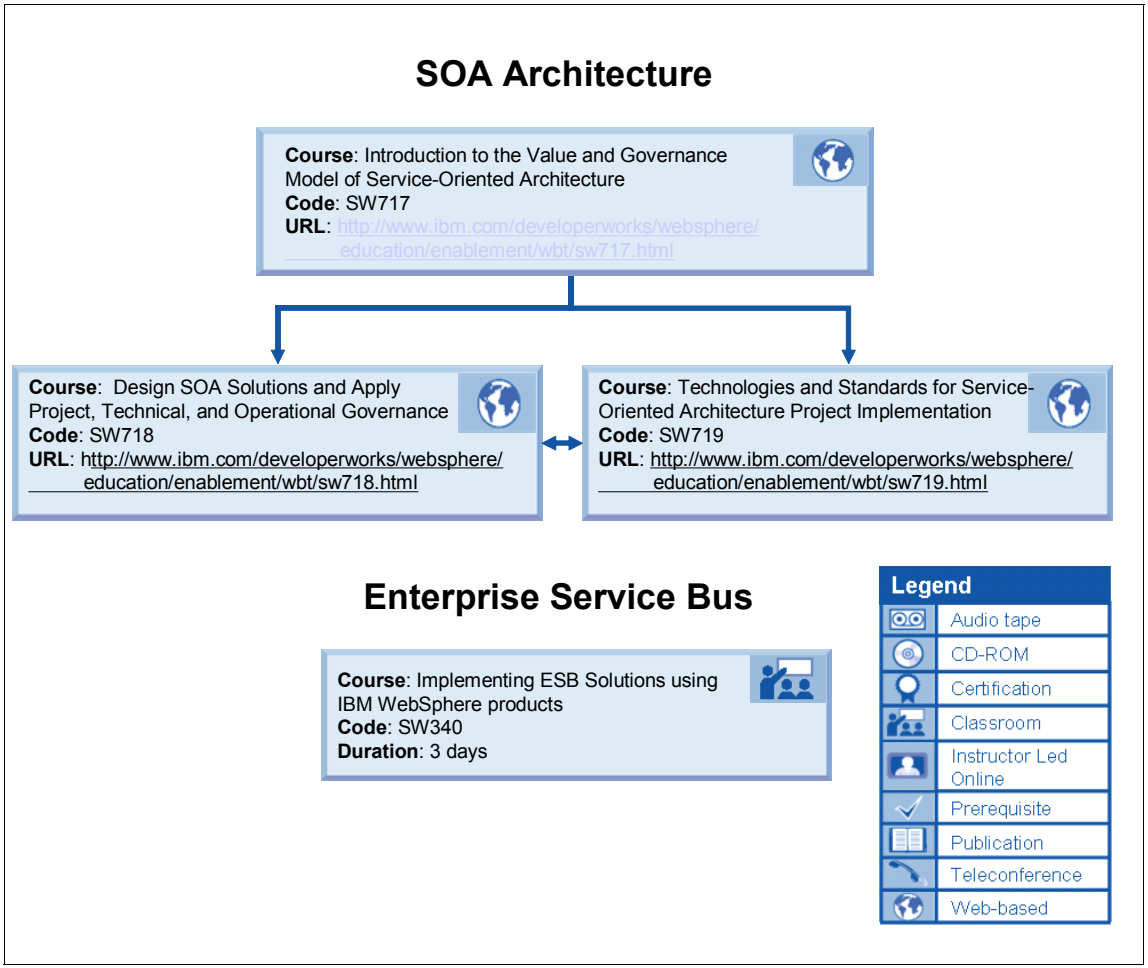


Figure 4-7 SOA skills development for SOA architect role

IBM ITSO workshops for SOA

In addition to the classes listed in the previous section, there are SOA-related workshops provided by IBM ITSO. For more information, refer to the following:

http://www.redbooks.ibm.com/projects.nsf/WorkshopIndex/

IBM ITSO redbooks for SOA

There are many other SOA-related redbooks available at the IBM Redbook site:

http://www.ibm.com/redbooks

Note: Refer to Table 5-5 on page 150 and Table 5-5 on page 150 for a listing of books for architecture and implementation of SOA based solutions.

There are three key IBM SOA Foundation scenario redbooks that you can use for further research:

- *Patterns: SOA Foundation - Service Creation Scenario*, SG24-7240
- *Patterns: SOA Foundation - Service Connectivity Scenario*, SG24-7228
- *Patterns: SOA Foundation - Business Process Management Scenario*, SG24-7234

4.3 SOA Governance

Governance is critical to the success of any SOA project. Governance helps clients extend the planned SOA across the enterprise in a controlled manner to ensure there is proper alignment of IT with business objectives.

In this section, we explore the following topics of SOA Governance:

- SOA Governance definition, challenges and benefits
- SOA Governance Framework
- SOA Governance life cycle
- SOA Governance and Management Method
- SOA Governance Organization
- Tools to Manage Assets and Govern Access
- Summary of SOA Governance challenges and solutions

4.3.1 SOA Governance definition, challenges and benefits

This section answers the following key questions about SOA Governance from the perspective of an Enterprise Architect:

- What is SOA Governance?
- Why is SOA Governance needed?
- What are the benefits of SOA Governance?

What is SOA Governance?

SOA Governance is an extension of IT Governance. IT Governance is used to establish decision-making rights associated with IT. IT Governance is also used to establish mechanisms and policies for measuring and controlling the way IT decisions are made and implemented.

SOA Governance is a catalyst for improving overall IT Governance. SOA Governance extends IT Governance and is focused on the life cycle of services to ensure the business value of SOA. SOA governance addresses the following aspects of the service life cycle:

- Planning
- Publishing
- Discovery
- Versioning
- Management
- Security

Why is SOA Governance needed?

Governance plays an increasingly important role in today's challenging business environment. There are many motivating factors that highlight the need for governance within organizations that are on the path to adoption of SOA.

Table 4-1 provides a summary of the key SOA Governance challenges and capabilities needed to address these challenges.

Table 4-1 SOA Governance challenges and capabilities needed

Challenge	Description	Capability needed
Establish decision rights		
	► Funding and ownership of shared services ► Executive commitment to governance ► Organizational design	► Service portfolio planning ► Organizational design ► Governance process for SOA
Define high value business services		

Challenge	Description	Capability needed
	► Decision rights and governance process for shared services ► Communicating and enforcing standards, platforms and policies ► Identifying and implementing shareable services	► Governance process for SOA ► Modeling to architect business processes, information services and identifying metrics ► Enterprise view of services & data ► Best practices for identifying and defining shared services
Manage the life cycle of assets		
	► Eliminate and prevent unnecessary service proliferation ► Change management for shared services ► Security & authentication ► Decision rights & process	► Governance process for SOA ► Change management ► Policies for publishing, using and retiring services ► Infrastructure to help organize and discover services assets, govern access and monitor service vitality
Measure effectiveness		
	► Measuring service utilization and cost ► Measuring project cost ► Measuring business benefit ► Access and visibility to information ► Decision rights and governance process	► Governance process for SOA ► Visibility to usage and project information ► Business and IT dashboards

What are the benefits of SOA Governance?

In this section, we elaborate on how SOA Governance is key element for the assessment of the SOA maturity level. In addition, we describe how the value of SOA Governance can be determined from both an internal and external business perspective. Lastly, we explain how SOA Governance is used to ensure IT alignment with business objectives.

In summary, SOA Governance provides the following benefits:

- Realize business benefits of SOA:
  - Business process flexibility
  - Improved time to market
- Mitigate business risk and regain control:
  - Maintaining quality of service (QoS)
  - Ensuring consistency of service

- Improved team effectiveness:
  - Measuring the right things
  - Communicating clearly between business and IT

SOA Governance key to assessment of SOA maturity level

In 4.2.3, “Assess and address capability gaps” on page 101, we introduced the IBM Service Integration Maturity Model (SIMM) used to assess the SOA maturity level. As seen in Figure 4-8, SOA Governance is a key consideration for assessing the SOA maturity.

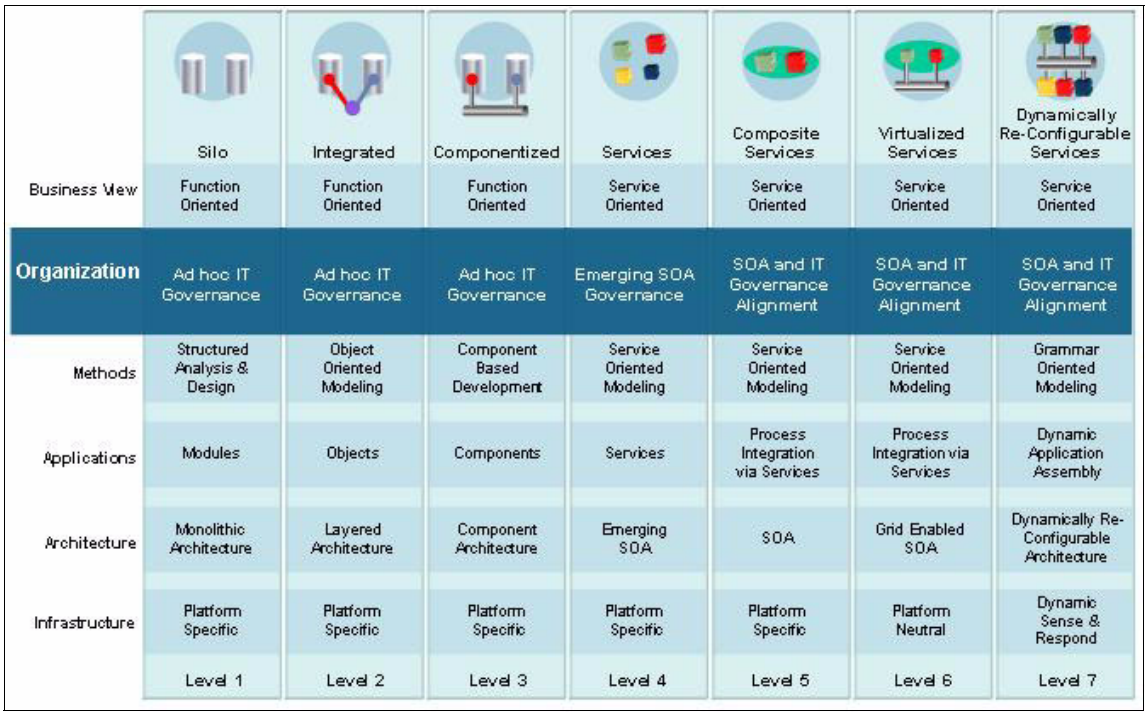


Figure 4-8 SOA Governance key consideration for assessing SOA maturity level

Business value of SOA Governance

The value of SOA Governance can be determined from both an internal and external business perspective. SOA Governance provides real value from both an internal and external business perspective. Governance is increasingly seen as a key indicator for how investors perceive the value of the enterprise, as well as how the company executives determine their effectiveness in cutting costs and increasing profits.

Compliance to regulatory standards such as Sarbanes-Oxley is critical to the perceived value of an enterprise. Market research shows that the current inclination of investors to put their money behind companies that enforce high governance standards. These regulatory acts stress the need to establish and maintain corporate accountability as well as periodically assess its effectiveness.

Good and efficient practices of corporate and IT governance are attracting investors as they attach more credibility and faith to the success and stability of companies that take governance seriously. Investors are more inclined to invest in companies that implement strict standards, and the general (and aptly justified) feeling is that adherence to standards can only be achieved through a governance mechanism. Statistics also reveal that firms with a well exercised IT governance have had 20 percent greater profit margins than their counterparts who make very little or no investment in IT governance. It is quite evident that the investment in strict governance standards has a direct impact to the bottom line of any IT-centric enterprise.

Internally, SOA Governance can facilitate cost-effective use of IT, effective use of IT for asset utilization, revenue growth and business flexibility; which are seen as the key enablers for enterprises that deliver higher profits.

SOA Governance ensures IT alignment with business objectives

Figure 4-9 shows the relationship of SOA Governance with the enterprise architecture, to ensure IT alignment with the business objectives.

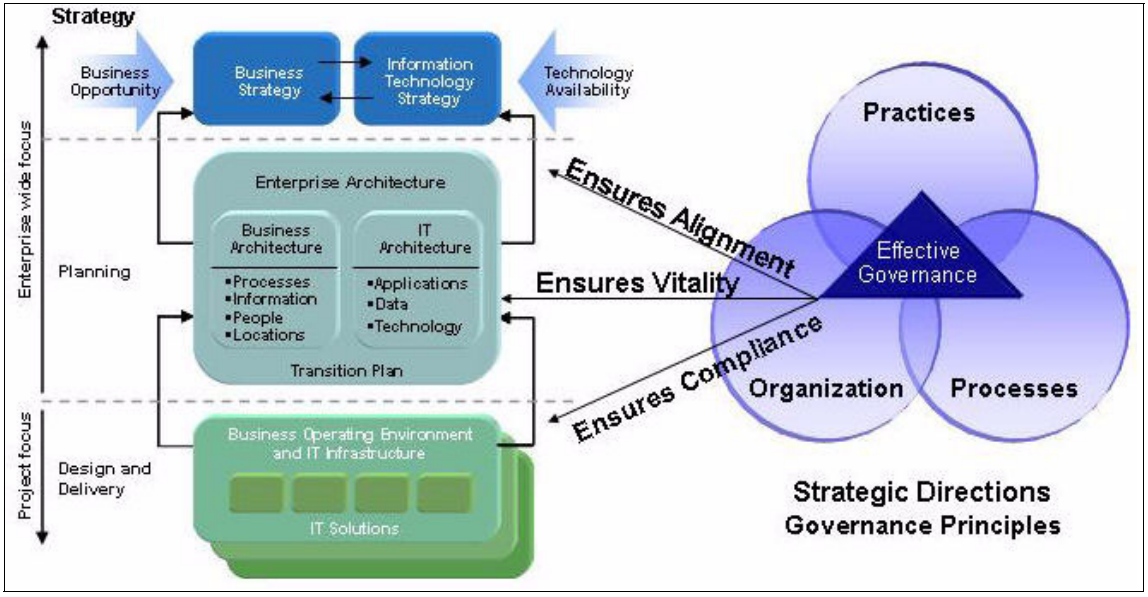


Figure 4-9 SOA Governance ensure IT alignment with business objectives

The SOA based enterprise architecture provides the basis for aligning IT with business needs through appropriate business and IT architectures, principles, and practices. Note that IT itself is a horizontal resource that is needed and used across all line-of-businesses within the enterprise.

A key premise of SOA is to make IT resources more flexible and better able to meet an increasingly broad range of business needs. The enterprise architecture becomes a critical link between corporate and IT governance, and as companies adopt SOA as enterprise architecture. SOA governance becomes the primary way that companies can establish principles for the control of their organizations.

Effective SOA Governance defines the processes, practices and organizations used to ensure the IT alignment with business objectives. This is achieved by ensuring compliance of projects to the defined SOA, and ensuring that the SOA itself is kept vital to meet the ever changing business needs.

4.3.2 SOA Governance Framework

The SOA Governance Framework defines the principles, processes, and roles required to manage, use, and update a service-oriented architecture (see Figure 4-10). The framework represents the interests of the SOA stakeholders of the business to create business driven solutions and IT inspired opportunities.

The SOA Governance Framework has three key objectives.

- The primary objective is to *derive maximum value* from the SOA assets by promoting its implementation, use and evolution.
- The framework defines how the service-oriented architecture and its associated models should be managed and updated in *response to changes* in business needs and available technologies.
- Third, the governance processes are fundamental to *enabling the business* to make conscious decisions about IT, the acquisition of IT assets, and the design and implementation of new IT solutions to meet business needs.

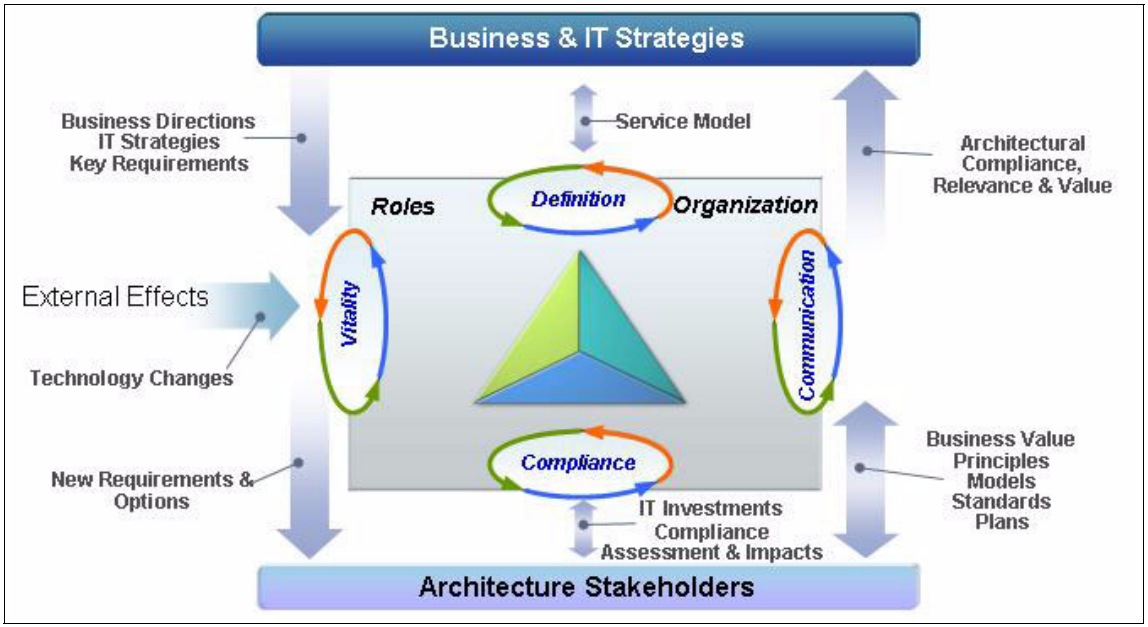


Figure 4-10 SOA Governance Framework

To achieve the objectives of the SOA Governance Framework, four distinct processes have been established. Table 4-2 provides a summary of the processes. These processes enable the enterprise to maintain alignment of business and IT.

Table 4-2 Summary description of SOA Governance Framework processes

Process	Description
SOA Definition Process	Process used to define the design activities that define build and deploy SOA components.
SOA Vitality Process	Process used to keep the SOA current.
SOA Compliance Process	Process used to ensure projects adhere to SOA.
SOA Communications Process	Process used to educate and communicate the architecture across the organization.

SOA Definition Process

This process specifies the architectural design activities that define, build, and deploy components of the SOA. This process includes modeling of business components, business services, and the design of service components that enable the business model activities. It also defines the organizational roles required to support the process.

SOA Vitality Process

This process maintains the applicability and currency of the architecture. This requires the architecture to be current, reflecting the business and IT direction and strategy, as well as anticipated changes in outsourced non-core processes. It must also refine the Architecture Management Process and supporting roles, organization and business functions to ensure its on going usage and relevance. The architectural principles are used to help guide this process.

For example, the Vitality Process ensures that the business direction and IT strategies from the enterprise leadership are examined against the backdrop of existing technologies and new trends. The process then defines new requirements and changes to the Enterprise Architecture stakeholders (generally in IT) for its implementation and update of the SOA vision

SOA Compliance Process

This process reviews, and then either approves or rejects the design to ensure the solution adheres to the principals of SOA. This process can be performed at various points during the business and project cycle. In many cases, it is an add-on to an existing enterprise architecture review and quality process. This process also allows for projects to appeal the non-compliance of a solution design or an IT investment with the architecture and be granted an exception.

For example, the Compliance Process is used to ensure that IT investments required to provide compliance with the enterprise vision are assessed against the impact on other enterprise initiatives.

SOA Communications Process

This process is aimed at educating and communicating the architecture across the organization. This also includes ensuring that the process is acknowledged within the associated processes as well as setting up environments and tools to allow easy access and use of architectural information.

For example, a consistent and frequent communication of the Business Value of the SOA and its principles, models, standards, plans is essential to achieve and maintain the SOA vision and the expected benefits to the business.

4.3.3 SOA Governance life cycle

Figure 4-11 highlights the phases and key elements of the SOA Governance life cycle.

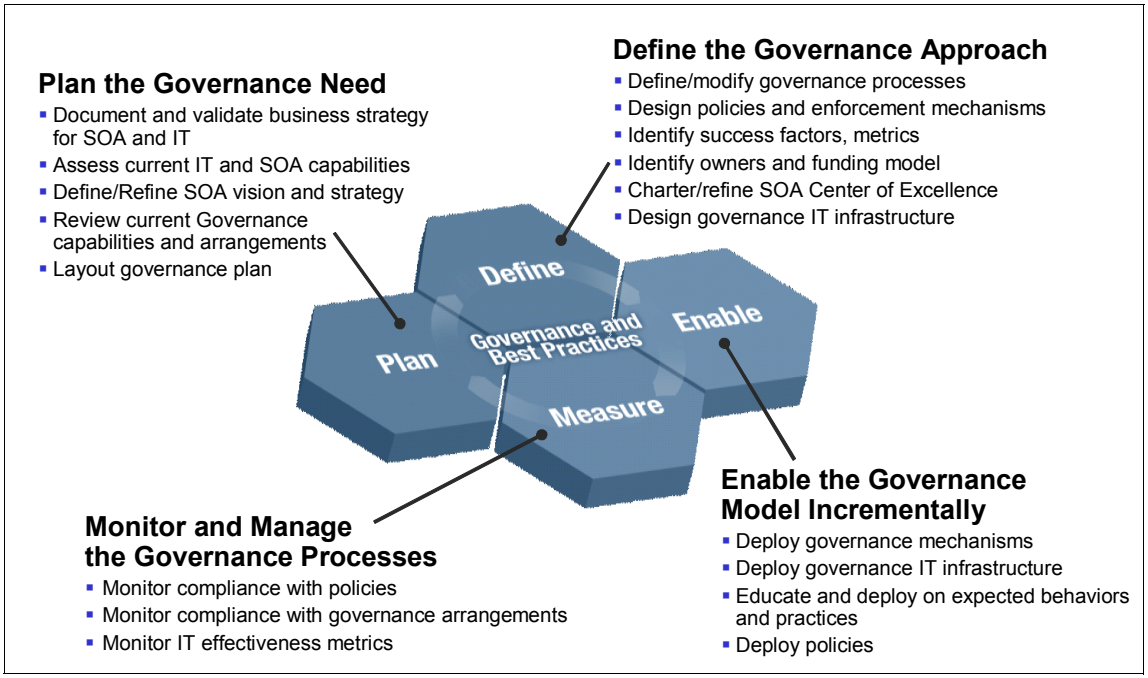


Figure 4-11 SOA Governance life cycle

To better understand the value of using the SOA Governance Life cycle approach depicted in Figure 4-11, we highlight some of the deficiencies of other approaches to governance.

1. Some practitioners deploy SOA technology such as a service registries and SOA management solutions as the solution to governance. It is true that a services registry is needed to manage services at runtime, however this is not sufficient on its own. Also, management is most effective when done in the context of governance.
2. Beware of fragmented and uncoordinated activities around SOA, as well as inconsistent approaches that result in limited ability for reuse.
3. Some view SOA Governance as business as usual and treat SOA projects the same as others.

The SOA Governance Life cycle addresses these issues by providing a comprehensive approach of encompassing the entire services life cycle with multiple entry points. The life cycle includes the use best practices, methodology and processes and tools and technology.

4.3.4 SOA Governance and Management Method

IBM offers a comprehensive consulting and technology offering called the *SOA Governance and Management Method*. The SOA Business Governance Method is included in this consulting offering as you can see represented by the first two phases of Figure 4-12 on page 116.

The SOA Governance and Management Method also includes governance consulting to help customers establish processes for Service Modeling, Service Implementation and Service Management as well as roadmap for continuous improvement.

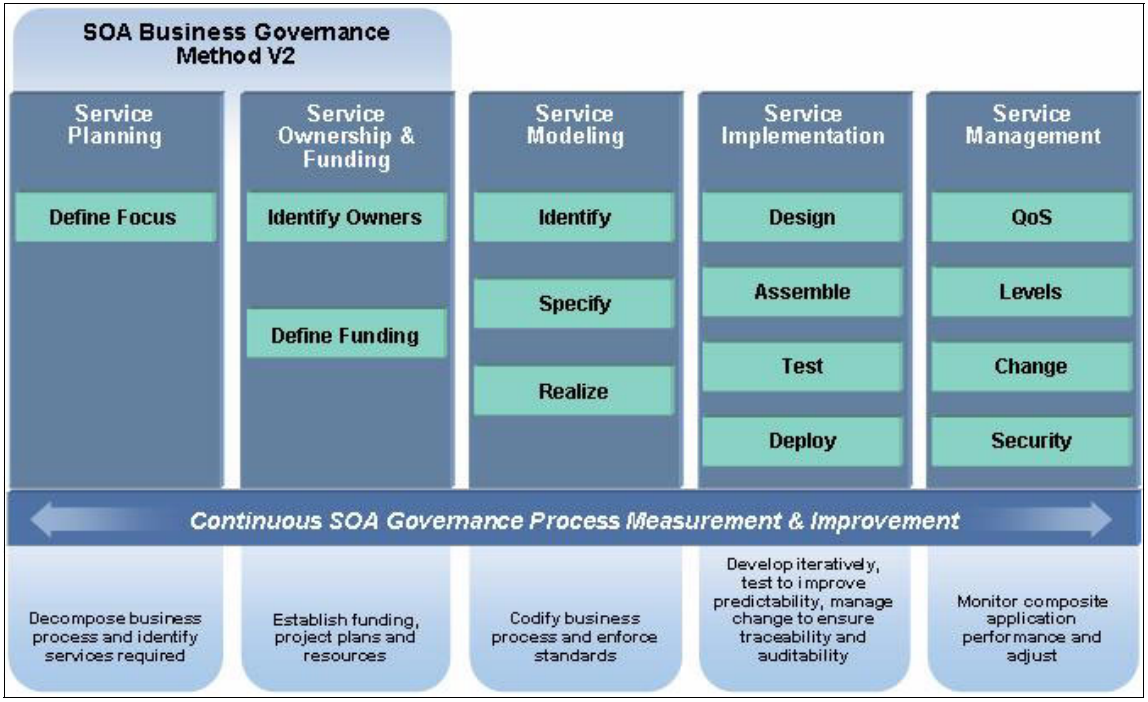


Figure 4-12 SOA Governance and Management Method

We want to highlight the key IBM tooling and techniques used in support of each area of the governance elements depicted in Figure 4-12:

- Service Planning: Service planning requires support for decomposing business process and identifying service interfaces. These processes are supported through Rational RequistePro and WebSphere Business Modeler.
- Service Ownership and Funding: Establishing funding, project plans and resources is supported through the Rational Portfolio Manager.
- Service Modeling: Within Service Modeling we focus on processes for creating and enforcing enterprise service standards through the use of Rational Unified Process (RUP) and Rational Software Architect (RSA). As described in 4.1.4, “Rational Method Composer” on page 88, IBM Rational Method Composer (RMC) V7.0.1 is used to instantiate RUP. RMC version 7.0.1 includes the RUP for SOA plug-in, which can be used to facilitate service modeling.
- Service Implementation: Service Implementation requires that processes support the development and testing of services. This is supported by Rational ClearCase/ClearQuest and Robot.
- Service Management: Service Management requires that the governance process provide support the monitoring of composite application performance. This allows for the proper tuning and debugging of composite applications. These processes are supported by the Tivoli Monitoring portfolio such as IBM Tivoli Composite Application Manager for SOA and WebSphere.

The evolution of the overall process is supported through Rational Portfolio Manager and the Rational Method Composer (see 4.1.4, “Rational Method Composer” on page 88).

4.3.5 SOA Governance Organization

The organizational changes required for effective SOA governance vary from enterprise to enterprise and even between organizations within an enterprise. This is especially true in global organizations with a distributed work force in different cultural settings. There are some roles that are extremely important, regardless of the specific titles and organizational hierarchies.

Classifying services into logical domains simplifies the implementation of an SOA. The business needs to have ownership of the services. The life cycle of these services is tightly associated with the business processes that execute them. A certain level of control is needed to drive the maintenance of the relevance of these services. Several approaches are possible, such as functional, technology-based, application-based, or combination of approaches.

In technology-based domains, it can be desirable to map services to technology-based domains, such as when functional service domains span multiple platforms. Infrastructure services, such as error logging and event handling, are good candidates for technology-based domains.

Table 4-3 provides a summary of roles and responsibilities to consider in SOA Governance decisions.

Table 4-3 Summary of Common SOA Governance Roles and Responsibilities

Organizational responsibility	Roles
Executive Leadership and Funding Sources	► *Executive Sponsor* is the principle stakeholder and the champion of the SOA CoE organization ► *Executive Steering Committee* provides strategy and initial funding and resolves final disputes and funding issues
Business Flexibility Directives	► *Business Process Owners* understand and maintain certain processes with all its business and IT implications ► *Business Unit Committees* are the functional business competencies stakeholders that have to be involved in the SOA Governance process, since SOA is business driven
IT Resources and Architecture	► *Architectural Review Board* oversees the whole IT. ► *Program Management Office* organizes different projects. It is essentially a Center of Excellence (CoE) for project management. SOA Governance effects them due to inspections and reviews.
Advice and Enablement	► The *SOA CoE Board* deals with the management and the operations of the SOA CoE ► The *SOA CoE Advisory Group* is like a community of practice; they are the first line review to ensure enterprise wide compliance with reuse and business agility guiding principles.

4.3.6 Tools to Manage Assets and Govern Access

Figure 4-13 depicts the tools used the manage assets and govern access.

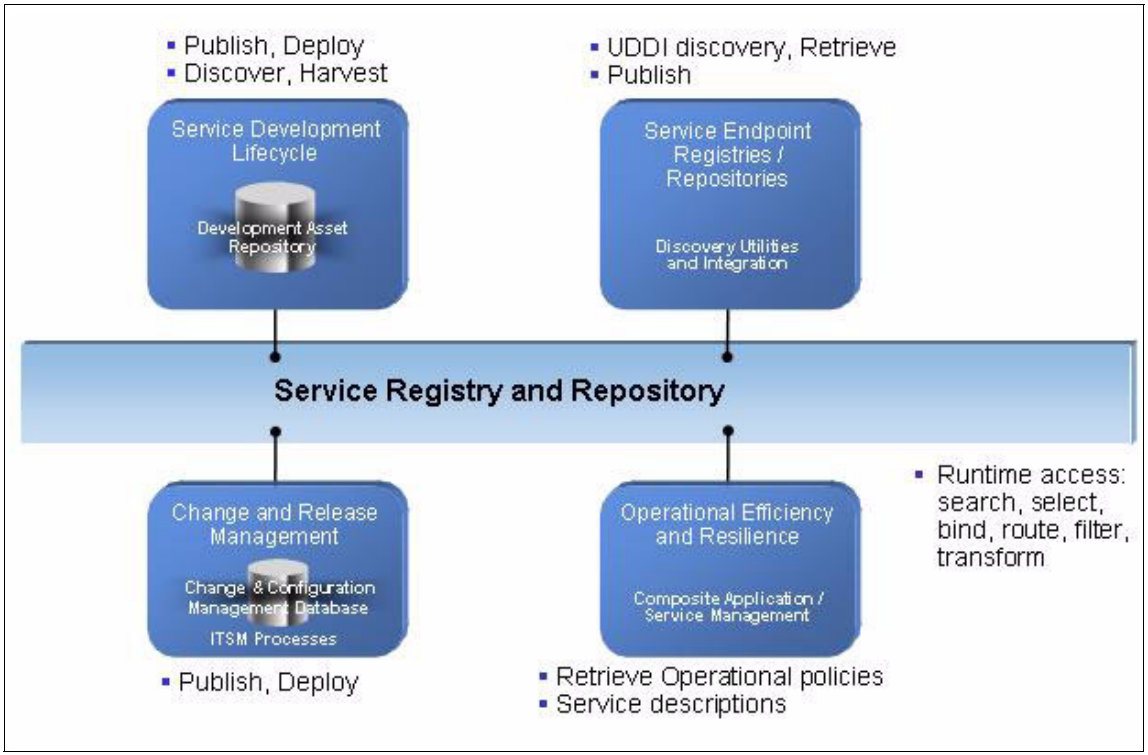


Figure 4-13 Tools used to manage assets and govern access

We have listed a summary description of some of the key management software used to manage assets and govern access:

- WebSphere Service Registry and Repository

WebSphere Service Registry and Repository: IBM currently intends to make available in the second half of 2006 a WebSphere service registry and repository capability that will allow customers, to securely register business services for finding, publishing and notifying changes to SOA infrastructure components such as enterprise service bus and process servers. Customers will also be able to house the metadata about business services in managing the life cycle of a service in SOA. The new capabilities will also include a model for governance to provide guidance and oversight for a SOA project.

While these are our future plans, all statements regarding IBM's plans, directions, and intent are subject to change or withdrawal without notice.

- Tivoli Federated Identity Manager:
  - Policy-based integrated security management for federated web services
  - Decrease security integration costs
  - Prevent fraudulent, unauthorized use of Services
  - Automates user provisioning for SOA transactions
  - Unified customer view across SOA
  - Simplifies security integration for cross-domain federated web services
- Tivoli Access Manager for Business Integration
  - Simplified Security Management across Enterprise
  - Enhance security of Enterprise Service Bus (ESB) processing SOA transactions
  - Application level protection to transactional data
  - Enforce fine-grained authorization on WebSphere MQ Sever resources by WMQ Clients
  - Generates audit data documenting compliance to security policy
- Tivoli Application Dependency Discovery Manager & Change and Configuration Management Database:

  Discovery and Relationship Management for SOA environments
  - Automated Discovery: Fully automated application and device discovery for unified view of configuration items in an SOA environment
  - Audit and Control: Manage the change process and provide record of change as a checkpoint for compliance and audit requirements
  - Integration: Provides an integration point for other IT Service Management processes and management data

4.3.7 Summary of SOA Governance challenges and solutions

Table 4-3 provides a summary of the SOA Governance solutions and techniques used to address the challenges and capability needs of SOA Governance.

Table 4-4 SOA Governance challenges and solutions

Challenges	Solutions
Establish decision rights	► Perform a Service Domain Definition and Ownership study ► Define organizational SOA Governance roles and responsibilities ► Implement an SOA Center of Excellence

Challenges	Solutions
Define high value business services	► Conformity and Validity Checkpoints for Effectiveness ► Define SOA Governance decision paths
Manage the life cycle of assets	► Use IT Service Management to Manage and Govern SOA ► Use IBM software and best practices to Manage Assets and Govern Access
Measure effectiveness	► Execution of Business Strategy Related to IT Goals ► Track and Manage Business Strategy Execution ► Define guidelines for SOA decision making (strategic and tactical)

4.4 Service identification and modeling

Appropriate identification of services is a key aspect of designing an SOA solution. This section provides key concepts, best practices and guidelines for identification of services and solutions to define the foundation of a SOA.

4.4.1 Service-oriented modeling, analysis and design

Experiences from early SOA projects suggest existing development processes and notations, such as Object Oriented Analysis and Design, Enterprise Architecture Frameworks and Business Process Modeling only cover a portion of what is required for a SOA. While traditional methods reinforce general software architecture principles such as information hiding, encapsulation, modularization and separation of concerns, SOA introduces additional themes such as service choreography, service repositories and service bus that also require consideration.

Analysts agree that what is needed to drive a SOA solution is a hybrid approach including a set of modeling, analysis and design techniques that incorporate best-of-breed techniques from object-oriented analysis and design, business process modeling and enterprise architecture as well as new elements. In adopting this hybrid approach, we fill the gaps between more traditional analysis and design techniques and what is required to develop a SOA foundation.

Service-oriented modeling and architecture (SOMA)

A key recommendation for development of a SOA is to adopt and adapt a solution development methodology to identify, design and build SOA components. The *Service-oriented modeling and architecture (SOMA),*

describes a set of product and technology agnostic modeling, analysis and design activities and techniques to build a SOA.

The SOMA process is comprised of three major steps including identification, specification, and realization as depicted in Figure 4-14 on page 122.

.

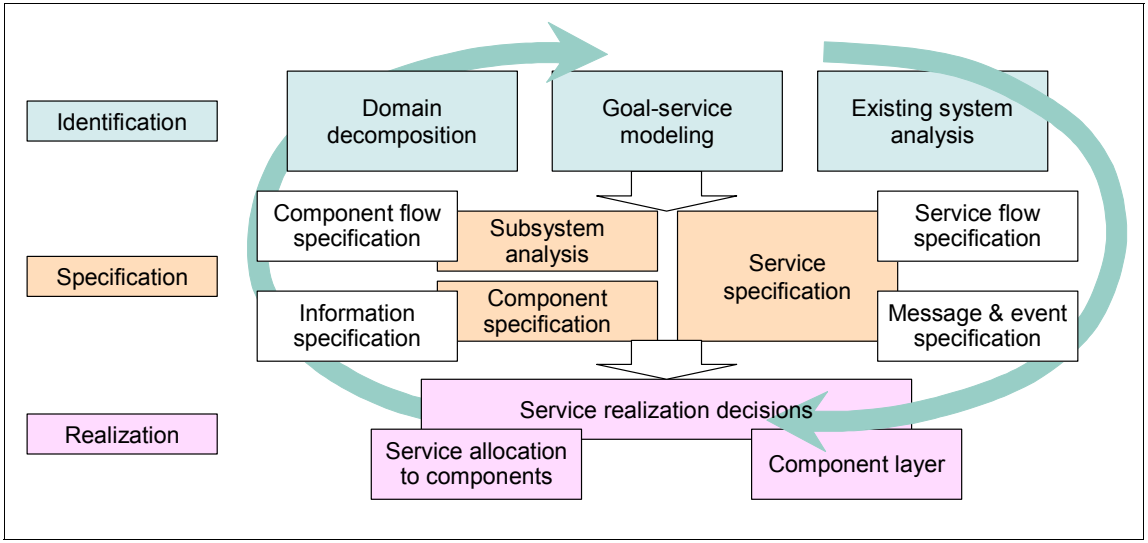


Figure 4-14 Service-oriented modeling and architecture

Through its methods, activities and techniques, SOMA addresses a number of antipatterns that have been encountered by practitioners on engagements involving identification and design of services. Activities and techniques within each of the three steps provides guidelines and solutions to the avoid occurrences of these antipatterns. As we review each of the SOMA steps, we identify and describe the associated antipatterns.

Process and methodology: RUP, SOA, and SOMA

RUP is based on software engineering best practices, offers a configurable framework and is scalable to support enterprise initiatives. Therefore, all aspects of RUP can also be applied to the development of an SOA. RUP provides a systematic approach to bridge the gap between business and IT to support a major area of concern, identification of services and how business processes are realized through execution of services. RUP also provides support for both the bottom-up and top-down approaches by acknowledgement of existing design elements and through activities such as architectural analysis to identify architectural elements such as services. Figure 4-15 on page 123 illustrates the position of SOMA within the RUP life cycle.

Tip: The RUP for SOA plugin V2.0.1 included with IBM Rational Method Composer V7.0.1, includes guidance and tooling that integrates many of the techniques provided by SOMA.

For more information, refer to “RUP for SOA plugin” on page 89.

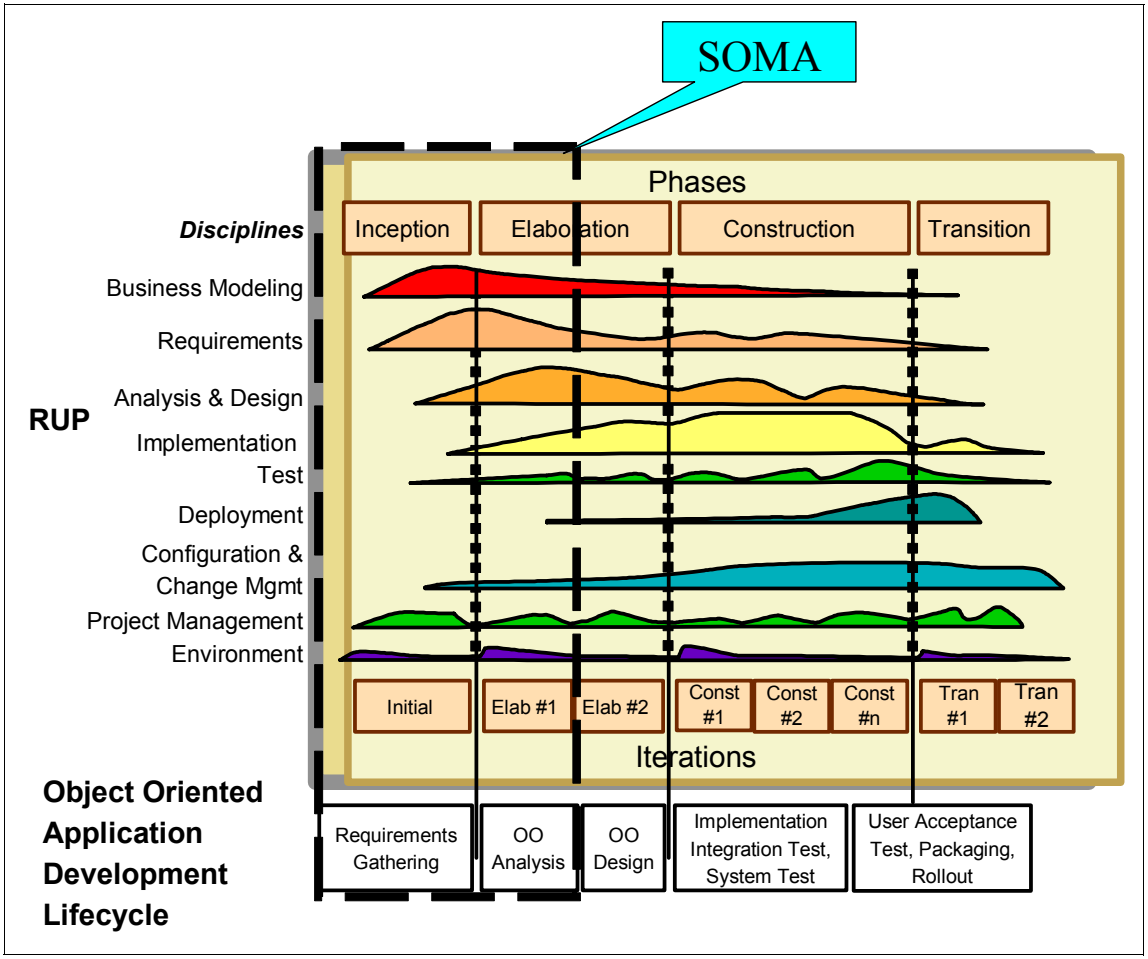


Figure 4-15 Position of SOMA within Life Cycle

Yet, there is some work in progress to provide additional support within RUP such as incorporating content for describing SOMA techniques and artifacts. Some extensions are required such as introduction of the service model as an artifact and some supporting activities.

Service identification

Identification begins by applying three complimentary techniques including domain decomposition, existing asset analysis and goal to service modeling to identify candidate services, candidate enterprise components, and flows.

The most important outcome of these activities is the service model. The service model is comprised of candidate services that, ultimately, support business services, processes and goals of the enterprise.

A key aspect of the identification step is that it employs a *meet-in-the-middle* approach including a combination of top-down, bottom-up and middle-out analysis techniques. In many cases a pure bottom-up approach is taken, however, this approach typically leads to poor definition of services that are driven mainly by architecture of legacy application interfaces and not from a business perspective.

Domain decomposition represents the top-down approach where business domains are decomposed into functional areas across the value net. Through this technique we can establish the scope of the effort. After domains have been decomposed into functional areas, each area can then be further decomposed into processes and sub-processes and high-level business use cases. Experience shows that the business use cases are considered good candidates for exposure.

Existing asset analysis represents the bottom-up approach where we analyze and leverage APIs, transactions and models from legacy and packaged applications as possible candidate services.

Goal-service modeling provides a middle-out approach that relates services to goals, sub goals, KPIs and metrics of the enterprise. This technique provides a certain level of validation in the form of a completeness check in that it may reveal candidate services that were not identified through the top-down and bottom-up activities.

Finally, *subsystem analysis* expands on subystems identified during domain decomposition and specifies interdependencies and flows between them.

Antipatterns

Techniques described within this step provide some best practices that can be applied to avoid some very common antipatterns related to identification of services. We summarized two key antipatterns:

- Service Proliferation Syndrome

  First, there is a strong tendency to equate Web services with SOA where, in reality, Web services is an entry point towards SOA adoption. It is possible to

create an SOA that does not use Web services and it is also possible to use Web services in a way that cannot really be considered service-oriented. Consequences of this viewpoint manifest themselves into a proliferation of services, commonly referred to as the *Service Proliferation Syndrome,* often resulting in inappropriate exposure of services that are not business aligned.

Determining the most appropriate level of service granularity is one of the most challenging aspects of service-oriented modeling. The rule of thumb is to model as coarse grained as possible. While fine-grained services are also possible, ultimately, the challenge is to find the balance between coarse and fine grained services that meet the business needs. Techniques described here help to determine the appropriate granularity of services and minimize proliferation of services that are too fine-grained.

- Silo approach

  The second antipattern involves what is commonly referred to as the *silo approach* where services are identified based on isolated applications rather than a applying more holistic, enterprise focus. In addition to the recommendation mentioned earlier in this chapter to establish a governance framework that ensures cross-tower service identification and communications, the meet-in-the-middle approach emphasized within SOMA promotes exploration and identification of common services and helps to shift thinking towards modeling of services at the enterprise level.

Service specification

In this step, we identify and specify components that will be required to realize services. Some major activities might include:

- Identification of rules, services, attributes, dependencies and variation points for each component.
- Exploration of non-functional requirements required by consumers of the services
- Specification of messaging, event specifications and management definition
- State management decisions

One of the most important activities within this step is to make crucial decisions to determine which services should be exposed. Some high-level questions related to service design analysis and principles can provide guidance in making these determinations, yet there are many more aspects to take into consideration:

- Traceable
  - Can the service be traced back to goals and objectives of the organization?

- Stateless
  - Does the service require information or state between requests?
- Discoverable
  - Can the service be exposed externally to the enterprise?
  - Does the service have a well defined interface and externalized service description?
- **Reusable:**
  - Does the service serve the interest of other processes?
  - Can this service be reused to realize many higher-level business processes?

Service allocation

In this step, we assign services to subsystems identified during previous activities. In addition, we assign services and components that realize them to layers in the SOA as depicted in Figure 4-16. A key activity within this step is the documenting of architectural decisions that relate to both application and technical operational architecture designed and used to support the SOA realization at runtime.

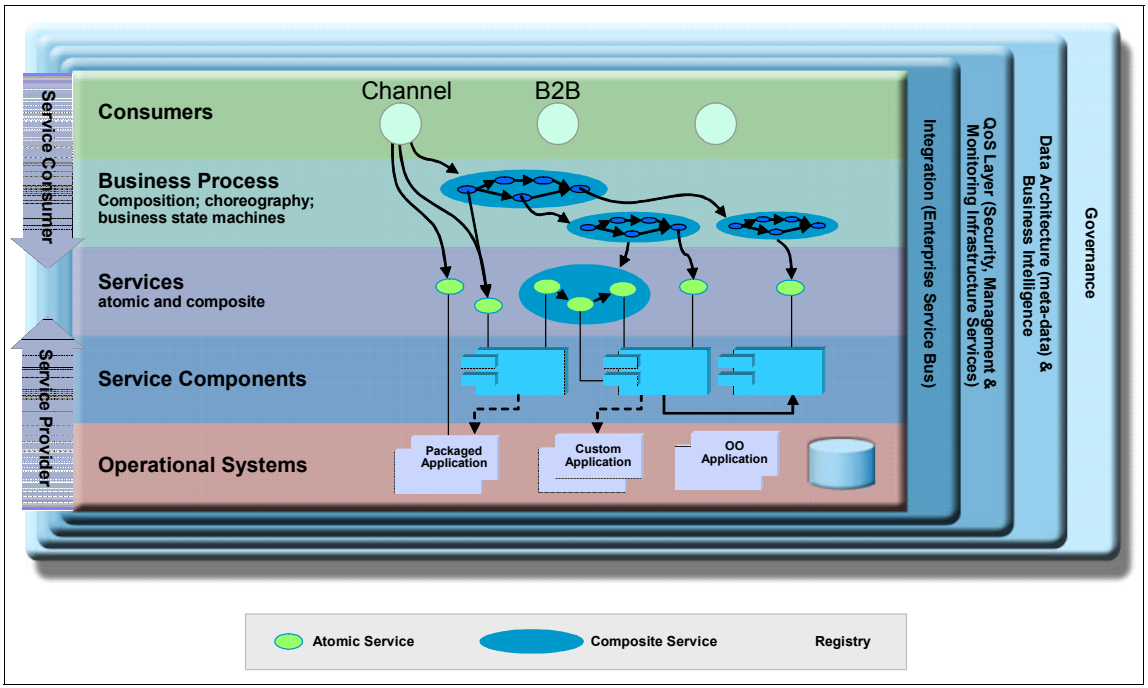


Figure 4-16 Layers of SOA

Through these techniques, we ensure that we find a place for all services and that they can be traced back to business goals and components.

Antipattern: componentless services

There is a tendency to jump directly into development and implementation of Web services without clear association with owning components. This antipattern typically occurs in projects where there is a lack of architectural discipline. Architecture patterns are neither considered nor applied.

Once again, lack of good service modeling and design techniques ultimately result in serious violations of basic principles such as modular design, information hiding, encapsulation and layering. These issues can lead to significant cost and effort related to potential reengineering.

In addition to leveraging J2EE and general EAD best practices, application of SOMA techniques such service allocation will help in avoiding occurrences of this antipattern.

Service realization

Realization is a key architectural step that involves the exploration of alternatives to realize the implementation of services. In addition, we identify and assess technical constraints to ensure technical feasibility of realization of services specifically for those identified during existing asset analysis

Alternatives for implementation of services goes beyond buy versus build. Some realization alternatives are:

- Build new component functionality (custom build).
- Transform legacy to enable reuse of functionality exposed as services.
- Integrate by wrapping existing systems.
- Buy and integrate with third-party products.
- Subscribe and outsource parts of the functionality, especially with Web interfaces. Subscription assumes that an enterprise application integration model has been implemented and there are services to subscribe to, in a hub and spokes architecture.

Transformation considerations

A service's implementation can be realized by wrapping an existing system with a message queue service or a Web service. However, in many cases, mere exposure of existing functionality is not sufficient. Componentization of the existing system or a small subset of the system must take place to properly expose the functionality required. A key factor is the scope of the componentization. Avoid a common tendency to break the entire existing system

down into parts. Select an appropriate subset and transform it through componentization.

Antipattern: chatty services

Similar to the identification and specification steps, there are antipatterns related to realization of a SOA. Though all three steps prescribed within SOMA provide techniques to avoid occurrence of these antipatterns, the consequences of this particular pattern are more related to realization.

In many cases, developers are asked to begin direct replacement of existing APIs with Web services without much thought to the benefits of SOA and conformance to SOA design and architecture principles. This antipattern, referred to as chatty services stemming from a chatty dialog communicating small pieces of information, may result in severe degradation in performance. In addition, significant development costs may be incurred to aggregate the fine-grained services into a service model that can actually realize benefits of SOA. Guidance and techniques to avoid occurrence of this antipattern really spans all three steps of SOMA.

- At the identification step, the meet-in-the-middle approach can help to ensure completeness of the model, attain an appropriate level of service granularity and ensure that there is traceability back from the service to business objectives and goals of the enterprise.
- At the specification step, application of a litmus test considering key service design principles can help to make the appropriate decisions involving service exposure.
- At the realization step, we assess the technical feasibility of services identified through existing asset analysis to further validation the decision of services to be exposed.

4.4.2 Summary

In summary, experience shows that service-oriented modeling, analysis, and design is necessary to build an SOA. It is important to reiterate some of the key aspects and benefits of SOMA:

- Techniques are designed to enable target business processes through identification, specification and realization of business aligned services forming the foundation of a SOA
- Builds SOAs that align clients' business goals and directly ties business processes to underlying applications through services, helping businesses realize benefits more rapidly.
- Approach helps to ensure that goals set by business process modeling can actually be implemented to generate the greatest result in an efficient manner.

Application of these best practices, guidelines and techniques in identification of services and solutions can help to realize the many benefits offered through SOA.

For additional information about the identification of services and solutions, refer to the following Web pages:

- *Service Oriented Modeling and Architecture*, found at:

  http://www.ibm.com/developerworks/webservices/library/ws-soa-design1/
- *Elements of Service-Oriented Analysis and Design*, found at:

  http://www.ibm.com/developerworks/webservices/library/ws-soad1/
- *SOA realization: Service design principles*, found at:

  http://www.ibm.com/developerworks/webservices/library/ws-soa-design/
- IBM SOMA and/or IBM SOA:

  http://www.ibm.com/services/soa

4.5 Patterns

There are many types of architecture and design patterns. Our focus in this redbook is on the Patterns for e-business.

The Patterns for e-business capture a wealth of experience of IBM architects to create solutions quickly, whether for a small local business or a large multinational enterprise. The Patterns for e-business leverage IBM best practices along with a collection of proven architectures that span thousands of successful internet engagements to build robust industrial strength applications.

There are several chapters within the book that provide guidance in use of patterns within the context of the SOA Foundation.

- Chapter 5, “Process for applying SOA scenarios and patterns” on page 131

  This chapter provides a process for applying the SOA scenario and related reusable patterns assets.
- Chapter 6, “Patterns for the Service Creation scenario” on page 155

  This chapter describes the reusable patterns assets from the Patterns for e-business, that can be applied to solution architecture of a Service Creation scenario.

- Chapter 7, “Business scenario and solution architecture” on page 173

  This chapter includes an example that demonstrates how to apply the reusable patterns assets to design the solution architecture for the fictitious ITSO Car Rental company (representative of the Service Creation scenarios.

5

Process for applying SOA scenarios and patterns

This chapter provides a process for applying the SOA scenarios and related reusable patterns assets. The process can be used by specialists in pre-sales and post-sales roles, and by architects in a customer / consulting engagement.

We have defined generic use cases representative of SOA capabilities possible with the software used in the SOA scenarios. The process includes a selection table which maps the generic use cases to the appropriate SOA scenario.

When the SOA scenario is identified, the user can then reference the related reusable patterns assets to help accelerate the solution design, or go directly to the appropriate implementation guide. The implementation guides include detailed examples on how to design the solution architecture and perform the implementation (model, assemble, deploy, manage).

The chapter is organized into the following sections:

- Process for using SOA scenarios and patterns
- Generic use cases for the SOA scenarios
- Select the SOA scenario
- Reuse patterns assets to accelerate the solution architecture
- Select the implementation guide

5.1 Process for using SOA scenarios and patterns

Figure 5-1 depicts the high-level process for applying the SOA scenarios and related patterns. We cover the process from two perspectives. First, we provide a description of the steps in the process. Second, we explain how the process can be used in different ways, depending on the role of the person using the process. The process described in this section is not meant to replace standard methodologies such as Rational Unified Process (RUP) or IBM Method, but rather provide assets that are reusable within the methodologies.

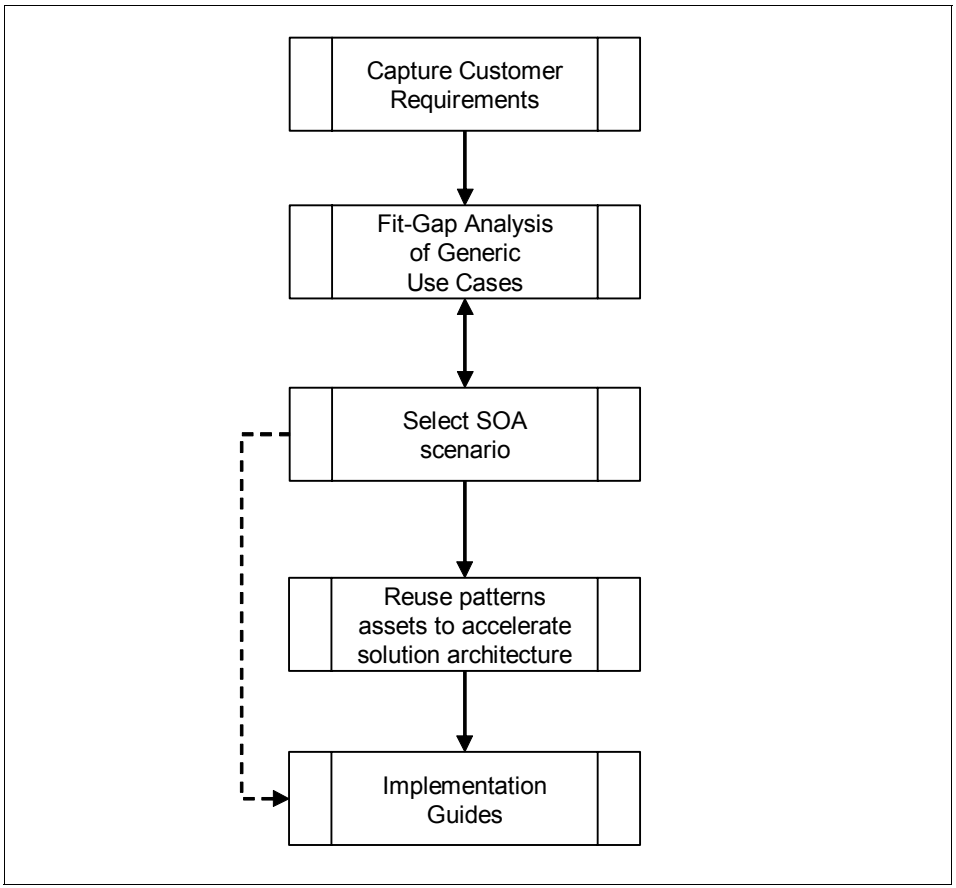


Figure 5-1 Process for using SOA scenarios and patterns

5.1.1 High-level process defined

In this section, we provide a brief description for each phase of the high-level process depicted in Figure 5-1:

1. Capture the Customer Requirements.

   The collection of the customer requirements is a standard activity for development methodologies (for example, RUP or IBM Method). This includes an initial context, a vision of where the customer wants to be, and use cases representing the refined customer requirements.

2. Fit-Gap Analysis of Generic Use Cases.

   The business analyst or architect evaluates which of the generic use cases are representative of the defined customer requirements (use cases).

   The generic use cases are defined in 5.2, “Generic use cases for the SOA scenarios” on page 136.

   Note: The upward pointing arrow in Figure 5-1 on page 132 between Select the SOA scenario and Fit-Gap Analysis is a special case where the user already owns the software representative of an SOA scenario, and wants to know how to leverage the SOA capabilities of the software. In this case, the user knows the SOA scenario and will look at what generic use cases are possible in the SOA scenario selection table (see Table 5-1 on page 142).

3. Select the SOA scenario.

   The business analyst or architect can use the SOA scenario selection table (see Table 5-1 on page 142), which lists the generic uses cases and the SOA scenario which enables the use case.

   Refer to 5.3, “Select the SOA scenario” on page 141 for more detailed information.

   When you have selected the SOA scenario, you have your choice of two options of where to go next:

   – Reuse patterns assets to accelerate the solution architecture.

     This option is appropriate if you are interested in reusable assets for the architecture and design of the solution. This option also leads to the implementation guides.

   – Select the implementation guide (model, assemble, deploy, manage)

     This option is appropriate if you are interested in a working example which demonstrates how to implement the scenario (model, assemble, deploy, manage).

4. Reuse patterns assets to accelerate the solution architecture.

   We have mapped the reusable assets found in the Patterns for e-business for each of the SOA scenarios. The reusable assets are used to accelerate the solution architecture.

   Refer to 5.4, “Reuse patterns assets to accelerate the solution architecture” on page 143 for more detailed information.

5. Select the implementation guide (model, assemble, deploy, manage).

   After you have identified the SOA scenario, and, optionally the corresponding patterns, you can go to the corresponding implementation guide. The implementation guides provide detailed examples demonstrating how to implement selected use cases for the SOA offering. This includes implementation details for each phase of the life cycle using the solution offering software (model, assemble, deploy, manage).

   Refer to 5.5, “Select the implementation guide” on page 151 for more detailed information.

5.1.2 Using the process based on the user role

The process is intended to be used by specialists in the role of pre-sales, specialists in the role of post-sales, business analysts and architects. We have provided following three ways we envision the process being used based on the role of the user:

- Pre-sales role
- Post-sales role
- Customer / consulting engagement role

Pre-sales role

The pre-sales role is typically performed by a technical pre-sale specialist. In this case, the specialist uses the process to identify the appropriate SOA scenario and corresponding software from the generic use case criteria (see Figure 5-1).

For example, a technical pre-sales specialist or business analyst evaluates the generic use cases to determine if the use case is representative of the customer requirements. Once the generic use cases are identified in the SOA scenario selection table (see Table 5-1 on page 142), the appropriate SOA scenario is identified along with the appropriate IBM software to recommend.

You can continue in the process and reuse the patterns assets to accelerate the solution architecture, or go directly to the appropriate implementation guide. These two options are further described in the “Customer / consulting engagement role” on page 135.

Post-sales role

The post-sale support role can be performed by a business analyst, specialist, or architect. In this case, the process is used to identify the SOA capabilities possible with the software found within the SOA scenario (see Figure 5-1 on page 132).

For example, the customer might have already purchased the IBM software and wants to better understand how to fully leverage the SOA capabilities. Each of the SOA scenarios has specific software products defined for each phase of the life cycle. In this case, the user can then work in a bottom up fashion from the SOA scenario representing the software they own, refer to the SOA scenario selection table (Table 5-1 on page 142), and then see what generic use cases are possible.

You can continue in the process and reuse the patterns assets to accelerate the solution architecture, or go directly to the appropriate implementation guide. These two options are further described in the “Customer / consulting engagement role” on page 135.

Customer / consulting engagement role

In a customer / consulting engagement, typically an architect develops a solution architecture.

The patterns based use case analysis is designed to identify the appropriate SOA scenario from the generic use case criteria, as a step in a larger process or methodology. The full process includes the mapping of reusable patterns assets that apply to the scenario. The reusable patterns are used to accelerate the solution architecture and provide support for key architectural decisions. This approach is typically used for customer / consulting engagements.

For example, a business analyst or architect works with the customer to collect and define the business requirements. The customer requirements are then mapped to the generic uses cases to better identify the capabilities of the appropriate solution offering. After the solution offering is selected, the process maps the use cases to the appropriate reusable patterns, which are used to accelerate the creation of the solution design.

We envision the process being used in two ways by an architect or specialist:

- Case 1: Reuse patterns assets to accelerate the creation of the solution architecture.

  In this case, the architect will use the process as follows:

  - Capture the Customer Requirements.
  - Fit-Gap Analysis of Generic Use Cases.
  - Select the SOA scenario.

  - Reuse patterns assets to accelerate the solution architecture
  - Select the implementation guide (model, assemble, deploy, manage)
- Case 2: Once the SOA scenario is identified, go directly to the implementation guide.
  - Capture the Customer Requirements.
  - Fit-Gap Analysis of Generic Use Cases.
  - Select the SOA scenario.
  - Select the implementation guide (model, assemble, deploy, manage)

5.2 Generic use cases for the SOA scenarios

This section includes a definition for the generic use cases. The generic use cases are used in 5.3, “Select the SOA scenario” on page 141 to select the appropriate SOA scenario.

5.2.1 U1: Reuse existing or create new application logic as a service within the enterprise

A customer has application functionality they wish to expose as a service for internal consumption. That functionality may reside in an existing application or in a new application.

Exposing application functionality as a service provides the application with a standards-based and well-defined interface. Other internal applications can use this well-defined interface to interact with the application functionality. This promotes reuse of application functionality within an enterprise, can reduce overall application maintenance costs, and makes the functionality available to a wide variety of applications.

A common example of this use case is the exposure of an existing Java application as a Web service (service provider). Applications within the enterprise use a Web service client proxy to invoke the Web service (service consumer).

5.2.2 U2: Reuse existing or create new, application logic as a service beyond the enterprise

A customer has application functionality they want to expose as a service to be accessed by an external application in the extended enterprise. That application functionality can reside in an existing application or in a new application.

The same benefits of providing a standards-based and well-defined interface for application functionality apply to this use case. Additionally, this use case carries the benefit of offering functionality to a variety of external applications.

The exposure of application functionality to an extended enterprise introduces additional non-functional requirements. For example, restricted access to the application, and security measures to protect the flow of data into and out of the extended enterprise must be considered.

A common example of this use case is the exposure of an existing Java application as a Web service, with WS-Security settings applied for confidentiality and integrity. Applications outside the enterprise use a Web service client proxy to invoke the Web service, using a digital signature and a public key.

5.2.3 U3: Point-to-point integration of enterprise apps using services

A customer wants to integrate a service with an internal application such as an enterprise information system. The internal application might not be defined as a service, and therefore might not offer a standards-based interface.

The interaction between the service and the internal application is a direct, point-to-point connection. Therefore no intelligent routing or decomposition of requests is required. The internal application could be at either end of the connection: either requesting the interaction, or responding to it.

The interaction between the service and the internal application may require an adapter to implement transport level and message level transformation.

A common example of this use case is a Web service invoking application functionality residing in CICS Transaction Server. The CICS Transaction Gateway is used as the adapter to transform the protocol and message format.

5.2.4 U4: Point-to-point integration of intra-enterprise applications using services

A customer wants to integrate a service with an external application, such as an enterprise information system. As with the previous use case, the external application may not offer a standards-based interface, the connection between the service and the external application is point-to-point, and the use of an adapter may be required.

The interaction with an external application requires additional considerations, such as the use of security in the extended enterprise.

A common example of this use case is a Web service invoking application functionality residing in CICS Transaction Server found in the extended enterprise. The CICS Transaction Gateway is used as the adapter to transform the protocol and message format.

5.2.5 U5: Allow users to invoke services simply

A customer has new or existing application functionality, already exposed as a service, that they want to make available to end users. End users can either invoke the service directly, or invoke it indirectly through presentation logic.

A common example of this use case is a Web service be invoked in a client application with a user interface such as a JSP™ and servlet based application. For example, a user selects Check Rental Car Rate and a Web service is invoked to retrieve the rate information.

5.2.6 U6: Enable loose coupling of service consumers and providers using static routing

A customer wants to decouple the point-to-point connections between service consumers and service providers. A middle tier is added to provide routing between the consumer and provider. The middle tier provides static routing, where a request for a particular service from a consumer is always routed to the same service provider. The middle tier provides connection security and basic protocol and message transformation between consumers and providers.

5.2.7 U7: Enable loose coupling of service consumers and providers using dynamic routing based on standards-based protocols

A customer wants to decouple the point-to-point connections between service consumers and service providers. A middle tier is added to provide routing between the consumer and provider. Dynamic routing determines the service provider to use based on routing rules applied to the request received from the service consumer. The middle tier provides security. It also provides basic protocol and message transformation between standards-based consumers and providers.

5.2.8 U8: Enable loose coupling of service consumers and providers using advanced dynamic routing and diverse protocols

A customer wants to extend the loose coupling between service consumers and service providers to include conversion of the message format and/or protocol

used by the service consumer to a suitable message format and protocol understood by the service provider. This requires a middle tier between the service consumer and provider to perform the conversion. The middle tier provides an advanced degree of routing logic, allowing the invocation of multiple services providers from a single service consumer request, in which case the middle tier must disaggregate and aggregate requests and responses from the service providers. The middle tier provides security. It also provides advanced protocol and message transformation between a diverse set of consumers and providers.

5.2.9 U9: Improve an existing business process flow through business process and policy modeling and simulation

A customer has an existing business process flow consisting of automated and manual tasks. They want to improve the efficiency of this business process. In order to do this, the customer fully documents the business process, and then runs simulated workloads through the process to identify bottlenecks, inefficiencies, and associated costs. This information is used to create a new, more efficient refined business process.

5.2.10 U10: Implement a new business process flow

A customer wants to create a new business process flow implementation. This implementation is either based on a completely new business process, or is a refinement of an existing business process that has previously been modeled and simulated.

The business process implementation is designed by a business analyst, and an implementation is created by an enterprise architect and a development team. The business process implementation can include automated activities exposed as services, and manual activities. These activities may reside within, or outside of, the enterprise.

5.2.11 U11: Analyze existing business process flow using monitoring

A customer has an existing business process flow implementation in place. They wish to monitor the execution of the process instances for this business process. The data captured for monitoring can use key performance indicators, dimension analysis, and situation events to analyze the process instance data.

When the monitored data has been analyzed, it can be used to drive continual business improvement, resulting in a refined business process model and business process implementation.

5.2.12 U12: Allow single-sign-on access to different services

A customer wants to provide seamless access to several services with single-sign on capability. The user will only need to authenticate once when invoking one of those services. Once authenticated, the user will be allowed to access to the other services without authentication within the same session.

In some cases, the security requirements are higher and the security context needs to be extended to include back-end systems, which enables non-repudiation of back-end system transactions.

5.2.13 U13: Personalize information based on user profile

A customer requires personalized and aggregated content-based user profile information. The personalized and aggregated content is retrieved by invoking multiple services.

For example, an agent logs on to a portal-based insurance application using a user ID and password. The content on the portal page is personalized and aggregated based on the user's profile. The agent sees a listing of his customers with pending claims in one portlet, and other personalized content in other portlets. The personalized content is retrieved by invoking corresponding services.

5.2.14 U14: Allow users to create and manage content

A customer needs to give users the possibility of easily collaborating, sharing data and information with other users on the network interactively. It also does not require the client to know the physical or direct address of other users of the solution.

Some examples of this include the use of an e-mail system, a team room, or a chat room.

5.2.15 U15: Allow users to access services through client devices

A customer wants to expose services to a wide range of client devices including traditional desktops, notebooks, PDAs, and mobile phones. The key driver for this use case is to extend the business processes to mobile users using a combination of client platform software, middleware, and wireless tautologies.

There are two primary scenarios where that use case is found:

- Scenario includes role-based and aggregated content (for example, extend the portal to browser-based devices using WebSphere Everyplace Mobile Portal)
- Scenario includes rich client platform devices (notebooks, PDAs) capable consuming and publishing services (for example, extend WebSphere Application Server to rich client devices using WebSphere Everyplace Deployment).

5.2.16 U16: Allow users to perform information inquiries

The customer wants to allow users to perform information inquiries. The data requested can reside in different data stores with multiple levels. The data is retrieved by invoking services.

A common scenario could be a Web site which presents categorized information and allows user to realize complex searches. The information may come from different sources. A data mart can contain data which is often used, while a data warehouse can contain the details.

5.2.17 U17: Populate information

The customer needs to distribute the information across different data bases, to make it accessible to a wider group of users. Thus, changes made in the main system are transmitted to several back-end sources, which can be spread across different geographic locations.

5.2.18 U18: Allow users seamless access to diverse data sources

The customer needs to give seamless access to different data sources. Thus the user that access these data is not aware of the data structures that are behind. This access may be read-only and read/write also.

5.3 Select the SOA scenario

This section provides the generic use cases as selection criteria to determine the appropriate SOA scenario. Table 5-1 on page 142 lists the generic use cases and corresponding SOA scenario that fulfill the use case requirement.

Note: We have included a description and common example of the generic use cases in 5.2, “Generic use cases for the SOA scenarios” on page 136.

Table 5-1 SOA scenario selection criteri

Generic use cases selection criteria	Service Creation	Service Connectivity	Interaction and Collaboration Services	Business Process Management	Information as a Service
U1: Reuse existing or create new application logic as a service within the enterprise	X				
U2: Reuse existing or create new, application logic as a service beyond the enterprise	X				
U3: Point-to-point integration of enterprise apps using services	X				
U4: Point-to-point integration of intra-enterprise applications using services	X				
U5: Allow users to invoke services simply	X				
U6: Enable loose coupling of service consumers and providers using static routing		X			
U7: Enable loose coupling of service consumers and providers using dynamic routing based on standards-based protocols		X			
U8: Enable loose coupling of service consumers and providers using advanced dynamic routing and diverse protocols		X			
U9: Improve an existing business process flow through business process and policy modeling and simulation				X	
U10: Implement a new business process flow.				X	
U11: Analyze existing business process flow using monitoring				X	
U12: Allow single-sign-on access to different services			X		
U13: Personalize information based on user profile			X		
U14: Allow users to create and manage content			X		
U15: Allow users to access services through client devices	X		X		
U16: Allow users to perform information inquiries					X
U17: Populate information					X

Generic use cases selection criteria	Service Creation	Service Connectivity	Interaction and Collaboration Services	Business Process Management	Information as a Service
U18: Allow users seamless access to diverse data sources					X

5.4 Reuse patterns assets to accelerate the solution architecture

The target audience of this section is IT architects interested in leveraging the knowledge captured in the Patterns for e-business, with the objective of accelerating the solution architecture.

The section is organized into the following topics:

- Introduction to the Patterns for e-business
- Reuse of patterns assets within development methodology
- Patterns for the SOA scenarios

Note: This section describes where to find the detailed reusable patterns information used to accelerate the solution architecture.

5.4.1 Introduction to the Patterns for e-business

This section provides an introduction to the Patterns for e-business.

The role of the IT architect is to evaluate business problems and build solutions to solve them. The architect begins by gathering input about the problem, developing an outline of the desired solution, and considering any special requirements that need to be factored into that solution. The architect then takes this input and designs the solution, which can include one or more computer applications that address the business problems by supplying the necessary business functions.

To improve the process over time, we need to capture and reuse the experience of the IT architects in such a way that future engagements can be made simpler and faster. We do this by capturing knowledge gained from each engagement and using it to build a repository of assets. IT architects can then build future solutions based on these proven assets. Reusing proven assets saves time, money, and effort and helps ensure delivery of a solid, properly architected solution.

The IBM Patterns for e-business (P4eb) help facilitate this reuse of assets. Their purpose is to capture and publish e-business artifacts that have been used, tested, and proven to be successful. The information captured by them is assumed to fit the majority, or 80/20, situation.

Patterns for e-business layered asset model

The Patterns for e-business approach enables architects to implement successful e-business solutions through the reuse of components and solution elements from proven successful experiences. The Patterns approach is based on a set of layered assets that can be exploited by any existing development methodology. These layered assets are structured in a way that each level of detail builds on the last and include:

- *Customer requirements* ultimately define the selected solution. We have developed a series of generic use cases that are representative of common SOA scenarios. The customer requirements are analyzed and when appropriate mapped to the generic use cases. The generic use cases are used as selection criteria to identify the appropriate SOA scenario and reusable patterns assets.
- *Business patterns* identify the interaction between users, businesses, and data.
- I*ntegration patterns* tie multiple Business patterns together when a solution cannot be provided based on a single Business pattern.
- *Composite patterns* represent commonly occurring combinations of Business patterns and Integration patterns.
- *Application patterns* provide a conceptual layout that describe how the application components and data within a Business pattern or Integration pattern interact.
- *Runtime patterns* define the logical middleware structure that supports and Application pattern. Runtime patterns depict the major middleware nodes, their roles, and the interfaces between these nodes.
- *Product mappings* identify proven and tested software implementations for each Runtime pattern.

- *Best-practice guidelines* discuss design, development, deployment, and management of e-business applications.

Figure 5-2 on page 145 shows these assets and their relationships to each other.

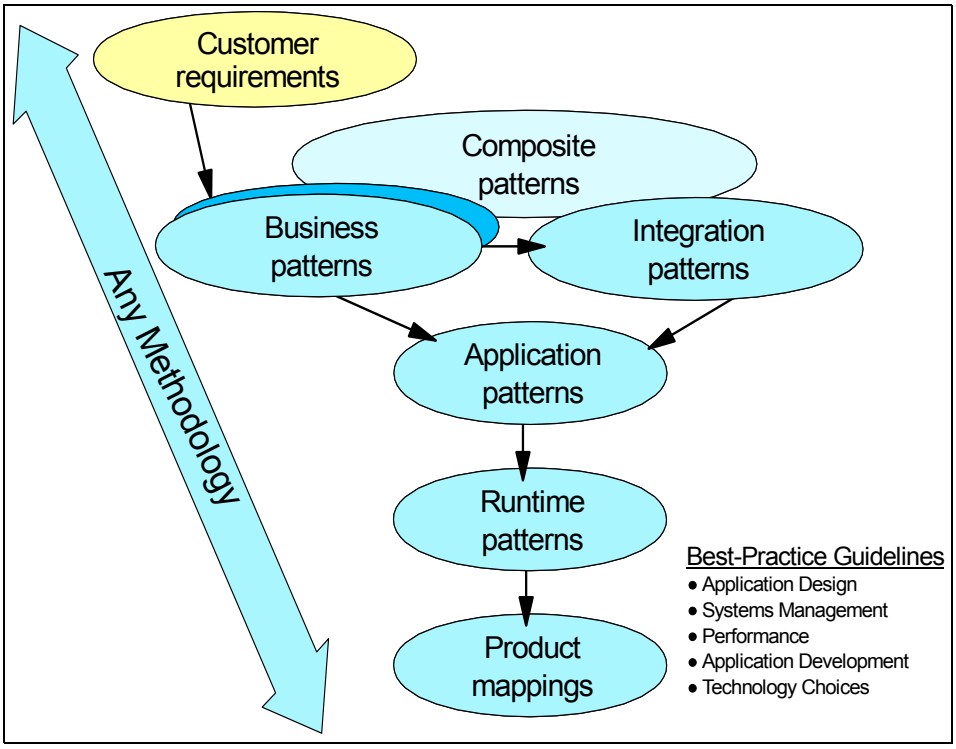


Figure 5-2 The Patterns for e-business layered asset model

Summary description of Business patterns

Table 5-2 provides a brief description of the Business, Integration and Composite patterns.

Table 5-2 Summary description for the Business, Integration and Composite patterns

Business, Integration, Composite Pattern	Description
Self-Service	The Self-Service business pattern captures the essence of direct interactions between interested parties and a business. Interested parties include customers, business partners, stakeholders, employees, and all other individuals with whom the business intends to interact.

Business, Integration, Composite Pattern	Description
Collaboration	The Collaboration business pattern enables interaction and collaboration between users. This pattern can be observed in solutions that support small or extended teams who need to work together in order to achieve a joint goal.
Extended Enterprise	The Extended Enterprise business pattern covers interactions between applications from different enterprises. This pattern can be observed in solutions that implement programmatic interfaces to connect inter-enterprise applications
Information Aggregation	The Information Aggregation business pattern exists in e-business solutions that allow users to access and manipulate data that is aggregated from multiple sources. This Business pattern captures the process of taking large volumes of data, text, images, video, and so on, and using various user-controlled tools to extract useful information from them.
Application Integration	The Application Integration pattern serves to integrate multiple Business patterns or to integrate applications and data within an individual Business pattern.
Access Integration	The Access Integration pattern gives users a single, consistent, and seamless access mechanism to various applications that would otherwise require the use of several different access mechanisms.
Portal Composite	The Portal composite pattern leverages various mechanisms (e.g. content management, user interface formatting and display, data aggregation) to bring together the appropriate information and existing systems to serve the goals of the business.

Patterns for e-business Web site

The layers of patterns, along with their associated links and guidelines, allow the architect to start with a problem and a vision for the solution and then find a pattern that fits that vision. In order to navigate from top down from one level to another, a decision matrix will be provided to assist the architect in making the right decision.

Then, by drilling down using the patterns process, the architect can further define the additional functional pieces that the application need to succeed. Finally, the architect can build the application using coding techniques that are outlined in the associated guidelines.

The Patterns Web site provides an easy way of navigating through the layered Patterns assets to determine the most appropriate assets for a particular engagement.

For easy reference, see the Patterns for e-business Web site:

http://www.ibm.com/developerWorks/patterns/

5.4.2 Reuse of patterns assets within development methodology

The objective of this section is to describe to business analysts, architects, developers and specialists how the IBM Patterns for e-business can be used within the context of a development life cycle methodology. For illustration purposes, we will use the Rational Unified Process (RUP), however the concepts can be applied to other methodologies such as IBM Method.

Note: The Rational Unified Process (RUP) is a development life cycle methodology. The IBM Rational Method Composer is an Eclipse based product that allows you to easily customize RUP.

More information on the RUP and the IBM Rational Method Composer can be found at:

http://www.ibm.com/software/awdtools/rup/

Figure 5-3 on page 148 depicts the RUP disciplines for each project phase and discipline. As seen in Figure 5-3 the architect provides his/her greatest effort in the Inception and the Elaboration phases. The Patterns for e-business are used primarily to accelerate the architecture development steps found in the RUP Requirements and Analysis and Design disciplines.

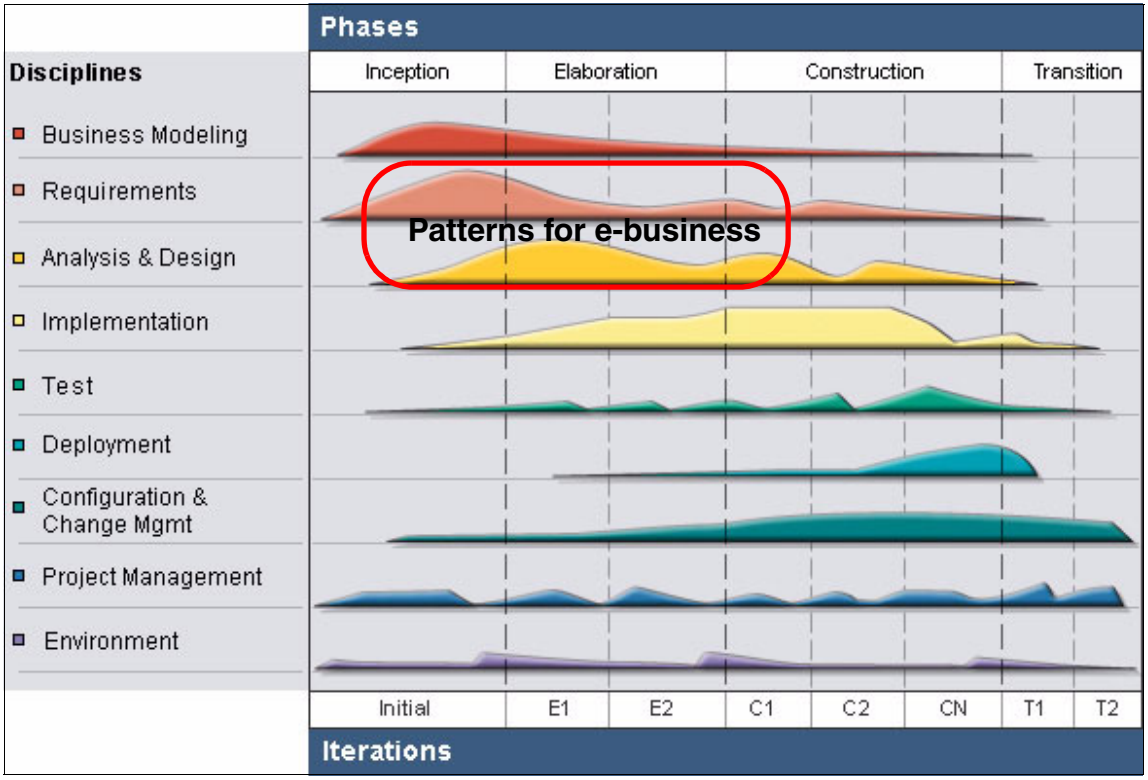


Figure 5-3 Rational Unified Process (RUP) disciplines and phases

Mapping of reusable assets to RUP work products

Table 5-3 provides a mapping of reusable assets found in the Patterns for e-business and the process described in 5.1, “Process for using SOA scenarios and patterns” on page 132, that can be used to accelerate the creation of RUP work products.

Note: The working examples found in the implementation guides (see “Select the implementation guide” on page 151) include end-to-end working examples illustrating how to use the process and assets listed in Table 5-3

Table 5-3 Mapping of reusable assets for RUP

RUP Disciplines	Reusable assets	RUP work products
Requirements	Generic use cases and SOA scenario selection criteria	► Vision ► Use Case View

RUP Disciplines	Reusable assets	RUP work products
Analysis and Design	Application patterns	► Component Model / Services Model
	Runtime patterns	► Component Model / Services Model
	Runtime patterns and Product mapping	► Implementation Model ► Deployment Model
Implementation	Implementation Guides working examples	► Implementation Model
Test		► Test
Deployment		► Deployment Model

Map reusable assets to IBM Method and SOMA work products

Table 5-4 provides a mapping of reusable assets found in the Patterns for e-business and the process described in 5.1, “Process for using SOA scenarios and patterns” on page 132, that can be used to accelerate the creation of IBM Method work products.

Table 5-4 Mapping of reusable assets for IBM Method and SOMA

IBM Method work products	Reusable assets
Use Case Model	Generic use cases and SOA scenario selection criteria
Architecture Overview Diagram	Application patterns
Component Model / Service Model	Application and Runtime patterns
Operational Model	Runtime patterns and Product Mapping
► Build Components / Services ► Deploy Components / Services	Implementation Guides working examples

5.4.3 Patterns for the SOA scenarios

In 5.3, “Select the SOA scenario” on page 141, we showed how to identify a suitable SOA scenario from the generic use cases. Once we know which SOA scenario meets our customer requirements, the next step is to identify reusable assets found in the Patterns for e-business that we can leverage to accelerate the creation of our solution architecture.

Table 5-5 provides a mapping from the SOA scenario to the appropriate patterns reference material for the solution architecture.

Table 5-5 Patterns reference material for the solution architecture

SOA scenario	Patterns reference material for solution architecture
Service Creation	Primary reference: ► *Patterns: SOA Foundation - Service Creation Scenario*, SG24-7240 (this redbook) – Chapter 6, “Patterns for the Service Creation scenario” on page 155 includes detailed information on each of the reusable patterns assets for the Service Creation scenario. – Chapter 7, “Business scenario and solution architecture” on page 173 includes an example that demonstrates how to apply the reusable patterns assets to design the solution architecture for the fictitious ITSO Car Rental company (representative of the Service Creation scenario). Additional reference: ► *Patterns: Extended Enterprise SOA and Web Services*, SG24-7135
Service Connectivity	Primary reference: ► *Patterns: SOA Foundation - Service Connectivity Scenario*, SG24-7228 Additional references: ► Application Integration pattern found at: http://www.ibm.com/developerworks/patterns/application/index.html ► *Enabling SOA Using WebSphere Messaging*, SG24-7163 ► *Patterns: Implementing Self-Service in an SOA Environment*, SG24-6680

SOA scenario	Patterns reference material for solution architecture
Interaction and Collaboration Services	Primary references: ► Portal Composite pattern found at: http://www.ibm.com/developerworks/patterns/portal/index.html ► *Portal Application Design and Development Guidelines*, REDP3829 Additional references: ► *WebSphere Portal Best Practices*, REDP4100 ► *Identity and Access Management Solutions, Using WebSphere Portal V5.1, Tivoli Identity Manager V4.5.1, and Tivoli Access Manager V5.1*, SG24-6692 ► *Develop and Deploy a Secure Portal Solution, Using WebSphere Portal V5 and Tivoli Access Manager V5.1*, SG24-6325 ► *A Portal Composite Pattern Using WebSphere Portal V5*, SG24-6087 ► *Patterns: SOA Client Access Integration Solutions*, SG24-6775 ► *Patterns: Portal Search Custom Design*, SG24-6881
Business Process Management	Primary reference: ► *Patterns: SOA Foundation - Business Process Management Scenario*, SG24-7234 Additional reference: ► *Business Process Management: Modeling through Monitoring Using WebSphere V6 Products*, SG24-7148
Information as a Service	Primary reference: ► *Patterns: Information Aggregation and Data Integration with DB2 Information Integrator*, SG24-7101 Additional reference: ► Information Aggregation business pattern found at: http://www.ibm.com/developerworks/patterns/bi/index.html

5.5 Select the implementation guide

Table 5-6 provides a mapping to the appropriate implementation guide given the SOA scenario. The resources referenced in this section include working examples which demonstrate how to implement (model, assemble, deploy, manage) the SOA scenario for a select set of generic use cases.

Table 5-6 Select the implementation guide given the selected SOA scenario

SOA scenario	Implementation guide
Service Creation	Primary reference: ► *Patterns: SOA Foundation - Service Creation Scenario*, SG24-7240 – Part 2, “Service Creation scenario example” on page 171 includes several chapters for an end-to-end implementation example. Additional references: ► *WebSphere Version 6 Web Services Handbook, Development and Deployment*, SG24-6461 ► *Patterns: Extended Enterprise SOA and Web Services*, SG24-7135 ► *Rational Application Developer V6 Programming Guide*, SG24-6449 ► *IBM Tivoli Composite Application Manager V6.0 Family: Installation, Configuration, and Basic Usage*, SG24-7151
Service Connectivity	Primary reference: ► *Patterns: SOA Foundation - Service Connectivity Scenario*, SG24-7228 Additional references: ► *Getting Started with WebSphere Enterprise Service Bus V6*, SG24-7212 ► *Patterns: Implementing Self-Service in an SOA Environment*, SG24-6680 ► *Enabling SOA Using WebSphere Messaging*, SG24-7163

SOA scenario	Implementation guide
Interaction and Collaboration Services	Primary references: ► *IBM Rational Application Developer V6 Portlet Application Development and Portal Tools*, SG24-6681 ► *IBM WebSphere Portal for Multiplatforms V5.1 Handbook*, SG24-6689 Additional references: ► *IBM WebSphere Portal for Multiplatforms V5.1 Handbook*, SG24-6689 ► *Identity and Access Management Solutions, Using WebSphere Portal V5.1, Tivoli Identity Manager V4.5.1, and Tivoli Access Manager V5.1*, SG24-6692 ► *Develop and Deploy a Secure Portal Solution, Using WebSphere Portal V5 and Tivoli Access Manager V5.1*, SG24-6325 ► *Patterns: SOA Client Access Integration Solutions*, SG24-6775
Business Process Management	Primary reference: ► *Patterns: SOA Foundation - Business Process Management Scenario*, SG24-7234 Additional reference: ► *Business Process Management: Modeling through Monitoring Using WebSphere V6 Products*, SG24-7148
Information as a Service	Primary reference: ► *Patterns: Information Aggregation and Data Integration with DB2 Information Integrator*, SG24-7101

6

Patterns for the Service Creation scenario

This chapter describes the reusable patterns assets from the Patterns for e-business, that can be applied to the solution architecture of a Service Creation scenario. The patterns provide a very useful decomposition of the Service Creation scenario solution domain to provide a deeper architectural understanding.

We have organized the sections by the Business or Integration pattern. The corresponding Application pattern(s), Runtime pattern, and Product mapping are included within the Business and Integration pattern sections.

The chapter is organized into the following sections:

- Patterns for the Service Creation scenario
- Self Service business pattern
- Application Integration pattern
- Extended Enterprise business pattern

6.1 Patterns for the Service Creation scenario

Table 6-1 includes a summary of the Application patterns that fulfill the generic use cases (defined in “Generic use cases for the SOA scenarios” on page 136) for the Service Creation scenario. In the remaining sections of this chapter, we will provide a detailed description of the Business and Integration patterns, Runtime patterns, and Product mappings related to the Application patterns summarized in Table 6-1.

Table 6-1 Summary of patterns for the Service Creation scenario

		Generic use cases				
Business and Integration pattern	**Application pattern**	**U1: Expose existing or new application logic as a service within the enterprise**	**U2: Expose existing or new application logic as a service beyond the enterprise**	**U3: Point-to-point integration of enterprise applications using services**	**U4: Point-to-point integration of enterprise applications using services**	**U5: Allow users to invoke services**
Self Service	Directly Integrated Single Channel	X		X		X
Application Integration	Direct Connection: ► Call Connection variation	X		X		
	► Message Connection variation	X		X		
Extended Enterprise	Exposed Direct Connection: ► Exposed Call Connection variation		X		X	
	► Exposed Message Connection variation		X		X	

6.2 Self Service business pattern

The Self-Service business pattern captures the essence of direct interactions between interested parties and a business. Interested parties include customers, business partners, stakeholders, employees, and all other individuals with whom the business intends to interact. For simplicity, these interested parties are referred to as users. In this definition, a business represents various types of organizations including large enterprises, small and medium businesses, government agencies, and so on.

Note: For more detailed information, refer to the following:

- Self Service business pattern on the Patterns for e-business Web site:

  http://www.ibm.com/developerworks/patterns/u2b/index.html

- IBM Redbooks:
  - *Patterns: Implementing Self-Service in an SOA Environment*, SG24-6680-01

6.2.1 Directly Integrated Single Channel

In this section we first describe this Application pattern, and the business and IT drivers used to select it. Next, we describe the corresponding Runtime pattern for the Application pattern. Lastly, we select the suitable products for the chosen Runtime pattern.

Application pattern

The Directly Integrated Single Channel application pattern, shown in Figure 6-1 on page 158, provides a structure for applications that need one or more point-to-point connections with back-end applications, but only need to focus on one delivery channel.

The user interface is handled by the presentation tier. The business logic can reside in the Web application tier and in the back-end application.

The Web-application tier has access to local data that exists primarily as a result of this application (for example, customer profile information or cached data). It is also responsible for accessing one or more back-end applications.

The back-end applications contain business logic and are responsible for accessing the existing back-end data. The communication between the presentation tier and Web application tier is synchronous. The communication between the Web application tier and the back-end can be either synchronous or

asynchronous, depending on the characteristics and capabilities of the back-end application.

Key business drivers for the Directly Integrated Single Channel application pattern are as follows:

- Improve the organizational efficiency and reduce the latency of business events. These improvements are achieved by providing real-time access to back-end applications and data from Web applications.
- Leverage existing investments and existing skills.

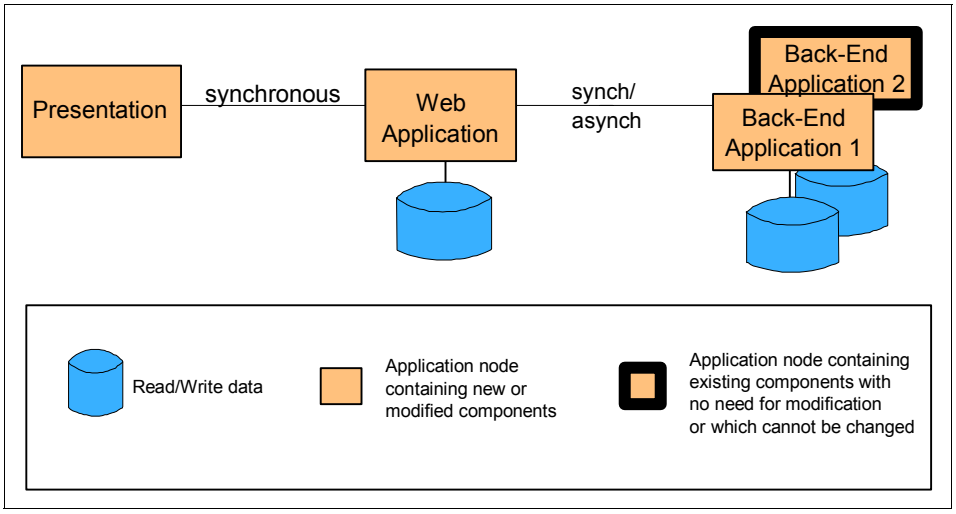


Figure 6-1 Self-Service::Directly Integrated Single Channel application pattern

Runtime pattern

The Runtime pattern shown in Figure 6-2 on page 159 represents one solution for the Directly Integrated Single Channel application pattern. This runtime is a starting point for extending business to the Web. The nodes in Figure 6-2 on page 159 depicted with dotted lines provide important functionality, but are optional for the focus of the Runtime pattern.

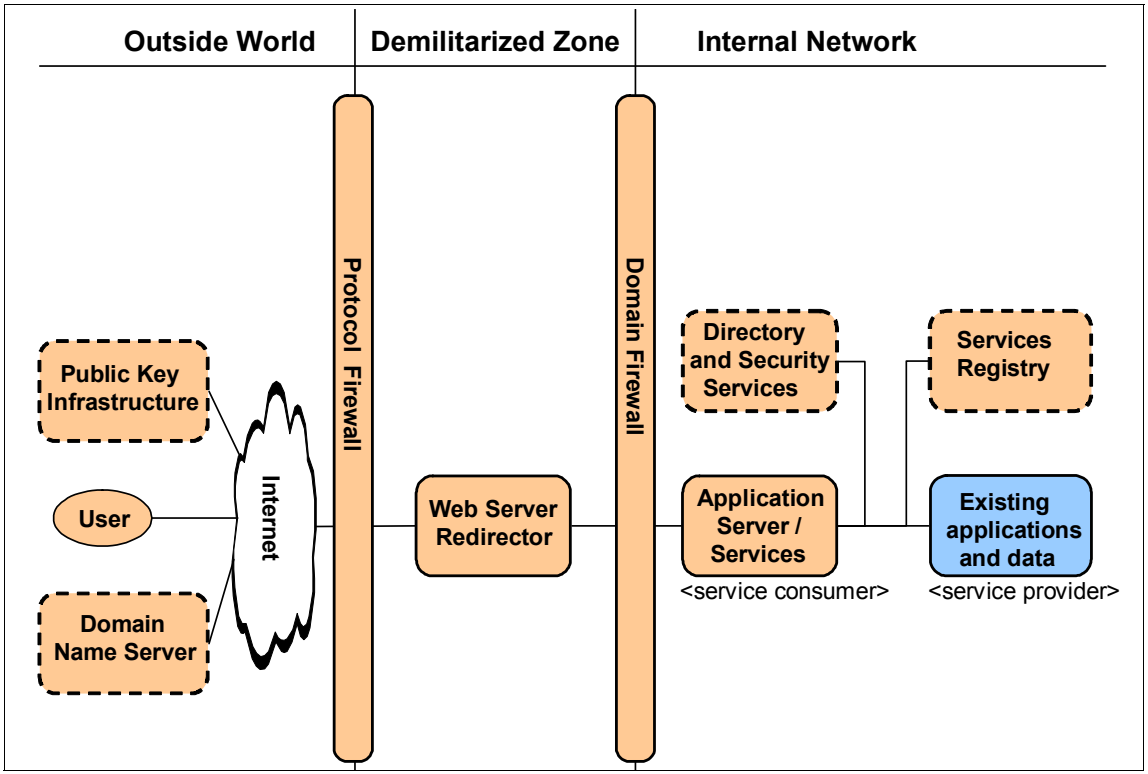


Figure 6-2 Self Service::Directly Integrated Single Channel runtime pattern

This Runtime pattern uses a Web Server Redirector containing the Web server and an application server plug-in, effectively splitting the function of a Web application server across two machines. Placing the application server in the internal network provides a higher level of security than you might find in installations with one Web application server protected by a single layer of firewall security. The application server node will run both presentation and business logic. The Web server remains in the DMZ and serves static pages. The Web server redirector forwards requests from the Web server to the application server.

