Oracle® Fusion Middleware Application Adapters

Application Adapter for J.D. Edwards OneWorld User's Guide
for 12*c* (12.2.1.2.0)

E84215-01

December 2016

Provides information on how to integrate with J.D. Edwards OneWorld systems and develop applications.

Oracle Fusion Middleware Application Adapter 12*c* (12.2.1.2.0) for J.D. Edwards OneWorld User's Guide for Oracle WebLogic Server, 12*c* (12.2.1.2.0)

E84215-01

Primary Author: Stefan Kostial

Contributors: Vikas Anand, Marian Jones, Sunil Gopal, Bo Stern

Contents

Preface ix

Audience ix
Documentation Accessibility ix
Related Documents ix
Conventions x

1 Introduction

Adapter Features 1-1
J.D. Edwards OneWorld Concepts 1-2
Integration with J.D. Edwards OneWorld 1-2
- Propagating Internal Listeners Out of J.D. Edwards OneWorld 1-3
- J.D. Edwards OneWorld Interoperability Framework 1-3

Adapter Architecture 1-5
- Oracle Adapter Application Explorer (Application Explorer) 1-6
- Resource Adapters 1-6
- Oracle WebLogic Server Adapter Business Services Engine (BSE) Architecture 1-6
- Oracale WebLogic Server Adapter Generic J2CA Architecture 1-7
- Processing Business Functions 1-8

BSE Versus Oracle Adapter J2CA Deployment 1-8
Sample Projects 1-9
Quick Start Guide 1-10
- Installation 1-10
- Copying Third-Party Library Files 1-11
- Configuration 1-11
- WebLogic Server Deployment and Integration 1-12
- Creating Configurations, Targets, and Channels in Application Explorer 1-12
- Working With Service Components in the SOA Suite 1-13
- Working With Oracle Service Bus 1-13
- Other Features 1-13

2 Configuring Oracle Application Adapter for J.D. Edwards OneWorld

Starting Application Explorer 2-1
Configuring Repository Settings 2-2
Creating a Repository Configuration 2-2
- Creating a Configuration for BSE 2-2

Creating a Configuration for J2CA 2-3
Connecting to a BSE or J2CA Configuration 2-4
Establishing a Connection (Target) for J.D. Edwards OneWorld 2-5
Defining a Target to J.D. Edwards OneWorld 2-5
Creating an XML Schema 2-10
Creating a Request and a Response Schema 2-10
Generating WSDL (J2CA Configurations Only) 2-11
Generating a WSDL for Outbound Interaction 2-11
Creating and Testing a Web Service (BSE Configurations Only) 2-13
Creating a Web Service 2-13
Testing a Web Serice 2-14
Identity Propagation 2-14
Configuring an Event Adapter 2-15
Creating and Editing a Channel 2-15
The J.D. Edwards OneWorld Event Listener 2-28
Configuring the J.D. Edwards OneWorld Event Listener 2-28
Runtime Overview 2-30
Modifying the JDE.INI File for Outbound and Inbound Processing 2-31

3 Oracle WebLogic Server Deployment and Integration

Adapter Integration with Oracle WebLogic Server 3-1
Deployment of Adapter 3-1
Updating Adapter Configuration 3-2
Creating a Managed Connector Factory Object 3-3
Creating Multiple Managed Connector Factory Objects 3-4
Modifying WSDL Files for Additional Connection Factory Values 3-6

4 Integration With BPEL Service Components in the Oracle SOA Suite

Overview 4-1
Deployment of Adapter 4-2
Configuring a New Application Server Connection 4-2
Designing an Outbound BPEL Process for Service Integration (J2CA Configuration) 4-7
Generating WSDL for Request/Response Service 4-8
Creating an Empty Composite for SOA 4-9
Defining a BPEL Outbound Process 4-11
Deploying the BPEL Outbound Process 4-28
Invoking the Input XML Document in the Oracle Enterprise Manager Console 4-31
Testing Outbound BPEL and Mediator Processes 4-33
Designing an Inbound BPEL Process for Event Integration (J2CA Configuration) 4-34
Generating WSDL for Event Integration 4-34
Creating an Empty Composite for SOA 4-39
Defining a BPEL Inbound Process 4-40
Deploying the BPEL Inbound Process 4-46
Triggering an Event in J.D. Edwards OneWorld 4-47
Designing an Outbound BPEL Process for Service Integration (BSE Configuration) 4-52
Generating a WSDL File for Request and Response Services Using a Web Service 4-52
Creating an Empty Composite for SOA 4-53