Product mapping

Figure 6-3 on page 160 shows the suitable products for the Directly Integrated Single Channel runtime pattern.

Note: The Existing applications and data node, lists WebSphere Application Server and Web services. There are many other possible Product mappings and technologies such as JMS and JCA that can be used as services.

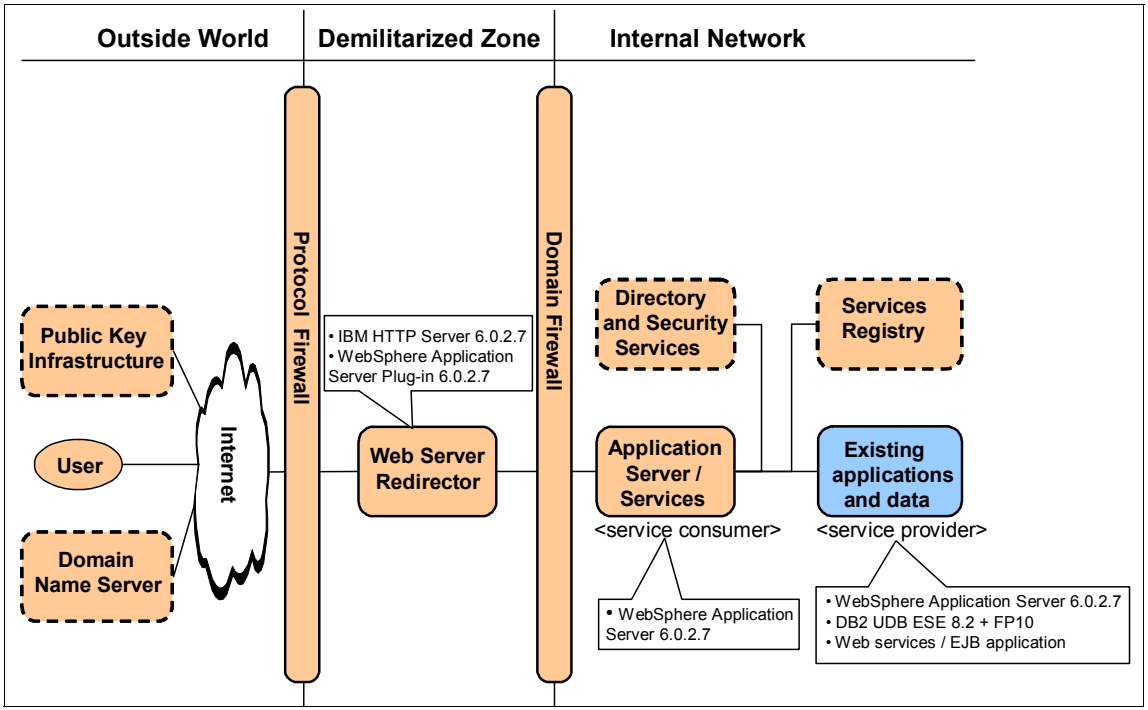


Figure 6-3 Self Service::Directly Integrated Single Channel::Product mapping

6.3 Application Integration pattern

Integration patterns can be used to combine different Business patterns that are involved in one solution, or to support more advanced features inside one Business pattern.

Application Integration patterns are a type of Integration pattern that refer to the integration of data and applications within a Business pattern or integration between patterns.

Note: For more detailed information refer to the following:

- Application Integration pattern on the Patterns for e-business Web site:

  http://www.ibm.com/developerworks/patterns/application/index.html

- IBM Redbooks:
  - *Patterns: Service-Oriented Architecture and Web Services*, SG24-6303

6.3.1 Direct Connection

In this section we first describe this Application pattern, and the business and IT drivers used to select it. Next we describe the corresponding Runtime pattern for the Application pattern. Lastly, we select the suitable products for the chosen Runtime pattern.

Application pattern

The Direct Connection application pattern shown in Figure 6-4 on page 161, represents the simplest interaction type, and is based on a 1-to-1 topology. It allows a pair of applications within the organization to directly communicate with each other. Interactions between a source and a target application can be arbitrarily complex. Generally, complexity can be addressed by breaking down interactions into more elementary interactions.

More complex point to point connections will have modeled connection rules, such as business rules associated with them. Connection rules are generally used to control the mode of operation of a connector depending on external factors.

The key business and IT drivers for the Direct Connection application pattern are as follows:

- Improve the organizational efficiency
- Support a structured exchange within the organization
- Support real-time one-way message flows
- Support real-time request/reply message flows, and reduce the latency of business events. This is achieved by allowing one application to gain real-time access to another.

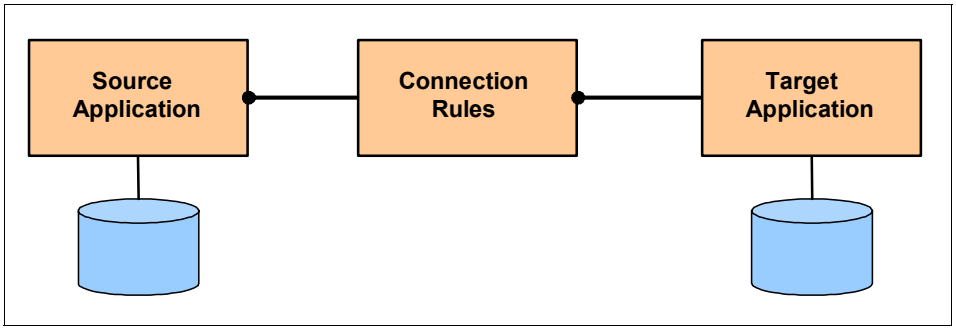


Figure 6-4 Application Integration::Direct Connection application pattern

The Application Integration::Direct Connection application pattern has two variations; Call Connection and Message Connection. The variation required

depends on whether the initiating source application needs an immediate response from the target application in order to continue with execution or not.

- Call Connection variation (see Figure 6-5): This variation is used when an immediate response is required from the target application (for example, request and response).
- Message Connection variation (see Figure 6-6): This variation is used when an immediate response from the target application is not required.

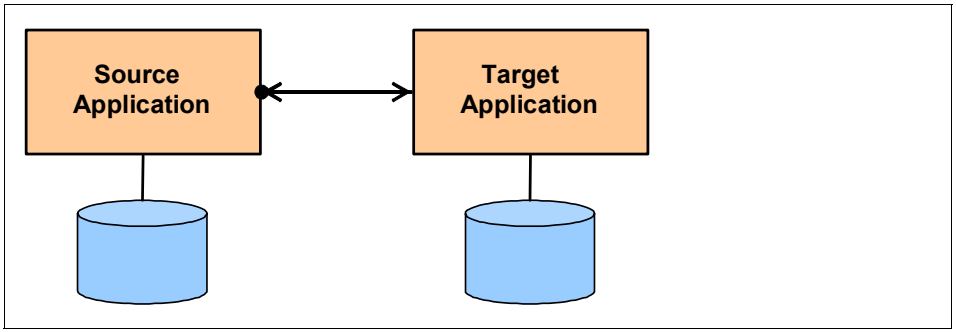


Figure 6-5 Application Integration::Direct Connection=Call Connection variation application pattern

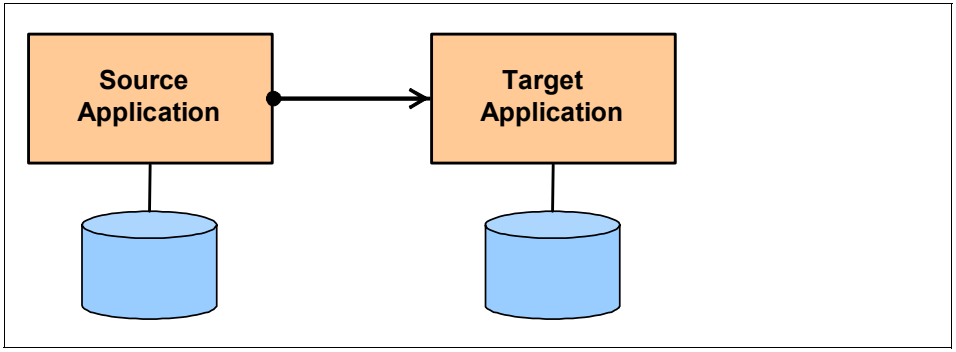


Figure 6-6 Application Integration::Direct Connection=Message Connection variation application pattern

Runtime pattern

The Direct Connection application pattern shares the same Runtime pattern (see Figure 6-7 on page 163) for both Call Connection and Message Connection.

The target and source applications are deployed to application servers as shown in the figure Figure 6-7 on page 163.

The connector itself can be explicitly or implicitly modeled. If the connector is explicitly modeled, the modeler can use decomposition and abstraction techniques to expand the connector to the appropriate level of detail. It can be used for integrating a source and target application that use different protocols (single adapter connector). Another possibility is to use federated adapters, to improve reuse in multiple point to point scenarios.

Additionally, Rules Directory and Domain QoS Providers can be needed.

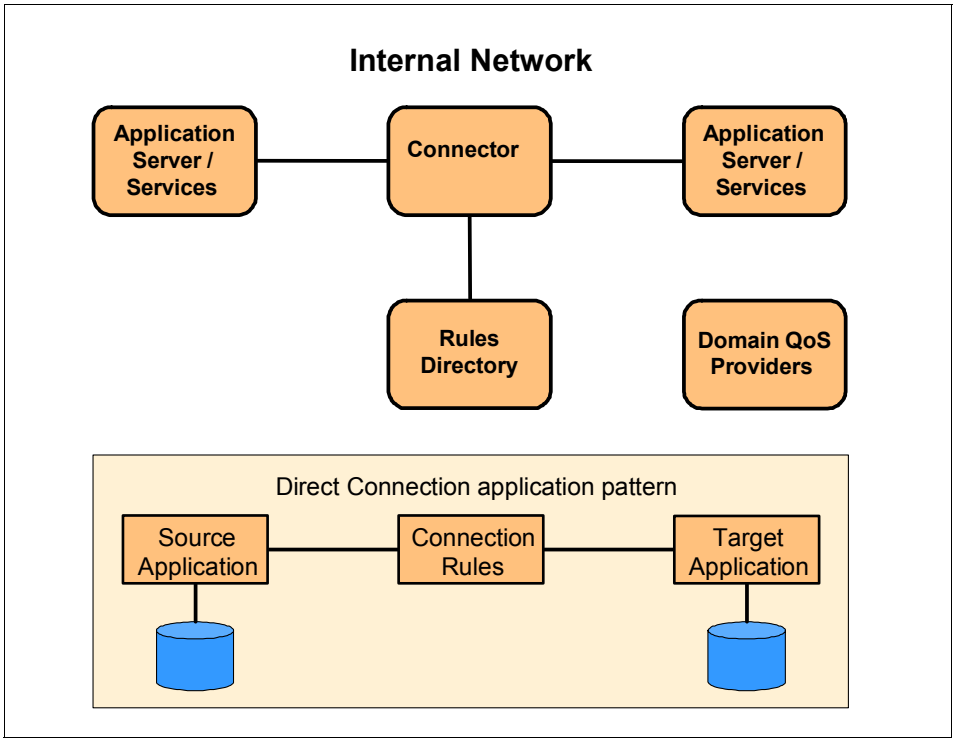


Figure 6-7 Application Integration::Direct Connection runtime pattern

Product mapping

The Product mapping displayed in Figure 6-3 on page 160 is appropriate for both the Call Connection and Message Connection variations of the Direct Connection runtime pattern.

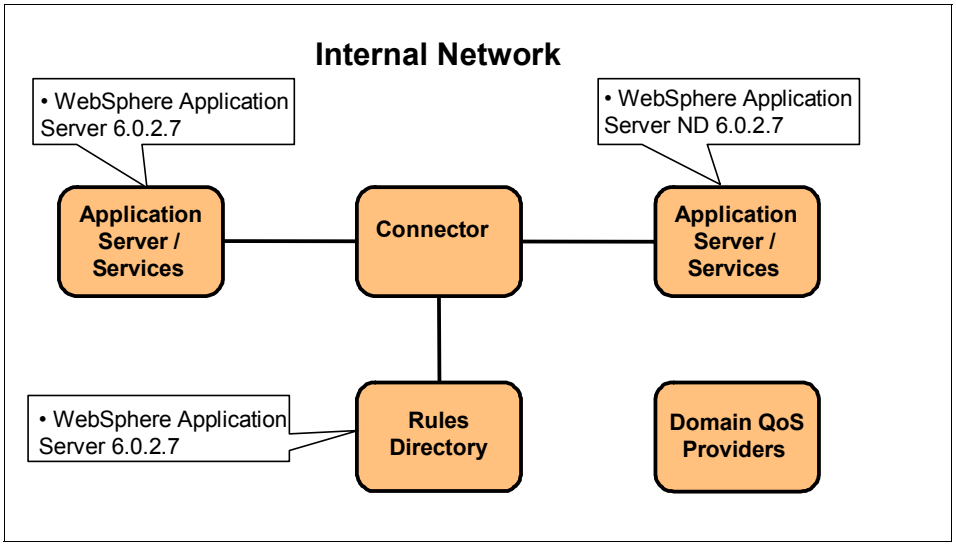


Figure 6-8 Application Integration::Direct Connection::Product mapping

6.4 Extended Enterprise business pattern

The Extended Enterprise business pattern, also known as the Business-to-Business or B2B patterns, covers interactions between applications from different enterprises.

Note: For more detailed information refer to the following:

- Extended Enterprise business pattern on the Patterns for e-business Web site:

  http://www.ibm.com/developerworks/patterns/b2bi/index.html

- IBM Redbooks:
  - *Patterns: Extended Enterprise SOA and Web Services*, SG24-7135

6.4.1 Exposed Direct Connection

In this section we first describe this Application pattern, and the business and IT drivers used to select it. Next, we describe the corresponding Runtime pattern for the Application pattern. Lastly, we select the suitable products for the chosen Runtime pattern.

Application pattern

The Exposed Direct Connection application pattern (see Figure 6-9) represents the simplest interaction between applications from different enterprises, based on a 1-to-1 topology.

When more complexity is needed, we often use Connection Rules. Some examples of these rules are as follows:

- Business data mapping rules
- Autonomic rules
- Security rules
- Capacity and availability rules

The key business and IT drivers for the Exposed Direct Connection application pattern are as follows:

- Improve organizational efficiency.
- Reduce the latency of business events.
- Support a structured exchange with trading partners.
- Support real-time one-way message flows to partner processes.
- Support real-time request/reply message flows to partner processes.
- Leverage existing skills.
- Leverage the existing investment.
- Enable back-end application integration.
- Minimize application complexity.

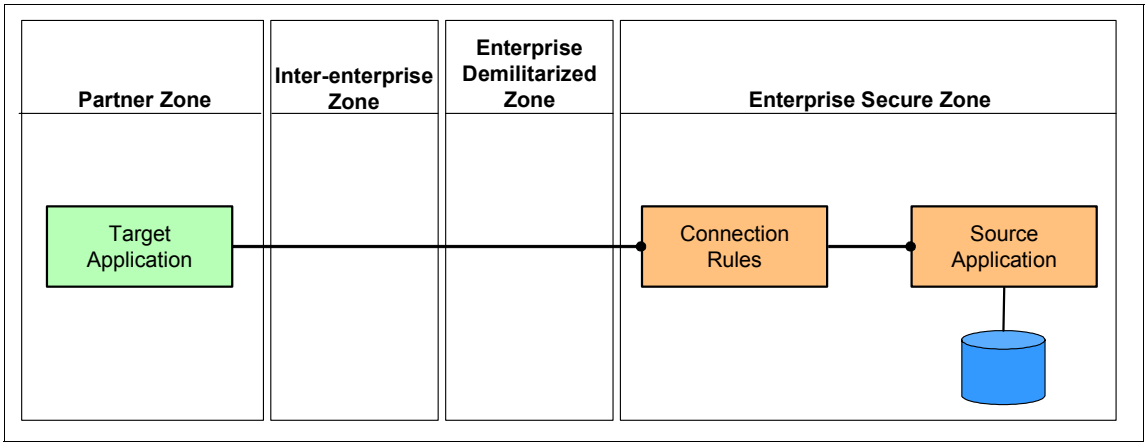


Figure 6-9 Extended Enterprise::Exposed Direct Connection application pattern

There are two variations of the Exposed Direct Connection application pattern:

- Call Connection variation (see Figure 6-10 on page 166): This variation is used when an immediate response is required from the target application (for example, request and response).

- Message Connection (see Figure 6-11 on page 166) variation: This variation is used when an immediate response from the target application is not required.

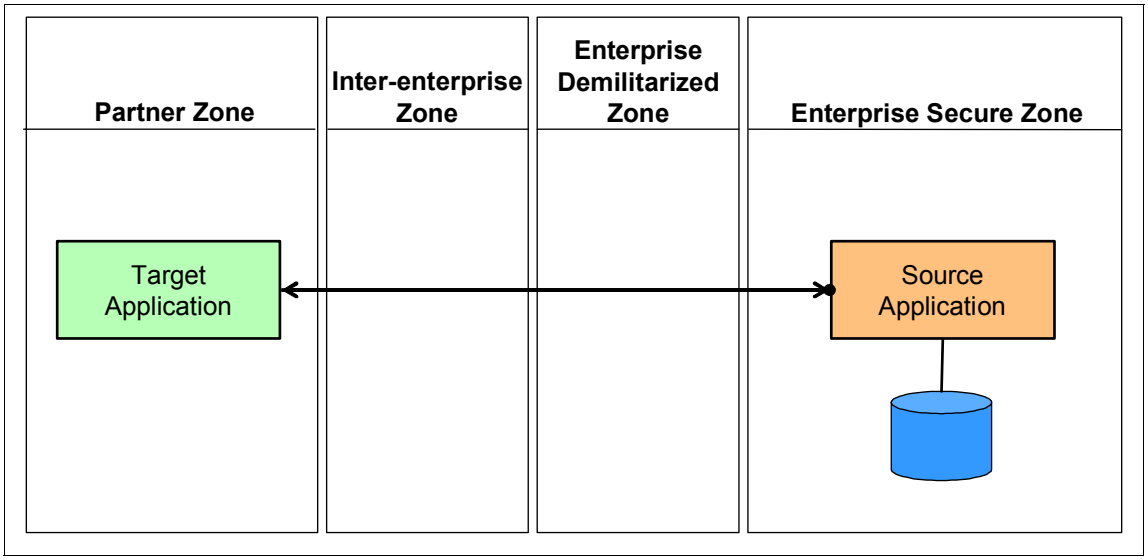


Figure 6-10 Extended Enterprise::Exposed Direct Connection=Call Connection variation

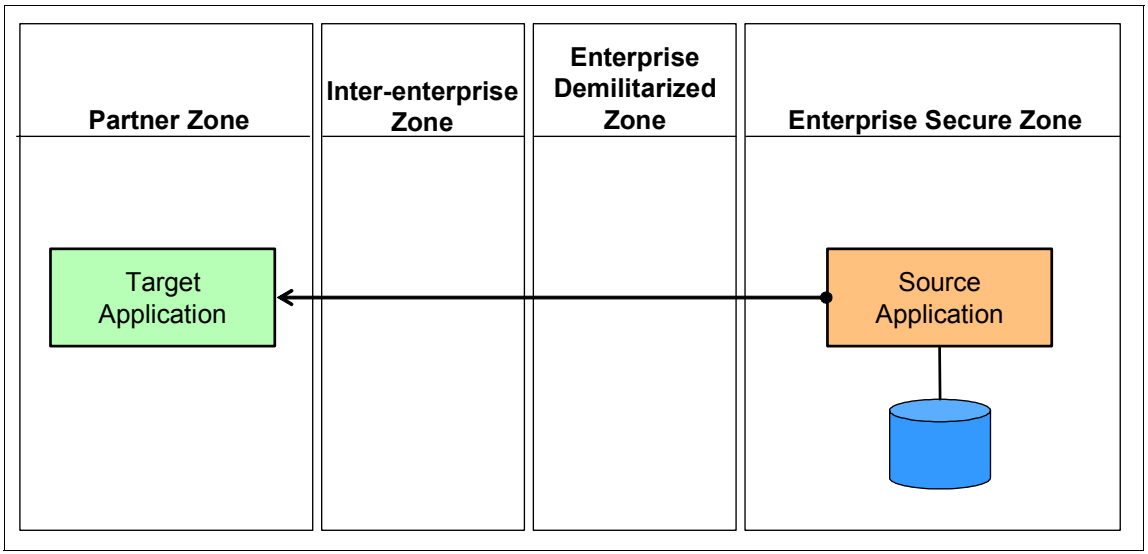


Figure 6-11 Extended Enterprise:: Exposed Direct Connection=Message Connection variation

Runtime pattern

This Exposed Direct Connection runtime pattern (see Figure 6-12) allows two different organizations to talk to each other with a mutually agreed upon message format and protocol. Each partner can use their own internal messaging format, then use a connector adapter to convert from the internal format to the external format.

The connector itself can be explicitly or implicitly modeled. If the connector is explicitly modeled, the modeler can use decomposition and abstraction techniques to expand the connector to the appropriate level of detail. The term *Connector* can be qualified by both the connector variation and by the interaction variation. Some examples are:

- Adapter Connector
- Path Connector
- Message Connector
- Call Connector
- Call Adapter Connector

The target application relies on services provided by its hosting server. These are modeled using the Application Server/Services component.

The Rules Directory might or might not exist. If it does exist, it is a modeling decision as to whether the rules must be shown in the Runtime pattern. For example, analysis might determine that connection rules are not an important part of the solution, so the Rules Directory might be left off the Runtime pattern. The Directory and Security Services node supplies authentication and authorization services. It also holds the user ID as well as password and related privileges. This node typically leverages LDAP-based directories. It also contains configuration information needed to support secure access between the enterprise and partner services.

Figure 6-12 shows a standard pattern of Path Connectors (firewalls and network infrastructure), but other variations do exist with fewer or more firewalls. The secure zone Connector is primarily concerned with logical connection of the Path Connector to the Application Services, and will therefore often be modeled as an Adapter Connector.

Less secure applications and connectors can be placed within the Demilitarized Zone, depending on local security policies. The less secure applications are usually placed as shown in Figure Figure 6-12.

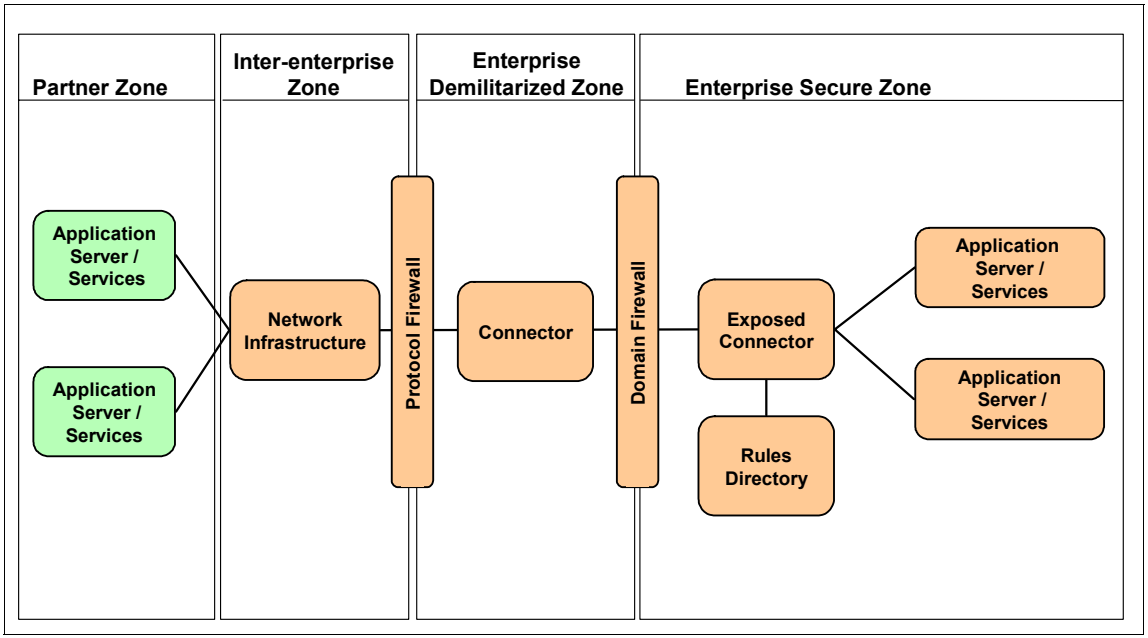


Figure 6-12 Exposed Direct Connection Pattern::Runtime pattern

Product mapping

The Product mapping displayed in Figure 6-13 on page 169 is appropriate for both the Call Connection and Message Connection variations of the Direct Connection runtime pattern.

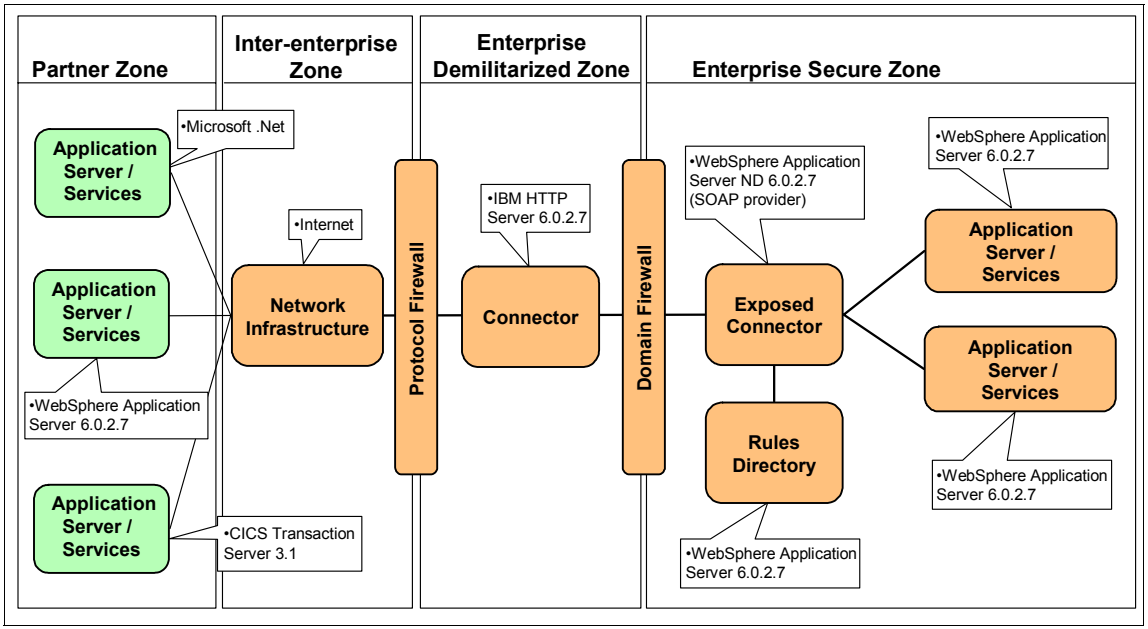


Figure 6-13 Exposed Direct Connection Pattern::Product mapping

Part 2

Service Creation scenario example

This part of the redbook provides an end-to-end working example representative of the direct exposure of existing J2EE applications as services for the Service Creation scenario.

7

Business scenario and solution architecture

A key objective of this chapter is to demonstrate how to model business and IT requirements and develop a solution architecture representative of the Service Creation scenario using best practices and patterns.

The business scenario presented in this chapter is for the fictitious ITSO Car Rental company. The scenario will describe how to expose existing functionality of the ITSO Car Rental application as services to be consumed by ITSO internal applications. In addition, the solution exposes selected services to be consumed by external Travel partner applications to make rental car reservations.

This chapter is organized into the following sections:

- Business model
- Requirements
- Service identification and design
- Solution architecture
- Where to find implementation details

Note: All the topics that can be found in a customer engagement business model and solution architecture are outside the scope of this book. The level of detail found in this chapter is intended to put the scenario in context for the hands on chapters that follow.

7.1 Business model

This business context section includes a description of the initial context of the ITSO Car Rental organization and insight into the key business drivers and IT challenges facing the organization.

7.1.1 Initial context

The ITSO Car Rental company conducts business from 1,500 locations world-wide. The company offers a wide variety of current-model cars on a short-term rental basis including daily, weekly, and monthly at a variety of locations such as airports, residential areas, and resorts.

Customers can rent a vehicle through any of the ITSO Car Rental call centers. The existing ITSO Car Rental application provides support for the following business processes to rent a vehicle (see Figure 7-1):

- Reserve vehicle
- Check-out vehicle
- Check-in vehicle
- Payment processing

The existing ITSO Car Rental application is a J2EE based Web application accessible internally to Web browser clients. The existing application consists of JSPs for the user interface, servlets, and EJBs to communicate with the application database. Servlets access the EJBs using session beans.

The J2EE application uses a JCA adapter to access a payment application hosted by CICS. At a more detailed level, a session EJB uses the CICS ECI resource adapter to communicate with the CICS Transaction Gateway (CTG) to access CICS Transaction Server. The CICS ECI resource adapter is a JCA adapter for WebSphere Application Server packaged with the CTG.

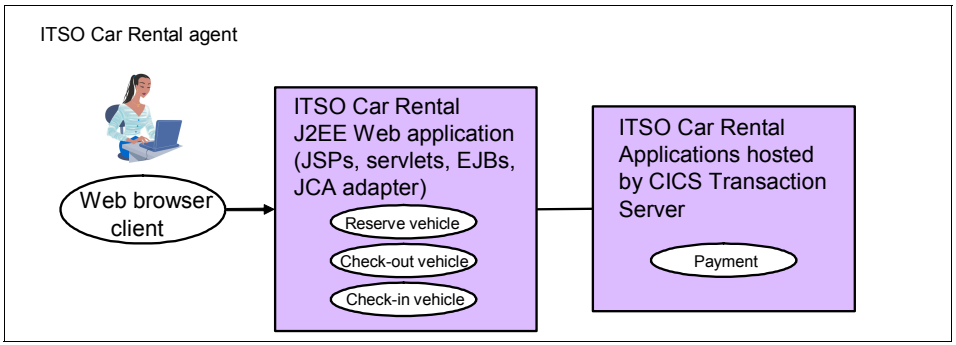


Figure 7-1 ITSO Car Rental Initial Context

7.1.2 Phases of SOA engagement

The ITSO Car Rental company has performed a series of workshops with the executive management team and major stakeholders to gain a better understanding of the company's business objectives and IT challenges. The ITSO views SOA as a strategic initiative they would like to pursue for both integration and application development at an enterprise level. The ITSO desires a transition to SOA that will leverage existing applications and infrastructure.

To facilitate the adoption of SOA, the company wishes to establish early proof points in phases.

- Phase 1: Expose and consume the Reserve Vehicle business process internally and externally as a service

  In the first phase, the ITSO will focus on the Reserve Vehicle business process. The first phase includes exposing existing Reserve Vehicle application functionality as services to other internal applications, as well as external Travel partner applications. This project will be the first in a series of evolutionary steps within a larger existing SOA transformation initiative.

  Note: The scope of this redbook example includes phase 1. This phase contains elements of the following realizations for the Service Creation scenario:

  - Directly expose existing applications as services
  - Consume services from third-party service providers

  Phases 2 and 3 are not implemented in the redbook example.

- Phase 2: Expose and consume the Check-out and Check-in Vehicle business processes

  This phase will involve a very similar set of tasks as phase 1.

- Phase 3: Expose the payment business process via a service component

  In phase 3, we will expose a COMMAREA COBOL based Payment application hosted by CICS Transaction Server (TS) indirectly by creating a middle-tier Web service to access CICS. The middle-tier Web service in this case is create top-down from a WSDL service definition. The middle-tier Web service wraps a session EJB that uses the CICS ECI resource adapter to communicate with the CICS Transaction Gateway (CTG) to access CICS TS. The CICS ECI resource adapter is a JCA adapter for WebSphere Application Server packaged with the CTG.

  This allows the Payment application to be shared by other Web services client applications such as the ITSO Car Sale application, thus eliminating the redundancy of payment functionality in each application.

Note: Phase 3 of the scenario is representative of the Indirect exposure of existing applications through service-component realization for the Service Creation scenario. For more information, refer to 3.3, “Indirectly expose existing applications with service components” on page 61.

7.1.3 Business objectives

This section outlines the key business drivers and IT challenges of the ITSO Car Rental organization for SOA adoption.

Business drivers

The key business drivers are as follows:

- *Enable multiple channels:* The ITSO wants to expand their presence in the marketplace and increase their market share by enabling external travel partners to make car rental reservations.
- *Lower total cost of ownership:* By making application functionality available as a service, other internal and external applications can consume the common services. This has the effect of lower development and maintenance costs by driving down redundancy.
- *Leverage existing core assets and skills:* Maximize the reuse of existing application assets, business processes, and skills of the organization.

IT challenges

The ITSO Car Rental organization faces many IT challenges. An assessment of the existing ITSO architecture revealed the following pain points:

- *Align business and IT:* The business processes need to be aligned with application architecture and development projects.
- *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.

Business context diagram

Figure 7-2 illustrates the proposed high-level business context for the ITSO Car Rental example scenario.

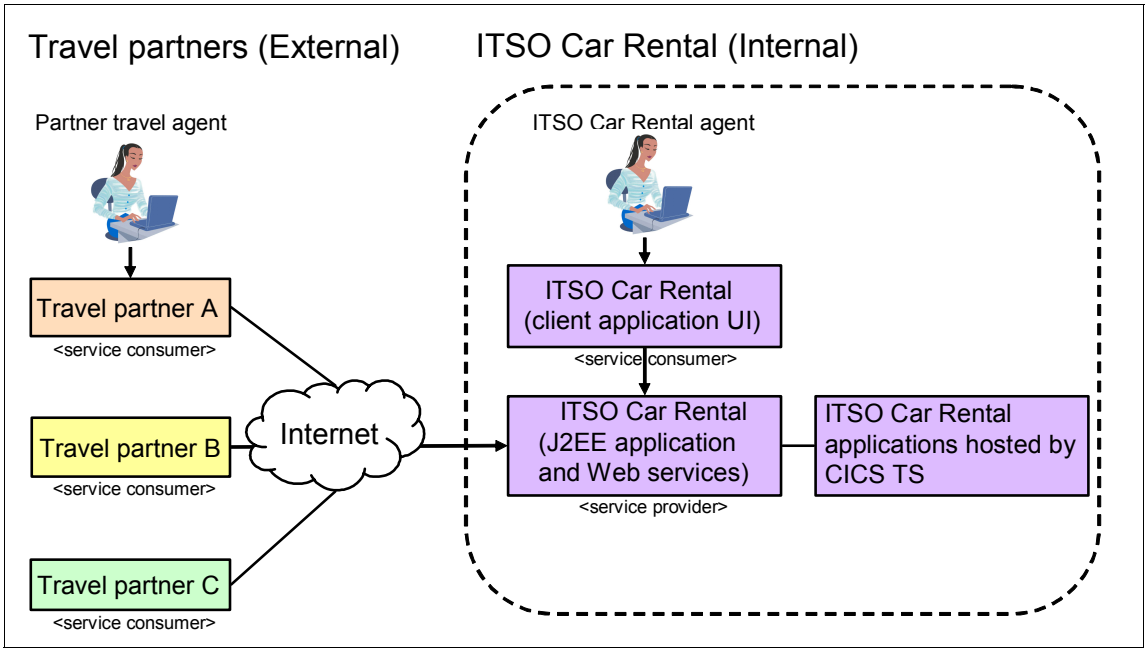


Figure 7-2 High level business context for the ITSO Car Rental example scenario

7.2 Requirements

This section includes functional and non-functional requirements for the ITSO Car Rental scenario. In addition, we have included a system context diagram to show the big picture of the solution architecture.

7.2.1 Functional requirements

This section highlights the functional requirements for phase one for the ITSO Car Rental working example. We have outlined the key functional requirements:

- FR1: Reuse existing application functionality as a Web service

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

- FR2: Create internal WS client application to consume services

  Create an internal Web services client application from the existing J2EE based application to consume the services of the ITSO Car Rental application.

- FR3: Expose selected Web services to external travel partners

  Expose selected Web services to external travel partners to make car rental reservations.

- FR4: Create template Web Services client application for external travel partners to consume services

  Create a template Web services client application to be used by external travel partners to consume the externally exposed services of the ITSO Car Rental application.

- FR5: Capture channel data the car reservation is made

  Modify the application to capture the channel on which the car reservation is made. This data will provide data to the business to measure the quantity and dollar amount of car rental reservations by external travel partners and internally.

Note: We will identify and design the services in more detail in 7.3, “Service identification and design” on page 180.

7.2.2 Nonfunctional requirements

The key IT stakeholders at the ITSO Car Rental company are primarily focused on the operational and security challenges they will face when implementing the first phase of the SOA transformation.

SOA management

We have identified the following key SOA management requirements related to service level agreements (SLAs):

- NR1: Monitor performance and availability of services with end-to-end view.
- NR2: Provide the ability to be proactive in troubleshooting.

Note: The monitoring and management for the example scenario uses IBM Tivoli Composite Application Manager for SOA. Refer to the following for more detailed information:

- 9.5, “Implement ITSO Monitor Server node” on page 330

  This section describes how to install and configure monitoring and management components in the runtime environment.

- Chapter 12, “Manage and monitor services with ITCAM for SOA” on page 405

  This chapter includes and overview of SOA management and ITCAM for SOA, and examples in support of the non-functional requirements.

Security

Phase one of the SOA transformation will address the following security requirements:

- Message level security

  Message level security, known as Web services security (WS-Security), is used to secure the SOAP message authentication, confidentiality and integrity. Our example will focus on *integrity* and *confidentiality*. We will not cover authentication because it is outside the scope of this book.

  - NR3: Integrity

    Ensure that information will not be changed, altered, or lost in an unauthorized or accidental manner.

  - NR4: Confidentiality

    Ensure that no unauthorized party or process can access or disclose the information.

- NR5: Transport level security

  Transport-level security uses secure sockets layer (SSL) to encrypt the transport layer of the communication between the Web services client application consumer and the Web services application provider. This is sometimes referred to as TLS/SSL.

Note: The security topics identified in the nonfunctional requirements for this example are further explained in Appendix D, “Web Services Security” on page 505. We reference other documents for implementation details.

7.2.3 System context diagram

Now that we have more specific technical requirements, we can continue to evolve the picture of the solution architecture in the system context diagram. The system context diagram in Figure 7-3 on page 180, shows the inputs and outputs of the system, as well as the context in which they are used.

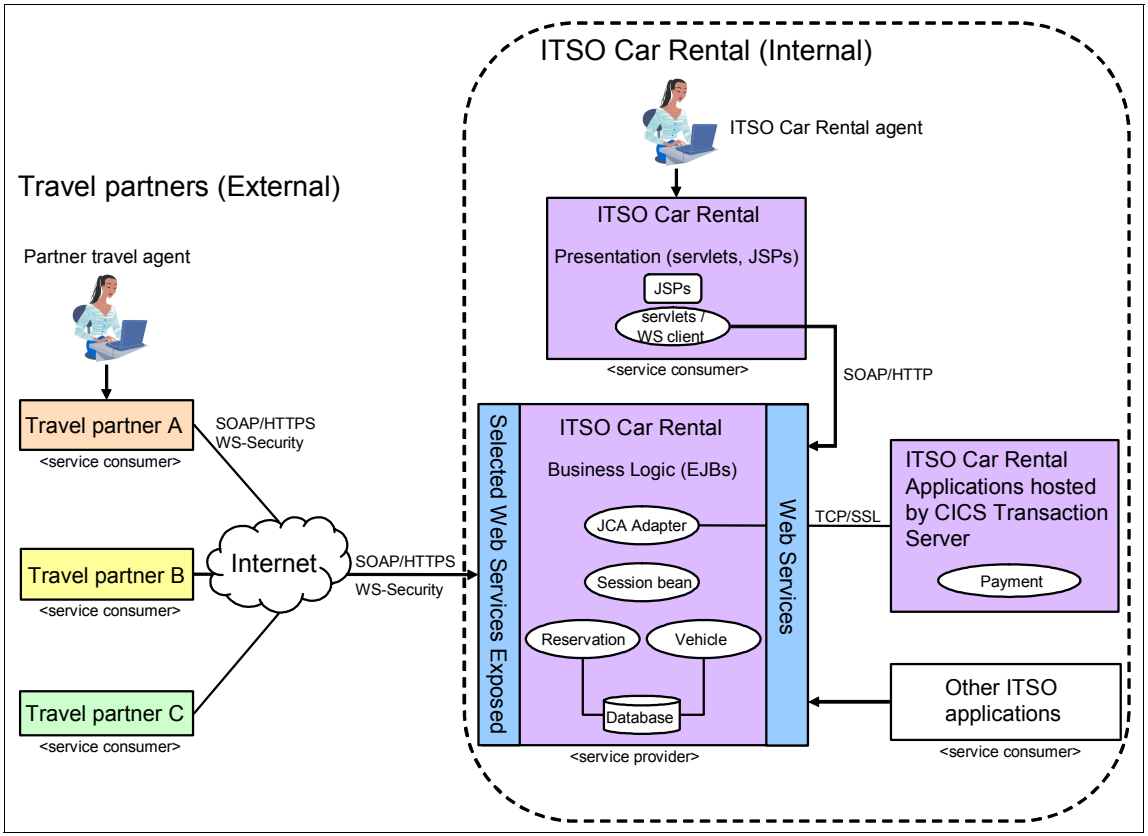


Figure 7-3 System context diagram for the ITSO Car Rental example scenario

7.3 Service identification and design

Like most companies engaged in a major business transformation, the ITSO wants a flexible business and IT architecture. This section includes techniques for service modeling, analysis, and design to bridge the gap between business objectives and IT to facilitate a flexible transformation and architecture.

In 4.4, “Service identification and modeling” on page 121, we introduced *Service-Oriented Modeling and Architecture* (SOMA). We will use the SOMA techniques and best practices for service modeling, analysis and design for the ITSO Car Rental scenario in addition to standard methodology.

Figure 7-4 depicts the key steps in the SOMA process we will follow for the ITSO Car Rental example scenario. This section is organized into the following three, key, high-level steps of SOMA:

- Identification
- Specification
- Realization

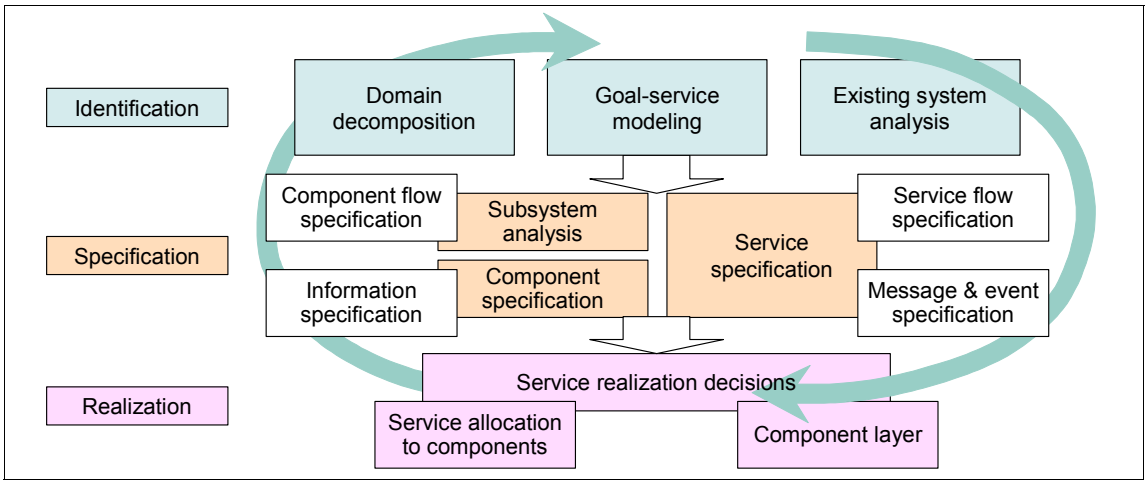


Figure 7-4 Modeling, analysis and design steps using SOMA techniques

As depicted in Figure 7-4, we initially work in a top-down approach starting with identification. Keep in mind that some tasks can be performed in parallel or serially based on dependencies, and specific requirements of the project. It is important to note that experience tells us that the service model is built *iteratively* and *incrementally*.

In many cases the customer scenario is based on an existing transformation. It is less common to have an SOA implementation that begins with a completely new project. Typically, migration to an SOA solution involves integration of existing systems by decomposing them into services, operations, business processes and rules. Next, we walk through the major steps to define our service model.

7.3.1 Identification

The identification step includes the following key sub steps and techniques to identify a list of candidate services for migration to an SOA:

- Service identification
  - Domain decomposition (top-down approach)
  - Goal-to-service model (middle-out approach)
  - Existing asset analysis (bottom-up approach)
- Service categorization
- Commonality and variability

Domain decomposition

In the domain decomposition, we work from the top-down by decomposing the business domain into major functional areas and subsystems. At the next level, we further decompose the functional areas into processes, subprocesses and high-level business use cases as depicted in Figure 7-5.

Customer engagement experience shows us that high-level business use cases, especially those that are well-suited for automation, are good candidates to be exposed as Web services and can provide an initial scope for the design.

For the ITSO Car Rental scenario, we focus on the decomposition of the *Rentals Management* domain and *Reservation* functional area.

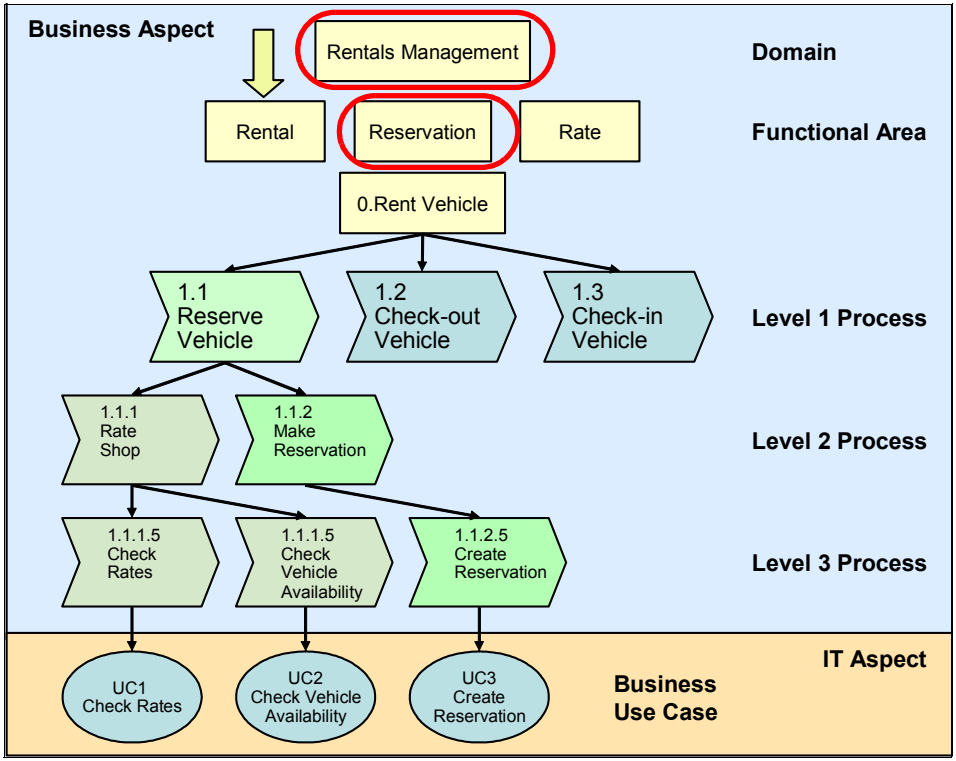


Figure 7-5 Domain Decomposition

Use case model

We have identified the following actors:

- Customer
- ITSO Car Rental agent
- Partner travel agent
- System

We have identified the following business use cases:

- UC1: Check Rates
- UC2: Check Vehicle Availability
- UC3: Create Reservation
- UC4: Check-out Vehicle (future project/phase)
- UC5: Check-in Vehicle (future project/phase)
- UC6: Process Payment (future project/phase)
- UC7: Log Reservation Activity
- UC8: View Reservation Activity

For the purposes of our scenario, we have excluded UC4, UC5 and UC6 from the implementation. Here, we focus on the processes used to reserve a vehicle.

Table 7-1 on page 184 shows the preliminary list of services that have been identified as part of the domain decomposition. Note, the remaining techniques within the identification step might identify additional services.

Table 7-1 Preliminary list of services identified in the domain decomposition

Preliminary list of services	Description
Check Rates	Search for vehicles and rates based on criteria including car class, location, pick-up and drop-off dates and times.
Check Vehicle Availability	Determine availability of a specific vehicle class
Create Reservation	Create and confirm a reservation of a vehicle for a customer
Check-out Vehicle	
Check-in Vehicle	
Locate Reservation	Locate a reservation based on given criteria
Modify Reservation	Change details of a reservation
Cancel Reservation	Cancel a reservation
Process Payment	Process payment for a reservation
Log Reservation Activity	Record reservation activity for all channels
View Reservation Activity	Provide a summary of reservation activity based on criteria

Table 7-1 on page 184 illustrates the top-down approach in terms of SOA layers as we begin to develop the service model.

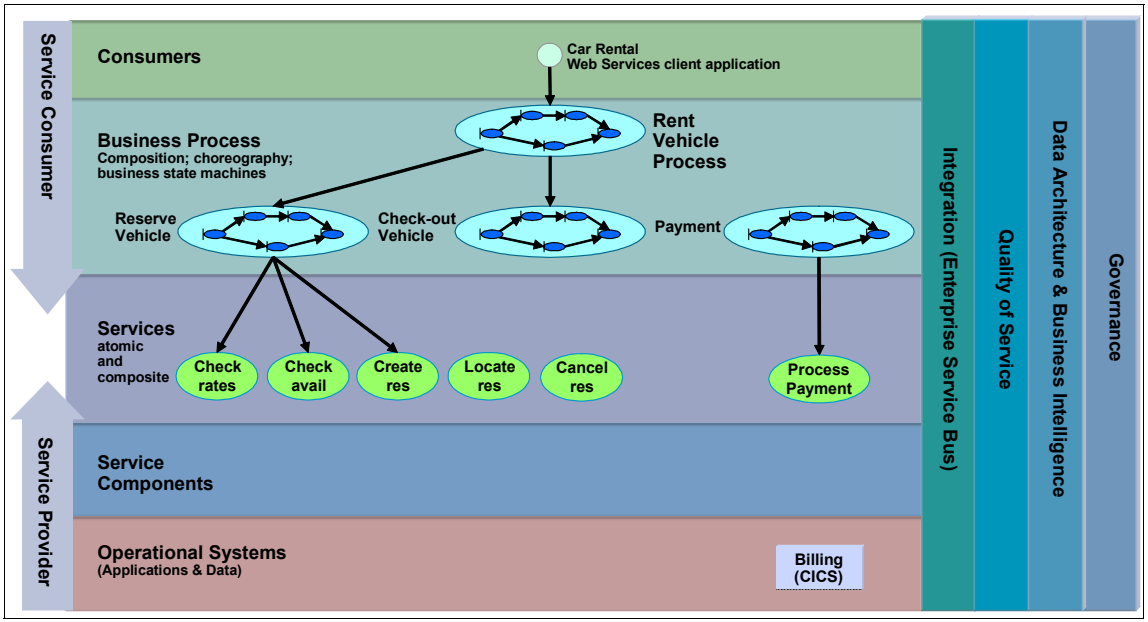


Figure 7-6 Top-down service identification candidates

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, we identify key performance indicators (KPIs) and metrics that will be used to assess success of the business and/or performance of the organization.

There are several possible notations that can be used to represent goal-to-service model analysis. Table 7-2 lists a goal-to-service matrix for the ITSO Car Rental scenario with the new candidate services identified. These services will provide information crucial to measuring the organization's progress towards its ultimate goal of increasing revenue.

Table 7-2 ITSO Car Rental scenario Goal-to-service model

Goal	KPIs	Metrics	Service
1.0 Increase Revenue	Revenue increase of 12% by Q4/2006		
1.1 Increase car rental sales			
1.2 Provide self-service rate shopping and reservation capability	Contribute revenue increase of 5		Check Rates Check Vehicle Availability Create Reservation
1.3 Track reservation activity	%	Record transaction s for new channels	Log Reservation Activity View Reservation Activity

Increase revenue by increasing sales

Beginning with the main goal of increasing revenue, we identify subgoals and associated services to realize this goal. The business plans on increasing by increasing the number of car rental sales. We can increase sales by expanding presence in the market place through multiple channels (internal, and external travel partners).

Provide self-service rate shopping and reservation capability

Currently, customers must go through an agent to reserve a vehicle either through a call center or visiting one of the ITSO Car Rental company locations. By exposing car rental services externally, Travel partners can create reservations in addition to existing channels. This subgoal will contribute to increasing sales and increase revenue.

Track reservation activity

To track the progress of the organization towards its main goal of increasing revenue, we need a method that can capture metrics on the reservation channel.

Existing asset analysis

The main objective of the existing asset analysis is to maximize the reuse of existing application transactions, modules from existing systems and packaged applications. When performing an existing asset analysis, we work bottom-up to identify candidate services.

With the assistance of subject matter experts and relevant documentation, we analyze the existing systems to identify candidate services. During this activity,

we also consider technical constraints related to existing systems that will assist in identification of potential risks. We begin to explore technical feasibility of service realization as early in the process as possible.

Next, we analyze each of the transactions in detail to identify the specific actions performed, information exchanged and discard services that are not considered to meet current and future business needs.

Figure 7-7 illustrates the bottom-up approach through identification of existing legacy transactions that we consider as candidate services.

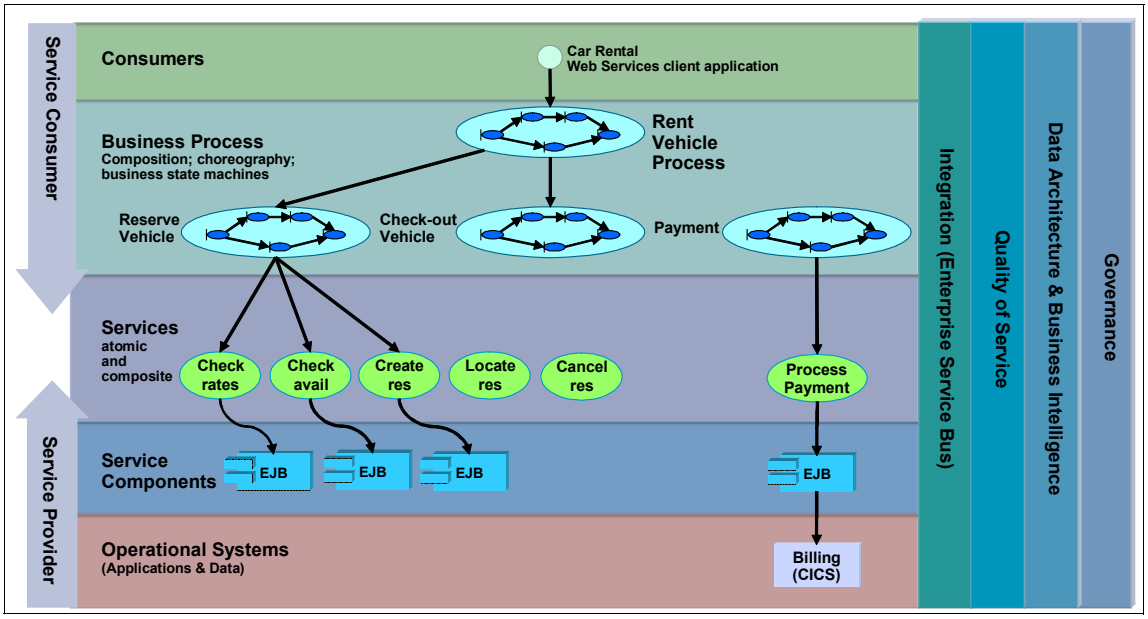


Figure 7-7 Existing transactions supporting business processes

Table 7-3 on page 187 lists the revised service portfolio for the ITSO Car Rental scenario with new services added as a result of the existing asset analysis and goal to service model.

Table 7-3 Revised service portfolio

Service	Description	Identification technique	Asset
Check Rates	Search for vehicles and rates based on criteria including car class, location, pick-up and drop-off dates and times.	Domain Decomposition / Existing Asset Analysis	Rate Shop API

Service	Description	Identification technique	Asset
Check Vehicle Availability	Determine availability of a specific vehicle class	Domain Decomposition / Existing Asset Analysis	Display Vehicle Availability API
Create Reservation	Create and confirm a reservation of a vehicle for a customer	Domain Decomposition / Existing Asset Analysis	Create Reservation
Display Manifest		Existing Asset Analysis	Display Manifest
Check-out Vehicle			
Check-in Vehicle			
Locate Reservation	Locate a reservation based on given criteria	Domain Decomposition / Existing Asset Analysis	
Modify Reservation	Change details of a reservation	Domain Decomposition / Existing Asset Analysis	
Cancel Reservation	Cancel a reservation	Domain Decomposition / Existing Asset Analysis	
Process Payment	Process payment for a reservation	Domain Decomposition	
Log Reservation Activity	Record reservation activity for all channels	Domain Decomposition / Goal-to-Service Modeling	
View Reservation Activity	Provide a summary of reservation activity based on criteria	Domain Decomposition / Goal-to-Service Modeling	

At this point, we have developed a list of services based on all three techniques; domain decomposition, goal-to-service modeling and existing asset analysis.

Service categorization

After the candidate services have been identified, we then categorize the services into a service hierarchy as illustrated in Table 7-4 on page 189. The categorization reflects the composite nature of services and to help determine composition and layering. Categorizations are typically identified based on functional area analysis during domain decomposition activities.

Table 7-4 Service categorization

Category	Service	Description
Reservation	Check Rates	Search for vehicles and rates based on criteria including car class, location, pick-up and drop-off dates and times.
	Check Vehicle Availability	Determine availability of a specific vehicle class
	Create Reservation	Create and confirm a reservation for a customer
Rental	Check-out	Check rental car out.
	Check-in	Check rental car in.
	Locate Reservation	Locate reservation.
	Modify Reservation	Modify the reservation details.
	Cancel Reservation	Cancel the reservation.
Payment	Process Payment	Process rental car payment.
Logging	Log Reservation Activity	Record reservation activity for all channels
	View Reservation Activity	Provides a summary of reservation activity based on criteria

Commonality and variability

As with traditional analysis and design techniques, it is also important to explore and model areas of commonalities and variability in structure, process, and rules. For example, through our analysis we found variations in characteristics and behavior for regular customers versus preferred customers. We can use class diagrams and sequence diagrams to model this behavior.

7.3.2 Specification

The objective of the specification step is to elaborate the service model. This section includes the following sub steps of specification for our scenario:

- Service exposure decisions
- Subsystem analysis
- Component specification

Service exposure decisions

Although we have identified many candidate services in our model, not all should be exposed. Key decisions must be made to develop a manageable set of services that the business wants to expose and that can be used later within compositions.

We have aligned our exposure criteria for our scenario to ensure they address the business drivers and requirements defined earlier in the chapter.

- Traceable
  - Can the service be traced back to goals and objectives of the organization?
- Stateless
  - Does the service require information or state between requests?
- Discoverable
  - Can the service be exposed externally to the enterprise?
  - Does the service have a well-defined interface and externalized service description?
- Reusable
  - Does the service serve the interests of other processes?
  - Can this service be reused to realize many higher-level business processes?

It is important to note that these criteria represent only a small example of characteristics to take into consideration when making service exposure decisions.

This step is key in managing the proliferation of services also known as the *Service Proliferation Syndrome*, whereby an organization has a tendency to replace existing APIs with Web services. Consequences of this syndrome can be significant in terms of service management and performance. In addition, it is highly probable that exposed services do not align with the goals and requirements of the business.

There are various possible notations to represent our decisions. We recorded our findings based on the analysis for this scenario in Table 7-5.

Table 7-5 Service exposure decisions

Service	Traceable	Stateless	Discoverable	Reusable	Expose	Comments
Check Rates	Pass	Pass	Pass	Pass	Yes	

Service	Traceable	Stateless	Discoverable	Reusable	Expose	Comments
Check Vehicle Availability	Pass	Pass	Pass	Pass	Yes	
Create Reservation	Pass	Pass	Pass	Pass	Yes	
Locate Reservation					Candidate	Review for exposure in future release
Modify Reservation					Candidate	Review for exposure in future release
Cancel Reservation					Candidate	Review for exposure in future release
Display Manifest	Pass	Pass	Fail	Fail	No	Isolated functionality for specific application
Check-out					Candidate	Review for exposure in future release
Check-in					Candidate	Review for exposure in future release
Process Payment					Candidate	Review for exposure in future release
Log Reservation Activity	Pass	Pass	Pass	Pass	Yes	
View Reservation Activity	Pass	Pass	Pass	Pass	Yes	Exposure within the enterprise exclusively

This example represents exposure decisions on several levels for our scenario:

- Services to be exposed within the scope of the current project
- Services that are candidates for exposure in future projects
- Services that will be exposed exclusively within the enterprise
- Services that will not be exposed

Subsystem analysis

Analysis resulting from the domain decomposition technique has already provided some insight into the various functional areas within the business domain. At this point, we can drill down further to identify subsystems as well as the interdependencies and flow between them. We typically make decisions to establish a one-to-one mapping between functional area and subsystem.

The following components are identified as part of the subsystem analysis:

- *Service* components are coarse-grained components that provide functionality required by the subsystem. High-level business use cases such as Reserve Vehicle are good candidates for exposure.
- *Functional* components are derived from the functional requirements and traditional object-oriented application development techniques to decompose service components into functional components that, ultimately, will be responsible for fulfilling the responsibilities of the subsystem. For example, the reservation service component delegates to the Vehicle functional component to determine vehicle availability.
- *Technical* components come from our use of nonfunctional requirements such as security, transaction management, systems management and messaging to identify technical components. For example, EJB services are identified within the ITSO Car Rental scenario as a technical component.

Figure 7-8 illustrates the reservation subsystem showing service, functional, technical components and their relationships.

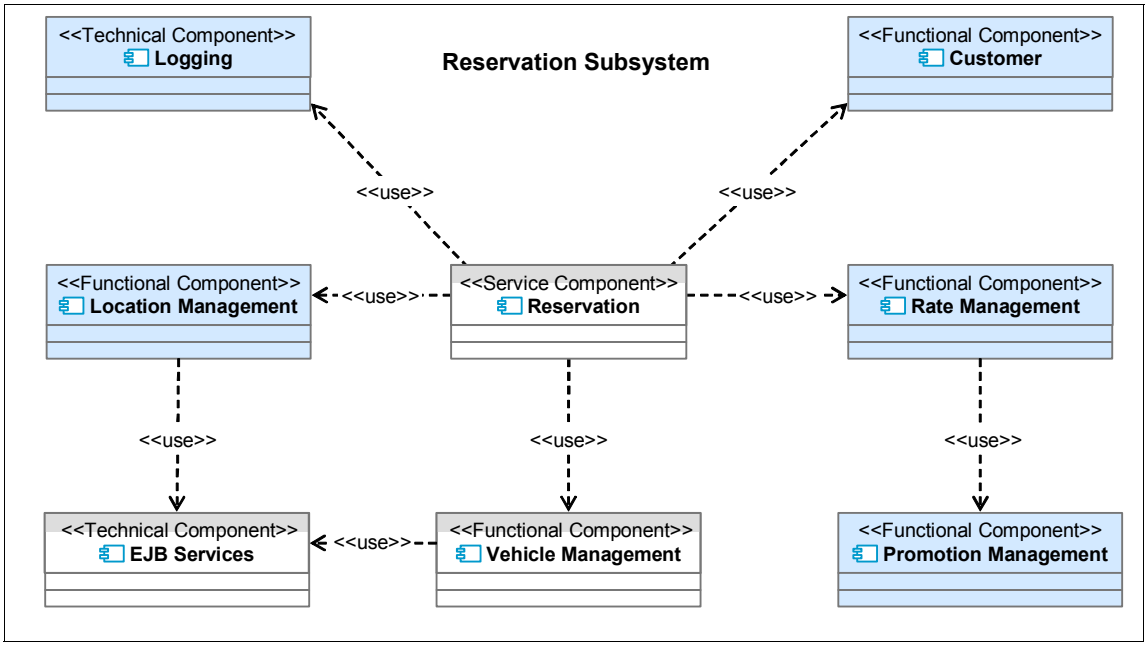


Figure 7-8 Reservation Subsystem component class diagram

For the purposes of the ITSO Car Rental scenario, the Location, Customer, Rate and Promotion Management functional components have not been implemented and are for illustration purposes only.

At this point we have completed the subsystem analysis and have identified the following coarse-grained components to be implemented as services for the ITSO Car Rental system in phase one:

- Reservation service: Provides the capability to rate shop, check vehicle availability and reserve vehicles.
- Logging service: Provides the capability to logging and retrieval of events related to vehicle rental.

Component specification

Now that we have identified the service, functional, and technical components we can specify more details about these components including component flow, relationships between components, attributes, events and messages.

We use sequence or activity diagrams to illustrate component flow and class diagrams to show the relationships between service, functional and technical components. In addition, we use variation oriented design techniques to identify commonalities and externalize variations through use of design patterns.

Figure 7-9 illustrates the internal component flow using a component diagram. The shaded boxes (Vehicle, Reservation) in Figure 7-9 have not been fully implemented for the ITSO Car Rental scenario.

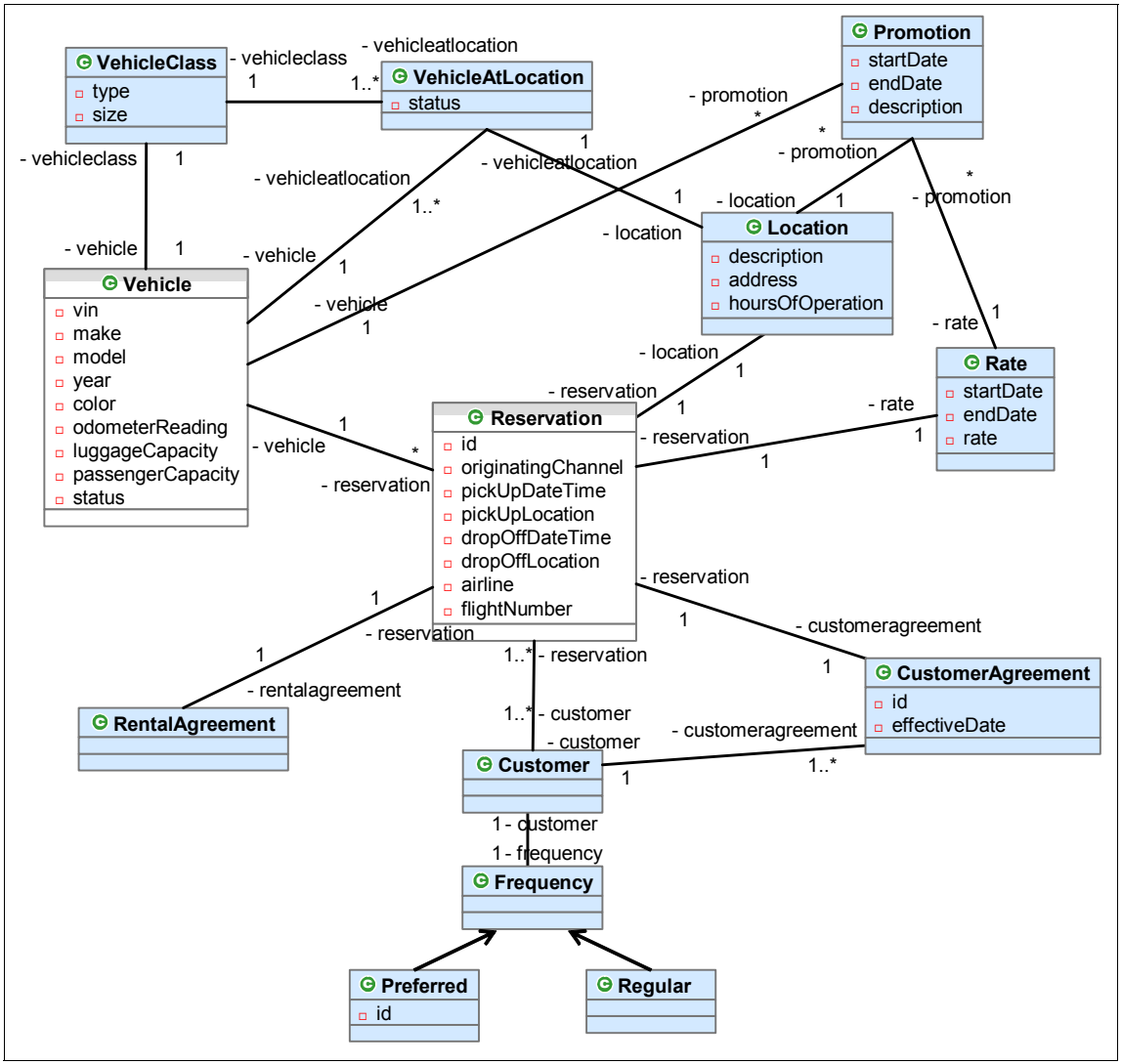


Figure 7-9 Reservation service component class diagram

7.3.3 Realization

In the realization step, we consider architectural and realization decisions that must be made to implement the required service functionality by taking into consideration qualities of service as well as business and IT constraints.

In this step, we make the decision as to which existing system module will be used to realize a given service and which services will be built from the ground up. Other realization decisions for services other than business functionality include, security, management, and monitoring of services.

Activities within this step are typically performed concurrently with other activities throughout the process. Specifically, we allocate services to components, allocate components to layers within the application architecture, and explore technical feasibility to realize services identified through the process.

Service allocation

In service allocation, we specify which component will provide the implementation and management of each service. Once again, it is important to demonstrate the traceability back from service components to business goals. This step also provides an additional level of validation to ensure that all services have been identified. In addition, we allocate components to layers within the application architecture.

At this point, we have identified and defined components necessary to provide the services we require, allocated services to components and developed the component specifications.

7.4 Solution architecture

This section demonstrates how to use the process defined in Chapter 5, “Process for applying SOA scenarios and patterns” on page 131 to accelerate the design of the solution architecture for the ITSO Car Rental working example scenario.

7.4.1 Fit gap analysis

In 7.2.1, “Functional requirements” on page 177, we identified the functional requirements for the ITSO Car Rental example. As part of the defined process, we now perform a fit gap analysis of the ITSO Car Rental functional requirements with the generic use cases defined in 5.2, “Generic use cases for the SOA scenarios” on page 136.

Table 7-6 provides a summary of the ITSO Car Rental functional requirements that map to the generic use cases. Likely, there will be functional requirements that are not mapped to a generic use case specific to your business scenario. For example, although the no-functional requirement FR5: Capture channel data the car reservation is made, is important to our alignment of business objectives and IT, there is no corresponding generic use case. We will still allow for use cases not covered in the fit gap analysis within the solution architecture.

Table 7-6 ITSO Car Rental - fit gap analysis

ITSO Car Rental example functional requirements	**Generic use cases**
FR1: Reuse existing application functionality as a Web service	► "U1: Reuse existing or create new application logic as a service within the enterprise" on page 136
FR2: Create internal WS client application to consume services	► "U3: Point-to-point integration of enterprise apps using services" on page 137 ► "U5: Allow users to invoke services simply" on page 138
FR3: Expose selected Web services to external travel partners	► "U2: Reuse existing or create new, application logic as a service beyond the enterprise" on page 136
FR4: Create template Web Services client application for external travel partners to consume services	► "U4: Point-to-point integration of intra-enterprise applications using services" on page 137 ► "U5: Allow users to invoke services simply" on page 138
FR5: Capture channel data the car reservation is made	**Note:** A generic use case for this functional requirement does not exist.

7.4.2 Select the SOA scenario

At this stage in the process, we have identified the generic use cases that apply to the ITSO Car Rental example. Now we use the generic use cases as criteria to select the appropriate SOA scenario containing software capable of fulfilling the requirements. This task is accomplished by using the SOA scenario selection table (Table 5-1 on page 142).

Table 7-7 shows a subset of the generic use cases found in Table 5-1 on page 142, that fulfill the functional requirements of the ITSO Car Rental example scenario (Service Creation).

Table 7-7 SOA scenario selection criteria identifying the Service Creation scenario

Generic use cases selection criteria	SOA scenarios				
	Service Creation	Service Connectivity	Interaction and Collaboration Services	Business Process Management	Information as a Service
U1: Reuse existing or create new application logic as a service within the enterprise	X				
U2: Reuse existing or create new, application logic as a service beyond the enterprise	X				
U3: Point-to-point integration of enterprise apps using services	X				
U4: Point-to-point integration of intra-enterprise applications using services	X				
U5: Allow users to invoke services simply	X				

7.4.3 Reuse patterns assets to accelerate solution architecture

This section demonstrates how to apply the reusable assets found in the Patterns for e-business to accelerate the design of the solution architecture.

Select the Business and Application patterns

This section describe how to identify the Business and Integration patterns, and corresponding Application patterns related to the Service Creation scenario.

Application patterns are used to provide a conceptual layout of the application components and data within a Business pattern or Integration pattern interact.

Use Section 5.4.3, “Patterns for the SOA scenarios” on page 149 to identify the appropriate reference material for the patterns details. In our example, we have identified the Service Creation scenario in Table 5-5 on page 150, as a result, we go to Chapter 6, “Patterns for the Service Creation scenario” on page 155 to review the patterns details related to the Service Creation scenario.

Table 7-8 on page 198 provides a summary of the Business and Integration patterns, and corresponding Application patterns that map to the generic use cases representative of the Service Creation scenario.

Note: The Application Integration::Direct Connection application pattern has two variations:

- Call Connection variation: This variation is used when an immediate response is required from target application such as request and response (for example, Web service is invoked to retrieve rate information). The ITSO Car Rental example only requires this variation.
- Message Connection variation: This variation is used when an immediate response from the target application is not required.

Table 7-8 Summary of patterns for the Service Creation scenario

		Generic use cases				
Business and Integration pattern	**Application pattern**	**U1: Expose existing or new application logic as a service within the enterprise**	**U2: Expose existing or new application logic as a service beyond the enterprise**	**U3: Point-to-point integration of enterprise applications using services**	**U4: Point-to-point integration of intra-enterprise applications using services**	**U5: Allow users to invoke services**
Self Service	Directly Integrated Single Channel	X		X		X
Application Integration	Direct Connection: - Call Connection variation	X		X		
Extended Enterprise	Exposed Direct Connection: - Exposed Call Connection variation		X		X	

As seen in Table 7-8, there are three Application patterns that apply to the generic use cases representative of ITSO Car Rental example.

Figure 7-10 depicts a Composite Application pattern for the ITSO Car Rental example, which includes the following Application patterns:

- Directly Integrated Single Channel application pattern

  For details refer to “Directly Integrated Single Channel” on page 157.

- Direct Connection application pattern::Call Connection variation

  For details refer to “Direct Connection” on page 161.

  Note: Only the Call Connection variation is required for the example.

- Exposed Direct Connection application pattern::Exposed Call Connection variation

  For details refer to “Exposed Direct Connection” on page 164.

  Note: Only the Exposed Call Connection variation is required for the example.

  The presentation layer in the partner zone is depicted with a dotted line to show this is a special case adoption of the patterns for our scenario.

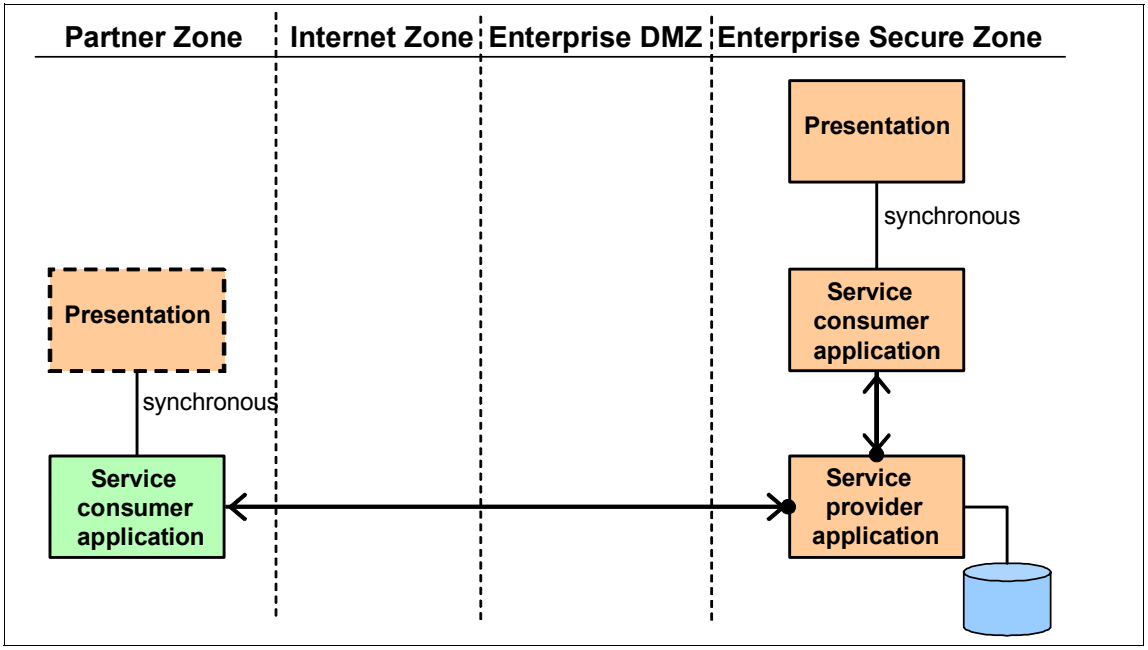


Figure 7-10 Composite application pattern for the ITSO Car Rental example

Runtime patterns

In the previous section we identified the Application patterns that apply to the Service Creation scenario. Each of the Application patterns has a corresponding Runtime pattern, which is used to define the logical middleware structure and interaction between nodes to support the Application pattern.

The Runtime pattern depicted in Figure 7-11 is a composite of the following three Runtime patterns identified for the ITSO Car Rental example.

- Directly Integrated Single Channel runtime pattern

  For details refer to “Directly Integrated Single Channel” on page 157.

- Direct Connection runtime pattern::Call Connection variation

  For details refer to “Direct Connection” on page 161.

- Exposed Direct Connection runtime pattern::Expose Call Connection variation

  For details refer to “Exposed Direct Connection” on page 164.

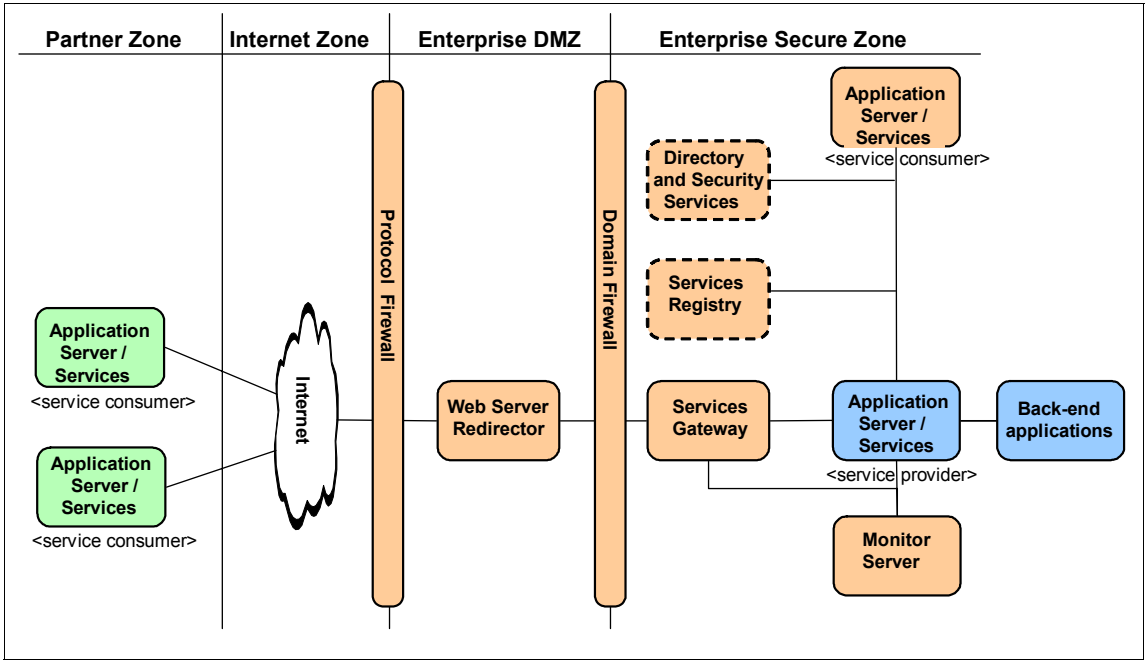


Figure 7-11 Composite runtime pattern for the ITSO Car Rental example

The Directory and Security Services, and Services Registry nodes depicted in Figure 7-11 on page 200 are optional for the scenario. We did not implement the

functionality of these nodes for the ITSO Car Rental example. In a production environment, they typically would be implemented.

Product mapping

Figure 7-12 depicts the Product mapping for the Composite Runtime pattern described in the previous section. The Product mapping is used to define the software products used to instantiate the functionality of the Runtime pattern node or component.

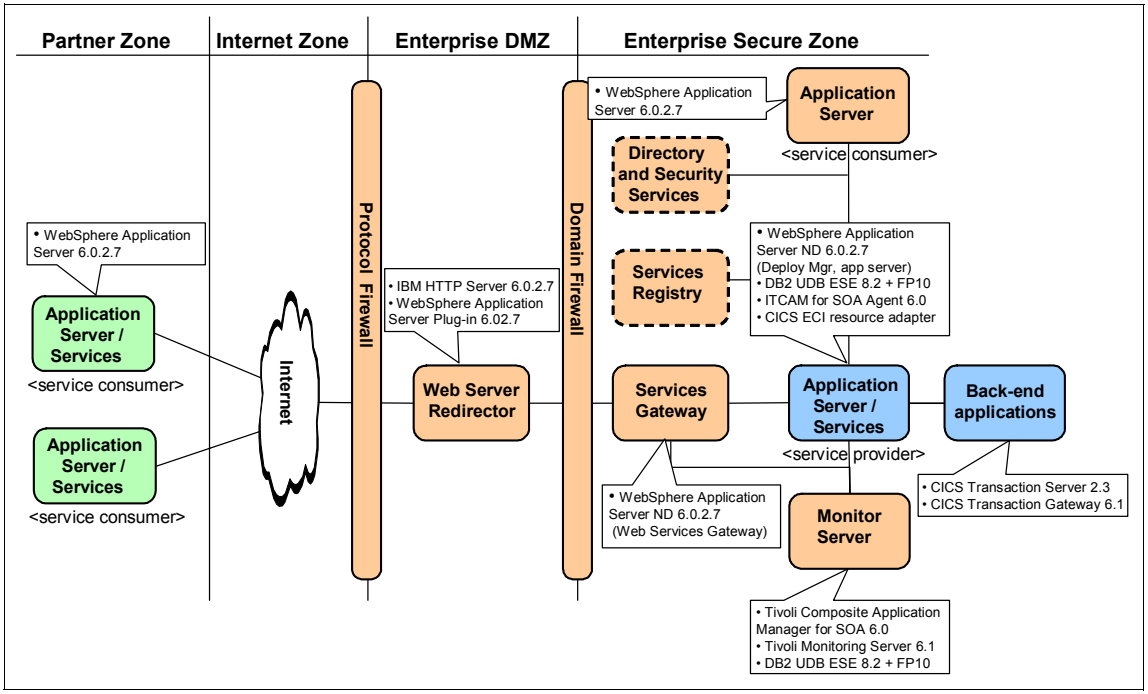


Figure 7-12 Composite product mapping for the ITSO Car Rental example

7.5 Where to find implementation details

This section highlights where to go next to find the implementation details for ITSO Car Rental working example, which is an instance of the Service Creation scenario. The ITSO Car Rental example implementation chapters illustrate how to assemble, deploy, and manage the implementation.

- Assemble
  - Chapter 8, “Assemble an application with Rational Application Developer” on page 203

- Deploy
  - Chapter 9, “Implement the runtime environment” on page 281
  - Chapter 10, “Deploy application to WebSphere Application Server” on page 351
  - Chapter 11, “Walk through the application” on page 381
- Manage
  - Chapter 12, “Manage and monitor services with ITCAM for SOA” on page 405

8

Assemble an application with Rational Application Developer

This chapter describes how to assemble the ITSO working example application within Rational Application Developer. The tasks found in this chapter are representative to the assemble tasks of the Service Creation scenario.

The chapter is organized into the following sections:

- Web services support in Rational Application Developer
- Development environment installation
- Prepare for sample application development
- Application development overview
- Create Web Services for ITSO Car Rental application
- Create the internal Web Service client application
- Create the external Web Service client application
- Export the EAR files for deployment
- Develop a Java client application to generate traffic
- Where to find more information

8.1 Web services support in Rational Application Developer

This section provides a summary of the key features of IBM Rational Application Developer (RAD) specific to Web Services and SOA. Throughout our sample, we demonstrate how to easy it is to develop Web Services from an existing J2EE application with Rational Application Developer.

Note: In our working example, we will demonstrate how to:

- Create a Web Service from an existing EJB

  Refer to 8.5.3, “Create the Web Services” on page 247.

- Create a Java client proxy from WSD

  Refer to 8.6, “Create the internal Web Service client application” on page 254, and 8.7, “Create the external Web Service client application” on page 267.

- Run and test the application within the WebSphere Application Server V6.0 Test Environment

  Refer to 8.3.1, “Create a test server within Rational Application Developer” on page 217.

8.1.1 Web services tooling

This section gives an overview of the Web services-specific functions in Rational Application Developer, which provides tools to assist with the following aspects for Web services development:

- *Create service provider:* Use the Application Developer tooling to create, deploy, test, and publish Web services bottom-up from existing Java beans, enterprise beans, DADX files, and URLs, and top-down from WSDL. When generating a Web service, the wizards support the automatic generation of additional artifacts, such as a JavaBean proxy to easily access the Web service, and a test client.

  To create a Web service, either graphical wizards or command-line tools can be used.

- *Create service consumer:* Use the Web services client tools (GUI wizard or command line) to create a client for any Web service. Only the WSDL file is needed to create a Web service client.

- *Secure:* The Web Service wizards and deployment descriptor editors assist you with configuring Web services security (WS-Security) for the WebSphere Application Server environment.
- *Run*: Run Web services provider and consumer components in WebSphere Application Server or Tomcat test environments. The deployment and administration for the WebSphere test environment is integrated in Application Developer.
- *Test:* Web services can be tested, running locally or remotely. For local tests, the WebSphere test environment can be used. The WebSphere test environment contains a complete WebSphere Application Server runtime environment. Rational Application Developer provides different functions to test Web services.
- *Discover*: Browse Universal Description, Discovery, and Integration registries (UDDI) or Web Services Inspection Language (WSIL) sites to find Web services for integration. The IBM Web Services Explorer provides the all necessary functions to discover a Web service.
- *Publish*: Publish Web services to a UDDI V2 or V3 Business Registry, using the Web Services Explorer.
- *Build skeletons*: Generate JavaBean and EJB skeletons from WSDL files. This can be helpful during the development and test phase of a project. For example, when the service is defined (WSDL), but not running at the service provider site, and the client must be tested, a test service provider can be created to emulate the provider.
- *Validate*: Use the WSDL and DADX validators to check for structural and semantic problems in these types of files. This feature is useful when receiving a service WSDL file from a service provider to check that the files are valid.
- *Compliance*: Different WS-I profile compliance tests and levels can be defined for the Web services development environment. Application Developer can check compliance for the Simple SOAP Basic 1.0 and the Attachment Profile 1.0.

  When creating or changing Web services, the WS-I compliance tester analyzes the service, and depending on the configuration, ignore, suggest, or require profile compliance. This can be defined in the Web services preferences, as described in “Web services configuration settings” on page 207.
- *WSDL support*: Application Developer provides wizards and functions to easily work with WSDL files:
  - Use the graphical editor to create a WSDL file from a template and to add WSDL elements (service, port, port types, messages).

  - Create WSDL documentation. This creates HTML documentation for the WSDL file, similar to a JavaDoc document.
  - Validate WSDL file for WS-I compliance.

- *Web services-specific navigation:* Application Developer now organizes Web services together in the Project Explorer in a Web Services group. This makes is easier to find and work with Web services.

8.1.2 Web services runtime environments

Rational Application Developer supports three Web service provider runtime environments:

- *IBM WebSphere runtime environment:* This is the recommended runtime environment for production use. Only the WebSphere runtime environment is full supported by IBM. It includes specialized serializers and deserializers for complex objects, JSR 109 support for enterprise Web services (EJBs), and SOAP over JMS support.
- *IBM SOAP runtime environment:* This was the only supported runtime environment in previous releases of WebSphere Studio (Version 5.0 and earlier). It should only be used for backward compatibility, and it supports Apache SOAP 2.3.

  Currently, DB2 Web services from SQL statements (DADX files) still require the SOAP runtime.
- *Apache Axis 1.0 runtime environment:* This is the third version of the Apache SOAP implementation. Apache Axis evolved from the Apache SOAP implementation (which began as IBM SOAP4J).

  The Apache Axis runtime is not suggested for WebSphere production environments, but can be used for Apache Tomcat servers.

Table 8-1 shows the supported runtime and server configurations.

Table 8-1 Supported runtime and server configurations

SOAP runtime	Server configuration
IBM WebSphere	WebSphere Application Server V5.0.2, V5.1, V6.0
IBM SOAP	Apache Tomcat V3.2, V4.0, V4.1 WebSphere Application Server V5.0, V5.1
Apache Axis 1.0	Apache Tomcat V4.0, V4.1, V5.0 WebSphere Application Server V5.0, V5.1, V6.0

8.1.3 Web services configuration settings

In this section, we explain the Web service-specific configuration settings in the Application Developer workbench. These options influence the behavior and generated artifacts of the Web service tooling.

Note: To develop Web services with Rational Application Developer, the Web services development capabilities must be enabled. Enabling this capability activates the required functions to develop Web services and the Web services options section in the workbench preferences page.

To enable the Web service capabilities, open the Application Developer preferences page (Figure 8-1):

1. Select **Windows** → **Preferences**.
2. Expand **Workbench** and select **Capabilities**.
3. In the right pane, expand **Web Services Developer** and select both **Component Test for Web Services** and **Web Services Development**.
4. Click **OK** to activate the changes and to close the preferences window.

Note: The Web services preferences section opens when reopening the preferences page, after the Web services capabilities have been enabled.

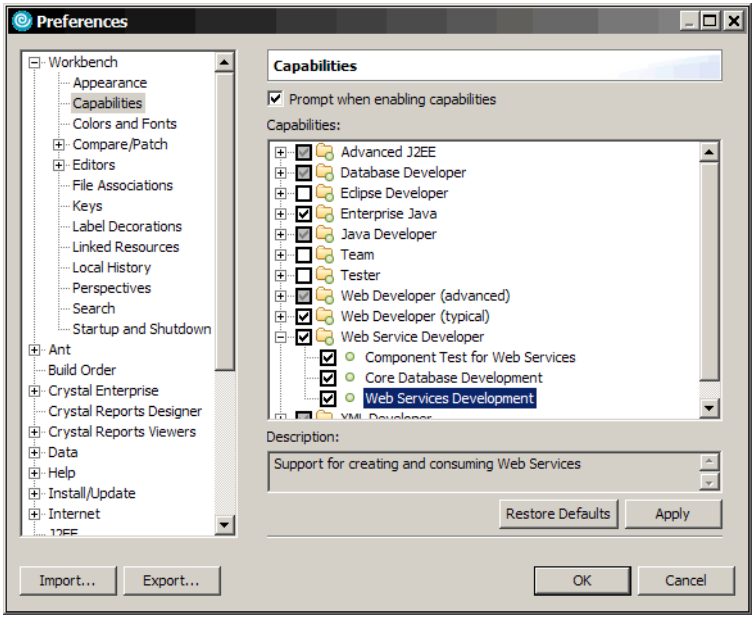


Figure 8-1 Enable Web services development capabilities

8.1.4 Web services preferences

To change the Web services preferences, select **Window** → **Preferences** and expand **Web Services** (Figure 8-2).

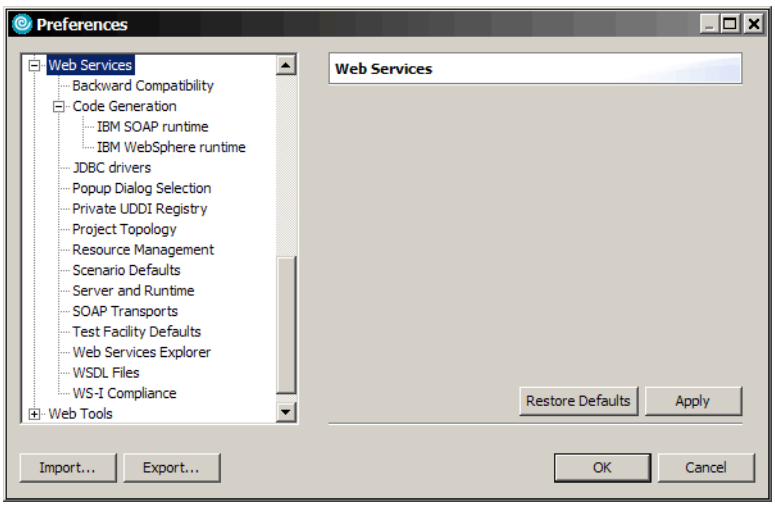


Figure 8-2 Web services preferences

We now provide an overview of the Web services preferences:

- *Backward Compatibility*: This option only applies when using the IBM SOAP runtime environment. Enabling this option will use WebSphere Application Server Version 4 mapping styles.
- *Code Generation*: Change the behavior of the code generation wizards for the SOAP and the WebSphere runtime.
  - When you enable **Disable data binding and use SOAPElement** for the WebSphere runtime, the `noDataBinding` option will be used for code generation. This is needed when a consistent mapping mechanism is required for an entire service endpoint, for example, when JAX-B (Java API for XML Binding) or Service Data Objects (SDO) have to be used for the binding.
  - When you enable **Do not overwrite loadable Java classes,** no duplicate classes are generated when the original class is available, for example, in a utility JAR file.
- JDBC™ drivers: This setting is only used for DADX validation in the Application Developer Linux version. It must be defined to provide the location to the correct JDBC driver JAR file.
- *Popup Dialog Selection:* Change the behavior of the pop-up dialog boxes for the Web Service wizards. These preferences can be used to disable some of the dialog steps in the wizard, for example, publishing to UDDI.
- *Private UDDI Registry:* Specify the delimiter for the used UDDI category data column and string. This is helpful when working with a private UDDI registry where self-defined tags are used.
- *Project Topology:* Specify whether the Web service server and client are generated into separate enterprise applications. This setting is highly recommended (select **Generate Web service and Web service client in different EAR projects**). Having the service and the client in the same project can cause conflicts and makes the production assembly and deployment more complex. The default sequence of client types (Web, Java, EJB, Application Client) can also be changed on this page.
- *Resource Management:* Specify the file and folder overwrite and creation behavior when creating a Web service. For example, select **Overwrite files without warning** as the default (if you regenerate the same service multiple times).
- *Scenario Defaults:* Specify initial selections for the Web Service wizard. For example, select **Start Web service in Web project, Generate a proxy,** and **Test the Web service.** These options will be preselected when running the wizard.

- *Server and Runtime:* Specify the default server (`WebSphere v6.0 Server`), the Web service runtime (`IBM WebSphere`), and the J2EE Version (`1.4`).
- *SOAP Transports:* Specify the default transport (HTTP or JMS).
- *Test Facility Defaults:* Specify the default settings for the test facility. We suggest to that you clear the option **Launch the sample when generated**, and leave the `Web service sample JSPs` at the top.
- *Web Services Explorer:* When using the IBM SOAP runtime and testing services that exchange array types, **Ignore schema for SOAP arrays** should be enabled on this page.
- *WSDL Files:* Specify the default target name space for WSDL files created with the WSDL editor.
- *WS-I Compliance***:** Specify the Application Developer behavior in respect to WS-I compliance checks. Application Developer can validate the compliance for WS-I Basic Profile 1.0 and WS-I Attachment Profile 1.0. Recommended selections are **Suggest compliance** for Basic Profile (WS-I SSBP compliance level) and **Ignore compliance** for Attachment Profile (WS-I AP compliance level).

Be sure to click **Apply** when making changes and close the dialog box by clicking **OK**.

8.2 Development environment installation

This section describes how we setup our development environment used to develop the sample application for this book.

8.2.1 Development environment planning

This section defines the software and hardware used within the ITSO development environment to develop the sample application.

Software used within the ITSO development environment

We used the software listed in Table 8-2 to implement the ITSO development environment.

Table 8-2 Developer node

Software	Version
Microsoft XP	Professional + Service Pack 2 + Critical Fixes

Software	Version
IBM Rational Application Developer * Integrated Development Environment * WebSphere Application Server V6 Test Environment	6.0.1.1 **Note:** 6.0 + 6.0.1 Fixpack + 6.0.1.1 Interim Fix + WAS Test Environment Update 6.0.2.5
IBM DB2 Universal Database™ Enterprise Server Edition	8.1.10.812 **Note:** 8.2 + Fixpack 10

Hardware used within the ITSO development environment

We used the following hardware to develop the ITSO sample application:

- IBM ThinkCentre M50 (8187)
  - Intel® Pentium® 4, 3.0GHz
  - 2.5GB RAM
  - 80GB IDE HDD

Note: We provide the sample application source code within a project interchange file that can be easily imported into Rational Application Developer or Rational Software Architect.

For more detailed information, refer to Appendix E, “Additional material” on page 515.

8.2.2 Rational Application Developer V6.0 installation

The IBM Rational Application Developer V6.0 product includes detailed installation documentation. This section highlights the installation issues we encountered while writing this book, as well as the components we installed.

Installation considerations

Prior to installing IBM Rational Application Developer V6.0, we had to be aware of the following installation considerations:

- UNC network shares: Do not install Rational Application Developer from a UNC network share (for example, \\server\shareA). Instead, map the network drive to a drive letter (for example, net use x: \\server\shareA) so that the Rational Application Developer installer works properly.
- Standardize an installation path for team development: Standardize the Rational Application Developer installation path for your development team. We found that many files within the projects have absolute paths based on

the installation path; thus when you import projects from a team repository such as CVS or ClearCase® you will get many errors.

- Installer window in foreground or background: After clicking IBM Rational Application Developer V6.0, sometimes the welcome screen does not display in the foreground. Simply select the new window from the task list to continue.

Rational Application Developer installation

While writing this book, we installed IBM Rational Application Developer V6.0 as follows:

1. Start the Rational Application Developer Installer by running `launchpad.exe` from CD 1.
2. The IBM Rational Application Developer V6.0 components have separate installations from the main Launchpad Base page, as seen in Figure 8-3 on page 212. When the Launchpad opens, click **Install IBM Rational Application Developer V6.0**. It is the only component required for our examples and includes the WebSphere Application Server V6.0 test environment.

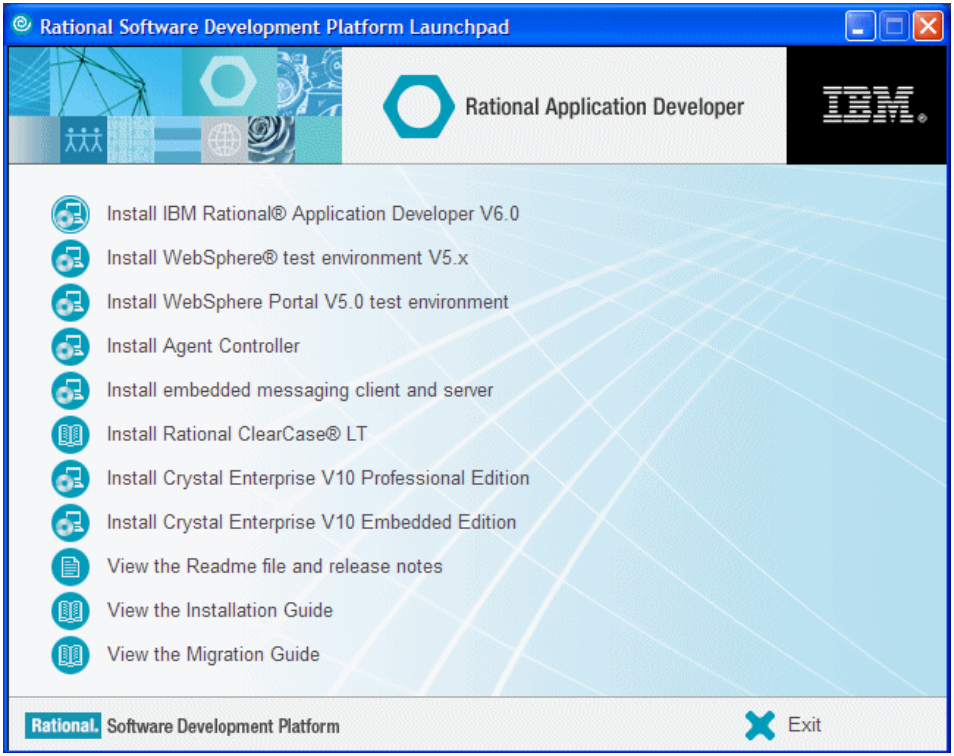


Figure 8-3 IBM Rational Application Developer V6.0 installation components

3. When the Welcome window opens, click **Next**.
4. When the License Agreement window opens, review the terms and, if in agreement, select **I accept the terms of the license agreement**. Click **Next**.
5. When the Install Directory window opens, we accepted the default C:\Program Files\IBM\Rational\SDP\6.0 and clicked **Next**.

> **Important:** It is very important that you understand the implication of changing the default installation path. For example, if you plan on team development, we strongly recommend that all developers use the same installation path (such as default); otherwise you will run into problems.
>
> We found that many files within the projects have absolute paths based on the installation path; thus when you import projects from a team repository such as CVS or ClearCase you will get many errors.

6. When the Select Features window opens, select the appropriate features for your environment. For example, we selected the features displayed in Figure 8-4 for this book (**Integrated Development Environment** and **IBM WebSphere Application Server V6.0 Test Environment)**.

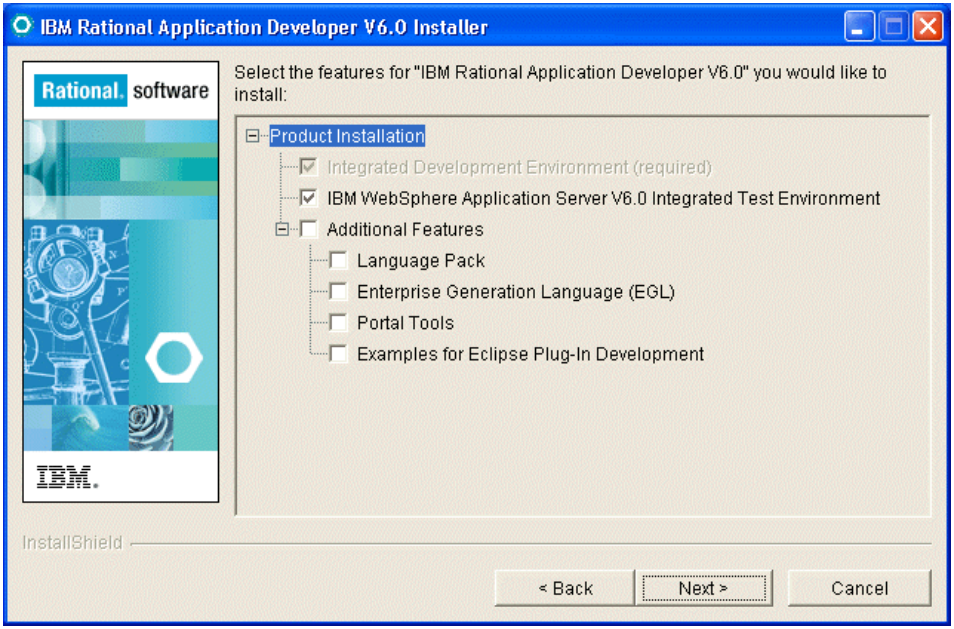


Figure 8-4 Install IBM Rational Application Developer V6.0 Features

7. When the installation summary window opens, review your selections and click **Next** to begin copying files as seen in Figure 8-5 on page 214.

> **Note:** The selected features require approximately 2.34 GB of disk space.

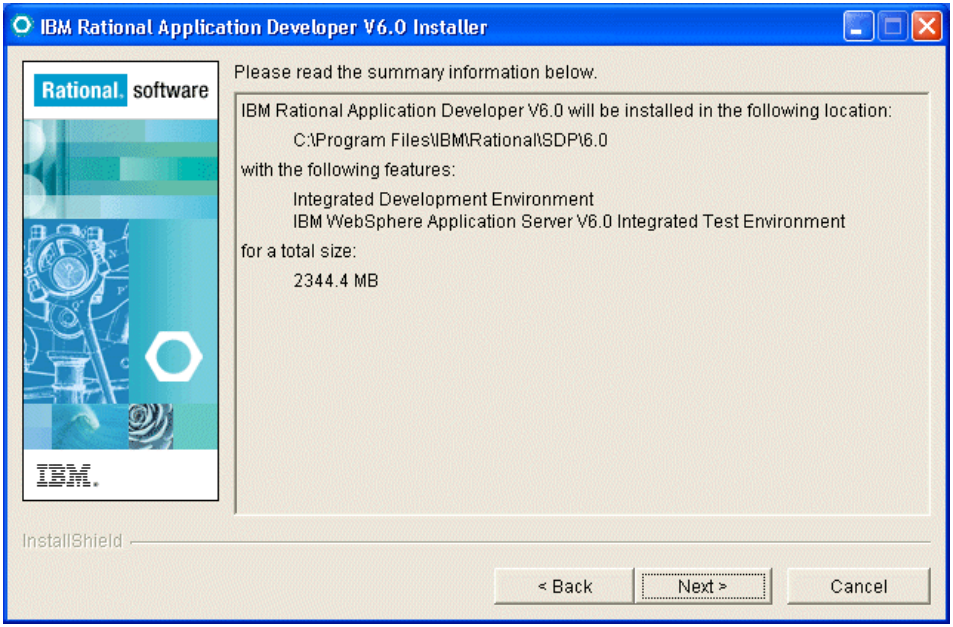


Figure 8-5 IBM Rational Application Developer V6.0 install summary

The installation can take from 30 minutes to two hours, depending on the processing speed of your system and the features selected.

8. When you see the message `The Installation Wizard has successfully installed IBM Rational Application Developer V6.0`, click **Next**.
9. The installation is now complete. We deselected **Launch Agent Controller install** and then clicked **Finish**.

8.2.3 Rational Product Updater - Refresh Pack V6.0.1.1

The *Rational Product Updater* tool is used to apply fixes to IBM Software Development tooling. In this case, we will install the Refresh Pack V6.0.1.1 on top of IBM Rational Application Developer V6.0.

The following procedure describes how to install Refresh Pack V6.0.1.1 using the Rational Product Updater:

1. Ensure that Rational Application Developer is closed. Test servers are stopped, and your computer has an Internet connection.
2. Start the Rational Product Updater by clicking **Start** → **Programs** → **IBM Rational** → **Rational Product Updater**.
3. Click **Find Updates**.
4. You might be prompted that there are required updates for the Product Updater. For example, in our case we have just installed IBM Rational Application Developer V6.0 and are prompted to install the IBM Rational Application Developer Refresh Pack V6.0.1.1. Click **OK**.

   Tip: If the Rational Product Updater requires an update to itself, you are prompted to install it before you can continue. The Rational Product Updater installs the update, restarts, and retrieves a list of available updates.

5. The IBM Rational Product Updater should be populated with updated fix information. Ensure that **Refresh Pack V6.0.1.1** is checked. Click **Install Updates**.

   Tip: Detailed information on the fix is displayed in the right-hand window by selecting the Refresh or Fix.

6. When prompted, click **Continue**.
7. When the License Agreement dialog box opens, if you are in agreement, select **I accept the terms in the license agreements**, and then click **OK** to begin the installation.

   Depending on the speed of your computer processor, the amount of RAM, and the speed of your Internet connection, the update might take an extended period of time to download and install.
8. After the installation is complete, click the **Installed Products** tab to verify that the refresh was installed successfully.
9. Close the Rational Product Updater.

Verify Rational Application Developer V6.0.1.1

After applying the Refresh Pack V6.0.1.1, we recommend that you take some steps to ensure that Rational Application Developer is working properly.

- Start Rational Application Developer.

- Ensure the WebSphere Application Server v6 test server starts properly.

8.2.4 WebSphere Application Server Test Environment Update V6.0.2.5

This section describes how to install the WebSphere Application Server Test Environment Update V6.0.2.5 using the Rational Product Updater. This update applies to WebSphere Application Server Refresh Pack V6.0.2 and Fixpack 5 (6.0.2.5).

1. Ensure that Rational Application Developer is closed. Test servers are stopped, and your computer has an Internet connection.
2. Start the Rational Product Updater by clicking **Start** → **Programs** → **IBM Rational** → **Rational Product Updater**.
3. Click the **Optional Features** tab.
4. Click **Clear All Selections**.
5. Check **IBM WebSphere Application Server V6.0 Test Environment Update** and then click **Install Features**.
6. When prompted, click **Continue**.
7. When the License Agreement dialog box opens, if you are in agreement, select **I accept the terms in the license agreements**, and then click **OK** to begin the installation.

   Depending on the speed of your computer processor, the amount of RAM, and the speed of your Internet connection, the update might take an extended period of time to download and install.
8. After the installation is complete, click the **Installed Products** tab to verify that the update was installed successfully.
9. Close the Rational Product Updater.

8.2.5 DB2 Universal Database installation

The ITSO Car Rental sample application requires IBM DB2 UDB V8.2 + Fixpack 10.

Refer to 9.3.2, “DB2 Universal Database installation” on page 295 for installation details.

After installing DB2 Universal Database the Windows services listed in Table 8-3 will be started automatically. To conserve on memory usage, we set some of the services to start manually.

Table 8-3 IBM DB2 Universal Database Windows services

Windows service name	Default startup mode	ITSO recommended startup mode
DB2 - DB2	automatic	automatic
DB2 Governor	manual	manual
DB2 JDBC Applet Server	automatic	manual
DB2 License Server	manual	manual
DB2 Remote Command Center	automatic	manual
DB2 Security Server	automatic	manual
DB2DAS - DB2DAS00	automatic	manual

8.3 Prepare for sample application development

This section describes the tasks that need to be completed prior to developing Web Service application from the ITSO Car Rental sample application. We will guide the reader through the tasks required to import and run the existing J2EE based ITSO Car Rental sample application.

Complete the following tasks to prepare for the sample application development:

1. Create a test server within Rational Application Developer
2. Download the sample code
3. Import J2EE ITSO Car Rental project interchange file
4. Create the DB2 UDB sample application database
5. Configure the data source
6. Create a database connection
7. Verify the J2EE ITSO Car Rental application
8. Enable the Web Services development capability

8.3.1 Create a test server within Rational Application Developer

By default when installing Rational Application Developer, the WebSphere Application Server v6.0 test server is configured. For simplicity we will refer to the server as *test server* in the remainder of this chapter.

If you already have a test server defined, you can continue to the next section.

If you do not have a test server configured, create a test server configuration by following these steps:

1. Open the J2EE perspective.
2. Click the **Servers** view.
3. Right-click a blank area in the Servers view, then select **New** → **Server**.
4. When the Define a New Server dialog box opens, select **WebSphere v6.0 Server** as seen in Figure 8-6 on page 218, and then click **Next**.

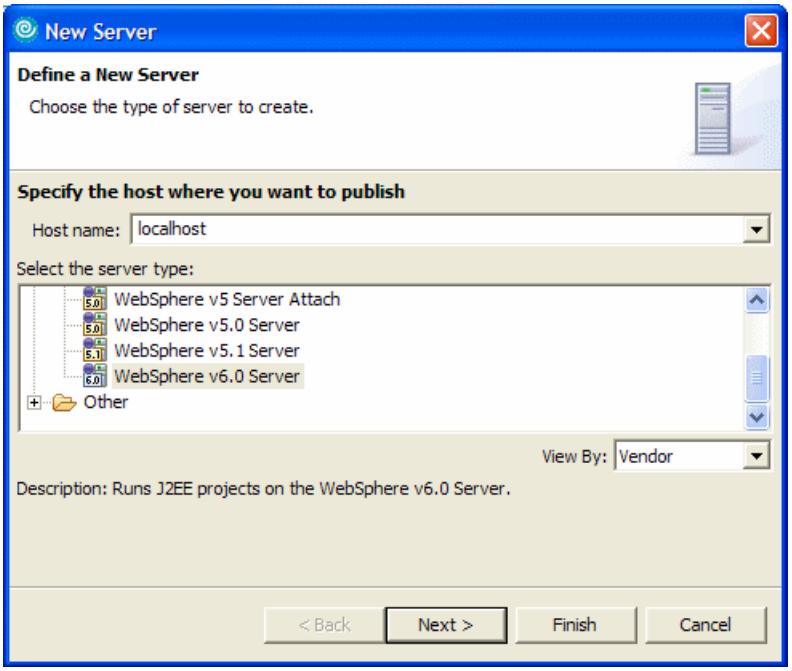


Figure 8-6 Create new server

5. When the WebSphere Server Settings, we accepted the default settings in Figure 8-7 on page 219, and then click **Finish**.

   Notice the WebSphere profile name is set to default and server1. If you want to create additional profiles, you will need to create and select the profile and server within the new server configuration in Rational Application Developer.

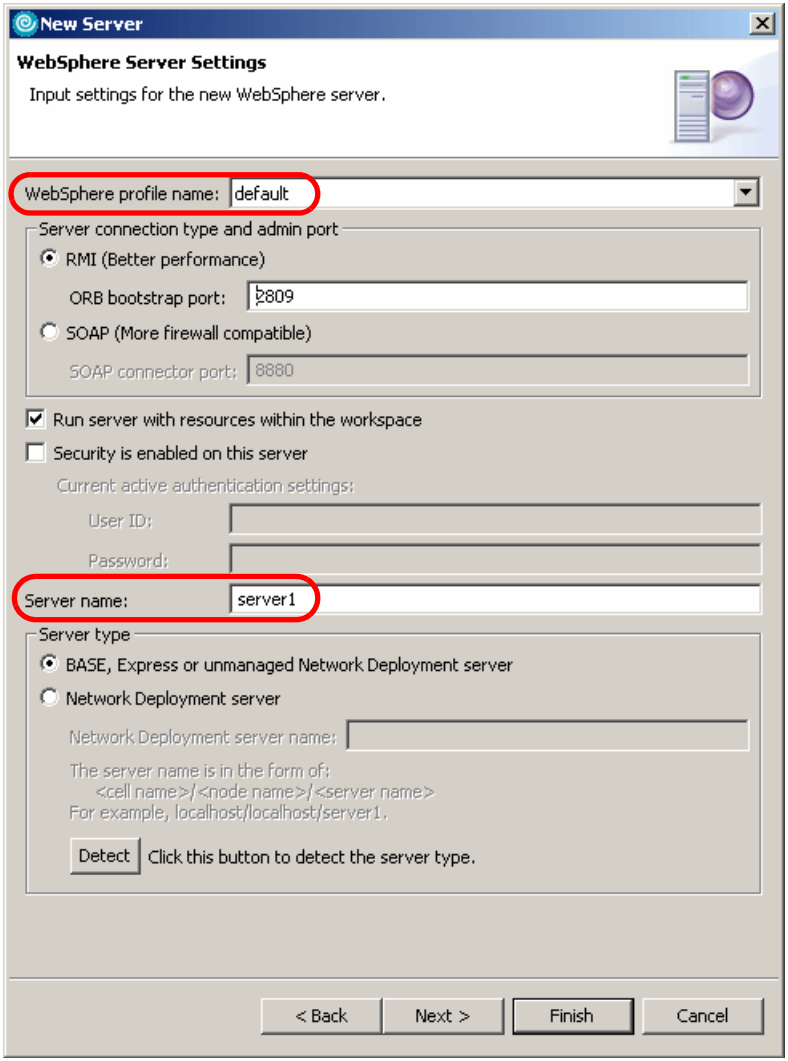


Figure 8-7 WebSphere Server Settings

8.3.2 Download the sample code

This chapter references files and database scripts supplied with this redbook's additional material. You will need to download the sample code. For details, refer to Appendix E, "Additional material" on page 515.

8.3.3 Import J2EE ITSO Car Rental project interchange file

In order to import the ITSO Car Rental sample application into the Rational Application Developer from the Project Interchange zip file, you must have completed the following tasks:

1. Start Rational Application Developer.
2. From the Workbench, select **File** → **Import**.
3. From the Import dialog box, select **Project Interchange** (as seen in Figure 8-8 on page 220),then click **Next**.

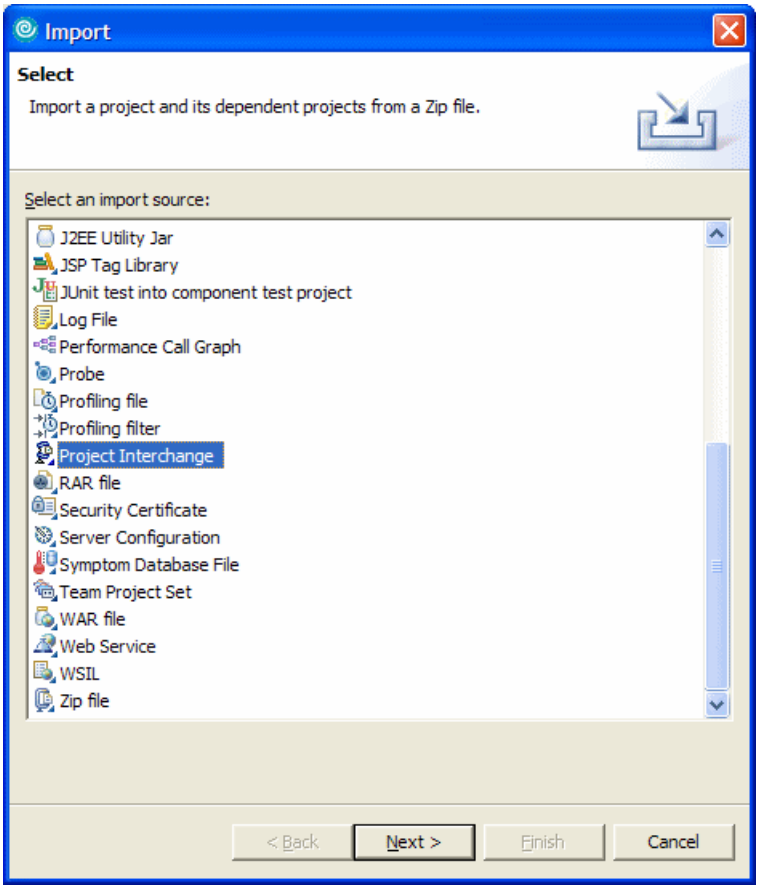


Figure 8-8 Import a Project Interchange file

4. When prompted for the Project Interchange path and filename, and target workspace location, enter the following:

   a. Enter the path and zip file name, for example:

      `C:\7240code\assemble\ITSOCarRental.j2ee\ITSOCarRental.zip`

      Click **Open**.

   b. Project location root: Enter the location of the desired workspace (for example: `C:\workspace`).

   c. Click **Select All** to check all three projects in the list, as seen in Figure 8-9 on page 221.

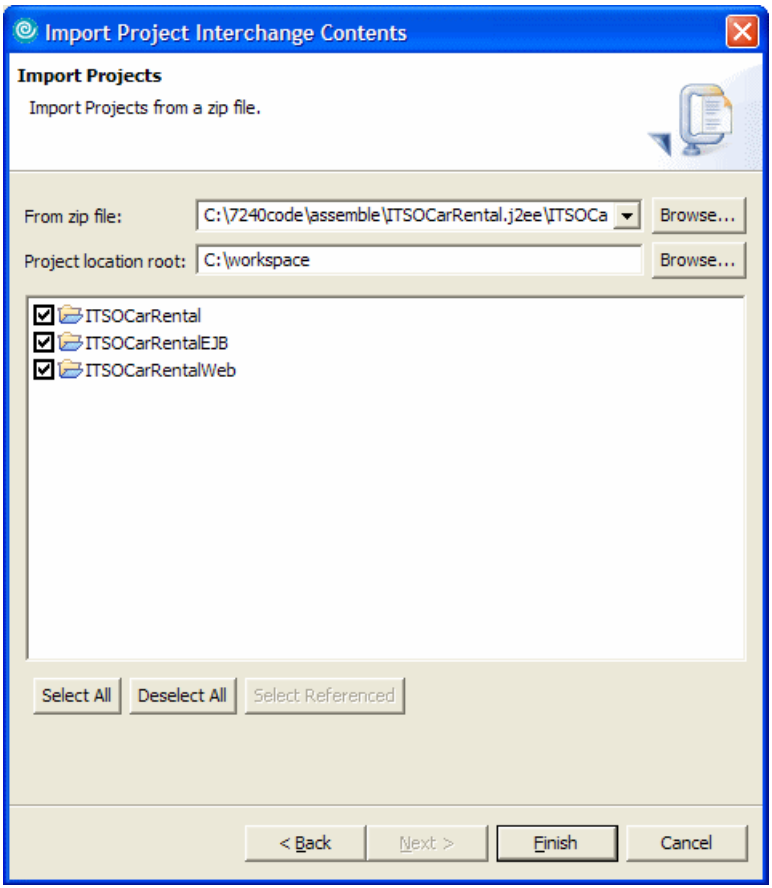


Figure 8-9 Import Project Interchange Projects location and contents

   d. Click **Finish** to being the import.

Note: The workspace will be built automatically after the import is complete. If the workspace is not built automatically, select **Project** → **Build Automatically**. After the build is complete, you can see several errors on the Problems view. We address these in the next step.

5. To clean up the errors displayed after the import, deploy the EJBProject as follows:
   a. From the Project Explorer view on J2EE perspective, expand **EJBProjects**.
   b. Right-click **ITSOCarRentalEJB** and select **Deploy** (as seen in Figure 8-10 on page 223). The errors should disappear, however there will still be several warnings.

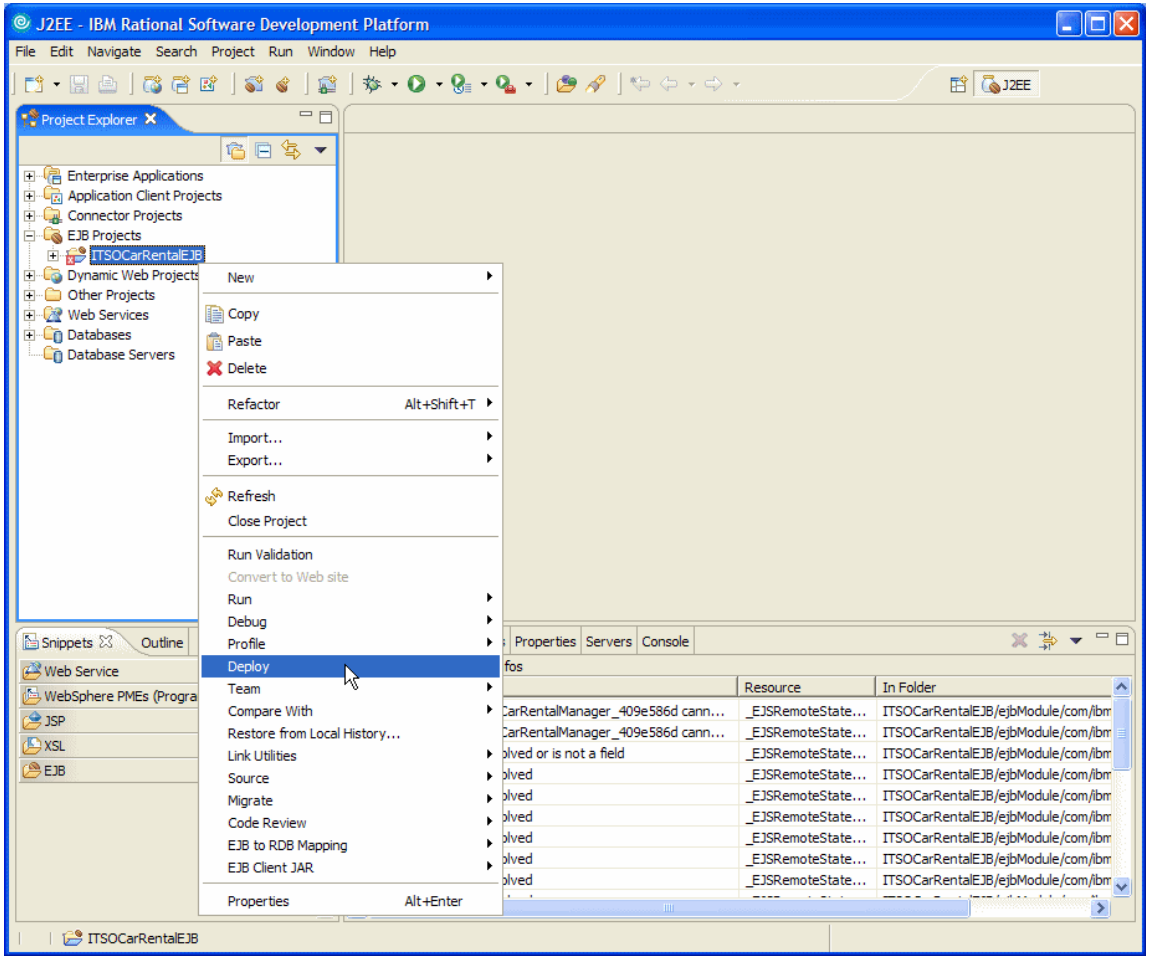


Figure 8-10 Deploy EJBProject

c. If you do not want to see these Unused imports warnings, you can change the setting to ignore these warnings by following these steps:

   i. Select **Window** → **Preferences** to open the Preferences window.

   ii. Expand **Java** on the tree, and click **Compiler**.

   iii. Click **Unused Code** tab on the right side, then select **Ignore** from the Unused imports drop-down list.

   iv. Click **OK**, and click **Yes** on the next dialog box, then these warnings will disappear.

8.3.4 Create the DB2 UDB sample application database

Create the ITSO Car Rental sample application database and populate the database with sample data. The ITSO provided sample code includes a database script to create the database tables (Table.dll) and load sample data (loadData.sql).

To create the ITSO Car Rental application database, do the following:

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 itsodb
   ```

3. Connect to the itosdb database.

   ```
   db2 connect to itsodb
   ```

4. Create the tables.

   Enter the following commands to run the **Table.ddl** to create database tables:

   ```
   db2 -tvf C:\7240code\assemble\ITSOCarRental.j2ee\database\db2\Table.ddl
   ```

5. Populate the tables with sample data.

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

   ```
   db2 -tvf C:\7240code\assemble\ITSOCarRental.j2ee\database\db2\loadData.sql
   ```

6. Disconnect from the itsodb database.

   ```
   db2 disconnect itsodb
   ```

8.3.5 Configure the data source

To configure the data source used to access the ITSO Car Rental application database, follow these steps:

1. Right-click the test server within the Servers view, and select **Start**.
2. Launch the Administrative Console.
   - To launch the WebSphere Administrative Console within Rational Application Developer, right-click the test server and select **Run administrative console**.

     or

– To launch the WebSphere Administrative Console, enter the following URL in a Web browser:

`http://localhost:9060/ibm/console`

3. Login to the WebSphere Administrative Console.
4. Create a J2C authentication alias.

   Refer to “Creating a J2C authentication alias” on page 357.
5. Create a JDBC provider.

   Refer to “Creating a JDBC provider for DB2 UDB” on page 357.
6. Create a data source.

   Refer to “Creating the DB2 data source” on page 358.

   Note: The server name should be the hostname or the IP address of the server where DB2 Universal Database is installed. In our example, we entered `localhost` for our development environment configuration.
7. Test the connection.

   Refer to “Test the connection” on page 358.

8.3.6 Create a database connection

When the database has been created, you can create a database connection to the DB2 UDB database within Rational Application Developer. This will allow you to examine the contents of the database from within Rational Application Developer, which is very useful for development and testing.

1. Open the Data Perspective.
   a. Click **Windows** → **Open Perspective** → **Other**.
   b. Check **Show All**.
   c. Select **Data** and click **OK**.
2. Right-click in the Data Explorer view and select **New Connection**.
3. When the New Database Connection dialog box opens, enter the following and then click **Next**:
   – Connection name: `itsodb connection`
   – Select **Choose a database manager and JDBC driver.**

4. When the Specify connection parameters dialog box opens, enter the following as seen in Figure 8-11 on page 226:
   a. Expand **DB2 Universal Database** under Select database manager, and select **V8.2**
   b. Select **IBM DB2 Application** from the JDBC driver drop-down list.
   c. Enter `itsodb` in the Alias field.
   d. Check **Use your operating system user ID and password**, under Specify user information.
   e. Click **Test Connection**.
   f. You should see a message, `Connection to ITSODB successful`. Click **OK**.
   g. When you are finished, click **Finish**.

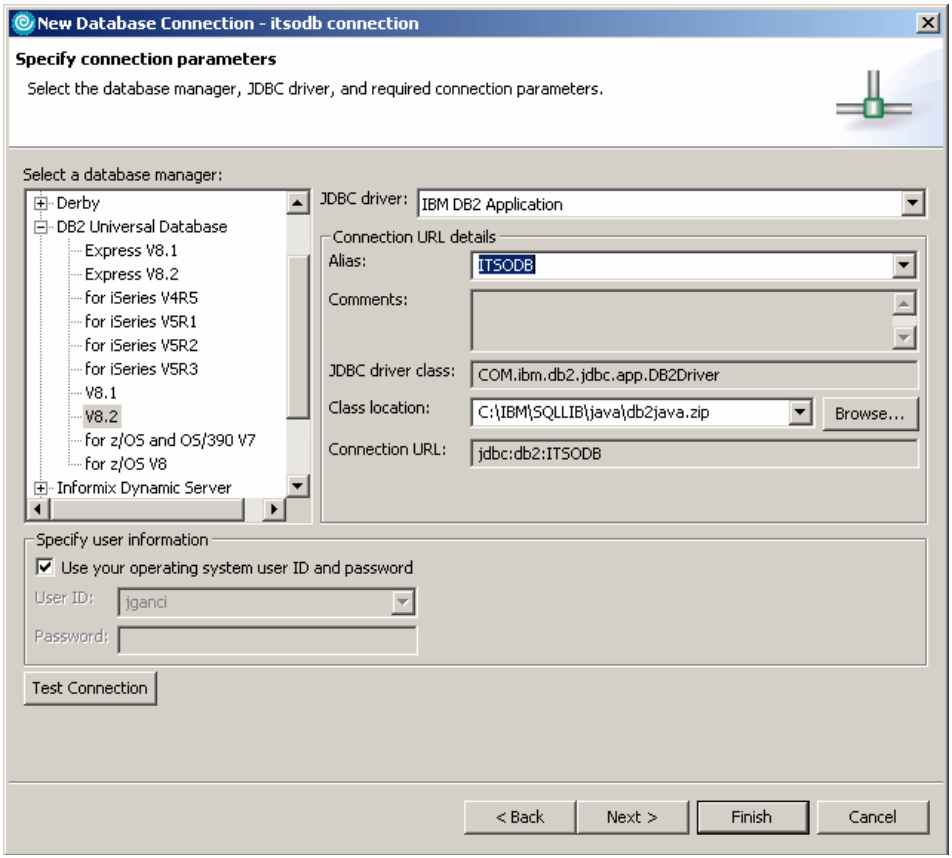


Figure 8-11 Specify connection for DB2 UDB database

5. When prompted with the message, `Do you want to copy the database metadata to a project folder?`, click **No**.

We will demonstrate how you can use the Database Explorer to view the contents of the database you have just configured within Rational Application Developer, as part of the verification of the sample application.

8.3.7 Verify the J2EE ITSO Car Rental application

This section provides a procedure to verify the application is imported and configured properly to run within the WebSphere Test Environment of Rational Application Developer.

1. Open the J2EE perspective.
2. Expand **Enterprise Applications** on Project Explorer view.
3. Right-click **ITSOCarRental** and select **Run** → **Run on server**.
4. When the Define a New Server dialog box opens, select the desired test server to run the application (for example, WebSphere Application Server v6.0) and click **Next**.
5. The ITSOCarRental application should be listed in the Configured projects column (if not, select the project and click Add) as seen in Figure 8-12 on page 228. When done click **Finish**.

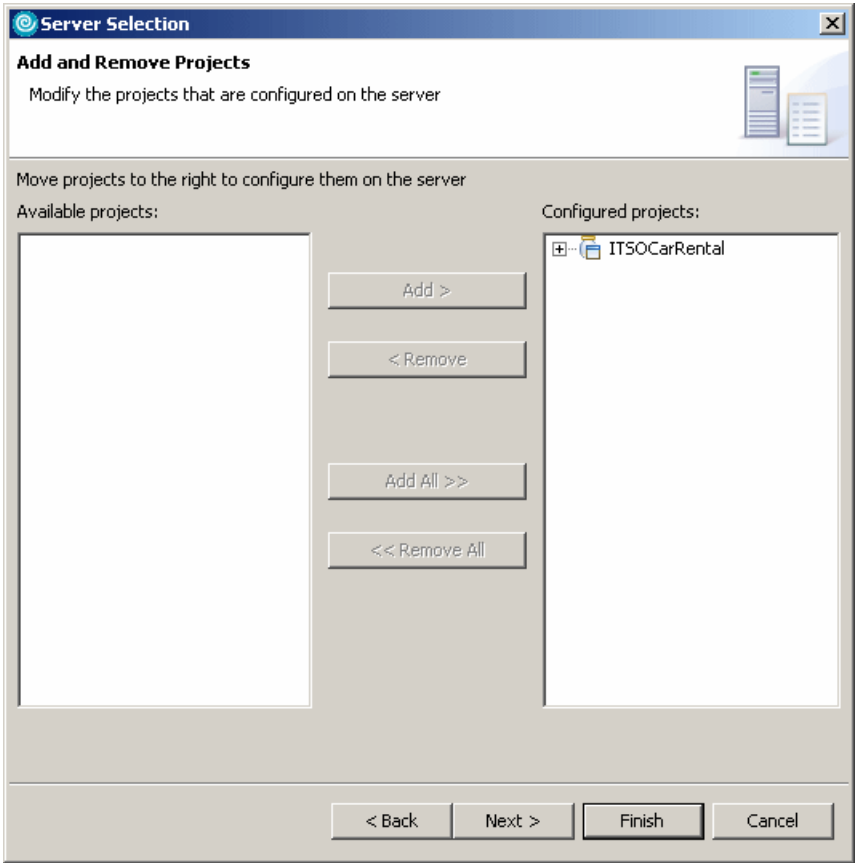


Figure 8-12 Add and Remove Projects

6. After the server status on the Server view has changed to Started, enter the following URL in a Web browser.

   `http://localhost:9080/ITSOCarRentalWeb/login.jsp`

   You should then see a login page similar to Figure 8-13 on page 229.

Figure 8-13 Login page of the sample application

7. The user ID and password fields are not used for authentication. The purpose of this login page is to create a session on the server side. You can enter any user ID and password and then click **Login**.
8. When the itinerary and preferences selection page appears (as seen in Figure 8-14 on page 230), you can select your preferences and click **Submit**. If you do not select the Car Class Preference, then all vehicle types will be displayed with rate information on the resulting page.

> **Note:** Pick up Date/Time and Drop off Date/Time must be a future date. Also, the date format mm/dd/yyyy must be entered. Otherwise, you will be just brought back this page without an error message.

Figure 8-14 Itinerary and Preference selection page

9. If you see vehicles listed on the vehicle selection page, that confirms the database configuration, because the vehicles are retrieved from the database.

   Click one of the **Select** links in the Check the Availability column (see Figure 8-14 on page 230), next to the desired vehicle.

   Note: If you see an error message here, it is likely caused by your database configuration. Check the log files (or Console view) and your configuration.

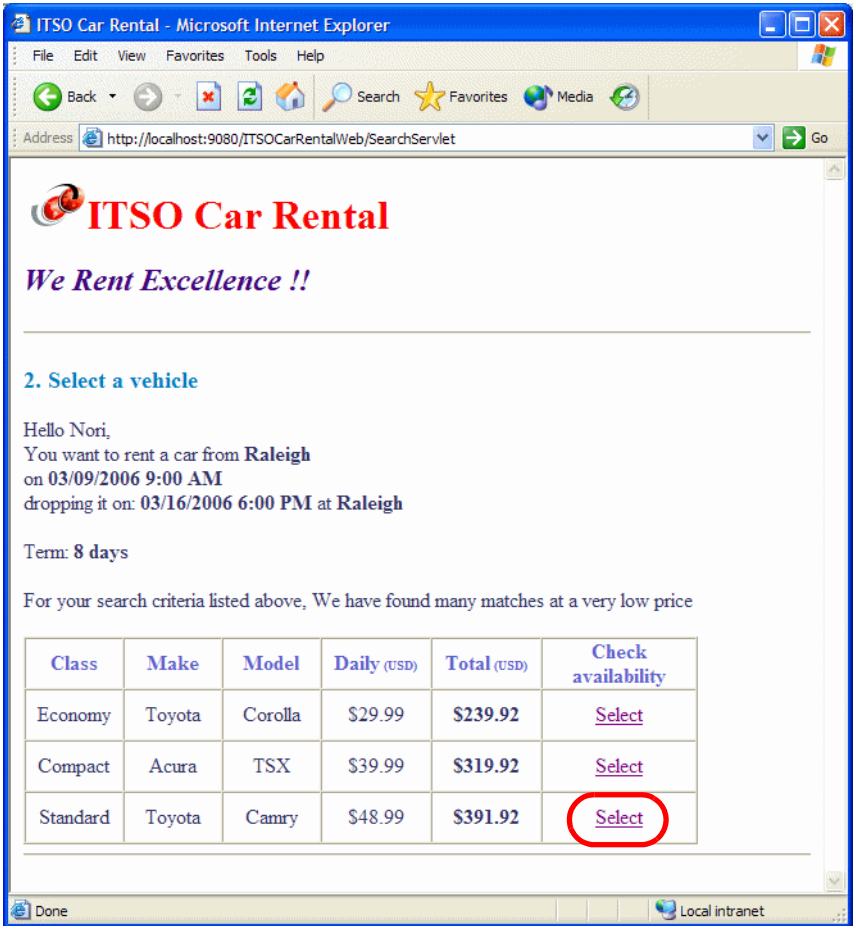


Figure 8-15 Vehicle selection page

10.:When the review and confirmation page opens, click **Book NOW** at the bottom of the page.

Figure 8-16 Review and Confirmation page

When the receipt page is shown (see Figure 8-17 on page 233), the transaction is complete.

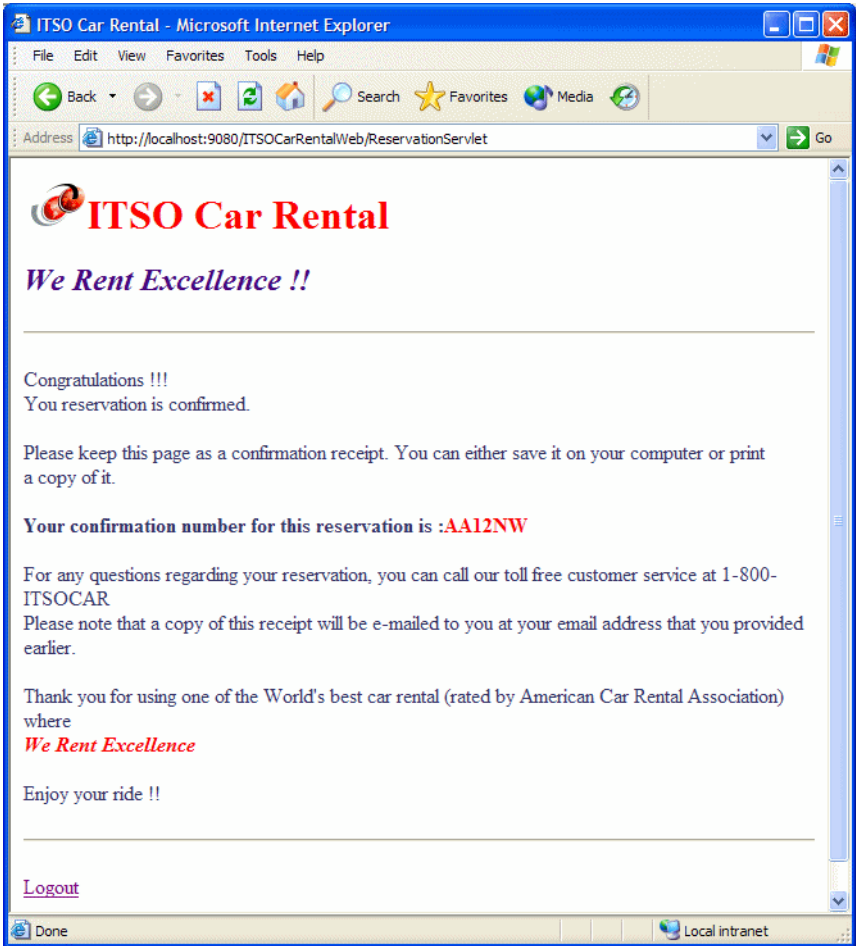


Figure 8-17 Receipt page

11. Verify that you can see a new record in RESERVATION table in the Database Explorer view of Rational Application Developer.

    a. From the Database perspective, Database Explorer view expand **itsodb connection** → **ITSODB** → **<db_owner>** → **Tables**.

    b. Right-click **<db_owner>.RESERVATION** and select **Sample contents**.

    You should see sample contents similar to what is displayed in Figure 8-18 on page 234.

Note: This procedure requires that you have created a database connection. Refer to 8.3.6, “Create a database connection” on page 225.

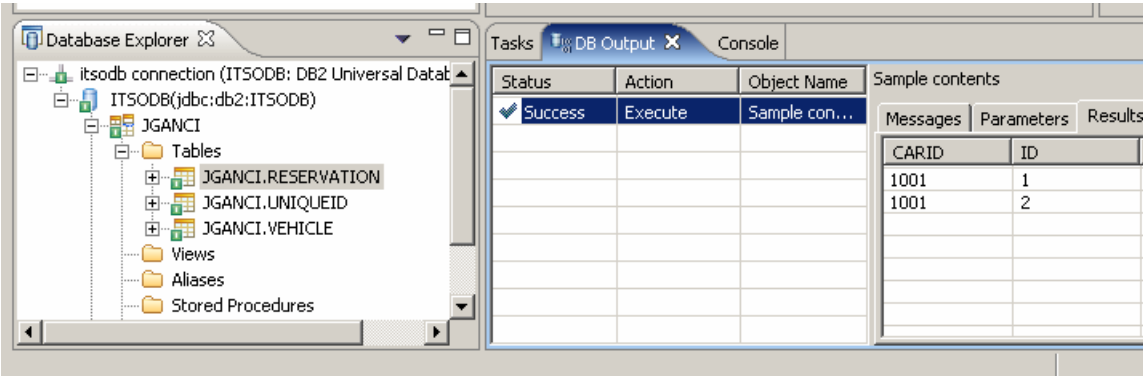


Figure 8-18 View sample data from the RESERVATION table

8.3.8 Enable the Web Services development capability

After installing Rational Application Developer, you must enable the Web Service Development capability, because by default it is not enabled.

To enable the Web Services Development capability, follow these steps:

1. From the Workbench, select **Window** → **Preferences**.
2. Expand **Workbench** and select **Capabilities**.
3. Check **Web Service Developer**.

   Expand the Web Service Developer and make sure all three items under the Web Service Developer are checked as seen in Figure 8-19 on page 235, and then click **OK**.

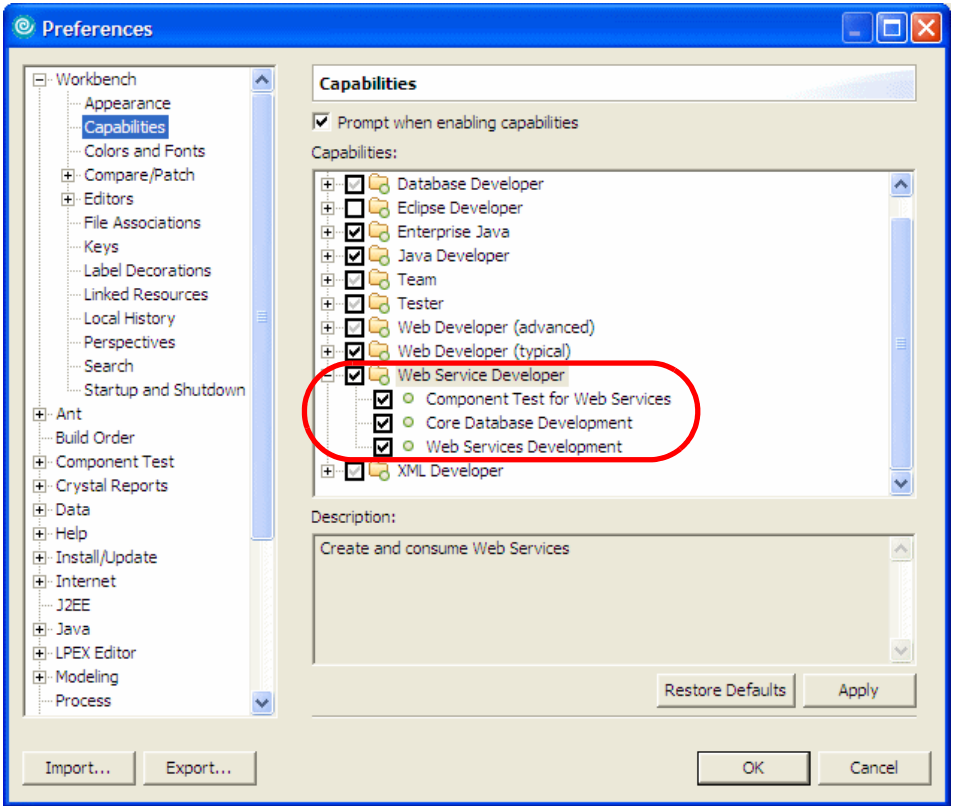


Figure 8-19 Enable Web Services Developer capability

8.4 Application development overview

This section provides an overview of the existing application and the development tasks performed to enable Web services for the sample application.

We first explain how to develop Web Services for the ITSO Car Rental application to make functionality of the application available to be invoked by other applications for reuse. Next we develop an internal Web services client application which will invoke the Web services of the ITSO Car Rental application. Lastly, we create an external Web services client application used by Travel partners to invoke a limited set of service operations found in the ITSO Car Rental Web exposed WSDL through the Web Services Gateway.

Projects from existing application

Thus far we have imported the existing ITSO Car Rental Web application projects that include EJBs and a JSF Web interface. This application will be used as a starting point.

The ITSO Car Rental base J2EE application consists of the following projects:

- ITSOCarRental

  The ITSOCarRental project is an Enterprise Applications project in within the Rational Application Developer workbench.

  - ITSOCarRentalEJB

    The ITSOCarRentalEJB is an EJB project in the workbench. This project is used to develop and package (export) an EAR file for deployment.

  - ITSOCarRentalWeb

    The ITSOCarRentalWeb is a Dynamic Web project in the workbench. This project is used to develop and package a WAR file for deployment.

Projects used to develop the Web services application

This section includes projects used within Rational Application Developer to develop Web Services to fulfill the business requirements defined in Chapter 7, “Business scenario and solution architecture” on page 173.

After development tasks within this chapter are complete, we will have the following three applications, and corresponding projects:

- ITSOCarRental
  - ITSOCarRentalEJB
  - ITSOCarRentalWeb (Web services client enabled)
- ITSOCarRentalWS
  - ITSOCarRentalWSWeb
- GlobalTravels
  - GlobalTravelsWeb

Figure 8-20 depicts the base ITSO Car Rental Application, and the Web service enabled ITSO Car Rental application.

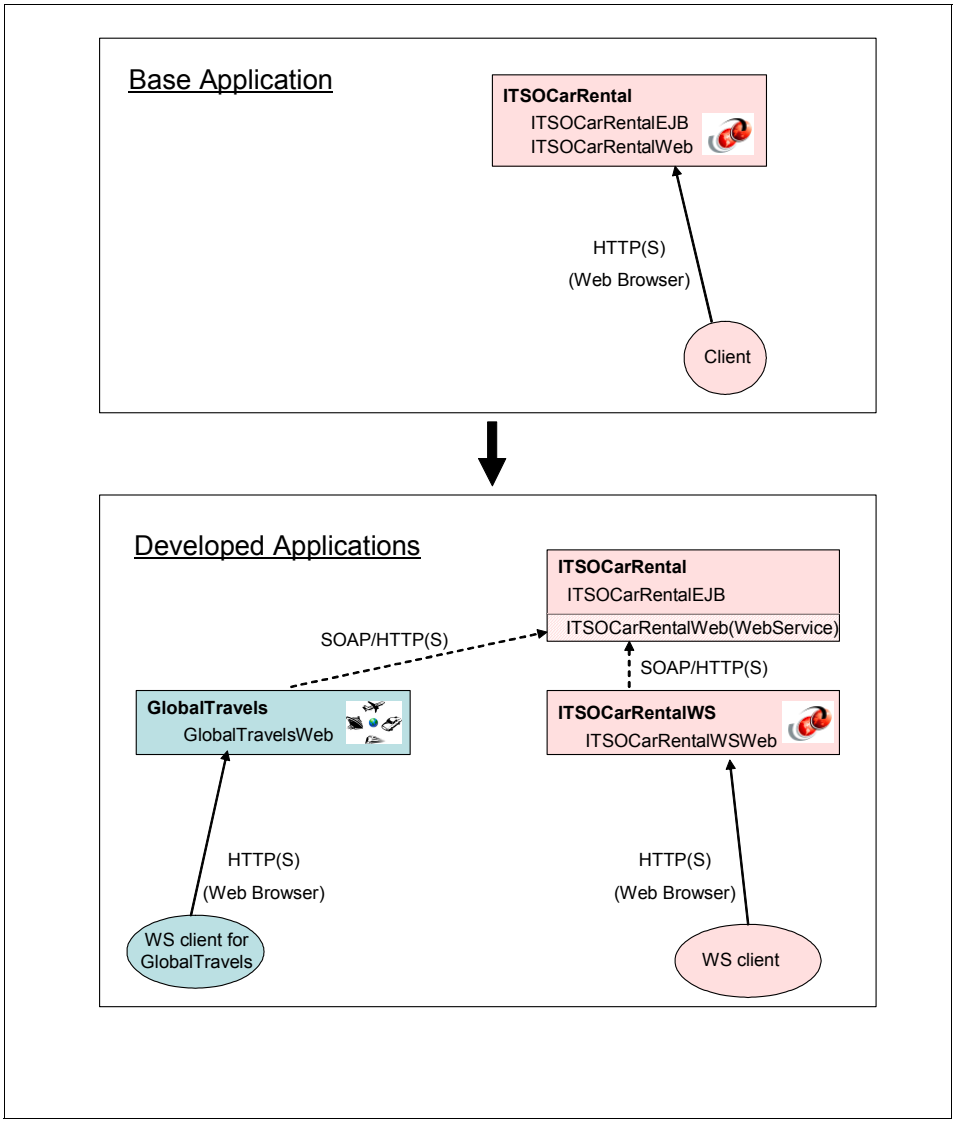


Figure 8-20 Application development overview

Completed sample code: The procedures found in the remainder of the chapter explain how to develop the Web services from the existing base application.

We have provided the completed Web services enabled sample application source in the following project interchange files:

```
C:\7240code\assemble\ITSOCarRental.ws\ITSOCarRentalFinal.zip
C:\7240code\assemble\GlobalTravels.ws\GlobalTravels.ws.zip
```

8.5 Create Web Services for ITSO Car Rental application

This section describes how to create Web services from the existing J2EE based ITSO Car Rental application functionality, and make the services available to be invoked by internal and external Web services client applications.

Complete the following tasks:

1. Modify application to capture reservation channel
2. Create project for the Web Service application
3. Create the Web Services

8.5.1 Modify application to capture reservation channel

This section describes how to modify the RESERVATION table and ITSO Car Rental application to identify the channel in which the reservation was created. This modification is being made to fulfill a business requirement of the sample. This will provide the ITSO business with an analytical means to measure the revenue generated by external travel partners.

Modify RESERVATION table to capture channel

As part of our sample application, we have provided the alterTable.sql used to modify the RESERVATION table to capture the channel the reservation was made.

To modify the existing RESERVATION table, follow these steps:

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

2. From the DB2 command window, enter the following commands to run the alterTable.sql to modify a table:

```
db2 connect to itsodb
db2 -tvf C:\7240code\assemble\ITSOCarRental.ws\database\db2\alterTable.sql
db2 connect reset
```

3. Verify that the new CUSTOMERID column has been added to the RESERVATION table. From the Database perspective, Database Explorer view expand **itsodb connection** → **ITSODB** → **<db_owner>** → **Tables** → **<db_owner>** → **RESERVATION**.

   You should see the COMPANYID column displayed.

Modify the application for channel of reservation

Now that we have the new column in the RESERVATION table, we need to modify our existing application to use and display the new channel of reservation information.

Modify the Reservation entity bean

To modify the Reservation entity bean to add the companyID CMP attribute, follow these steps:

1. Open J2EE perspective.
2. From the Project Explorer view, expand **EJB Projects** → **ITSOCarRentalEJB** → **Deployment Descriptor: ITSOCarRentalEJB** → **Entity Beans**.
3. Double-click **Reservation** under the Entity Beans to open the Deployment Descriptor editor.
4. Select **Reservation** on the left side of the editor as seen in Figure 8-21, and then click **Add** for CMP fields on the top right of the editor.

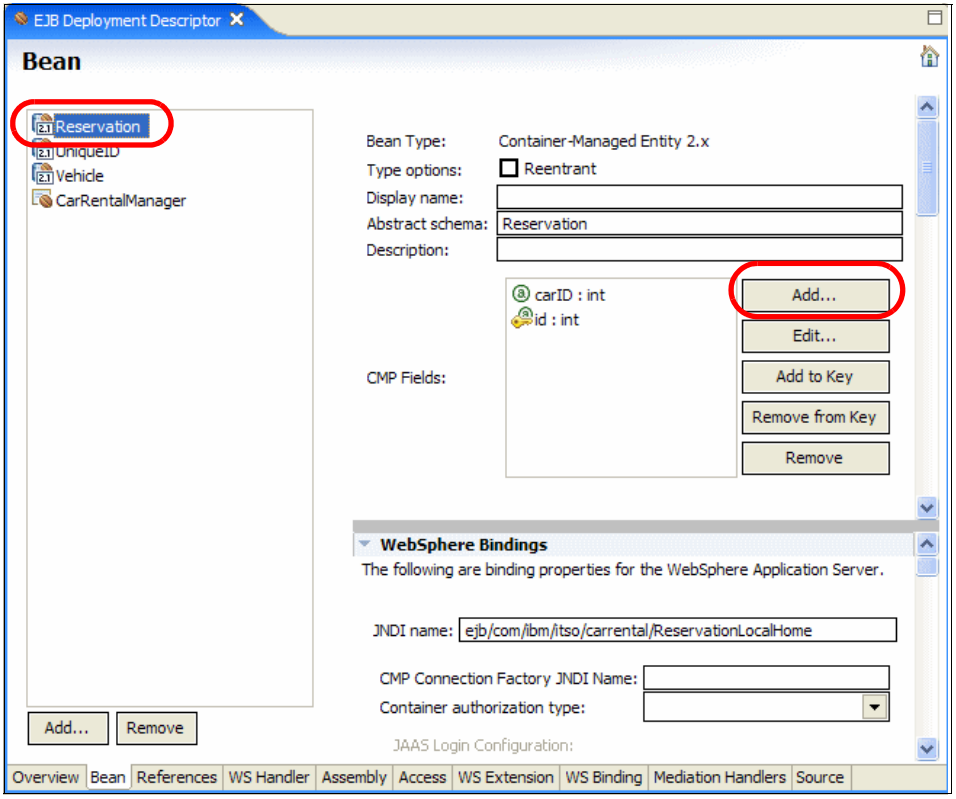


Figure 8-21 EJB Deployment Descriptor editor

5. When the CMP Fields dialog box opens, click **Add** as seen in Figure 8-22.

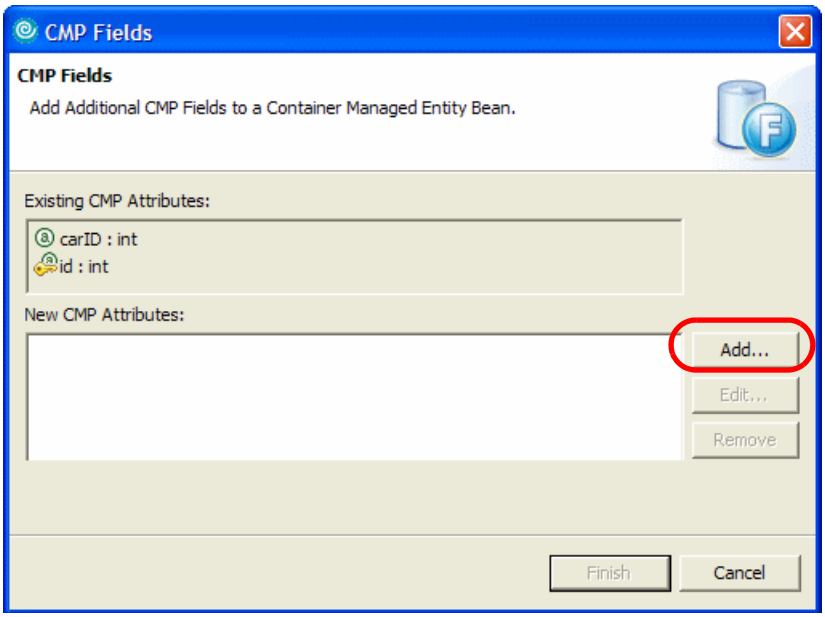


Figure 8-22 CMP Fields window

6. When the CMP Attribute dialog box opens, enter the following as seen in Figure 8-23:
   a. Enter `companyID` in the Name field.
   b. Enter `java.lang.String` in the Type field.
   c. Click **Apply**, and then click **Close**.

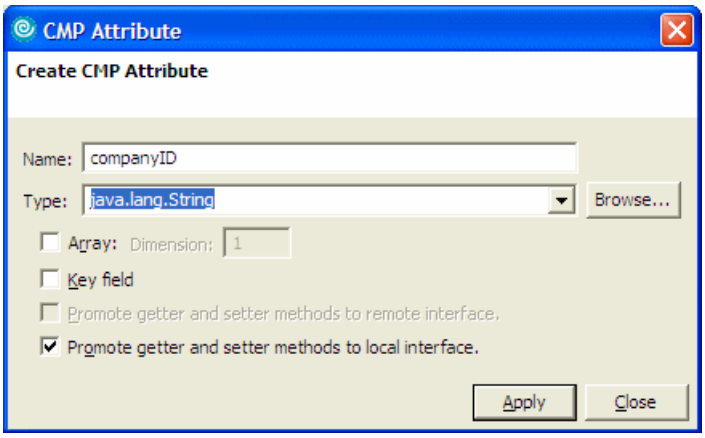


Figure 8-23 CMP Attribute window

7. The companyID should now appear in the New CMP Attributes dialog box. Click **Finish**.
8. Select **File** → **Save** to save the change on the Deployment Descriptor. You will see errors, which we fix in a later step, in the Problems view.
9. Generate EJB to RDB mapping.
    a. Expand **EJBProjects** → **ITSOCarRentalEJB** → **ejbModule** → **META-INF** on the Project Explorer view.
    b. Right-click **backends** and select **Delete**. When prompted to confirm, click **Yes**.
    c. Right-click **ITSOCarRentalEJB** project on Project Explorer view and select **EJB to RDB Mapping** → **Generate Map**.
    d. When the EJB to RDB Mapping dialog box opens, select **Create a new backend folder**, and then click **Next**.
    e. When the Create new EJB / RDB Mapping dialog box opens, select **Top-Down**, and then click **Next**.
    f. Select **DB2 Universal Database V8.2** from the Target Database drop-down list, and then click **Finish**.
10. Deploy modified EJBs.
    a. To delete existing EJB deploy code for Cloudscape™, right-click the **com.ibm.itso.carrental.websphere_deploy.CLOUDSCAPE_V51_1** package under ejbModule on Project Explorer view and select **Delete**, then click **Yes**.
    b. To delete existing EJB deploy code for DB2, right-click the **com.ibm.itso.carrental.websphere_deploy.DB2NT_V82_1** package under ejbModule on Project Explorer view and select **Delete**, then click **Yes**.
    c. Right-click the **ITSOCarRentalEJB** project on Project Explorer view and select **Deploy**.

Modify the CarRentalManagerBean session bean

Now that the CMP has been modified, we need to modify CarRentalManagerBean session bean. The session bean uses the CMP so that the companyID is inserted to the itsodb database when a reservation is made.

1. Modify CarRentalManagerBean.java.
    a. Expand **EJB Projects** → **ITSOCarRentalEJB** → **ejbModule** → **com.ibm.itso.carrental**.
    b. Double-click **CarRentalManagerBean.java** to open in the Java Editor.

c. From the Outline view, do the following:

   i. Right-click the **createReservation(int)** method and select **Enterprise Bean → Demote from Local Interface** (see Figure 8-24).

   ii. Right right-click the **createReservation(int)** method again and select **Enterprise Bean → Demote from Remote Interface** (see Figure 8-24).

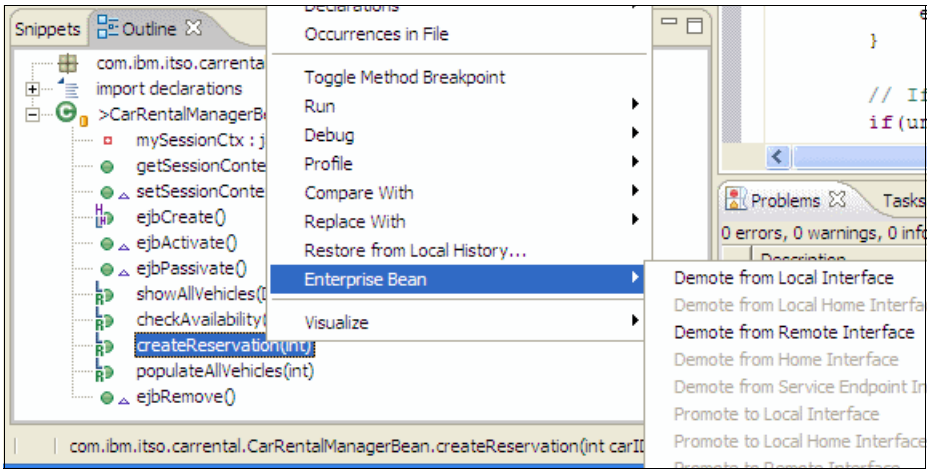


Figure 8-24 Demote from interface

Note: After it is demoted, the pop-up will display Promote from ....

d. Add an argument for the companyID on the createReservation method in the CarRentalManagerBean.java as follows:

   - Original code:

     ```
     public int createReservation(int carID)
     ```

   - Modified code:

     ```
     public int createReservation(int carID, String companyID)
     ```

e. Add the following code under `reservationLocal.setCarID(carID)`:

```
reservationLocal.setCarID(carID);
reservationLocal.setCompanyID(companyID);
```

Note: The completed CarRentalManagerBean.java is provided with the sample code in the following directory:

`C:\7240-SOA\7240code\assemble\intermediatefiles\modify_ejb\`

f. From the Outline view, follow these steps:

i. Right-click **createReservation(int, String)** method and select **Enterprise Bean → Promote to Local Interface** (see Figure 8-25).

ii. Right-click the **createReservation(int, String)** method again and select **Enterprise Bean → Promote to Remote Interface** (see Figure 8-25).

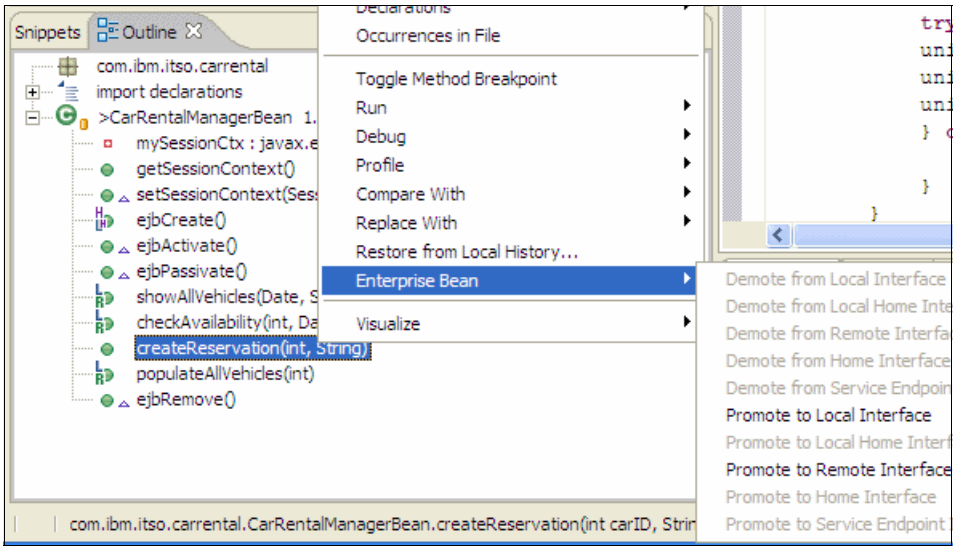


Figure 8-25 Promote the interface

2. Modify the ReservationWrapper.java.

a. Expand **EJB Projects → ITSOCarRentalEJB → ejbModule → com.ibm.itso.carrental.wrapper**.

b. Double-click **ReservationWrapper.java** to open in the Java Editor.

c. Add the following field, setter and getter at the bottom of the class:

```
private String companyID;
public String getCompanyID() {return this.companyID;}
public void setCompanyID(String companyID){this.companyID = companyID;}
```

Note: The completed ReservationWrapper.java is provided with the sample code in the following directory:

`C:\7240-SOA\7240code\assemble\intermediatefiles\modify_ejb\`

3. Modify the ReservationServlet.java.
   a. Expand **Dynamic Web Projects** → **ITSOCarRentalWeb** → **Java Resources** → **JavaSource** → **com.ibm.itso.carrental.web**.
   b. Double-click **ReservationServlet.java** to open in the Java Editor.
   c. Modify the ReservationServlet.java to have the additional argument:
      - Original code:

        ```
        reservationID = carRentalMgr.createReservation(carID);
        ```

      - Modified code:

        ```
        reservationID = carRentalMgr.createReservation(carID,
        "INT_ITSO_ORG");
        ```

   Note: The completed ReservationServlet.java is provided with the sample code in the following directory:

   ```
   C:\7240-SOA\7240code\assemble\intermediatefiles\modify_ejb\
   ```

4. Deploy the new code.
   a. Expand **EJB Projects** → **ITSOCarRentalEJB** → **ejbModule** → **com.ibm.itso.carrental**.
   b. Delete the following files under the com.ibm.itso.carrental package, which should have some errors.

      ```
      _EJSRemoteStatelessCarRentalManagerHome_409e586d_Tie.java
      _CarRentalManagerHome_Stub.java
      _CarRentalManager_Stub.java
      ```

   c. Right-click **ITSOCarRentalEJB** project on Project Explorer view and select **Deploy**.

   Note: If you still encounter files with errors, delete these files and Deploy them again.

5. Verify that the application is working properly.

   Refer to 8.3.7, “Verify the J2EE ITSO Car Rental application” on page 227 for details. Verify data has been inserted into the companyID column of the RESERVATION table.

8.5.2 Create project for the Web Service application

This section describes how to create a new project named ITSOCarRentalWS, that will be used as Web service client as well as Web application.

To create a new Enterprise Application project, do the following:

1. Open the J2EE perspective.
2. Select **File** → **New** → **Enterprise Application Project**.
3. When the Enterprise Application Project dialog box opens, enter `ITSOCarRentalWS` in the Name file, and then click **Next**.
4. When the EAR Module Projects dialog box opens, click **New Module**.
5. When the New Module Project dialog box opens, do the following (as seen in Figure 8-26), and then click **Finish**:
   - Deselect **Application Client project.**
   - Deselect **EJB project.**
   - Deselect **Connector project.**

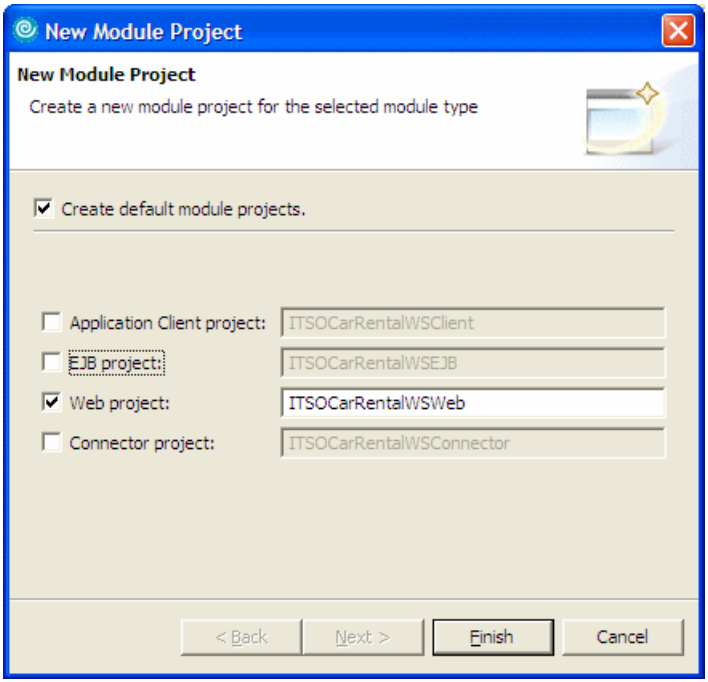


Figure 8-26 New Module Project

6. When returning to the EAR Module Projects dialog box, select **ITSOCarRentalWSWeb** (as seen in Figure 8-27 on page 247), and then click **Finish**.

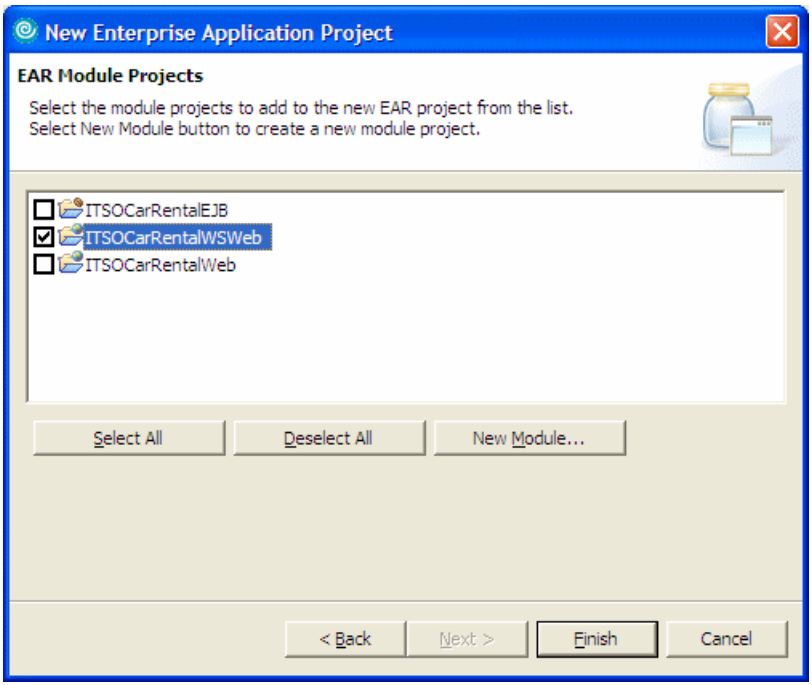


Figure 8-27 New Enterprise Application project

8.5.3 Create the Web Services

This section describes how to expose the session bean CarRentalManager as a Web service from the generated WSDL file.

Generate wrapper classes

To generate wrapper classes for the RESERVATION and VEHICLE tables, we took a short cut and added a dummy EJB method that has these wrapper classes as arguments.

To automatically generate a stub, add a dummy method to the session bean as follows:

1. Expand **EJB Projects -> ITSOCarRentalEJB** → **ejbModule** → **com.ibm.itso.carrental**.

2. Open CarRentalManager.java and add the method listed in Example 8-1.

Example 8-1 CarRentalManager.java code snippet

```
public void generateWSStubs(
   VehicleWrapper vWrapper, ReservationWrapper rWrapper)
   throws java.rmi.RemoteException;
```

3. Open CarRentalManagerBean.java and add the method listed in Example 8-2.

Example 8-2 CarRentalManagerBean.java code snippet

```
public void generateWSStubs(
   VehicleWrapper vWrapper, ReservationWrapper rWrapper){
}
```

4. Open CarRentalManagerLocal.java and add the method listed in Example 8-3.

Example 8-3 CarRentalManagerLocal.java code snippet

```
public void generateWSStubs(
   VehicleWrapper vWrapper, ReservationWrapper rWrapper);
```

5. Open CarRentalManagerBean.java and view the file in the Outline view.
   a. Ensure **generateWSStubs** has been promoted (the icon has “L” and “R”).
   b. If not, right-click the **generateWSStubs** and select **Enterprise Bean** → **Promote from Local Interface**, then right-click the **generateWSStubs** again and select **Enterprise Bean** → **Promote from Remote Interface**.
6. Delete `_CarRentalManager_Stub.java` if it has an error.
7. Right-click the **ITSOCarRentalEJB** project on Project Explorer view and select **Deploy**.

Create the Web service

To create the Web service, follow these steps:

1. Expand **EJB Projects** → **ITSOCarRentalEJB** → **Deployment Descriptor: ITSOCarRentalEJB** → **Session Beans** on the Project Explorer view.
2. Right-click **CarRentalManager** and select **WebServices** → **Create Web service**.

> **Note:** If you do not see the Web Services item on the pop-up window, return to 8.3.8, “Enable the Web Services development capability” on page 234 to confirm that the capability has been enabled.

3. When the Web Services dialog box opens, select **EJB Web Service** from the Web service type drop-down list (as seen in Figure 8-28 on page 249), and then click **Next**.

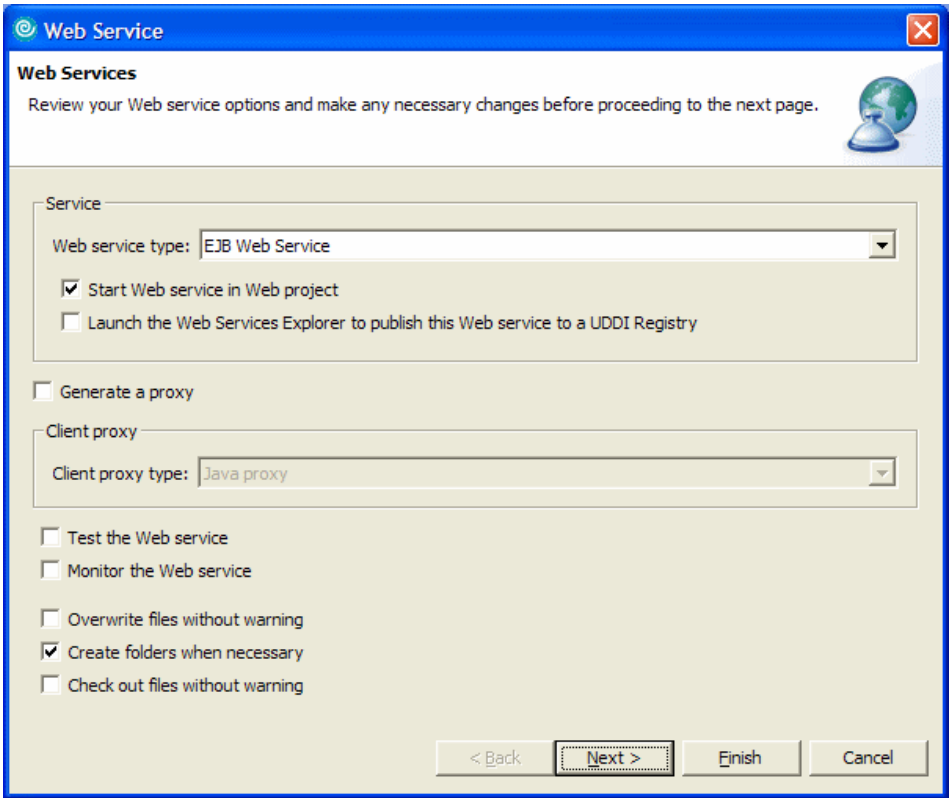


Figure 8-28 Web Services

4. When the Object Selection dialog box opens, accept the defaults (as seen in Figure 8-29 on page 250) and click **Next**.

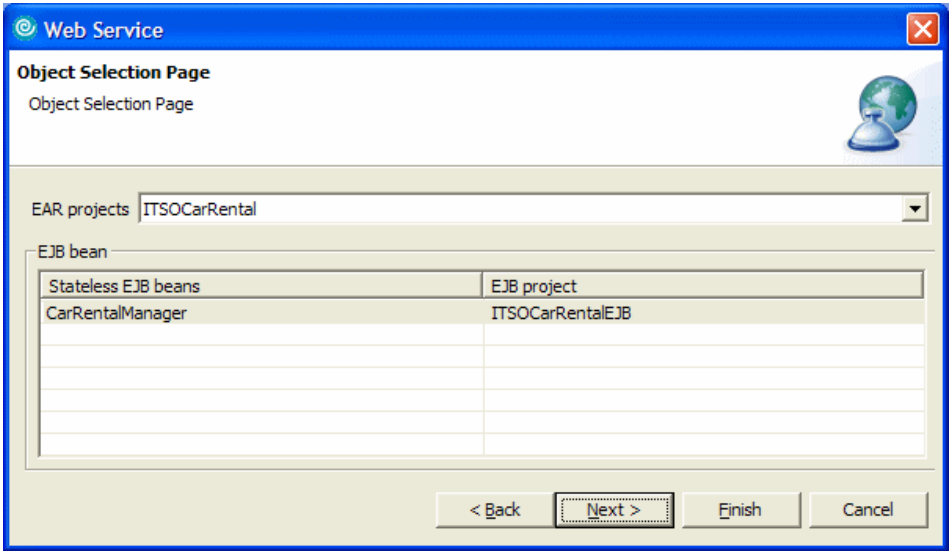


Figure 8-29 Object Selection Page

5. When the Service Deployment Configuration dialog box opens, accept the default settings (as seen in Figure 8-30), and then click **Next**.

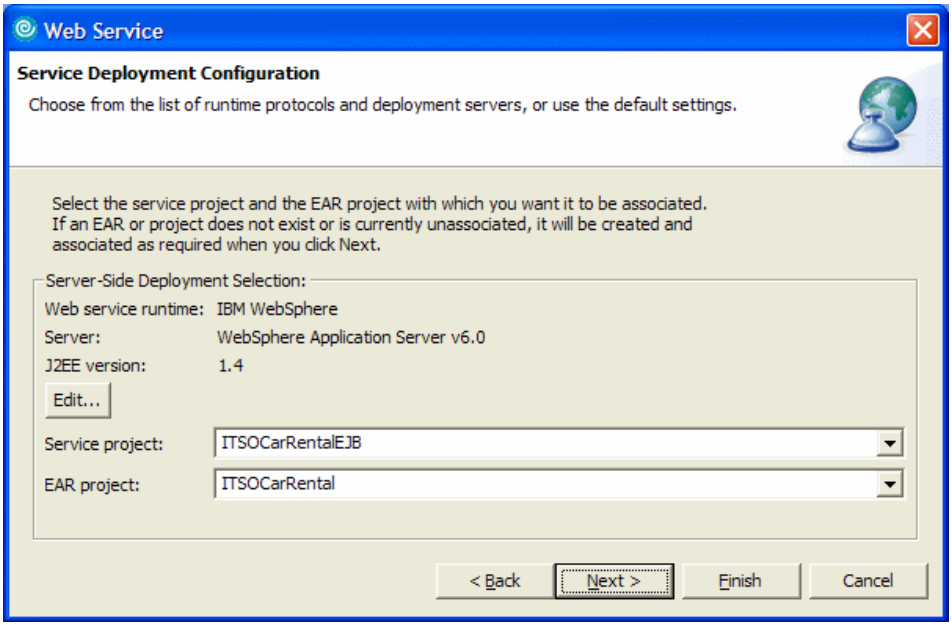


Figure 8-30 Service Deployment Configuration

6. When the Web Service EJB Configuration dialog box opens, select **ITSOCarRentalWeb** from the Select Router Project drop-down list, select **SOAP over HTTP** from Select transports (as seen in Figure 8-31), and then click **Next**.

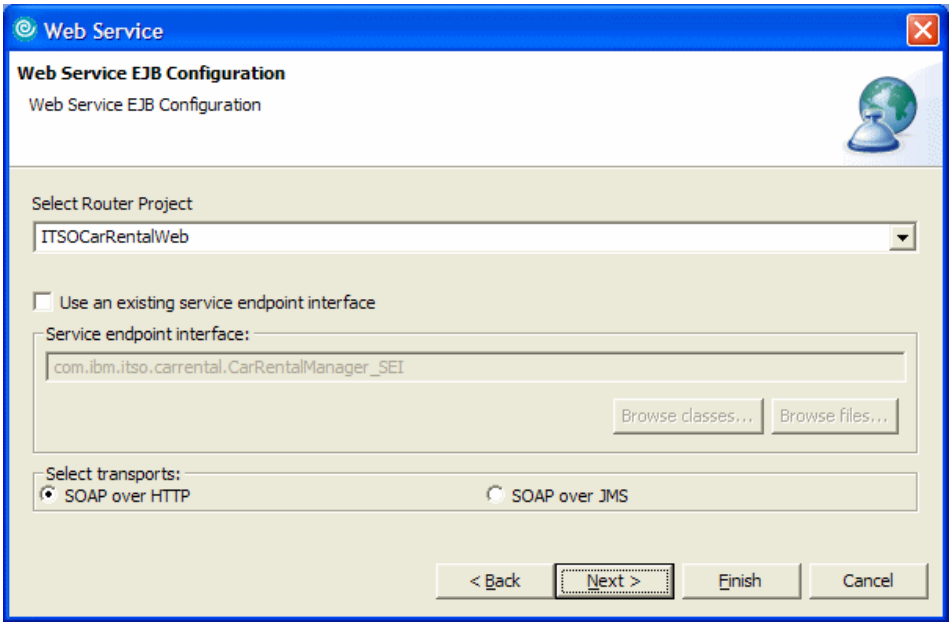


Figure 8-31 Web Service EJB Configuration

7. When the Web Service Java Bean Identity dialog box opens, accept the default settings (as seen in Figure 8-32 on page 252) and then click **Next**.

> **Note:** The generated WSDL file will be placed in the WSDL folder displayed in Figure 8-32 on page 252.

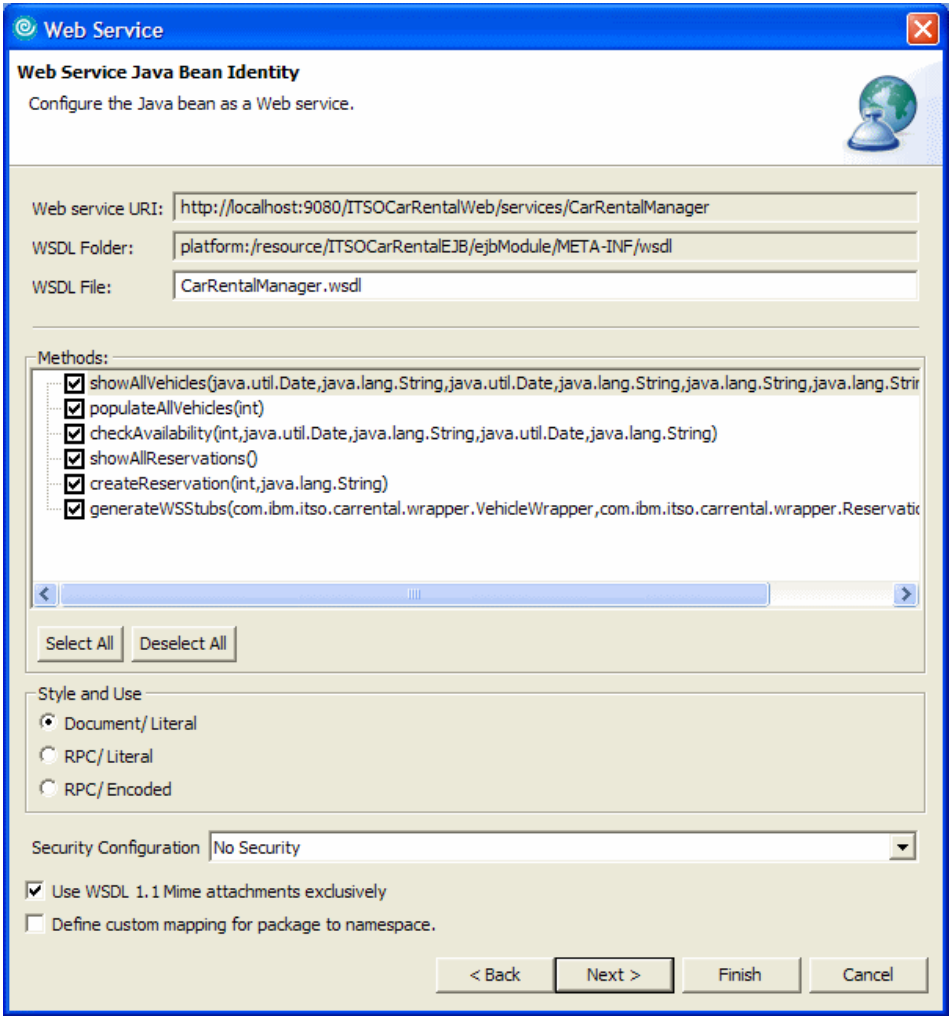


Figure 8-32 Web Service Java Bean Identity

8. When the Web Service Publication dialog box opens, we accepted the default settings (as seen in Figure 8-33 on page 253) and then clicked **Finish**.

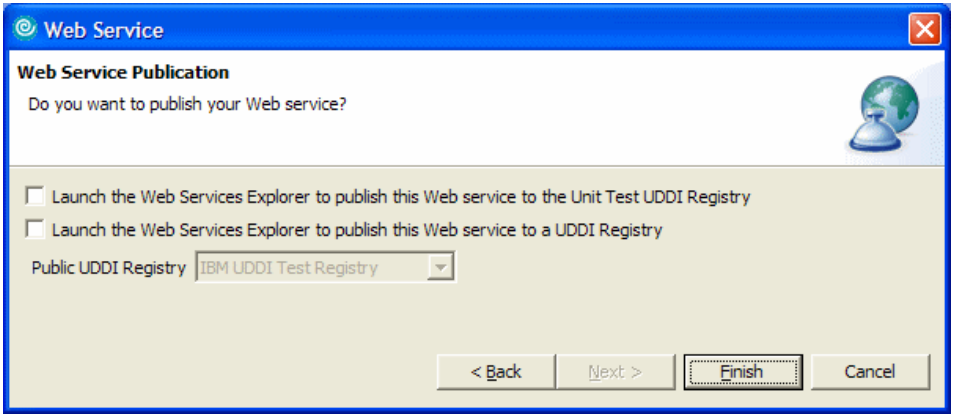


Figure 8-33 Web Service Publication

Verify the Web service

To verify that the Web Service has been created correctly, follow these steps:

1. Enter the following URL in a Web browser on the Developer node:

   `http://localhost:9080/ITSOCarRentalWeb/services/CarRentalManager`

   You should see a resulting page similar to Figure 8-34.

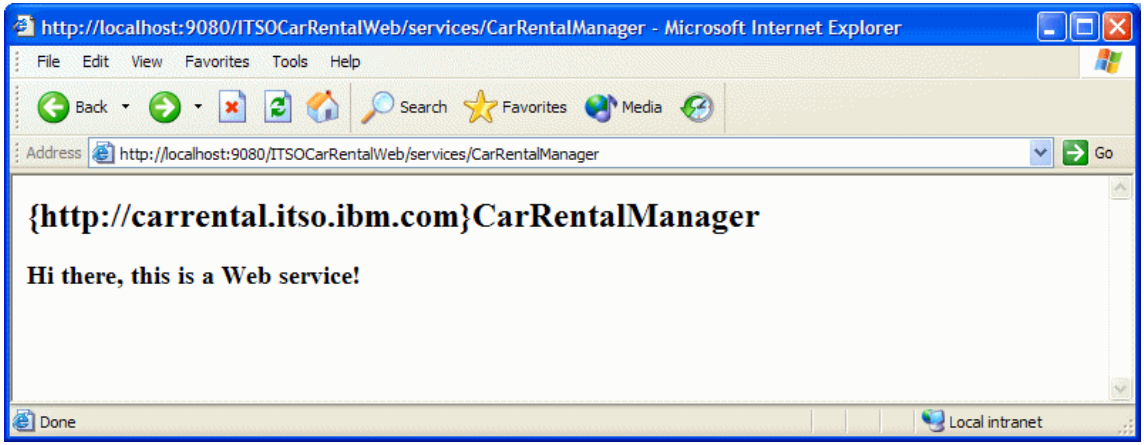


Figure 8-34 Web Service view through Web browser

2. Enter the following URL in a Web browser on the Developer node:

   `http://localhost:9080/ITSOCarRentalWeb/services/CarRentalManager?wsdl`

   You should see a resulting page similar to Figure 8-35.

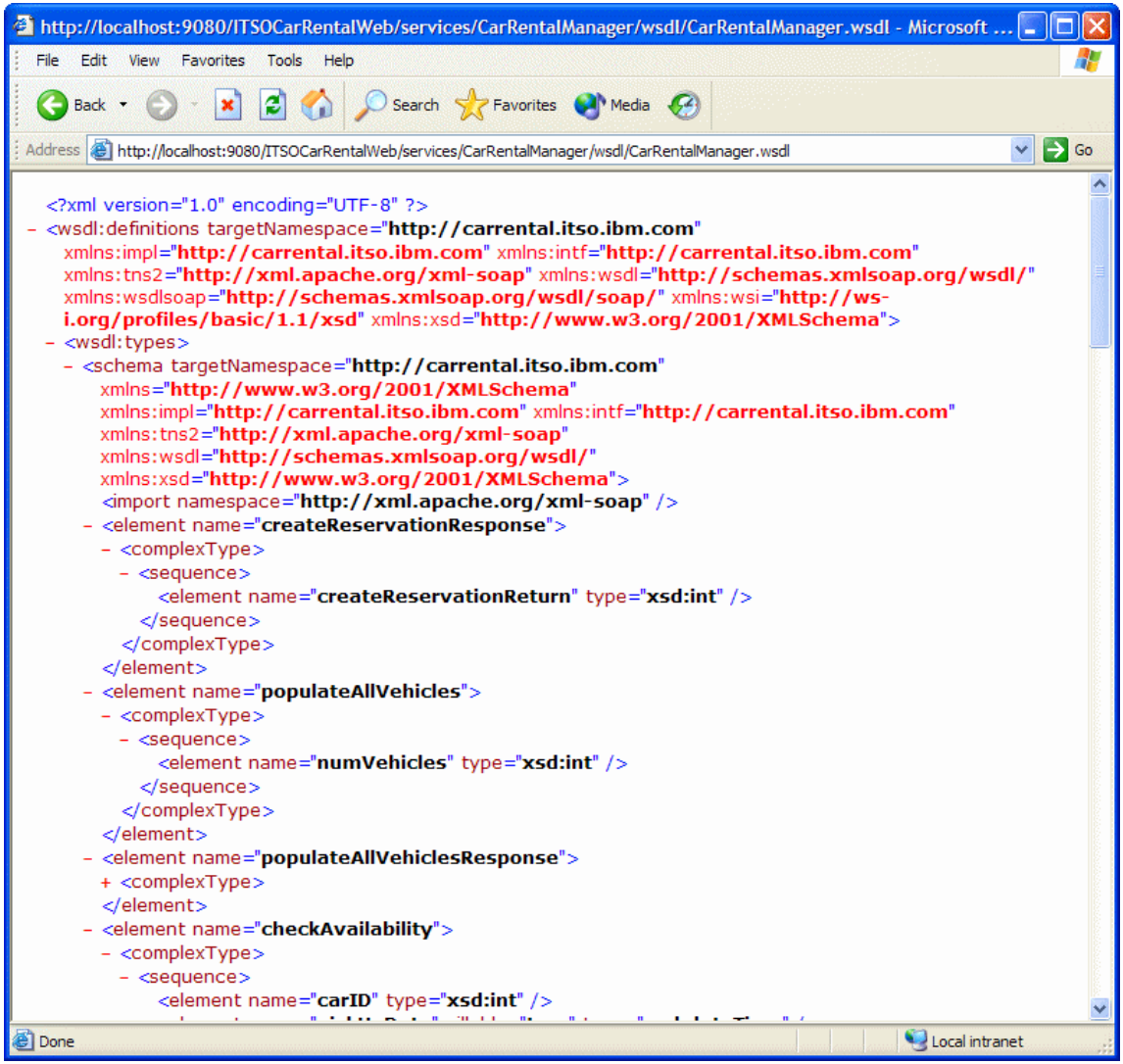


Figure 8-35 WSDL view through Web browser

8.6 Create the internal Web Service client application

This section describes how to create the internal Web services client from the WSDL file generated in 8.5.3, “Create the Web Services” on page 247. In the ITSO sample application, the Web Services client will invoke the ITSO Car Rental Web service internal to the ITSO.

Complete the following tasks:

- Generate the Web Service Client
- Modify Web application to invoke the Web service
- Add ability to view car rental channel statistics

8.6.1 Generate the Web Service Client

To generate the Web Service Client from the Web Service WSDL file, follow these steps:

1. Expand **EJBProject** → **ITSOCarRentalEJB** → **ejbModule** → **META_INF** → **wsdl** on the Project Explorer view.
2. Right-click **CarRentalManager.wsdl** and select **WebServices** → **Generate Client** (as seen in Figure 8-36).

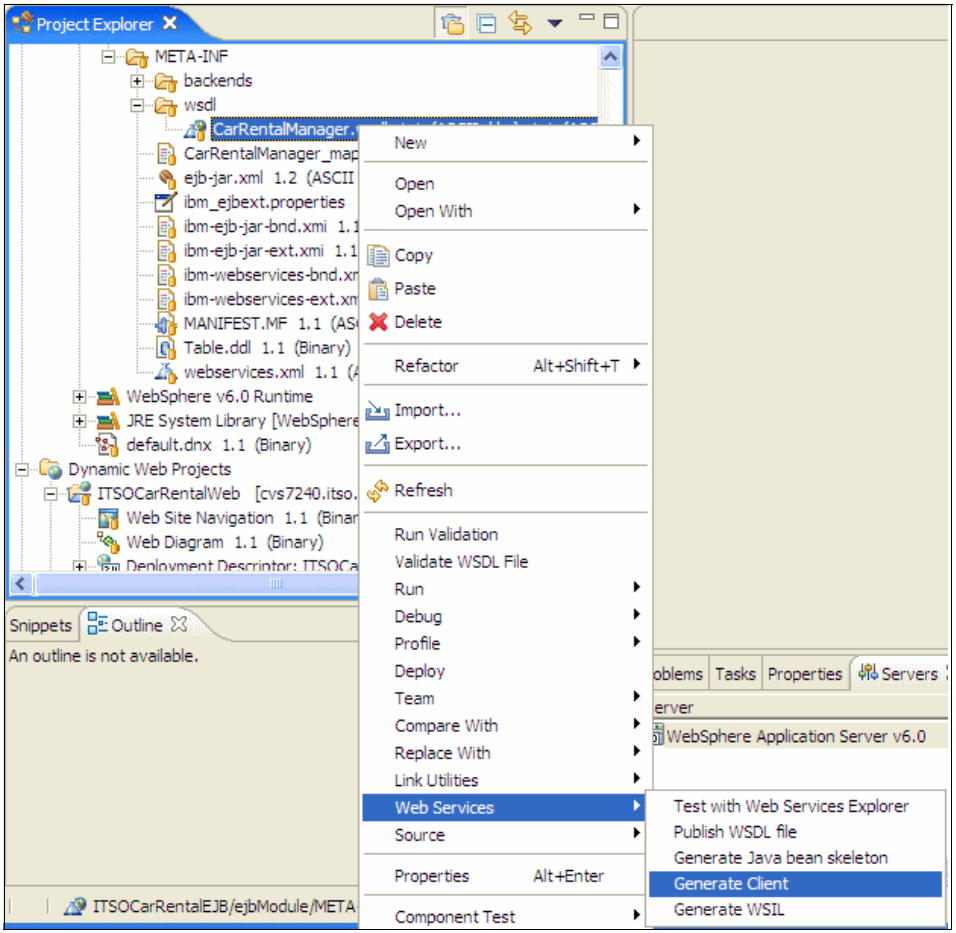


Figure 8-36 Generate Client

3. When the Web Services dialog box opens, accept the default settings as seen in Figure 8-37 on page 257, and then click **Next**.

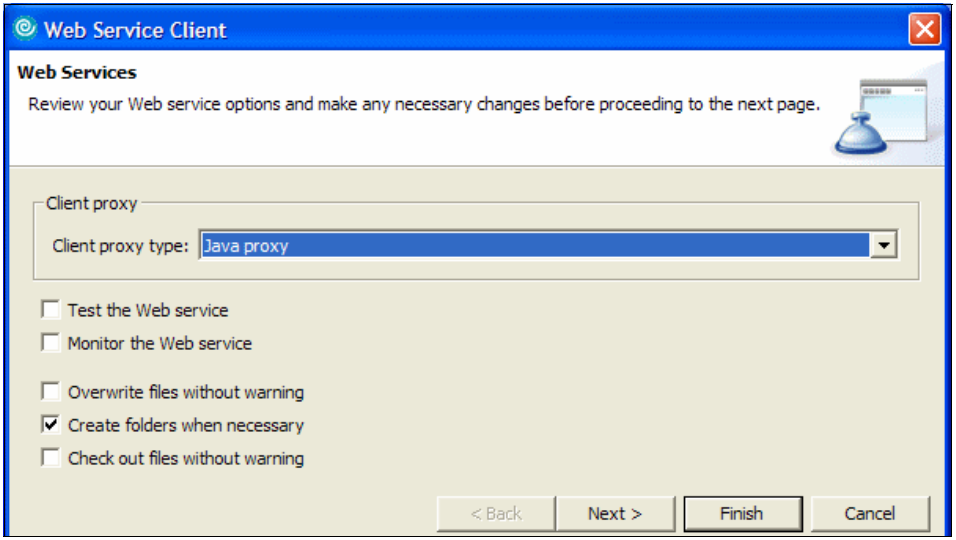


Figure 8-37 Web Services

4. When the Web Service Selection dialog box opens, we accepted the default setting as seen in Figure 8-38, and clicked **Next**.

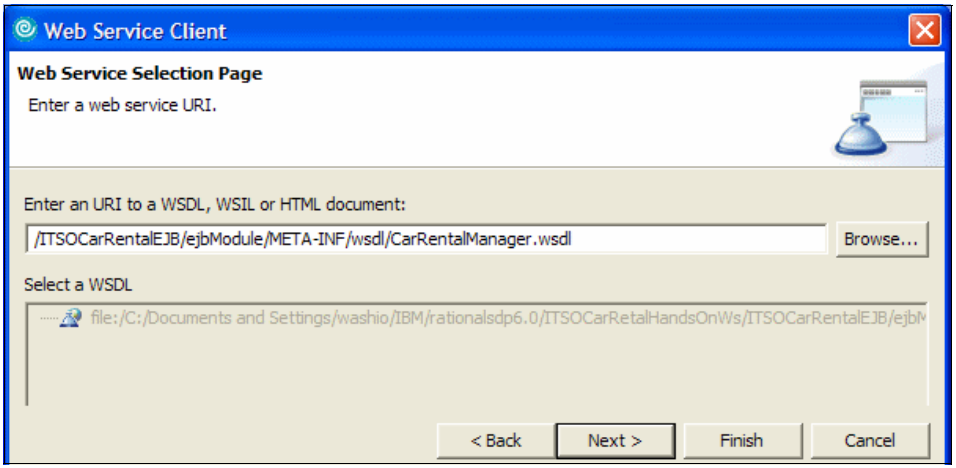


Figure 8-38 Web Service Selection

5. When the Client Environment Configuration dialog box opens, do the following (as seen in Figure 8-39 on page 258) and then click **Next**:
   - Client type: select **Web**
   - Client project: select **ITSOCarRentalWSWeb**
   - EAR project: **ITSOCarRentalWS**

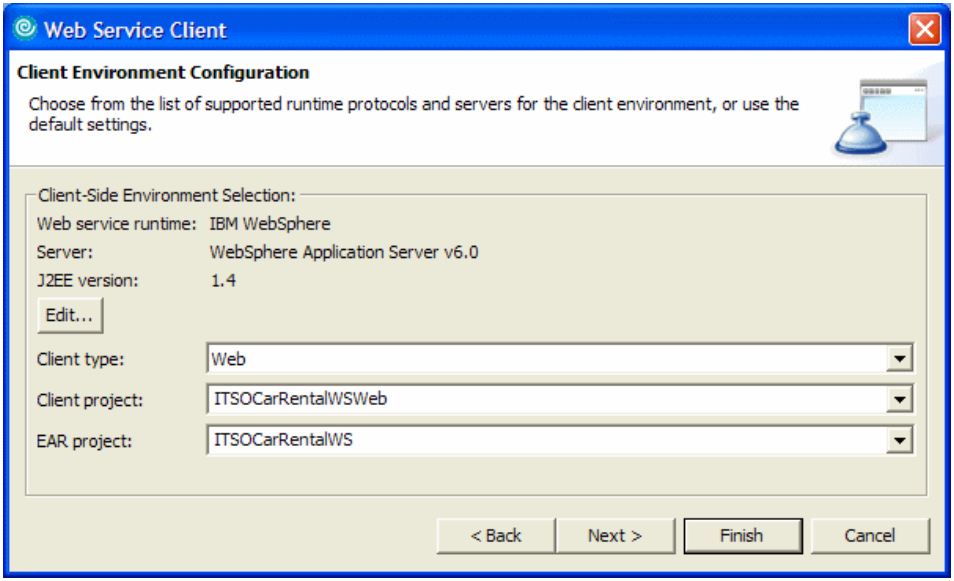


Figure 8-39 Client Environment Configuration

6. When the Web Service Proxy dialog box opens, accept the default settings (as seen in Figure 8-40) and click **Finish**.

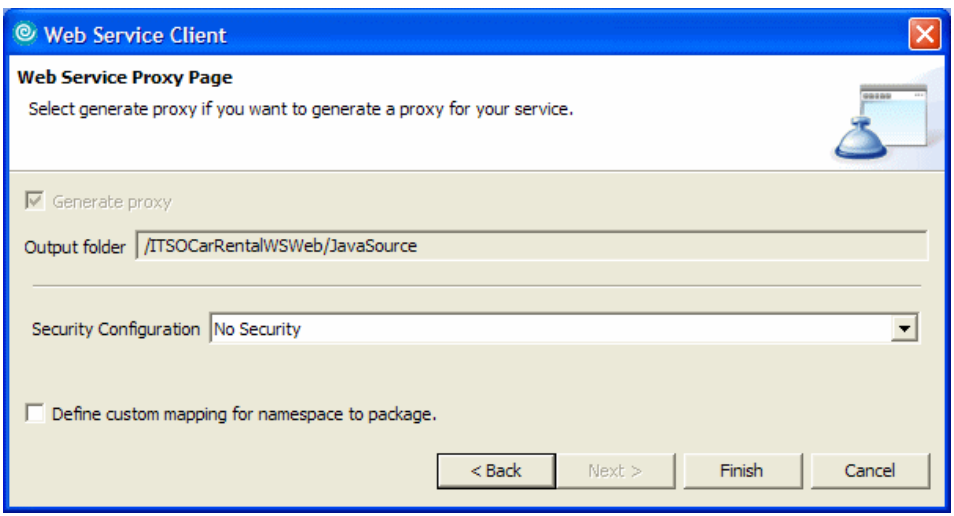


Figure 8-40 Web Service Proxy Page

7. When prompted with the Warning dialog box as seen in Figure 8-41 on page 259, click **Yes**.

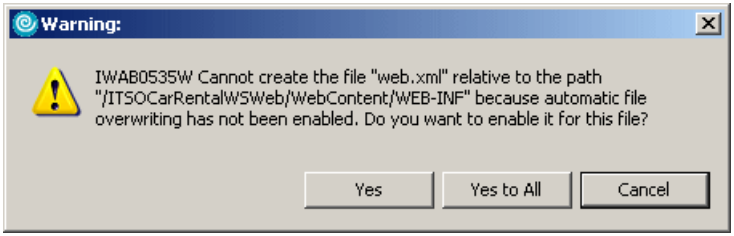


Figure 8-41 Warning dialog

8.6.2 Modify Web application to invoke the Web service

Now that the Web Service client code has been generated, we are ready to modify the ITSO Car Rental Web application to invoke the Web Service from the generated Web Services client. We use the existing ITSO Car Rental Web application as a base for our changes.

Note: The completed source code files for this section are provided in following sample code directory:

`C:\7240-SOA\7240code\assemble\intermediatefiles\internal_ws_client`

Copy existing ITSO Car Rental Web application

Copy the existing ITSO Car Rental Web application assets to the Web Service client enabled Web application.

1. Copy the **com.ibm.itso.carrental.Utils** and **com.ibm.itso.carrental.web** packages from the ITSOCarRentalWeb project to ITSOCarRentalWSWeb project.
   a. Expand **Dynamic Web Projects** → **ITSOCarRentalWeb** → **Java Resources** → **JavaSource** on Project Explorer view.
   b. Select **com.ibm.itso.carrental.Utils** and **com.ibm.itso.carrental.web** packages, and right-click and select **Copy**.
   c. Expand **Dynamic Web Projects** → **ITSOCarRentalWSWeb** → **Java Resources** on Project Explorer view.
   d. Right-click **JavaSource** and select **Paste**.
2. Copy the JSP files from the ITSOCarRentalWeb project to the ITSOCarRentalWSWeb project.
   a. Expand **Dynamic Web Projects** → **ITSOCarRentalWeb** → **WebContent**.

b. Select the JSPs (login.jsp, receipt.jsp, result.jsp, reviewandconfirm.jsp, search.jsp), right-click and select **Copy**.

c. Expand **Dynamic Web Projects** → **ITSOCarRentalWSWeb** → **WebContent**.

d. Right-click **WebContent** and select **Paste**.

e. To distinguish from the original application, change the `<H1>` tag content to `ITSO Car Rental (Web Service)` in each JSP.

3. Copy the **itso_logo.gif** and **Master.css** under the theme folder from ITSOCarRentalWeb project to ITSOCarRentalWSWeb project. Since there is already Master.css on the ITSOCarRentalWSWeb project, it should be overwritten.

4. Add the <servlet> and <servlet-mapping> elements to the ITSOCarRentalWSWeb project web.xml to utilize the existing servlets.

   a. Expand **Dynamic Web Projects** → **ITSOCarRentalWSWeb** → **WebContent** → **WEB-INF**.

   b. Double-click **web.xml** to open in the Web Deployment Descriptor Editor.

   c. Click the **Source** tab.

   d. Insert the <servlet> and <servlet-mapping> elements listed in Example 8-4 before the <welcome-file-list> element.

   > **Note:** We recommend that you copy the <servlet> elements and <servlet-mapping> elements from ITSOCarRentalWeb project web.xml.

   e. Save the modified web.xml.

Example 8-4 Modified web.xml snippet

```
...
<servlet>
      <description>
      </description>
      <display-name>
      LoginServlet</display-name>
      <servlet-name>LoginServlet</servlet-name>
      <servlet-class>
      com.ibm.itso.carrental.web.LoginServlet</servlet-class>
   </servlet>
   <servlet>
      <description>
      </description>
      <display-name>
      SearchServlet</display-name>
```

```
    <servlet-name>SearchServlet</servlet-name>
    <servlet-class>
    com.ibm.itso.carrental.web.SearchServlet</servlet-class>
</servlet>
<servlet>
    <description>
    </description>
    <display-name>
    CheckAvailabilityServlet</display-name>
    <servlet-name>CheckAvailabilityServlet</servlet-name>
    <servlet-class>
    com.ibm.itso.carrental.web.CheckAvailabilityServlet</servlet-class>
</servlet>

<servlet>
    <description>
    </description>
    <display-name>
    ReservationServlet</display-name>
    <servlet-name>ReservationServlet</servlet-name>
    <servlet-class>
    com.ibm.itso.carrental.web.ReservationServlet</servlet-class>
</servlet>
<servlet>
    <description>
    </description>
    <display-name>
    LogoutServlet</display-name>
    <servlet-name>LogoutServlet</servlet-name>
    <servlet-class>
    com.ibm.itso.carrental.web.LogoutServlet</servlet-class>
</servlet>
<servlet>
    <description>
    </description>
    <display-name>
    AdminLoginServlet</display-name>
    <servlet-name>AdminLoginServlet</servlet-name>
    <servlet-class>
    com.ibm.itso.carrental.web.AdminLoginServlet</servlet-class>
</servlet>
<servlet>
    <description>
    </description>
    <display-name>
    AdminShowReservationsServlet</display-name>
    <servlet-name>AdminShowReservationsServlet</servlet-name>
    <servlet-class>
    com.ibm.itso.carrental.web.AdminShowReservationsServlet</servlet-class>
```

```
   </servlet>
   <servlet>
      <description>
      </description>
      <display-name>
      AdminLogoutServlet</display-name>
      <servlet-name>AdminLogoutServlet</servlet-name>
      <servlet-class>
      com.ibm.itso.carrental.web.AdminLogoutServlet</servlet-class>
   </servlet>
   <servlet>
      <display-name>
      Web Services Router Servlet</display-name>
      <servlet-name>CarRentalManager</servlet-name>
      <servlet-class>

com.ibm.ws.webservices.engine.transport.http.WebServicesServlet</servlet-class>
   </servlet>

<servlet-mapping>
      <servlet-name>LoginServlet</servlet-name>
      <url-pattern>/LoginServlet</url-pattern>
   </servlet-mapping>
   <servlet-mapping>
      <servlet-name>SearchServlet</servlet-name>
      <url-pattern>/SearchServlet</url-pattern>
   </servlet-mapping>
   <servlet-mapping>
      <servlet-name>CheckAvailabilityServlet</servlet-name>
      <url-pattern>/CheckAvailabilityServlet</url-pattern>
   </servlet-mapping>
   <servlet-mapping>
      <servlet-name>ReservationServlet</servlet-name>
      <url-pattern>/ReservationServlet</url-pattern>
   </servlet-mapping>
   <servlet-mapping>
      <servlet-name>LogoutServlet</servlet-name>
      <url-pattern>/LogoutServlet</url-pattern>
   </servlet-mapping>
   <servlet-mapping>
      <servlet-name>AdminLoginServlet</servlet-name>
      <url-pattern>/AdminLoginServlet</url-pattern>
   </servlet-mapping>
   <servlet-mapping>
      <servlet-name>AdminShowReservationsServlet</servlet-name>
      <url-pattern>/AdminShowReservationsServlet</url-pattern>
   </servlet-mapping>
```

```
   <servlet-mapping>
      <servlet-name>AdminLogoutServlet</servlet-name>
      <url-pattern>/AdminLogoutServlet</url-pattern>
   </servlet-mapping>
```

Modify the Java source code to invoke Web Service

In the original ITSO Car Rental Web application, there are three servlets that access the database through the session bean, and utility class that looks up the EJB. We will modify the servlets to use the generated Web services client to invoke the ITSO Car Rental Web Services.

1. Modify ITSOCarRentalUtils.java.
   a. Expand **Dynamic Web Projects** → **ITSOCarRentalWSWeb** → **Java Resources** → **JavaSource** → **com.ibm.itso.carrental.Utils** from the Project Explorer view.
   b. Double-click **ITSOCarRentalUtils.java** to open in the Java Editor.
   c. Modify the ITSOCarRentalUtils.java as in Example 8-6. We have included the original ITSOCarRentalUtils.java in Example 8-5 for reference.
   d. Save the modified ITSOCarRentalUtils.java.

Note: The modified ITSOCarRentalUtils.java source file can be found in the following sample code directory:

`C:\7240-SOA\7240code\assemble\intermediatefiles\internal_ws_client\com.ibm.itso.carrental.Utils`

Example 8-5 Original ITSOCarRentalUtils.java snippet

```
private static CarRentalManagerHome carRentalMgr = null;

static {
   try {
      Context ctx = new InitialContext();
      carRentalMgr = (CarRentalManagerHome)ctx.
         lookup("java:comp/env/ejb/CarRentalManager");
   }
   catch (Exception e){
      e.printStackTrace();
   }
}

public static CarRentalManagerHome getcarRentalMgr(){
   return carRentalMgr;
}
```

Example 8-6 Modified ITSOCarRentalUtils.java snippet

```
private static CarRentalManagerService carRentalMgrService = null;

static {
   try {
      Context ctx = new InitialContext();
      carRentalMgrService = (CarRentalManagerService)ctx.
         lookup("java:comp/env/service/CarRentalManagerService");
   }
   catch (Exception e){
      e.printStackTrace();
   }
}
   public static CarRentalManager getcarRentalMgr(){
   CarRentalManager carRentalManager = null;
   try{
      carRentalManager = carRentalMgrService.getCarRentalManager();
   }catch(Exception e){
      e.printStackTrace();
   }
   return carRentalManager;
}
```

2. Modify **SearchServlet.java**.

   a. Expand **Dynamic Web Projects -> ITSOCarRentalWSWeb -> Java Resources -> JavaSource -> com.ibm.itso.carrental.web** from the Project Explorer view.

   b. Double-click **SearchServlet.java** to open in the Java Editor.

   c. Modify the SearchServlet.java as seen in Example 8-8.

      We have included the original SearchServlet.java in Example 8-7 for reference purposes.

   d. Save the modified SearchServlet.java.

Note: The modified SearchServlet.java source file can be found in the following sample code directory:

`C:\7240-SOA\7240code\assemble\intermediatefiles\internal_ws_client\com.ibm.itso.carrental.web`

Example 8-7 Original SearchServlet.java snippet

```
   try {
      CarRentalManagerHome home = ITSOCarRentalUtils.getcarRentalMgr();
      CarRentalManager crm = home.create();
      vehicles = crm
```

```
         .showAllVehicles(pickUp, location, dropOff, dropOffLocatoin,
            null, null, null);
      Thread.sleep(2500);
   } catch (Exception e){
      e.printStackTrace();
   }
```

Example 8-8 Modified SearchServlet.java snippet

```
Calendar pickUpCal = Calendar.getInstance();
Calendar dropOffCal = Calendar.getInstance();
pickUpCal.setTime(pickUp);
dropOffCal.setTime(dropOff);
...
try {
   CarRentalManager carRentalManager = ITSOCarRentalUtils.getcarRentalMgr();
   vehicles = carRentalManager.showAllVehicles(pickUpCal, location,
dropOffCal, dropOffLocatoin,null, null, null);
   Thread.sleep(2500);
}catch (Exception e){
   e.printStackTrace();
}
```

Note: The `java.util.Date` class maps to `xsd:dateTime` of XML schema, however `xsd:dateTime` maps back to `java.util.Calendar`. The service method generated from the WSDL file contains Calendar in it's arguments, though it was originally Date.

To address this issue within our sample, we converted the Date instances to Calender instances before passing them to the method (see Example 8-8).

3. Modify CheckAvailabilityServlet.java and ReservationServlet.java to address the java.util.Date class maps to xsd:dateTime issue described in the previous step.
4. Modify ReservationServlet.java to insert the following variable.

   ```
   String companyID = "INT_ITSO_01";
   ```

   Note: The modified CheckAvailabilityServlet.java and ReservationServlet.java files can be found in the following sample code directory:

   `C:\7240-SOA\7240code\assemble\intermediatefiles\internal_ws_client\com.ibm.itso.carrental.web`

5. Verify the application has been modified correctly.
   a. Ensure the test server is started.
   b. Enter the following URL in a Web browser:

      `http://localhost:9080/ITSOCarRentalWSWeb/login.jsp`

   c. Verify the ITSO Car Rental application is working properly. In this case, the Web Services client invokes the ITSO Car Rental application Web services.

8.6.3 Add ability to view car rental channel statistics

This section describes how to add the ability for an administrator user of the ITSO Car Rental Company to observe channel statistics, such as how many reservations were created by each channel (external travel partners).

> **Note:** This section requires that you have completed 8.5.1, “Modify application to capture reservation channel” on page 238.

To add the ability to view car rental channel statistics, follow these steps:

1. Add the following servlets to the `com.ibm.itso.carrental.web` package.

   ```
   AdminLoginServlet.java
   AdminShowReservationsServlet.java
   AdminLogoutServlet.java
   ```

   > **Note:** The completed servlets can be found in the following sample code directory:
   >
   > `C:\7240-SOA\7240code\assemble\intermediatefiles\internal_ws_client\com.ibm.itso.carrental.web`

2. Add the following JSPs to the WebContent folder.

   ```
   adminlogin.jsp
   adminmenu.jsp
   adminreservations.jsp
   ```

   > **Note:** The completed servlets can be found in the following sample code directory:
   >
   > `C:\7240-SOA\7240code\assemble\intermediatefiles\internal_ws_client\jsp`

3. Modify web.xml to utilize the newly added servlets.

Note: The completed servlets can be found in the web.xml in the following sample code directory:

`C:\7240-SOA\7240code\assemble\intermediatefiles\internal_ws_client`

4. Verify the application is working properly:
   a. Enter the following URL in a Web browser:

      `http://localhost:9080/ITSOCarRentalWSWeb/adminlogin.jsp`

   b. Refer to “Verify UC8: View Reservation Activity” on page 392 for details on verifying the functionality of the application.

8.7 Create the external Web Service client application

This section describes how to create an external Web services client application used by travel partners to invoke the ITSO Car Rental Web Services. The internal Web Services client described in 8.6, “Create the internal Web Service client application” on page 254, uses all services available, including those only available to internal operations of the ITSO Car Rental company, such as showAllReservations. In the case of the external Web Services client, only selected operations are permitted based on the exposed services found in the WSDL file.

Complete the following tasks:

- Modify WSDL file for external Web Services client
- Create a new project for the external Web Services client
- Generate the Web Services client code
- Copy ITSOCarRentalWS to GlobalTravels
- Modify the Global Travels Web Services client application

Note: Keep in mind, for our sample we are fulfilling the role of developer for both the ITSO Car Rental company and the Global Travels company. In a real-world scenario, it is likely two different development teams would be developing the application.

In a production environment, the exposed WSDL file is created from the Web Services Gateway. We demonstrate how to generate the Web Services client code using the exposed WSDL file from the Web Services Gateway as part of our deployment procedure in 10.3.1, “Reassemble Global Travels application using Gateway WSDL” on page 373.

8.7.1 Modify WSDL file for external Web Services client

For development and testing purposes, we will manually modify the WSDL file used by the Global Travels Web Services client application.

1. Copy the CarRentalManager.wsdl to CarRentalManagerExt.wsdl.
   a. Expand **EJB Projects** → **ITSOCarRentalEJB** → **ejbModule** → **META-INF** → **wsdl**.
   b. Right-click **CarRentalManager.wsdl** and select **Copy**.
   c. Right-click the **wsdl** folder, and select **Paste**.
   d. When prompted for the new file name, enter `CarRentalManagerExt.wsdl` and then click **OK**.
2. Modify the WSDL file used for the external Web Services client to only include selected operations available to invoke the Web Services.
   a. Expand **EJB Projects** → **ITSOCarRentalEJB** → **ejbModule** → **META-INF** → **wsdl**.
   b. Double-click **CarRentalManagerExt.wsdl** to open in the WSDL Editor.
   c. Expand the **Bindings**, then right-click **showAllReservations** and select **Delete** as seen in Figure 8-42.

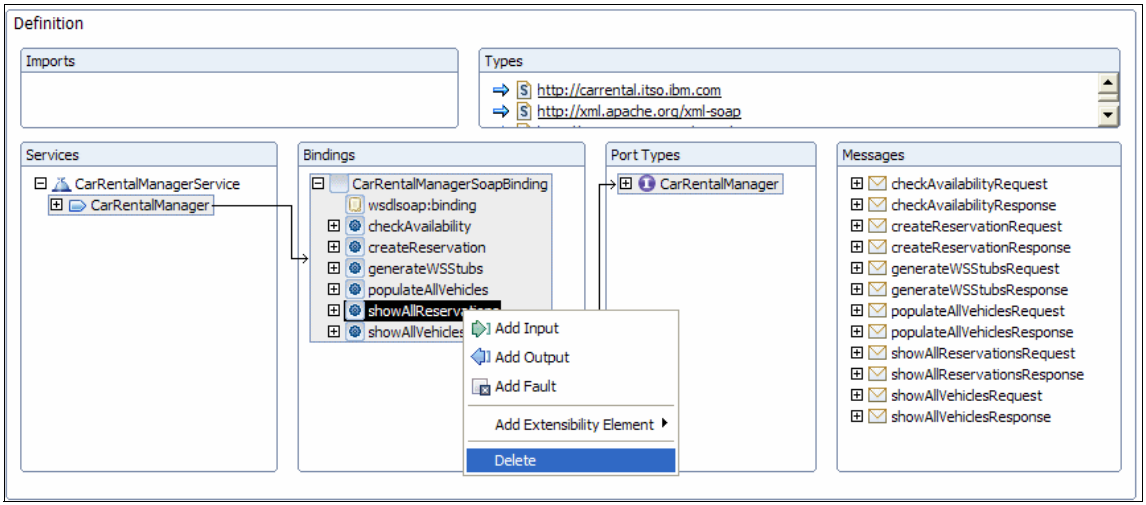


Figure 8-42 CarRentalManagerExt.wsdl displayed in the Graph view of the WSDL Editor

   d. From Bindings, right-click **populateAllVehicles** and select **Delete**.
   e. Expand **Port Type**, right-click **showAllReservations** and select **Delete**.

f. When prompted to Delete associated Messages and Parts (leave checked) and click **OK**.

g. From Ports, right-click **populateAllVehicles** and select **Delete**.

h. When prompted to Delete associated Messages and Parts (leave checked) and click **OK**.

i. Select **File** → **Save** (CarRentalManagerExt.wsdl).

8.7.2 Create a new project for the external Web Services client

To create a new project for the external Web Services client application, follow these steps:

1. Select **File** → **New** → **Enterprise Application Project** on the Workbench.
2. When the Enterprise Application Project dialog appears, enter `GlobalTravels` in the Name field, and then click **Next**.
3. When the EAR Module Projects dialog appears, click **New Module**.
4. When the New Module Project dialog appears, do the following and then click **Finish**:
   - Check **Create default module projects**
   - Deselect **Application Client project**
   - Deselect **EJB project**
   - Check **Web project**
   - Deselect **Connector project**
5. When returning to the EAR Module Projects, ensure only **GlobalTravelsWeb** is check, and then click **Finish**.

8.7.3 Generate the Web Services client code

To generate Web Service client code for the external Web project, do as follows:

1. Expand **EJB Projects** → **ITSOCarRentalEJB** → **ejbModule** → **META-INF** → **wsdl**.
2. Right-click **CarRantalManagerExt.wsdl** created in above, select **Web Services** → **Generate Client**.
3. When the Web Services dialog appears, select **Java proxy** and click **Next**.
4. When the Web Service Selection dialog appears, accept the default settings and click **Next**.

5. When the Client Environment Configuration dialog appears, do the following and then click **Next**:
   - Client type: select **Web**
   - Client project: select **GlobalTravelsWeb**
   - EAR project: select **GlobalTravels**
6. When the Web Service Proxy dialog box opens, accept the default settings and click **Finish**.
7. When the warning dialog box opens, click **Yes**.

8.7.4 Copy ITSOCarRentalWS to GlobalTravels

Now that the Web Service client code has been generated, we will develop the web application that utilizes the Web Service. We can reuse most of the source code from ITSOCarRentalWS for GlobalTravels.

Copy com.ibm.itso.carrental.Utils and com.ibm.itso.carrental.web packages from ITSOCarRentalWSWeb project to GlobalTravelsWeb project.

1. Expand **Dynamic Web Projects** → **ITSOCarRentalWSWeb** → **Java Resources** → **JavaSource** on Project Explorer view.
2. Select the **com.ibm.itso.carrental.Utils** and **com.ibm.itso.carrental.web** packages, right-click and select **Copy**.
3. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **Java Resources** on Project Explorer view.
4. Right-click **JavaSource** and select **Paste**.

> **Note:** We will change these package names later using the refactor feature provided with Rational Application Developer. Refactor can be used to change references in the Java files and JSP files in the same activity.

8.7.5 Modify the Global Travels Web Services client application

This section describes how to modify the Global Travels Web Services client application.

> **Note:** The completed files can be found in the following sample code directory:
>
> `C:\7240-SOA\7240code\assemble\intermediatefiles\external_ws_client`

1. Delete the servlets found in the com.ibm.itso.carrental.web package that have the prefix `Admin` from the GlobalTravelsWeb project.
   a. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **Java Resources** → **JavaSource** → **com.ibm.itso.carrental.web**.
   b. Select the servlets with the prefix `Admin`, right-click and select **Delete**.
2. Change the companyID in ReservationServlet.java to **"**`EXT_GLOBTRAV_01`**"**. This is used to populate the companyID column in the RESERVATION table when a reservation is created.
3. Import the GlobalTravel.gif.
   a. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **WebContent** → **theme**.
   b. Right-click **theme**, select **Import**.
   c. Select **File System** and click **Next**.
   d. Enter the path to the gif file. For example, we entered the following directory:

      ```
      C:\7240-SOA\7240code\assemble\intermediatefiles\external_ws_client\theme
      ```

   e. Check **GlobalTravel.gif** and **Master.css**, and then click **Finish**.
   f. Click **Yes** to overwrite.
4. Copy the JSP files, except those with admin prefix, from the WebContent folder of the ITSOCarRentalWSWeb project to the WebContent folder of the GlobalTravelsWeb project.
5. Modify the JSPs to distinguish from the original application, change <TITLE> or <H1> tag content to like `Global Travel Company` and GlobalTravel.gif logo in each JSP.
6. Refactor (rename) the package names.
   a. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **Java Resources** → **JavaSource.**
   b. Right-click **com.ibm.itso.carrental.Utils**, select **Refactor** → **Rename.**
   c. When the Rename Package dialog box opens, do the following:
      - New name: `com.ibm.itso.globaltravels.Utils`
      - Check **Update references**
      - Check **Update textual matches in command and strings**
      - Click **Preview**
   d. When the Changes to be performed dialog box opens, click **OK**.
   e. Rename com.ibm.itso.carrental.web to `com.ibm.itso.globaltravels.web`.
7. Modify web.xml for GlobalTravelsWeb to utilize the servlets.

Note: The completed web.xml file can be found in the following sample code directory:

`C:\7240-SOA\7240code\assemble\intermediatefiles\external_ws_client`

8. Verify the application is working properly.
   a. Enter the following URL in a Web browser:

      `http://localhost:9080/GlobalTravelsWeb/login.jsp`

   b. Verify the Global Travels application.

Note: After a reservation is created through the GlobalTravel, you can see the recode in the RESERVATION table has “EXT_GLOBTRAV_01” in the companyID column. Also, you can see it through the administration page as seen in Figure 11-10 on page 395.

8.8 Export the EAR files for deployment

This section describes how to export the EAR files containing the ITSO Car Rental Web services enabled application, and the Global Travels Web Services client application. The EAR files are used to deploy the applications to WebSphere Application Server.

To export an EAR file, follow these steps:

1. Expand **Enterprise Applications** from the Project Explorer view.
2. Right-click **ITSOCarRental** project and select **Export** → **EAR file**.
3. Enter the destination (for example: `c:/temp/ITSOCarRental.ear`), and then click **Finish**.
4. In the same way, export ITSOCarRentalWS project and GlobalTravels project as well.

Note: The EAR files are provided in following directory of the sample code:

```
C:\7240-SOA\7240code\deploy\ITSOCarRental.ws
C:\7240-SOA\7240code\deploy\GlobalTravels.ws
```

8.9 Develop a Java client application to generate traffic

This section describes how to develop a Java console (command line interface) Web services client application to invoke Web services to generate traffic for testing purposes.

In our example, we create a Java console application that invokes the ITSO Car Rental application Web services iteratively to generate traffic. The client application can be used in conjunction with IBM Tivoli Composite Application Manager for SOA for monitoring and performance analysis.

Note: Refer to Chapter 12, "Manage and monitor services with ITCAM for SOA" on page 405 for more information on IBM Tivoli Composite Application Manager for SOA.

The completed version of the sample described in this section can be found in the following sample code directory:

```
C:\7240code\assemble\ITSOCarRentalLoadTest
```

For more information refer to Appendix E, "Additional material" on page 515.

8.9.1 Create the Java console WS client application

This section describes how to create the command line driven Java console Web services client application within Rational Application Developer.

Complete the following tasks:

- Create projects for the application client
- Generate client code
- Create projects for the application client

Create projects for the application client

To create a new Enterprise Application project, do the following:

1. Open the J2EE perspective.
2. Select **File** → **New** → **Enterprise Application Project**.
3. When the Enterprise Application Project dialog appears, enter `ITSOCarRentalLoader` in the Name file, and then click **Next**.
4. When the EAR Module Projects dialog appears, click **New Module**.

5. When the New Module Project dialog appears, do the following and then click **Finish**:
   - Deselect **EJB project**
   - Deselect **Web project**
   - Deselect **Connector project**

Generate client code

Do the following to generate the client code:

1. Expand **EJBProject** → **ITSOCarRentalEJB** → **ejbModule** → **META_INF** → **wsdl** on the Project Explorer view.
2. Right-click **CarRentalManager.wsdl** and select **WebServices** → **Generate Client**.
3. When the Web Services dialog box opens, accept the default settings, and then click **Next**.
4. When the Web Service Selection dialog box opens, we accepted the default setting, and clicked **Next**.
5. When the Client Environment Configuration dialog box opens, do the following (as seen in Figure 8-43 on page 275) and then click **Next**:
   - Client type: select **Application Client**
   - Client project: select **ITSOCarRentalLoaderClient**
   - EAR project: **ITSOCarRentalLoader**

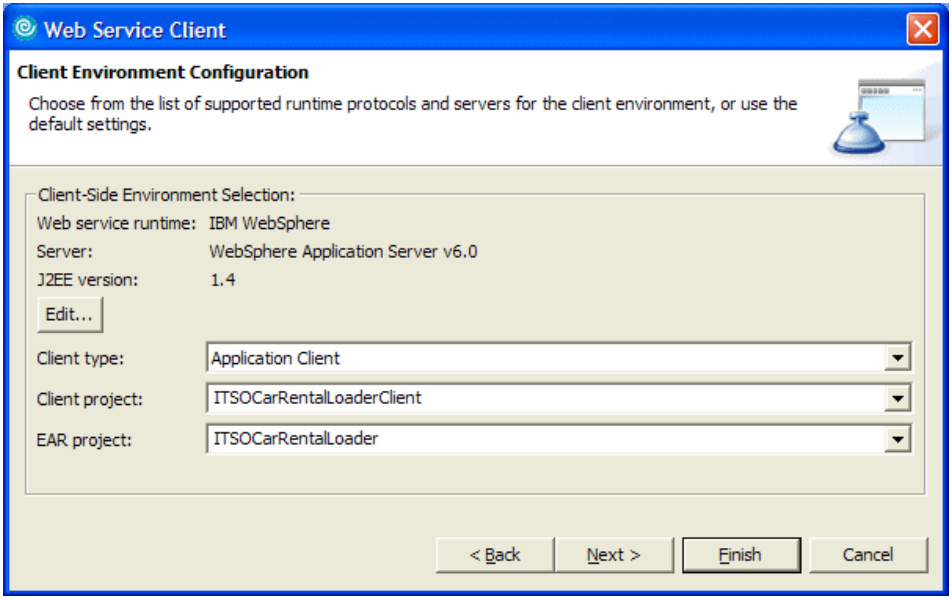


Figure 8-43 Client Environment Configuration

6. When the Web Service Proxy dialog box opens, accept the default settings and click **Finish**.

Modify application client to invoke Web service

After generating the client code, you can see Main.java which has main method under the ITSOCarRentalLoaderClient project. We implement logic to invoke the Web service in the Main.java.

1. Expand **Application Client Projects** → **ITSOCarRentalLoaderClient** → **appClientModule** → **(default package)** on the Project Explorer view.
2. Open the **Main.java**.
3. Add the following code as shown in Example 8-9 to the Main.java

Example 8-9 Main.java

```
import java.util.*;
import com.ibm.itso.carrental.*;

public class Main {
   public static void main(String[] args) {

      int count = Integer.parseInt(args[0]);
      String host = null;
      if(args.length==2)
```

```
            host = args[1];

        CarRentalManagerProxy proxy = new CarRentalManagerProxy(); //...(1)
        if(host!=null){
            String[] temp = proxy.getEndpoint().split("localhost:9080");
            String endpoint = temp[0] + host + temp[1];
            proxy.setEndpoint(endpoint);
            System.out.println("Endpoint has been overwritten : "
                + proxy.getEndpoint());
        }
        CarRentalManager mgr = proxy.getCarRentalManager(); //...(2)

        try {
            Calendar pickup = Calendar.getInstance();
            Calendar dropoff = Calendar.getInstance();
            pickup.setTime(new Date());
            dropoff.setTime(new Date());
            dropoff.set(Calendar.DATE, pickup.get(Calendar.DATE)+7);

            for(int i=0; i<count; i++){
                mgr.checkAvailability(1001, pickup, "Raleigh",
                    dropoff, "Durham"); //...(3)
                mgr.showAllVehicles(pickup,"Raleigh", dropoff,
                    "Durham", "Midsize", null, null); //...(4)
                mgr.createReservation(1001, "TEST_LOADER_ID"); //...(5)
            }
            System.out.println("Done : count = " + count);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
...
```

Note: At the (1) and (2) in the example, the client configures parameters for the Web Service such as the target URL. At the (3), (4) and (5), the client invokes the Web service operations respectively.

By default, the target hostname is hard coded as `localhost:9080` in CarRentalManagerServiceLocator class. You can overwrite the hostname in the class directly, or you can use a method, setEndpoint(String), in the CarRentalManagerProxy, which overwrites the target URL including the hostname. In our example, the URL is overwritten by the setEndpoint method with the second argument.

4. Save the modified Main.java.

8.9.2 Run the client application

The client application is dependent on several classes included in JAR files found in the WebSphere Application Server lib directory. This section will demonstrate how to run the command line driven client application on the WebSphere Application Server Test Environment included within Rational Application Developer, as well as run on an external WebSphere Application Server. In both cases, we will make use of the launchClient command provided with WebSphere Application Server to set the environment.

Run client application within Rational Application Developer

To run the command line driven client application within the WebSphere Application Server Test Environment included with Rational Application Developer, follow these steps:

1. Ensure the test server is started and ITSOCarRental and ITSOCarRentalWS application are started.
2. Select **Run** → **Run...** on the workbench menu.
3. When the Run diagram box opens, select **WebSphere v6.0 Application Client**, then click **New**.
4. When the Application tab appears on the diagram, do the following (as seen in Figure 8-44 on page 278) and then click **Arguments** tab:
   - Name: input **ITSOCarRentaLoader** (arbitrary)
   - WebSphere Runtime: select **WebSphere Application Server v6.0**
   - Enterprise application: **ITSOCarRentalLoader**
   - Application client module: **ITSOCarRentalLoaderClient**

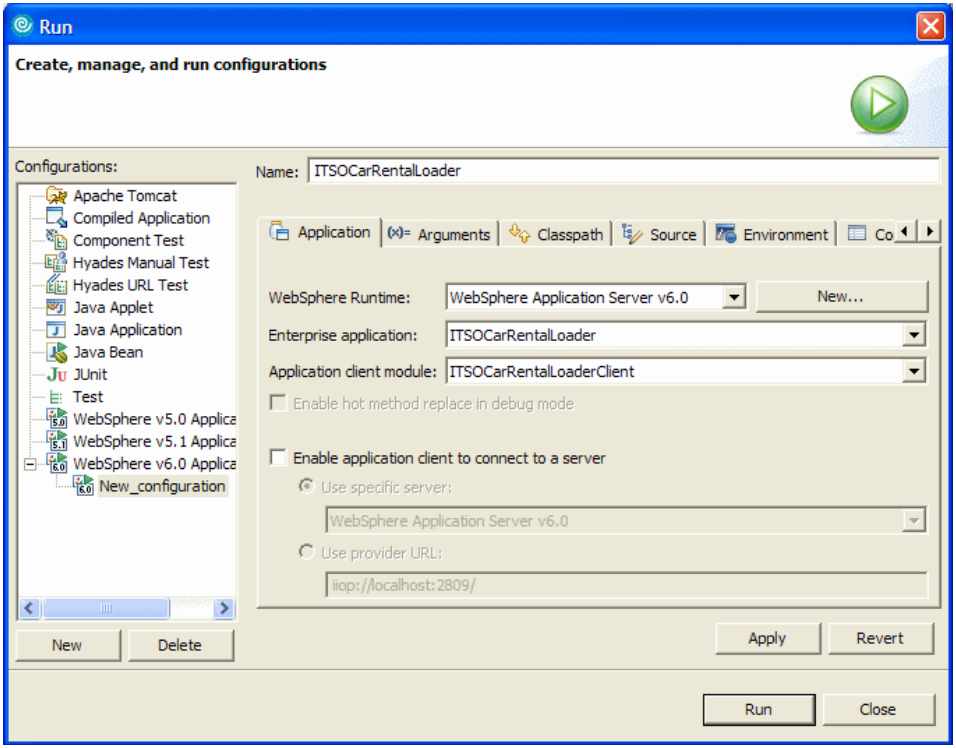


Figure 8-44 Run - Application tab

5. On the **Arguments** tab, put two arguments for the number of iteration (for the For-clause in Main.java) and hostname with port number at the end of Program arguments. When the hostname should be localhost:9080, the hostname does not need to be designated. For example we entered 10, which will make 10 reservations to localhost. Then, click **Run**. See Figure 8-45 on page 279.

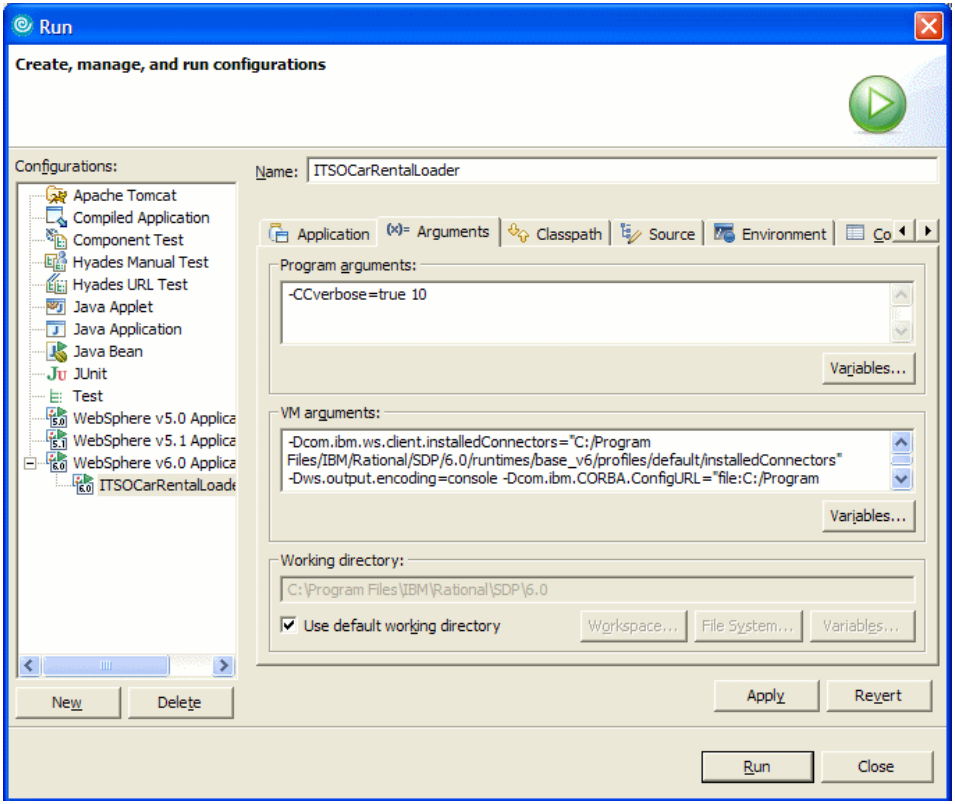


Figure 8-45 Run - Arguments tab

6. To verify the application is working properly, Enter the following URL in a Web browser and go to the reservation list page. You can see reservations have “TEST_LOADER_ID” in the companyID column.

   `http://localhost:9080/ITSOCarRentalWSWeb/adminlogin.jsp`

Export the application client

If you want to run the client application on WebSphere Application Server, you must first export an EAR file for deployment. To export an EAR file, follow these steps:

1. Expand **Enterprise Applications** from the Project Explorer view.
2. Right-click **ITSOCarRentalLoader** project and select **Export** → **EAR file**.
3. Enter the destination (for example: `c:/temp/loader.ear`), and then click **Finish**.

Run on WebSphere Application Server

To run the client application on a WebSphere Application Server, follow these steps:

1. Copy the EAR file (the `loader.ear` in the previous step) to the node where WebSphere Application Server is installed that you wish to run the client application. For example, we copied the `loader.ear` to the `C:\temp` directory.
2. Open a Windows command window and navigate to the following directory:

   ```
   C:\IBM\WebSphere\AppServer\bin
   ```

3. Run the client application using the launchClient environment provided with WebSphere Application Server:

   Syntax:

   ```
   launchClient <ear_file_path> <iterate_count> <hostname:port>
   ```

   Example:

   ```
   launchClient C:\temp\loader.ear 10 itsoapp1.ral.ibm.com:9080
   ```

8.10 Where to find more information

This section includes some key references for developing Web Services applications:

- IBM Redbooks:
  - *WebSphere Version 6 Web Services Handbook, Development and Deployment*, SG24-6461
  - *Rational Application Developer V6 Programming Guide*, SG24-6449
- IBM developerWorks:
  - *Web services programming tips and tricks: Roundtrip issues, an introduction* found at:
    http://www.ibm.com/developerworks/webservices/library/ws-tip-roundtrip1.html

9

Implement the runtime environment

This chapter describes how to install and configure the software components for each node of the ITSO working example runtime environment, which demonstrates how to implement a Service Creation solution.

The value-add of the working example runtime implementation is three-fold. First, we provide detailed procedures to install the software by node. Second, we include sample values to help put in context the information found in the product guides. Third, we include best practices and tips.

The chapter is organized as follows:

- Planning
- Implement Web Server Redirector node
- Implement ITSO Application Server node
- Implement Travel Application Server node
- Implement ITSO Monitor Server node

9.1 Planning

This section describes the runtime topology used for the ITSO Car Rental working example. In addition, we have included information on the type of hardware, and software levels we used for our runtime environment.

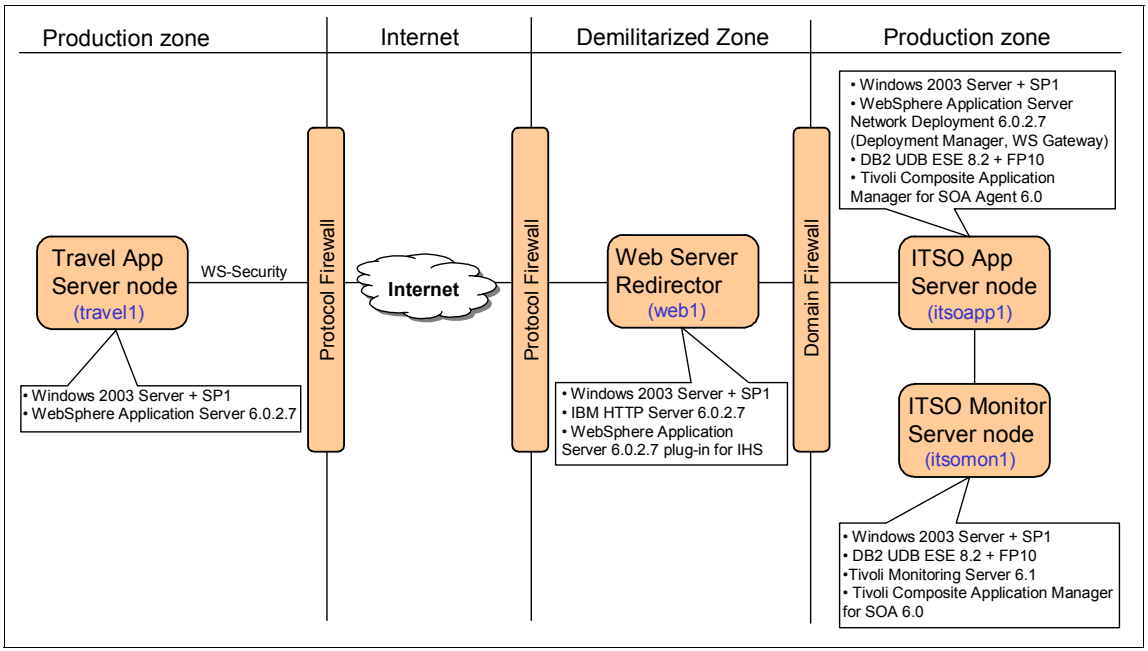


Figure 9-1 ITSO Car Rental working example runtime environment and product mapping

Note: For information on WS-Security, refer to Appendix D, “Web Services Security” on page 505.

The runtime environment consists of four nodes (see Figure 9-1):

- Web Server Redirector node

  This node is a Web Server redirector to the Application Server using the WebSphere plug-in for IBM HTTP Server.

- ITSO Application Server node

  This node includes the following components and functionality:

  – Web Services Gateway used to externalize selected ITSO Car Rental Web Services to external Travel companies.

- Network Deployment Manager used to manage the Web Services Gateway (Dmgr01 profile) and WebSphere Application Server (AppSrv01 profile) resources and applications.
- WebSphere Application Server used to host the ITSO Car Rental application.
- Tivoli Composite Application Manager Agent used to monitor and collect Web Services data, which is transmitted to the ITCAM Server.

- Travel Application Server node

  This node includes WebSphere Application Server and the Travel application, which is a Web services client to the ITSO Car Rental application.

  Note: For simplicity of setting up the environment we did not use a Web Server Redirector for the external Travel Application Server. In a production environment it is likely a redirector would be used.

- ITSO Monitor Server node

  This node is used to monitor the ITSO Car Rental application Web Services using ITCAM for SOA.

9.1.1 Hardware used within ITSO runtime environment

We used the following hardware to implement the nodes of the ITSO working example runtime environment:

- Web Server Redirector node
  - IBM NetVista M42 (8306-E3U)
    - 3 GHz Intel P4 CPU
    - 1GB RAM
    - 30 GB Hard Disk
    - 1 Ethernet Adapter
    - Hostname: `web1.itso.ral.ibm.com`
- ITSO Application Server node
  - IBM xSeries® 225 (8657-5AX)
    - 2.8 GHz Intel Xeon® CPU
    - 2 GB RAM
    - 18 GB Hard Disk
    - 1 Ethernet Adapter
    - Hostname: `itsoapp1.itso.ral.ibm.com`

- ITSO Monitor Server node
  - IBM xSeries 225 (8657-5AX)
    - 2.8 GHz Intel Xeon CPU
    - 2 GB RAM
    - 18 GB Hard Disk
    - 1 Ethernet Adapter
    - Hostname: `itsomon1.itso.ral.ibm.com`
- Travel Application Server node
  - IBM xSeries BladeCenter® HS20
    - 2.4 GHz Intel Xeon CPU
    - 2.5 GB RAM
    - 30 GB Hard Disk
    - 1 Ethernet Adapter
    - Hostname: `travel1.itso.ral.ibm.com`

9.1.2 Software used within ITSO runtime environment

We used the following software to implement the ITSO runtime environment nodes.

Table 9-1 Web Server Redirector node

Software	Version
Microsoft Windows 2003 Server	2003 + Service Pack 1 + Critical fixes
IBM HTTP Server	6.0.2.7 **Note:** 6.0 + Refresh 6.0.2 + Fixpack 6.0.2.7
IBM WebSphere Application Server plug-in for IBM HTTP Server	6.0.2.7 **Note:** 6.0 + Refresh 6.0.2 + Fixpack 6.0.2.7

Table 9-2 ITSO Application Server node

Software	Version
Microsoft Windows 2003 Server	2003 + Service Pack 1 + Critical fixes
IBM WebSphere Application Server Network Deployment Edition * Network Deployment Manager * WebSphere Application Server * Web Services Gateway	6.0.2.7 **Note:** 6.0 + Refresh 6.0.2 + Fixpack 6.0.2.7

Software	Version
IBM DB2 Universal Database Enterprise Server Edition	8.1.10.812 **Note:** 8.2 + Fixpack 10
IBM Tivoli Composite Application Manager Agent	6.0

Table 9-3 Travel Application Server node

Software	Version
Microsoft Windows 2003 Server	2003 + Service Pack 1 + Critical fixes
IBM WebSphere Application Server	6.0.2.7 **Note:** 6.0 + Refresh 6.0.2 + Fixpack 6.0.2.7

Table 9-4 ITSO Monitor Server node

Software	Version
Microsoft Windows 2003 Server	2003 + Service Pack 1 + Critical fixes
IBM WebSphere Application Server	6.0.2.7 **Note:** 6.0 + Refresh 6.0.2 + Fixpack 6.0.2.7
IBM Tivoli Monitoring * IBM Tivoli Enterprise™ Monitor Server * IBM Tivoli Enterprise Portal Server * IBM Tivoli Enterprise Portal Desktop Client	6.1 + FP001
IBM Tivoli Composite Application Manager Server for SOA	6.0

9.2 Implement Web Server Redirector node

The primary software used on the Web Server Redirector node is IBM HTTP Server and the WebSphere Application Server plug-in for IBM HTTP Server. This section describes how to install IBM HTTP Server and configure this node as a host for our application WSDL files.

Complete the following tasks:

1. Microsoft Windows 2003 Server and system preparation
2. IBM HTTP Server installation

Note: We will configure this node to be a remote Web server to the ITSO Application Server node in 9.3.7, “Configure the Application Server with remote Web Server” on page 324.

9.2.1 Microsoft Windows 2003 Server and system preparation

This section describes the steps taken to prepare the Windows 2003 Server.

Microsoft Windows 2003 Server installation

For our example, we installed Microsoft Windows 2003 Server, Standard Edition + Service Pack 1 + critical fixes.

Verify network configuration

Verify that the network configuration for your system is working properly before proceeding. This includes verifying that the hostname and fully qualified hostname can be resolved using **`ping`** and **`nslookup`** command line utilities.

Ensure Microsoft Internet Information Server (IIS) is disabled

Ensure that the Microsoft Internet Information (IIS) is disabled from Windows services to avoid TCP/IP port conflicts with the IBM HTTP Server.

Ensure firewall is disabled during the installation

Prior to installation, ensure that you do not have a firewall enabled to avoid conflicts with TCP/IP ports used by other applications.

Create Windows administrative user ID

During the installation of software, you must be logged in to the system with a Microsoft Windows administrative user ID.

We created an administrative user ID named *admin*.

Assign Windows user right assignments

Ensure the following user right assignments are assigned from within the Windows Local Security Settings for the administrative user ID (for example, admin) used during installation:

- Act as part of the operating system
- Adjust memory quotas for a process
- Create a token object
- Log on as a service
- Replace a process level token

You can access the Windows Local Security Settings by selecting **Start** → **Control Panel** → **Administrative Tools** → **Local Security Policy**. The User Right Assignments can be accessed by expanding **Security Settings** → **Local Policies** → **User Right Assignment**.

Logon as admin user ID during installation

Prior to installation, logon to the system with the administrator ID you have assigned the proper user right assignments (see “Assign Windows user right assignments” on page 286).

9.2.2 IBM HTTP Server installation

For consistency across nodes, we install IBM HTTP Server V6.0, with the V6.0.2 Refresh and V6.0.2.5 Fixpack.

Complete the following tasks:

1. Install IBM HTTP Server V6.0
2. Install IBM HTTP Server V6.0.2 Refresh
3. Install IBM HTTP Server V6.0.2.7 Fixpack

Install IBM HTTP Server V6.0

To install IBM HTTP Server V6.0, follow these steps:

1. Run `launchpad.bat` to start the installer from the root of the WebSphere Application Server CD.

   Note: When installing software on a Microsoft Windows 2003 Server system, you might see a dialog box with the message, `The publisher could not be verified. Are you sure you want to run this software?` Click **Run** to continue.

2. When the Launchpad home dialog box opens, click **IBM HTTP Server Installation**.
3. Click **Launch the installation wizard for IBM HTTP Server**.
4. When the Welcome dialog box opens, click **Next**.
5. When the Software License Agreement dialog box opens, review the terms and if you are in agreement, select **I accept the terms in the license agreement**, then click **Next**.
6. When the Installation directory dialog box opens, we entered `C:\IBM\HTTPServer` in the Directory name field and clicked **Next**.

7. When the Setup type option dialog box opens, we chose **Typical** and clicked **Next**.
8. When the Windows service definition dialog box opens, enter we entered the user name and password to run the IBM HTTP Server process as a Windows service, and select the startup type (manual or automatic) as seen in Figure 9-2, and click **Next**.

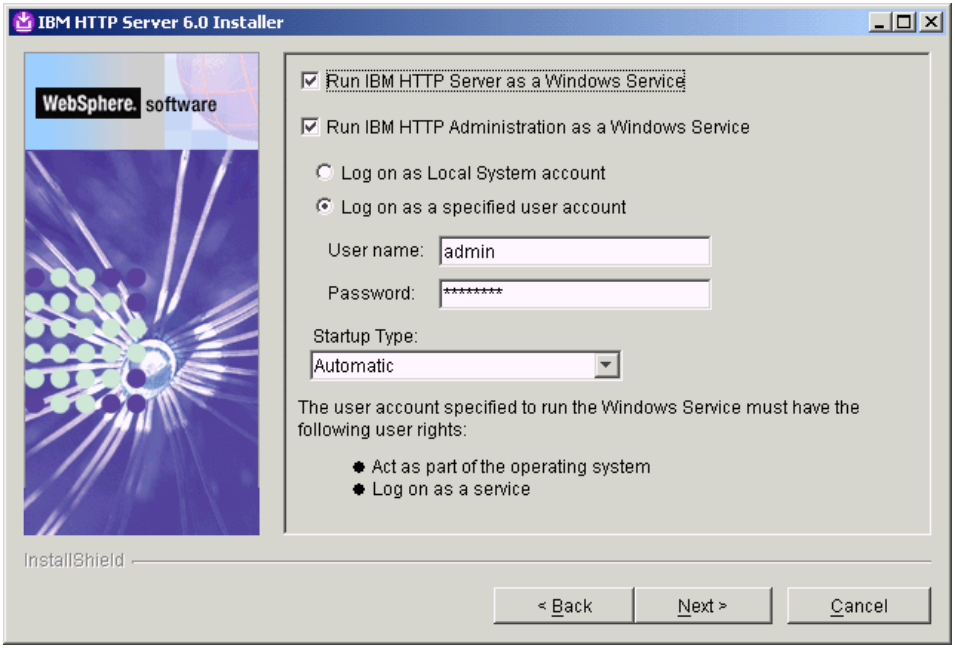


Figure 9-2 Windows service for IBM HTTP Server

9. When the Summary dialog box opens, review the selections and then click **Next**.
10. You should see a Installation successful dialog box. Click **Next**.
11. Deselect the **Launch the WebSphere Application Server - Plugin** install, and click **Finish**.
12. Close the Launchpad Base dialog box.

Install IBM HTTP Server V6.0.2 Refresh

In this section, we list the key options we selected during the IBM HTTP Server V6.0.2 Refresh installation:

1. Ensure the following IBM HTTP Server Windows services are stopped:
   - IBM HTTP Administration 6.0
   - IBM HTTP Server 6.0
2. Download the IBM HTTP Server V6.0.2 Refresh to a temporary directory. The Refresh (Intel IBM HTTP Server) can be found at this Web site:

   `http://www.ibm.com/support/docview.wss?rs=180&uid=swg24009813`
3. Unzip the `6.0-WS-WASIHS-WinX32-RP0000002.zip` to the root of the IBM HTTP Server directory. For example, after the unzip, you will have a directory structure similar to the following:

   `C:\IBM\HTTPServer\updateinstaller`
4. Navigate to the updateinstaller directory (for example, `C:\IBM\IBMHTTPServer\updateinstaller`) and run **`update.exe`** to start the WebSphere Update Installer.
5. We accepted the default options for the remaining installation dialog boxes, unless noted.
6. When prompted, enter the Directory Name for IBM HTTP Server. For example, we entered `C:\IBM\HTTPServer`.
7. Select **Install maintenance package** and then click **Next**.
8. Enter the file name of the maintenance pack and click **Next** (for example, `C:\IBM\HTTPServer\updateinstaller\maintenance\6.0-WS-WASIHS-WinX32-RP0000002.pak`).
9. The maintenance package should be listed. Click **Next** to begin the update.
10. You should see the Upgrade successful dialog box, click **Finish**.

Install IBM HTTP Server V6.0.2.7 Fixpack

In this section, we list the key options we selected during the IBM HTTP Server V6.0.2.7 Fixpack installation:

1. Ensure the following IBM HTTP Server Windows services are stopped:
   - IBM HTTP Administration 6.0
   - IBM HTTP Server 6.0
2. Download the IBM HTTP Server V6.0.2.7 Fixpack to a temporary directory. The Fixpack (Intel IBM HTTP Server) can be found a this Web page:

   `http://www.ibm.com/support/docview.wss?rs=180&uid=swg24011539`
3. Rename the existing updateinstaller directory in the root of the IBM HTTP Server directory (for example, `updateinstaller.602`).

4. Unzip the `6.0-WS-WASIHS-WinX32-FP0000007.zip` to the root of the IBM HTTP Server directory. For example, after the unzip you will have a directory structure like the following:

   `C:\IBM\HTTPServer\updateinstaller`

5. Navigate to the updateinstaller directory (for example, `C:\IBM\IBMHTTPServer\updateinstaller`) and run **`update.exe`** to start the WebSphere Update Installer.
6. We accepted the default options for the remaining installation dialog boxes, unless noted.
7. When prompted, enter the Directory Name for IBM HTTP Server. For example, we entered `C:\IBM\HTTPServer`.
8. Select **Install maintenance package** and then click **Next**.
9. Enter the file name of the maintenance pack and click **Next** (for example, `C:\IBM\HTTPServer\updateinstaller\maintenance\6.0.2-WS-WASIHS-WinX32-FP0000007.pak`).
10. The maintenance package should be listed. Click **Next** to begin the update.
11. You should see the Upgrade successful dialog box, then click **Finish**.

9.2.3 WebSphere Web Server plug-in installation

This section describes how to install the WebSphere Web Server V6.0 plug-in for IBM HTTP Server, and Refresh V6.0.2 and Fixpack V6.0.2.7.

Complete the following tasks:

1. Install WebSphere Web Server V6.0 plug-in
2. Install WebSphere Web Server V6.0.2 plug-in Refresh
3. Install WebSphere Web Server V6.0.2.7 plug-in Fixpack

Install WebSphere Web Server V6.0 plug-in

To install WebSphere Web Server plug-in, follow these steps:

1. Run **`launchpad.bat`** to start the installer from the root of the WebSphere Application Server CD.
2. When the Launchpad home dialog box opens, click **Web Server plug-ins Installation**.
3. Click **Launch the installation wizard for Web Server plug-ins**.
4. When the Welcome dialog box opens, deselect **Installation roadmap** and click **Next**.

5. When the Software License Agreement dialog box opens review the terms and, if you are in agreement, select **I accept the terms in the license agreement**, and then click **Next**.
6. When the System perquisites check dialog box opens, click **Next**.
7. When the Plug-in selection dialog box opens, select **IBM HTTP Server V6** and click **Next**.
8. When the Web Server plug-in scenario (remote or local) dialog box opens, select **Web Server machine (remote)** and click **Next**.
9. When the Installation directory for the Web Server plug-ins dialog box opens, we entered `C:\IBM\WebSphere\Plugins` in the Directory name field and clicked **Next**.
10. When the Web Server configuration file location dialog box opens, we entered `C:\IBM\HTTPServer\conf\httpd.conf` and then clicked **Next**.
11. When the Web Server definition dialog box opens, we accepted the default (webserver1) and clicked **Next**.
12. When the Web Server plug-cfg.xml location dialog box opens, accept the default. We accepted (`C:\IBM\WebSphere\Plugins\config\webserver1\plugin-cfg.xml`) and then clicked **Next**.
13. When the Identify the host name of the Application Server machine is shown, enter the hostname of your ITSO Application server node. We entered `itsoapp1.itso.ral.ibm.com`. Then click **Next**.
14. On the Manual configuration steps dialog box, click **Next**.
15. On the Summary dialog box, review the selections and then click **Next**.
16. When the installation completes, you will see a Manual configuration dialog box. Click **Next**.
17. Close the browser window, and click **Finish**.
18. Close the Launchpad Base dialog box.

Install WebSphere Web Server V6.0.2 plug-in Refresh

We have listed the key options we selected during the WebSphere Web Server V6.0.2 plug-in Refresh installation:

1. Ensure the following IBM HTTP Server Windows services are stopped:
   - IBM HTTP Administration 6.0
   - IBM HTTP Server 6.0

2. Download the WebSphere Web Server V6.0.2 plug-in Refresh to a temporary directory. The Refresh (Intel Plug-ins) can be found at this Web page:

   http://www.ibm.com/support/docview.wss?rs=180&uid=swg24009813

3. Unzip the `6.0-WS-WASPlugIn-WinX32-RP0000002.zip` to the root of the WebSphere Web Server plug-in directory. For example, after the unzip you will have a directory structure similar to the following:

   `C:\IBM\WebSphere\Plugins\updateinstaller`

4. Navigate to the updateinstaller directory (for example, `C:\IBM\WebSphere\Plugins\updateinstaller`) and run **`update.exe`** to start the WebSphere Update Installer.
5. We accepted the default options for the remaining installation dialog boxes, unless noted.
6. When prompted, enter the Directory Name. For example, we entered `C:\IBM\WebSphere\Plugins`.
7. Select **Install maintenance package** and then click **Next**.
8. Enter the file name of the maintenance pack and click **Next** (for example, `C:\IBM\WebSphere\Plugins\updateinstaller\maintenance\6.0-WS-WASPlugIn-WinX32-RP0000002.pak`).
9. The maintenance package should be listed. Click **Next** to begin the update.

   Note: The WebSphere Web Server V6.0.2 Refresh includes updates to the JDK™. The WebSphere Update Installer is a Java based application that uses the JDK. The Update Installer will make a copy of the JDK files and restart using the copied version thus allowing the JDK used by WebSphere Application Server to be updated (files not in use).

   Ensure that you relaunch the Update Installer to install the V6.0.2 Refresh.

10. Click **Relaunch** and repeat the above listed steps to install Refresh V6.0.2 now that the JDK has been updated.
11. Look for the Upgrade successful dialog box, click **Finish**.

Install WebSphere Java SDK V6.0.2.7 Fixpack

We installed WebSphere Java SDK V6.0.2.7 Fixpack.

1. Ensure the following IBM HTTP Server Windows services are stopped:
   - IBM HTTP Administration 6.0
   - IBM HTTP Server 6.0

2. Download and install the WebSphere Java SDK V6.0.2.7 Fixpack. The Fixpack (Intel Java SDK) can be found at this Web page:

   http://www.ibm.com/support/docview.wss?rs=180&uid=swg24011539

3. Rename the existing updateinstaller directory in the root of the WebSphere Web Server plug-in directory (for example, updateinstaller.602).

4. Unzip the 6.0.2-WS-WASJavaSDK-WinX32-FP0000007.zip to the root of the WebSphere Web Server plug-in directory. For example, after the unzip you will have a directory structure similar to the following:

   C:\IBM\WebSphere\Plugins\updateinstaller

5. Navigate to the updateinstaller directory (for example, C:\IBM\WebSphere\Plugins\updateinstaller) and run **update.exe** to start the WebSphere Update Installer.

6. We accepted the default options for the remaining installation dialog boxes, unless noted.

7. When prompted, enter the Directory Name for WebSphere Web Server plug-in. For example, we entered C:\IBM\WebSphere\Plugins.

8. Select **Install maintenance package** and then click **Next**.

9. Enter the file name of the maintenance pack and click **Next** (for example, C:\IBM\WebSphere\Plugins\updateinstaller\maintenance\6.0.2-WS-WASJavaSDK-WinX32-FP0000007.pak).

10. The maintenance package should be listed. Click **Next** to begin the update.

11. Click **Relaunch** and repeat the above listed steps to install the Java SDK V6.0.2.7 Fixpak now that the JDK has been updated.

12. Look for the Upgrade successful dialog box and click **Finish**.

Install WebSphere Web Server V6.0.2.7 plug-in Fixpack

We have listed the key options we selected during the WebSphere Web Server V6.0.2.7 plug-in Fixpack installation:

1. Ensure the following IBM HTTP Server Windows services are stopped:
   - IBM HTTP Administration 6.0
   - IBM HTTP Server 6.0

2. Download the and install the WebSphere Web Server V6.0.2.7 plug-in Fixpack. The Refresh (Intel Plug-ins) can be found at:

   http://www.ibm.com/support/docview.wss?rs=180&uid=swg24011539

3. Rename the existing updateinstaller directory in the root of the WebSphere Web Server plug-in directory (for example, updateinstaller.6027java).

4. Unzip the `6.0.2-WS-WASPlugIn-WinX32-FP0000007.zip` to the root of the WebSphere Web Server plug-in directory. For example, after the unzip you will have a directory structure similar to the following:

   `C:\IBM\WebSphere\Plugins\updateinstaller`

5. Navigate to the updateinstaller directory (for example, `C:\IBM\WebSphere\Plugins\updateinstaller`) and run **`update.exe`** to start the WebSphere Update Installer.
6. We accepted the default options for the remaining installation dialog boxes, unless noted.
7. When prompted, enter the Directory Name for WebSphere Web Server plug-in. For example, we entered `C:\IBM\WebSphere\Plugins`.
8. Select **Install maintenance package** and then click **Next**.
9. Enter the file name of the maintenance pack and click **Next** (for example, `C:\IBM\WebSphere\Plugins\updateinstaller\maintenance\6.0.2-WS-WASPlugIn-WinX32-FP0000007.pak`).
10. The maintenance package should be listed. Click **Next** to begin the update.
11. Look for the Upgrade successful dialog box and click **Finish**.
12. Verify the version is V6.0.2.7 by enter the following in a command window:

    ```
    cd \IBM\WebSphere\Plugins\bin
    versionInfo
    ```

    You should see output similar to Example 9-1.

Example 9-1 WebSphere Plugin Fixpack version verification

```
Installed Product
------------------------------------------------------------------------------
Name                    Web server plug-ins for IBM WebSphere Application
Server
Version                 6.0.2.7
ID                      PLG
Build Level             cf70605.08
Build Date              1/30/06
------------------------------------------------------------------------------
```

9.3 Implement ITSO Application Server node

This section describes the high-level steps required to install the ITSO Application Server node.

Complete the following tasks:

1. Microsoft Windows 2003 Server and system preparation
2. DB2 Universal Database installation
3. WebSphere Application Server ND installation
4. Create the WebSphere profiles
5. Federate the Application Server
6. Create and configure the Web Services Gateway
7. Configure the Application Server with remote Web Server

9.3.1 Microsoft Windows 2003 Server and system preparation

Refer to 9.3.1, “Microsoft Windows 2003 Server and system preparation” on page 295 for details.

9.3.2 DB2 Universal Database installation

This section describes how to install IBM DB2 Universal Database V8.2 Enterprise Server Edition with Fixpack 10. DB2 UDB will be used by the ITSO Car Rental sample application.

Complete the following tasks:

1. Install DB2 UDB ESE V8.2
2. Install DB2 UDB ESE V8.2 Fixpack 10
3. Verify DB2 UDB

Install DB2 UDB ESE V8.2

We have listed the key options we selected during the IBM DB2 Universal Database V8.2, Enterprise Server Edition installation:

1. Run **setup.exe** start the installer from the root of the DB2 Universal Database CD.
2. When the DB2 Setup dialog box opens, click **Install Product** to begin the installation.
3. Select **DB2 UDB Enterprise Server Edition**, and click **Next**.
4. When the Welcome dialog box opens, click **Next**.
5. When the Software License Agreement dialog box opens, review the terms and, if you are in agreement, select **I accept the terms in the license agreement**, and then click **Next**.
6. When the Installation type dialog box opens, select **Typical** and click **Next**.
7. When the Installation action dialog box opens, we accepted the default (Install DB2 Enterprise Server Edition on this computer) and clicked **Next**.

8. When the Installation directory dialog box opens, we entered `C:\IBM\SQLLIB` in the Directory name field and clicked **Next**.
9. When the Set user information for DB2 Administration Server dialog box opens, enter information in the following fields. We entered the following information and then clicked **Next**:
   - User name: `db2admin`
   - Password: `<db2admin_password>`
   - Confirm password: `<db2admin_password>`
   - Check **Use the user name and password for the remaining DB2 services**
10. On the Set up the administration contact list dialog box, accept the default and click **Next**.
11. When the Warning dialog for SMTP notification opens, click **OK**.
12. When the Configure DB2 Services dialog box opens, accept the default DB2 instance (DB2), and then click **Next**.
13. When the Prepare the DB2 tools catalog dialog box opens, accept default (Do not prepare the DB2 tools catalog on this computer), and click **Next**.
14. When the Specify a contact for health monitor notification box opens, select **Defer the task until after installation is complete**, and then click **Next**.
15. When the Start copying files dialog box opens, review the installation options and then click **Install**.
16. When the Setup is complete dialog box opens, click **Finish**.
17. If you system is connected to the Internet, you can see a DB2 Product Updates dialog box. Click **No**.
18. When the First Steps dialog box opens, click **Exit First Steps**.

Install DB2 UDB ESE V8.2 Fixpack 10

We have listed the key options we selected during the IBM DB2 Universal Database V8.2 Fix pack 10 installation:

Note: For information on downloading and installing (FixpackReadme.txt) IBM DB2 UDB V8 Enterprise Server Edition Fixpack 10 refer to:

`http://www.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/v8fphist.d2w/report#WIN-32`

1. Download IBM DB2 Universal Database V8.2 Fix pack 10 for at this Web site:

   `http://www.ibm.com/cgi-bin/db2www/data/db2/udb/winos2unix/support/v8fphist.d2w/report#WIN-32`

2. Run **FP10_WR21362_ESE.exe**, which is a self extracting installer.

   Note: If the installer does not run automatically, run the **setup.exe** from the directory you have extracted the fix pack.

3. Click **Install Product** to begin the fix pack installation.
4. We accepted the default options for the remaining installation dialog boxes, unless noted.
5. The Fix Pack installer prompts you to stop the necessary DB2 services. Click **Yes**.
6. You are prompted to restart your system for the changes to take effect. Click **Yes**.

   Note: You might be prompted to check for DB2 Universal Database Product Updates. Click **No**.

7. After the system restarts, the DB2 Product Updates and First Steps dialog boxes open.
   - Click **No** on the DB2 Product Updates dialog box.
   - Click **Exit First Steps** on the First Steps dialog box.

Verify DB2 UDB

After installing IBM DB2 Universal Database V8.2 + Fix Pack 10, we recommend that you verify the DB2 Universal Database environment.

1. Open a DB2 command window and enter the following command:

   ```
   db2level
   ```

   It should return the internal level `8.1.10.812` for DB2 Universal Database V8.2 + Fix pack 10.

2. Launch the DB2 Universal Database First Steps by clicking **Start** → **Programs** → **IBM DB2** → **Set-up Tools** → **First Steps**.
3. Create a sample database by clicking **Create Sample Database** from the First Steps dialog box.
4. Click **Work with Databases** from the First Steps dialog box to launch the Control Center. Verify that the SAMPLE database exists.

9.3.3 WebSphere Application Server ND installation

This section describes how to install IBM WebSphere Application Server Network Deployment V6.0 with Refresh V6.0.2 and Fixpack V6.0.2.5.

Attention: Most Windows OS have a file path limit of 259 character paths. When an application is deployed with a long name, it is possible the deployed application files will reach this path limit. Currently, there is no check performed and the error messages when trying to create such long files are not very descriptive.

We strongly recommend that you shorten the default WebSphere Application Server installation path, cell name, node name, and server name to avoid the path-limit problem.

Complete the following tasks:

1. Install WebSphere Application Server ND V6.0
2. Install WebSphere Application Server V6.0.2 Refresh
3. Install WebSphere Java SDK V6.0.2.7 Fixpack
4. Install WebSphere Application Server V6.0.2.7 Fixpack

Install WebSphere Application Server ND V6.0

This section describes how we installed IBM WebSphere Application Server Network Deployment V6.0 for our example runtime environment.

1. Run **`launchpad.bat`** to start the installer from the root of the WebSphere Application Server CD.

   Note: When installing software on a Microsoft Windows 2003 Server system, you might see a dialog box with the message, `The publisher could not be verified. Are you sure you want to run this software?` Click **Run** to continue.

2. When the Launchpad home dialog box opens, click **WebSphere Application Server Network Deployment Installation**.
3. Click **Launch the installation wizard for WebSphere Application Server Network Deployment**.
4. On the Welcome dialog box, click **Next**.
5. On the Software License Agreement dialog box, review the terms and, if you are in agreement, select **I accept the terms in the license agreement**, then click **Next**.
6. When the System prerequisites check dialog box opens, click **Next**.
7. When the Installation directory dialog box opens, enter `C:\IBM\WebSphere\AppServer` in the Directory name field and click **Next**.

8. When the Features dialog box opens, accept the default features (Core product files, Application Server Samples, Javadocs), and clicked **Next**.
9. On the Installation Summary dialog box, review the selections and click **Next**.
10. On the Installation complete dialog box, deselect **Launch the Profile creation wizard**, and click **Next**.

> **Note:** We will run the Profile creation wizard after we have installed the WebSphere Application Server Refresh V6.0.2 and Fixpack V6.0.2.5.

11. When you see the message, `Warning : This installation is not yet operational appears`, click **Next**.
12. Deselect **First steps console**, and click **Finish**.
13. Close the WebSphere Application Server Network Deployment Launchpad home dialog box.

Install WebSphere Application Server V6.0.2 Refresh

This section describes the key steps we performed to install the IBM WebSphere Application Server V6.0.2 Refresh.

1. Ensure all WebSphere Application Server processes are stopped.
2. Download the WebSphere Application Server V6.0.2 Refresh to a temporary directory. The Refresh (Intel Application Server) can be found at the following URL:

   `http://www.ibm.com/support/docview.wss?rs=180&uid=swg24009813`
3. Rename the updateinstaller directory under the root of the WebSphere Application Server installation directory, if it exists.
4. Unzip the `6.0-WS-WAS-WinX32-RP0000002.zip` to the root of the WebSphere Application Server installation directory. For example, after the unzip you will have a directory structure like the following:

   `C:\IBM\WebSphere\AppServer\updateinstaller`
5. Navigate to the updateinstaller directory (for example, `C:\IBM\WebSphere\AppServer\updateinstaller`) and run **`update.exe`** to start the WebSphere Update Installer.
6. On the Welcome dialog box, you can see a warning to stop all WebSphere related processes. Click **Next**.
7. When prompted for the Directory Name, enter `C:\IBM\WebSphere\AppServer` and click **Next**.
8. Select **Install maintenance package** and then click **Next**.

9. Enter the file name of the maintenance pack and click **Next** (for example, `C:\IBM\WebSphere\AppServer\updateinstaller\maintenance\6.0-WS-WAS-WinX32-RP0000002.pak`).
10. The maintenance package should be listed. Click **Next** to begin the update.

> **Note:** The WebSphere Application Server V6.0.2 Refresh includes updates to the JDK. The WebSphere Update Installer is a Java based application that uses the JDK. The Update Installer makes a copy of the JDK files and restarts using the copied version, thus allowing the JDK used by WebSphere Application Server to be updated (files not in use).
>
> Ensure that you relaunch the Update Installer to install the V6.0.2 Refresh.

11. Click **Relaunch** and repeat these steps to install Refresh V6.0.2 now that the JDK has been updated.
12. When the Refresh installation is complete, click **Finish**.

Install WebSphere Java SDK V6.0.2.7 Fixpack

We installed WebSphere Java SDK V6.0.2.7 Fixpack.

1. Ensure the following IBM HTTP Server Windows services are stopped:
   - IBM HTTP Administration 6.0
   - IBM HTTP Server 6.0
2. Download and install the WebSphere Java SDK V6.0.2.7 Fixpack to a temporary directory. The Refresh (Intel Java SDK) can be found at this Web page:

   `http://www.ibm.com/support/docview.wss?rs=180&uid=swg24011539`
3. Rename the existing updateinstaller directory in the root of the WebSphere Web Server plug-in directory (for example, `updateinstaller.602`).
4. Unzip the `6.0.2-WS-WASJavaSDK-WinX32-FP0000007.zip` to the root of the WebSphere Web Server plug-in directory. For example, after the unzip you will have a directory structure like the following:

   `C:\IBM\WebSphere\Plugins\updateinstaller`
5. Navigate to the updateinstaller directory (for example, `C:\IBM\WebSphere\Plugins\updateinstaller`) and run **`update.exe`** to start the WebSphere Update Installer.
6. We accepted the default options for the remaining installation dialog boxes, unless noted.
7. When prompted, enter the Directory Name for WebSphere Web Server plug-in. For example, we entered `C:\IBM\WebSphere\Plugins`.

8. Select **Install maintenance package** and then click **Next**.
9. Enter the file name of the maintenance pack and click **Next** (for example, `C:\IBM\WebSphere\Plugins\updateinstaller\maintenance\6.0.2-WS-WASJavaSDK-WinX32-FP0000007.pak`).
10. The maintenance package should be listed. Click **Next** to begin the update.
11. Click **Relaunch** and repeat these steps to install the Java SDK V6.0.2.7 Fixpak now that the JDK has been updated.

    You should see the Upgrade successful dialog box, click **Finish**.

Install WebSphere Application Server V6.0.2.7 Fixpack

This section describes the key steps we performed to install the IBM WebSphere Application Server V6.0.2.7 Fixpack.

1. Ensure all WebSphere Application Server processes are stopped.
2. Download and install the WebSphere Application Server V6.0.2.7 Fixpack. The Fixpack (Intel Application Server) can be found at the following URL:

   `http://www.ibm.com/support/docview.wss?rs=180&uid=swg24011539`
3. Rename the existing updateinstaller directory in the root of the WebSphere Web Server plug-in directory (for example, `updateinstaller.6027java`).
4. Unzip the `6.0.2-WS-WAS-WinX32-FP0000007.zip` to the root of the WebSphere Web Server plug-in directory. For example, after the unzip you will have a directory structure similar to the following:

   `C:\IBM\WebSphere\AppServer\updateinstaller`
5. Navigate to the updateinstaller directory (for example, `C:\IBM\WebSphere\AppServer\updateinstaller`) and run **`update.exe`** to start the WebSphere Update Installer.
6. We accepted the default options for the remaining installation dialog boxes, unless noted.
7. When prompted, enter the Directory Name for WebSphere Web Server plug-in. For example, we entered `C:\IBM\WebSphere\AppServer`.
8. Select **Install maintenance package** and then click **Next**.
9. Enter the file name of the maintenance pack and click **Next** (for example, `C:\IBM\WebSphere\AppServer\updateinstaller\maintenance\6.0.2-WS-WAS-WinX32-FP0000007.pak`).
10. The maintenance package should be listed. Click **Next** to begin the update.
11. Look for the Upgrade successful dialog box, click **Finish**.

12. Verify that the fixpack V6.0.2.7 is installed by entering the following commands in a command window:

```
cd \IBM\WebSphere\AppServer\bin
versioninfo
```

You should see a message similar to Example 9-2.

Example 9-2 Verifying WebSphere Application Server Fixpack version

```
Installed Product
------------------------------------------------------------------------------
Name                        IBM WebSphere Application Server - ND
Version                     6.0.2.7
ID                          ND
Build Level                 cf70605.08
Build Date                  1/30/06
------------------------------------------------------------------------------
```

9.3.4 Create the WebSphere profiles

In our example, we create two WebSphere profiles, a Deployment Manager profile and an Application Server profile. The Deployment Manager profile will include an application server to host the Deployment Manager Administrative Console. The Application Server profile will include an application server to host the Web Services Gateway, and the ITSO Car Rental Application. In a production environment, the Deployment Manager, and Web Services Gateway would likely be on a separate node from the application.

Note: Launch the WebSphere Profile Wizard

- Click **Start** → **Programs** → **IBM WebSphere** → **Application Server Network Deployment v6** → **Profile creation wizard**.

  or

- You can launch the WebSphere Profile Wizard manually by running **`pctWindows.exe`** from the <was_home>\bin\ProfileCreator directory.

Create the Deployment Manager profile

To create the Deployment Manager profile, follow these steps:

1. Launch the WebSphere Profile Wizard (see notebox on this page).
2. When the Welcome dialog box opens, click **Next**.
3. Select **Create a deployment manager profile**, and click **Next**.

Attention: At the time of writing, Microsoft Windows 2003 Server had a file path limit of 259 character paths. When an application is deployed with a long name (including path), it is possible the deployed application files will reach this path limit. Currently, there is no check performed and the error messages when trying to create such long files are not very descriptive.

We strongly recommend that you shorten the default WebSphere Application Server installation path, cell name, node name, and server name to avoid the path-limit problem.

4. In the Profile name dialog box, accept the default profile name (Dmgr01) and click **Next**.
5. When the Profile directory dialog box opens, accept the default directory, as we did (`C:\IBM\WebSphere\AppServer\profiles\Dmgr01`) and clicked **Next**.
6. When the Node, host, and cell names dialog box opens, accept the following default values and then click **Next**, as we did:
   - Node name: `itsoCellMgr1`

     Note: We shortened the default node name to avoid the Windows path-limit problem.
   - Host name: `itsoapp1.itso.ral.ibm.com`
   - Cell name: `itsoCell1`

     Note: We shortened the default cell name to avoid the Windows path-limit problem (default <hostname>Cell01).
7. When the Port value assignment dialog box opens, accept the default values seen in Figure 9-3, and then click **Next**, as we did.

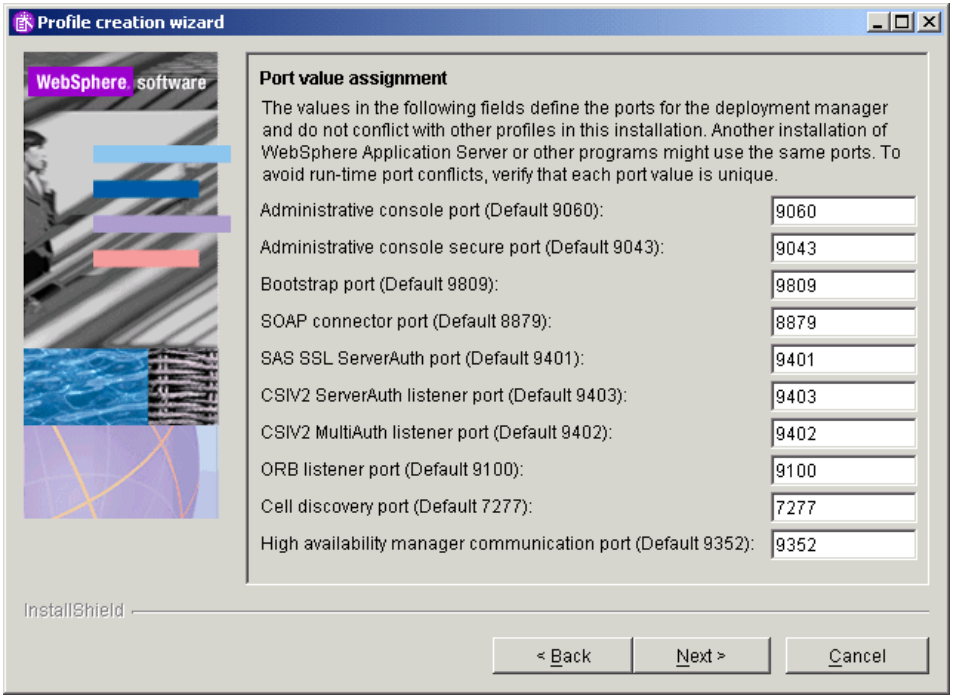


Figure 9-3 Create profile: Port values for Deployment Manager profile

> **Note:** Every time a new profile is created, the ports increment from the default value to avoid port conflicts with or profiles that might be running simultaneously.

8. When the Windows service definition dialog box opens, enter the user name and password to run the Application Server process as a Windows service. Select the startup type (**Manual** or Automatic) as seen in Figure 9-4 on page 305, and click **Next**.

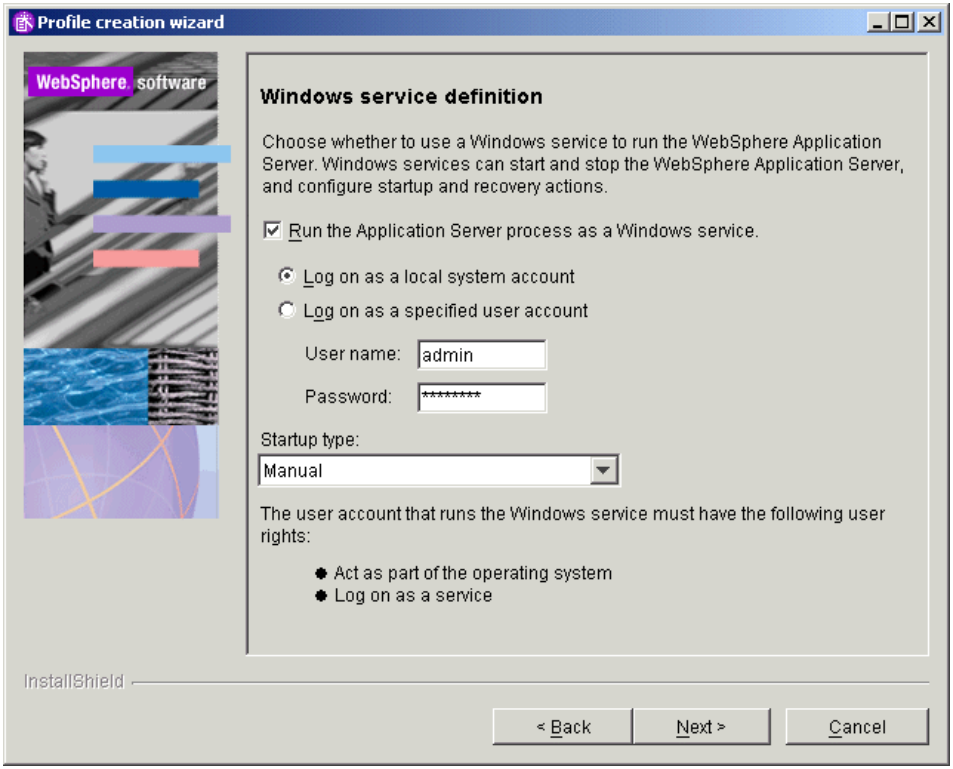


Figure 9-4 Create profile: Windows service definition

9. On the Profile summary dialog box, review the settings and then click **Next** to create the profile.
10. When complete, look for a dialog box with the message, `Profile creation is complete`. Deselect **Launch the First steps console,** then click **Finish**.

Create the Application Server profile

To create the Application Server profile used by the Web Services Gateway and the ITSO Car Rental application, follow these steps:

1. Launch the WebSphere Profile Wizard.
2. When the Welcome dialog box opens, click **Next**.
3. Select **Create an Application Server profile**, and click **Next**.
4. When the Profile name dialog box opens, accept the default profile name (AppSrv01) and click **Next**, as we did.
5. When the Profile directory dialog box opens, accept the default directory (C:\IBM\WebSphere\AppServer\profiles\AppSrv01) and click **Next**.

6. When the Node and host names dialog box opens, accept the following default values and then click **Next**, as we did:
   - Node name: `itsoNode1`

     Note: We shortened the default cell name to avoid the Windows path-limit problem (default <hostname>Node01).
   - Host name: `itsoapp1.itso.ral.ibm.com`
7. When the Port value assignment dialog box opens, accept the default values and click **Next**, as we did.

   Note: Every time a new profile is created, the ports increment from the default value to avoid port conflicts with or profiles that might be running simultaneously.

8. On the Windows service definition dialog box, enter the user name and password to run the Application Server process as a Windows service, select the startup type (**Manual** or Automatic), and click **Next**.
9. When the Profile summary dialog box opens, review the settings and then click **Next** to create the profile.
10. When you are finished, you should see a dialog box with the message, `Profile creation is complete`. Deselect **Launch the First steps console**, then click **Finish**.

9.3.5 Federate the Application Server

In our example, we created an application server profile, and now want to add the node and server to the cell:

1. Start the Deployment Manager:

   ```
   cd \IBM\WebSphere\AppServer\profiles\Dmgr01\bin
   startmanager
   ```

   Note: Alternatively, start the IBM WebSphere Application Server - *itsoCellMgr1* Windows service. Where *itsoCellMgr1* is the node name including Deployment Manager.

2. Start the server1 application server for the Application Server profile:

   ```
   cd \IBM\WebSphere\AppServer\profiles\AppSrv01\bin
   startserver server1
   ```

Note: Alternatively, start the IBM WebSphere Application Server - *itsoNode1* Windows service. Where *itsoNode1* is the node name including server1.

3. Start the Deployment Manager Administrative Console by entering the following URL in a Web browser:

   `http://itsoapp1.itso.ral.ibm.com:9060/ibm/console`

4. When prompted, enter a user ID.

Note: At this stage, WebSphere security is not enable, so we are not prompted for a password.

5. Select **System administration** → **Nodes**.
6. Click **Add Node**.
7. Select **Managed node** and click **Next**.
8. On the Add Managed Node dialog box, enter the following as seen in Figure 9-5 on page 308, and then click **OK**:
   - Host: `itsoapp1`
   - Select **Include applications**

Note: If this is a newly created application server profile, it will contain the sample applications, so be sure to check this box if you want to keep the samples. We included them for testing purposes.

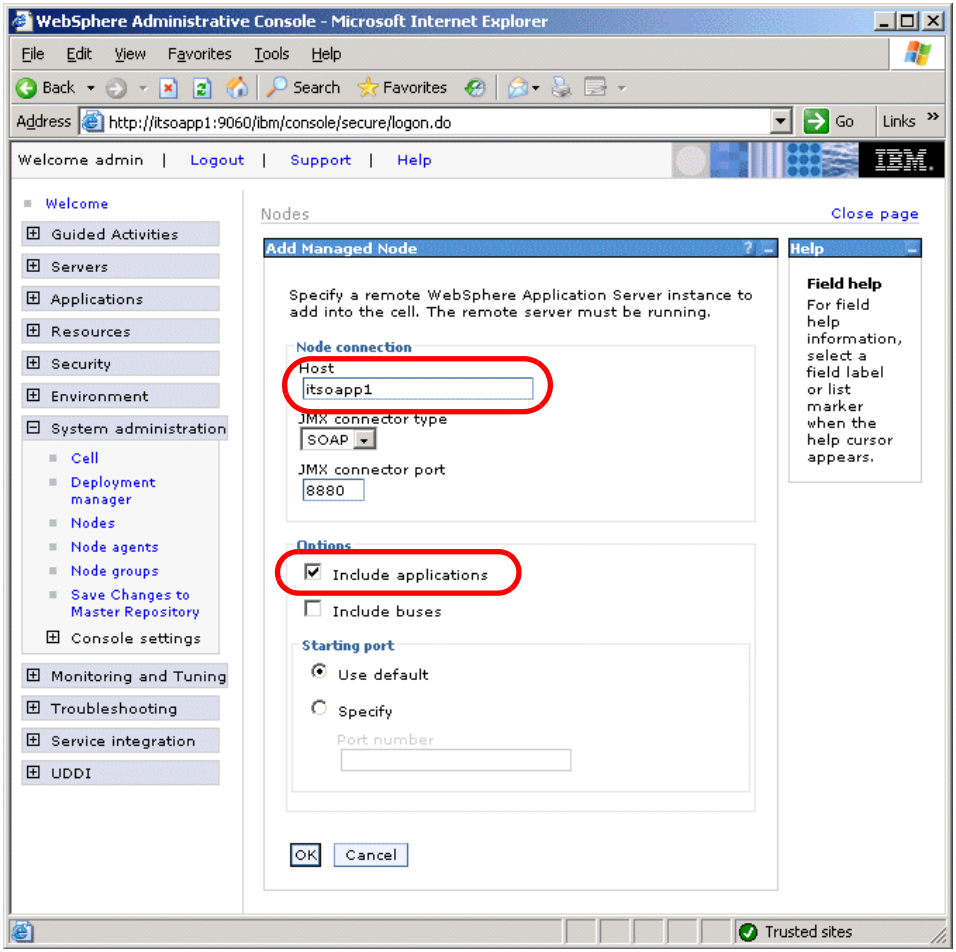


Figure 9-5 Add Managed node

9. If the node is a Windows node, you have the opportunity to register the new node agent as a Windows service. Make your selection and click **OK** as seen in Figure 9-6 on page 309.

> **Note:** The federation process stops the application server. It creates a new node agent for the node, and adds the node to the cell. The application server becomes a managed server in the cell. It then starts the node agent, but not the server.

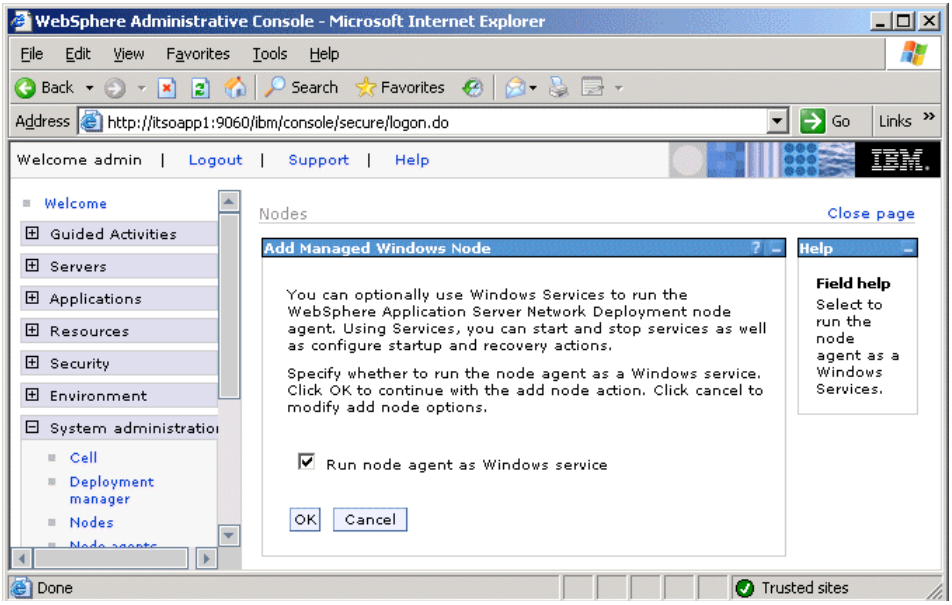


Figure 9-6 Run a node agent as a Windows service

10. After the node has been added, click **Logout from the Administrative Console**.

    You can now display the new node, node agent, and application server from the console. You can also start the server from the console.

11. Configure the node agent Windows service startup to manual.

    Once the Application Server node is federated, the node agent will become active. By default, the node agent Windows service will be set to start automatically. We recommend that you change the startup to manual.

At the completion of the process:

- The profile directory for the application server still exists and is used for the application server.
- The old cell name for the application server has been replaced with a profile directory with the cell name of the deployment manager.

  `<profile_home>/config/cells/<dmgr_cellname>/`
- A new entry in the deployment manager profile directory has been added for the new node.

  `<dmgr_profile_home>/config/cells/<dmgr_cellname>/nodes/<federated node>`

- An entry for each node in the cell is added to the application server profile configuration. Each node entry contains the serverindex.xml file for the node.

  `<profile_home>/config/cells/<dmgr_cellname>/nodes/<federated node>`

  In turn, an entry for the new node is added to the nodes directory for each node in the cell with a serverindex.xml entry for the new node.

9.3.6 Create and configure the Web Services Gateway

The Web Services Gateway is a tool that gives administrators an easy way to map Web services to an external Web services client, and enables the mapping of different protocols. In our scenario, we use SOAP over HTTP from the external travel company Web service invocation to the ITSO Car Rental application Web services.

Complete the following tasks to configure the Web Services Gateway:

1. Create a bus.
2. Add a bus member.
3. Install the Service Data Objects repository application.
4. Configure the data source for the SDO repository database.
5. Install the Service Integration Bus Web Services applications.
   - Install the Resource Adapter.
   - Install Web services support application.
   - Install the SOAP over HTTP endpoint listener application.
6. Create the SOAP over HTTP endpoint listener.
7. Create a gateway instance.

Create a bus

To create a bus, follow these steps:

1. In the Deployment Manager Administrative Console, expand **Service integration** and click **Buses**.
2. Click **New**.
3. When the New bus configuration dialog box opens, enter the following information as seen in Figure 9-7 on page 311, and then click **OK**:
   - Name: `WSGBus`
   - Deselect the **Secure** checkbox.

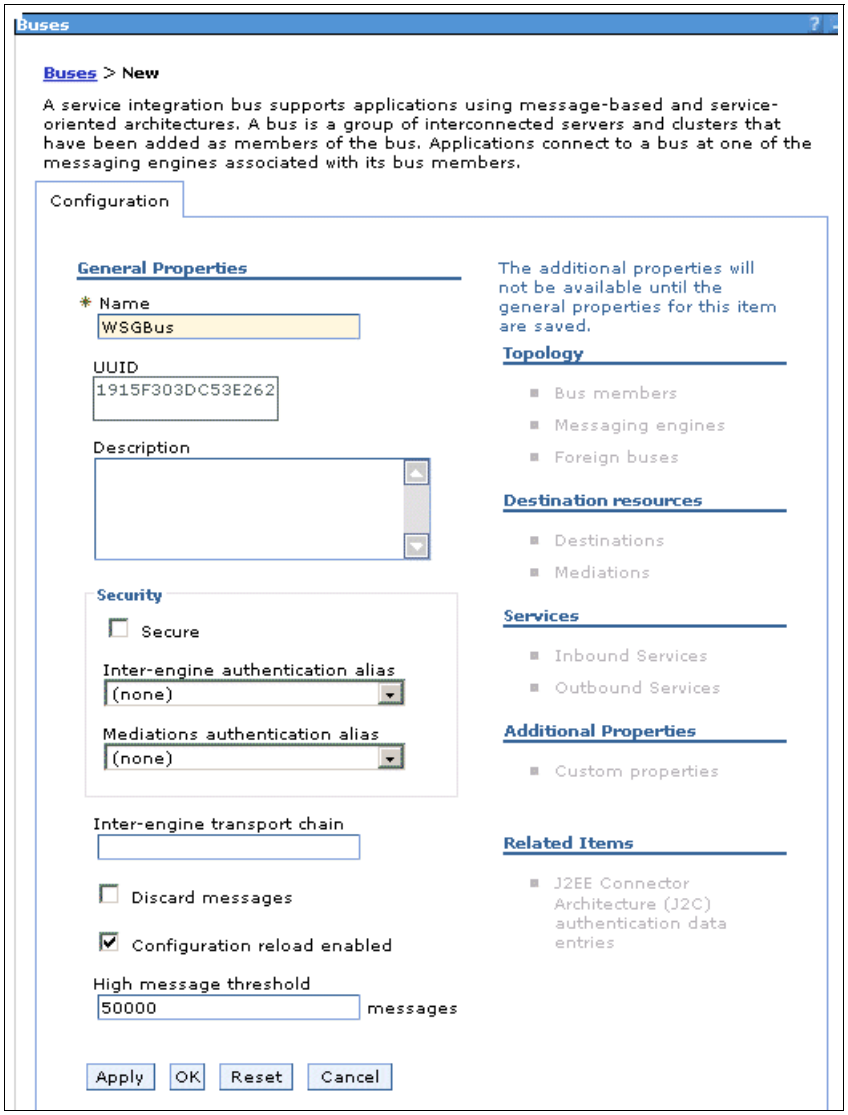


Figure 9-7 New bus details entry page

4. Click **Save**. Click **Save** to save to master configuration.

Add a bus member

In the previous section, we create the bus. The creation of the bus is just an administrative entity. It does not create any resources for messaging. To create resources, we must add a bus member, which has the effect of creating a

messaging engine. The messaging engine is used by the bus to forward requests and responses between service providers and service consumers.

To add a bus member, follow these steps:

1. Click **WSGBus** to show its properties.
2. Click **Bus members** under Topology.
3. Click **Add**.
4. Accept the defaults, and click **Next**.
5. When the summary page opens, click **Finish**. The server is added as a member of the bus, and a messaging engine is created.
6. Click **Save**. Click **Save** to save to master configuration.

Install the Service Data Objects repository application

The Web Services Gateway requires the use of the service integration bus Web services (SIBWS). The SIBWS enablement uses a Service Data Objects (SDO) repository for storing and serving WSDL definitions.

When WebSphere Application Server Network Deployment is installed, it does not install the SDO repository, so this step must be performed manually. When the SDO repository is installed, it will use Cloudescape by default.

Note: The SDO repository can be configured to use other databases, such as DB2 UDB, Oracle, Informix®, and Sybase. This section provide a detailed procedure to configure the SDO repository to use DB2 UDB.

Create the DB2 UDB SDO repository database

To create the DB2 UDB SDO repository database, follow these steps:

1. Open the DB2 Command Window by clicking **Start** → **Programs** → **IBM DB2** → **Command Line Tools** → **Command Window**.
2. Enter the following commands, in the DB2 Command Window:

```
db2 create db sdodb
db2 connect to sdodb
db2 -tvf C:\IBM\WebSphere\AppServer\util\SdoRepository\DB2UDB_V82\Table.ddl
db2 disconnect sdodb
```

Note: In our example, we are logged into Microsoft Windows as the administrative user ID admin (see “Microsoft Windows 2003 Server and system preparation” on page 286). The user logged in by default will be used to create and be the owner of the database.

Install the SDO repository application on Deployment Manager

To install the SDO repository with a DB2 UDB backend to the Deployment Manager, complete these tasks:

1. Open a command window and navigate to the <WASND_HOME>/bin directory, where <WASND_HOME> is the directory where you installed WebSphere Application Server Network Deployment. For example:

   ```
   C:\IBM\WebSphere\AppServer\bin
   ```

2. Run the following command to install the SDO repository enterprise application on Deployment Manager, with a DB2 UDB backend:

   ```
   wsadmin -f installSdoRepository.jacl -editBackendId DB2UDB_V82
   ```

   You should see a message similar to the following if it is successful:

   ```
   SDO repository installation completed successfully.
   ```

Install the SDO repository application on Application Server

Install the SDO repository application on each application server that will use the SDO repository.

1. Open a command window and navigate to the <WASND_HOME>/bin directory, where <WASND_HOME> is the directory where you installed WebSphere Application Server Network Deployment. For example:

   ```
   C:\IBM\WebSphere\AppServer\bin
   ```

2. Run the following command to install the SDO repository enterprise applications on each node that will use the SDO repository. In our example, we installed to itsoNode1 server1 as follows:

   ```
   wsadmin -f installSdoRepository.jacl itsoNode1 server1
   ```

   You should see a message similar to the followingon if it is successful:

   ```
   SDO repository installation completed successfully.
   ```

Verify that the enterprise application is installed

After the script completes successfully, the SDO repository should be installed. You can confirm the installation by examining the installed enterprise applications from the Administrative Console.

1. Start the Administrative Console by entering the following URL in a Web browser:

   ```
   http://itsoapp1.itso.ral.ibm.com:9060/ibm/console
   ```

2. When prompted, enter a user ID.

3. Ensure the server where the SDO repository application is installed is started.
   a. Expand **Servers** → **Application servers**.
   b. If the server is not started, select the server checkbox and click **Start**.
4. Expand **Applications** → **Enterprise Applications**.

   You should see a new enterprise application named, *SDO Repository* listed. The SDO Repository application status should display a green arrow, meaning that the application is running. If not, start the application.

Configure the data source for the SDO repository database

Thus far, we have created the SDO repository database and tables. Now we need to configure the connectivity to the database from WebSphere Application Server using a data source.

Create a J2C authentication alias

The data source used by the SDO repository needs to have a component-managed authentication alias. An authentication alias is used to allow the same user ID and password combination to be used in many different places. In this case the DB2 UDB SDO repository database has security configured so you need to specify the same user ID and password as created during the DB2 UDB install.

Complete the following steps to create an authentication alias:

1. Click **Security** → **Global security** from the Administrative Console.
2. Under Authentication, expand **JAAS Configuration** and click **J2C Authentication data**.
3. Click **New**.
4. When the Configuration dialog box opens, enter the following values as seen in Figure 9-8 on page 315, and then click **OK**:
   - Alias: `db2sdoalias`
   - User ID: `admin`

     Note: In our example, we created the admin user ID as an administrative user (see “Microsoft Windows 2003 Server and system preparation” on page 286). By default, the user logged into Windows when the database is created is used as the database owner.

   - Password: `<password>`
   - Description: `DB2 SDO authentication alias`

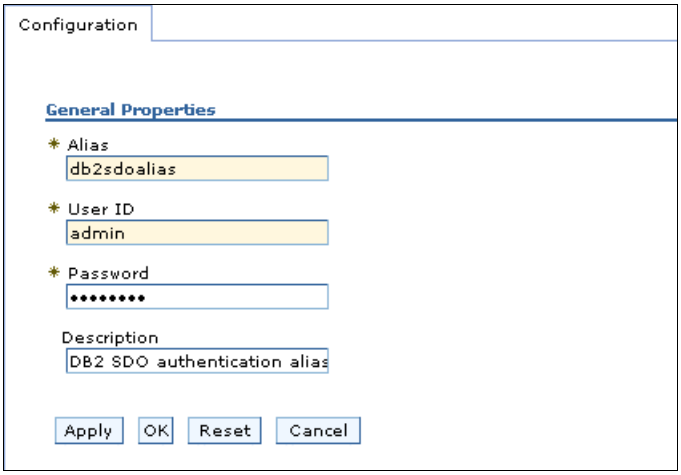


Figure 9-8 Create a J2C authentication alias

5. Click **Save**. Click **Save** again to save to master configuration.
6. Click **OK**. The new authentication alias should be displayed.

Create a JDBC provider for DB2 UDB

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

1. Click **Resources** → **JDBC Providers** from the Administrative Console.
2. Clear the **Node** entry field and click **Apply**. The cell scope should now be highlighted with the red arrow.
3. With the cell scope now applied, click **New** to create the JDBC provider.
4. Select the following items on the General Properties dialog box, as seen in Figure 9-9 on page 316, 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 **XA data source** from the Step 3: Select the implementation type drop-down list.

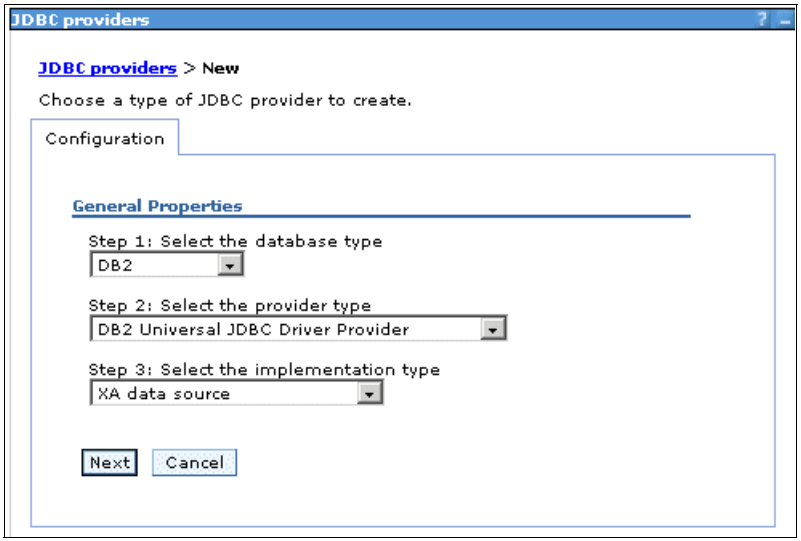


Figure 9-9 Create the JDBC provider

5. On the Variables dialog box, change the variables displayed in the classpath text box (see Figure 9-10 on page 317), 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

   Note: In our example, the JDBC provider and datasource created in a later step are defined at the cell level. At the time of writing using IBM WebSphere Application Server Network Deployment V6.0.2.7, we found that variables defined at the node level were not found to resources at the cell level. For this reason, we opted to enter the physical path to the jar files in the classpath dialog box. Also, this method reduces the number of steps required to configure a datasource.

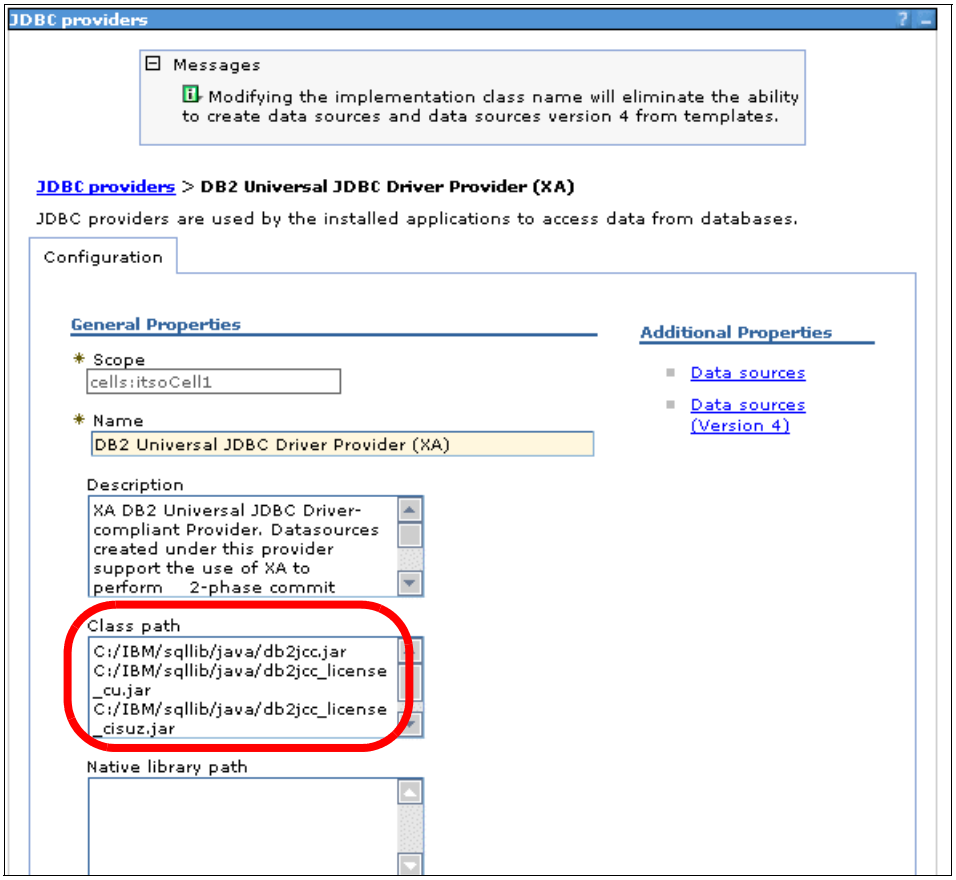


Figure 9-10 JDBC provider - DB2 UDB JDBC driver classpath

6. Click **Save**. Click **Save** again to save to master configuration.
7. Click **OK**. The new DB2 Universal JDBC Driver Provider (XA)

Create the DB2 SDO data source

To create the DB2 SDO data source, follow these steps:

1. Click **Resources** → **JDBC Providers**.
2. Clear the **Node** entry field and click **Apply**. The cell scope should now be highlighted with the red arrow.
3. Click **DB2 Universal JDBC Driver Provider (XA)**.
4. Click **Data sources** under Additional Properties.
5. Click **New**.

6. Enter the following information as seen in Figure 9-11 on page 319 and Figure 9-12 on page 320:
   - Name: `DB2 SDO data source`
   - JNDI name: `jdbc/com.ibm.ws.sdo.config/SdoRepository`
   - Component-managed authentication alias: select **itsoCellMgr1/db2sdoalias**
   - Database name: `sdodb`
   - 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: `itsoapp1.itso.ral.ibm.com`

     This is the hostname of the server where DB2 Universal Database is installed.
   - Port number: `50000`

     This is the port defined in the services file for the DB2 Universal Database connection port.
   - Leave all other values as default. Click **OK**.
7. Click **Save**. Click **Save** again to save to master configuration.
8. Click **OK**.

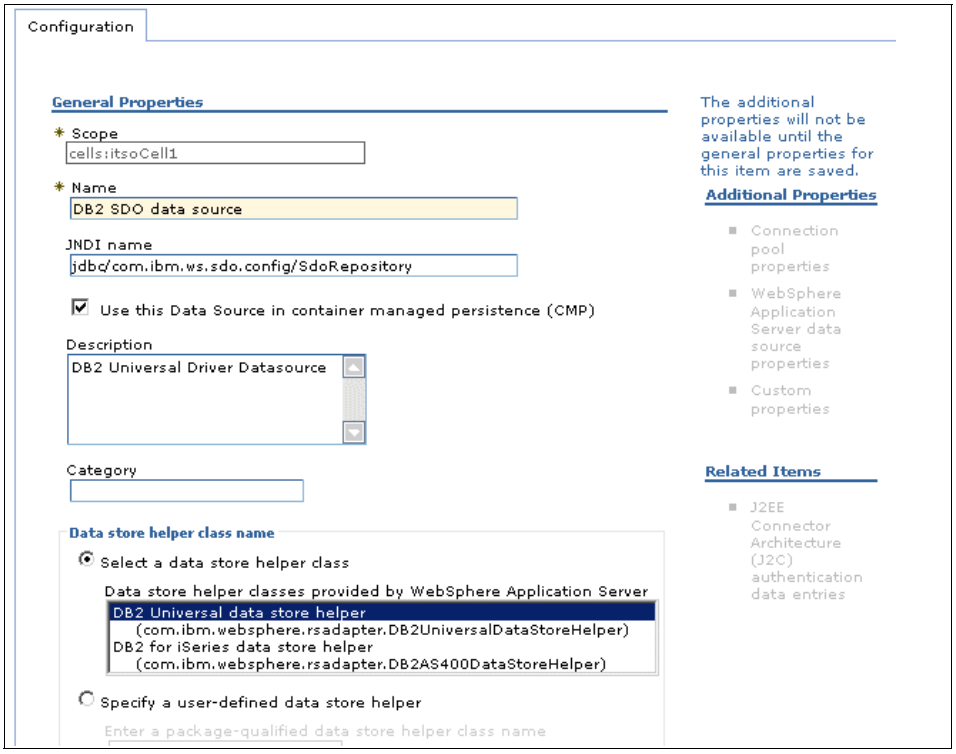


Figure 9-11 Properties for the DB2 SDO data source - part 1

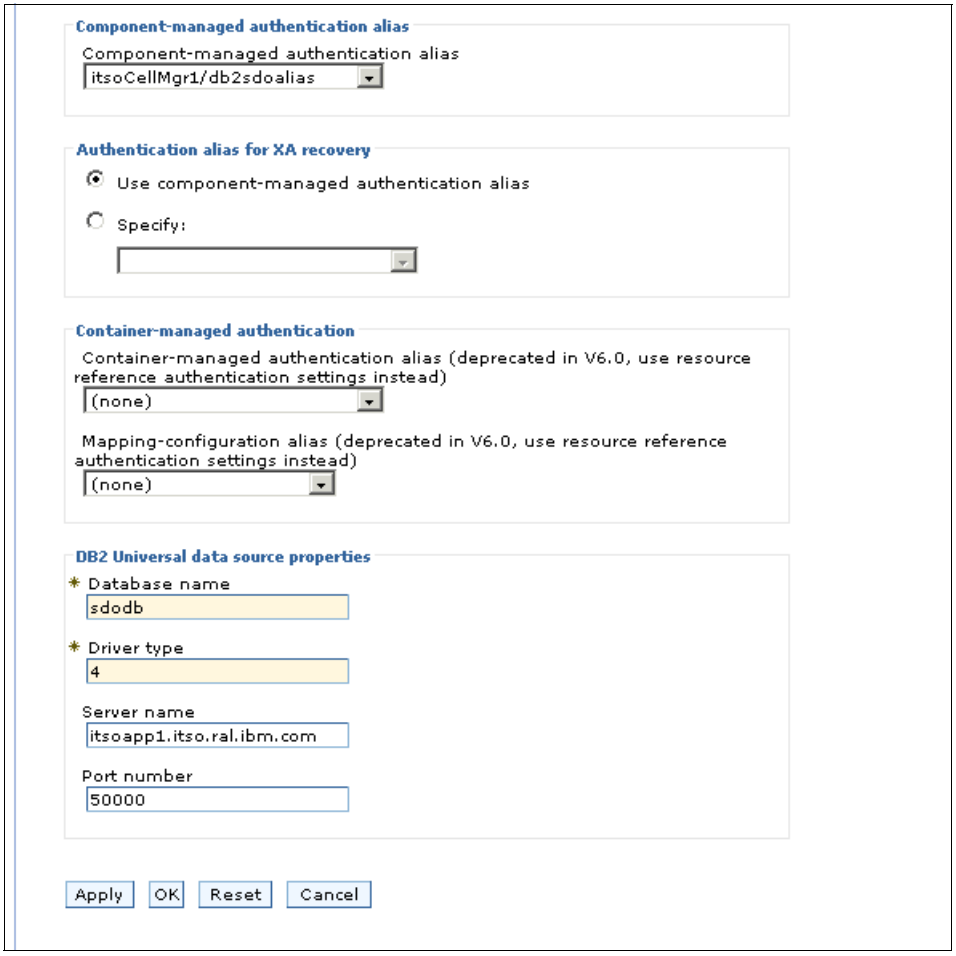


Figure 9-12 Properties for the DB2 SDO data source - part 2

Test the connection

To test the data source connection, follow these steps:

1. Select **Resources** → **JDBC Providers**.
2. Click **DB2 UNIVERSAL JDBC DRIVER PROVIDER**.
3. Click **Data sources**.
4. Check the **DB2 SDO data source** and click **Test connection**. You should see a successful message.

Tip: If the data source connection fails, then check the following:

- Verify the settings. Do you have the proper scope (cell versus node) for your settings?
- Try restarting the deployment manager, node, and the application server.

Install the Service Integration Bus Web Services applications

The service integration bus Web services (SIBWS) support is packaged into four separate applications including Resource Adapter, Web Services support, SOAP over HTTP endpoint listener, and SOAP over JMS endpoint listener. In our example scenario, we only install three of the applications, because the SOAP over JMS endpoint listener is not needed.

Complete the following tasks:

1. Install the Resource Adapter
2. Install Web services support application
3. Install the SOAP over HTTP endpoint listener application

Install the Resource Adapter

To install the Resource Adapter on node itsoNode1, complete these tasks:

1. Open a command window and navigate to the `<WAS_HOME>\bin` directory. Where `<WAS_HOME>` in our example is `C:\IBM\WebSphere\AppServer\bin`.
2. Enter the following command to install the Resource Adapter:

```
wsadmin -f C:/IBM/WebSphere/AppServer/util/sibwsInstall.jacl INSTALL_RA
-installRoot C:/IBM/WebSphere/AppServer -nodeName itsoNode1
```

Install Web services support application

To install the Web services support application on itsoNode1 server1 enter the following:

```
wsadmin -f C:/IBM/WebSphere/AppServer/util/sibwsInstall.jacl INSTALL
-installRoot C:/IBM/WebSphere/AppServer -nodeName itsoNode1 -serverName
server1
```

Install the SOAP over HTTP endpoint listener application

Although the Web services support has been installed, you cannot use it until at least one endpoint listener application is installed.

To install the SOAP over HTTP endpoint listener application on itsoNode1 server1, enter the following:

```
wsadmin -f C:/IBM/WebSphere/AppServer/util/sibwsInstall.jacl INSTALL_HTTP
-installRoot C:/IBM/WebSphere/AppServer -nodeName itsoNode1 -serverName
server1
```

Create the SOAP over HTTP endpoint listener

An endpoint listener, listens for incoming Web service requests and forwards them to the appropriate inbound server. Inbound services are bound to an endpoint listener when they are created.

To create an endpoint listener for SOAP over HTTP, follow these steps:

1. Ensure the IBM HTTP Server is started on the Web Server Redirector node (location of WSDL files hosted by IBM HTTP Server).
2. From the Administrative Console on the ITSO Application Server, expand **Servers** and click **Application servers**.
3. Click **server1**.
4. Click **Endpoint Listeners** under Additional Properties.
5. Click **New**.
6. When the end point listener configuration dialog box opens, enter the following information, then click **Apply**:
   - Name: `SOAPHTTPChannel1`
   - URL root:

     `http://itsoapp1.itso.ral.ibm.com:9080/wsgwsoaphttp1`

     The URL root is the base URL for Web service requests into this endpoint listener. The URLs that are used for making Web service requests to the SIBWS will have this root at the beginning. Replace `9080` with the correct port number for your server.

     Note: The value `wsgwsoaphttp1` is defined in the SOAP/HTTP endpoint listener application. You must enter `wsgwsoaphttp1` at the end of the URL root.

   - WSDL server HTTP URL root:

     `http://itsoapp1.itso.ral.ibm.com:9080/sibws/wsdl`

Note: Each inbound service is described in a WSDL document. Type the root of the Web address for the WSDL files of the inbound services that are available at this endpoint listener. The WSDL serving HTTP URL root is used by the HTTP and JMS endpoint listeners, and also by UDDI registries, to retrieve the WSDL for your inbound services. This URL comprises the root of the HTTP address at which external clients access your endpoint listener application, followed by /sibws/wsdl

7. Click **Connection Properties** under Additional Properties.
8. Click **Save**. Click **Save** again to save to master configuration. Click **OK**.
9. Click **New** to create a new connection property.
10. Select **WSGBus** from the Bus Name list and click **OK**.
11. Click **Save**. Click **Save** again to save to master configuration. Click **OK**.

Create a gateway instance

Create a gateway instance on the bus. The gateway instance allows you to partition gateway and proxy services into logical groups for ease of management.

1. Expand **Service Integration** and click **Buses**.
2. Click **WSGBus**.
3. Under Additional Properties, click **Web service gateway instances**.
4. Click **New**.
5. When the new Web services gateway instance configuration dialog box opens, enter the following and click **OK**:
   - Name: `WSG`

     This is an arbitrary name of the gateway instance.
   - Gateway namespace: `http://nswsg.ibm.com`

     This is an arbitrary name of the gateway namespace.
   - Default proxy WSDL URL:

     `http://itsoapp1.itso.ral.ibm.com:9080/sibws/proxywsdl/ProxyServiceTemplate.wsdl`

     This is the generic template WSDL file supplied with WebSphere Application Server Network Deployment. The template file is supplied with the installation of the SIBWS application.
6. Click **Save**. Click **Save** again to save to master configuration. Click **OK**.

9.3.7 Configure the Application Server with remote Web Server

WebSphere Application Server provides Web server plug-ins that work with a Web server to route requests for dynamic content, such as servlets, from the Web server to the proper application server. The Web server plug-in is specific to the type of Web server. It is installed on the Web server machine and configured in the Web server configuration.

A plug-in configuration file generated on the application server and placed on the Web server is used for routing information. In order to manage the generation and propagation of these plug-in configuration files, Web servers are defined to the WebSphere Application Server configuration repository.

This section describes how to configure our example Web Server Redirector node running IBM HTTP Server and WebSphere plug-in, to redirect requests to the ITSO Application Server node running WebSphere Application Server Network Deployment.

Complete the following tasks:

1. Add unmanaged Web Server node to WebSphere.
2. Create Web Server.
3. Map modules to servers.
4. Propagate plugin-cfg.xml to remote Web server.
5. Verify remote Web server redirect to WebSphere application.

Add unmanaged Web Server node to WebSphere

To add an unmanaged Web Server node to WebSphere Application Server, follow these steps:

1. Click **System administration** from the Administrative Console.
2. Click **Nodes**.
3. Click **Add Node**.
4. Select **Unmanaged node** (unmanaged nodes are nodes without a node agent) and click **Next**.
5. When the Configuration dialog box opens, enter the following as seen in Figure 9-13, and then click **OK**:
   - Name: `web1Node`
   - Host Name: `web1.itso.ral.ibm.com`
   - Platform : select **Windows**

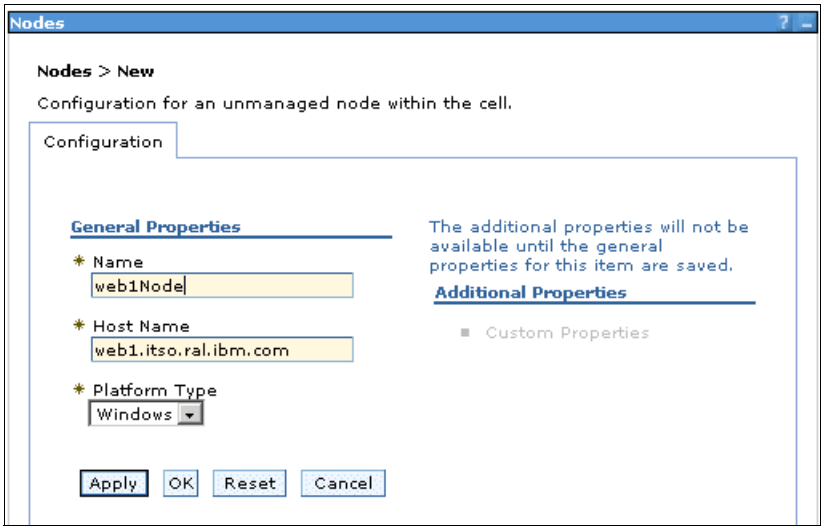


Figure 9-13 Configure Web Server name and host name

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

Create Web Server

To create the Web Server in WebSphere Application Server, follow these steps:

1. Click **Servers** from the Administrative Console.
2. Click **Web servers**.
3. Click **New**.
4. When the Step 1: Select a node dialog appears, enter the following as seen in Figure 9-14, and then click **Next**:
   - Node name: select **web1Node**

     Node name defined in “Add unmanaged Web Server node to WebSphere” on page 324).
   - Server name: `webserver1`

Note: When we installed the WebSphere plug-in for IBM HTTP Server, we defined the server name as webserver1 (see “Install WebSphere Web Server V6.0 plug-in” on page 290). The settings for the default WebSphere plug-in installation are stored in the configurewebserver1.bat.

The following procedure allows us to change the node name and name of the Web server. This configuration will override the configuration set at installation time.

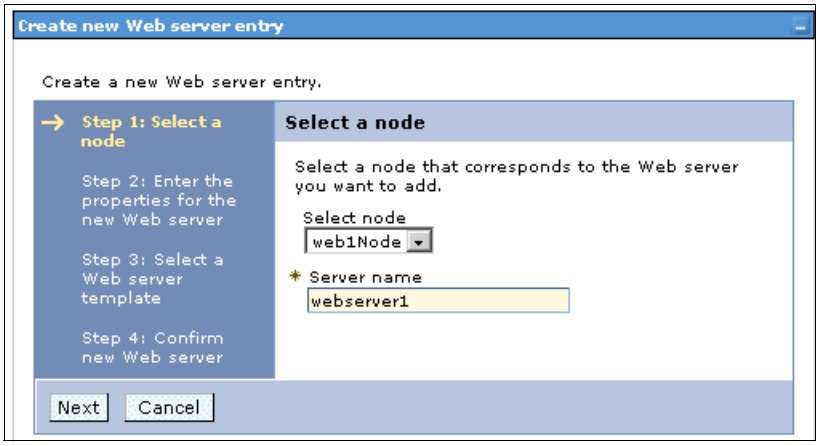


Figure 9-14 Create a Web server: Select a node

5. When the Enter the properties dialog box opens, enter the following as seen in Figure 9-15, then click **Next**:
   - Type: select **IHS**
   - Port: 80
   - Installation path: `C:\IBM\HTTPServer`
   - Service name: `IBMHTTPServer6.0`
   - Deselect **Use secure protocol**

     Note: The Use secure protocol is used to enable secure transport for IBM HTTP Server Administration for an unmanaged node of WebSphere Application Server.

   - Plug-in installation directory: `C:\IBM\WebSphere\Plugins`

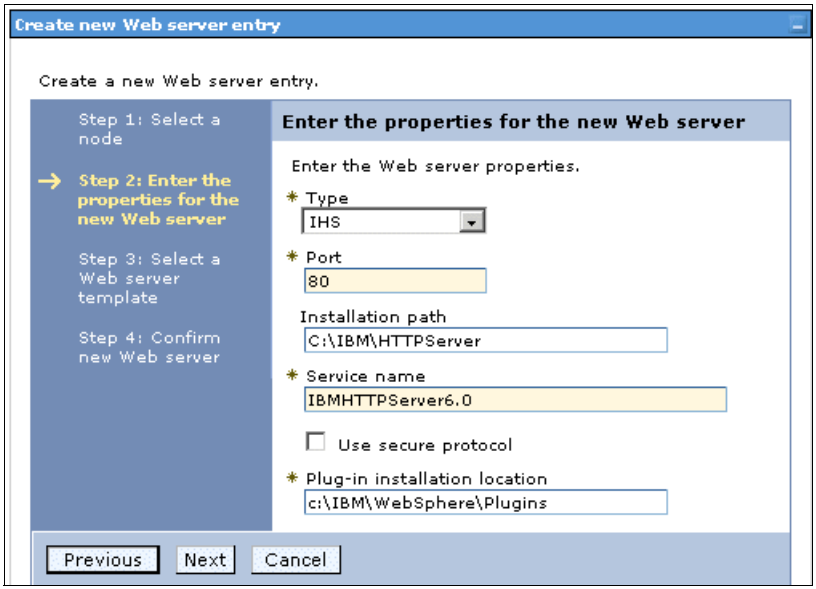


Figure 9-15 Create Web server: properties for Web server

6. Continuation of step 2, enter the following information, then click **Next**:
   - Port: 8008
   - User ID: `admin`

     Note: This User ID runs IBM HTTP Server as a Windows service.
   - Password: `<password>`
7. When the Select Web server template dialog box opens, select **IHS**, then click **Next**.
8. On the Summary dialog box, click **Finish**.
9. Click **Save**. Click **Save** again to save to master configuration.
10. Click **OK**.

Map modules to servers

In our example, we mapped the DefaultApplication for the purposes of verifying we can access applications such as snoop via the WebSphere plug-in.

Complete the following steps to map application modules to servers.

1. Select **Applications** → **Enterprise Applications**.
2. Click **DefaultApplication** (or whatever you wish to map).
3. Click **Map modules to servers**.

4. Check the modules, select the servers (use the Shift key for multiple servers) and click **Apply**. Click **OK**.
5. Click **Save**. Click **Save** again to save to master configuration.

Note: By default, the plugin-cfg.xml will automatically be regenerated.

Propagate plugin-cfg.xml to remote Web server

WebSphere Application Server V6.0 provided a Propagate Plug-in feature to copy the plugin-cfg.xml to the remote Web server. Alternatively, you can copy this file manually.

1. Generate a new Plug-in (plugin-cfg.xml).
   a. Expand the **Servers** tab, click **Web servers** from the Administrative Console.
   b. Check **webserver1**, and click **Generate Plug-in**.
2. In our example, we chose to copy the plugin-cfg.xml file manually:
   – From (Application Server node):

     ```
     C:\IBM\WebSphere\AppServer\profiles\Dmgr01\config\cells\itsoCell1\nodes\
     web1Node\servers\webserver1
     ```

   – To (IBM HTTP Server node):

     ```
     C:\IBM\WebSphere\Plugins\config\webserver1
     ```

3. When you have completed these tasks, restart IBM HTTP Server for new plugin-cfg.xml to be read immediately. The default refresh interval is 60 seconds.

Verify remote Web server redirect to WebSphere application

To verify the remote Web server is redirecting requests to WebSphere Application Server, do the following:

Enter the following URL in a Web browser:

```
http://web1.itso.ral.ibm.com/snoop
```

You should see the Snoop Servlet displayed. This verifies that the Web server is redirecting requests on port 80 to the application server.

9.4 Implement Travel Application Server node

The Travel Application Server node will be used to host the Travel application. The Travel application includes a Web services client to the ITSO Car Rental

application to make rental car reservations. This section describes the steps required to install WebSphere Application Server on this node to be used to host the Travel application.

9.4.1 Windows 2003 Server and system preparation

Refer to 9.3.1, “Microsoft Windows 2003 Server and system preparation” on page 295 for details.

9.4.2 WebSphere Application Server installation

In our example, we chose to use IBM WebSphere Application Server V6 (base) as our application server for the Travel company.

Complete the following tasks:

1. Install WebSphere Application Server V6.0
2. Install WebSphere Application Server V6.0.2 Refresh
3. Install WebSphere Application Server V6.0.2.7 Fixpack

Install WebSphere Application Server V6.0

This section describes how we installed IBM WebSphere Application Server V6.0 for our example runtime environment.

1. Run **`launchpad.bat`** to start the installer from the root of the WebSphere Application Server CD.

   Note: When installing software on a Microsoft Windows 2003 Server system, you might see the message, `The publisher could not be verified. Are you sure you want to run this software`? Click **Run** to continue.

2. When the Launchpad home dialog box opens, click **WebSphere Application Server Installation**.
3. Click **Launch the installation wizard for WebSphere Application Server**.
4. lick **Next** on the Welcome dialog box.
5. On the Software License Agreement dialog appears, review the terms and, if you are in agreement, select **I accept the terms in the license agreement**, and then click **Next**.
6. Click **Next** on the System prerequisites dialog box.

7. When the Installation directory dialog box opens, enter `C:\IBM\WebSphere\AppServer` in the Directory name field and click **Next**, as we did.
8. On the Setup type dialog box, select **Full installation** and click **Next**.
9. On the Installation Summary dialog box, review the selections and click **Next**.
10. On the Installation complete dialog box, deselect **First steps console**, and click **Finish**.
11. Close the WebSphere Application Server Launchpad home dialog box.

Install WebSphere Application Server V6.0.2 Refresh

Refer to “Install WebSphere Application Server V6.0.2 Refresh” on page 299.

Install WebSphere Java SDK V6.0.2.7 Fixpack

Refer to “Install WebSphere Java SDK V6.0.2.7 Fixpack” on page 300.

Install WebSphere Application Server V6.0.2.7 Fixpack

Refer to “Install WebSphere Application Server V6.0.2.7 Fixpack” on page 301.

Note: At this point in the chapter, you have completed the runtime environment setup required as a prerequisite for application deployment described in Chapter 10, “Deploy application to WebSphere Application Server” on page 351.

The next section in this section is required for monitoring the services using IBM Tivoli Composite Application Manager for SOA.

9.5 Implement ITSO Monitor Server node

This section describes how to install and configure of the ITSO Monitor Sever node (itsomon1). The umbrella name for the components is called IBM Tivoli Composite Application Manager (ITCAM) for SOA. To setup ITCAM for SOA you need to install Tivoli Management Services on the ITSO Monitor Server node (itsomon1) as well as installing monitoring agent on the ITSO Application Server node (itsoapp1).

Note: Chapter 12, “Manage and monitor services with ITCAM for SOA” on page 405 will demonstrate how to use ITCAM for SOA within the context of the Service Creation solution working example.

Complete the following tasks to install and configure ITCAM for SOA:

- Installation prerequisites and overview
- Steps for ITSO Monitor Server node (itsomon1):
  - Microsoft Windows 2003 Server and system preparation
  - IBM DB2 Universal Database installation
  - IBM Tivoli Enterprise Monitoring Server installation
  - ITCAM for SOA - Application Support installation
- Steps for the ITSO Application Server node (itsoapp1):
  - ITCAM for SOA Agent installation

9.5.1 Installation prerequisites and overview

This section includes the following installation prerequisites and overview information:

- Dependencies
- Software used for ITSO working example scenario
- Hardware and software prerequisites

Dependencies

There are some dependencies in the sequence in which the components should be installed. The components should be installed in the order we describe them.

Important: The assumption for this installation is that the monitoring server is to be installed on a separate system from the application server node where the agent runs.

If there is a requirement to install both the monitoring node and the application server node on the same system (such as for demonstration or proof-of-concept purpose), then extra consideration must be given during the installation process. Refer to the installation instructions in the online documentation for ITCAM for SOA at this URL:

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

- Install Tivoli Monitoring components on the Monitor Server node.
- Install Application Support for Tivoli Monitoring (ITCAM for SOA application support) on the Monitor Server node
- Install the ITCAM for SOA agent on the Application Server node

- Configure the ITCAM for SOA agent to monitor Application Server node
- Test and verify correct installation of agent
- Test and verify correct installation of Tivoli Monitoring

Software used for ITSO working example scenario

The table in Figure 9-5 lists the software CDs (or images) needed for the ITCAM for SOA installation. In our example, we used IBM DB2 UDB V8.2 + Fixpack 10 since this level was used on other nodes of our runtime environment.

Table 9-5 Software details for Installation of ITCAM for SOA

File Name	Description	Installation Node
	IBM DB2 Universal Database V8.2 + Fixpack 10	Monitor Server
C89XQIE	Tivoli Monitoring V6.1 for Windows: ► Tivoli Enterprise Monitoring Server ► Tivoli Enterprise Portal Server ► Tivoli Enterprise Portal Desktop Client ► Tivoli Enterprise Monitoring Agents – Tivoli Enterprise Monitoring Agent Framework – Universal Agent – Warehouse Proxy – Monitoring Agent for Windows OS – Summarization and Pruning Agent	Monitor Server
C881JIE	IBM Tivoli Monitoring - Application Support (ITCAM for SOA)	Monitor Server
C881IIE	Optional Install: IBM Web Services Navigator for Windows (Eclipse based). Can also be installed on an existing installation of IBM Rational Application Developer v6.x or other Eclipse based tools.	Monitor Server or Developer node
C881HIE	IBM Tivoli Composite Application Manager for SOA V6.0.0 (ITCAM for SOA Agent)	Application Server

Hardware and software prerequisites

For information on hardware and software prerequisites, refer to the *Installation and User's Guide, IBM Tivoli Composite Application Manager for SOA*, GC32-9492 found in the product InfoCenter at this Web site:

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

Table 9-6 lists account information used by default for the Tivoli components.

Table 9-6 Login accounts used by Tivoli components

User name	Password	Description
db2admin	<db2amin password> eg. db2admin	Logon as a DB2 DBA. Must have administrator privileges as described in "Microsoft Windows 2003 Server and system preparation" on page 286
admin on Windows or root on Linux/Unix	<OS administrator> eg. its0ral0 on Windows	Logon as system administrator for installation of Tivoli components. Must have correct privileges as described in "Microsoft Windows 2003 Server and system preparation" on page 286
ITMUser	Default: itmpswd1	Logon to Tivoli Warehouse database
TEPS	Default: no password	Logon to Tivoli Enterprise Portal Server database
sysadmin	Default: no password	Logon to as administrator to Tivoli Enterprise Portal Clients either through the desktop client or a web browser

9.5.2 Microsoft Windows 2003 Server and system preparation

Prior to the installation of software, you must ensure that correct user names with the appropriate privileges for DBA and system administration are set up. As a best practice, it is a good idea to create separate user names for DBA activities and for installing and configuring Tivoli components.

The process of creating user names with correct privileges has already been described in the previous sections. See 9.2.1, "Microsoft Windows 2003 Server and system preparation" on page 286.

9.5.3 IBM DB2 Universal Database installation

The IBM Tivoli Enterprise Monitoring Services V6.1 require the use of a database. IBM Tivoli Enterprise Monitoring Services supports IBM DB2 UDB, Microsoft SQL Server and Oracle.

For our scenario, we installed IBM DB2 UDB V8.2 + Fixpack 10 on the ITSO Monitor Server node (itsomon1). For installation details refer to 9.3.2, “DB2 Universal Database installation” on page 295.

9.5.4 IBM Tivoli Enterprise Monitoring Server installation

The section describes the procedure we used for Tivoli Enterprise Monitoring Server (TEMS) and its associated components in our example runtime environment. The installation of Tivoli Enterprise Monitoring Agent (TEMA) will be discussed separately.

Install Tivoli Enterprise Monitoring

Follow these steps to install the IBM Tivoli Monitoring components on the ITSO Monitor Server node (itsomon1):

1. Login to Windows with an administrator ID (for example, `admin`).
2. Launch the IBM Tivoli Monitoring installation wizard by running **`setup.exe`**.

> **Important:** On Microsoft Windows 2003 and Windows XP, the default security setting for software publisher might prevent installation process from proceeding. You might see a warning dialog box. Click **Run** to continue with the installation.
>
> If the installable software is on a Windows hard disk which is mapped as a drive over the network, clicking **Run** will not have any effect. The installation process will terminate quietly without any warnings.
>
> The workaround is to copy the installable software on to the local hard drive such as `c:\temp` and then restart the installation process.

3. When the Welcome to IBM Tivoli Monitoring dialog box opens, click **Next**.
4. On the Software License Agreement dialog box, review the terms and, if you are in agreement, click **Accept**.
5. On the Information dialog box, click **Next**.

> **Note:** The installation will perform a prerequisite check to verify that a supported database is installed. In our example, we have already installed DB2 UDB.
>
> If you have not installed a database, cancel the installation and install a database, then perform the installation of Tivoli Monitoring.

6. In the Choose Destination Location dialog box, accept the default (C:\IBM\ITM) and click **Next**, as we did.
7. On the User Data Encryption Key dialog box, accept the default key (IBMTivoliMonitoringEncryptionKey) and click **Next**.

> **Note:** The encryption key is used to establish a secure connection using SSL between the Hub TEMS and other components within the IBM Tivoli Monitoring system such as Remote TEMS connected to a hub.
>
> Do not use the special characters equal(=) comma (,) or single quote (‘) in the definition of your unique key.
>
> Take note of the key;- it will be used later in the installation.

8. On the Encryption Key confirmation dialog box, click **OK**.
9. On the Select Features dialog box, make the following selections as seen in Figure 9-16 on page 336, then click **Next**:
   - Select **Tivoli Enterprise Monitoring Agent.**

> **Note:** In order to analyze the offline data captured from an Agent installed and running on a client node, you need to install the *Warehouse Proxy*, and *Summarization and Pruning Agent* on the Tivoli Monitoring Server node.
>
> Refer to the following product guides if you need to install the Agent on the same system as the Monitoring Server:
>
> - *Installation and User’s Guide, IBM Tivoli Composite Application Manager for SOA*, GC32-9492
> - *Installation and Setup Guide, IBM Tivoli Monitoring Services V6.1,* GC32-9407-00

   - Select **Tivoli Enterprise Monitoring Server.**
   - Select **Tivoli Enterprise Portal Server.**
   - Select **Tivoli Enterprise Portal Desktop Server.**
   - Select **IBM Eclipse Help Server.**

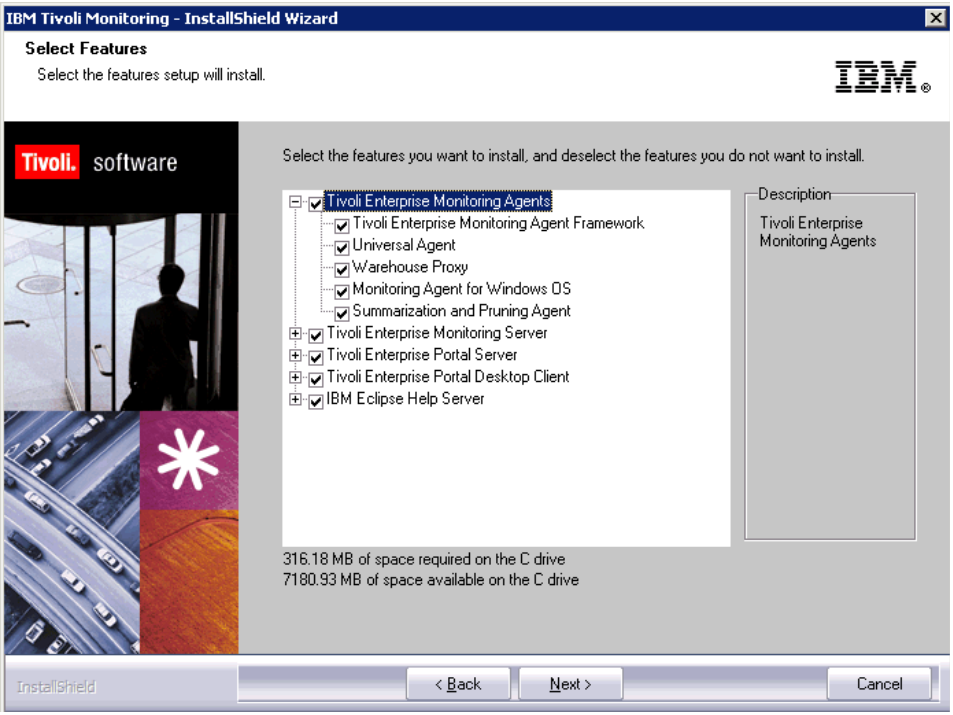


Figure 9-16 Choose the TEMS component to install

10. On the Agent Deployment dialog box, accept the default (none selected) and click **Next**.
11. On the Select Program Folder dialog box, accept the default and click **Next**.
12. On the Start Copying Files dialog box, review your selections and click **Next** to begin copying files.

Configure Tivoli Enterprise Monitoring

After the IBM Tivoli Enterprise Monitoring installation is complete, the configuration dialog box is launched. This section describes how we configured IBM Tivoli Enterprise Monitoring for the working example runtime environment.

Note: After the installation is complete, you can modify the configure by launching the Tivoli Enterprise Monitoring Console by clicking **Start** → **Programs** → **Tivoli Enterprise Monitoring** → **Console**.

13. In our example, we are continuing from the install directly into the configuration. When the Setup Type dialog box opens, accept the default settings as seen in Figure 9-17, and then click **Next**:

> **Attention:** The configuration of Tivoli Enterprise Portal and Tivoli Monitoring Server is mandatory.

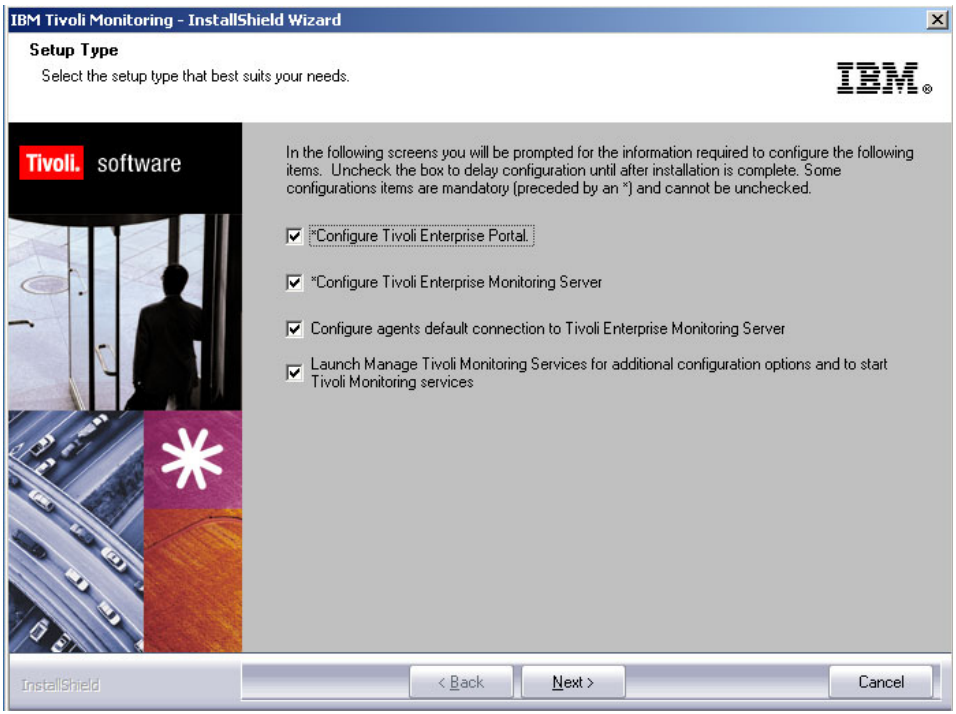


Figure 9-17 Tivoli Enterprise Monitoring Server - configuration

14. On the Define TEP Host Information dialog box, enter the hostname of the machine where TEP server resides. The default server for our example is `ITSOMON1`. The server name was detected since our Portal Server is install on the same node. Click **Next**.

15. On the TEPS Data Source Config Parameters - DB2 dialog box, enter the following information as seen in Figure 9-18 on page 338, and then click **OK**:

    Database Administrator ID:

    – Admin User ID: `db2admin`
    – Admin Password: `<db2admin_password>`

Database User ID (used by Tivoli Enterprise Portal Server):

– Database User ID: `TEPS`

> **Note:** The database user ID will be created by this dialog box, provided the Database Administrator ID is a valid DB2 UDB administrator ID and password. We recommend that you specify the `<TEPS_password>`. Otherwise, the default password of `itmpswd1` will be used.

– Database Password: `<TEPS_password>`
– Reenter Password: `<TEPS_password>`

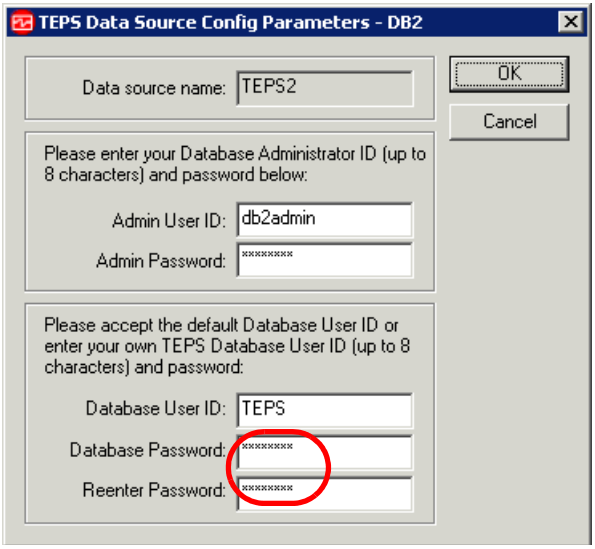


Figure 9-18 Settings for TEPS Data Source Config Parameters - DB2

> **Note:** The TEPS user is created in Windows 2003 Server, TEPS database is created in DB2 UDB, and the data source (ODBC) is configured in Windows 2003 Server.

16. On the TEPS configuration completed successfully dialog box, click **OK**.
17. When the Warehouse ID and Password for TEP Server dialog box opens, enter the following, and then click **Next**:

– ID: `ITMUser`
– Password: `<ITMUser_password>`

Note: We recommend that you specify the `<ITMUser_password>`. Otherwise, the default password of `itmpswd1` will be used.

– Confirm Password: `<ITMUser_password>`

18. On the TEP Server Configuration (Protocol Selection) dialog box, enter the he following information (see Figure 9-19 on page 339), then click **OK**:

– Check Protocol 1: select **IP.PIPE** (default)

Note: The Tivoli Enterprise Portal Server and Tivoli Enterprise Monitoring Server must be configured to use the same protocol.

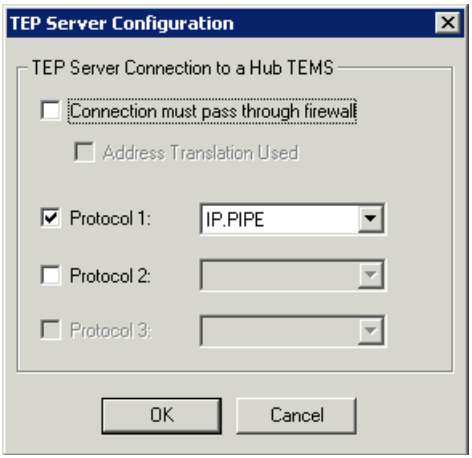


Figure 9-19 TEP Server Configuration (Protocol selection)

19. On the TEP Server Configuration (Protocol Settings) dialog box, enter the Hostname or IP address (for example, `ITSOMON1`), and Port number (for example, `1918`), as seen in Figure 9-20 and then click **OK**.

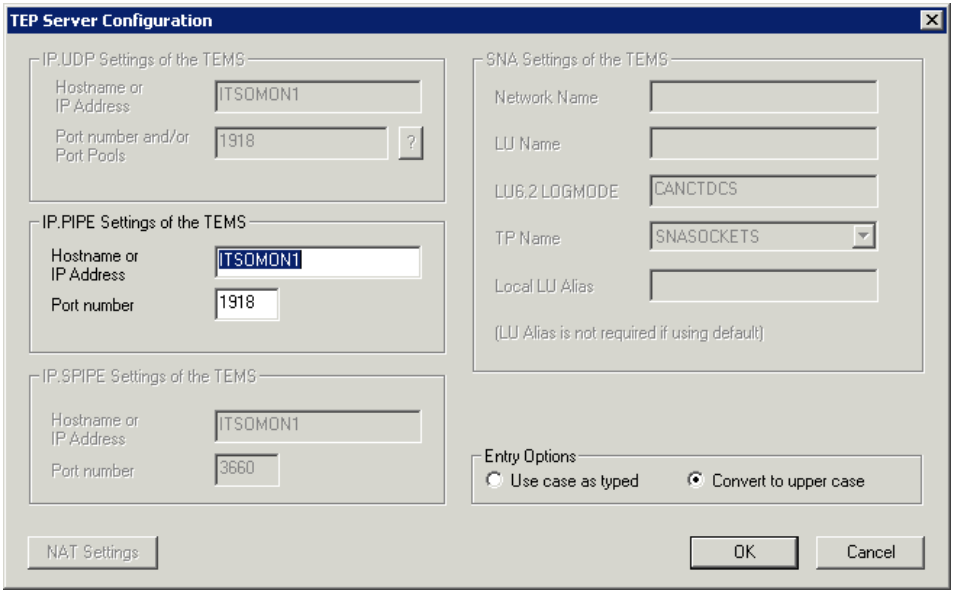


Figure 9-20 TEP Server Configuration (Protocol Settings)

20. On the confirmation dialog box for the warehouse connection information, click **Yes**.
21. When the Warehouse Proxy Database Selection dialog box opens, select **DB2** and click **OK**.
22. When the Configure DB2 Datasource for Warehouse Proxy dialog box opens, enter the following as seen in Figure 9-21 on page 341, and then click **OK**:
    - Warehouse information:
      - Data Source Name: `ITM Warehouse` (default)
      - Database Name: `Warehous` (default)
    - Database Administrator ID:
      - Admin User ID: `db2admin`
      - Admin Password: `<db2admin_password>`
    - Database User ID (Warehouse user ID):
      - Database User ID: `ITMUser`

> **Note:** The database user ID will be created by this dialog box, provided the Database Administrator ID is a valid DB2 UDB administrator ID and password. We recommend that you specify the `<ITMUser_password>`. Otherwise, the default password of `itmpswd1` will be used.

– Database Password: <ITMUser_password>
– Reenter Password: <ITMUser_password>

Note: The ITMUser user is created in Windows 2003 Server, Warehous database is created in DB2 UDB, and the data source (ODBC) is configured in Windows 2003 Server.

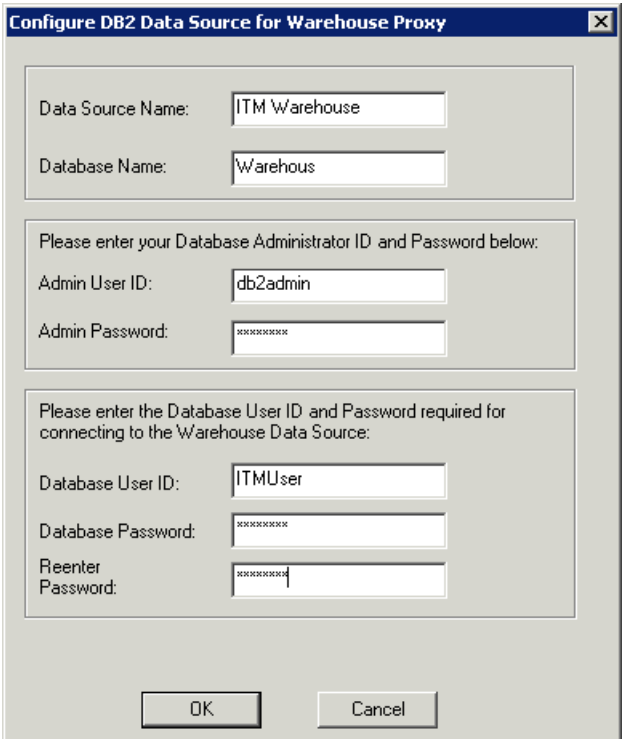


Figure 9-21 Configure DB2 Datasource for Warehouse Proxy

23. When the Successfully configured warehouse data source dialog box opens, click **OK**.
24. When the Tivoli Enterprise Monitoring Server Configuration dialog box opens, we accepted the default settings as seen in Figure 9-22 on page 342, click **OK**:

Note: The protocol IP.PIPE should be used for TEMS since we have previously defined this protocol. If you are using TEMS V6.1 w/ FP001, the default will be to have Security Validate User checked.

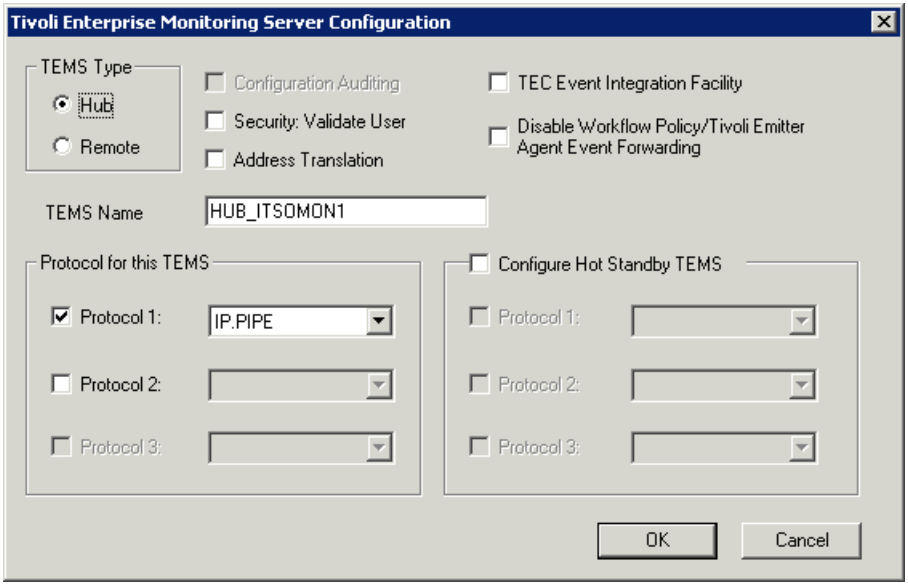


Figure 9-22 Tivoli Enterprise Monitoring Server Configuration

25. When the Hub TEMS Configuration dialog box opens, accept the defaults (see Figure 9-23 on page 342) and click **OK**.

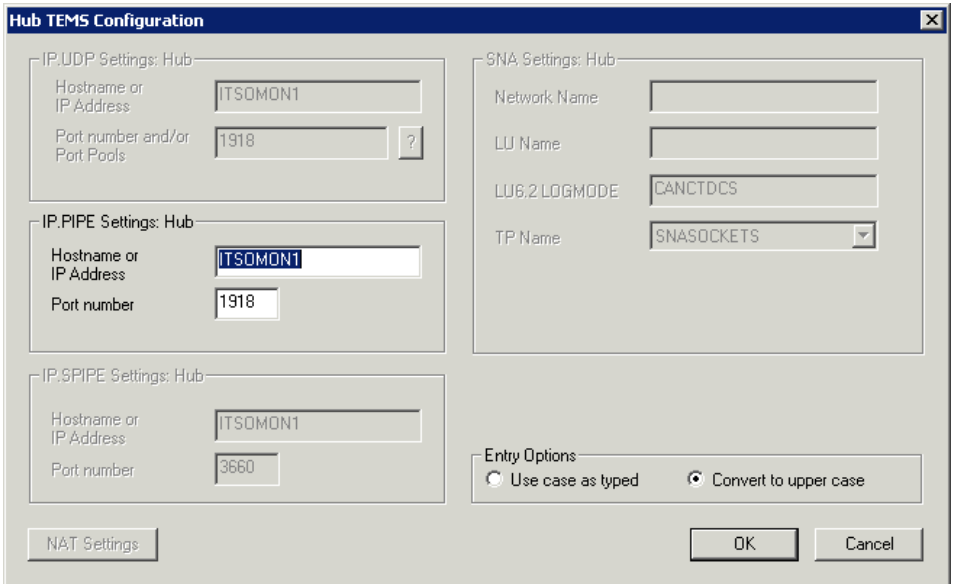


Figure 9-23 Hub TEMS Configuration

26. When the Add application support to the TEMS dialog box opens, select **On this computer** (default) for the location of TEMS, and click **OK**.
27. Click **OK** on the information dialog box in Figure 9-24.

Figure 9-24 Start Tivoli Enterprise Monitoring Server

28. When the Select the application support to add to the TEMS dialog appears, make the selections as seen in Figure 9-25, and then click **OK**.

> **Note:** If you chose to install extra components such as agents for Data Warehouse, then you will see a dialog box like Figure 9-25.

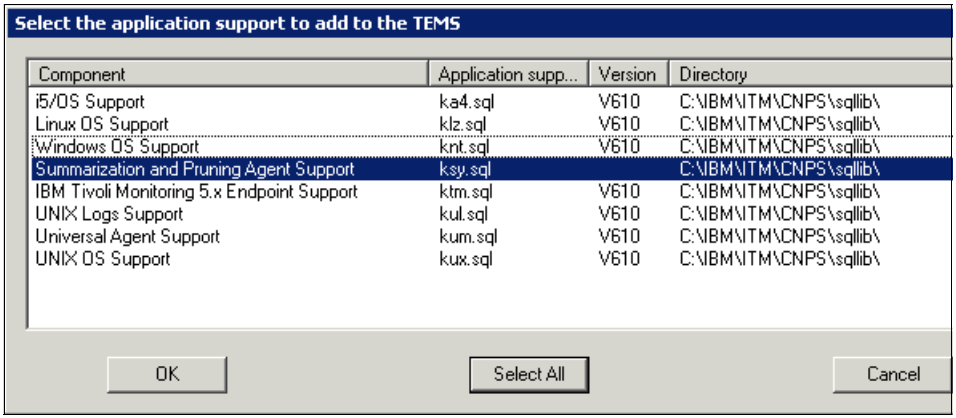


Figure 9-25 Select the application support to add to the TEMS

29. On the Application support addition complete dialog box, click **Next**.
30. When the Configuration Defaults for Connecting to a TEMS (Select Protocol) dialog box opens, accept the defaults, and then click **OK**.
31. On the Configuration Defaults for Connecting to a TEMS (Protocol Settings) dialog box opens, accept the defaults, then click **OK**.
32. When the InstallShield Wizard Complete dialog box opens, click **Finish**.

33. After clicking Finish, the Manage Tivoli Enterprise Monitoring Services - TEMS Mode is launched (see Figure 9-26 on page 344). You can see the services and status. Close by clicking **Actions** → **Exit**.

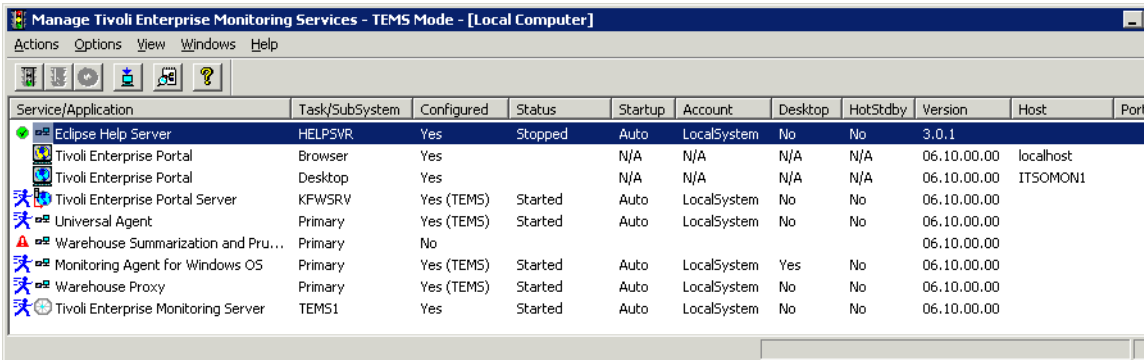


Figure 9-26 Manage Tivoli Enterprise Monitoring Services - TEMS Mode (aka Console)

9.5.5 ITCAM for SOA - Application Support installation

This section describes how to install the ITCAM for SOA - Application Support (aka ITCAM for SOA server component). In order for ITCAM for SOA Agent to communicate with IBM Tivoli Enterprise Monitoring Server, you must install the ITCAM for SOA Application Support on the Monitoring Server node.

To install ITCAM for SOA - Application Support, follow these steps:

1. To launch the installer, run **`setupwin32.exe`**.
2. When the InstallShield Wizard for IBM Tivoli Monitoring - Application Support appears, click **Next**.
3. When the Installation directory dialog box opens, accept the default (C:\IBM\ITM) and click **Next**.
4. When the Features dialog box opens, accept the default (TEMS, TEPS, TEPD), and click **Next**.
5. When the Applications to select dialog box opens, check **IBM Tivoli Composite Application Manager for SOA**, then click **Next**.
6. When the Install summary dialog box opens, click **Next** to begin copying files.
7. When the Installation completed successfully message is displayed, click **Finish**.

Note: If there are any problems during the installation, check the following log file for errors: `c:\Documents and Settings\<login name>\Local Settings\Temp\ITM_AppSupport_Intall.log`

9.5.6 ITCAM for SOA Agent installation

The ITCAM for SOA agent must be installed on the application server node, where Web Services are required to be monitored. In our example, we will install the ITCAM for SOA Agent on the ITSO Application Server node (itsoapp1).

Install ITCAM for SOA

To install the ITCAM for SOA Agent, follow these steps the following:

1. Login to Windows with an administrator ID (for example, `admin`).
2. Launch the installation wizard by running **`setup.exe`**.
3. When the Welcome to IBM Tivoli Composite Application Manager for SOA opens, click **Next**.
4. When the Prerequisites dialog opens, click **Next**.

   Attention: The prerequisites screen refers to OMEGAMON platform and Candle® Management Server. This really means Tivoli Enterprise Monitoring Services.

5. On the Software License Agreement dialog box, review the terms and, if you are in agreement click **Accept**.
6. When the Information dialog box opens, click **Next**.
7. On the Choose Destination Location dialog box, you must accept the default C:\Candle, and click **Next**.

   Attention: At the time of writing, we found changing the Location path from its default c:\Candle, resulted in the Agent not working properly after the installation. Although special manual changes might be possible, we recommend you accept the default location path.

8. On the Select Features dialog box, check **Agent Support** (ITCAM for SOA Agent under this feature), and click **Next**.
9. When the Select Program Folder dialog box opens, accept the default and click **Next**.

10. On the Start Copying Files dialog box, review the selections and the click **Next** to begin copying files.

Configure ITCAM for SOA Agent

After the ITCAM for SOA Agent installation is complete, the configuration dialog box is launched. This section describes how we configured ITCAM for SOA for the working example runtime environment.

11. When the Setup Type dialog box opens, accept the default (see Figure 9-27) and then click **Next**.

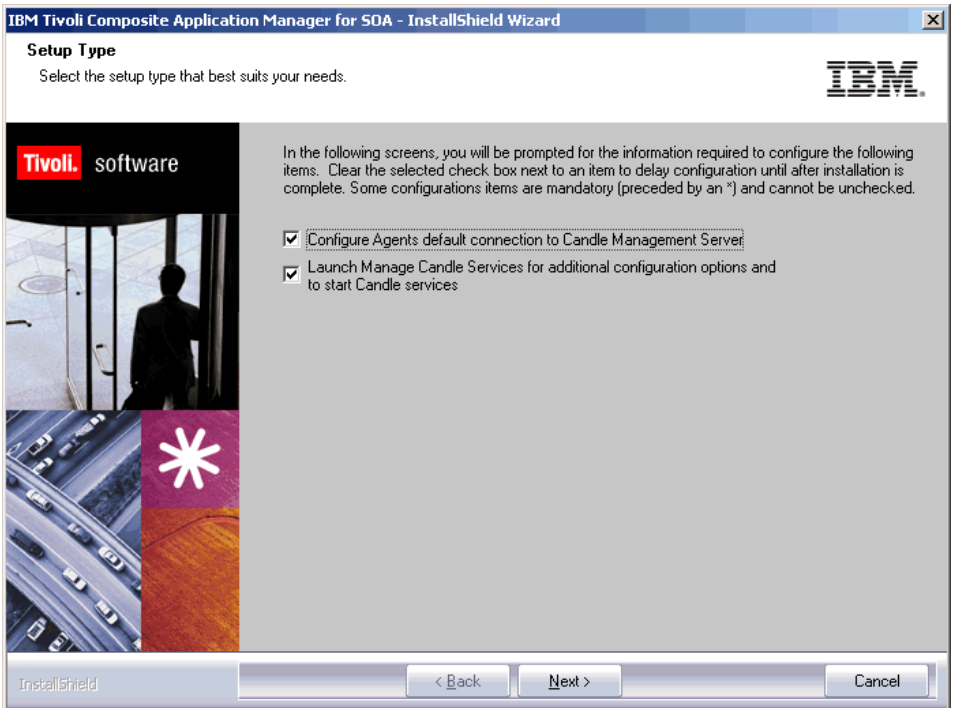


Figure 9-27 ITCAM for SOA Agent

12. On the Configuration Defaults for Connecting to a CMS (Select Protocol) dialog box, select **IP.PIPE** in the Protocol 1 drop down list (see Figure 9-28 on page 347). The protocol used by the Agent must match the protocol defined for TEMS. In our example, we configured TEMS to use the IP.PIPE protocol. After selecting the protocol, click **OK**.

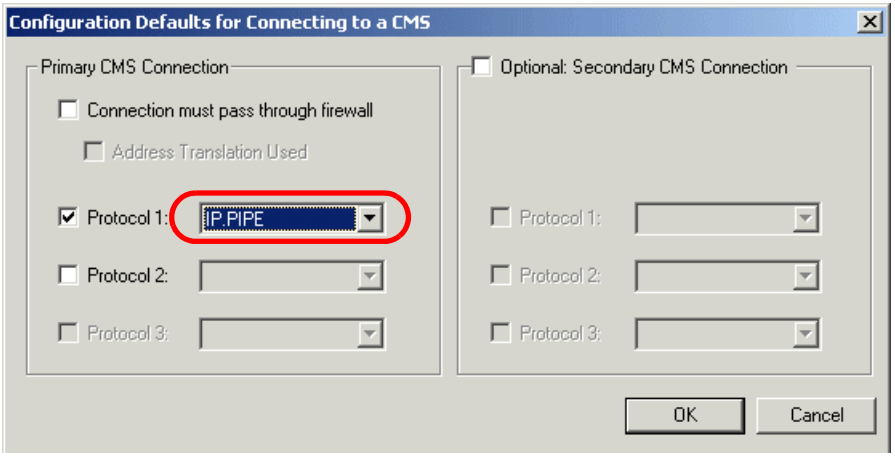


Figure 9-28 Configuration Defaults for Connecting to a CMS (Select Protocol)

13. When the Configuration Defaults for Connecting to a CMS (Protocol Settings) dialog box opens, enter the hostname or IP address of the TEMS node. For example, we entered `ITSOMON1` in the hostname field (see Figure 9-29 on page 348) and then clicked **OK**.

> **Note:** Ensure your Monitor Server node can be resolved by name before proceeding.

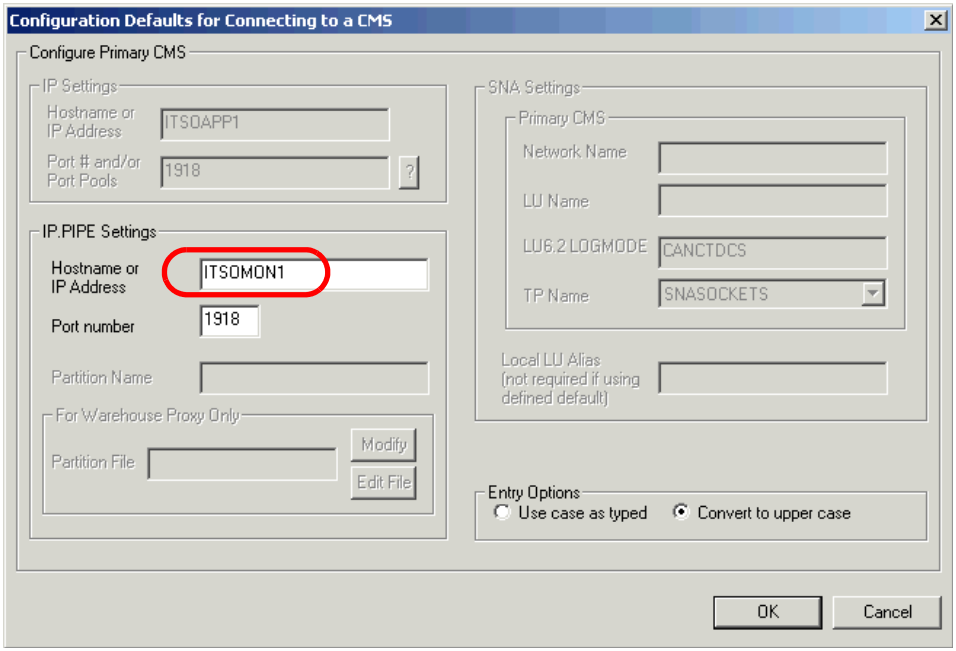


Figure 9-29 Configuration Defaults for Connecting to a CMS (Protocol Settings)

14. When the InstallShield Wizard Complete dialog box opens, click **Finish**.
15. The Manage Candle Services - CMS Mode is launched. Select **Actions** → **Configure Using Defaults**.
16. Right-click **ITCAM for SOA** and select **Start**. That status should change to Started as seen in Figure 9-30.

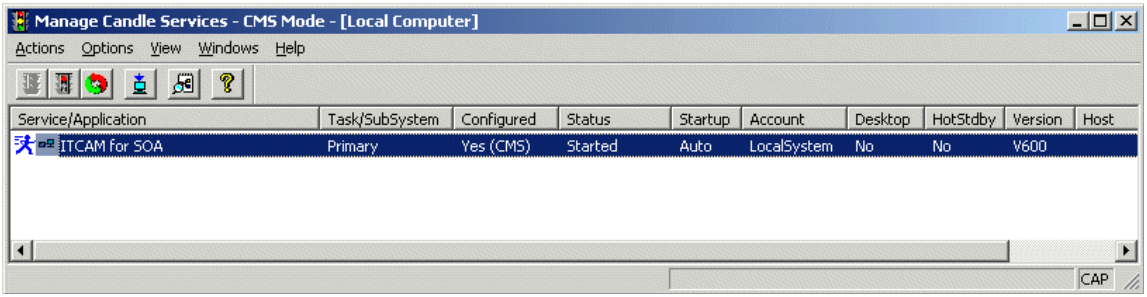


Figure 9-30 ITCAM for SOA Agent Console

17. Verify the ITCAM for SOA Agent is communicating with the ITCAM for SOA Server.
    a. After the ITCAM for SOA Agent has started, select **Advanced** → **View Trace Log** from the ITCAM for SOA Agent Console.
    b. Scroll through the trace log to look for connection failure. Viewing the trace log is a useful troubleshooting technique should you have problems with your Agent and Server configuration.

Enable ITCAM for SOA Agent for data collection

After the installation and configuration of ITCAM for SOA Agent has completed, the Agent needs to be enabled for data collection for the target application server.

To enable ITCAM for SOA Agent to collect data for the application server, do the following:

1. Ensure the ITSO Application Server node, server1 application server (ITSO Car Rental application is deployed) is started.
2. On the Application Server node, open a Windows command window and navigate to the C:\Candle\CMA\KD4\bin directory.
3. Enter the following command to enable the Agent for WebSphere Application Server:

   ```
   KD4configDC -enable -env 1 C:\IBM\WebSphere\AppServer
   ```

   Syntax:

   ```
   KD4configDC -enable -env x <application server specific arguments>
   ```

   Where, `-env x` specifies the application environment to configure; the value of x is one of the following:

   ```
   1 = IBM WebSphere Application Server
   2 = Microsoft .Net platform
   3 = BEA WebLogic Server
   ```

   To get more help on the `KD4configDC` command, enter the following:

   ```
   KD4configDC -h
   ```

   If WebSphere Application Server was installed in a directory under `c:\Program Files`, the arguments need to enclosed in double quotes:

   ```
   KD4configDC -enable -env 1 "C:\Program Files\WebSphere\Appserver"
   ```

4. Verify the KD4configDC enabled WebSphere Application Server to use the Agent data collector as a JAX-RPC handler. Ensure the file `kd4dcagent.jar` is copied into the `<WAS_HOME>\lib\ext` directory.
5. After the Agent has been enabled, you must restart the application server (for example, server1).

Verify and test the ITCAM for SOA Agent

After running the `KD4configDC` script, perform the following steps to verify that the ITCAM for SOA Agent is working properly:

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

   ```
   C:\Windows\system32\drivers\etc
   ```

2. Verify that the file named KD4BaseDirConfig.properties exists in the directory. This file was created as part of the installation.
3. Open the KD4BaseDirConfig.properties file in text editor, and verify the following entry exists:

   ```
   INSTALLDIR=C:\\CANDLE\\CMA\\
   ```

   Note: The value in the INSTALLDIR variable represents the directory location where the \KD4 folder is located. The \KD4 folder contains logging and properties files used with ITCAM for SOA.

   If by chance you did not follow our recommendation to accept the default installation path (C:\Candle), you can modify the INSTALLDIR value in this file to get the Agent working.

10

Deploy application to WebSphere Application Server

This chapter provides a procedure for deploying the ITSO Car Rental and Global Travels applications to the ITSO working example runtime environment.

The chapter is organized into the following sections:

- Application deployment prerequisites
- Deploying the ITSO Car Rental application
- Deploying Global Travels application

10.1 Application deployment prerequisites

Prior to deploying the applications, you need to ensure the following prerequisites have been met:

- Installing and configuring runtime environment software
- Downloading the sample code
- Starting the servers

10.1.1 Installing and configuring runtime environment software

The details on how to install and configure the runtime environment can be found in Chapter 9, “Implement the runtime environment” on page 281.

10.1.2 Downloading the sample code

The deployment procedure references EAR files and database scripts supplied with the additional material for this book.

For details, refer to Appendix E, “Additional material” on page 515.

10.1.3 Starting the servers

This section describes how to start the servers on each of the nodes of the ITSO working example runtime environment in preparation for application deployment.

Starting servers on the ITSO Application Server node

Follow these steps to start the servers required to deploy the application on the ITSO Application Server node:

1. Start the Deployment Manager:

   ```
   cd \IBM\WebSphere\AppServer\profiles\Dmgr01\bin\
   startManager
   ```

   Look for a message similar to this one if the server starts successfully:

   ```
   Server dmgr open for e-business
   ```

   Tip: The command to stop the Deployment Manager is the followings:

   ```
   cd \IBM\WebSphere\AppServer\profiles\Dmgr01\bin\
   stopManager
   ```

2. Start the node agent for the federated Application Server node:

   ```
   cd \IBM\WebSphere\AppServer\profiles\AppSrv01\bin
   startNode
   ```

Look for a message similar to this one if the node starts successfully:

```
Server nodeagent open for e-business
```

Note: The node agent must be started before the application server - server1 can be started. By default, the node agent Windows service is set to start at startup.

Tip: The command to stop the node agent is:

```
cd \IBM\WebSphere\AppServer\profiles\AppSrv01\bin\
stopNode
```

3. Start the server1 application server for the Application Server as follows:

   ```
   cd \IBM\WebSphere\AppServer\profiles\AppSrv01\bin
   startServer server1
   ```

   Look for a message similar to this one if the server1 application server starts successfully:

   ```
   Server server1 open for e-business
   ```

   Tip: The command to stop the server1 application is the followings:

   ```
   cd \IBM\WebSphere\AppServer\profiles\AppSrv01\bin\
   stopServer server1
   ```

Starting servers on the Web Server Redirector node

To start the servers on the Web Server Redirector node, follow these steps:

1. Click **Start** → **Settings** → **Control Panel** → **Administrative Tools** → **Services**.
2. Right-click **IBM HTTP Server 6.0** and click **Start**.
3. Right-click **IBM HTTP Administration 6.0** and click **Start**.

Starting servers on the Global Travels Application Server node

Start the server1 application server for the Application Server profile on the Global Travels Application Server node:

```
cd \IBM\WebSphere\AppServer\profiles\AppSrv01\bin
startserver server1
```

Look for a message similar to this one if the server1 application server for the Global Travels Application Server node starts successfully:

```
Server server1 open for e-business
```

10.2 Deploying the ITSO Car Rental application

Complete the following steps to deploy the ITSO Car Rental application to WebSphere Application Server Network Deployment on the ITSO Application Server node:

1. Installing the ITSO Car Rental enterprise application
2. Creating the ITSO Car Rental application database
3. Creating the ITSO Car Rental application database data source
4. Installing the ITSO Car Rental Web Services client application
5. Generating new plugin-cfg.xml
6. Verifying the ITSO Car Rental application
7. Creating a gateway service

10.2.1 Installing the ITSO Car Rental enterprise application

To install the ITSO Car Rental enterprise application, follow these steps:

1. Start the Deployment Manager Administrative Console by entering the following URL in a Web browser for the ITSO Application Server node:

   `http://itsoapp1.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 **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:\7240code\deploy\ITSOCarRental.ws\ITSOCarRental.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, complete these tasks, then click **Next**:
   - Check **Deploy enterprise beans**
   - Check **Deploy Web services**
   - Accept default settings for remaining options
7. When the Step 2: Map modules to servers dialog box opens, select these options, then click **Next**:
   - Select the following using the shift key:

     WebSphere:cell=itsoCell1,node=web1Node,server=webserver1

     WebSphere:cell=itsoCell1,node=itso1Node,server=server1

- Check **ITSOCarRentalEJB** and **ITSOCarRentalWeb.**
- Click **Apply.**
- Click **Next.**

8. When the Step 3: Provide options to perform EJB Deploy dialog box opens, select the following option, then click **Next**:
   - Deploy EJB option - Database type: select **DB2UDB_V82**
9. When the Step 4: Provide JNDI Names for Beans dialog box opens, accept the default, 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/itsodb and is defined in the application deployment descriptor.

    You might see an Application Resources Warning dialog box regarding the scope. Click **Continue**.
11. For the next few dialog boxes, accept the defaults and click Next.
    a. Step 6: Map data sources for 2.x CMP beans, accept the default (data source already defined at module level), and click **Next**.
    b. Step 7: Map EJB references to beans, accept default and click **Next**.
    c. Step 8: Map virtual hosts for Web modules, accept default and click **Next**.
    d. Step 9: Ensure all unprotected 2.x methods have the correct level of protection, accept the defaults and click **Next**.
    e. Step 10: Provide options to perform the Web services deployment, accept the defaults and click **Next**.
12. When the Step 11: Summary dialog box opens, review the settings then click **Finish**.

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

    ```
    Application ITSOCarRental installed successfully
    ```
13. Click **Save to Master Configuration**.
14. When prompted, check **Synchronize changes with Nodes** and click **Save**. Click **OK**.

Next we will create the ITSO Car Rental application database and then create the data source through the Administrative Console.

10.2.2 Creating the ITSO Car Rental application database

To create the ITSO Car Rental application database, 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 itsodb
   ```

3. Connect to the itosdb database.

   ```
   db2 connect to itsodb
   ```

4. Create tables.

   Enter the following command to run the **`Table.ddl`** to create database tables:

   ```
   db2 -tvf C:\7240code\deploy\ITSOCarRental.ws\database\db2\Table.ddl
   ```

   Note: Don't use another `Table.ddl` under `C:\7240code\assemble\ITSOCarRental.j2ee\database\db2\`. This ddl doesn't work on this chapter.

5. Populate the tables with sample data.

   Enter the following commands to run the **`loadData.sql`** to populate the database with sample data:

   ```
   db2 -tvf C:\7240code\deploy\ITSOCarRental.ws\database\db2\loadData.sql
   ```

6. Disconnect from the itsodb database.

   ```
   db2 disconnect itsodb
   ```

10.2.3 Creating the ITSO Car Rental application database data source

This section describes how to configure the ITSO Car Rental application database data source.

Complete the following tasks:

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

Creating 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: `itsodbalias`
   - User ID: `admin`

   Note: In our example, we created the admin user ID as an administrative user (see "Microsoft Windows 2003 Server and system preparation" on page 286). By default, the user logged into Windows when the database is created is used as the database owner.

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

Creating 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 **itsoNode1** 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 **XA 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.

Creating 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 (XA)**.
3. Click **Data sources** under Additional Properties.
4. Click **New**.
5. Enter the following:
   - Name: `itsodbDS`
   - JNDI name: `jdbc/itsodb`
   - Component-managed authentication alias: **itsoCellMgr1/itsodbalias**
   - Database name: `itsodb`
   - 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: `itsoapp1.itso.ral.ibm.com`

     This is the hostname of the server where DB2 Universal Database is installed.

   - Port number: `50000`

     This is the port defined in the services file for the DB2 Universal Database 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(XA)**.

3. Click **Data sources**.
4. Check the data source and click **Test connection**. Look for a successful message.

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

10.2.4 Installing the ITSO Car Rental Web Services client application

Follow these steps to install the ITSO Car Rental Web Services client application:

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:\7240code\deploy\ITSOCarRental.ws\ITSOCarRentalWS.ear`
3. On the Generate default bindings dialog box, accept the defaults then click **Next**.
4. On the Step 1: Select installation options dialog box, complete these tasks then click **Next**:
   - Check **Deploy Web services**
   - Accept default settings for remaining options
5. When the Step 2: Map modules to servers dialog box opens, complete these tasks:
   - Select these options using the Shift key:

     WebSphere:cell=itsoCell1,node=web1Node,server=webserver1

     WebSphere:cell=itsoCell1,node=itsoNode1,server=server1
   - Check **ITSOCarRentalWSWeb.**
   - Click **Apply**, then click **Next.**
6. When the Step 3: Map virtual hosts for Web modules dialog box opens, accept the default and click **Next**.
7. When the Step 4: Provide options to perform the Web Services deployment dialog box opens, accept the default and click **Next**.

8. When the Step 5: Summary dialog box opens, review the settings then click **Finish**.

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

   ```
   Application ITSOCarRentalWS installed successfully
   ```

9. Click **Save to Master Configuration**.
10. When prompted, check **Synchronize changes with Nodes** and click **Save**. Click **OK**.
11. Restart the server1 application server.

10.2.5 Generating new plugin-cfg.xml

After the application is deployed, we must generate a new WebSphere plugin-cfg.xml. The new plugin-cfg.xml contains the application URIs needed to redirect HTTP requests from the Web Server Redirector to the ITSO Application Server.

1. Generate a new Plug-in (plugin-cfg.xml).
   a. Expand the **Servers** tab, click **Web servers** from the Administrative Console.
   b. Check **webserver1**, and click **Generate Plug-in**.

   Note: The plugin-cfg.xml should be generated automatically when the configuration changes require and update. The above procedure is a method of generating manually.

2. Copy the plugin-cfg.xml file manually:
   – From (Application Server node):

     ```
     C:\IBM\WebSphere\AppServer\profiles\Dmgr01\config\cells\itsoCell1\nodes\
     web1Node\servers\webserver1
     ```

   – To (IBM HTTP Server node):

     ```
     C:\IBM\WebSphere\Plugins\config\webserver1
     ```

3. When you are finished, restart IBM HTTP Server V6.0 Windows service for the new plugin-cfg.xml to be read immediately. The default refresh interval is 60 seconds.

10.2.6 Verifying the ITSO Car Rental application

Verify the ITSO Car Rental application by accessing it with a Web browser and creating a reservation in the application.

1. Verify the base ITSO Car Rental Web application can be accessed to verify database access:

   `http://web1.itso.ral.ibm.com/ITSOCarRentalWeb/login.jsp`

2. Verify the ITSO Car Rental Web services client application is working (verify Web services):

   `http://web1.itso.ral.ibm.com/ITSOCarRentalWSWeb/login.jsp`

Refer to Chapter 11.3, “ITSO Car Rental application walkthrough” on page 383 for details.

10.2.7 Creating a gateway service

The gateway service is associated with a single external service, defined by the WSDL supplied when creating the gateway service.

To create the gateway service, follow these steps on the ITSO Application Server node:

1. Expand **Service Integration** and click **Buses** from the Administrative Console.
2. Click **WSGBus**.
3. Click **Web service gateway instances** under Additional Properties.
4. Click **WSG**.
5. Click **Gateway services** under Additional Properties.
6. Click **New**.
7. Select **WSDL-defined web service provider** as the type of target service, shown in Figure 10-1, then click **Next**.

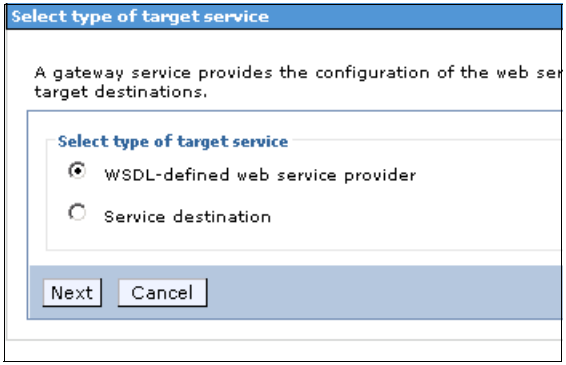


Figure 10-1 Selecting the type of target service

8. When the Step 1: Specify gateway services name, services destinations and mediations dialog box opens, enter the following information then click **Next**:
   - Gateway service name: `WSGService` (This is an arbitrary name.)
   - Gateway request destination name: `WSGServiceRequestDest`

     This destination processes the request messages for the gateway.
   - Gateway response destination name: `WSGServiceResponseDest`

     This destination processes the response messages for the gateway.

   Note: In WebSphere Application Server Network Deployment, a *destination* is defined as a virtual location within a service integration bus. Applications can attach as producers, consumers or both to exchange messages.

9. When the Step 2: Locate the target service WSDL page opens (Figure 10-2), select a URL for the WSDL location type, enter the following value for the WSDL location, then click **Next**:

   `http://itsoapp1.itso.ral.ibm.com:9080/ITSOCarRentalWeb/services/CarRentalManager/wsdl/CarRentalManager.wsdl`

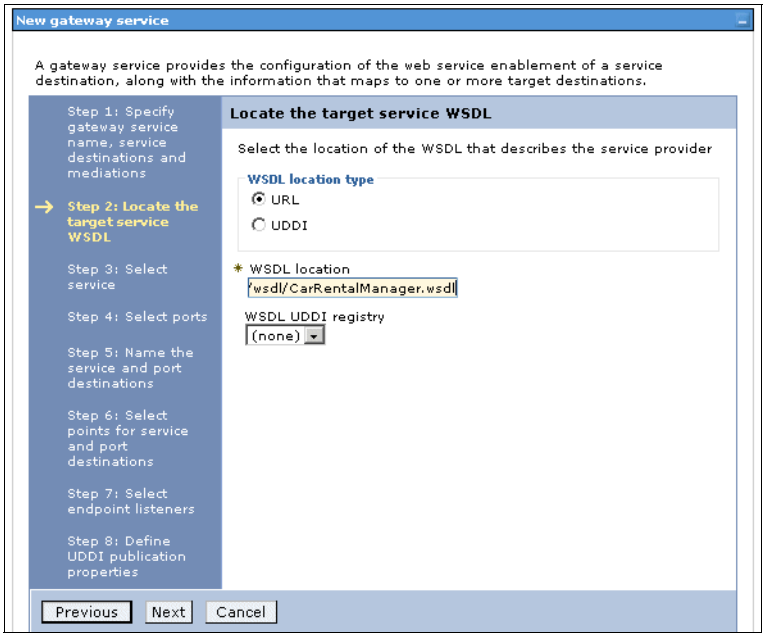


Figure 10-2 Locating the target service WSDL

10. When the Step 3: Select service dialog box opens, select the service in the WSDL to configure. The ITSO Car Rental application exposes one service (three operations), so we can accept the default value, and click **Next**.
11. When the Step 4: Select ports dialog box opens, accept the default port and click **Next**.
12. When the Step 5: Name the service and port destinations dialog box opens, the external service is configured as an outbound service. We associate its destination with the gateway service destination created earlier. In the lab, we accepted the following default values and clicked **Next**:
    - Outbound service name: `CarRentalManagerService`
    - Service destination name: `http://carrental.itso.ral.ibm.com:CarRentalManagerService`
    - Port destination name: `http://carrental.itso.ral.ibm.com:CarRentalManagerService`
13. When the Step 6: Select points for service and port destinations dialog box opens, select the bus member then click **Next**.
14. When the Step 7: Select endpoint listeners dialog box opens, select the endpoint listener (`itsoNode1:server1:SOAPHTTPChannel1`) and click **Next**.

15. We can now select the endpoint listener we created earlier by accepting the default value and clicking **Next**.
16. For our scenario, we are not be publishing to a UDDI registry, so click **Finish** on the final page.
17. Click **Save**. Click **Save** again to save to master configuration.
18. Restart the Deployment Manager and Application servers.

Renaming the inbound port

Rename the inbound port as a best practice to a more meaningful (and shortened) name, by completing the following steps:

1. Expand **Service Integration** and click **Buses**.
2. Click **WSGBus**.
3. Under Services, click **Inbound Services**.
4. Click **WSGService**.
5. Click **Inbound Ports** under Additional Properties.
6. Click **itsoNode1_server1_SOAPHTTPChannel1_InboundPort**.
7. Replace itsoNode1_server1_SOAPHTTPChannel1_InboundPort with **SOAPHTTPChannel1_InboundPort** (Figure 10-3 on page 365) then click **OK**.

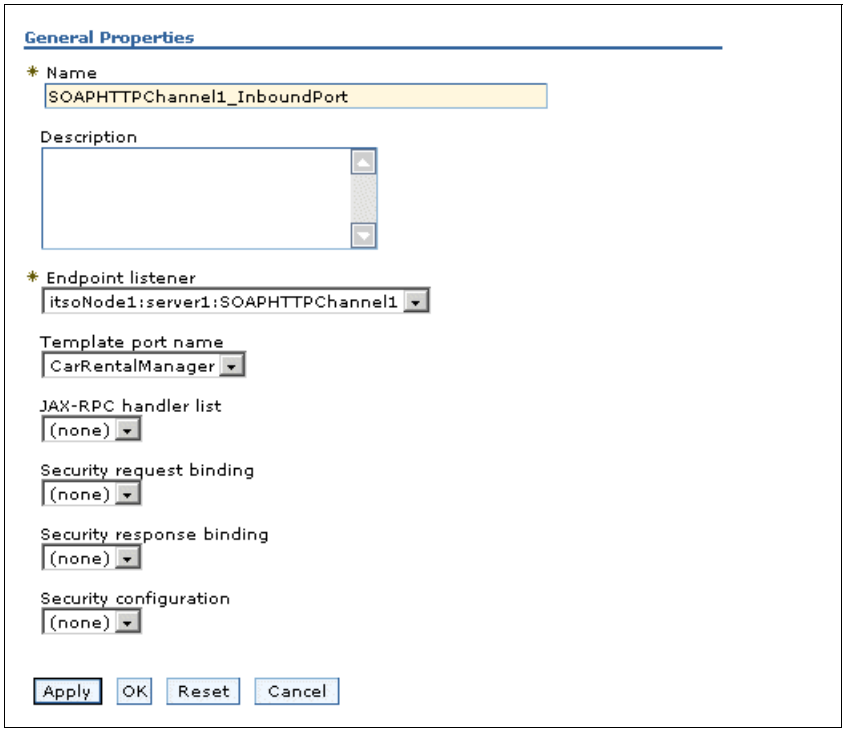


Figure 10-3 Rename the inbound port

8. We have now configured a service integration bus to contain a Web service gateway.
9. Click **Save**. Click **Save** again to save to master configuration.

10.2.8 Regenerating the plugin-cfg.xml (includes gateway)

Now that we have create the Web Services Gateway service, we must regenerate the plugin-cfg.xml.

1. Update the Map modules to servers.
   a. Expand **Applications** and click **Enterprise Applications**.
   b. Click **sibwshttp1.itsoNode1.server1**.
   c. Click **Map modules to servers** under Additional Properties.
   d. Select the following items using the Shift key:

      WebSphere:cell=itsoCell1,node=web1Node,server=webserver1

      WebSphere:cell=itsoCell1,node=itso1Node,server=server1

e. Check **SOAPHTTPChannel1** and Click **Apply.**

f. Click **OK**, then click **Save**.

g. Check **Synchronize changes with Nodes**, and click **Save**.

2. Generate a new Plug-in (plugin-cfg.xml).

   a. Expand the **Servers** tab, click **Web servers** from the Administrative Console.

   b. Check **webserver1**, and click **Generate Plug-in**.

3. Copy the plugin-cfg.xml file manually as follows:

   – From (Application Server node):

   ```
   C:\IBM\WebSphere\AppServer\profiles\Dmgr01\config\cells\itsoCell1\nodes\
   web1Node\servers\webserver1
   ```

   – To (IBM HTTP Server node):

   ```
   C:\IBM\WebSphere\Plugins\config\webserver1
   ```

4. When you are finished, restart IBM HTTP Server V6.0 Windows service for the new plugin-cfg.xml to be read immediately. The default refresh interval is 60 seconds.

10.2.9 Preparing WSDL for external travel organization use

In our example, we must expose selected services to be used by external travel organizations. Within our example scenario, this section would be performed by ITSO Car Rental organization IT staff in preparation for use by external organizations.

There are a couple of options for providing the WSDL files to external organizations.

- Create multiple port types and generate a separate or one WSDL file with the selected port types, which list the operations invoked by external organizations.
- If you only have one port type and want to expose selected operations to external organizations, you can edit the WSDL file manually.

Note: The example in our book demonstrates how to modify the WSDL file with one port type, to expose only selected operations.

Exporting the gateway WSDL

Complete the following steps to export the WSDL that defines the inbound service for the ITSO Car Rental Web service gateway service, so that the external organization (Global Travel application) can be configured to access the service.

1. Expand **Service Integration** and click **Buses**.
2. Click **WSGBus**.
3. Click **Inbound Services** Under Services.
4. Click **WSGService**.
5. Click **Publish WSDL files to ZIP file**.
6. On this page, click **WSGService.zip**.
7. When prompted, click **Save**. You are asked to provide a file name and path to save the zip of the WSDL files. You will need this to rebuild the external Web services client application (for example, Global Travels).
8. To view the contents of the zip file, click **WSGService.zip** then click **Open** on the file window. This exercise assumes you have a file compression and decompression program such as WinZip. You should see a window similar to Figure 10-4.

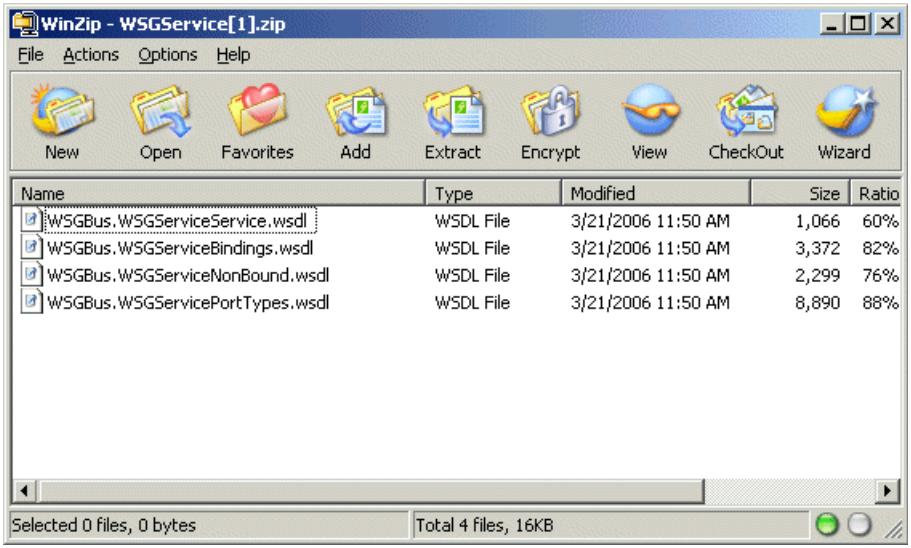


Figure 10-4 WSDL files displayed for the WSGService.zip

Modifying WSDL to expose selected operations

Do the following to modify the WSDL files exported in the previous step, so that only selected operations are exposed to external organizations:

1. Start Rational Application Developer.

   Note: For details about Rational Application Developer, refer to “Assemble an application with Rational Application Developer” on page 203.

2. Create a new Dynamic Web project named `ExposedWSDL`.
   a. Open the Web perspective.
   b. Select **File** → **New** → **Dynamic Web Project**.
   c. Enter `ExposedWSDL` in the Name field, and click **Finish**.
3. Import the exported gateway WSDL (WSGService.zip).
   a. Expand **Dynamic Web Projects** → **ExposedWSDL** → **WebContent** → **WEB-INF**.
   b. Right-click **WEB-INF** → **New** → **Folder.**
   c. Enter `wsdl` in the Name field, and click **Finish**.
   d. Right-click the new **wsdl** folder, and select **Import** → **Zip File**, and then click **Next**.
   e. Click Browse and navigate to the location you saved the WSGService.zip (exported WSDL zip file from “Exporting the gateway WSDL” on page 367). Select the **WSGService.zip** and click **Open**.
   f. Check all WSDL files and click **Finish**.

      After the import, you should have the following WSDL files in the wsdl folder:

      ```
      WSGBus.WSGServiceBinding.wsdl
      WSGBus.WSGServiceNonBound.wsdl
      WSGBus.WSGServicePortTypes.wsdl
      WSGBus.WSGServiceService.wsdl
      ```
4. Modify WSGBus.WSGServiceBinding.wsdl.
   a. Double-click on the **WSGBus.WSGServiceBinding.wsdl** file to open in the WSDL Editor.
   b. Click the **Graph** tab (bottom of dialog box) to view the WSDL in a graphical format.
   c. Expand Bindings by clicking the + symbol.
   d. Select **showAllReservations**, right-click and select **Delete** as seen in Figure 10-5.

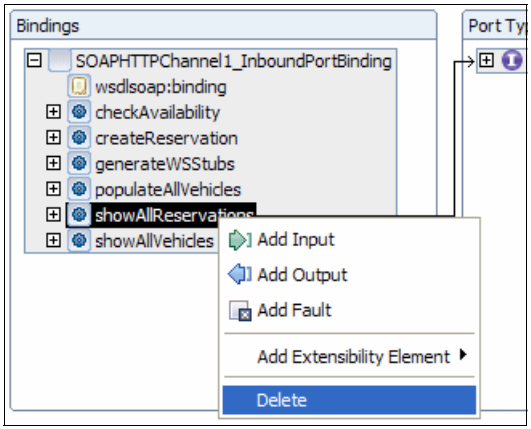


Figure 10-5 Delete ShowAllReservations binding

e. After deleting the desired binding, select **File** → **Save**.

f. Close the file.

5. Modify WSGBus.WSGServiceNonBound.wsdl.

   a. Double-click on the **WSGBus.WSGServiceNonBound.wsdl** file to open in the WSDL Editor.

   b. Click the **Graph** tab (bottom of the dialog box) to view the WSDL in a graphical format.

   c. Expand Bindings by clicking the **+** symbol.

   d. Select **showAllReservations**, right-click and select **Delete**.

   e. After deleting the desired binding, select **File** → **Save**.

   f. Close the file.

6. Modify WSGBus.WSGServicePortTypes.wsdl.

   a. Double-click the **WSGBus.WSGServicePortTypes.wsdl** file to open in the WSDL Editor.

   b. Click the **Graph** tab (bottom of the dialog box) to view the WSDL in a graphical format.

   c. Expand Port Types by clicking the **+** symbol.

   d. Select **showAllReservations**, right-click and select **Delete**.

   e. Check **Delete associated Messages and Parts** (default), and click **OK**.

   f. We found that some elements are not deleted by the graphical WSDL Editor. For this reason, you must delete the elements listed in Example 10-1 from the WSGBus.WSGServicePortTypes.wsdl manually.

Example 10-1 Elements to be deleted from the WSGBus.WSGServicePortTypes.wsdl

```
<element name="showAllReservations">
    <complexType>
     <sequence/>
    </complexType>
</element>
<element name="showAllReservationsResponse">
    <complexType>
     <sequence>
      <element name="showAllReservationsReturn" nillable="true"
type="tns2:Vector"/>
     </sequence>
    </complexType>
</element>
```

g. After deleting the binding, select **File** → **Save**.

h. Close the file.

7. Edit the target location from ITSO Application node to Web server node.

   a. Double-click on the **WSGBus.WSGServiceService.wsdl** file to open in the WSDL Editor.

   b. Click the **Source** tab (bottom of dialog box).

   c. Overwrite the location URL with the following:

      `http://`**`web1.itso.ral.ibm.com`**`/wsgwsoaphttp1/soaphttpengine/WSGBus/WSGService/SOAPHTTPChannel1_InboundPort`

   d. Select **File** → **Save**.

   e. Close the file.

8. Export the modified WSDL files to a new zip file.

   a. Select each of the following WSDL files using the Shift key, right-click them and select **Export...** as seen in Figure 10-6 on page 371.

      ```
      WSGBus.WSGServiceBindings.wsdl
      WSGBus.WSGServiceNonBound.wsdl
      WSGBus.WSGServicePortTypes.wsdl
      WSGBus.WSGServiceService.wsdl
      ```

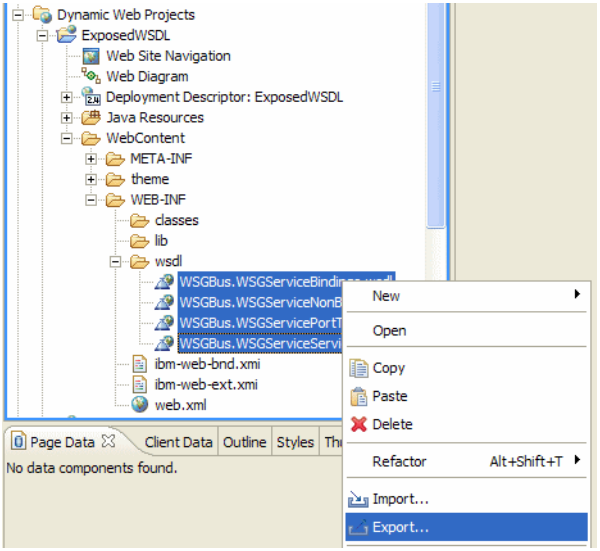


Figure 10-6 Export WSDL files

b. Select **Zip file** and click **Next**.

c. When the Export Zip File dialog box opens, complete these tasks as seen in Figure 10-7 on page 372:

   i. Enter `ExposedWSDService.zip` in the Name field.

   ii. Select **Create only selected directories**.

   iii. Click **Save**.

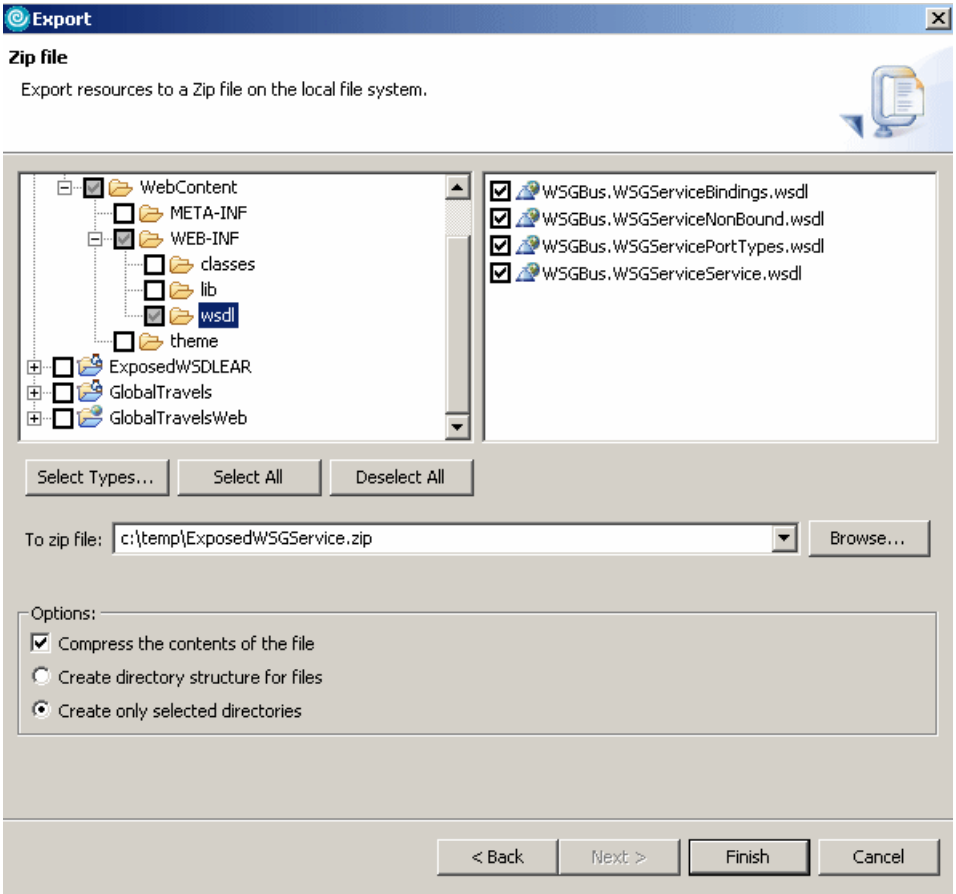


Figure 10-7 Export Zip Files

d. Click **Finish**.

Note: The WSDL files are now ready to be supplied to external organizations (for example, Global Travels) to rebuild the Web services client application. The rebuilding of the Web services client application is based on the modified exposed WSDL files.

The WSDL files can be published to a UDDI Registry or provided to the external organization via a mutual agreement. In our example, we will supply the files based on agreement between organizations.

10.3 Deploying Global Travels application

This section describes how to deploy the Global Travels Web services client application on the Global Travels application server (travel1).

10.3.1 Reassemble Global Travels application using Gateway WSDL

In our example, the Global Travels organization is provided with the WSDL files from ITSO Car Rental organization. The WSDL files are used to generate the Global Travels Web services client static stubs, which are used to invoke Web services offered by the ITSO Car Rental application.

Note: The Global Travels Web services client application was originally assembled using the ITSO Car Rental Web services WSDL and directly accessed the service (no gateway). This was done to simplify the initial development and testing process. In a real-world scenario, the external company would not be likely to have the ITSO Car Rental Web services WSDL.

Importing GlobalTravels.ws.zip project interchange file

To import the GlobalTravels.ws.zip project interchange file into Rational Application Developer, follow these steps:

Tip: If you have completed Section 8.7, “Create the external Web Service client application” on page 267, you can reuse the project.

1. Open the J2EE perspective.
2. Select **File** → **Import** → **Project Interchange**, then click **Next**.
3. When the Import Projects dialog box opens, enter the path and project interchange filename (for example, `GlobalTravels.ws.zip`), and click **Open**.

   Note: The GlobalTravels.zip project interchange file can be found in the following sample code directory:

   `C:\7240code\assemble\GlobalTravels.ws`

4. Check **Select All**, and click **Finish**.

Removing existing Web services client stubs and WSDL files

This section describes how to remove the existing Web services client stubs and WSDL files used for testing. When the client stubs and WSDL files are removed,

we demonstrate how to generate the Web services client stubs for the gateway WSDL.

To remove existing Web services client stubs and WSDL files, follow these steps:

1. Remove Java packages.
   a. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **Java Resources** → **JavaSource**.
   b. Select the following packages using the Shift key, right-click and select **Delete**:
      - **com.ibm.itso.carrental**
      - **com.ibm.itso.carrental.wrapper**
   c. Click **Yes** to confirm.
2. Delete existing CarRentalManagerExt.wsdl file.
   a. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **WebContent** → **WEB-INF** → **wsdl**.
   b. Select **CarRentalManagerExt.wsdl**, right-click and select **Delete**.
   c. Click **Yes** to confirm.
3. Delete CarRentalManagerExt_mapping.xml file.
   a. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **WebContent** → **WEB-INF**.
   b. Select **CarRentalManagerExt_mapping.xml**, right-click and select **Delete**.
   c. Click **Yes** to confirm.
4. Remove the service reference, Web services client binding, and extensions.
   a. Double-click **web.xml** to open in the Deployment Descriptor Editor.
   b. Remove service reference.
      i. Click the **References** tab.
      ii. Select **ServiceRef service/CarRentalManagerService**, and click **Remove**.
   c. Remove the Web services client binding.
      i. Click the **WS Binding** tab.
      ii. Select **Service/CarRentalManagerService**, and click **Remove**.
   d. Remove the Web services extensions.
      i. Click the **WS Extensions** tab.

ii. Select **Service/CarRentalManagerService**, and click **Remove**.

e. Select **File** → **Save**.

As a result of the save operation, the following files will be updated:

```
web.xml
ibm-webservicesclient-bnd.xmi
ibm-webservicesclient-ext.xmi
```

Importing gateway WSDL

To import the exported gateway WSDL (ExposesWSGService.zip), follow these steps:

1. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **WebContent** → **WEB-INF** → **wsdl**.
2. Right-click the new **wsdl** folder, and select **Import** → **Zip File**, then click **Next**.
3. Click Browse and navigate to the location you saved the ExposedWSGService.zip (exported WSDL zip file from “Modifying WSDL to expose selected operations” on page 368). Select **ExposedWSGService.zip** and click **Open**.
4. Check all WSDL files and click **Finish**.

Generating Web services client stubs

To generate the Web services client stubs, follow these steps the following:

1. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **WebContent** → **WEB-INF** → **wsdl**.
2. Right-click **WSGBus.WSGServiceService.wsdl**, select **Web Services** → **Generate Client**.
3. On the Web Services page, accept the default settings and click **Next**.
4. On the Web Services Selection page, accept the default settings and click **Next**.
5. When the Client Environment Configuration page opens, select **Web** for Client type, **GlobalTravelsWeb** for Client project, and **GlobalTravels** for EAR project, then click **Next**.
6. On the Web Services Proxy page, accept the default settings and click **Finish**.
7. When warned to overwrite WSDL file, click **Yes All**.

Modifying the Global Travels application client code

This section describes how to modify the Global Travels application client code to invoke the services exposed by the Web Services Gateway.

1. Expand **Dynamic Web Projects** → **GlobalTravelsWeb** → **Java Resources** → **JavaSource** → **com.ibm.globaltravels.utils**.
2. Double-click **GlobalTravelsUtils.java** to open in the Java Editor.

   The original java source is displayed in Example 10-2.

Example 10-2 Original GlobalTravelsUtils.java

```
import com.ibm.itso.carrental.*;
...
private static CarRentalManagerService carRentalMgrService = null;
...
carRentalMgrService =
(CarRentalManagerService)ctx.lookup("java:comp/env/service/CarRentalManagerService");
...
public static CarRentalManager getcarRentalMgr(){
...
carRentalManager = carRentalMgrService.getCarRentalManager();
...
}
```

3. Modify the source as displayed in Example 10-3.

> **Note:** The `com.ibm.nswsg` package name is created from the Gateway namespace that was provided during the configuration of the Web Services Gateway as seen in “Create a gateway instance” on page 323. Note, the reverse order of the package name with respect to the namespace.

Example 10-3 Modified GlobalTravelsUtils.java

```
import com.ibm.itso.carrental.*;
import com.ibm.nswsg.*;
...

private static WSGService carRentalMgrService = null;
...
carRentalMgrService = (WSGService)ctx.lookup("java:comp/env/service/WSGService");
...
public static CarRentalManager getcarRentalMgr(){
...
carRentalManager = carRentalMgrService.getSOAPHTTPChannel1_InboundPort();
...
}
```

4. Save modified GlobalTravelsUtils.java
5. Select **Project** → **Clean** and click **OK** to rebuild projects.
6. If your development environment can reach the runtime environment , for example web1.itso.ral.ibm.com is accessible from Rational Application Developer through 80 port, you can verify the Global Travels application this way:
   a. Select the WebSphere Application Server V6.0 test server, right-click **Publish** to test within Rational Application Developer.
   b. Enter the following URL:

      `http://localhost:9080/GlobalTravelsWeb/login.jsp`

   c. Verify the application.

Export the GlobalTravels.ear in preparation for deployment

Export the GlobalTravels.ear in preparation for deployment, as follows:

1. Open the J2EE perspective.
2. Expand **Enterprise Applications**.
3. Select **GlobalTravels**, right-click **Export** → **EAR file**.
4. Enter `C:\7240code\deploy\GlobalTravels.ws\GlobalTravels.ear` in the destination field, and click **Finish**.

10.3.2 Deploy the Global Travels application

Do the following to install the Global Travels Web Services client application to the Travel Application Server node:

1. Start the WebSphere Administrative Console by entering the following URL in a Web browser for the ITSO Application Server node:

   `http://travel1.itso.ral.ibm.com:9060/ibm/console`

2. Expand **Applications** tab, click **Install New Application** from the Administrative Console.
3. When the Preparing for the application installation dialog appears, do the following and click **Next**:
   – Select **Local file system**
   – Specify path: `C:\7240code\deploy\GlobalTravels.ws\GlobalTravels.ear`
4. On the Generate default bindings dialog box, accept the defaults and click **Next**.

5. When the Step 1: Select installation options dialog opens, check these items, then click **Next**:
   - Check **Deploy Web services.**
   - Accept default settings for remaining options.
6. In the Step 2: Map modules to servers dialog box, complete these items then click **Next**:
   - Select

     WebSphere:cell=travel1Node01Cell,node=travelNode01,server=server1
   - Check **GlobalTravelsWeb.**
   - Click **Apply, then** click **Next.**
7. On the Step 3: Map virtual hosts for Web modules dialog box opens, accept default and click **Next**.
8. When the Step 4: Provide options to perform the Web Services deployment dialog box opens, accept the default and click **Next**.
9. When the Step 5: Summary dialog box opens, review the settings, then click **Finish**.

   You should see a message similar to this one if the installation is successful:

   ```
   Application GlobalTravels installed successfully
   ```
10. Click **Save to Master Configuration**.
11. Click **Save**.
12. Restart the server1 application server.

10.3.3 Verify the application

Now that the Global Travels application is deployed and configured, verify that it is working properly.

1. Start the Global Travels application.
   a. Expand **Applications** tab, click **Enterprise Applications** from the Administrative Console.
   b. Check **GlobalTravels**.
   c. Click **Start**.
2. Verify the Global Travels application by accessing the following URL with a Web browser and creating a reservation:

   ```
   http://travel1.itso.ral.ibm.com/GlobalTravelsWeb/login.jsp
   ```

   For details about the application, refer to 11.4, “Global Travels application walkthrough” on page 395.

11

Walk through the application

The objective of this chapter is to verify the application functionality of the ITSO Car Rental application, and Global Travels application. The verification includes screen captures of the applications to demonstrate the requirements of the ITSO company have been fulfilled. In addition, we will describe the activities taking place behind the scenes as the Web services are invoked.

The chapter is organized into the following sections:

- Application overview
- Prerequisites for application walkthrough
- ITSO Car Rental application walkthrough
- Global Travels application walkthrough

11.1 Application overview

Our sample application consists of three enterprise applications.

- ITSO Car Rental applications
  - ITSO Car Rental application (ITSOCarRental.ear includes Web Services)
  - ITSO Car Rental Web application (ITSOCarRentalWS.ear includes internal Web services client)
- Global Travels Web application (GlobalTravels.ear includes external Web services client)

Users inside the intranet of ITSO Car Rental company have access to the ITSO Car Rental Web application, the Web service client for the ITSO Car Rental application, to create a reservation for a rent-a-car. When a user requests the reservation, the Web service client invokes the targeted Web service. The target Web service application inserts a reservation record into the database.

The Global Travels Web application is a Web service client for the ITSO Car Rental Applications (Web service).

The ITSO Car Rental Web application runs on the same application server as the Web service. In contrast, the Global Travels Web application is assumed to connect through the Internet to the Web service.

11.1.1 ITSO Car Rental application

The ITSO Car Rental application (ITSOCarRental.ear) has four operations.

- showAllVehicles
- checkAvailability
- createReservations
- showAllReservations

ITSOCarRentalWS.ear is a Web service client to the reservation service, and provides a Web interface to the users.

11.1.2 Global Travels application

The Global Travels application is a Web service client of ITSO Car Rental application accessed externally via the Web Services Gateway. It includes the interfaces to Web service operations below:

- showAllVehicles
- checkAvailability
- createReservations

Global Travels application has been developed by the WSDL file in the Web Service Gateway.

11.2 Prerequisites for application walkthrough

Prior to performing the application walkthrough, ensure that the following prerequisites have been met:

1. Runtime Environment is installed and configured.

   The details on how to install and configure the runtime environment can be found in Chapter 9, “Implement the runtime environment” on page 281.

2. ITSO Car Rental application is deployed.

   For details refer to Chapter 10, “Deploy application to WebSphere Application Server” on page 351.

3. The servers are started.

   For details refer to 10.1.3, “Starting the servers” on page 352.

11.3 ITSO Car Rental application walkthrough

In the following sections, we show how to create a reservation with the ITSO Car Rental Web service application.

11.3.1 Verify UC1: Check Rates

To check rates, follow these steps:

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

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

2. On the ITSO Car Rental home page, enter the user ID (for example, `webinfra`) and password as seen in Figure 11-1 on page 384, and click **Login**.

Note: The password is not required in this application because the purpose of this login page is just for creating a session in the server side. Any user ID can be used.

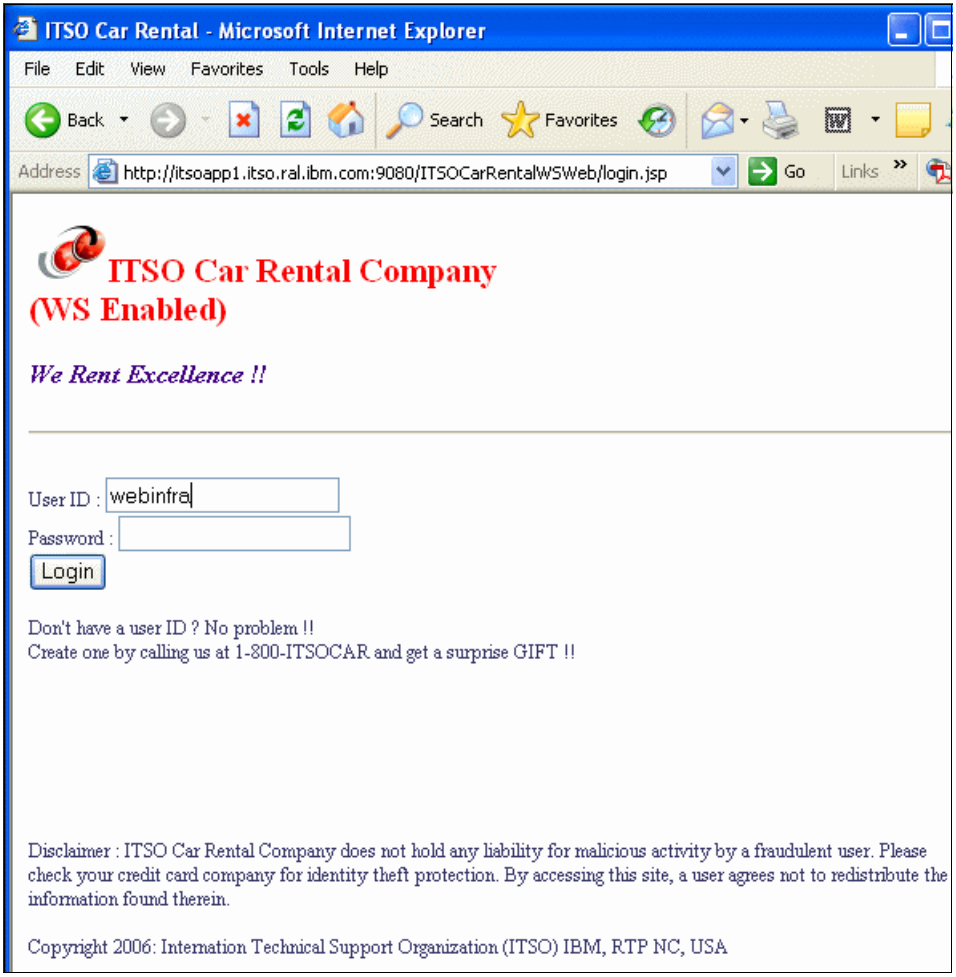


Figure 11-1 ITSO Car Rental home page

3. On the itinerary and preferences page (see Figure 11-2 on page 385), you can select your preferences and click **Submit**. If you do not select the Car Class Preference, all types will be displayed with rate information.

> **Note:** Pick up Date/Time and Drop off Date/Time must be a future date. Also these must meet the date format; mm/dd/yyyy. Otherwise, you will be brought back this page.

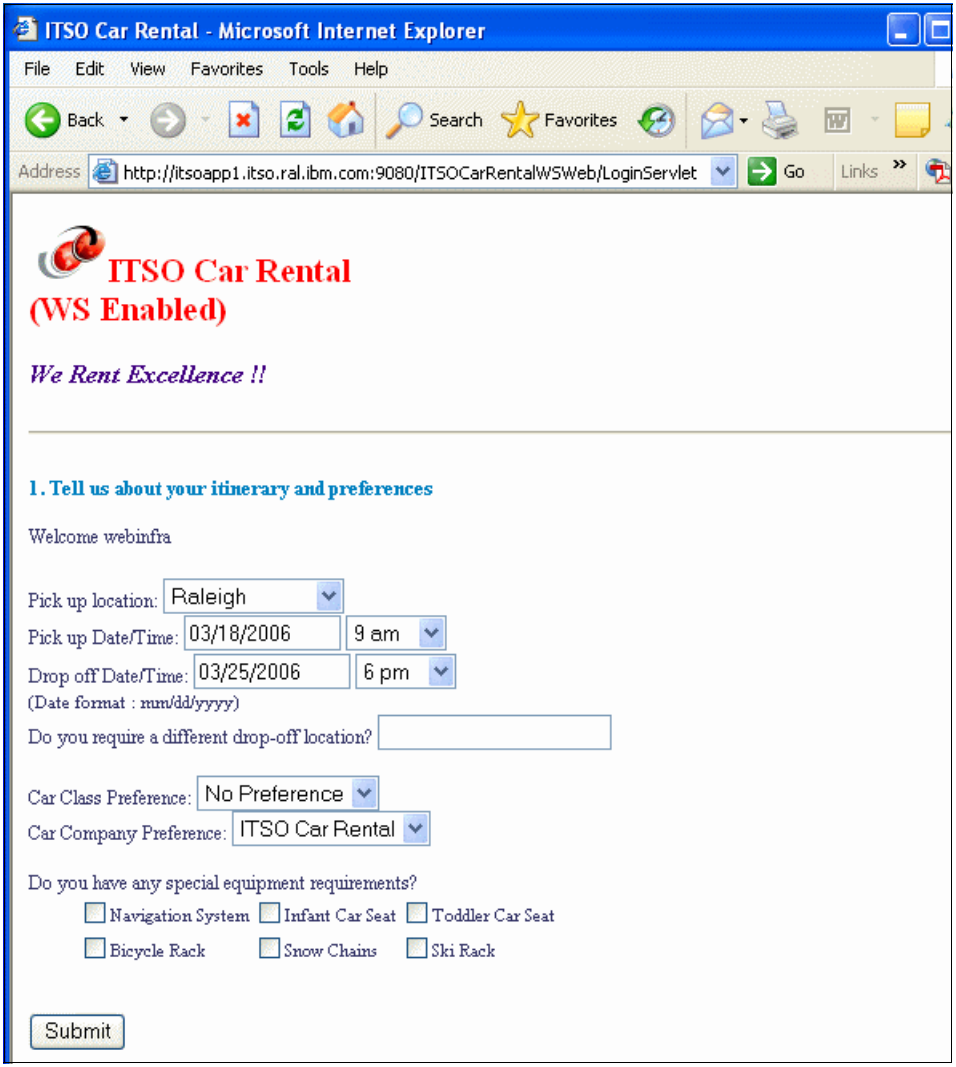


Figure 11-2 Itinerary and preferences selection page

4. The vehicle selection page as seen in Figure 11-3 on page 386 displays the vehicle rates.

> **Note:** Behind this step, the ITSO Car Rental Web service client application, ITSOCarRentalWSWeb.ear, invokes the showAllReservation operation in the Web Service in the ITSO Car Rental application, ITSOCarRental.ear, that accesses the database through the EJB.

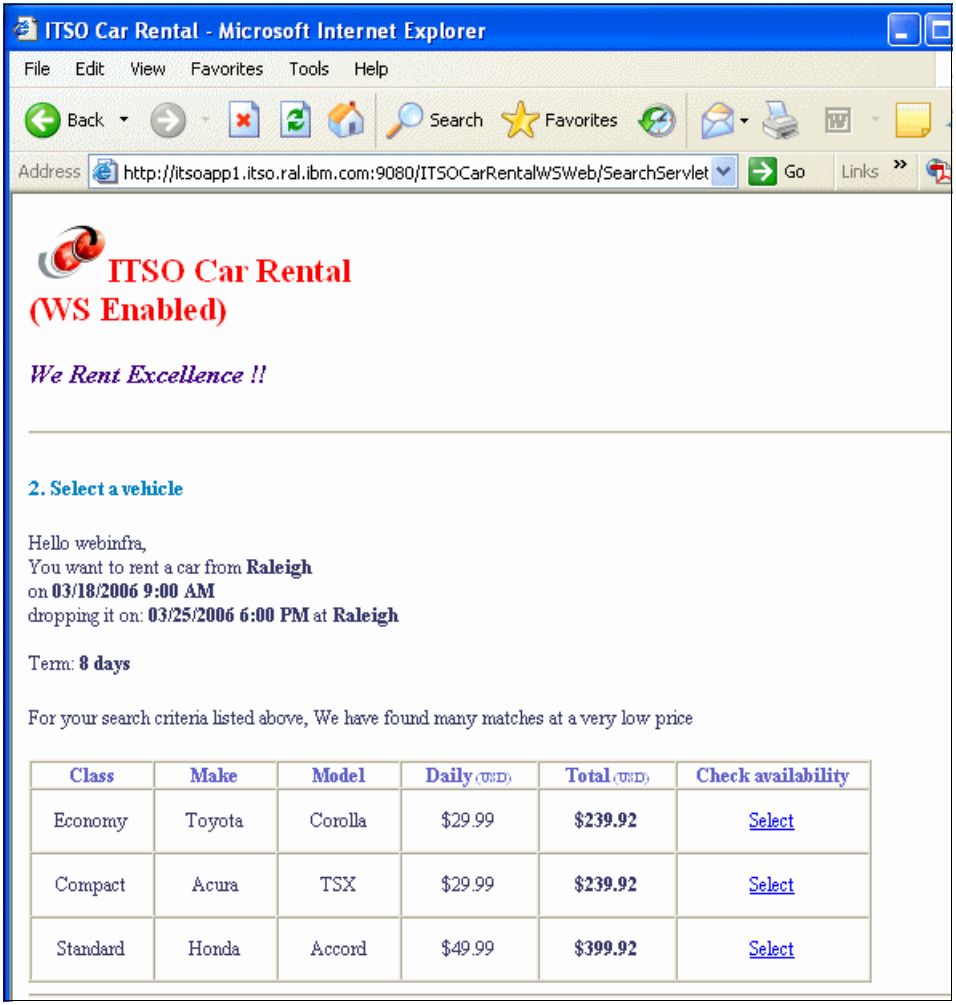


Figure 11-3 Vehicle selection page

Note: If you see an error message here, it is likely caused by your database configuration, check the log files (or Console view) and your configuration.

11.3.2 Verify UC2: Check Vehicle Availability

After the vehicles have successfully been listed in the vehicle selection page, you can check if the vehicles are available.

1. To check the vehicle availability, click one of the **Select** links for the corresponding vehicle, as seen in Figure 11-4. For example, we selected the Economy class car.

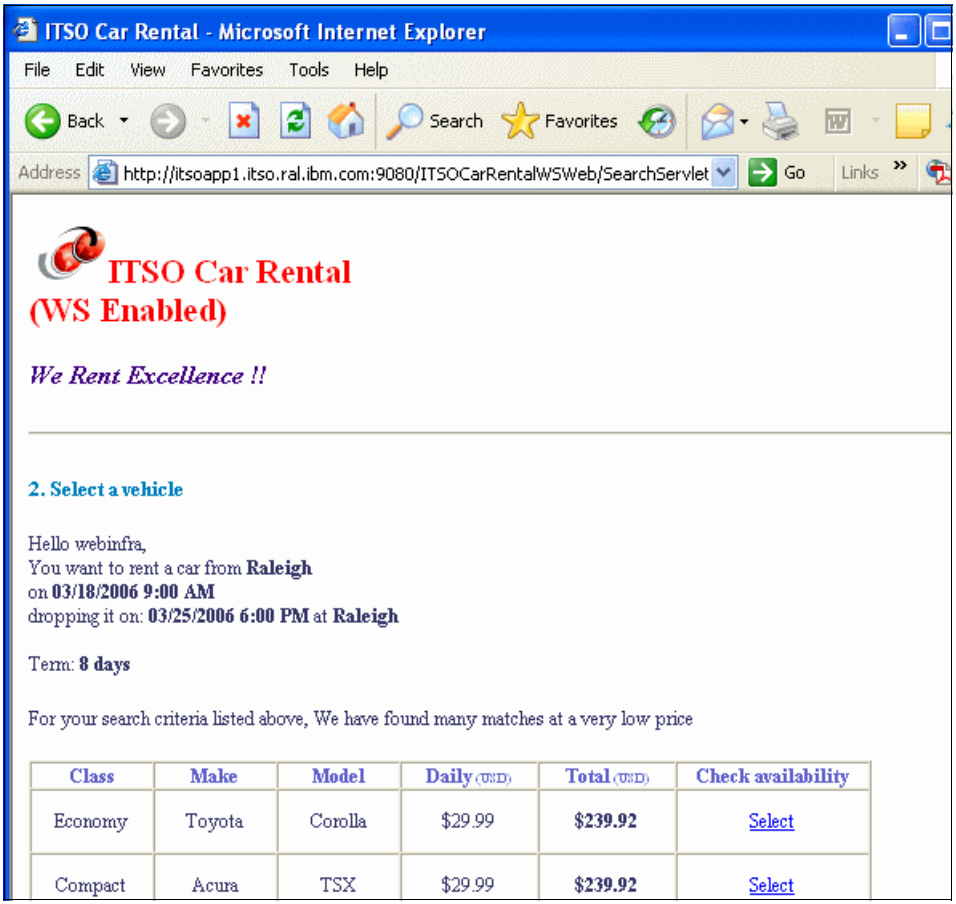


Figure 11-4 Vehicle selection page

2. The availability information is displayed as seen in Figure 11-5 on page 389.

Note: Behind this step, the ITSO Car Rental Web service client application, ITSOCarRentalWSWeb.ear, invokes the checkAvailability operation in the Web Service in ITSO Car Rental application, ITSOCarRental.ear.

For the purpose of making this sample application simple, the checkAvailability operation returns always `true` and does not access the database.

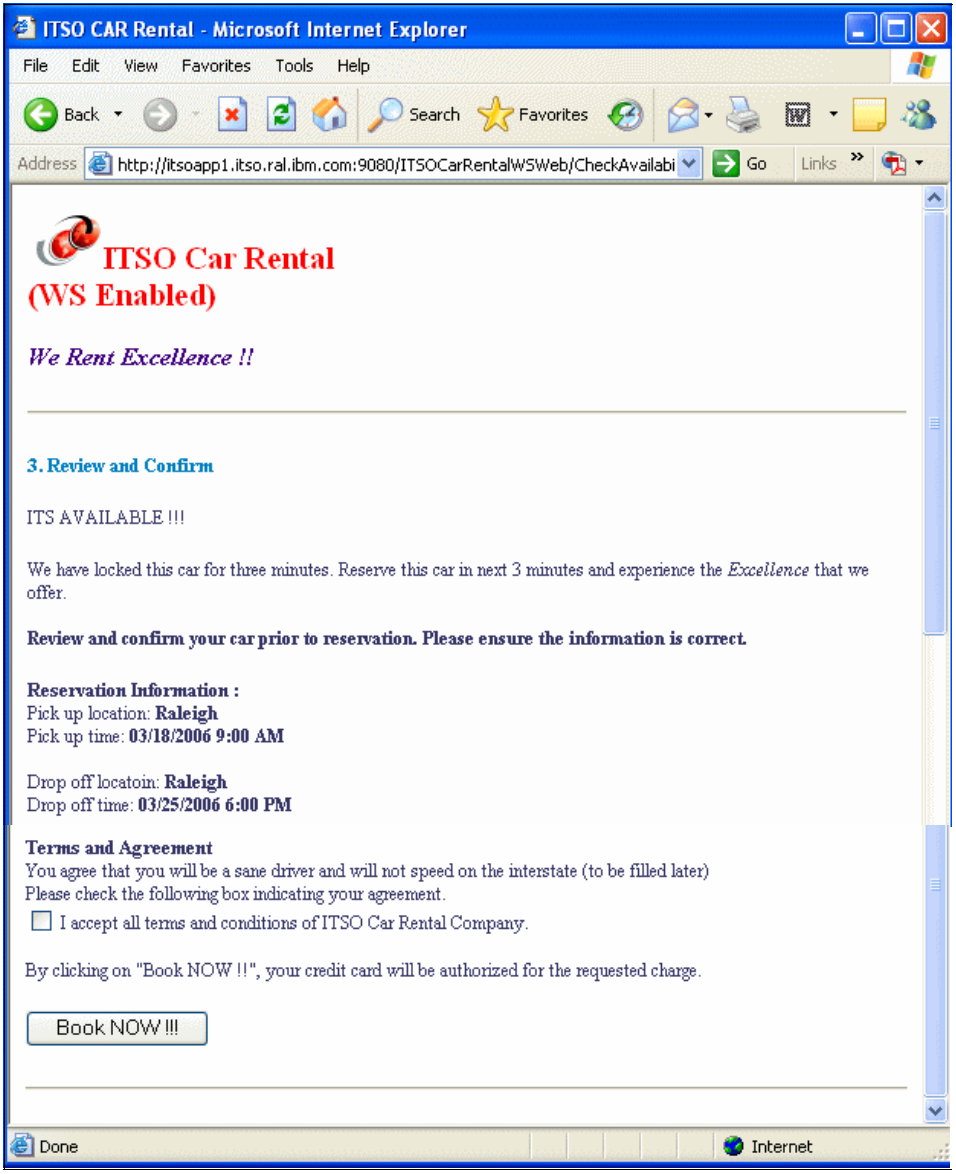


Figure 11-5 Review and Confirm page

11.3.3 Verify UC3: Create Reservation

To verify the reservation is created, follow these steps:

1. After the review and confirmation page appears, click the **Book NOW** button at the bottom of the page, as seen in Figure 11-6 on page 390.

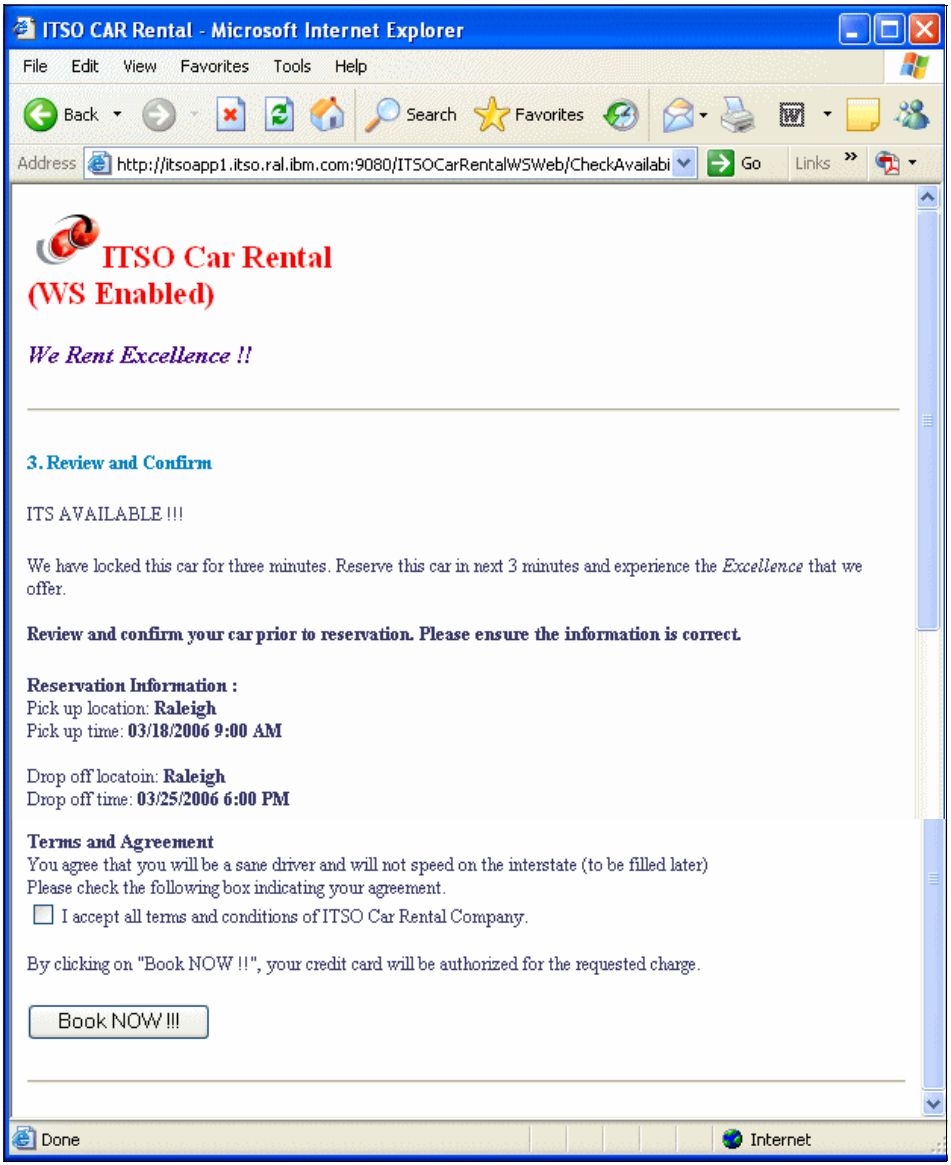


Figure 11-6 Review and Confirm page

2. When the receipt page opens (as seen in Figure 11-7 on page 391), the transaction is complete. You can see a new record in the RESERVATION table.

> **Note:** Behind this step, the ITSO Car Rental Web service client application, ITSOCarRentalWSWeb.ear, invokes the createReservation operation in the Web Service in the ITSO Car Rental application, ITSOCarRental.ear, that accesses the database through the EJB.

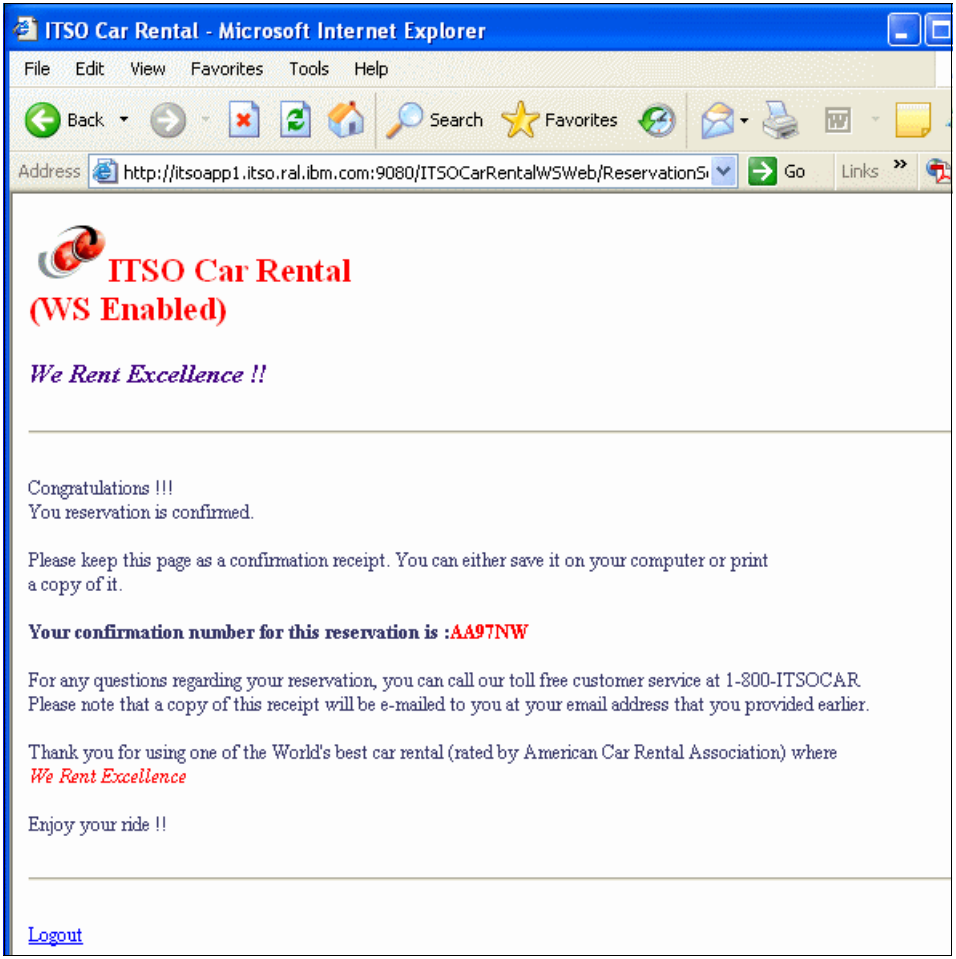


Figure 11-7 Receipt page

11.3.4 Verify UC8: View Reservation Activity

There is an administrative feature in the ITSO Car Rental application for a business analyst or administrator. This feature retrieves the reservation data from the database, and displays the following information:

- Channel and the number of reservations
- Reservation details

To view the Reservation Activity, follow these steps:

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

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

2. When the ITSO Car Rental home page appears, accept the default login user ID (admin) and password (blank). The admin user ID is a special ID used for our sample application to make the View Reservation Activity page visible. Click **Login** as seen in Figure 11-8.

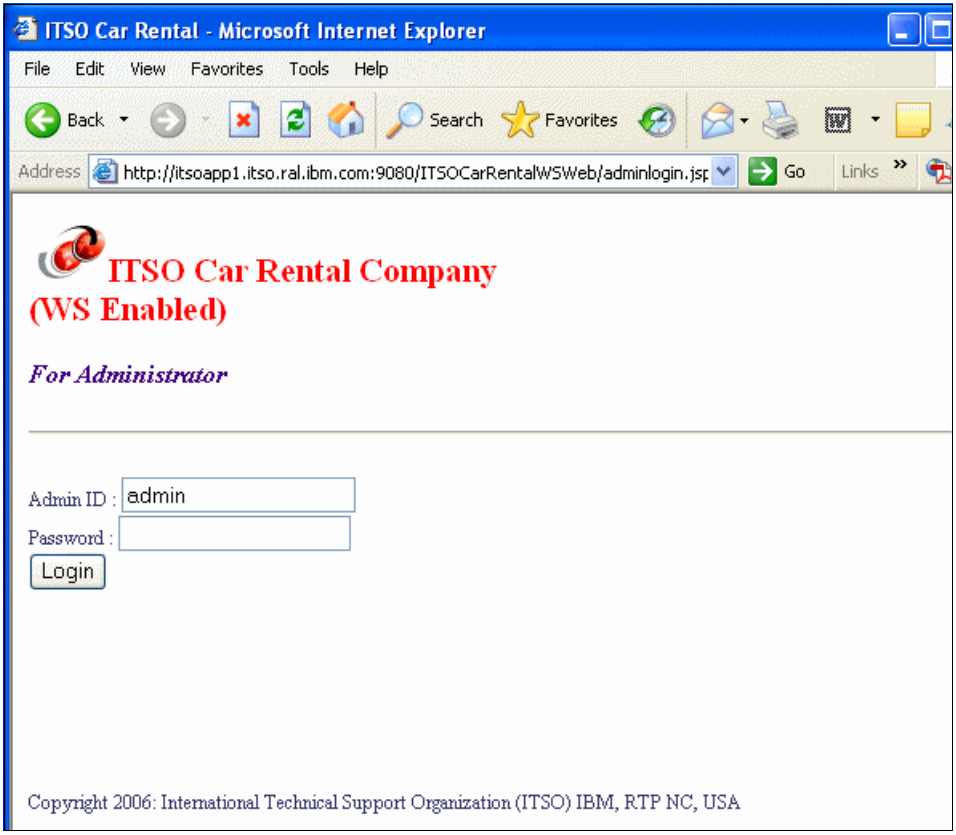


Figure 11-8 Login page to the View Reservation Activity

3. When the Main menu page opens, click the **List Reservations** link as seen in Figure 11-9 on page 394.

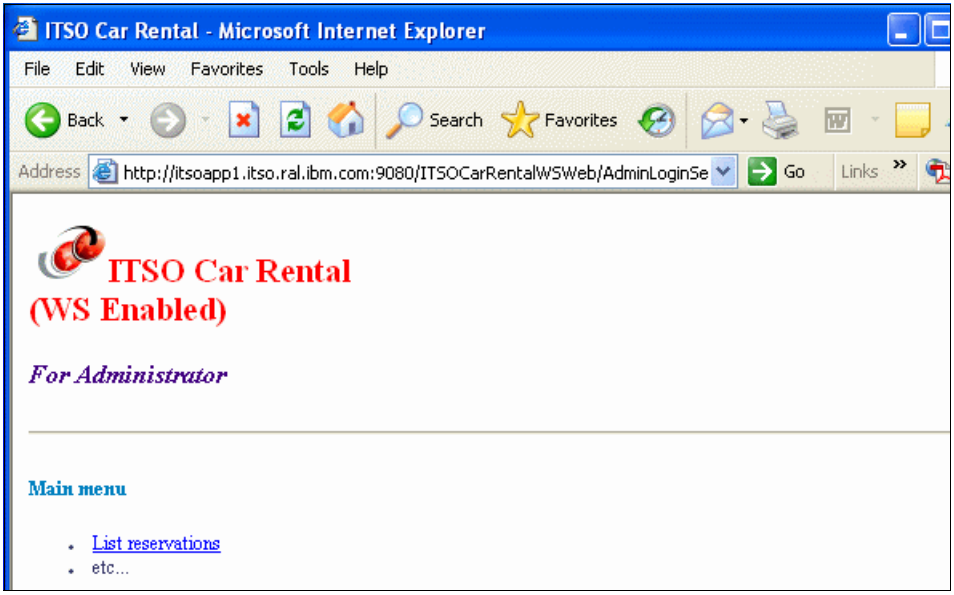


Figure 11-9 Main menu page

4. The resulting page displays the list of Channel and number of reservations and Reservation Details (see Figure 11-10 on page 395).

> **Note:** Behind this step, the ITSO Car Rental Web service client application, ITSOCarRentalWSWeb.ear, invokes the showAllReservation operation in the Web Service in the ITSO Car Rental application, ITSOCarRental.ear, that accesses the database through the EJB.

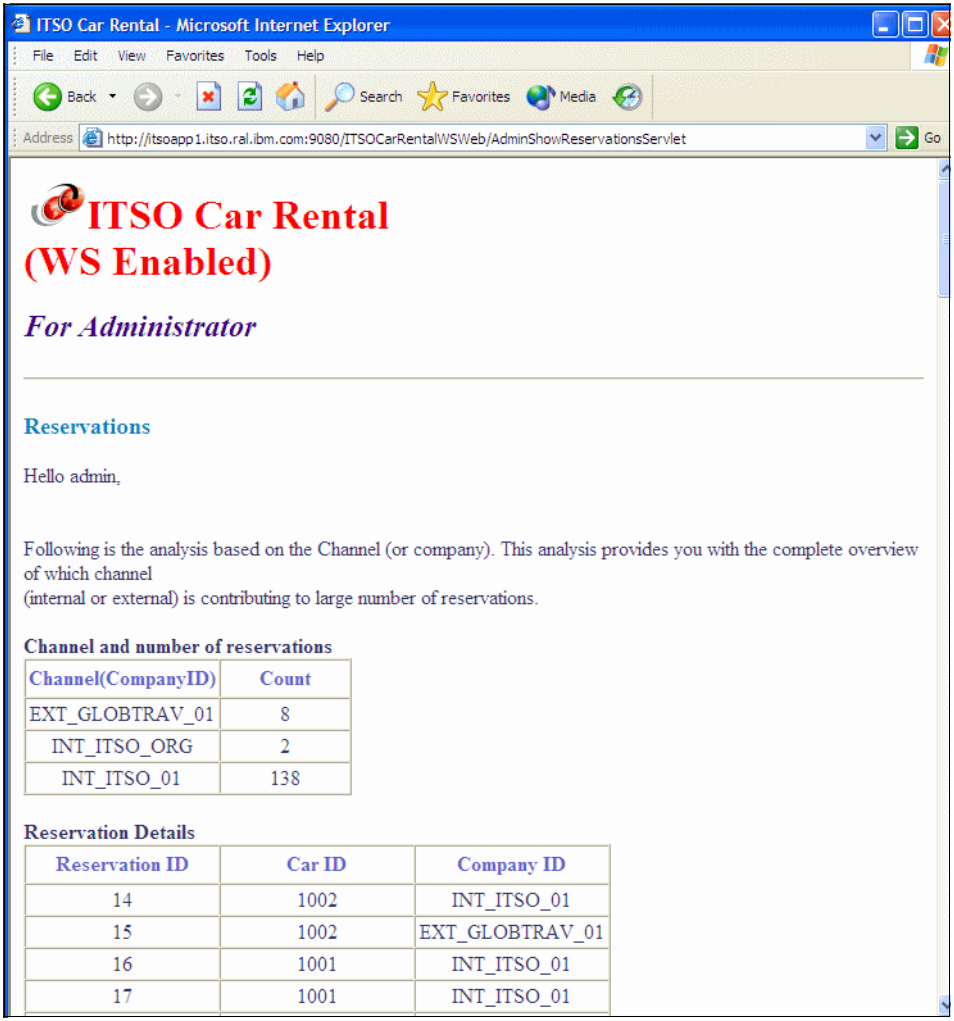


Figure 11-10 The Reservation List page

11.4 Global Travels application walkthrough

The Global Travel Web application is a Web services client for the ITSO Car Rental application.

- It is implemented by Java Servlet and search the target Web service (Car Rental application) using JNDI lookup and invoke it.
- It runs on the remote WebSphere Application Server node and accesses the Web service on the WebSphere Application Server-ND node

(itsoapp1.itso.ral.ibm.com) over IBM HTTP Server node (web1.itso.ral.ibm.com).

- It has the same operations for the Web service as the ITSO Car Rental Web application, except the showAllReservation operation.

In the following steps, we show how to create a reservation using the ITSO Car Rental Web service application.

11.4.1 Verify UC1: Check rates

To create a new car rental reservation, follow these steps:

1. Enter the following URL in a Web browser to launch the Global Travels Web application:

   `http://itsoapp1.itso.ral.ibm.com:9080/GrobalTravelsWeb/login.jsp`

2. When the Global Travel home page opens, enter the user ID (for example, `webinfra`) and password as seen in Figure 11-11 on page 397, and click **Login**.

> **Note:** The password is not required in this application because the purpose of this login page is just for creating a session in the server side. Any user ID can be used.

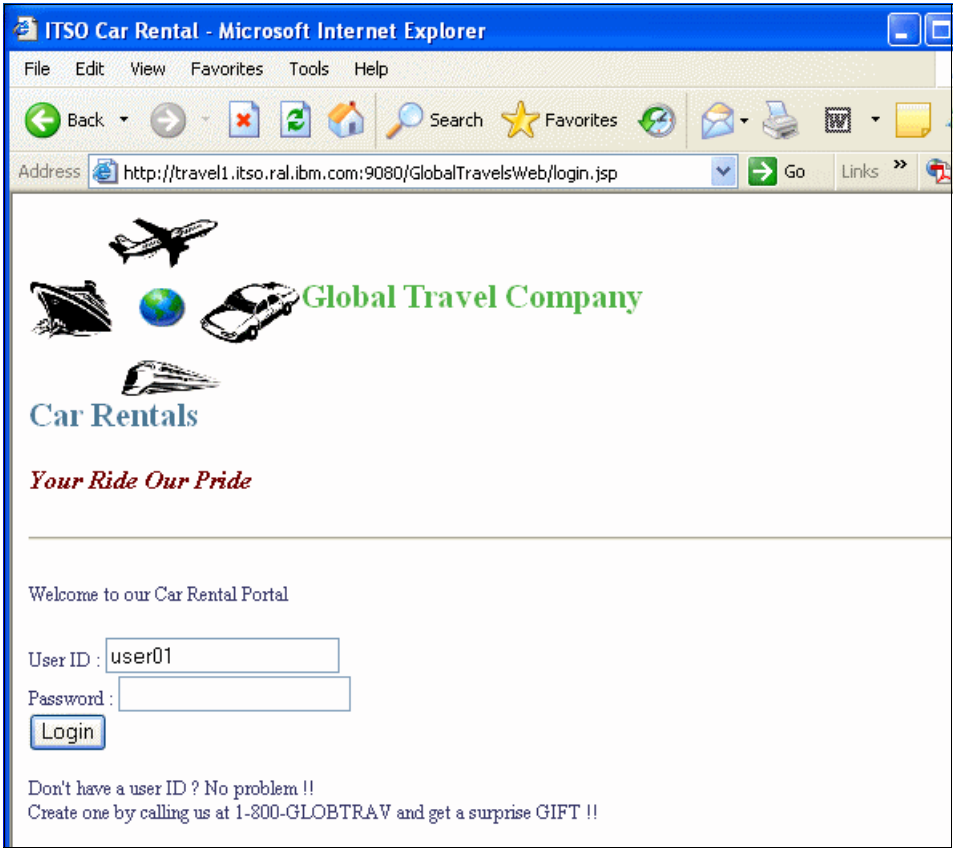


Figure 11-11 ITSO Car Rental home page

3. When the itinerary and preferences page opens (see Figure 11-12), you can select your preferences and click **Submit**. If you do not select the Car Class Preference, all types will be displayed with rate information.

> **Note:** Pick up Date/Time and Drop off Date/Time must be a future date. Also these must meet the date format; mm/dd/yyyy. Otherwise, you will be brought back this page.

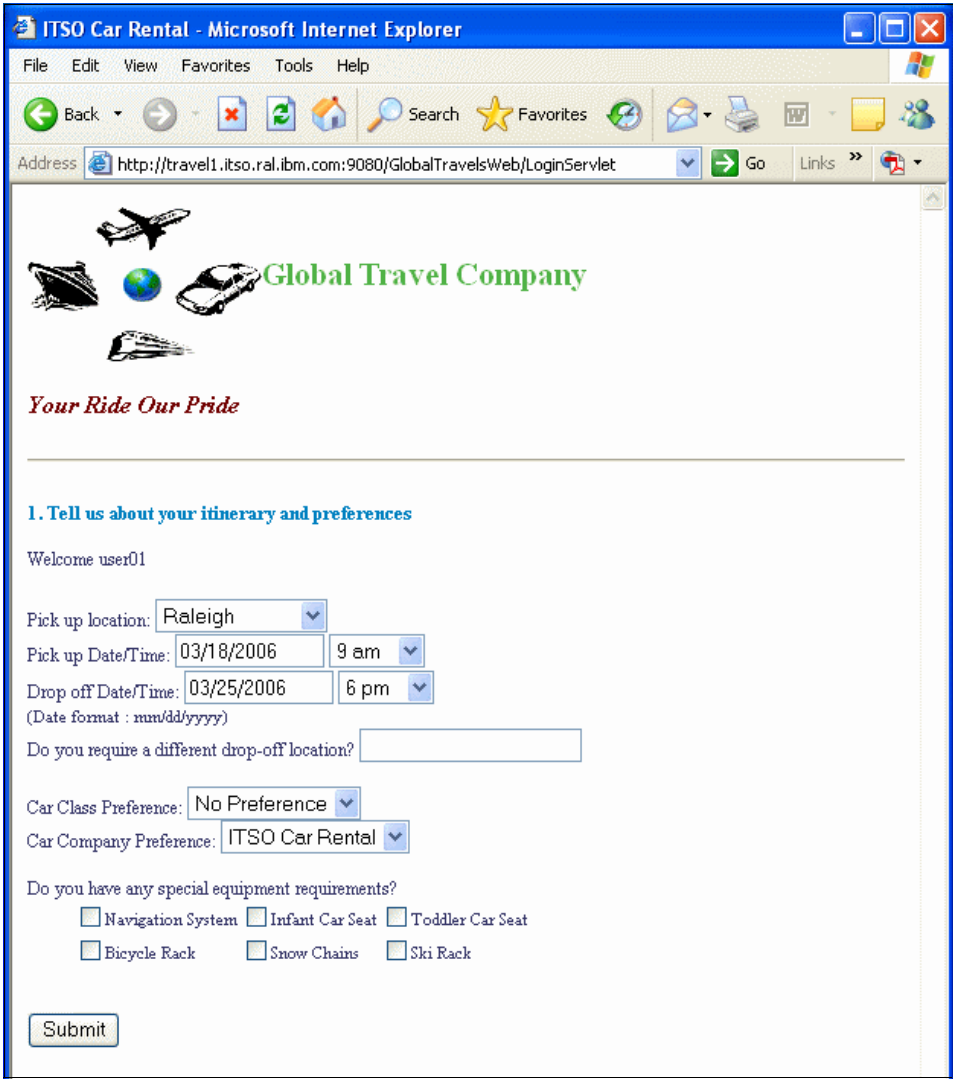


Figure 11-12 Itinerary and preferences selection page

4. The vehicle selection page displays the vehicle rates as seen in Figure 11-13 on page 399.

> **Note:** Behind this step, the Global Travels web application, GlobalTravelsWeb.ear, invokes the showAllVehicle operation in the Web Service in the ITSO Car Rental application, ITSOCarRental.ear, that accesses the database through the EJB.

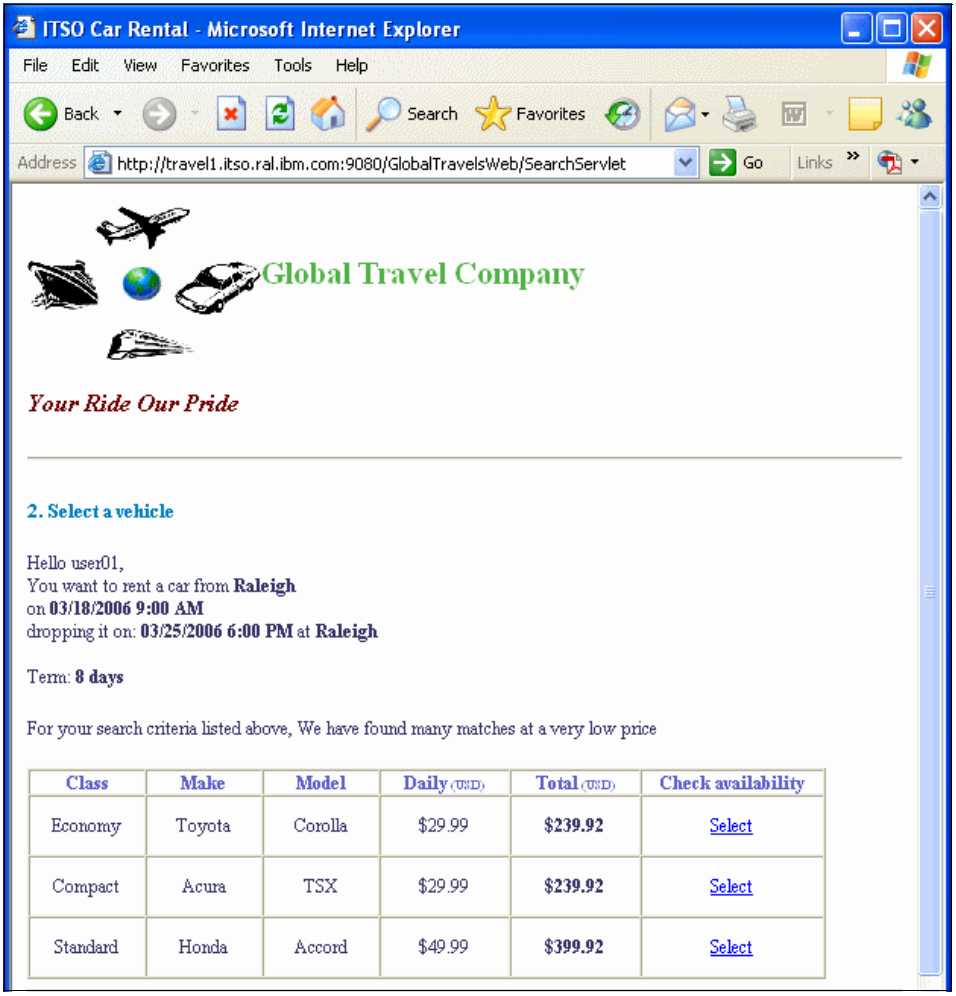


Figure 11-13 Vehicle selection page

Note: If you see an error message here, it is likely caused by your database configuration, check the log files (or Console view) and your configuration.

11.4.2 Verify UC2: Check vehicle availability

After the vehicles have been listed in the vehicle selection page successfully, you can check if the vehicles are available.

1. To check the vehicle availability, click one of the **Select** links for the corresponding vehicle as seen in Figure 11-14. For example, we selected the Economy class car.

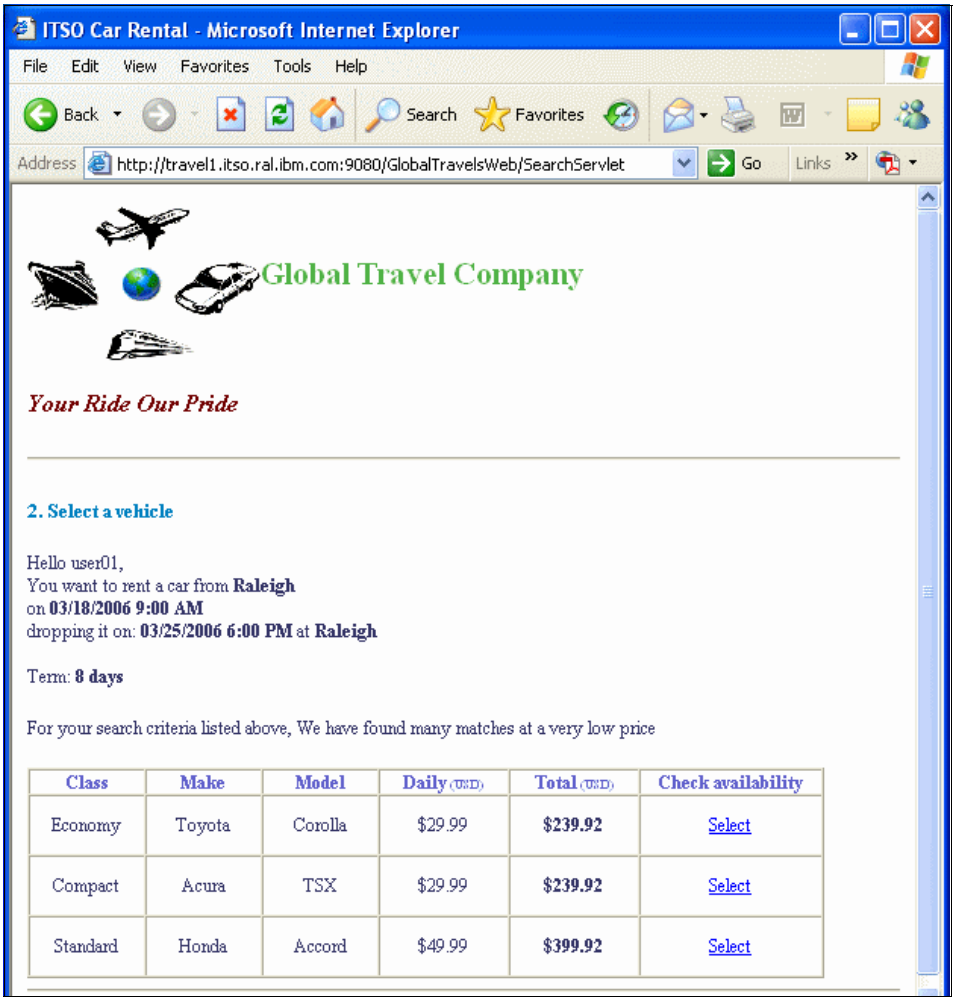


Figure 11-14

2. The availability information is displayed as seen in Figure 11-15 on page 401.

Note: Behind this step, the GlobalTravels web application, GlobalTrabels.ear, invokes the **checkAvailability** operation in the Web Service in ITSO Car Rental application, ITSOCarRental.ear.

For the purpose of making this sample application simple, the operation returns always “true” and does not access the database.

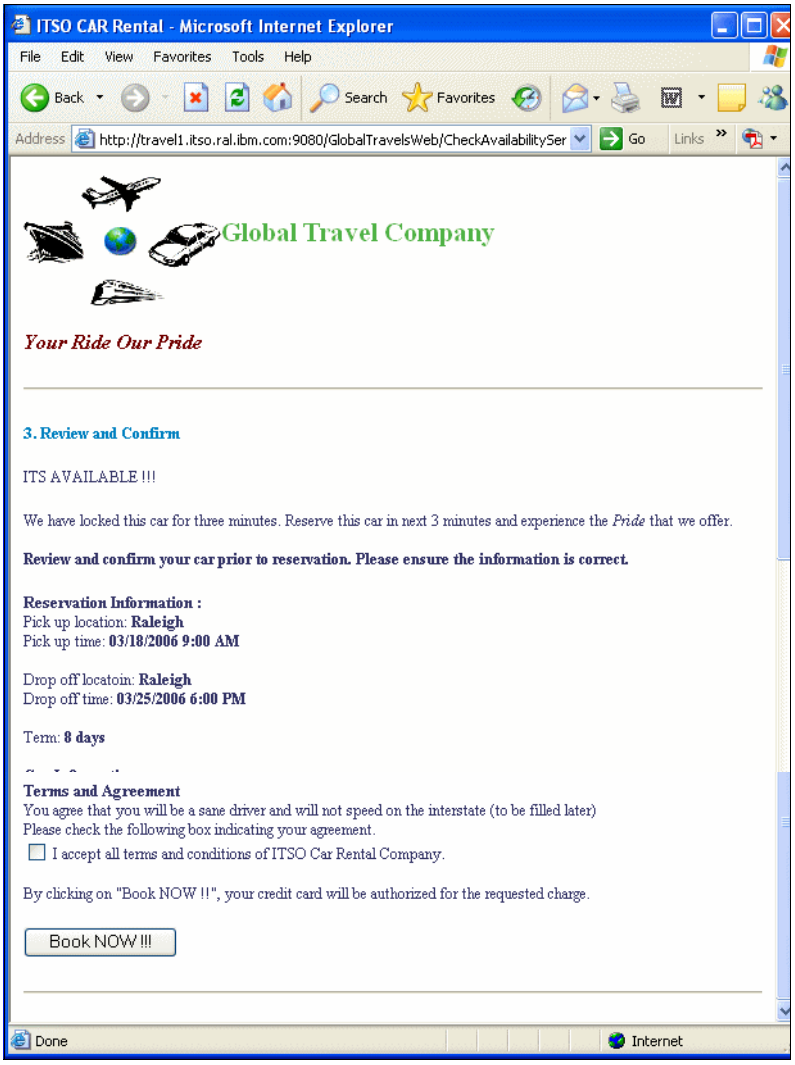


Figure 11-15 Review and Confirm page

11.4.3 Verify UC3: Create reservation

To verify that the reservation is created, follow these steps:

1. On the review and confirmation page, click **Book NOW** button at the bottom of the page as seen in Figure 11-16 on page 402.

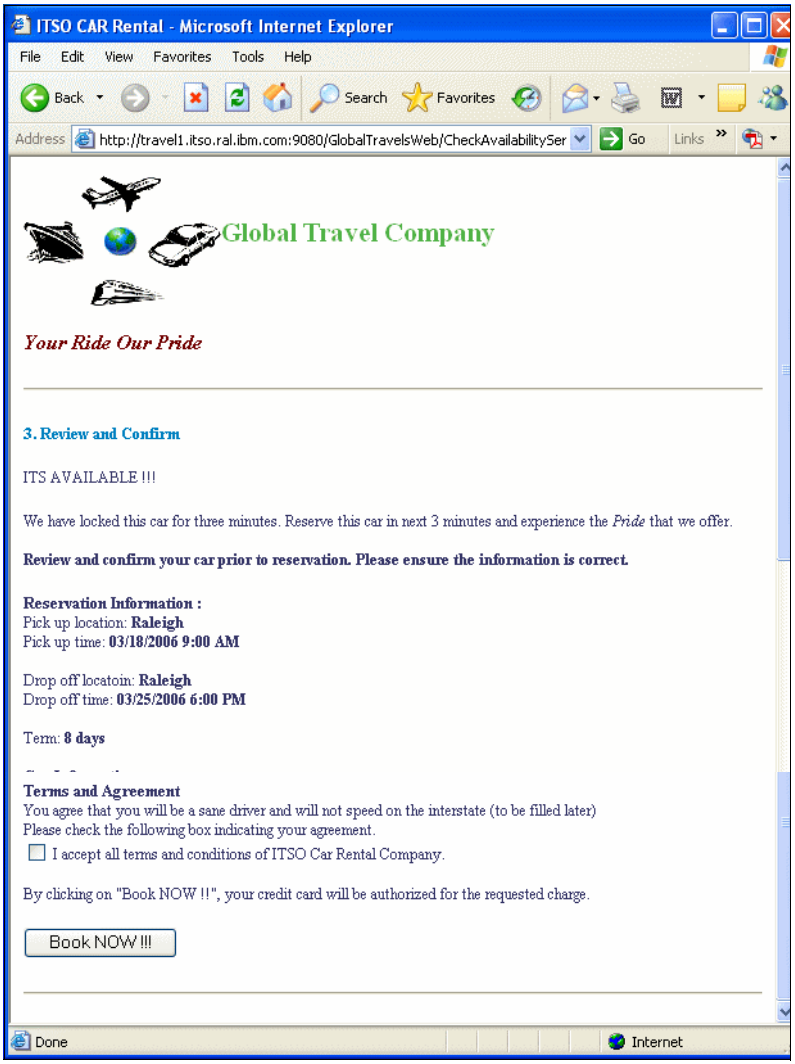


Figure 11-16 Review and Confirm page

2. When the receipt page opens (as seen in Figure 11-17 on page 403), the transaction is complete. You can see a new record in the RESERVATION table.

Note: Behind this step, the Global Travels web application, GlobalTravelsWeb.ear, invokes the createReservation operation in the Web Service in the ITSO Car Rental application, ITSOCarRental.ear, that accesses the database through the EJB.

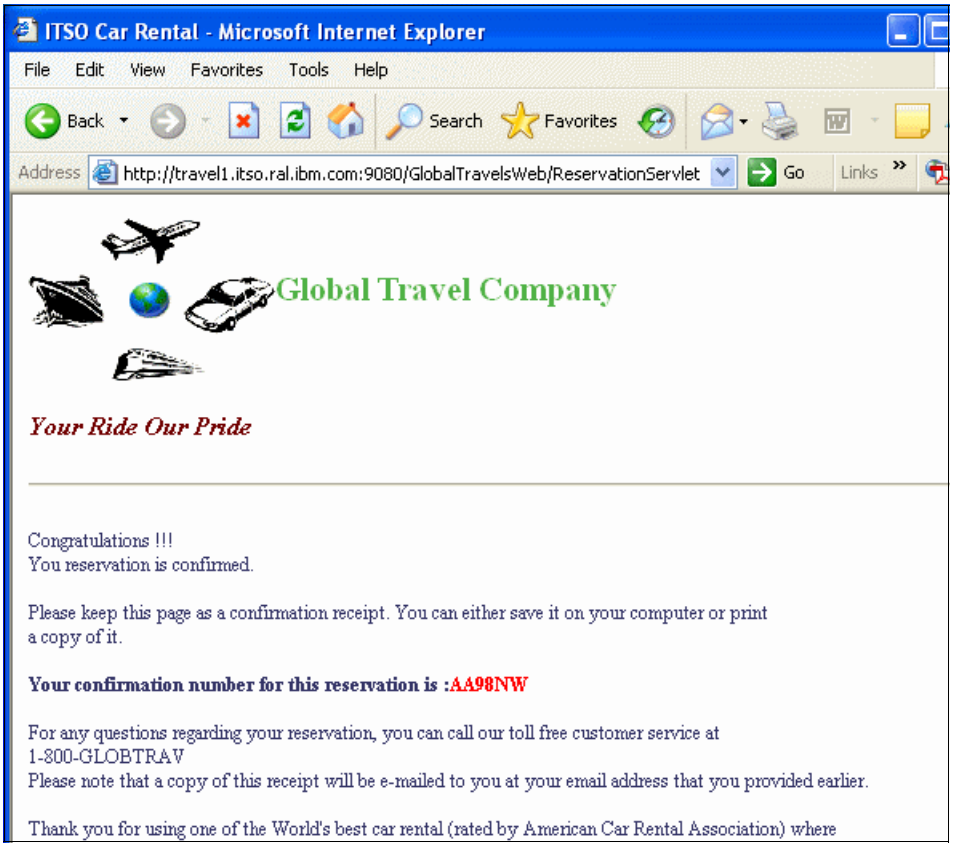


Figure 11-17 Receipt page

12

Manage and monitor services with ITCAM for SOA

IBM Tivoli Composite Application Manager (ITCAM) for SOA is used to monitor and manage the services layer of SOA applications.

This chapter provides an introduction to SOA management. Next we provide an overview of the features and architecture of ITCAM for SOA. Finally, we describe how to use ITCAM for SOA to fulfill the non-function requirements of the ITSO Car Rental working example for managing and monitoring Web services.

This chapter is organized into the following sections:

- Introduction to SOA management
- ITCAM for SOA product overview
- ITCAM for SOA working example

12.1 Introduction to SOA management

This section provides an introduction to the challenges, architecture and IBM Tivoli products used for SOA management. In addition, we describe how IBM Tivoli management products can be used to manage and monitor a composite application for the SOA Foundation - Service Creation scenario.

We have organized the topics into the following sections:

- SOA management challenge
- Managing the SOA Foundation architectural layers
- IBM IT Service Management
- IBM Tivoli products for SOA management
- SOA management use case for the Service Creation scenario

12.1.1 SOA management challenge

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.

Note: In “Greater need for a flexible architecture” on page 8 we provide a comparison of SOA applications with monolithic (silos) and component-based architectures.

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

- Managing and monitoring the end-to-end view
- Understanding the relationship of services
- Managing services as resources
- Ensuring nonfunctional requirements are achieved by design

Managing and monitoring the end-to-end view

In a silos-based application architecture, resources are often managed by separate operators or specialists, by using 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 provide detailed information about performance and availability metrics for the individual resources. Also, the operator should have an end-to-end monitoring role.

Understanding the relationship of services

It is important to understand the relationship of service consumers and providers for composite applications. SOA brings the benefits of application reuse. The importance of monitoring and managing the availability and performance of the application functionality exposed as a service increases when reused by service consumers that depend on this functionality.

For example, in a silos approach each application can have its own tax calculation application functionality. In an 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.

Managing 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).

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.

Ensuring nonfunctional requirements are achieved by design

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.

12.1.2 Managing the SOA Foundation architectural layers

The SOA Foundation Reference Architecture Solution View includes architectural layers as seen in Figure 12-1 on page 408. We explain how the resources for these architectural layers are managed in an SOA environment. In some cases, the monitoring spans the architectural layers. In others, it is focused on monitoring specific resources within the layers.

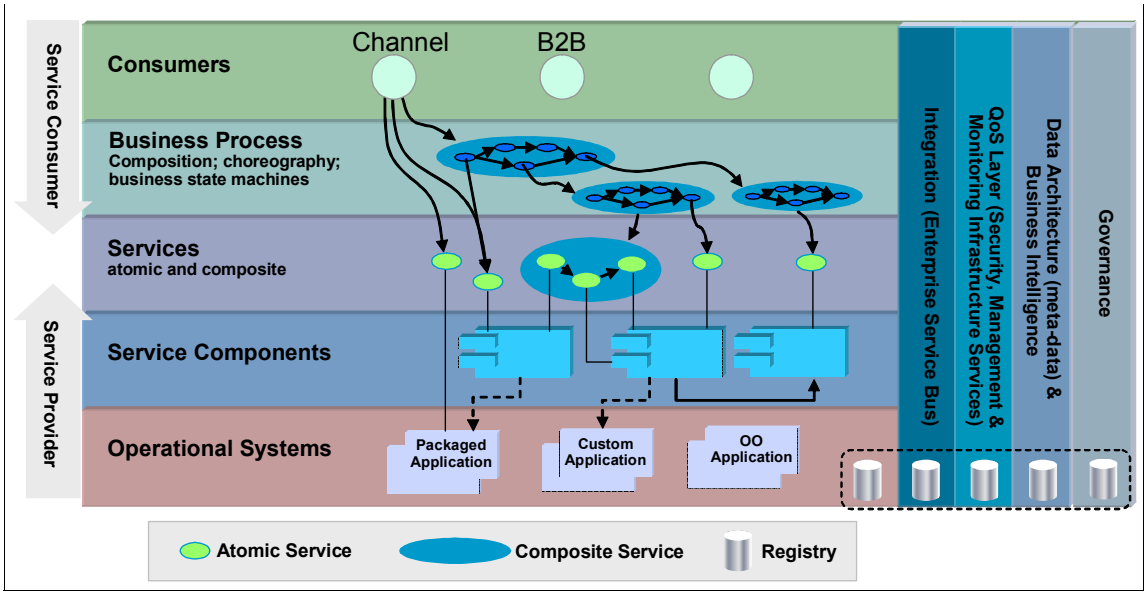


Figure 12-1 SOA Foundation Reference Architecture Solution View

Managing services

Management of services includes service consumers and service providers. Service consumers invoke services, while service providers expose application functions to be consumed. For example, both a J2EE Web Services client application (service consumer) and an session EJB Web service (service provider) can be monitored and managed.

Some key elements of managing the services layer are:

- Understand how services relate to each other and to the IT infrastructure, as well as how the service relates to the business process layer.
- Control the message flow in the service environment through management mediations such as log, filter, and route. The message flow often spans the architectural layers.
- Centralize services management policy.
- Define 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 very important to understand the relationship of service providers and consumers. This also includes understanding the impact to the business process or business service.

Managing transactional performance

Managing and monitoring the end-to-end transactional performance is a key measurement for SLAs. Managing transaction performance is also a very useful analysis and troubleshooting tool. For example, you can be proactive in detecting a slow down 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 failing.

Managing transactional performance includes the following:

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

Managing the supporting middle ware

Many of the resources used by services are found in the middle ware. 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 middle ware includes the following concepts:

- Understanding the health of the infrastructure that support 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 the following concepts:

- Understand the health of the infrastructure that support the services.
- Correlate problems in the services to infrastructure issues such as a queue filling up or an exhausted thread pool.

Note: Within the SOA Foundation scenarios, *SOA security* has been defined as a separate solution area from *SOA management*. Some organizations may view SOA security as part of the SOA management domain. For the purposes of this chapter, we do not cover SOA security.

For more information on SOA security, refer to the following IBM Redbook:

- *Understanding SOA Security Design and Implementation*, SG24-7310

12.1.3 IBM IT Service Management

IBM IT Service Management helps organizations better manage their IT infrastructure to more effectively and efficiently deliver IT services. IBM IT Service Management is comprised of the following elements as seen in Figure 12-2:

- IT Process Management products
- IT Service Management platform

> **Note:** SOA management tooling can be found within the scope of the IT Service Management platform. We will highlight the products defined in the IT Service Management platform for SOA management in 12.1.4, “IBM Tivoli products for SOA management” on page 411.

- IT Operational Management products

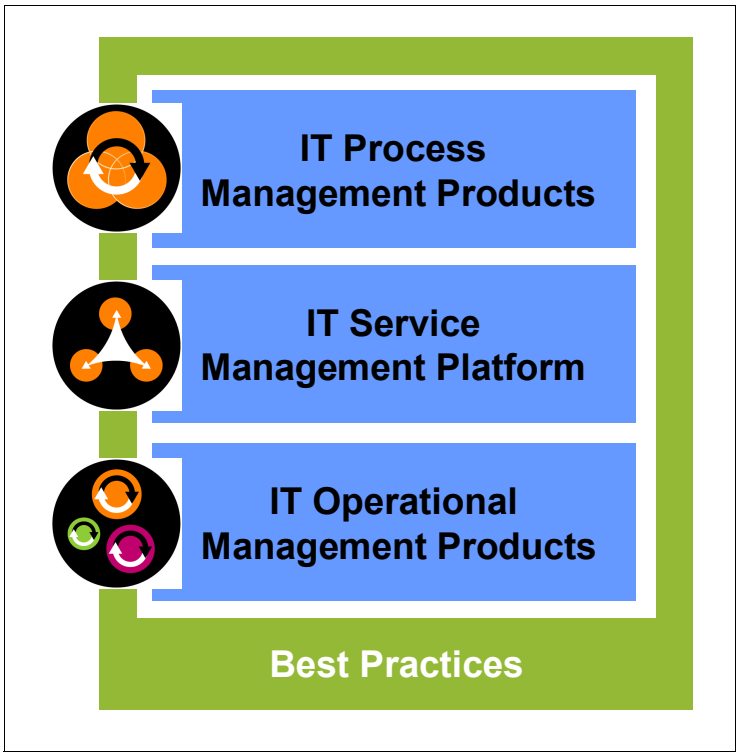


Figure 12-2 IBM IT Service Management

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

- ITIL-aligned workflows, based on WebSphere Process Server
- Open and standards-based Configuration Management Database (CMDB)
- Automated, infrastructure-aligned tasks
- Best Practices and Implementation Support

Note: To find more information, visit these Web sites:

- IBM IT Service Management

  http://www.ibm.com/software/tivoli/features/it-serv-mgmt/

- Information Technology Infrastructure Library (ITIL)

  ITIL began as a series of reference books and has evolved into an important standard for IT service management best practices. The intent of ITIL is to align your Information Technology with your business requirements, improve service quality, and lower the long-term cost of IT service provision. These concepts are consistent with the objectives of an SOA. More information on ITIL can be found at this Web site:

  http://www.ibm.com/software/tivoli/features/it-serv-mgmt/ITIL.html

12.1.4 IBM Tivoli products for SOA management

There are many Tivoli management products available for a wide range of solutions. In this section we focus on the following product families used for SOA management:

- IBM Tivoli Monitoring (ITM) products
- IBM Tivoli Composite Application Manager products

IBM Tivoli Monitoring (ITM) products

IBM Tivoli Monitoring (ITM) is the primary product that provides the base infrastructure of 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 greater 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 work space consoles. The TEP 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 for customization through workspace views, situations (thresholds) and workflows.

  A database manager such as DB2 UDB, must be installed on the same host system prior to the TEPS installation. This prerequisite is necessary because the TEPS installation will create the required TEPS database and supporting tables. Also, an ODBC (data source name) is configured to connect directly to the Tivoli Data Warehouse database. This ODBC connection is used whenever a pull of historical data from the Data Warehouse is requested.

- 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

    The desktop client is a Java application.

  - TEP Browser client

    The browser client loads a Java applet 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 Tivoli Enterprise Console® (TEC)
  - IBM OMEGAMON XE product family
  - IBM Tivoli Monitoring product family
  - NetView® for z/OS (release 5.2)

- Tivoli Enterprise Monitoring Agents

  The Tivoli Enterprise Monitoring Agents are installed on the systems or subsystem 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.

Note: Monitoring applications are identified by product code, a three-letter identifier that starts with the letter K. Some examples of product codes are:

- KUM: Universal Agent (included with ITM)
- KNT: Windows OS monitoring
- KD4: ITCAM for SOA
- KYN: ITCAM for WebSphere
- KT2: ITCAM for RTT

IBM Tivoli Composite Application Manager products

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

Note: For more information on IBM Tivoli Composite Application Manager for SOA, refer to the IBM Redbook *IBM Tivoli Composite Application Manager V6.0 Family: Installation, Configuration, and Basic Usage*, SG24-7151.

- ITCAM for SOA V6.0

  Monitor and manage the services layer of a SOA solution.

  Note: For more information on ITCAM for SOA refer to the following:

  - 12.2, “ITCAM for SOA product overview” on page 417

    This section provides a more detailed look at the features provided by ITCAM for SOA.

  - 9.5, “Implement ITSO Monitor Server node” on page 330

    This section describes how to install and configure ITCAM for SOA for the ITSO working example.

  - 12.3, “ITCAM for SOA working example” on page 430

    This section includes a working example scenario for using ITCAM for SOA to manage and monitor services for the ITSO working example.

- ITCAM for Response Time Tracking V6.0

  Proactively recognize, isolate, and resolve transaction performance problems.

- ITCAM for WebSphere V6.0

  Isolate the root cause of bottlenecks in a WebSphere application runtime environment.

- ITCAM for CICS Transactions V6.0

  Capture data from CICS systems to be analyzed in ITCAM for Response Time Tracking.

- ITCAM for IMS Transactions V6.0

  Capture data for IMS systems to be analyzed in ITCAM for Response Time Tracking.

- OMEGAMON XE for Messaging

  Monitor WebSphere MQ family runtimes and provide automatic corrective actions to improve performance and availability.

Note: Additional information on the ITCAM family of products can be found in "ITCAM family products" on page 488.

ITCAM product family and Candle terminology

IBM acquired the Candle company which was incorporated into the Tivoli brand of solutions. It is important to note that the ITCAM family of products is a result of integration of Candle OMEGAMON products and IBM Tivoli products.

Table 12-1 provides a mapping of the new IBM Tivoli terminology and the old Candle terminology. Although the products have acquired the new IBM terminologies and names, you can see references to the old OMEGAMON names within the product and documentation. ITCAM for SOA is designed to operate within the IBM Tivoli Monitoring.

Table 12-1 Summary of ITCAM product family and Candle terminology

IBM Tivoli terminology	Candle/OMEGAMON Terminology
IBM Tivoli management services	OMEGAMON Platform
IBM Tivoli Monitoring	OMEGAMON for distributed products
Tivoli Enterprise Monitoring Server	Candle Management Server (CMS)
Tivoli Enterprise Monitor Agent	OMEGAMON Monitoring Agent®
Tivoli Enterprise Portal Server	CandleNET Portal Server
Tivoli Enterprise Portal	CandleNET Portal
Situation Event Console	Enterprise Event Console/Event Console

IBM Tivoli terminology	Candle/OMEGAMON Terminology
Tivoli Enterprise Portal	Candle Management Workstation
Data Warehouse	Candle Data Warehouse

12.1.5 SOA management use case for the Service Creation scenario

As described in Chapter 12, “Manage and monitor services with ITCAM for SOA” on page 405, the Service Creation scenario has four realizations. For the purposes of illustrating an SOA management use case, we focus on the Indirect Exposure realization of the Service Creation scenario. We use a banking scenario theme as an example.

The Indirect Exposure realization application components are (see Figure 12-3 on page 416):

- J2EE Web Services client application hosted by WebSphere Application Server. The WS client application is used by online bank customers with a Web browser.
- J2EE Web Services hosted by WebSphere Application Server Network Deployment Edition. The middle-tier Web services wrap session EJBs that use the CICS ECI resource adapter to communicate with the CICS Transaction Gateway (CTG) to access the CICS Transaction Server (CICS TS). This exposes the bank application functionality running on CICS.
- CICS COMMAREA COBOL banking application is hosted by the CICS TS.

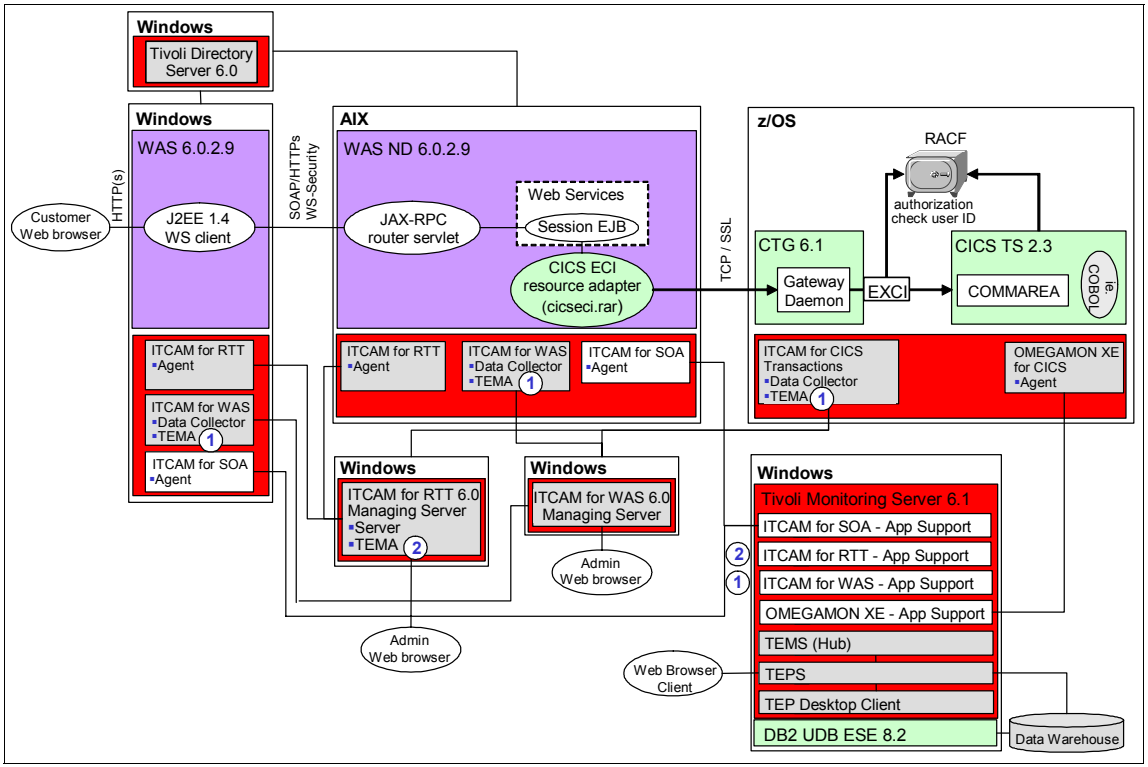


Figure 12-3 SOA management for the Service Creation scenario Indirect Exposure realization

The SOA management components are as follows:

- IBM Tivoli Monitoring Server V6.1
- ITCAM for SOA V6.0
- ITCAM for WebSphere V6.0
- ITCAM for Response Time Tracking V6.0
- OMEGAMON XE for CICS V3.10

The key requirements of SOA management for this scenario are:

- Identify resources to manage within each architectural layer in support of composite application.
- Integration of management products with central TEMS / TEPS

  By integrating the data collected from the agents and data collectors of ITCAM for WebSphere, ITCAM for RTT, ITCAM for SOA, and OMEGAMON XE for CICS into a common Tivoli Monitoring Server V6.1 (TEMS / TEPS), we establish an environment for a centralized data view.

- Create customized TEP workspaces and situations for centralized monitoring view

  TEP provides the capability to create customized workspaces and situations (events and thresholds) to have a customized view of the data collected for the composite application. The Operator uses this customized workspace and situations to monitor the composite application with an end-to-end view.

- Monitoring and troubleshooting

  After the customization of the TEP is complete, we are ready to validate monitoring and troubleshooting effectiveness. Troubleshooting is to be performed by the Operator from the customized TEP with the ability to drill down to the specific management products to give detailed analysis of the problem. If the Operator is not able to resolve the problem in 15 minutes, a management specialist for the specific management products is assigned for more in-depth troubleshooting.

  ITCAM for RTT and ITCAM for SOA will be used as key indicators to alert the Operator of a problem. We look at two use cases to identify and troubleshoot problems.

  ITCAM for RTT is used to monitor the transactional performance of the application, with specific threshold metrics established by SLAs. ITCAM for RTT will provide an end-to-end transactional performance view of the application within the TEP. For example, service response time for banking customer to complete a funds transfer. The problem could be slow performance as a result of the service not working, connection pooling issue, memory leak, and so on. Provided the system is customized properly, we expect to launch into the appropriate management product for lower level troubleshooting.

  ITCAM for SOA is used to monitor the composite application services components. This includes service consumers and providers. We will monitor for availability, performance and set thresholds. We will drill down to individual management components to further manage and troubleshoot.

12.2 ITCAM for SOA product overview

IBM Tivoli Composite Application Manager (ITCAM) for SOA V6.0 is used to monitor and manage services in a SOA solution running on Web application servers and the WebSphere Service Integration Bus. ITCAM for SOA monitors and performs simple control of message traffic between Web services.

We have organized the overview of ITCAM for SOA into the following topics:

- Key features summary
- Product packaging

- Component architecture
- Common usage scenarios in a support environment

Note: For more information on IBM Tivoli Composite Application Manager for SOA V6.0 refer to the IBM Redbook *IBM Tivoli Composite Application Manager V6.0 Family: Installation, Configuration, and Basic Usage*, SG24-7151.

12.2.1 Key features summary

The key features of IBM Tivoli Composite Application Manager for SOA are:

- Service problem identification and resolution

  Content-rich views and cross-workspace linkages enable drill-down from services to application components and IT resources to identify the source of a bottleneck or failure.

- Integrated Console

  Integrated with IBM Tivoli Monitoring with common Tivoli Enterprise Portal Server (TEPS). Service views, alerts and automation included within TEPS.

- TEP workspaces for service monitoring
  - Performance Summary, Messages Summary, Faults Summary, Configuration
- TEP situations leveraged for customer specified thresholds
  - Number of messages received by a service or operation within a specified time window
  - Size of the messages
- Basic mediation support
  - Ability to reject messages from a particular client, or reject all messages to a service
  - Ability to log requests/response messages
- Heterogeneous platform coverage
  - Support for Microsoft .NET, BEA WebLogic, IBM WebSphere Application Server JAX-RPC and SCA, IBM DataPower, and JBoss.
  - Target IBM ESB Platforms: WebSphere Application Server 5.x and 6.x, WebSphere Business Integration Server Foundation 5.1.1, and IBM DataPower

- Life-cycle Management

  Web Services Navigator provides deep understanding of service flows and relationships to developers and architects using operational data from Tivoli Data Warehouse.

12.2.2 Product packaging

ITCAM for SOA works with several application server environments including IBM WebSphere Application Server, Microsoft .Net, and BEA WebLogic Server, and is supported on a variety of platforms.

Note: Additional functionality available in Sparkler Web downloads

Additional functionality for ITCAM for SOA is available for Web download. The downloads are known as *Sparklers*. The Sparklers provide support for SCA in IBM WebSphere Process Server, IBM WebSphere Enterprise Services Bus (WESB), IBM DataPower, JBoss, SAP NetWeaver, and IBM WebSphere Community Edition. More information about ITCAM for SOA Sparklers can be found at this Web page:

http://www.ibm.com/software/sysmgmt/products/support/IBMTivoliCompositeApplicationManagerforSOA.html

Note: Platforms and prerequisites

For information on the platforms and prerequisites, refer to the *Installation and User's Guide, IBM Tivoli Composite Application Manager for SOA* found in the product InfoCenter at:

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

Section 9.5, "Implement ITSO Monitor Server node" on page 330, describes how we installed ITCAM for SOA for our working example runtime environment.

Simply put, ITCAM for SOA includes a set of server components, and a set of agents that are installed on the target node to monitor. Table 12-2 lists the server and agent software components, as well as the target node on which they are installed.

For simplicity, we have identified only the application server node for the location where the agent is to be installed, primarily because this is the focus of our

working example. There are other agents available, such as an agent to monitor a DataPower device.

Table 12-2 Summary of software components of ITCAM for SOA

Description	Installation node
IBM DB2 Universal Database V8.2	Monitor Server
IBM Tivoli Enterprise Monitoring V6.1: ► Tivoli Enterprise Monitoring Server ► Tivoli Enterprise Portal Server ► Tivoli Enterprise Portal Desktop Client ► Tivoli Enterprise Monitoring Agents – Tivoli Enterprise Monitoring Agent Framework – Universal Agent – Warehouse Proxy – Monitoring Agent for OS – Summarization and Pruning Agent	Monitor Server
IBM Tivoli Monitoring - Application Support **Note:** This is the ITCAM for SOA server component	Monitor Server
IBM Tivoli Composite Application Manager for SOA V6.0 **Note:** This is the ITCAM for SOA agent component	Application Server
IBM Web Services Navigator for Windows (Eclipse based). Can also be installed on an existing installation of IBM Rational Application Developer v6.x or other Eclipse based tools. **Note:** This is component is optional.	Developer node or Monitor Server

12.2.3 Component architecture

In the previous section we outlined the software components included with ITCAM for SOA. In this section we will describe how the components are used within the ITCAM for SOA architecture.

Figure 12-4 on page 421 depicts the ITCAM for SOA component architecture in its most basic form. We have included a brief description for each of the following components of ITCAM for SOA:

► Runtime environment:
  – Tivoli Enterprise Monitoring Server (TEMS)
  – Tivoli Enterprise Portal Server (TEPS)
  – Tivoli Enterprise Portal (Desktop and Browser Clients)
  – Tivoli Monitoring - Application Support
  – Tivoli Enterprise Monitoring Agents
  – ITCAM for SOA Agent (installed on Application Server node to monitor)

- Development environment:
  - IBM Web Services Navigator

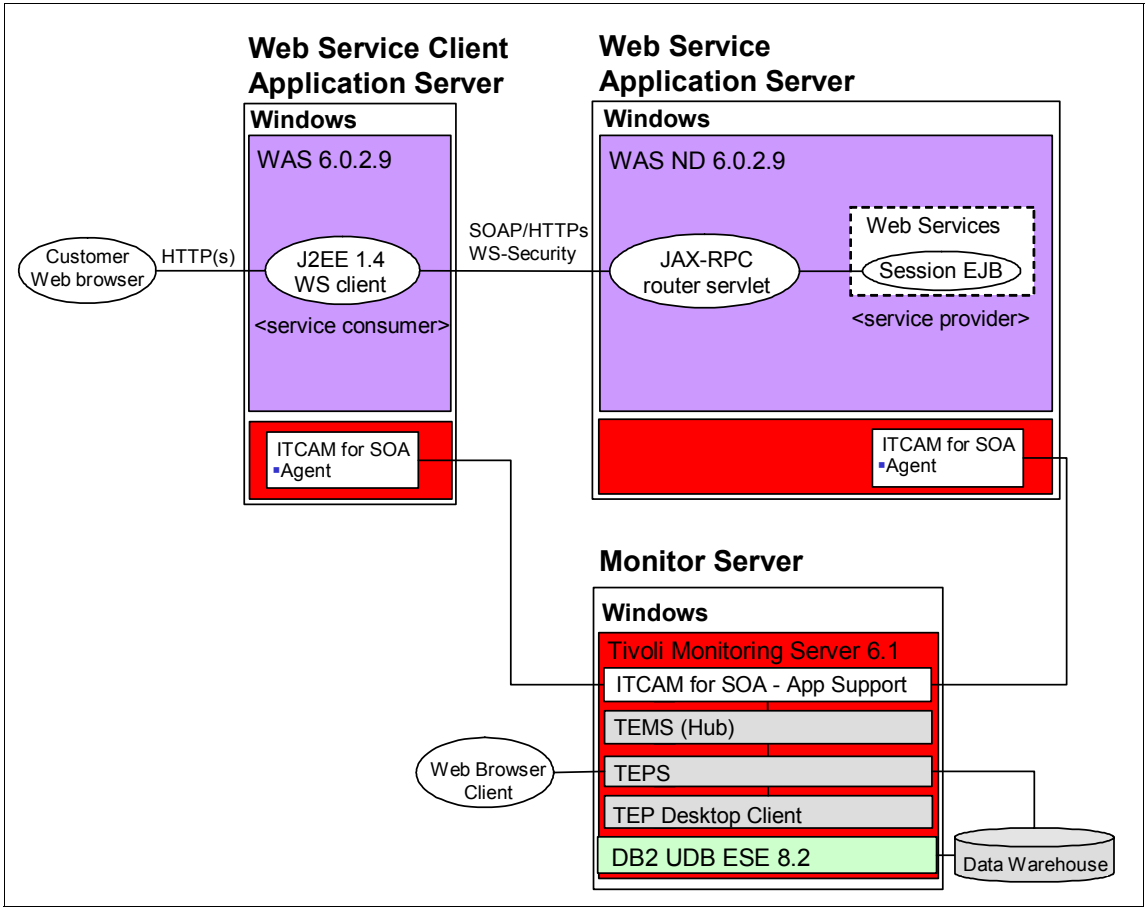


Figure 12-4 ITCAM for SOA component architecture

ITCAM for SOA V6.0 includes IBM Tivoli Monitoring V6.1 components (TEMS, TEP, TEP Desktop Client, TEM Agents), DB2 UDB ESE 8.2. We have listed the product specific components provided with ITCAM for SOA:

- Tivoli Monitoring - Application Support (ITCAM for SOA server module)
- ITCAM for SOA Agent (installed on Application Server to monitor)
- IBM Web Services Navigator
- Predefined workspaces for TEPS
- Attributes
- Situations
- Workflows
- Take Actions Commands

- Log files

Tivoli Monitoring - Application Support

The Tivoli Monitoring - Application Support is the server component for ITCAM for SOA. The Application Support provides and interface into the TEMS and TEPS to monitor and interact with ITCAM for SOA Agents.

ITCAM for SOA Agent

The ITCAM for SOA Agent is an application agent to the Tivoli Enterprise Monitoring Server with Application Support installed (ITCAM for SOA server). The ITCAM for SOA Agent is installed on the application server node to monitor and collect data for the Web Services layer and WebSphere Service Integration Bus.

The monitoring agent's function is to monitor traffic that flows between applications. The agent collects data on the objects being monitored. When you install ITCAM for SOA agent on an application server node, the installer also installs the code for the data collector. The data collector intercepts request and response messages for Web service calls that are being monitored.

The monitoring agent data collector is implemented as a JAX-RPC handler or service extension that is installed into the application servers that host the monitored Web services and their clients.

This handler is given control when either of the following actions occurs:

- A client application invokes a Web service. This is referred to as a *client-side* interception.
- The Web service request is received by the hosting application server. This is referred to as a *server-side* interception.

Similarly, when you install ITCAM for WebSphere (which is a separate installation from ITCAM for SOA), the installation installs a J2EE data collector inside WebSphere Application Server. This data collector then collects information on the usage of WebSphere Application Server environment. This information can be combined with the monitoring information from ITCAM for SOA agent to give you a very detailed picture of the runtime environment for WebSphere Application Server. However, for the purpose of this chapter, our focus is purely on ITCAM for SOA.

The monitoring agent records monitoring information into one or more local log files, or writes the information into a database for later retrieval and importation into the IBM Web Services Navigator. You can combine two or more log files from various monitored application servers into a single log file. When this combined log file is imported into the IBM Web Services Navigator, you can

display information about all of the monitored Web services and the interactions between them at the same time. You can also import multiple log files and easily switch between them without leaving the IBM Web Services Navigator or importing them again. This release of IBM Tivoli Composite Application Manager for SOA focuses on the simple object access protocol (SOAP) engine of IBM WebSphere Application Server, the Microsoft .Net framework, BEA WebLogic Server, and the WebSphere Service Integration Bus.

Note: The Web services data collector supports both Java 2 Enterprise Edition (J2EE) application client and server container environments because JAX-RPC handlers are supported only in these environments. It is possible to write Web services client programs that adhere to the conventions described in Java Specification Request 101 (JSR 101) and to run them inside a J2EE container or in a standard Java virtual machine (non-J2EE container).

The Java Specification Requests are implemented in such a way that within the WebSphere Application Server environment, the JAX-RPC handler might not be invoked by these client applications. To ensure proper operation of the JAX-RPC handler, verify that the client applications are written according to the conventions described in Java Specification Request 109 (JSR 109), Java Web Services for J2EE.

For additional information, refer to the specification found at the following Web site:

`http://www.jcp.org/aboutJava/communityprocess/final/jsr109/`

IBM Web Services Navigator

The IBM Web Services Navigator is a component installed on Eclipse V3.0.2 based products such as IBM Rational Application Developer or Rational Software Architect. The Web Services Navigator can be used to retrieve and visualize the characteristics of the Web services that are being monitored.

Metric information about Web services requests and responses is intercepted by ITCAM for SOA and is written to multiple log files and optionally stored in a database. IBM Web Services Navigator then retrieves this data from the database, and converts the data into a log file format that can be displayed in an Eclipse-based viewer.

IBM Web Services Navigator also can, with assistance from a companion tool, process locally stored metric and message content logs into a format suitable for importing. These log files are then imported into the viewer to generate visual representations of the various characteristics of the monitored Web services.

The IBM Web Services Navigator includes the following features:

- An Eclipse-based import wizard used to retrieve data from a database and format it into a local log file that can be imported into the IBM Web Services Navigator.
- A command line based Log Assembler tool used to combine multiple log files into a single log for importing into the IBM Web Services Navigator.
- A new perspective that contains a new set of views (service topology, transaction flows, flow patterns, statistics tables) in the Eclipse environment. Each of the views provides a different visual representation of your Web services data from the imported log file.

Predefined workspaces for TEPS

ITCAM for SOA comes with a set of predefined workspaces, that you can select from the Navigator view in the TEPS desktop client or Web browser interface. Each of the following workspaces has its own set of views that display Web services data and metrics in various levels of detail:

- Message Arrival workspace
- Performance Summary workspace
- Message Summary workspace
- Faults Summary workspace
- Services Management Configuration workspace

The views included in the workspaces at the Services Management Agent Environment level or below, except the Fault Details view, and are based on the Services Inventory table. This table contains a summary of calculations for the monitored Web services traffic. The calculations are performed in five-minute intervals. For each interval, the traffic is analyzed to calculate the fields available within the table. After the five-minute interval is finished, the record is marked as complete. All queries of the Services Inventory table that are provided with the product include a check for only complete records.

Attributes

Attributes are measurements that are collected by the IBM Tivoli Monitoring V6.x family of products. ITCAM for SOA stores specific measurement or attributes which are pertinent to its functions. Attributes are grouped into tables.

For a full list of details on the attributes and associated tables, refer to the following product guide:

- *Installation and User's Guide, IBM Tivoli Composite Application Manager for SOA*, GC32-9492

Situations

A *situation* is a type of an event that is generated in Tivoli management services. ITCAM provides predefined situations. These situations are defined to help you monitor critical activity and serve as a template for creating customized situations of your own.

A *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.

Predefined situations are activated when they are distributed to the node that you want to monitor. After they have been configured, the situation alerts provide ITCAM trigger event notification.

You can modify an existing situation to create your own. The default sampling interval for a situation is 15 minutes. If you modify an existing situation to create your own, do not forget to change the sampling time.

The predefined situations provided with ITCAM for SOA are:

- *Fault* monitors the messages in the Web services flow to determine whether a Web service fault has occurred.
- *Message Arrival Critical* raises an alert when an excessive amount of traffic occurs (for example, when the number of messages received from one or more remote clients exceeds a user-defined threshold).
- *Message Arrival Clearing* clears a previously triggered Message Arrival Critical situation. The situation can also be used to alert for message falls below a specified threshold, and create an alert for this lack of activity.
- *Message Size* monitors in length in bytes, for each message during the Web services flow. If the length is > threshold value, then the situation is triggered.
- *Response Time Critical m*onitors in milliseconds the elapsed round-trip response time, for the completion of a Web services request.
- *Response Time Warning* monitors in milliseconds the response time for the completion of a Web services request.

The default situations are based on service calls stored in the Service_Metric table. Analyzing on summarized information in the Service_Inventory table can reduce some of the monitoring overhead.

Workflows

Workflows are the starting point for creating your own situations. A workflow is a series of steps that are automated to respond to a particular situation.

Take Actions Commands

Take Action Commands are directives that can be run from the Tivoli Enterprise Portal desktop client or included in a situation policy. When included in a situation, the command runs when the situation becomes TRUE.

When you enable a Take Action command in a situation, you automate the response to system conditions.

Advance automation uses policies to perform actions, schedule work, or automate manual tasks.

A policy consists of a series of activities. *Activities* are automated steps that are connected together to create a workflow.

The ITCAM for SOA agent includes the predefined Take Action Commands as listed in Table 12-3. These commands change the configuration file for the agent and control the operation of the data collector.

Table 12-3 Predefined Take Action Commands for ITCAM

Take Action Command	Description
AddFltrCntrl	Creates new filter control settings to reject messages
AddMntrCntrl	Creates new monitor control settings
DelFltrCntrl	Deletes existing filter control settings
DelMntrlCntrl	Deletes existing monitor control settings
DisableDC	Disable data collection and the ability to reject messages
EnableDC	Enables data collection and the ability to reject messages
updateLogging	Defines the level of logging information
UpdMntrCntrl	Updates existing message logging levels for monitor control
updateTracing	Enables or disables tracing

The Take Action commands displayed in Table 12-4 can be executed interactively from the TEP desktop client or they can be executed automatically when included in a situation or a workflow. In a situation, the Take Action command is executed when the situation evaluates to true. As a result, you can automate a

response to system conditions. For example, you can use a Take Action command to send a message or start a process on the managed system.

There are two variants of the Take Action commands which are prefixed with SI- or SM-. The SI- and SM- prefix provide a convenience function by prefilling many of the command arguments. The commands that are prefixed with SI- correspond to the Service Inventory (KD4INV) attributes table. The commands that are prefixed with SM- correspond to the Services Message Metric (KD4SMT) table.

Note: For more detailed information for the attributes refer to the *Installation and User's Guide, IBM Tivoli Composite Application Manager for SOA*, GC32-9492 product guide.

Log files

Log files are created as a standard action when starting agents and Tivoli Enterprise Monitoring Server. Proper management of log file is important as the size and the number can grow depending on the managed systems and the level of activities in the systems

Following are some examples of log files:

- TEMS error log file, %CANDLE_HOME%\CMS\Logs\kmsras1.log
- %CANDLE_HOME%\CMS\{cms.msg | kdsmain.msg }
- Agent log files for ITCAM for SOA, c:\candle\cms\KD4\logs
- Metric logs, content logs, action logs, operation logs and trace logs

12.2.4 Common usage scenarios in a support environment

This section highlights the use of ITCAM for SOA and supporting Tivoli monitoring components for common usage scenarios in a support environment.

The scenario includes the following roles:

- Service Architect
- Service Application Support
- Application Developer
- Operator
- Administrator

Troubleshooting a support call

The Service Application Support specialist receives a service request indicating the customer has detected a problem with an application. The specialist recognizes that this application is dependent on many Web services.

To troubleshoot the problem, the specialist uses a Web browser to access the Tivoli Enterprise Portal (TEP). From the TEP Navigator view, the specialist displays the performance summary workspace for the agent running on the application server hosting the Web services.

From the workspace, the specialist notices that the response time for a particular service is nearing the threshold settings. The specialist knows that this service is running on a particular WebSphere Application Server, and launches the WebSphere Administrative Console.

Note: ITCAM for WebSphere V6.0 can be used to monitor WebSphere Application Server and managed from a common TEP with ITCAM for SOA.

The specialist determines from the WebSphere Administrative Console, that the application server is having some difficulty, but is not exceeding any threshold settings. The specialist updates the service request with this information and directs the service request to the WebSphere application support team for additional troubleshooting and resolution.

Notification of a problem with a key service flow

The service application support specialist receives notification by a situation that there is a problem with a critical service flow. This situation includes the hostname where the situation was generated. The specialist launches the TEP Desktop Client and navigates to the services Faults Summary workspace for the designated application server.

This workspace displays the number of faults returned by service operation, and a table showing the details of each fault returned. The application support specialist sorts the list to see only those faults from the particular service, and sees they are all receiving an application error.

The support specialist turns on the full message logging for the particular service, using the `AddMntrCntrl Take Action`. As the support specialist monitors the system, it continues to generate the same application error. The support specialist determines he or she has enough message data, and runs the `Log Assembler` tool to capture the message data, and then imports the resulting log files from the production systems into IBM Web Services Navigator. The support specialist then turns off message capture in production system using the `UpdMntrCntrl Take Action`, thus resetting the logging level to `None` to keep the system from collecting additional data.

The service architect launches IBM Rational Software Architect environment and opens the IBM Web Services Navigator (Eclipse based). The service architect creates a project and imports the selected log files using the Log Assembler tool

to display the information, including the message content. The service architect then focuses on the message content for one of the messages that received a fault. The message contents show incorrect data in the message body, so the service architect works with the application developer, who navigates to the application code, searching for the field in the message contents that contains the incorrect data. The application developer finds the code problem and fixes the problem. The application developer then performs a test with the fix. When satisfied the fix has been tested, the application change is deployed on the production system.

Review the deployment of services

The services architect has learned that a new set of services, designed to provide support for a set of critical business processes, have been deployed into production. Wanting to make sure the flows are as expected, the services architect launches the IBM Rational Software Architect environment, and opens the Web Services Profiling perspective.

The services architect then opens the connection to the warehouse data and views the data in the Service Topology view. The services architect can see that the flow from a topology view looks correct, and goes on to display the Flow Patterns view. From the patterns view, he or she can tell that the flows are as expected, but he or she notices that the workload is not equally dispersed between the available servers. The services architect opens a low priority trouble ticket with the information and assigns it to the service application support person to have a look.

Service-related event in TEP

The operator sees a service related event in the TEP event console. From the event the operator opens the associated situation page to display the situation values, and the expert advice. The advice indicates that for this event, the operator should look at the Performance Summary workspace for this service, followed by the application server information.

The operator navigates to the Performance Summary workspace for the system where the event was triggered. The operator can see not only the excessive response time that triggered the situation, but also that a number of calls to this same service are nearing the maximum allowed response time threshold. The operator navigates to another TEP workspace that manages the application servers. On the application server workspace, the operator sees that a threshold in memory utilization is being approached, though it has not yet passed the threshold. The operator examines the memory utilization and can see that the system seems to have less memory than it should, possibly indicating a hardware failure. The operator creates a trouble ticket and assigns it to the Server maintenance team to check the hardware for a problem.

Configure a TEP situation

An administrator responsible for configuring a TEP situation navigates to the Services Management Agent Environment and right-clicks to select the situations option. The administrator selects the existing Message Arrival Critical situation, right-clicks on it and selects Create Another to create another situation in the Message Arrival table. The administrator modifies the situation with the correct number of messages, and information about each client to monitor. The administrator modifies the expert advice to indicate a that flood of messages are being received from a client and the identified client needs to be reviewed for a problem. The administrator sets the threshold and saves the work as a new situation.

Configure historical data collection

The administrator navigates to the History Configuration, and selects ITCAM for SOA. The administrator then configures the History collection for the Services Inventory and Services Metric tables, to collect the data every five minutes, and to send it to the warehouse database every hour. The administrator now has an environment configuration that is generating a significant traffic load. It is not necessary to use the IBM Web Services Navigator on the production data on a regular basis, so the administrator turns off the collection of the Services Metric data, but continues the collection of the Services Inventory table.

Configure a Workflow

The administrator navigates to the Workflow Editor, and selects the existing Message Arrival Critical policy. The administrator right-clicks the name and clicks Copy Policy, which creates another policy with the template of the original. The administrator then modifies the action with the rejection configuration for the service, and saves his work as a new policy.

12.3 ITCAM for SOA working example

The ITCAM for SOA working example is organized into the following sections:

- Working example overview
- Discovery of services
- Managing and monitoring services
- Customizing the TEP for monitoring

12.3.1 Working example overview

In order to maximize the value of ITCAM for SOA, it is important to understand the relationship of the services and application resources being monitored. Typically this is not the job of a support person in an operations environment.

The operation person performs the initial investigation, such as looking at trace and log information to identify the cause of the problem. This sort of capability is provided out-of-the-box with ITCAM for SOA.

To gain more application-specific information, some customization in the TEP is required. This can be performed by the support person working in conjunction with an application developer. We monitor the Web services of the ITSO Car Rental application. for the working example.

For the purpose of testing, we use the internal ITSO Car Rental Web services client application to invoke Web services of the ITSO Car Rental application. The application can be accessed at this Web address:

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

We focus on the four use cases displayed in Figure 12-5. The user initially logs in to the application, enters the itinerary, selects a vehicle and makes a booking. As a result of this booking, the service consumer of the Web service invokes the operations for the service (aka `CarRentalManager`). The provider is also `CarRentalManager` which responds by invoking its operations. As a result, Web service traffic is generated which can be monitored by ITCAM for SOA. There is an internal admin function which also generates Web service calls.

Use cases:

1) **Login**
2) **Itinerary and preferences**
3) **Select a vehicle**
4) **Review and Confirm: Book**

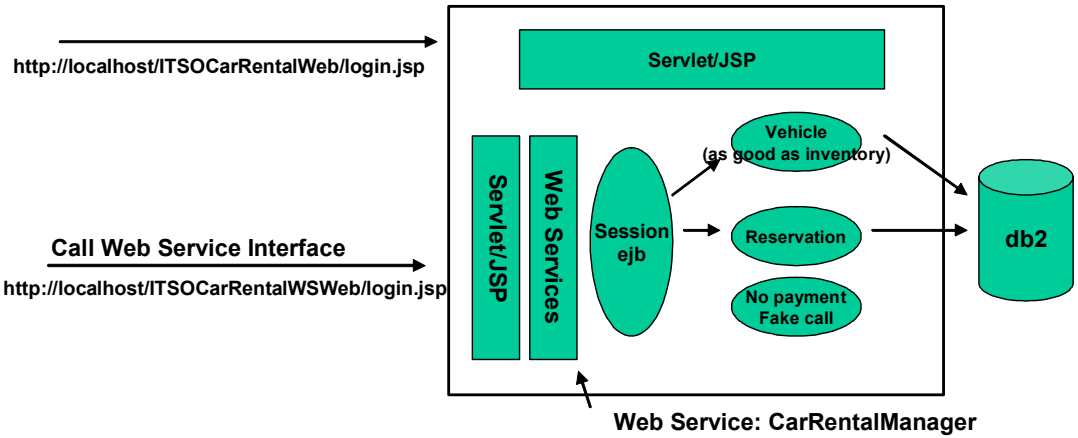
Consumer

Service Name: CarRentalManager
Operations: checkAvailability()
createReservation()
showAllReservations()
showAllVehicles() – 2.5 seconds delay

Provider

Service Name: CarRentalManager
Operations: checkAvailability()
createReservation()
showAllReservations()
showAllVehicles()

http://localhost/ITSOCarRentalWSWeb/adminlogin.jsp
(Admin ID= admin)



http://localhost/ITSOCarRentalWeb/services/CarRentalManager
http://localhost/ITSOCarRentalWeb/services/CarRentalManager/wsdl/CarRentalManager.wsdl

Figure 12-5 Use Cases and Web Service calls in our test scenario

Tip: ITCAM for SOA gives you a summary of Web service calls out-of-box in a non-intrusive way without requiring any specific monitoring for the application. This helps identify what Web services are running (for example, who the service consumers (requestors) and providers are by looking at the message summary.

If you want to perform any customization in monitoring, filtering calls or creating threshold events, some application knowledge is required to understand the flow of messages. It is useful to create a use case scenario to identify the Web services as well as the relationship between the service providers and consumers, such as those in Figure 12-5 to help formulate monitoring strategy.

In our scenario, the working example highlights the following ITCAM for SOA features:

- Discovery of services
- Managing and monitoring services
- Customizing the TEP for monitoring

Note: Our sample application, includes some delays to simulate performance issues. For example, the operation showAllVehicles() has a time delay of 2.5 seconds. This allowed us to monitor and demonstrate this as a potential bottleneck in the Web service.

12.3.2 Discovery of services

This section describes how to discover the Web services found in the ITSO Car Rental sample application using the ITCAM for SOA tooling.

Complete the following tasks:

- Prerequisites
- Launch the Tivoli Enterprise Portal Desktop Client
- Configure TEMS to validate users
- Explore the Tivoli Enterprise Portal Workspace
- Generate traffic by running the Web services application
- Review the Web services traffic in TEP

Prerequisites

This section describes the prerequisite tasks that must be completed before discovering Web services:

1. Ensure the runtime environment is installed and configured. For this chapter, our focus is on the ITSO Application Server node (itsoapp1) and the ITSO Monitor Server node (itsomon1).

   For details refer to Chapter 9, “Implement the runtime environment” on page 281.

2. Ensure the following Tivoli Windows services are started on the ITSO Monitor Server node (itsomon1):
   - Tivoli Enterprise Monitoring Svcs - TEMS1
   - Tivoli Enterprise Portal Server
   - IBM DB2 UDB services
   - Tivoli Universal Agent Primary (optional)
   - Tivoli Warehouse Proxy (optional - requires that DB2 is started).

3. Ensure the following servers and agent are started on the ITSO Application Server node (itsoapp1).

   a. Start the Deployment Manager as follows:

   ```
   cd \IBM\WebSphere\AppServer\profiles\Dmgr01\bin
   startmanager
   ```

   b. Start the node agent for the federated Application Server node as follows:

   ```
   cd \IBM\WebSphere\AppServer\profiles\AppSrv01\bin
   startNode
   ```

   c. Start the server1 application server for the Application Server as follows:

   ```
   cd \IBM\WebSphere\AppServer\profiles\AppSrv01\bin
   startserver server1
   ```

   d. Ensure the ITCAM for SOA Primary Windows service is started.

   Note: In addition, you can verify the agent status by do the following:
   - Click **Start** → **Programs** → **Candle OMEGAMAN XE** → **Manage Candle Services** to launch the Services Console.
   - If successfully started, the status will display `started`.

4. Ensure the ITSO Car Rental Application is deployed to the ITSO Application Server node.

   For details refer to Chapter 10, “Deploy application to WebSphere Application Server” on page 351.

Launch the Tivoli Enterprise Portal Desktop Client

This section includes a procedure to launch the Tivoli Enterprise Portal (TEP) Desktop Client.

Note: If the Tivoli Enterprise Portal does not start properly, refer to “Tivoli Enterprise Monitoring Server” on page 501 for some troubleshooting tips.

Do the following on the ITSO Monitor Server node, to launch the TEP Desktop Client:

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.

2. When the Logon dialog box opens, logon with this information as seen in Figure 12-6 and then click **OK**:
   - Logon ID: `sysadmin`

     Note: The default Logon ID is sysadmin.
   - Password: leave blank

     Note: By default the `sysadmin` ID does not have a password set. Leave the password field blank.

   Note: If the Tivoli Enterprise Monitoring Server has been set to Validate User, the user ID for Tivoli Enterprise Portal must be added to the Windows user account (local or domain). See “Configure TEMS to validate users” on page 436 for details.

   Just before publishing this book, we learned that after the IBM Tivoli Monitoring V6.1 Fixpack is installed, a password is required.

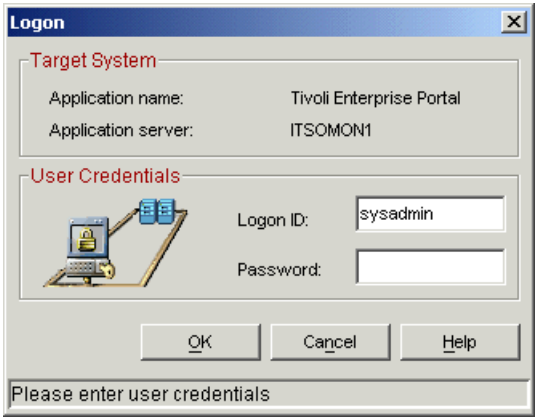


Figure 12-6 Tivoli Enterprise Portal Desktop Client Logon

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

Configure TEMS to validate users

To configure Tivoli Enterprise Monitoring Server to validate a user, do the following:

1. Create a Windows user ID.
   a. Right-click **My Computer** and select **Manage**.
   b. Expand **Local Users and Groups**.
   c. Select **User**, right-click and select **New User**.
   d. When the New User logon dialog appears, enter the following and then click **Create**:
      - User name: `sysadmin`
      - Password: *<password>*
        Note: Enter a password of your choosing.
      - Deselect **User must change password at next logon**.
   e. Click **OK**.
2. Click **Start** → **Programs** → **IBM Tivoli Monitoring** → **Manage Tivoli Enterprise Monitoring Services**.
3. From the Console, right-click **Tivoli Monitoring Server** and select **Reconfigure**.
4. When the Tivoli Enterprise Monitoring Server Configuration dialog box opens, check **Security:Validate User**, and then click **OK**.
5. Click **OK**.

6. Restart the Tivoli Enterprise Monitoring Svcs - TEMS1 Windows service.

Note: Now that Validate User is enabled, you must specify the logon ID and password to logon to the Tivoli Enterprise Portal.

Explore the Tivoli Enterprise Portal Workspace

This section provides a quick review of the Tivoli Enterprise Portal default workspace.

1. After you have logged into the Tivoli Enterprise Portal Desktop, you can see the default workspace. Expand **Enterprise** → **Windows Systems** → **ITSOAPP1** → **Services Management Agent** (see Figure 12-7 on page 437).

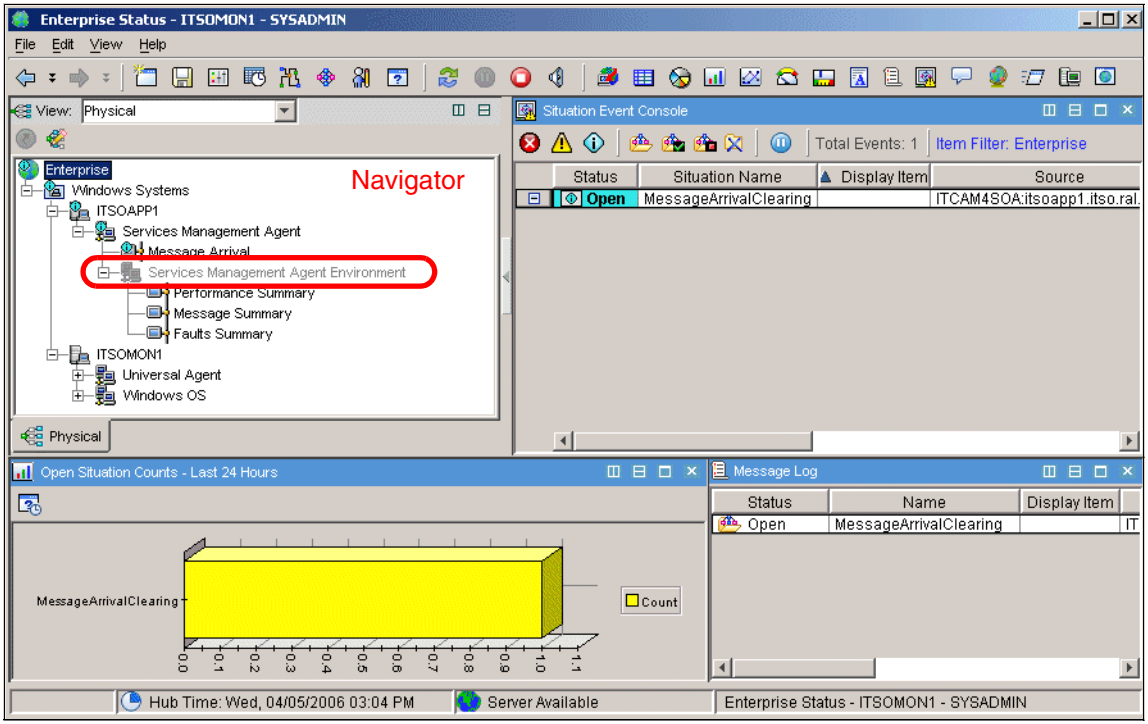


Figure 12-7 Tivoli Enterprise Portal Desktop default workspace

Troubleshooting tip: If you do not see the Services Management Agent Environment displayed in the default workspace as seen in Figure 12-7, the ITCAM for SOA agent may not be started or may be mis-configured.

Refer to “Tivoli Enterprise Portal” on page 502 for troubleshooting tips.

2. The Services Management Agent Environment will be greyed out, until the Web service flow has been observed 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 use the ITSO Car Rental Web application which includes Web services.

Note: For more detailed information on using the ITSO Car Rental application, refer to Chapter 11, “Walk through the application” on page 381.

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

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

2. When the logon dialog box opens, enter a user ID and click **Login**.
3. When the Itinerary and preferences dialog opens, enter the desired criteria and click **Submit**.

   By clicking **Submit**, the `showAllVehicles()` operation for the CarRentalManager Web service is invoked. The showAllVehicles operation is a candidate to be monitored by ITCAM for SOA.

4. When the Select a vehicle dialog box opens, click **Select** for the desired vehicle.

   By clicking **Select**, the checkAvailability operation will be invoked.

5. When the Review and Confirm dialog box opens, click **Book Now**.

   By clicking **Book Now,** the `createReservation` operation will be invoked.

6. When the reservation confirmed page opens, click **Logout**.

Note: We have listed a few possible methods for generating traffic for testing:

- Manually generate traffic by using the application as described above.
- Develop and run a Java application to invoke services to generate traffic.

  For more information refer to 8.9, “Develop a Java client application to generate traffic” on page 273.

- Use IBM Rational Performance Tester (RPT)

  For more information refer to the following URL:

  `http://www.ibm.com/software/awdtools/tester/performance/index.html`

Review the Web services traffic in TEP

After the Web services traffic is generated, it can be reviewed within the TEP Desktop workspace.

1. Now that the Web services traffic has been generated, click the **Refresh Now (F5)** button found in the toolbar of the Tivoli Enterprise Portal Desktop.
2. There are several icons to take note of in the toolbar, which as used to launch editors and display graphs (see Figure 12-8 on page 439).

   Clicking these icons launches an editor such as the Situation Editor. You can drag and drop the display objects to an empty view during customization.

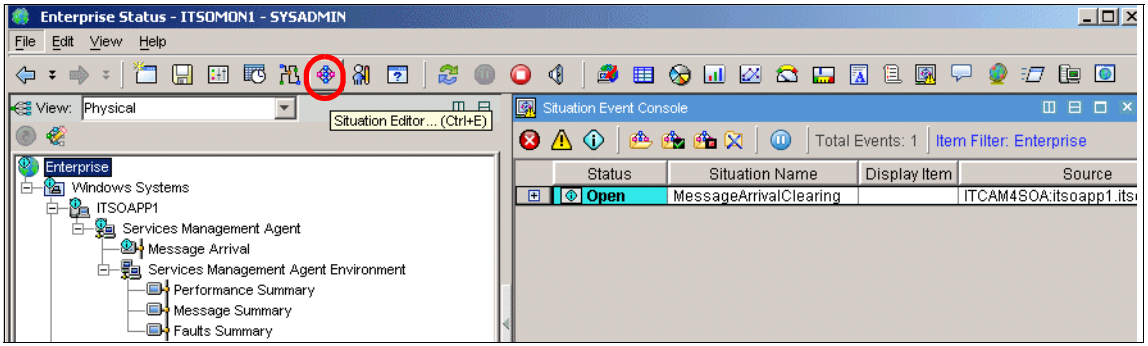


Figure 12-8 TEP Workspace with editor icons on the top toolbar and information and warning icons

3. 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 such that it reaches the defined threshold, the `MessageArrivalCritical` situation will become True. Similarly when the number of messages drop below the threshold, the `MessageArrivalClearing` situation will become true as shown in Figure 12-9 on page 440.

4. For example, notice the blue information icon for `MessageArrivalClearing` in Figure 12-9. You can click the link (icon with blue chain) and it will take you to another workspace which provides more details about the situation.

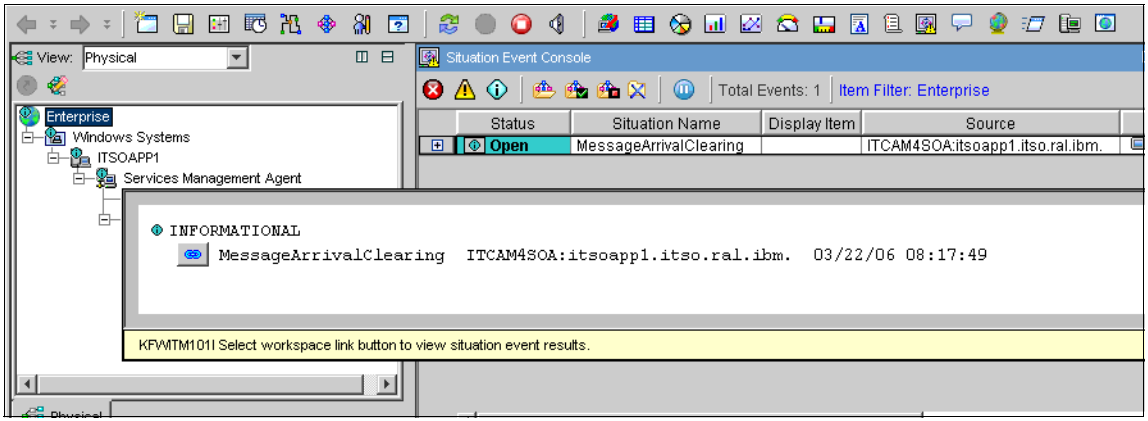


Figure 12-9 TEP Workspace with editor icons on the top toolbar and information and warning icons

5. In TEP, the icon with an arrow in the top left corner above the **Navigator** is green, click it to update the workspace.
6. Click various summaries under the Services Management Agent Environment (for example, Performance Summary, Message Summary, Faults Summary).

   If you see charts showing data on the service, `CarRentalManager` and its operations, then the Web services traffic between requesters and providers have been intercepted by ITCAM for SOA, indicating discovery of services.

12.3.3 Managing and monitoring services

The ITCAM for SOA workspaces in the Tivoli Enterprise Portal are arranged to show Web services calls by servers. The Web services calls are typically identified by the following attributes:

- Frequency
- Response time
- Message length

This section includes the following examples which demonstrate key features of ITCAM for SOA used to manage and monitor Web services:

- Modify collection interval for History Collection Configuration
- Display the Message and Performance Summary views
- Turn Data Collector On/Off
- Turn on/off logging
- Turn on/off tracing
- Control flow of Web services traffic
- Filtering a Web service call

Modify collection interval for History Collection Configuration

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

1. From the TEP Desktop default workspace, select **Edit** → **History Configuration**.
2. When the History Collection Configuration dialog box 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 **1 hour** for Warehouse interval.
   e. Click **Configure Groups**.
   f. Click **Services_Inventory** from the Select Attribute Groups.
   g. Click **Start Collection** as seen in Figure 12-10 on page 442.

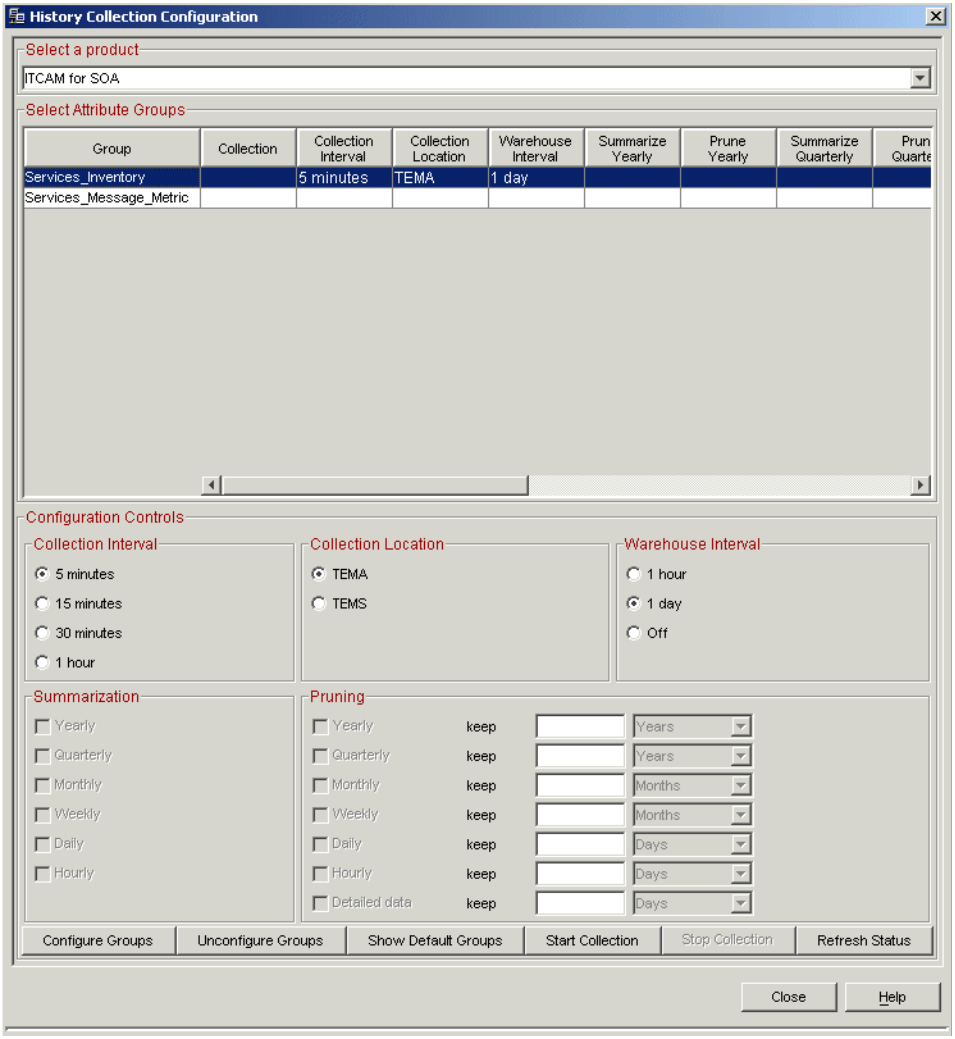


Figure 12-10 History Collection Configuration - set collection interval

h. Repeat the above process for **Services_Message_Metric**.
i. Click **Close**.

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

Display the Message and Performance Summary views

To display the Message Summary and Performance Summary views within Tivoli Enterprise Portal, follow these steps:

1. Generate Web services traffic by running the ITSO Car Rental Web services application.

   Refer to “Generate traffic by running the Web services application” on page 438 for details.

2. Launch the TEP Desktop Client and logon.

   Refer to “Launch the Tivoli Enterprise Portal Desktop Client” on page 435 for details. Note, if following the redbook we set the password for the sysadmin user in “Configure TEMS to validate users” on page 436.

3. Select **View** → **Refresh Now (F5)**.

4. Display the Message Summary view.

   a. Click **Services Management Agent**.

      It should display the predefined agent configuration information about Data Collector Monitor and Global Configuration and Data Collector Filters. Currently there are no user defined filters and configurations.

   b. Click **Message Arrival** under System Management Agent.

      Notice the message arrival details in the workspace (see Figure 12-11 on page 444).

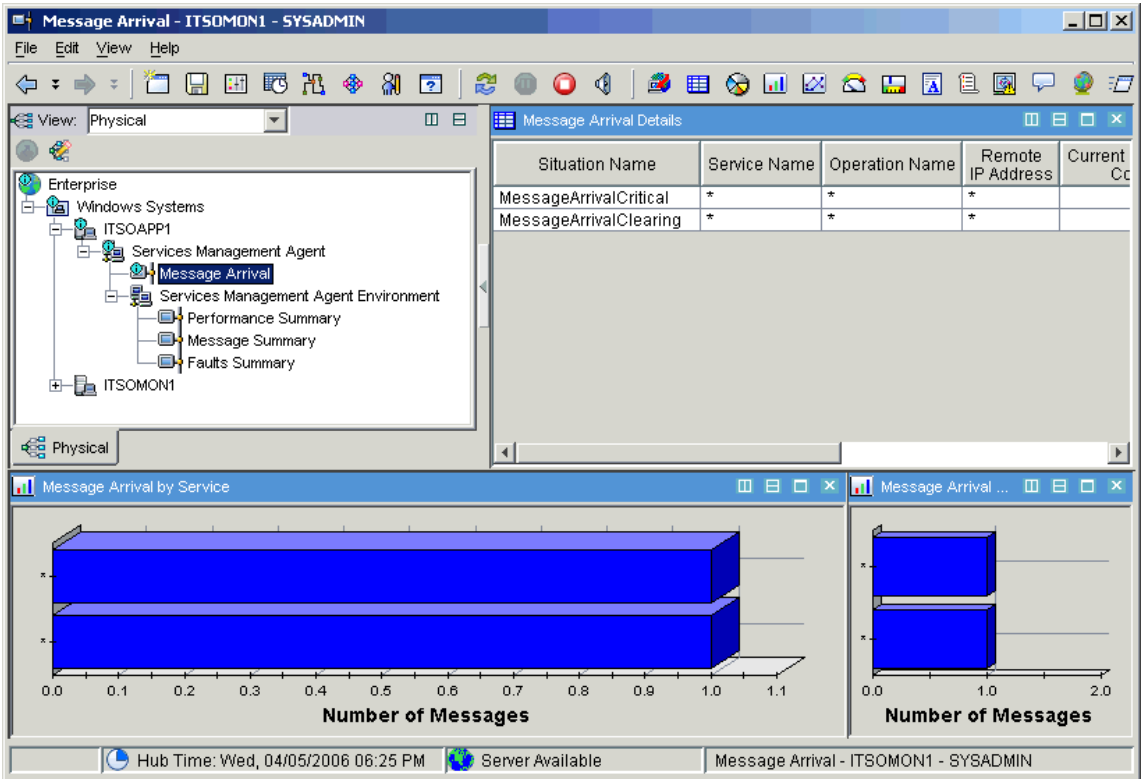


Figure 12-11 Message Arrival

c. Rerun the ITSO Car Rental application a few times.

d. Refresh in TEP by selecting **View** → **Refresh Now**.

   Notice the message arrival by service and operation data has changed.

e. Click **Message Summary** under Service Management Agent Environment.

   This provides information on the following as seen in Figure 12-12 on page 445:

   - Number of messages by service operation type
   - Average message size by service operation type

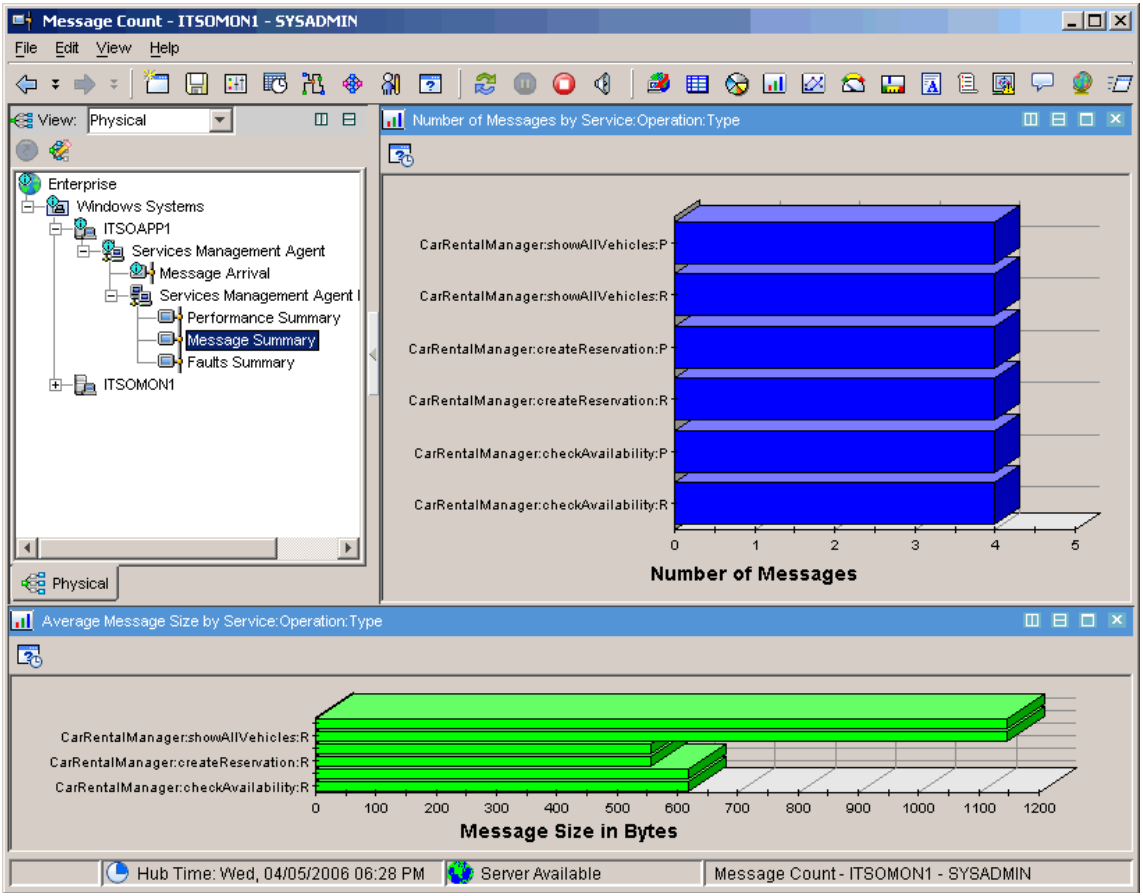


Figure 12-12 Message Summary

f. Click **Services Management Agent Environment** to get more detailed information on the messages, calls and average response time for Web services (see Figure 12-13 on page 446).

Notice the average response time for the operation, `showAllVehicles()` has the largest size for messages and it takes longer to execute. In a real-world scenario you could expect some delay, if a long list of vehicle details have been transmitted in the call.

Note: We added an artificial delay of two seconds in our sample application to simulate more realistic Web application behavior. In a real monitoring scenario, this would be a visual trigger point for further investigation of the Web service call.

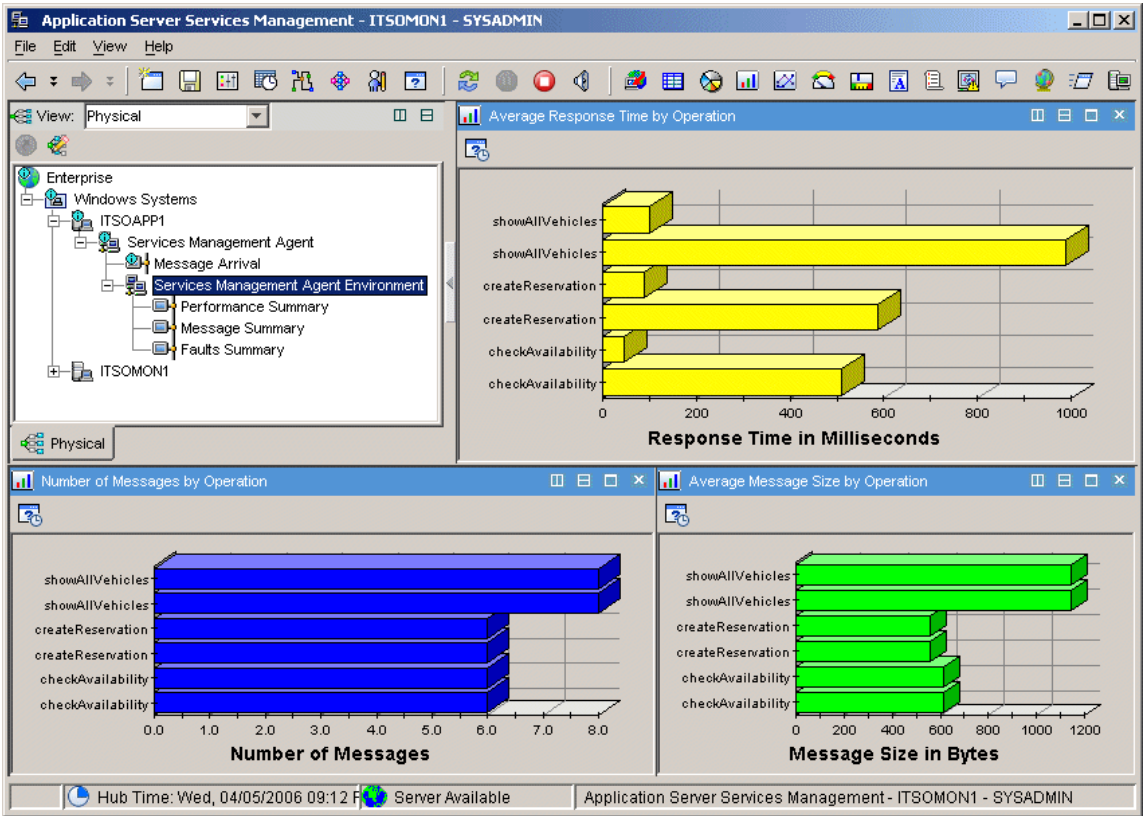


Figure 12-13 Metrics on Web service operation

5. Display the Performance Summary view.

   Click **Performance Summary** under Services Management Agent Environment (see Figure 12-14 on page 447).

   The Performance Summary provides the following:

   – Service inventory by identifying the service requester and provider (for example, `CarRentalManager`).

   – Identifies which Web services operations were performed (for example, `showAllVehicles`).

   – Statistics on the response time for the calls.

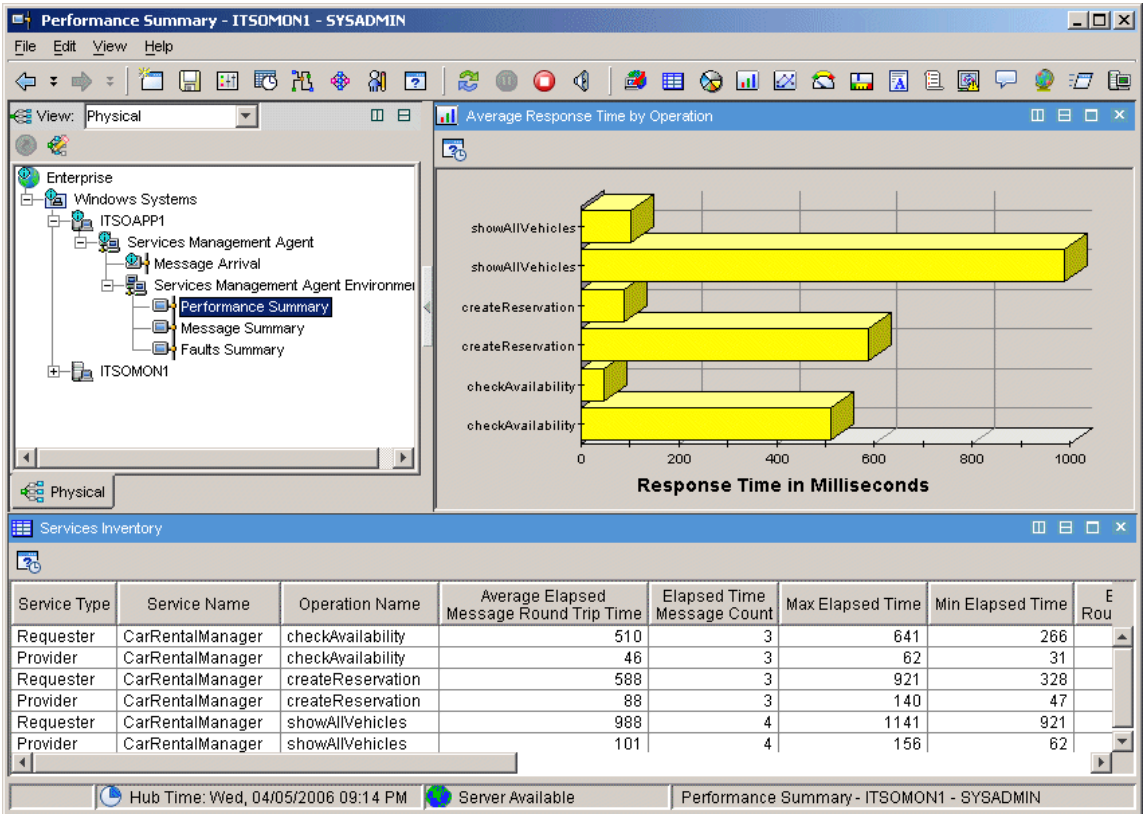


Figure 12-14 Performance Summary

Turn Data Collector On/Off

The action command EnableDC is used to set data collection for a specific application server environment, and DisableDC is used to turn off data collection. Data collection can be turned on and off for an application server. Data collection is turned on by default.

This example demonstrates how to turn off data collection. We will show that no data is being collected for the Web services. Next, we will turn data collection back on.

1. Select **Services Management Agent**.
2. Turn off data collection.
   a. From the Data Collector Global Configuration window, right-click **WebSphere_Application_Server** and select **Take Action** → **Select**.

Note: Notice that the Data Collector On/Off value is set to *On* (see Figure 12-15).

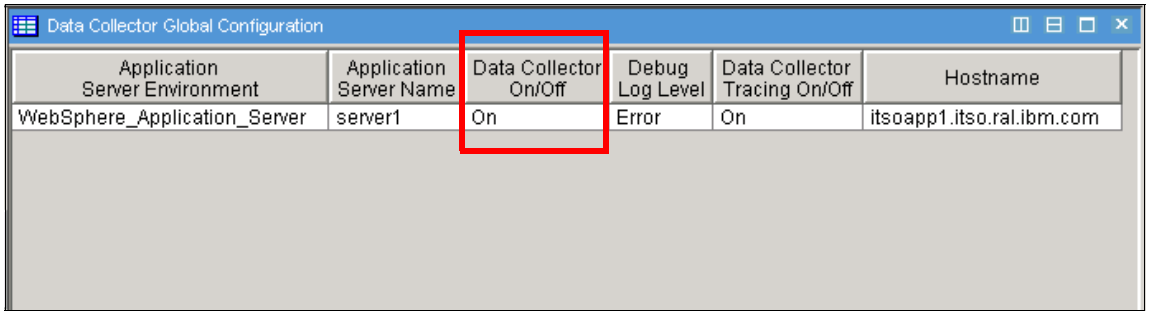


Figure 12-15 Data Collector Global Configuration window

b. When the Take Action dialog box opens, select these items (see Figure 12-16):

   i. Select **DisableDC** from the Name drop-down list.

   ii. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm** from Destinations Systems and click **OK**.

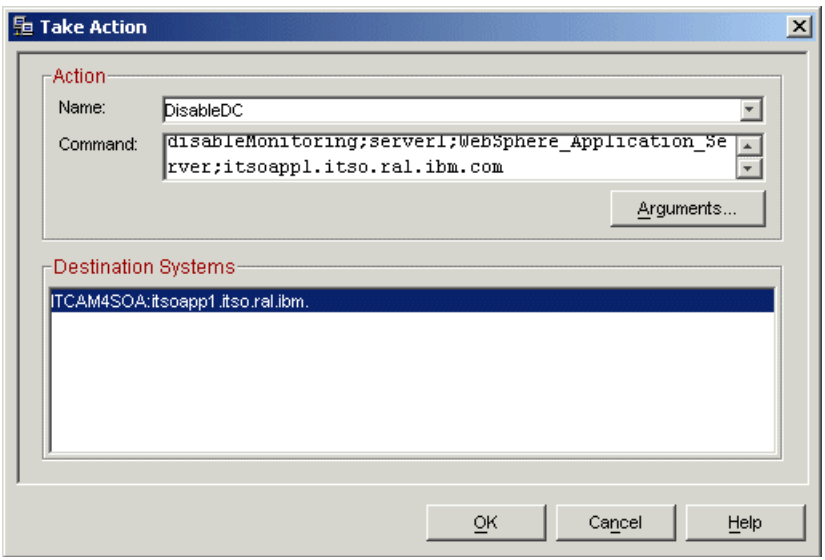


Figure 12-16 Take Action dialog

c. When the Action Status dialog box opens, you should `Return Code: 0`, to indicate the command was successful. Click **OK**.

The Return codes and description are listed in Table 12-4.

Table 12-4 Return codes for Take Action commands

Take Action Command Return Code	Description
0	Action completed successfully
1	No change was made to the monitoring criteria
2	The action completed unsuccessfully

d. Select **View** → **Refresh Now**, and verify the status change in column named *Data Collector* is now set to Off.

e. Generate new traffic by re-running the ITSO Car Rental application.

f. Select **View** → **Refresh Now**.

g. Check the following summary views and notice that no data has been collected on the Web services traffic since data collection is turned off.

i. Click **Performance Summary**
ii. Click **Message Summary**.
iii. Click **Faults Summary**.

Note: It is important to note that the sample data from the agent is transmitted every five minutes for real-time updates and every hour hour for historical data (according to our History Configuration in previous steps).

This means after running the ITSO Car Rental application, if you then refresh immediately in the TEP, the time interval may not have yet expired for sampling. As a result, you might not see the desired effect of data collection immediately. In that case wait for at least five minutes, then perform another refresh.

3. Set data collection to **On**.

After you are finished testing, use the action command EnableDC to set data collection to **On**.

a. From the Data Collector Global Configuration window, right-click **WebSphere_Application_Server** → **Take Action** → **Select**.

b. When the Take Action dialog box opens, select these items:

   i. Select **EnableDC** from the Name drop-down list.
   ii. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm** from Destinations Systems, and click **OK**.

c. When the Action Status dialog box opens, you should `Return Code: 0`, to indicate the command was successful. Click **OK**.

d. Select **View** → **Refresh Now (F5)**.

   Verify the status for Data Collector On/Off is set to On within the Data Collector Global Configuration window.

Turn on/off logging

The logging action is used to set the level of logging for a Web service interception point. An interception is made when a Web service call is made by the Web services client (requester), as well as when the Web service call is received by the client. In both cases, the interception is recorded in the logs. Similarly two logs are recorded for the server (provider). Hence a Web service round trip leads to four logs being recorded.

The log information is categorized as follows:

- Error
- Warning
- Informational

This example demonstrates how to turn on and off content logging.

1. Select **Service Management Agent**.
2. Turn on content logging.

   a. From the Data Collector Global Configuration window, right-click **WebSphere_Application_Server** and select **Take Action** → **Select**.

   b. When the Take Action dialog appears, do the following:

      i. Select **updateLogging** from the Name drop-down list.

      ii. The Command field is prefilled with default arguments. Click **Arguments** to display Edit Argument Values window.

      iii. Change the value for Debug_Log_Level. The default value is set to Error. It may be set either `Info` or `Warning`. For our example, we entered the value `Info` as seen in Figure 12-17 and clicked **OK**.

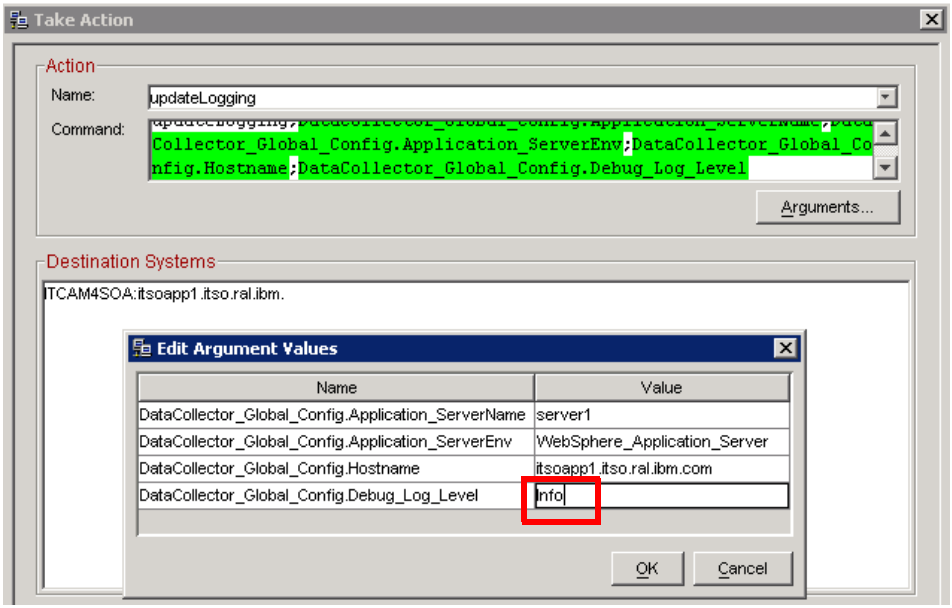


Figure 12-17 Set log level to Info

iv. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm** from Destinations Systems, and click **OK**.

c. When the Action Status dialog box opens, you should `Return Code: 0`, to indicate the command was successful. Click **OK**.

d. Select **View** → **Refresh Now (F5)**.

The Debug Log Level value should display *Info* within the Data Collector Global Configuration window.

e. Run the ITSO Car Rental application to generate Web services traffic.

3. Turn off content logging.

a. From the Data Collector Global Configuration window, right-click **WebSphere_Application_Server** and select **Take Action** → **Select**.

b. When the Take Action dialog box opens, select these items:

i. Select **UpdMntrCntrl** from the Name drop-down list. Set the value of `Message_Logging_Level` to be **None** in order to turn off logging.

ii. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm** from Destinations Systems, and click **OK**.

c. Select **View** → **Refresh Now (F5)**.

d. Alternatively, to delete a previously defined AddMntrCntrl command, select the original entry associated with it, in Data Collector Monitor Control Configuration view. Right-click the entry and issue DelMntrCntrl.

Turn on/off tracing

The tracing action is used to turn on/off Data Collector tracing by setting the appropriate value in the Data Collector Global Configuration table. The trace information associated with Web services traffic is written to a trace log file by the Data Collector.

To turn on/off tracing, do the following:

1. Click **Service Management Agent**.
2. From the Data Collector Global Configuration window, right-click **WebSphere_Application_Server** → **Take Action** → **Select**.
3. When the Take Action dialog box opens, complete these tasks:
   a. Select **updateTracing** from the Name drop-down list.
   b. The Command field is prefilled with default arguments. Click **Arguments** to display the Edit Argument Values window.
   c. Set the value for DataCollector_Global_Config.TraceOnOff to **On** if not already set. Click **OK**.
   d. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm** from Destinations Systems and click **OK**.
4. When the Action Status dialog box opens, you should `Return Code: 0`, to indicate the command was successful. Click **OK**.
5. Select **View** → **Refresh Now (F5)**.

   The Data Collector Tracing On/Off value should display On, within the Data Collector Global Configuration window.
6. Run the ITSO Car Rental application to generate Web services traffic.
7. Change to the `C:/CANDLE/CMA/KD4/logs directory` on the ITSO Application Server node (itsoapp1). This is application server where ITCAM for SOA Agent is installed and running.
8. Verify that the new log file exists.

   There should be a new log file in the `C:/CANDLE/CMA/KD4/logs` directory. In our example, the log file is named:

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

   The log is an ASCII file and therefore can be easily read. Use an editor to read the contents of the file.

Note: At the time of writing this book using ITCAM for SOA V6.0, the log file time stamp was not updated as new entries are written to the log file. The log file entries within the log file to have the proper timestamp.

The log files are moved periodically to the following directory:

`C:/Candle/CMA/KD4/logs/KD4.DCA.CACHE/archive`

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 prefilled to reduce key strokes for editing the argument for your needs.

By 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 or operation, then message data for all services and operations is logged.

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 12-5 and then click **OK**.

Table 12-5 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 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 12-5 on page 454).

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 12-6 lists the order in which the criteria are evaluated.

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 the following book:

- *Installing and Troubleshooting IBM Web Services Navigator*, GC32-9494.

Table 12-6 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 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 will demonstrate how set the filtering to reject messages for `Provider.CarRentalManager.showAllVehicles()`.

1. Navigate to **Performance Summary** under Services Management Agent Environment.
2. In 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 prefills the arguments.
4. Click **Arguments**. The 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 **Services Management Agent** item in the navigator.

   The Data Collector Filter Control Configuration table should show an additional entry with details of 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 reason that the Data Collector failed to reject the message was that it did not recognize the remote IP address because 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 for making 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 with 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 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 Action Status dialog box opens, you should see `Return Code: 0` to indicate the command was successful. Click **OK.**
17. Repeat steps 2-7. 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 application error where the Web page failed to display and issued 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. 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 message. This may be because we only want to allow 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.

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 both running on the same system.
6. Select **ITCAM4SOA:itsoapp1.itso.ral.ibm.com** from Destinations 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 in **Faults Summary** to ensure that the fault on `showAllVehicles()` is not raised.

12.3.4 Customizing the TEP for monitoring

This section demonstrates how to customize TEP for the following common requirements:

- Create a situation to monitor a threshold
- Create a custom workspace
- Create a custom workflow

Create a situation to monitor a threshold

A situation describes conditions you want to test on a managed system. The condition is a set of attributes used to tested against a threshold within a filtering rule. If the situation evaluates to TRUE at predefined intervals then invoke an automated response. Invoke the Situation Editor for all managed systems by selecting **Edit** → **Situation Editor** (Ctrl+E).

Note: The predefined, threshold level for `MessageArrivalCritical` situation is > 50 messages arriving within a 90-second time interval.

The default threshold level for `MessageArrivalClearing` is < 40 messages arriving within a 90-second time interval. This situation clears a previously triggered `MessageArrivalCritical` situation.

You can view the definitions of these situation by invoking the Situation Editor from the top toolbar. In the Situation Editor, use the Explorer to navigate to Services Management Agent, click MessageArrivalCritical to view its definition.

Similarly, click **Services Management Agent Environment** to view the definitions of other predefined situations for MessageSize. For example, when Message Length exceeds 1600 bytes in a Web service call or ResponseTime (elapsed round trip response time between Requester and Provider) exceeding 10000 milliseconds.

The predefined situations are activated when they are distributed to a host that you want to monitor. The also serve as a templates for creating your own situations.

The predefined situations are created from data in the Service Metric (KD4SMT) table, so you can generate your own situations based on the data in the table. You can also use the Service Inventory table to create situation based on average data. The metrics are calculated on a 5 minute time intervals.

The high-level steps to create a situation are:

- Create a situation from an existing situation.
- Define or modify the formula (for example, define the query and conditions).
- Define or modify the action for the situation (for example, what happens when situation evaluates to TRUE). You can send a message, run a command or invoke another situation which might be to reject further messages.
- Define the duration of the situation (for example, how long is the situation valid for?).
- Start the situation to make it active.

In our scenario we want to create a situation with the attributes listed in Example 12-1.

Example 12-1 Definition of a situation for operation showAllVehicles

```
situation = when round trip response time on CarRentalManager.showAllVehicles()
> 100 milliseconds
if the situation evaluates to TRUE, then start rejecting messages for
CarRentalManager.showAllVehicles()
```

1. Navigate to the specific agent and the specific application server to define a situation. For our scenario, right-click **Performance Summary** and select **Situations**.
2. From the Situations Editor for Performance Summary, select **ResponseTimeCritical** and then click the icon **Create another Situation** (plus sign in Situation Editor toolbar) as seen in Figure 12-18.

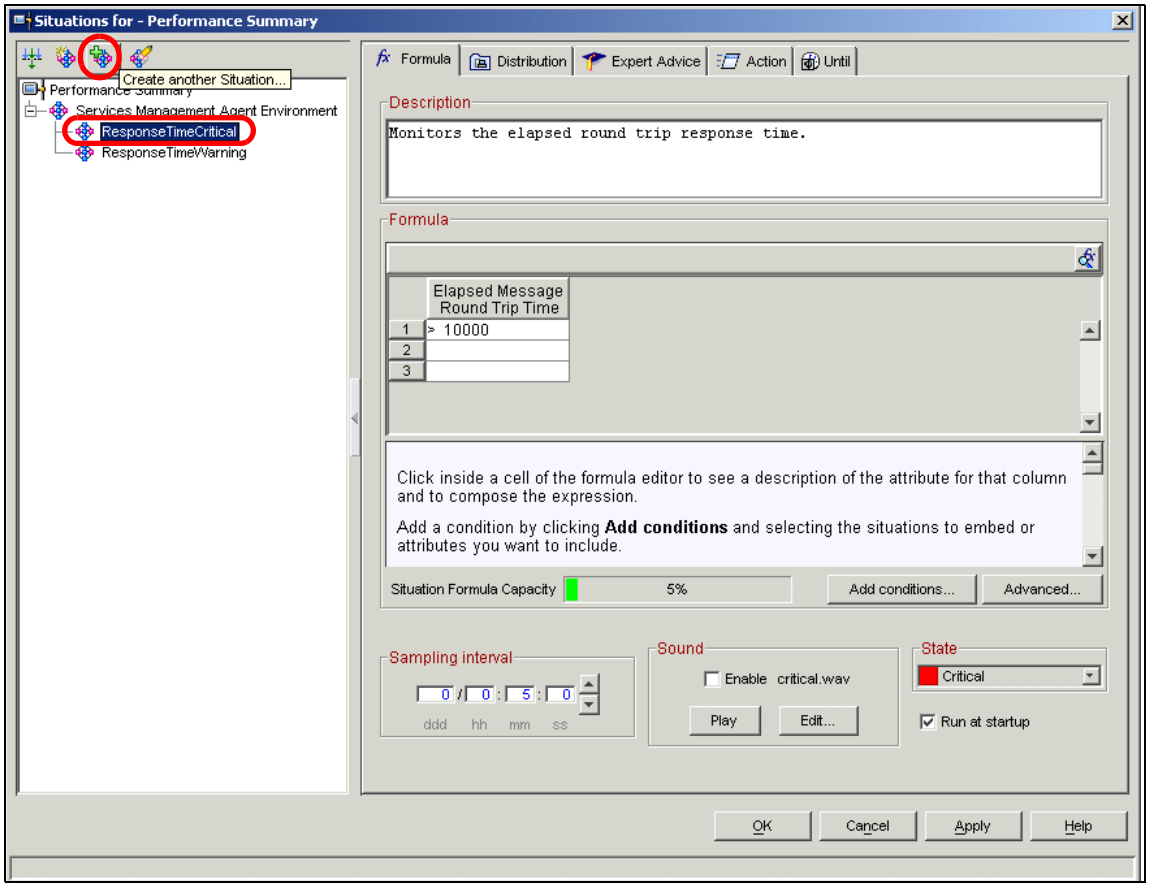


Figure 12-18 Example of a Situation Editor for Performance Summary - ResponseTimeCritical

3. When the Create Situation windows opens, enter `showAllVehiclesCritSit` in the Name field, enter `Reject messages when roundtrip > 100 milliseconds` in the Description field, and then click **OK**.
4. Select **showAllVehiclesCritSit** from the list.

Note: In the Situations Editor, there are five tabs labeled Formula, Distribution, Expert Advice, Action and Until. These describe when the situation is raised and what action should be taken when the situation occurs and for how long. The Distribution tab describes the node names and application server to which the situation applies.

When the situation becomes true, an event indicator (in this case yellow warning sign) lights up in the Navigator. When you hover over the indicator, a dialog box opens. In the dialog box you can click the situation list and it takes you to a workspace for that event. In the workspace, it shows the Expert Advice message as a view.

In Expert Advice, you can enter text embedded with HTML tags to define steps or instructions to perform an action. You can also add links to other HTML pages, using <a href "http://www.ibm.com/tivoli">ITCAM for SOA Diagnostics

In Situation Editor, check the online help for more details about how you can embedded scripting language to build more detailed diagnostic guidance.

5. Click the **Formula** tab.
   a. Click the cell in column Elapsed Message Round Trip Time and row 1.
   b. Enter value `100` (default 10000) in the Elapsed Message Round Trip Time field.
   c. Click **Add Conditions**.
   d. When the Select condition window opens, select **Operation Name** and **Service Name** from the Attribute Item list, and then click **OK**.

      Note: Use the Ctrl key to make multiple selections.

   e. Click the cell in column Operation Name and row 1, and enter the value `'showAllVehicles'`.
   f. Click cell in column Service Name and row 1, and enter the value `'CarRentalManager'`.

g. In the panel Formula, click the blue icon with symbol fx in the top right corner to display the formula as follows:

```
( Elapsed Message Round Trip Time > 100 AND Operation Name ==
'showAllVehicles' AND Service Name = 'CarRentalManager')
```

h. Check the **Show detailed formula** at the bottom of the panel. Notice it shows the table name and the attribute names used in the formula. Click **OK**.

i. Click the **State** drop-down list, and select **Warning** (yellow).

6. Click the **Distribution** tab.

   a. Select **D4:<52587ec2>:itsoapp1-server1** from the Available Managed Systems list.

   b. Click the left blue arrow to move `server1` to the Assigned list.

   c. Click **Apply**.

7. Click the **Expert Advice** tab.

   a. In Text or Advice Location, select the current text and overwrite with the following new text:

```
The response time for CarRentalManager.showAllVehicles() exceeded 100
milliseconds.
```

   b. Click **Preview** and then click **Close**.

8. Click the **Action** tab.

   a. In System Command field, enter the following:

```
net send * ITSO Car Rental App Warning: Response Time critical threshold
reached.
```

   b. Notice you have the option to run the system command at the Agent node or at the TEMS server node. Accept the default (TEMS).

9. Click the **Until** tab.

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

   b. Click **Apply**.

10. Right-click **showAllVehiclesCritsit** and select **Start Situation**.

11. Click **OK**.

12. When finished the Situation Editor screen for `showAllVehiclesCritSit` should look like Figure 12-19 on page 463.

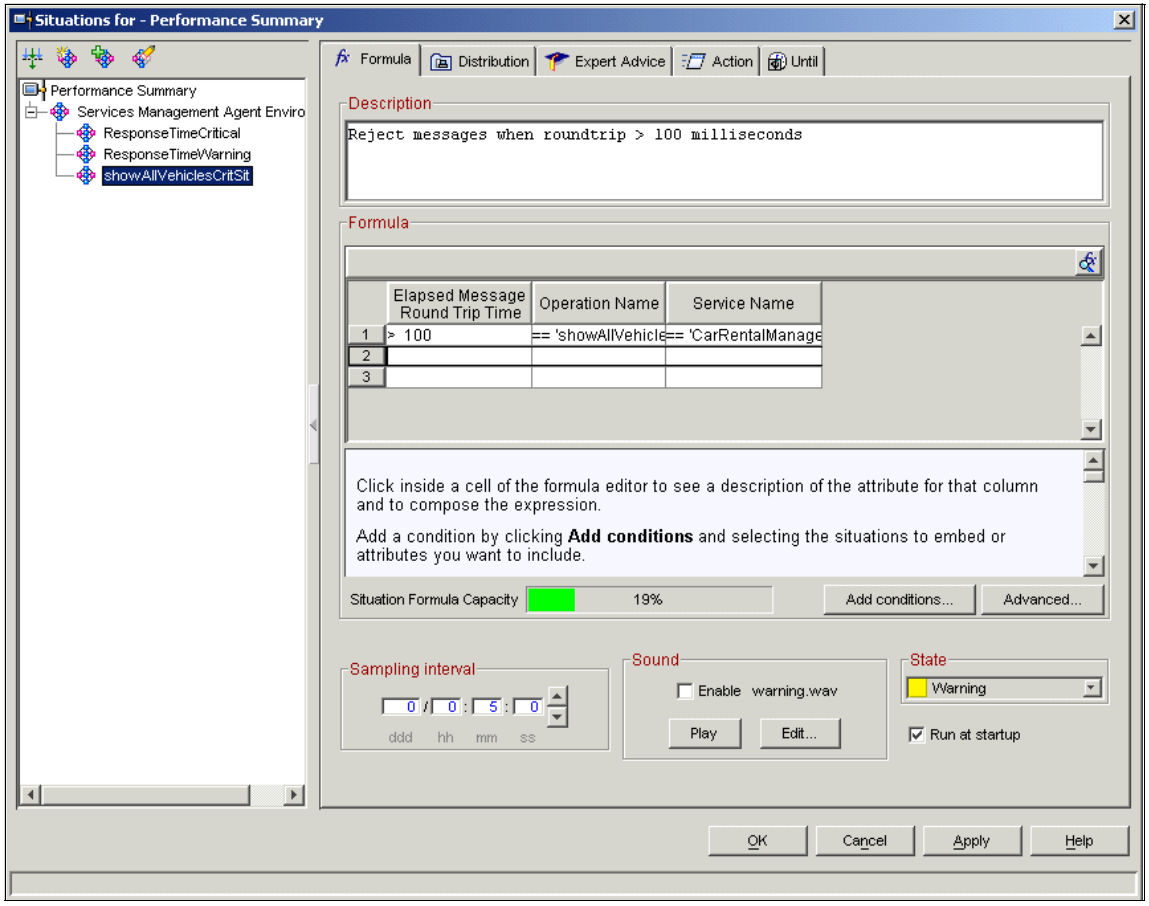


Figure 12-19 Situation formula editor for showAllVehicles

13. Rerun the ITSO Car Rental application in succession for five or six times. This should cause the situation to become true and, consequently, you should see Warning messages appearing in the workspace, as shown in Figure 12-20 on page 464.

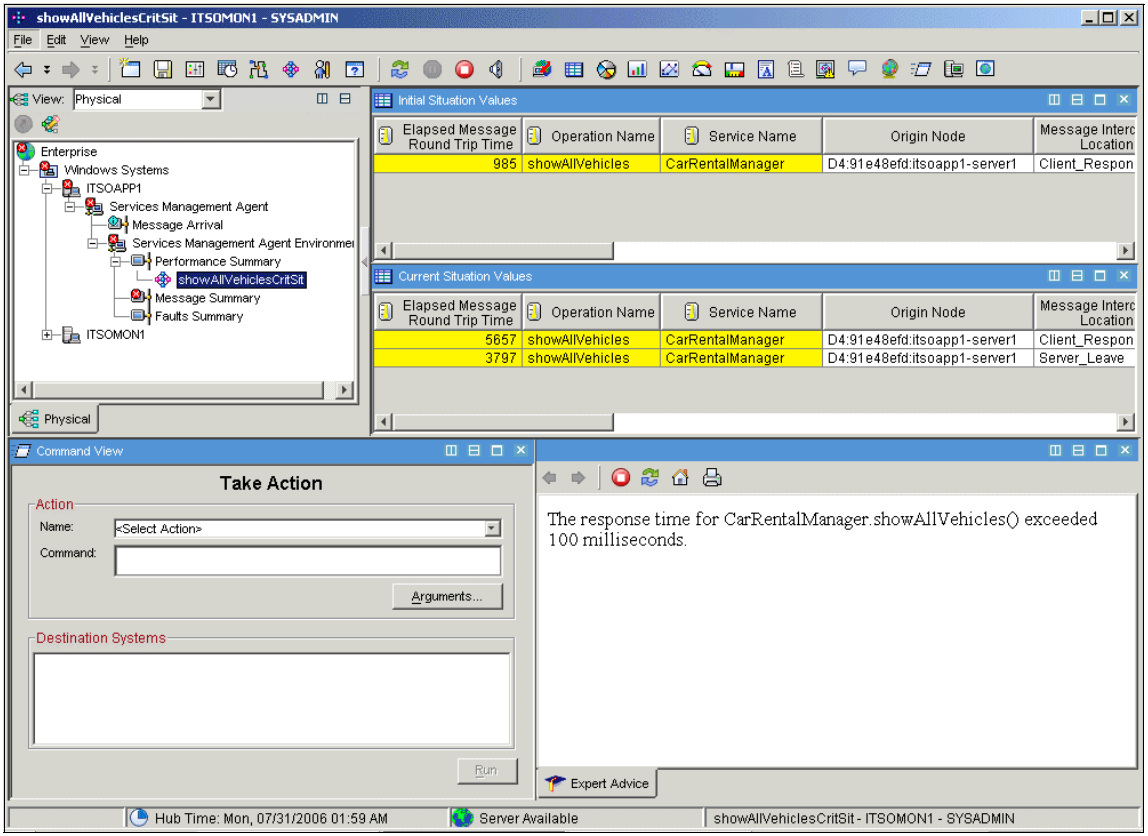


Figure 12-20 Warning message when situation showAllVehiclesCritSit becomes True

Create a custom workspace

In this scenario, we create a very basic customized workspace in TEP named ITSOCarRentalWorkspace. The idea of a customized workspace is to take a predefined workspace and customize it to be application-specific so that it can provide views which are more specific to the application operations.

Complete the following steps to create a customized workspace consisting of graphs, charts, and tabular information for metric data:

1. In the top left of the Navigator window, click **Edit Navigator View** icon .
2. When the Edit Navigator View window opens, click **Create New Navigator View** icon within the Target View.
3. When the Create New Navigator View window opens, enter the following and then click **OK**:
   - Name: `ITSOCarRentalWorkspace`

– Description: `Customized Workspace for ITSO Car Rental App`

4. From the Source View, expand the Explorer by clicking **Enterprise -> Windows System**.
5. Select **ITSOAPP1** node, drag it and drop it on top of **ITSOCarRentalWorkspace** in Target View.
6. Click **Close**.
7. In the Navigator, if you should see a message `KFWITM024I 1 Navigator update pending`, **click** the green icon with arrow  to update the Navigator.
8. At the top of the Navigator, select **ITSOCarRentalWorkspace** to edit the workspace from the in View drop-down list.

   The newly created workspace looks similar to the one in Figure 12-21. We expanded ITSOAPP1 and children in the Navigator.

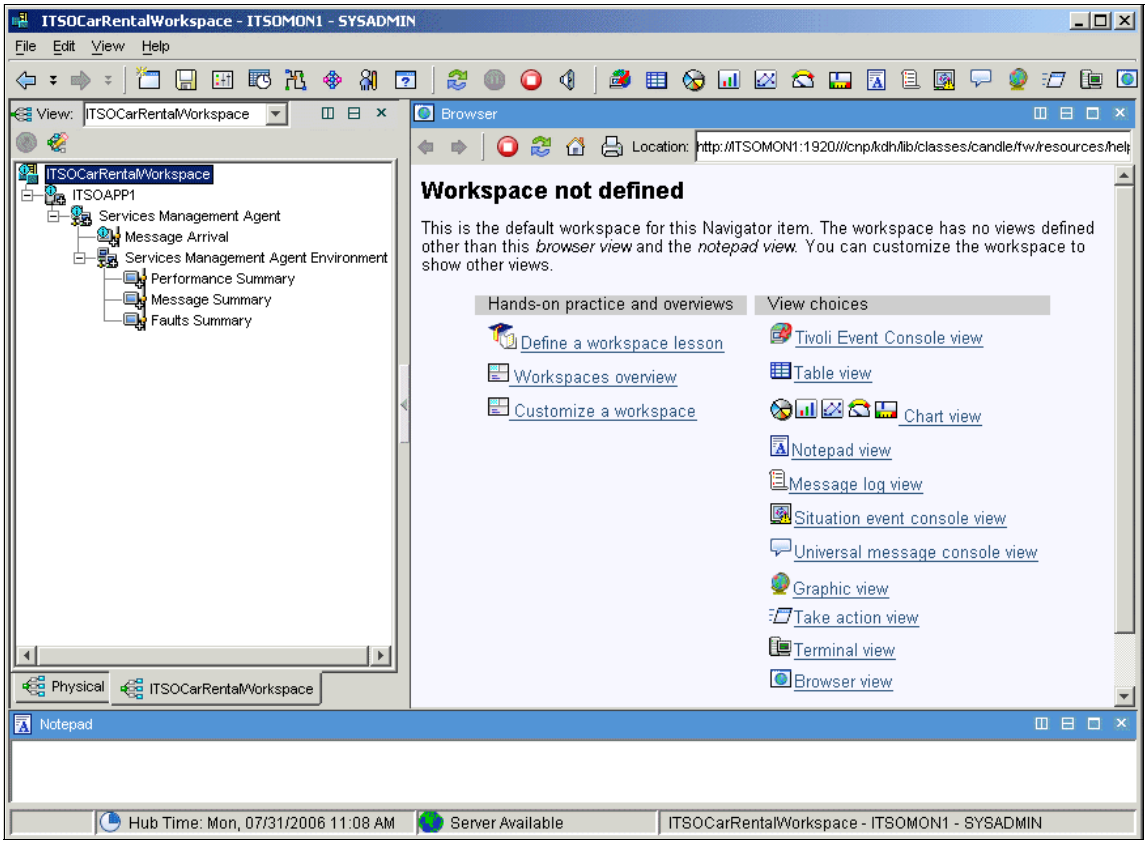


Figure 12-21 Newly created workspace with blank views - drag and drop objects from the toolbar

Note: As necessary, click **File** → **Save Workspace** or **File** → **Save Workspace As** to save changes.

9. Click **Performance Summary**.
10. In the Service Inventory view, there are four icons  for each view. Two of the icons allow you to split the screen either vertically or horizontally. Click **Split Vertically**. Notice that it has created a new table called, `Service Inventory:1`.
11. From the top toolbar, click the **table icon**  and drop it on top of table **Service Inventory:1**. Notice it has created a new table view.
12. Right-click **Properties** from within the table.
13. When the Table Query Editor opens, select the newly created table from the View.
14. Click the **Filters** tab.
15. Ensure only the following columns are selected (deselect others) as seen in Figure 12-22 on page 467:
    - Local Hostname
    - Operation Name
    - Service Name
    - Service Type
    - Average Elapsed Message Round Trip Time

Note: Optionally, click the **Query** tab. Click the icon with (**?**) and label, **Click here to assign a query**. Experiment by choosing different attributes or columns to form a query.

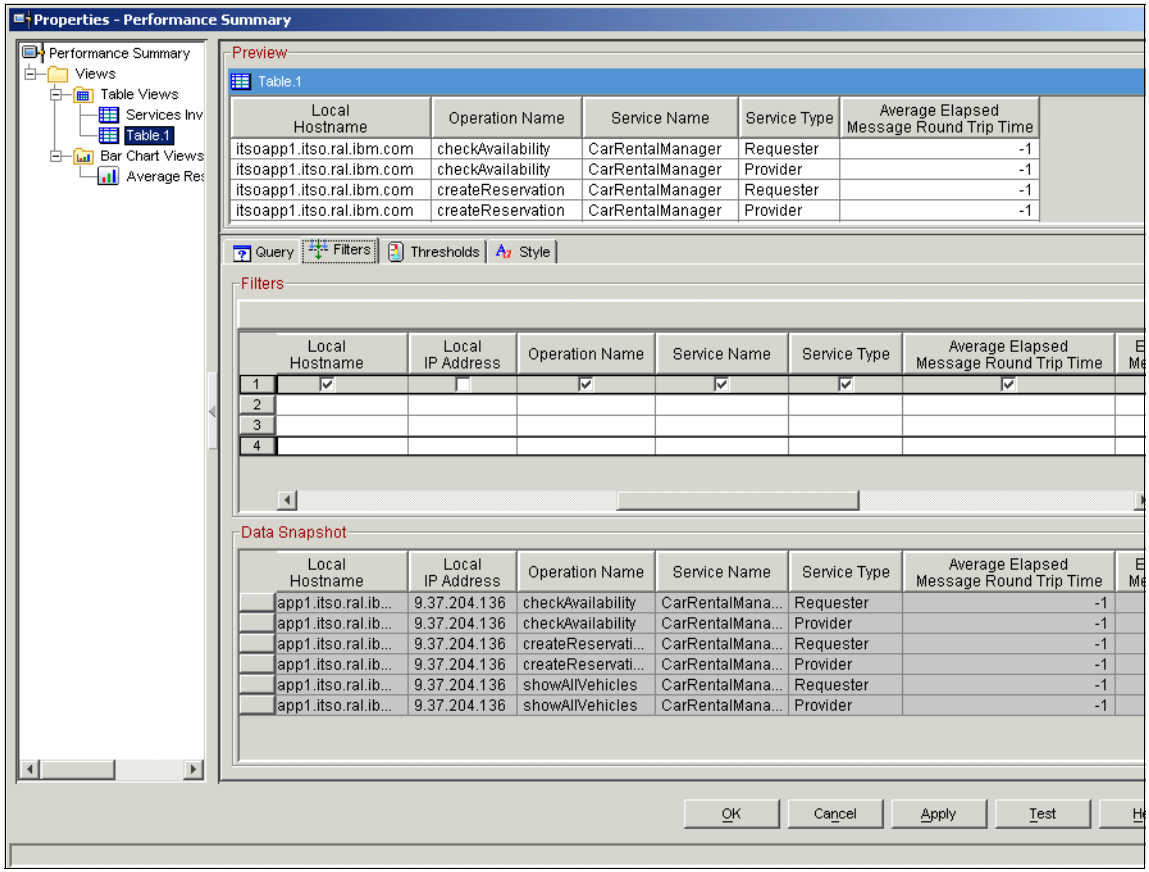


Figure 12-22 Table Query Editor

16. Click **Apply**. Click **Test**. Click **OK**.

> **Note:** By splitting the current view either vertically or horizontally and then dragging and dropping table and graph objects, you can build up a new workspace. For table objects, you can then customized the query attributes pertinent to the application requirements. You can drag other objects and map queries to them. For example, split a view and then drag and drop the **Universal Message Console** from the top toolbar. Notice the system message being logged in the console.

17. Generate some traffic and then view the customized workspace for the specific application. It should look something displayed in Figure 12-23.
18. Select **File** → **Save Workspace**.

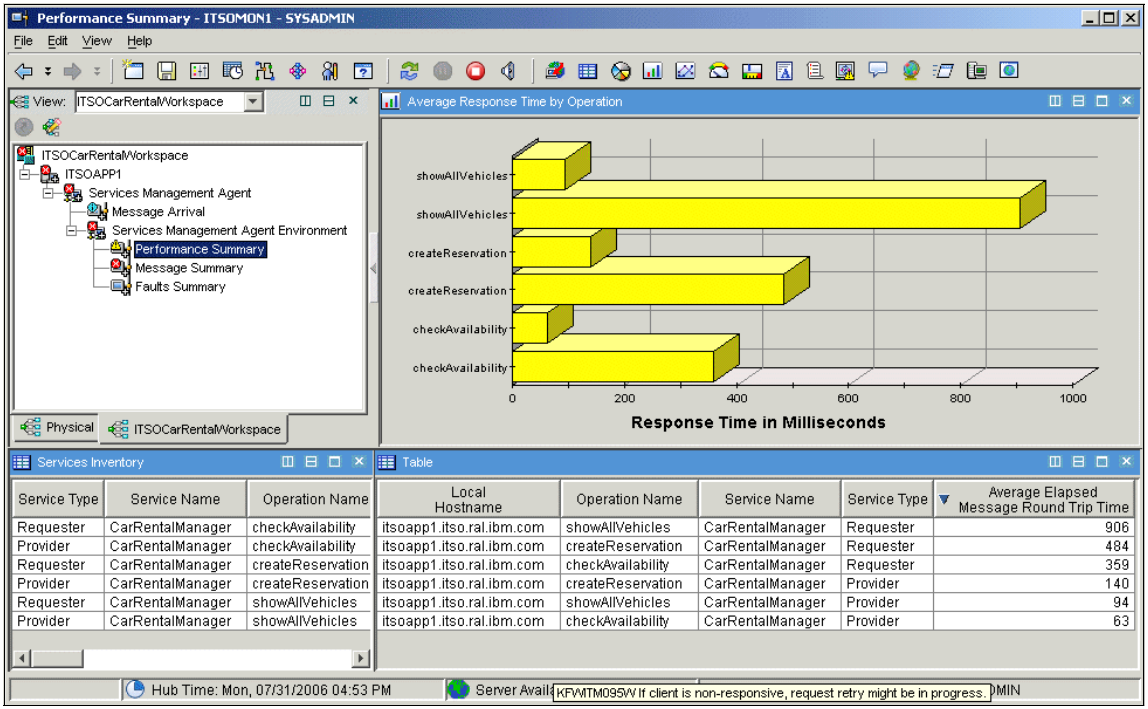


Figure 12-23 Customized TEP Workspace specific to an application

Create a custom workflow

Workflows in TEP provide the ability to automate responses such as perform actions when situations arise, schedule work for users, etc. There are some predefined workflows that you can use as template to create workflows. For example, you may define a filter to reject messages from `Provider.showAllVehicles()`.

To create a customized workflow in TEP, do the following:

1. Select **Performance Summary** in the Navigator view.
2. Select **Edit** → **Workflow Editor (Ctrl+W)**.
3. Select **MessageArrival** under the Policy name column, click the **Edit Workflow** column. This should show the Editor with Message Arrival workflow as seen in Figure 12-24 on page 469.

Note: This predefined workflow is based on the message arrival situation, when too many messages arrive for a particular service, the action is performed to start rejecting messages. When the clearing situation is fired, the filter control is deleted.

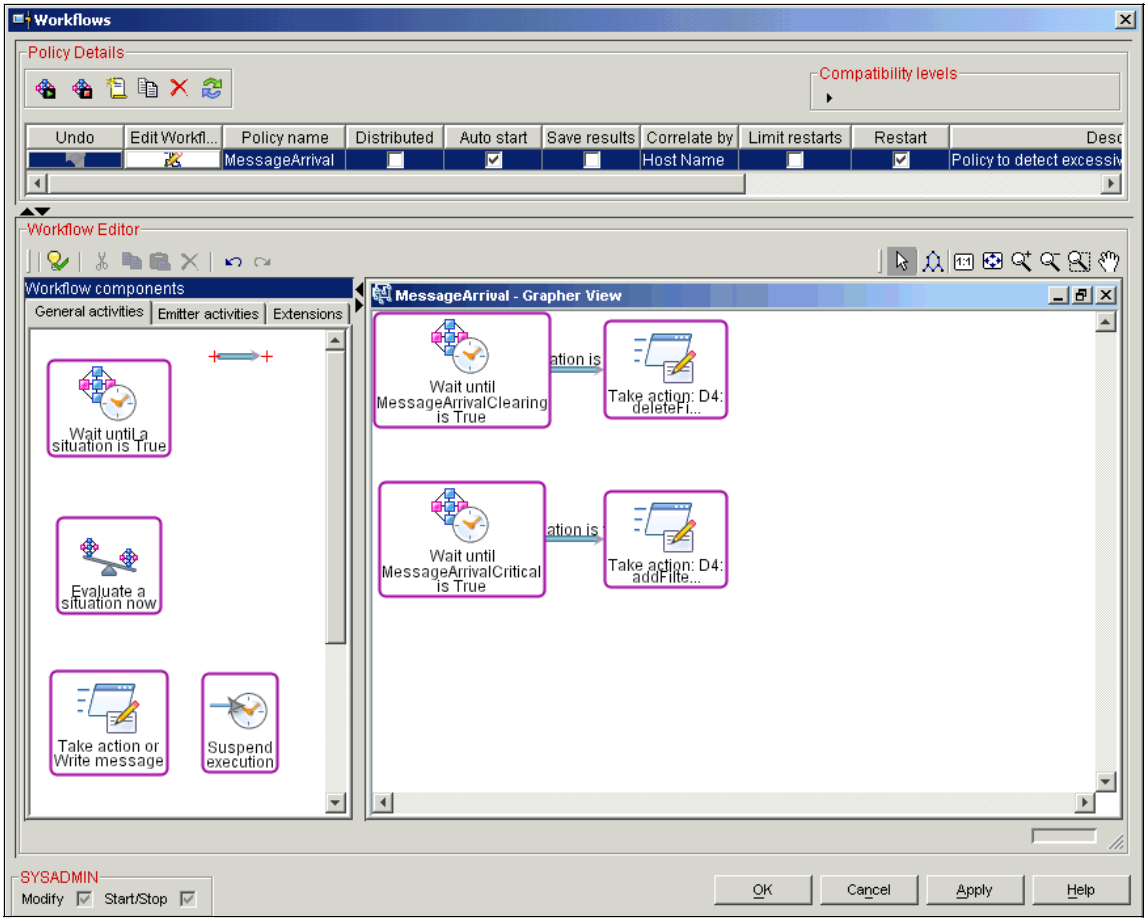


Figure 12-24 Create a user defined workflow with Workflow Editor

4. From the General Activities tab, click **Wait until a Situation is True** then drag and drop it on the right side in **MessageArrival - Graph View**.
5. When the Select a Situation window opens, select **showAllVehiclesCritSit** from the list and click **OK**.
6. From the General Activities tab, click **Take action or Write message** drag and drop it on right side in **MessageArrival - Graph View**.
7. When the Action Settings window opens, complete these tasks:
   a. Select **System Command**.
   b. Enter `net send * ITSOCarRental App Warning: Message threshold reached` in the System Command field.
   c. Click **More Options**.

d. When the Action Settings window opens, select **Execute the action at TEMS**. Click **OK**.

8. From the General Activities tab, select the icon **with blue line**  to connect two activities.
9. Click **Wait until showAllVehiclesCritSit is True** and drag the line to **Take Action: net** to join the two nodes.
10. When the Select a link condition window opens, select **Situation is true** and click **OK**.

    When complete the Workflow Editor should look like Figure 12-25.
11. Click **Apply** and then click **OK**.
12. Rerun the ITSO Car Rental application five or six times to cause the situation to become True.

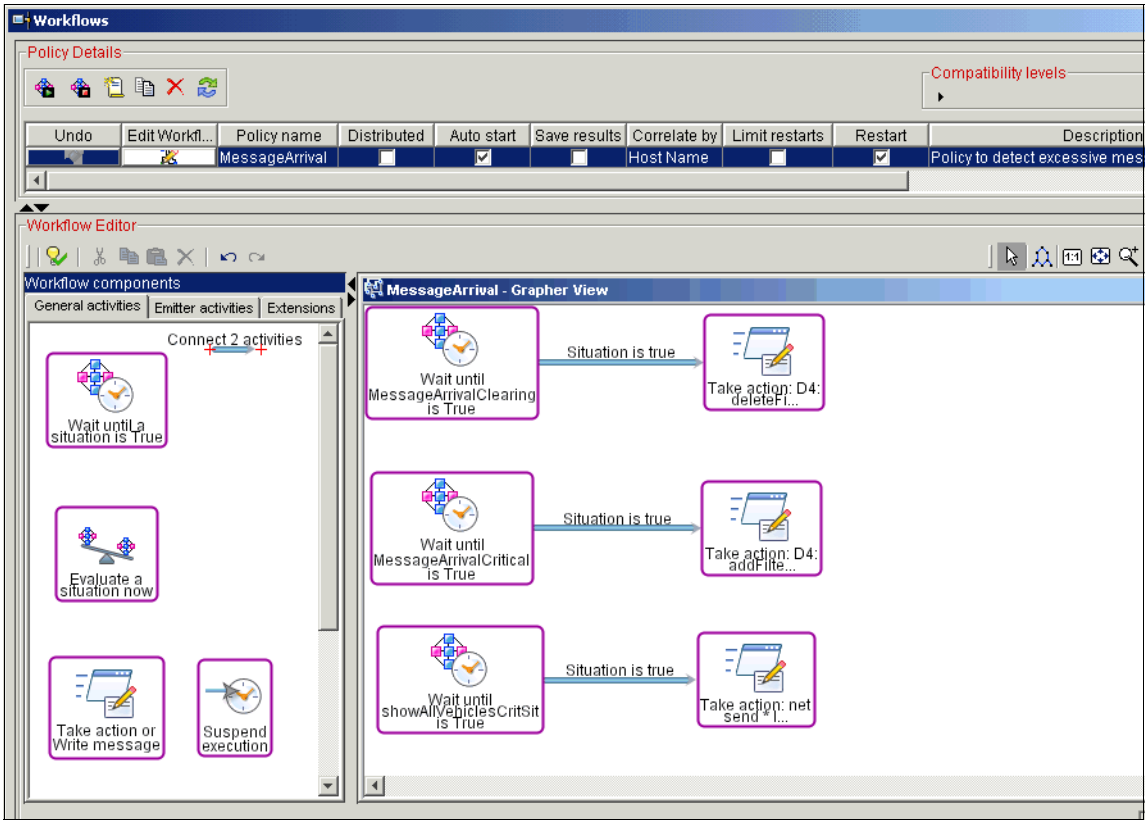


Figure 12-25 Completed customized Workflow

Part 3

Appendixes

A

Role descriptions

This appendix includes a brief description of the following user roles defined for the target audience of this redbook:

- Business Analyst
- Integration Specialist
- Software Architect
- Application Developer
- Enterprise Developer
- Enterprise Architect

Business Analyst

The Business Analyst is responsible for knowing the business processes contained in the business design and capturing them in a model that can be used by the rest of the development team. Ideally, the model is in a form which can be used also to automate those processes in the information system. Generally this is done using modeling tools.

The business analyst participates throughout the development cycle to refine and optimize the business design and establish its key performance indicators. The analyst later uses that information to refine the business design, drive changes in its implementation, and ultimately to verify that the IT realization implements the business design.

Integration Specialist

The Integration Specialist is responsible for integrating existing and new services, and end-users into the business process definition the service composition components. The specialist will typically use visual composition tools and service-bus configuration tools to wire abstract service components that comprise the business processes.

The integration specialist, along with the enterprise architect, is involved in establishing an approach to satisfying the security and quality of service requirements of the enterprise when composing services from business partners and other service providers outside of the enterprise operational environment.

Software Architect

The Software Architect is responsible for translating the business design into a set of software component design specifications for implementing the service definitions and business objects called for in that design. This can go beyond just defining the services in the business, and include designing the internal workings and structure of the actual service components. Building the software architecture will also include making decisions about the appropriateness of legacy function to that design, and determining how to wrapper, extend or refactor an older function to best fit the design.

Application Developer

The Application Developer is responsible for implementing the design for service provided by the software architect. This includes using an appropriate language and technology in which to implement the services, and following the design for those components provided by the software architect.

Enterprise Developer

The Enterprise Developer is a specialist in existing application functions, languages and technologies. The developer is responsible for assisting the software architect in identifying potential reuse of these functions, and in helping determine how best to extend or refactor those functions to enable a better fit with the business design.

Enterprise Architect

Overseeing and conducting all of this is the Enterprise Architect, the person responsible for ensuring consistency across each role's tasks, in a manner that balances the creation of a set of implementation artifacts representing the business design against the constraints, preferences, and existing capabilities of the operational environment.

B

IBM product descriptions

This appendix provides a brief description of the IBM products used within each phase of the SOA life cycle for the SOA scenarios.

The appendix is organized into the following sections:

- IBM products used to model
- IBM products used to assemble
- IBM products used to deploy
- IBM products used to manage

IBM products used to model

This section includes a brief product description as well as a reference to more detailed information for the following IBM products used to model within the SOA life cycle:

- WebSphere Business Modeler
- Rational Software Architect
- Rational Data Architect

WebSphere Business Modeler

IBM WebSphere Business Modeler, known as Modeler, is targeted at Business Analysts to help you capture your business design. You can use Modeler for documentation and compliance purposes: providing a visual and textual representation of your processes, organization, resources, collaborations and business measurements. You can import any static diagrams that you have created previously in Microsoft Visio. Modeler includes a simulation tool with which you can analyze your processes and test how well your processes will hold up under different operating assumptions. You can use this analysis to refine and optimize your business design. Modeler is built on the Eclipse tool framework making it easy for you to share information about your business design with other parts of your organization and tools. In particular, you can export your design into WebSphere Integration Developer and Rational Software Architect so that your application developers can use that as a blueprint for designing process flows for automating your business design.

More information on IBM WebSphere Business Model can be found here:

http://www.ibm.com/software/integration/wbimodeler/

Rational Software Architect

IBM Rational Software Architect is an Eclipse based UML and Java integrated design and development tool that leverages model-driven development for creating well-architected applications and for building services in a service-oriented architecture (SOA). It also includes the J2EE development tooling found in IBM Rational Application Developer.

With Rational Software Architect you can unify all aspects of software design and development.

- Develop applications and Web services more productively than ever
- Exploit the latest in modeling language technology
- Review and control the structure of Java and service-oriented applications
- Leverage an open and extensible modeling platform
- Simplify your design and development tool solution
- Integrate with other facets of the life cycle

Rational Software Architect supports UML 2.0 for analysis and design using Use Case, Class, Sequence, Activity, Composite Structure, State Machine, Communication Component and Deployment diagrams. It supports UML class diagram editing for Java, Enterprise Java Beans and Database objects. As well as supporting full features of UML, it provides many additional capabilities.

Rational Software Architect is useful for turning the output of WebSphere Modeler into service specifications and component-detailed designs that you can then hand over to software developers for implementation.

Rational Software Architect has support for plugging in design patterns that help automate development and promote re-use. It comes already populated with the classic patterns described in *Design Patterns: Elements of Reusable Object-Oriented Software* by Erich Gamma, et al [GAMMA]. You can obtain other predefined patterns or create your own to capture specific design practices that you use commonly throughout your software.

More information on IBM Rational Software Architect can be found here:

http://www.ibm.com/software/awdtools/architect/swarchitect/features/index.html

Rational Data Architect

IBM Rational Data Architect helps data architects model, visualize, relate, and develop data assets to understand data assets and their relationships to each other, accelerate new or composite application development, reuse prior design and development investments, and assess and manage schema changes. Built with integration challenges in mind, it combines traditional data modeling capabilities with metadata discovery, mapping, and analysis, and creates integrated and abstracted views and deploys them directly onto WebSphere Information Integrator accelerating value delivery. It plugs into the Eclipse modeling framework, and provides full life cycle design, development and debugging for database schema, SQL, and Java-based procedures, triggers, and functions.

More information on IBM Rational Data Architect can be found on this Web site:

http://www.ibm.com/software/data/integration/rda/

IBM products used to assemble

This section includes a brief product description as well as a reference to more detailed information for the following IBM products used to assemble within the SOA life cycle:

- Rational Application Developer
- CICS Web Services Assistant

- WebSphere Developer for zSeries
- Bowstreet Portlet Factory
- WebSphere Integration Developer

Rational Application Developer

IBM Rational Application Developer is an Eclipse based integrated development environment for constructing, testing, assembling, deploying and debugging applications that are built with Java/J2EE, Web Services and many other open standards.

Rational Application Developer is optimized for IBM WebSphere software, but also supports non-IBM run-time environments, such BEA WebLogic and JBOSS.

IBM Rational Application Developer for WebSphere Software is powered by the Eclipse open source platform so developers can adapt and extend their development environment to match their needs and increase their productivity. When used with the IBM Software Development Platform, developers can access a broad range of requirements and change-management functions directly from Rational Application Developer for WebSphere Software. Some key features and benefits are:

- Accelerate portal, SOA and J2EE using RAD tools and wizards.
- Leverage existing skills and shorten the Java learning curve with drag-and-drop UI components and point-and-click database connectivity.
- Improve code quality with automated tools for applying coding standard reviews, component and Web Service unit testing and multi-tier runtime analysis.
- Integrate your business applications with interoperable Web services and service-oriented architectures.
- Visualize and graphically edit code through the UML Visual Editor for Java and EJB.
- Collaborate and share assets across the team using built-in IBM Rational ClearCase LT version control.
- Adapt and extend your development environment with Eclipse-based plug-ins to match your needs.
- Quickly build and deploy interactive reports using drag-and-drop UI components and Crystal Reports.

More information on IBM Rational Application Developer can be found here:

`http://www.ibm.com/software/awdtools/developer/application/index.html`

CICS Web Services Assistant

The CICS Web Services Assistant, included with IBM CICS Transaction Server V3.1, is a set of batch utilities used to create Web Services for CICS applications, as well as enable CICS applications to invoke Web Services.

The CICS Web services assistant provides two utility programs:

- DFHLS2WS: This utility is used to generates a Web Service binding file from a language structure. It can also be used to generates a Web Service description.
- DFHWS2LS: This utility is used to generate a Web Service binding file from a Web service description. It can also generate a language structure that you can use in your application programs.

The assistant supports rapid deployment of CICS applications for use by service providers and service requesters, with the minimum of programming effort. When using the CICS Web Services Assistant, you do not have to write your own code for parsing inbound messages and for constructing outbound messages; CICS maps data between the body of a SOAP message and the application program's data structure.

In most cases, CICS can generate and install the resource definitions automatically. You do have to define PIPELINE resources, but in many cases you can use one of the pipeline configuration files provided with CICS (basicsoap11provider.xml, basicsoap11requester.xml).

The assistant can create a WSDL document from a simple language structure, or a language structure from an existing WSDL document, and supports COBOL, C/C++, and PL/I. However, the assistant cannot deal with every possibility, and there will be times when you will need to take a different approach.

If you decide not to use the CICS Web Services Assistant, you will have to:

- Provide your own code or parsing inbound messages and constructing outbound messages (unless you use WebSphere Developer).
- Provide your own pipeline configuration file.
- Define and install your own URIMAP and PIPELINE resources.

More information on IBM CICS Transaction Server V3.1, Web Services Assistant can be found at this Web site:

http://publib.boulder.ibm.com/infocenter/cicsts/v3r1/index.jsp

WebSphere Developer for zSeries

WebSphere Developer for zSeries V6.0.1, is built upon the Rational Software Developer Platform (Eclipse based), facilitates the development of both Java and

z/OS-based applications. WebSphere Developer contains tools that support the development of Web services and the XML enablement of existing CICS COBOL applications.

The XML Services for the Enterprise (XSE) capability of WebSphere Developer provides tools that let you adapt COBOL-based applications so that they can consume and produce XML messages. XSE supports the creation of driver programs that work with existing CICS (or IMS) applications.

The Web Services Enablement wizard is the XSE tool that supports the bottom-up approach for creating Web services based on existing CICS COBOL programs. It takes as input the COMMAREA copybook. The XML structure and data types are then derived from the COBOL data declarations. Based on these, the Web Services Enablement wizard generates the set of artifacts.

Tip: Building the ultimate CICS development and unit test environment

The combination of WebSphere Developer for zSeries V6.0.1 and the bundled CICS Transaction Server for WebSphere Developer test environment, and the CICS Transaction Gateway V6.0.2, provides an incredible powerful environment to develop by direct exposure and indirect exposure Web Services for CICS.

We recommend that you refer to the following article on IBM developerWorks for information on building this CICS development and test environment:

http://www.ibm.com/developerworks/websphere/library/techarticles/0509_barosa/0509_barosa.html

IBM WebSphere Developer for zSeries V6.0.1 prerequisites the following:

- IBM Rational Application Developer or Rational Software Architect V6.0.1.1 + EGL feature must be pre-installed
- WebSphere Developer for zSeries CD images (disks) 1-4 are required, and CD 5 includes the CICS Transaction Server for WebSphere Developer test environment

The CICS Transaction Gateway V6.0.1 (or V6.0.2) is a separate CD image.

More information on IBM WebSphere Developer for zSeries V6 can be found here:

http://www.ibm.com/software/awdtools/devzseries/

Bowstreet Portlet Factory

Now acquired by IBM, Bowstreet Portlet Factory for WebSphere is a development technology that accelerates the development, customization, deployment and maintenance of portlets for WebSphere Portal. With Bowstreet Portlet Factory, Bowstreet for short, developers can quickly and easily repurpose a company's core assets, automatically assembling them into custom, high-value portlets. These portlets can then be deployed into WebSphere Portal.

The Bowstreet solution is built on open standards and supports Java, J2EE, XML and JSR 168. Bowstreet speeds up portlet development through dynamic generation of code which can be quickly modified by business users. It can rapidly generate portlets for legacy systems such as Domino®, Siebel, SAP, PeopleSoft, Hyperion and many others. The portlets created can automatically present themselves as standalone Web applications, Web services or portlets based on JSR 168 specification without requiring any coding of assets.

WebSphere Integration Developer

IBM WebSphere Integration Developer (WID) is also an Eclipse based tool designed to help create business process flows, state machines and business rules.

WebSphere Integration Developer V6.0 simplifies integration with its Service Component Architecture (SCA). SCA uses Business Process Execution Language for assembling business process tasks into workflows, which can then be deployed to IBM WebSphere Process Server.

WebSphere Integration Developer can directly import business models from the IBM Business Modeler. You can then use a wiring editor for assembling service components, for importing service interface definitions and for setting binding policies to build SOA enabled applications.

More information on IBM WebSphere Integration Developer can be found here:

http://www.ibm.com/software/integration/wid/

IBM products used to deploy

This section includes a brief product description as well as a reference to more detailed information for the following IBM products used to deploy within the SOA life cycle:

- WebSphere Application Server Network Deployment
- WebSphere Enterprise Service Bus
- WebSphere Message Broker
- WebSphere Portal

- WebSphere Process Server
- WebSphere Information Integration Server

WebSphere Application Server Network Deployment

IBM WebSphere Application Server Network Deployment serves two roles within the SOA Foundation. It is the hosting environment for basic SOA business services – primarily those implemented with J2EE Enterprise JavaBeans™ (EJBs). These services can be exposed with WSDL and integrated through standard Web service protocols and encodings, or can be integrated in a more tightly coupled fashion through Remote Method Invocation/InterORB Protocol (RMI/IIOP) bindings.

WebSphere Application Server also serves as the underlying execution platform for WebSphere Portal, WebSphere Process Server, WebSphere Enterprise Service Bus, and a variety of other offerings within the IBM portfolio. This foundational role enables these products to be tightly integrated (albeit hosting a set of loosely-coupled service artifacts) with a common approach to installation, clustering, scaling, administration, service and security.

WebSphere Application Server is offered in a variety of flavors, including a very compact and simple platform for J2EE applications based on Apache Geronimo, the WebSphere Community Edition (CE); WebSphere Application Server base edition is a standalone, single server application server that provides full run-time support for J2EE v1.4 and Web services.

A customized variant packaged with tooling and targeted specifically at the ISV community that serves the small and medium business marketplace, the WebSphere Application Server Express Edition and a scalable version that supports clustering, dynamic fail-over, and a centralized administration model, the WebSphere Application Server Network Deployment Edition (ND).

WebSphere Extended Deployment (XD) builds on the capabilities of WebSphere ND and offers a dynamic goals-directed, high performance environment for running mixed application types and workload patterns in WebSphere.

More detailed information on IBM WebSphere Application Server family can be found at this Web site:

`http://www.ibm.com/software/webservers/appserv/was/`

More detailed information on IBM WebSphere Application Server Network Deployment Edition can be found at this Web site:

`http://www.ibm.com/software/webservers/appserv/was/network/`

WebSphere Enterprise Service Bus

IBM WebSphere Enterprise Service Bus provides Web services connectivity, JMS messaging and service-oriented integration for SOA. It instantiates the enterprise service bus architectural pattern, providing for a basic fabric for transparent interconnection of services across an enterprise distributed network.

The WebSphere Enterprise Service Bus includes support for service bus mediations for reconciling different types of connectivity, including support for transformation of service requests, content based routing and constructing side-logging for auditing or traceability purposes.

WebSphere Enterprise Service Bus is optimized to operate on service requests that have been bound using SOAP encodings and context processing semantics. It supports a variety of binding transports, including HTTP, the default embedded JMS message provider in WebSphere Application Server, and WebSphere MQ.

More information on IBM WebSphere Enterprise Service Bus can be found here:

http://www.ibm.com/software/integration/wsesb/

WebSphere Message Broker

IBM WebSphere Message Broker delivers an advanced Enterprise Service Bus providing connectivity and universal data transformation for both standard and non-standards-based applications and services.

WebSphere Message Broker enhances the WebSphere Enterprise Service Busby by providing support for message transformation support for non-XML message types such AL3, HL7, Swift, HIPAA and EDI. WebSphere Message Broker provides optional support for DataStage TX for complex message transformations. It also has built-in support to help you implement complex event processing without programming.

WebSphere Message Broker has its roots in message-based integration, a form of business composition and hub-centered mediation that predated the latest standards for SOA. While it is reasonable to think of WebSphere Message Broker as extending the capabilities of WebSphere Enterprise Service Bus, most often it will be appropriate to use them together in a bridged interconnection, leveraging the abilities of WebSphere Message Broker to provide interconnection with non-XML based services, native interconnectivity with CICS and IMS, and for those cases where you need complex event processing and transformation, and then using WebSphere Enterprise Service Bus to optimize your interconnectivity between XML and standard SOA-based services.

More information on IBM WebSphere Message Broker can be found here:

http://www.ibm.com/software/integration/wbimessagebroker/

WebSphere Portal

IBM WebSphere Portal delivers a single point of personalized interactions with applications, content, processes and people.

WebSphere Portal is a hosting environment for the user interaction logic of your SOA application. More than that, the Portal server provides a platform on which multiple service user interfaces can be aggregated on a single user page within a layout template defined by the business. This provides for a unique combination of structured layout and freedom for end users to customize the content they find most important to getting their job done.

In addition, portlets rendered within the page layout can be configured with published properties that when managed through the WebSphere Portal's built-in property broker can be used to create visual integration between the services rendered by each of those Portlets. In effect, WebSphere Portal offers the enterprise user the ability to form a customized visual composition of business services.

WebSphere Portal is built on the J2EE and Web Services foundations provided by WebSphere Application Server. WebSphere Portal enables rapid assembly of application artifacts via the portlets.

More information on IBM WebSphere Portal can be found here:

http://www.ibm.com/software/genservers/portal/

WebSphere Process Server

IBM WebSphere Process Server is the primary hosting environment for business processing. Built on WebSphere Enterprise Service Bus, WebSphere Process Server includes support for both BPEL based process flows as well as business state machines. WebSphere Process Server also supports the integration of business rules in process and service selection. The Process Server is the first product within the IBM suite to offer direct support for the Service Component Architecture SOA programming model.

WebSphere Process Server integrates with WebSphere Portal to deliver business process management through a portal – there is support for human tasks within a business process. *Human tasks* are defined as activities within the process definition that will be carried out by human end-users. This includes built-in support for task assignment, pick-lists, scheduling and escalation policies in case a task is not processed in a timely fashion.

More information on IBM WebSphere Process Server can be found here:

http://www.ibm.com/software/integration/wps/

WebSphere Information Integration Server

IBM WebSphere Information Integration server is design specifically to enable information integration services as defined by the SOA Foundation. It provides composition of data from a variety of underlying data sources, enabling you to compose views on data that match the information needs of our composed business services. WebSphere Information Integration server can be used to build views, set import and caching strategies, and virtualize the schema of multiple, heterogeneous data systems into a homogenized relational type system. For example, you can combine data from IMS DL/I, VSAM, DB2 and Oracle databases to build a view on that joined data that appears to your application as though it is all coming from a single relational database.

More detailed information about WebSphere Information Server can be found at this Web site:

http://www-306.ibm.com/software/data/integration/db2ii/

IBM products used to manage

This section includes a brief product description as well as a reference to more detailed information for the following IBM products used to manage within the SOA life cycle:

- WebSphere Business Monitor
- Tivoli Composite Application Manager for SOA
- Tivoli OMEGAMON XE for Messaging
- Tivoli Access Manager
- Tivoli Federated Identity Manager

WebSphere Business Monitor

IBM WebSphere Business Monitor enables you to monitor business processes in real-time, providing a visual display of business process status.

WebSphere Business Monitor the compliment to WebSphere Business Modeler. It helps you create dashboards for visualizing the performance of your business based on the key performance indicators that you identified in your business design. You can use this to track time, cost and resources used within your processes. WebSphere Business Monitor provides tools to let you set situational triggers and notifications to bring your attention to potential bottlenecks or workload imbalances in your business.

Ultimately WebSphere Business Monitor helps you better understand how your business design achieves your business objectives, and provides guidance on

how to refine and optimize that business design in the case your goals are not being met.

More information on IBM WebSphere Business Monitor can be found here:

http://www.ibm.com/software/integration/wbimonitor/

Tivoli Composite Application Manager for SOA

IBM Tivoli Composite Application Manager (ITCAM) offering is designed specifically to enable IT service management. It has been designed to understand the unique semantics and loosely coupled characteristics of service-oriented architecture (SOA) based services. ITCAM has three editions that are relevant directly to the SOA Foundation: ITCAM for WebSphere, ITCAM for SOA, and ITCAM for Response Time Tracking. These cover the gamut from application server monitoring and resource consumption, deep-dive diagnostics and correlation as service invocations cascade across multiple systems, and service level response times and problem isolation.

Tivoli Composite Application Manager for SOA utilizes the foundation services of Tivoli Enterprise Monitoring Services.

More information on IBM Tivoli Composite Application Manager can be found at:

http://www.ibm.com/software/tivoli/solutions/application-management/

Tivoli OMEGAMON XE for Messaging

IBM Tivoli OMGEGAMON XE for Messaging is part of the Tivoli monitoring products suite.

Tivoli OMEGAMON product came as part of the Candle acquisition by IBM. It helps you to:

- Improve WebSphere MQ, Message Broker, MQ Integrator and InterChange Server Availability
  - Identify common problems and automate corrective actions.
  - Auto-discovery and immediate monitoring of complex environments
  - Drill-down to locate a problem, identify the root cause and resolve bottlenecks or outages
- Proactively Prevent Problems
  - Correctly configure and deploy your WebSphere MQ infrastructure
  - Detect and repair problems as they happen, or alert you to an imminent concern
  - Provides key MQ and Message Broker metrics for real-time and historical data analysis

- Simplify Management with a Single Tool
  - Manages WebSphere MQ, Message Broker, MQ Integrator and InterChange Server in distributed and mainframe environments
  - User-customized displays including business, platform and resource views

More information on IBM Tivoli OMEGAMON XE can be found at this Web site:

http://www.ibm.com/software/tivoli/products/omegamon-xe-websphere-bus-int/

Tivoli Access Manager

Tivoli Access Manager for e-business is a versatile solution for authentication and authorization problems. Primarily focused on Web applications, Access Manager implementations vary from simple Single Sign-on (SSO) to more complex security infrastructure deployments.

Tivoli Access Manager for e-business can help you manage growth and complexity, control escalating management costs, and address the difficulties of implementing security policies across a wide range of Web and application resources. It works by centrally managing security and audit policy for enforcement points that can be placed as a proxy in front of Web applications, or through authorization and authentication plug-ins direct into a Web server or application-server environment. You can use Tivoli Access Manager to control wired and wireless access to applications and data to help bar unauthorized users. For authorized users, Tivoli Access Manager integrates with Web applications and servers to deliver a secured and unified business experience. It helps you secure access to business-critical applications and data spread across the extended enterprise, allowing highly available, scalable transactions with partners, customers, suppliers, and employees.

More information on IBM Tivoli Access Manager can be found at this Web site:

http://www.ibm.com/software/tivoli/products/access-mgr-e-bus/

Tivoli Federated Identity Manager

IBM Tivoli Federated Identity Manager provides a simple, loosely-coupled model for managing identity and access to resources that span companies or security domains. Rather than replicate identity and security administration at both companies, IBM Tivoli Federated Identity Manager provides a simple model for managing identities and providing them with access to information and services in a trusted fashion. For companies deploying service-oriented architecture (SOA) and Web Services, Tivoli Federated Identity Manager provides policy-based integrated security management for federated Web services. The foundation of Federated Identity Management is trust, integrity, and privacy of data.

Tivoli Federated Identity Manager implements a number of industry standard mechanisms, including SAML, the Liberty Alliance, and the Web Service Federation Language (WS-Federation) specification for federating user information and authentication schemes that may be used across different parts of your business or between you and your business partners.

Tivoli Federated Identity Manager operates on the recognition that no one entity in the world will have sole authority over the authenticity of users in the internet, or even the relationships between multiple business partners. Instead, it enables you to recognize the authority that each partner has in establishing the authenticity of the users that exist at each partner, and then coordinating how that authenticity may be used to control access to your own services and resources.

This can dramatically reduce operational expenses by eliminating duplicate registry entries for all your users at each location that hosts services you want to use. It simplifies your user's job by enabling a single sign-on to all the systems your user needs. The single sign-on also increases the overall integrity of your system by reducing the number passwords that your users must remember.

More information on Tivoli Federated Identify Manager can be found here:

http://www.ibm.com/software/tivoli/products/federated-identity-mgr/

ITCAM family products

This section includes complimentary product information for the ITCAM.

More information on the IBM Tivoli Composite Application Manager family of products can be found with the following resources:

- IBM Tivoli Composite Application Manager products page:

  http://www.ibm.com/software/tivoli/sw-atoz/indexC.html

- IBM Redbook *IBM Tivoli Composite Application Manager V6.0 Family: Installation, Configuration, and Basic Usage*, SG24-7151

ITCAM for Response Time Tracking V6.0

IBM Tivoli Composite Application Manager for Response Time Tracking V6.0 (ITCAM for RTT) can proactively recognize, isolate, and resolve transaction performance problems using robotic and real-time techniques. It is an end-to-end transaction management solution that monitors end-user response time and helps you visualize a transaction's path through your application systems, including response time contributions of each step. You can drill down to each step that the transaction takes as it travels across multiple systems, and measure how each component of a transaction contributes to the overall

response time. The entire transaction analysis process is transparent to customers and application developers.

ITCAM for SOA V6.0

IBM Tivoli Composite Application Manager for SOA (ITCAM for SOA) can monitor, manage, and control the Web services layer of IT architectures while drilling down to the application or resource layer to identify the source of bottlenecks or failures, and pinpoint services that take the most time or use the most resources.

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

ITCAM for WebSphere V6.0

IBM Tivoli Composite Application Manager for WebSphere (ITCAM for WebSphere) can help you quickly pinpoint, in real time, the source of bottlenecks in application code or server resources. Real-time information includes throughput, response time, and details for a specific transaction. ITCAM for WebSphere also provides predefined reports that you can use to display historical data, so you can plan for future growth.

ITCAM for CICS Transactions V6.0

ITCAM for CICS Transactions is a data collector installed on CICS systems to provide data for ITCAM for Response Time Tracking or ITCAM for WebSphere. ITCAM for CICS Transactions does not provide in-depth application analysis or CICS system analysis; this requires Tivoli OMEGAMON XE for CICS on z/OS. ITCAM for CICS Transactions provides the following information:

- CICS call summary and response time
- CICS transaction response time
- CICS processor and memory usage

ITCAM for CICS Transactions is not a standalone product. It needs either ITCAM for Response Time Tracking, ITCAM for WebSphere, or both.

ITCAM for IMS Transactions V6.0

ITCAM for IMS Transactions is a data collector installed on IMS systems to provide data for ITCAM for Response Time Tracking or ITCAM for WebSphere. ITCAM for IMS Transactions does not provide in-depth application analysis or IMS system analysis; this requires IBM Tivoli OMEGAMON XE for IMS on z/OS.

ITCAM for IMS Transactions provides the following information:

- IMS transaction response time
- IMS processor and memory usage

ITCAM for IMS Transactions is not a standalone product. It needs ITCAM for Response Time Tracking, ITCAM for WebSphere, or both.

OMEGAMON XE for WebSphere Business Integration V1.1

IBM Tivoli OMEGAMON XE for WebSphere Business Integration helps improve the availability and performance of business-critical applications and business integration systems. It can identify common problems and automate corrective actions using predefined industry best-practice situations, while monitoring key WebSphere MQ and WebSphere Business Integration Message Broker metrics.

- It improves WebSphere MQ, Message Broker, MQ Integrator and InterChange Server Availability:
  - Identify common problems and automates corrective actions
  - Auto-discovery and immediate monitoring of complex environments
  - Drill-down to locate a problem, identify the root cause, and resolve bottlenecks or outages
- Proactively Prevent Problems
  - Correctly configure and deploy your WebSphere MQ infrastructure
  - Detect and repair problems as they happen, or alert you to an imminent concern
  - Provides key WebSphere MQ and Message Broker metrics for real-time and historical data analysis
- Simplify Management with a Single Tool
  - Manages WebSphere MQ, Message Broker, MQ Integrator and InterChange Server in distributed and mainframe environments
  - User-customized displays including business, platform and resource views

When used collectively, the ITCAM products provide end-to-end transactional view of monitored systems with detail analysis of resource utilization in a heterogeneous environment.

C

Troubleshooting tips

While configuring the run-time environment for the working example Service Creation solution in this book, we encountered some configuration issues with the Web Services Gateway included with IBM WebSphere Application Server Network Deployment V6.0, and IBM Tivoli Composite Application Manager for SOA. We have included a description of these problems and possible resolutions. In addition, we have included information on enabling tracing and testing.

This appendix is organized into the following sections:

- Troubleshooting the WebSphere Web Services Gateway
- Troubleshooting ITCAM for SOA

Troubleshooting the WebSphere Web Services Gateway

This section includes tips for troubleshooting the Web Services Gateway configuration. In some cases, the information is specific to the redbook scenario, but much of the information will apply to any Web Services Gateway configuration.

Enable tracing

Do the following to enable tracing

Change log / trace file size and number of historical log files

The default log file size is 1 MB with history of 1 log file. We enabling tracing, the logs will quickly be filled detailed component information and debug statements. To ensure the logs do not wrap (contents overwritten), we recommend that you increase the log file size and number of historical log files.

In our example run-time environment, we increased the log file size and number of historical log files for both server1 and dmgr.

1. From the Administrative Console, expand **Troubleshooting** and click **Logs and Trace**.
2. Click the desired server (for example, server1 and dmgr).
3. To set SystemOut.log and SystemErr.log, follow these steps:
   a. Click **JVM™ logs**.
   b. Modify the settings as seen in Figure C-1 on page 493 (top of page) for SystemOut.log (System.err displayed on page).

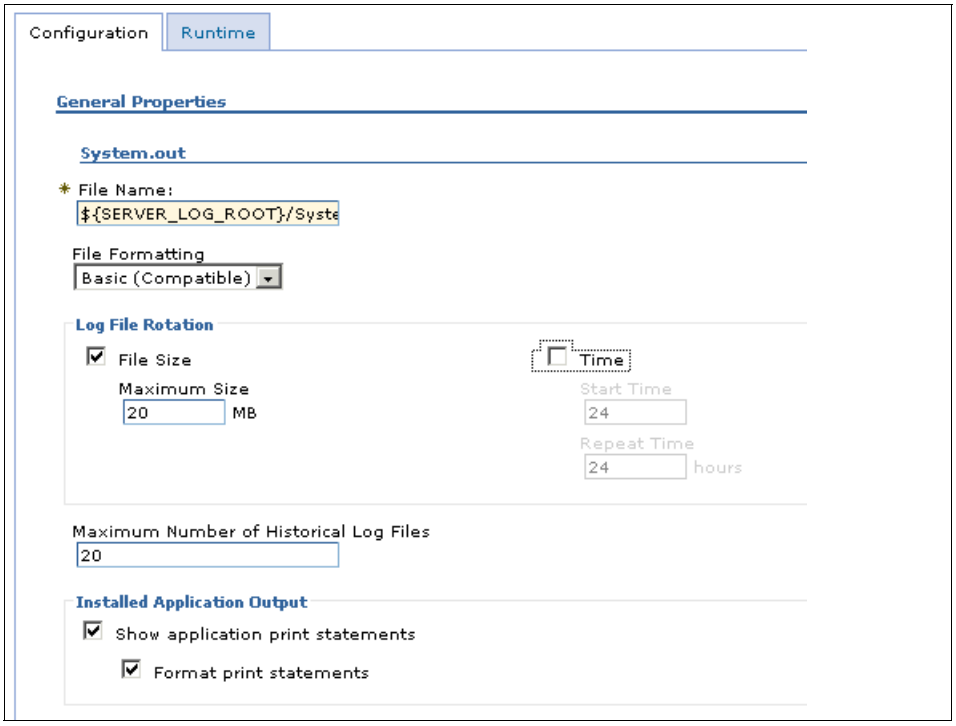


Figure C-1 Modified settings for SystemOut.log to increase log file size and number of historical log files

c. Modify the settings as seen in Figure C-2 on page 494 (bottom of page) for SystemErr.log (System.err displayed on page).

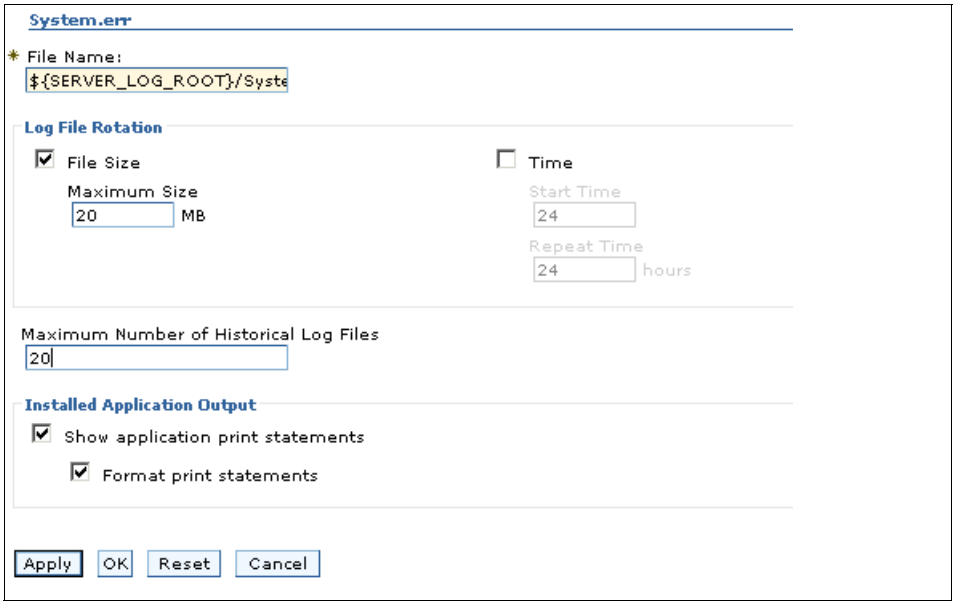


Figure C-2 Modified settings for SystemErr.log to increase log file size and number of historical log files

d. When you are finished, click **Apply** then click **OK**.

e. Click **Save**. Click **Save** again to save to master configuration.

> **Note:** The settings for SystemOut.log and SystemErr.log will not take effect until the application server is restarted.

4. To set trace.log, follow these steps:

   a. Click **Diagnostic Trace**.

   b. Modify the settings as seen in Figure C-3 on page 495.

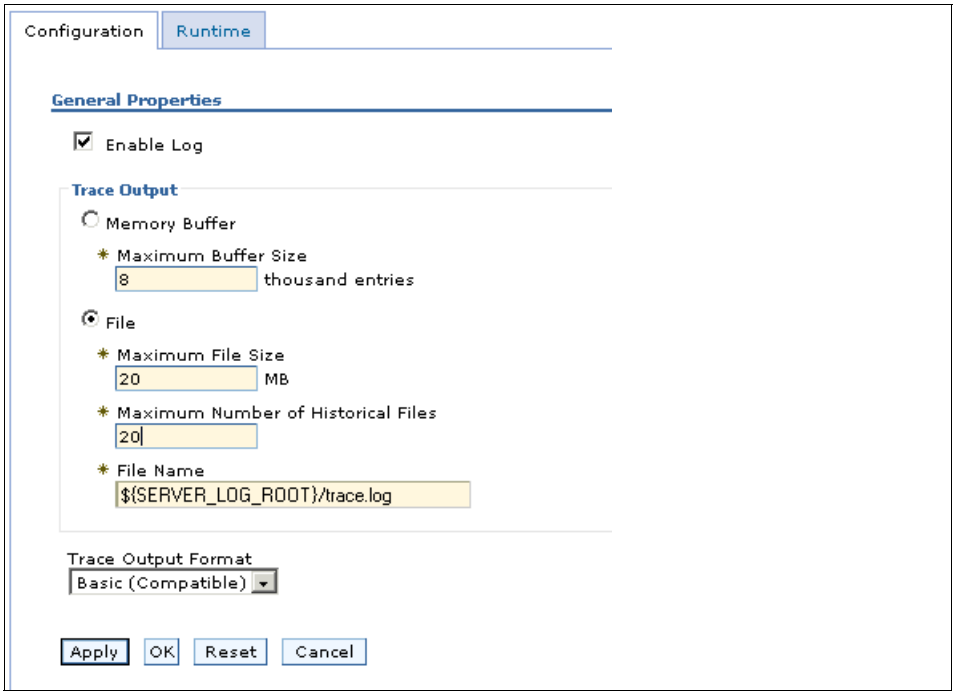


Figure C-3 Modified settings for trace.log to increase log file size and number of historical log files

c. When you are finished click **Apply**, click **OK**.

d. Click **Save**. Click **Save** again to save to master configuration.

Note: The settings for trace.log will not take effect until the desired application server is restarted if done in the Configuration tab.

To avoid the need to restart the server, you can enable these settings using the Runtime tab as seen in Figure C-4 on page 496. The settings entered in the Runtime tab, can be saved to the Configuration tab by checking the **Save runtime changes to configuration as well checkbox**.

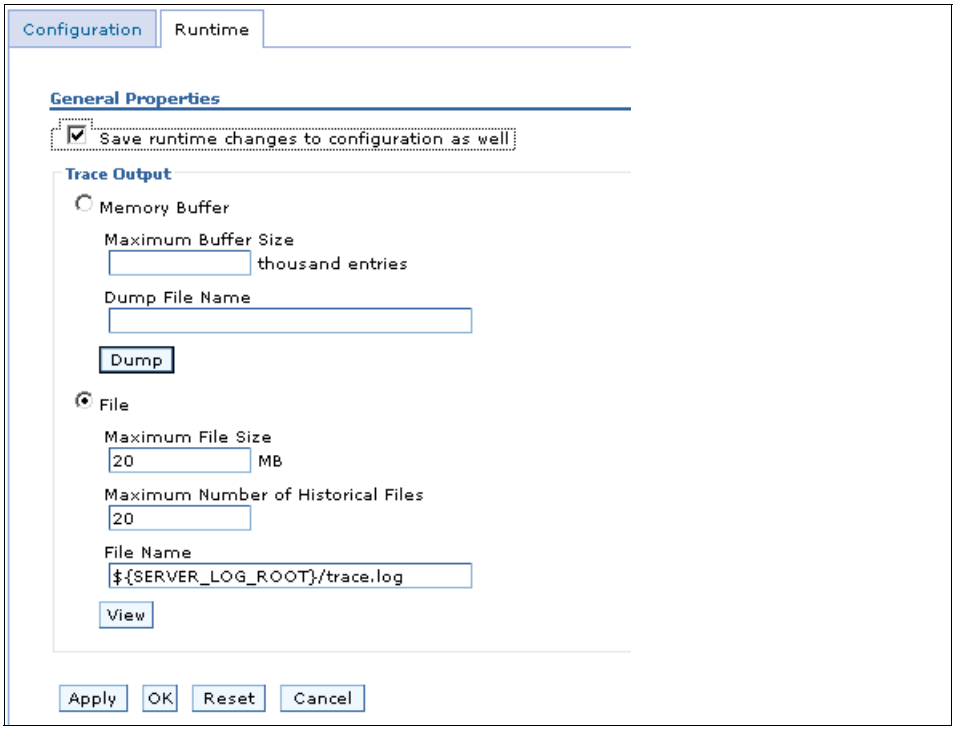


Figure C-4 Enable trace.log file settings using Runtime tab feature

Enable trace

Enabling trace information is a useful method of getting more detailed information about your application interaction with WebSphere components. This technique is used for advanced troubleshooting by IT specialists and the IBM support team.

1. From the Administrative Console, expand **Troubleshooting** and click **Logs and Trace**.
2. Click the desired server (for example, server1 and dmgr).
3. Click **Change Log Detail Levels**.
4. Enter the following trace strings for Web Services and gateway in the text box as seen in Figure C-5 on page 497:

```
com.ibm.ws.sib.webservices.*=all
com.ibm.ws.webservices.*=all
com.ibm.ws.wsgw.*=all
```

Figure C-5 Enable trace for Web Services Gateway

5. When you are finished click **Apply**, then click **OK**.
6. Click **Save**. Click **Save** again to save to master configuration.

Note: The settings for trace enablement will not take effect until the desired application server is restarted if done in the Configuration tab. To avoid the need to restart the server, you can enable trace using the Runtime tab.

Testing techniques

This section includes Web services testing techniques. In our redbook example, we needed to test that the Web services client application (for example, Global Travels) could successfully invoke a Web service (ITSO Car Rental) using the Web Services Gateway. There are a couple of methods of testing, including

- Using Web Services Explorer
- Generate Web services client static stubs

Using Web Services Explorer

Developers can use the Web Services Explorer included with IBM Rational Application Developer V6.x to test the Web services dynamically without creating static stubs on the client side application. This is a very simple and easy to use method of verifying invocation of a Web service.

We have included an example of how we imported the Exposed Gateway WSDL files into Rational Application Developer, and used the wsdl within Web Services Explorer to invoke a Web service (ITSO Car Rental Web service).

1. Start Rational Application Developer.
2. Create a new Dynamic Web project named ExposedWSDL.
   a. Open the Web perspective.
   b. Select **File** → **New** → **Dynamic Web Project**.
   c. Enter `TestWebService` in the Name field, and click **Finish**.
   d. Expand Dynamic **Web Projects** → **TestWebService** → **WebContent** → **WEB-INF.**
   e. Right-click **New** → **Folder**.
   f. Enter `wsdl` and click **OK**.
3. Expand Dynamic **Web Projects** → **TestWebService** → **WebContent** → **WEB-INF** → **wsdl**.
4. Import the wsdl files into the wsdl folder.
5. Right-click the **<filename>Service.wsdl** (for example, WSGBus.WSGServiceService.wsdl), select **Web Services** → **Test with Web Services Explorer**.

   You should see the Web Services Explorer (see Figure C-6 on page 499).

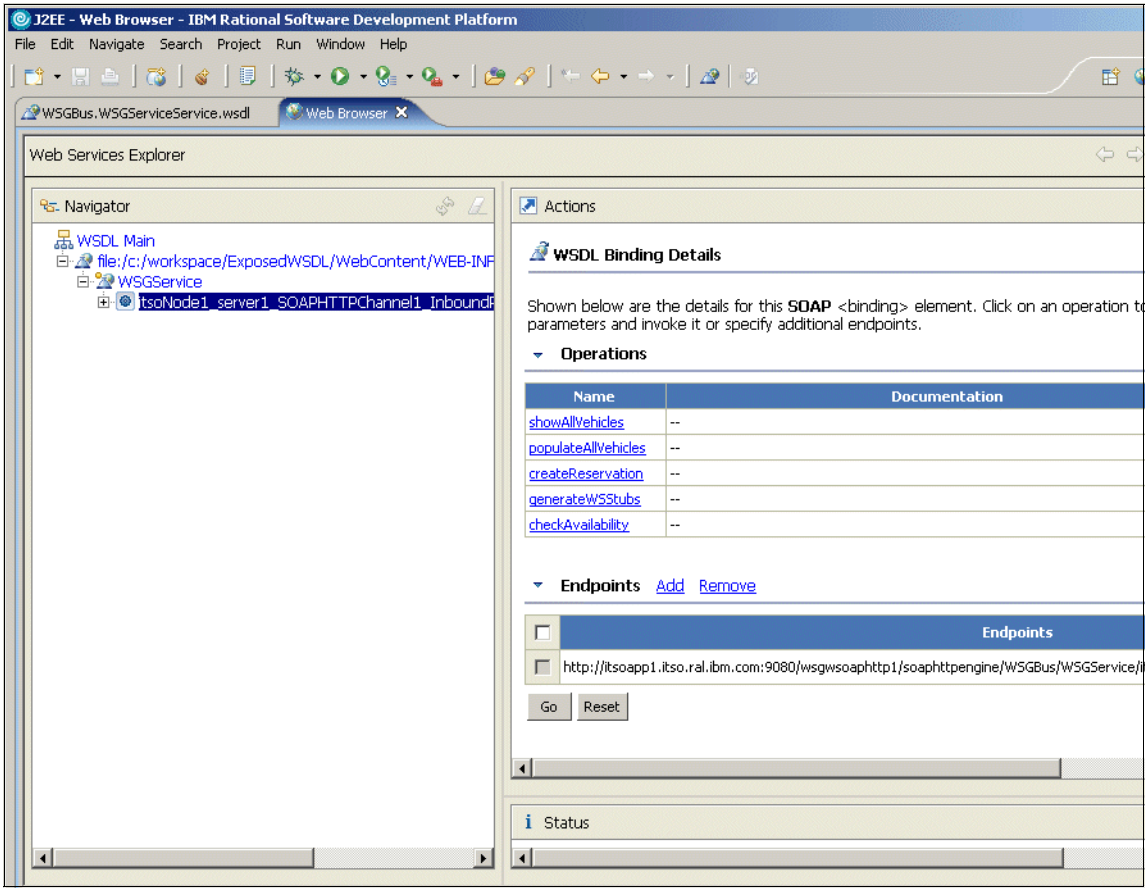


Figure C-6 Web Services Explorer example

6. Click **checkAvailability** under Operations.
7. Enter the application parameters required for the operation checkAvailability as seen in Figure C-7 on page 500, and then click **Go**.

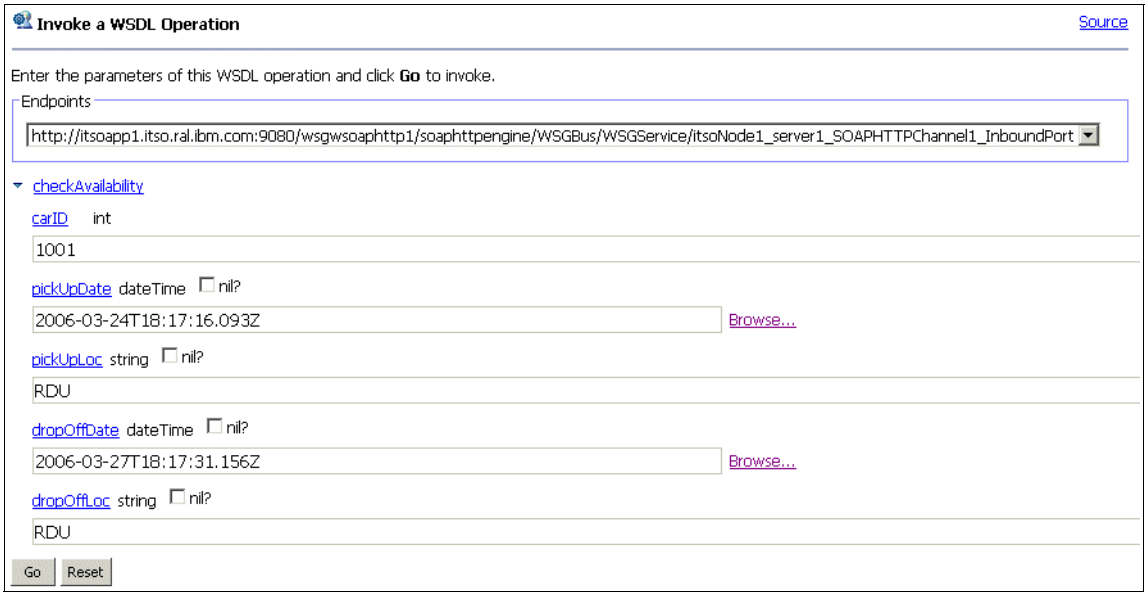


Figure C-7 Invoke operation checkAvailability

8. You should see output similar to the following in the Status window to confirm the operation executed successfully:

```
checkAvailabilityResponse
checkAvailabilityReturn (Boolean): 1
```

Generate Web services client static stubs

To develop and test a Web service, you can follow a similar procedure to what is found in 8.6, “Create the internal Web Service client application” on page 254.

Override endpoint URL

To override and endpoint URL, do the following:

1. Expand **Applications** tab, click **Enterprise Applications** from the Administrative Console.
2. Click **GlobalTravels**.
3. Click **Web modules** under Related Items.
4. Click **GlobalTravelsWeb.war**.
5. Click **Web services client bindings** under Additional Properties.
6. Click **Edit** under Port information.

7. Enter `http://web1.itso.ral.ibm.com/wsgwsoaphttp1/soaphttpengine/WSGBus/WSGService/itsoNode1_server1_SOAPHTTPChannel1_InboundPort` in the Overriden Endpoint URL field and then click **OK**.
8. Click **Save**.
9. Click **Save to Master Configuration**.

Troubleshooting ITCAM for SOA

This section includes troubleshooting tips for ITCAM for SOA.

Tivoli Enterprise Monitoring Server

This section includes troubleshooting tips for ITCAM for SOA specific to the Tivoli Enterprise Monitoring Server.

View trace log

If the Tivoli Enterprise Monitoring Server does not start properly or is not communicating with the agent, we recommend that you view the trace log to troubleshoot the problem.

To view the Tivoli Enterprise Monitoring Server trace log, do the following:

1. Click **Start** → **Programs** → **IBM Tivoli Monitoring** → **Manage Tivoli Monitoring Services**.
2. From the Console, right-click **Tivoli Enterprise Monitoring Server** → **Advanced** → **View Trace Log**.
3. Select the desired log (listed by date), and click **OK**.

Warehouse Proxy database selection

In **Warehouse Proxy Database Selection**, make sure that the correct database type is selected. For example, Database Type is DB2 in our scenario.

In **Configure DB2 Data Source for Warehouse Proxy**, make sure that the login username matches with the administrator ID of the database. For example, in our case, the username `ITMUser` is used for logging into the Warehouse database. Check in DB2 that the administrator username for Warehouse is also `ITMUser`. If there is a mismatch between the usernames specified in this window and the DB2, then the login sequence will fail, causing TEP desktop logon to fail.

Tip: During installation you must remember to set your own password for the userids `ITMUser` and `TEPS`. The default password for these is `itmpswd1.`

Tivoli Enterprise Portal

This section includes troubleshooting tips for ITCAM for SOA specific to the Tivoli Enterprise Portal.

TEP Desktop login

There are a couple of common issues to be aware of when logging into the Tivoli Enterprise Portal Deskop.

- Certificate's signer is not trusted

  When you login to the TEP Desktop, you will see a Security Alert dialog box, with the option to Always Accept, Accept, or Reject.

- Logon failed

  If the logon fails, start the **Manage Tivoli Enterprise Monitoring Services - TEMS Mode** console. Select **Tivoli Enterprise Portal Server**, right-click and select **Advanced** → **Reconfigure**.

  Ensure the communication protocols match (for example, we used IP.PIPE). The TEP Server Configuration window should show that the TEP server status is configured to communicate with the correct monitoring server (for example, TEMS server `itsomon1`).

- Validate user

  By default, the sysadmin used to logon to the TEP Deskop does not require a password. Refer to "Configure TEMS to validate users" on page 436 to enable validating the user ID and password credentials.

- Logon failure as a result of a database logon problem

  If you are not able to logon to TEP successfully, it could be the result of a database logon failure. During installation you must remember to set your own password for the user IDs `ITMUser` and `TEPS`. The default password for these is `itmpswd1`.

ITCAM for SOA agent and server communication

We have listed some common issues encountered with the ITCAM for SOA agent and server communication.

- TEP default workspace does not display Service Management Agent Environment in Navigator.

  We have listed a couple of possible causes and resolutions:

  - Has the KD4 command been run, which copies the kd4dcagent.jar to the <was_home>\lib\ext directory. The jar is loaded when the application server is started. Ensure this file exists.
  - Ensure the ITCAM for SOA agent is started. If it is not started, the Service Management Agent Environment will not be displayed or greyed out.

D

Web Services Security

In this section we will first put in context where Web Services Security (WS-Security) fits within the overall security domain, and identify the possible WS-Security exposures. Next, we will describe the key technologies used to secure Web services. Lastly, we will highlight the impact of implementing security on the system.

The appendix is organized into the following topics:

- Security overview
- Web services security exposures
- Securing Web Services
- Understanding the impact of implementing security
- Implementation references

Security overview

Security is a very vast topic. When developing a strategy for providing a secure environment for your enterprise applications, it is critical to understand the areas of security risk as well as how to reduce security risk. The WS-Security topic falls within the Applications, and Middleware and Application Software categories displayed in Figure 12-26. The other topics are out of scope for this redbook.

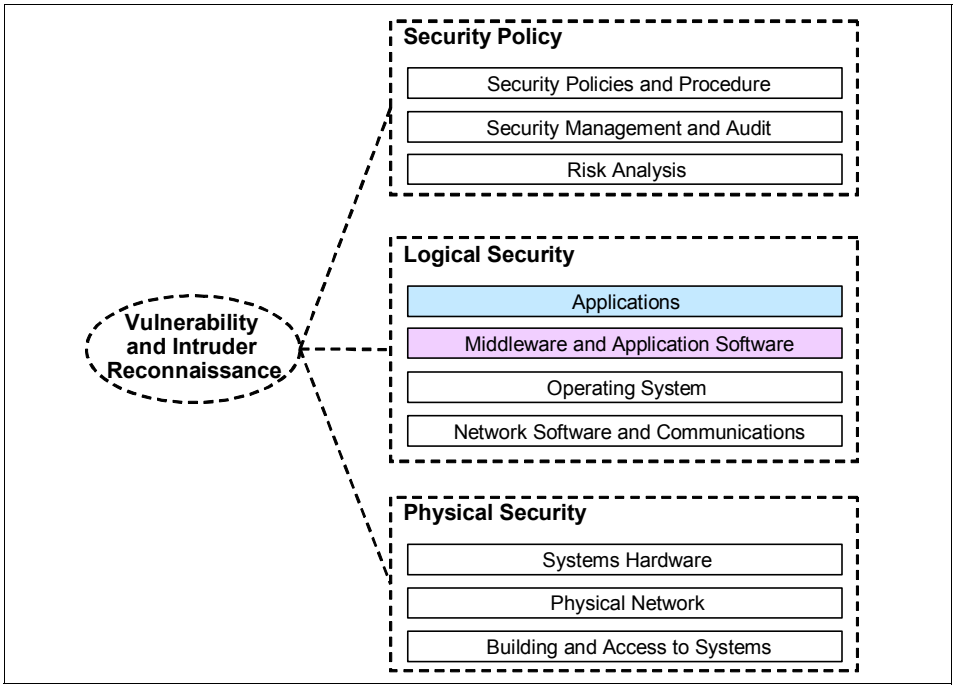


Figure 12-26 Elements of the security domain

We have listed the key elements of logical security framework for the Applications, Middleware and Application Software categories (see Figure 12-26):

- Identity management**:** Identity management entails the tasks required to manage a user's identity within an organization structure such as a directory. This includes the ability to create or provision a user, manage a user, and delete as user. The identity (credentials) of a user is used for authentication and authorization to resources the user is granted access.

- Authentication**:** Authentication is the process where the client identity is validated. The client can be an end user, a machine, or an application.

  The identity of the user, authenticated or unauthenticated, is used to acquire the user's credentials to determine if the user has permissions for the requested resource (authorization).

  Single sign-on is an authentication technique use to provide users with the ability to log on once (authenticate) and be able to access resources or applications within the enterprise the user has been granted permissions.
- Authorization**:** The authorization process provides the capability to permit or deny access to resources based on the policies and users that access the resources. If the resource is protected, the user will first need to be authenticated to determine their identity. Next the privileges for the defined resource will be checked.
- Integrity**:** Integrity ensures that information will not be changed, altered, or lost in an unauthorized or accidental manner.
- Confidentiality**:** No unauthorized party or process can access or disclose the information.
- Auditing**:** All transactions are recorded to ensure system integrity. The transaction data recorded can be analyzed after the fact.
- Non-repudiation**:** Both parties are able to provide legal proof to a third party that the sender did send the information, and the receiver received the identical information. Neither involved side is "unable to deny."

Tip: We recommend that you refer to the following reference information to further understand the general security issues common to Web environments:

- System Administration, Networking and Security Institute (SANS):

  http://www.sans.org/

- The Center for Internet Security (CIS):

  http://www.cisecurity.org/

- *Enterprise Security Architecture Using IBM Tivoli Security Solutions, SG24-6014*
- *WebSphere Version 6 Web Services Handbook, Development and Deployment*, SG24-6461
- *WebSphere Application Server V6 Security Handbook*, SG24-6316
- *Identity and Access Management Solutions, Using WebSphere Portal V5.1, Tivoli Identity Manager V4.5.1, and Tivoli Access Manager V5.1*, SG24-6692
- *Develop and Deploy a Secure Portal Solution, Using WebSphere Portal V5 and Tivoli Access Manager V5.1*, SG24-6325
- *Hacking Exposed: Network Security Secrets & Solutions*, Fifth Edition, Stuart McClure et al.

Web services security exposures

When using Web services, similar security exposures exists as for other Internet, middleware-based applications, and communications.

We have listed three major security risk areas to consider:

- **Spoofing (no authentication):** An attacker could send a modified SOAP message to the service provider, to get confidential information or perform a transaction.

  By applying authentication to the Web service, this security exposure can be eliminated.

- **Tampering (no integrity):** A SOAP message may be intercepted between the Web service consumer and provider. An attacker could modify the message (for example, deposit the money into another account by changing the account number). Because there is no integrity constraint, the Web service server does not check if the message is valid and will accept the modified transaction.

  By applying an integrity mechanism to the Web service, this security exposure can be eliminated.

- **Eavesdropping (no confidentiality):** An attacker can intercept the SOAP message and read all contained information. Because the message is not encrypted, confidential of the information can go the wrong people. This exposure exists because the information in the SOAP message is sent over the network in plain text.

  By applying a confidentiality mechanism to the Web service, this security exposure can be eliminated.

Securing Web Services

To prevent the security exposures outlined in “Web services security exposures” on page 508, the following mechanisms can be applied to secure a Web services environment (Figure 12-27):

- Message-level security (WS-Security)
  - Authentication
  - Confidentiality
  - Integrity
- Transport-level security (TLS/SSL)

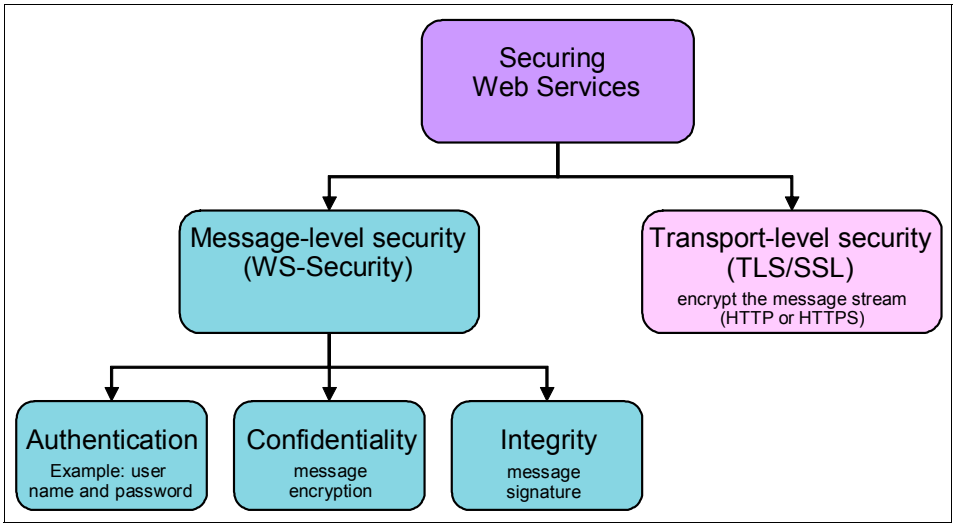


Figure 12-27 Securing Web services

Message-level security (WS-Security)

The WS-Security specification provides message-level security, which is used when building secure Web services to implement message content integrity and confidentiality, and authentication. In WebSphere Application Server V6.0, there are many options to apply these security mechanisms. Figure 12-28 shows an example of Web service security elements when the SOAP body is signed and encrypted.

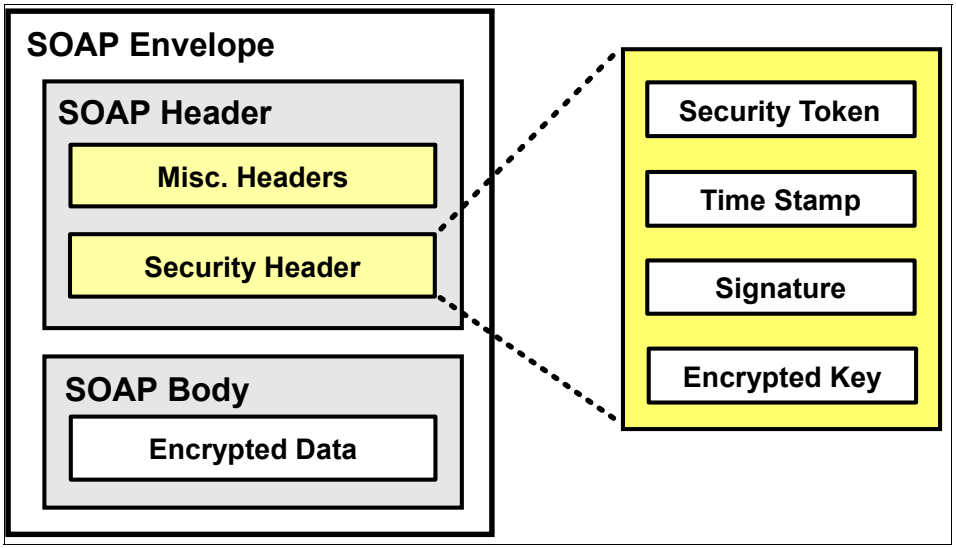


Figure 12-28 SOAP message security with WS-Security

There are three key elements possible for WS-Security (see Figure 12-27 on page 510):

- **Authentication:** There are several methods of implementing WS-Security authetication to validate the identity of the client (user or system credentials) before allowing or denying access to the requested resource.

  When using basic authentication, the user name and password are included in the `<UsernameToken>` tag of the SOAP message. Once the user name token is received by the Web service provider, the user name and password are extracted and verified. The message is only accepted and processed if the user name and password credentials are valid.

  Other forms of authentication are digital signature, identity assertion, LTPA token, and custom tokens (identity assertion, LTPA token, and custom tokens are extensions of IBM WebSphere Application Server V6.0).

- **Confidentiality:** When using WS-Security confidentiality, the message body part is protected using XML encryption and other security information is added. Once the SOAP message is encrypted, only a service that knows the key for confidentiality can decrypt and read the message.

- **Integrity:** Integrity is applied to the application to ensure that no one illegally modifies the message while it is in transit. Integrity is provided by generating an XML digital signature on the contents of the SOAP message.

  The signature is created based on a key that the sender is authorized to have. Unauthorized sniffers do not have this key. When the receiver gets the

message, it too creates a signature using the message contents. The receiver only honors the integrity of the message if the two signatures match. If the signatures are different, a SOAP fault is returned to the sender.

The advantage of using WS-Security over SSL is that it can provide end-to-end message-level security. This means that the message security can be protected even if the message goes through multiple services, called intermediaries.

Additionally, WS-Security is independent of the transport layer protocol, thus it can be used for any Web service binding (for example, HTTP, SOAP, RMI). Using WS-Security, end-to-end security can be obtained (Figure 12-29).

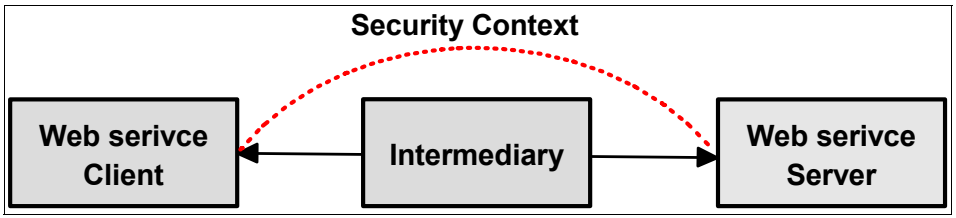


Figure 12-29 End-to-end security with message-level security

Transport-level security (TLS/SSL)

HTTP is the most commonly used Internet communication protocol, and is also the most popular protocol for Web services. HTTP is an inherently insecure protocol, given that all information is sent in clear text between unauthenticated peers over an insecure network.

To secure HTTP, transport-level security can be applied. Transport-level security is a well-known and often used mechanism to secure HTTP Internet and intranet communications. Transport-level security is based on Secure Sockets Layer (SSL) or Transport Layer Security (TLS) that runs beneath HTTP.

HTTPS allows client-side and server-side authentication through certificates, which have been either self-signed or signed by a certification agency.

For Web services bound to the HTTP protocol, HTTPS/SSL can be applied in combination with message-level security (WS-Security).

Unlike message-level security, HTTPS encrypts the *entire* HTTP data packet. There is no option to apply security selectively on certain parts of the message. SSL and TLS provide security features including authentication, data protection, and cryptographic token support for secure HTTP connections.

Although HTTPS does not cover all aspects of a general security framework, it provides a security level regarding party identification and authentication, message integrity, and confidentiality. It does not provide authentication, auditing, and non-repudiation. SSL cannot be applied to other protocols, such as JMS. To run HTTPS, the Web service port address must be in the form `https://`.

Even with the WS-Security specification, SSL should be considered when thinking about Web services security. Using SSL, a point-to-point security can be achieved (Figure 12-30).

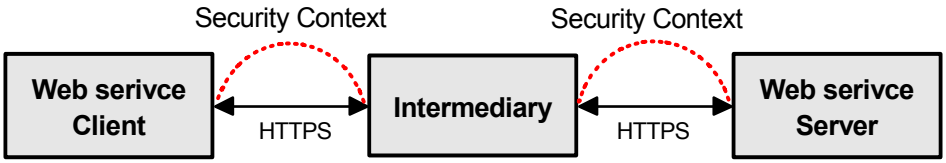


Figure 12-30 Point-to-point security with HTTPS

Here are a few simple guidelines to help decide when transport-level security should be used:

- No intermediaries are used in the Web service environment.

  With intermediaries, the entire message has to be decrypted to access the routing information. This would break the overall security context.

- The transport is only based on HTTP.

  No other transport protocol can be used with HTTPS.

- The Web services client is a stand-alone Java program.

  WS-Security can only be applied to clients that run in a J2EE container (EJB container, Web container, application client container). HTTPS is the only option available for stand-alone clients.

Understanding the impact of implementing security

Depending on the demanded level of application security, one or more of these security mechanisms can be applied. Also depending on other non-functional requirements, a combination of message-level security and transport-level security can be implemented.

The more security mechanisms implemented, which increase the security effect, the more influence other non-functional requirements are given. Therefore, when designing a Web services security solution, keep in mind that security has an impact on system capacity and performance non-functional requirements.

System capacity

Any applied security mechanism has impact on system resource usage (for example, CPU and memory usage). So, when planning a Web service environment, the required *security overhead* must be considered in the system capacity and volume planning.

The non-functional requirements, capacity and volume, cover the number of concurrent users and the number of transactions per second. This has influence on the required system infrastructure (hardware, network).

Performance

Security mechanisms and functions also impact the application's response time. When defining the Web service system response time requirements, keep in mind that the response time will be affected when applying security.

The performance requirement for a system defines the response time for a main application operation (for example, less than one second for 90% of all transactions).

Implementation references

For implementation details on Web Services security, refer to the following:

- *WebSphere Version 6 Web Services Handbook, Development and Deployment*, SG24-6461
- *Patterns: Extended Enterprise SOA and Web Services*, SG24-7135

In the ITSO Car Rental working example, the connection between the Travel Application Server node (travel1) and the Web Server Redirector node (web1) can optionally be SSL enabled.

For implementation details on enabling SSL refer to the following:

- *Patterns: Extended Enterprise SOA and Web Services*, SG24-7135
- IBM WebSphere Application Server V6 InfoCenter found at:

  `http://www.ibm.com/software/webservers/appserv/was/library/library60.html`

E

Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

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

ftp://www.redbooks.ibm.com/redbooks/SG247240

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

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG247240.

Using the Web material

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

File name	*Description*
7240code.zip	Sample code zip file for redbook.

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space: 10 MB
Operating System: Windows
Processor: 1 GHz or higher
Memory: 512 MB

Unzip the Web material

Unzip the contents of the Web material zip file. After the unzip of the 7240code.zip, you will have a directory named C:\7240code.

Description of the Web material

Table E-1 displays the directories and description of the Web material after the unzip.

Table E-1 Description of Web material

Directory	Description
C:\7240code\assemble\ITSOCarRental.j2ee	Contains a project interchange file and database scripts for the base J2EE ITSO Car Rental application used as a starting point for our working example.
C:\7240code\assemble\ITSOCarRental.ws	Contains the completed Web Services project interchange file for the ITSO Car Rental sample application. It also, include a database script used to extend a database table for the sample.
C:\7240code\assemble\GlobalTravels.ws	Contains the completed Web Services project interchange file for the Global Travels application.
C:\7240code\assemble\intermediatefiles	Contains completed individual files during assembling.
C:\7240code\assemble\ITSOCarRentalLoadTest	Contains completed code to generate traffic.
C:\7240code\deploy\ITSOCarRental.ws	This directory contains the EAR files used to deploy the ITSO Car Rental application. It also includes scripts used to create and populate the application database.

Directory	Description
C:\7240code\deploy\GlobalTravels.ws	This directory contains the EAR file used to deploy the Global Travels enterprise application.

Related publications

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

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 522. Note that some of the documents referenced here may be available in softcopy only.

- *WebSphere Version 6 Web Services Handbook, Development and Deployment*, SG24-6461
- *Patterns: Extended Enterprise SOA and Web Services*, SG24-7135
- *Patterns: SOA with an Enterprise Service Bus in WebSphere Application Server V6*, SG24-6494
- *Patterns: SOA Foundation - Service Connectivity Scenario*, SG24-7228
- *Patterns: SOA Foundation - Business Process Management Scenario*, SG24-7234
- *Patterns: Implementing Self-Service in an SOA Environment*, SG24-6680-01
- *Rational Application Developer V6 Programming Guide*, SG24-6449
- *Patterns: Information Aggregation and Data Integration with DB2 Information Integrator*, SG24-7101
- *Patterns: Portal Search Custom Design*, SG24-6881
- *Patterns: Service-Oriented Architecture and Web Services*, SG24-6303
- *Business Process Management: Modeling through Monitoring Using WebSphere V6 Products*, SG24-7148
- *Patterns: SOA Client Access Integration Solutions*, SG24-6775
- *Identity and Access Management Solutions, Using WebSphere Portal V5.1, Tivoli Identity Manager V4.5.1, and Tivoli Access Manager V5.1*, SG24-6692
- *IBM Tivoli Composite Application Manager V6.0 Family: Installation, Configuration, and Basic Usage*, SG24-7151
- *Revealed! Architecting e-business Access to CICS*, SG24-5466
- *Application Development for CICS Web Services*, SG24-7126

- *CICS Transaction Gateway for z/OS Version 6.1*, SG24-7161
- *IBM Tivoli OMEGAMON XE V3.1 Deep Dive on z/OS*, SG24-7155
- *A Portal Composite Pattern Using WebSphere Portal V5*, SG24-6087
- *Patterns: Implementing Self-Service in an SOA Environment*, SG24-6680
- *WebSphere Portal Best Practices*, REDP4100
- *Develop and Deploy a Secure Portal Solution, Using WebSphere Portal V5 and Tivoli Access Manager V5.1*, SG24-6325
- *Enabling SOA Using WebSphere Messaging*, SG24-7163
- *Getting Started with WebSphere Enterprise Service Bus V6*, SG24-7212
- *IBM Rational Application Developer V6 Portlet Application Development and Portal Tools*, SG24-6681
- *IBM WebSphere Portal for Multiplatforms V5.1 Handbook*, SG24-6689
- *Portal Application Design and Development Guidelines*, REDP3829
- *Understanding SOA Security Design and Implementation*, SG24-7310

Other publications

These publications are also relevant as further information sources:

- *Installation and User's Guide, IBM Tivoli Composite Application Manager for SOA*, GC32-9492
- *Installation and Setup Guide, IBM Tivoli Monitoring V6.1*, GC32-9407
- *Installing and Troubleshooting IBM Web Services Navigator*, GC32-9494
- Stuart McClure, et al., *Hacking Exposed: Network Security Secrets & Solutions*, Osborne/McGraw Hill, Fifth Edition, May 2005, ISBN 0072260815

Online resources

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

- IBM WebSphere Application Server InfoCenter found at:

  `http://www.ibm.com/software/webservers/appserv/was/library/`

- *IBM SOA Foundation, An Architectural Introduction and Overview V1.0* found at:

  `http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf`

- *SOA - Patterns and Best Practices* found at:

  ftp://www6.software.ibm.com/software/developer/techbriefings/presentations/soa/patterns.pdf
- Web Services Interoperability Organization (WS-I) found at:

  http://ws-i.org
- *SOA antipatterns* found at:

  http://www.ibm.com/developerworks/webservices/library/ws-antipatterns/
- *Increase flexibility with the Service Integration Maturity Model* found at:

  http://www.ibm.com/developerworks/webservices/library/ws-soa-simm/
- *Service Oriented Modeling and Architecture* found at:

  http://www.ibm.com/developerworks/webservices/library/ws-soa-design1/
- *Elements of Service-Oriented Analysis and Design* found at:

  http://www.ibm.com/developerworks/webservices/library/ws-soad1/
- *SOA realization: Service design principles*, found at:

  http://www.ibm.com/developerworks/webservices/library/ws-soa-design/
- IBM SOMA and/or IBM SOA:

  http://www.ibm.com/services/soa
- *Insight and outlook, Part 1: Why and when should you choose SOA?* found at:

  http://www.ibm.com/developerworks/library/ar-itio1/
- *UML 2.0 Profile for Software Services* (conceptual overview of the plug-in) found at:

  http://www.ibm.com/developerworks/rational/library/05/419_soa/
- *RUP Plug-In for SOA V1.0*, provides a link to download the plug-in:

  http://www.ibm.com/developerworks/rational/library/05/510_soaplug/
- *RUP for SOA and SOMA - confused? you may be not..*, found at:

  http://www.ibm.com/developerworks/blogs/dw_blog.jspa?blog=352
- *Realizing service-oriented solutions with the IBM Rational Software Development Platform* found at:

  http://www.findarticles.com/p/articles/mi_mOISJ/is_4_44/ai_n15969319/pg_7

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

D

E

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Patterns: SOA Foundation Service Creation Scenario

(1.0” spine)
0.875”<->1.498”
460 <-> 788 pages

Patterns: SOA Foundation Service Creation Scenario

Redbooks

Key concepts and architecture of the IBM SOA Foundation

Process for identifying SOA scenarios, patterns and services

Service Creation scenario working examples

This IBM Redbook provides an introduction to the IBM SOA Foundation, and includes a detailed implementation example for the Service Creation scenario.

Part 1, “Getting started with IBM SOA Foundation” on page 1, includes an introduction to SOA from a business and architecture perspective. We describe the key elements of the IBM SOA Foundation, including the SOA life cycle, logical architecture, and SOA scenarios. Next we describe the Service Creation scenario in more detail since this is the focus of redbook example. We describe best practices and guidelines for SOA. In addition, we include a process for applying the SOA scenarios and patterns.

Part 2, “Service Creation scenario example” on page 171, provides an end-to-end working example representative of the Service Creation scenario. We start by modeling the business requirements of a fictitious car rental company that has an existing J2EE™ based application. We demonstrate how to identify services using SOMA, and use a process for applying the SOA scenarios and reusable patterns to accelerate the creation of a solution architecture. The remaining chapters provide the implementation details to assemble using Rational® Application Developer V6.0.1, deploy using WebSphere® Application Server Network Deployment V6.0.2, and manage the solution using Tivoli® Composite Application Manager for SOA V6.0

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

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

For more information:
ibm.com/redbooks

SG24-7240-00 ISBN 0738494852