Defining a BPEL Outbound Process 4-53

5 Integration With Mediator Service Components in the Oracle SOA Suite

Configuring a New Application Server Connection 5-2
Configuring a Mediator Outbound Process (J2CA Configuration) 5-2
Creating an Empty Composite for SOA 5-2
Defining a Mediator Outbound Process 5-3
Deploying the Mediator Outbound Process 5-10
Invoking the Input XML Document in the Oracle Enterprise Manager Console 5-11
Configuring a Mediator Inbound Process (J2CA Configuration) 5-11
Creating an Empty Composite for SOA 5-11
Defining a Mediator Inbound Process 5-11
Configuring a Mediator Outbound Process (BSE Configuration) 5-17
Creating an Empty Composite for SOA 5-18
Defining a Mediator Outbound Process 5-18

6 Integration With BPM Service Components in the Oracle SOA Suite

Overview 6-1
Deployment of Adapter 6-2
Configuring a New Application Server Connection 6-2
Designing an Outbound BPM Process Using Transformations for Service Integration (J2CA Configuration) 6-2
Creating an Empty Composite for BPM 6-2
Defining a BPM Outbound Process 6-3
Adjusting for Known Deployment Issues With 12c 6-25
Deploying the BPM Outbound Process 6-25
Invoking the Input XML Document in the Oracle Enterprise Manager Console 6-26
Designing an Inbound BPM Process Using Transformations for Event Integration (J2CA Configuration) 6-27
Creating an Empty Composite for BPM 6-27
Defining a BPM Inbound Process 6-28
Designing an Outbound BPM Process Using Transformations for Service Integration (BSE Configuration) 6-42
Creating an Empty Composite for BPM 6-42
Defining a BPM Outbound Process 6-42

7 Configuring an Outbound and Inbound Process for Oracle Service Bus Using sbconsole

Overview of Application Adapter Integration with Oracle Service Bus 7-1
Configuring an Outbound Process Using Sbconsole (J2CA Configuration) 7-2
Starting Oracle Service Bus and Creating Project Folders 7-2
Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus 7-6
Publishing a WSDL From Application Explorer to Oracle Service Bus 7-6
Configuring a WSDL-based Business Service 7-7
Configuring a File Type Business Service 7-9
Configuring a Pipeline With Proxy Service 7-13

Configuring an Inbound Process Using sbconsole (J2CA Configuration) 7-22
Starting Oracle Service Bus and Creating Project Folders 7-22
Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus 7-22
Generating WSDL for Event Integration 7-23
Configuring a WSDL-based Proxy Service 7-23
Configuring a File Type Business Service 7-25
Configuring a Pipeline 7-29
Configuring an Outbound Process Using Sbconsole (BSE Configuration) 7-36
Starting Oracle Service Bus and Creating Project Folders 7-36
Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus 7-36
Publishing a WSDL From Application Explorer to Oracle Service Bus 7-37
Configuring a File Type Business Service 7-38
Configuring a WSDL-based Business Service 7-38
Configuring a Pipeline With Proxy Service 7-40
Configuring JMS Proxy Services Using Oracle Service Bus (J2CA Configuration) 7-49
Configuring HTTP Proxy Services Using Oracle Service Bus (J2CA Configuration) 7-64

8 Configuring an Outbound and Inbound Process for Oracle Service Bus Using JDeveloper

Configuring an OSB Outbound Process Using JDeveloper (J2CA Configuration) 8-1
Creating a Service Bus Application for OSB 8-2
Defining an OSB Outbound Process 8-3
Deploying the OSB Outbound Process 8-15
Configuring an OSB Inbound Process Using JDeveloper (J2CA Configuration) 8-18
Creating a Service Bus Application for OSB 8-18
Defining an OSB Inbound Process 8-18
Deploying the OSB Inbound Process 8-26
Configuring an OSB Outbound Process Using JDeveloper (BSE Configuration) 8-27
Creating a Service Bus Application for OSB 8-27
Defining an OSB Outbound Process 8-27
Deploying the OSB Outbound Process 8-41
Configuring a JMS Inbound Process Using JDeveloper (J2CA Configuration) 8-41
Configuring a JMS Outbound Process Using JDeveloper (J2CA Configuration) 8-47
Configuring an HTTP Outbound Process Using JDeveloper (J2CA Configuration) 8-54

9 Key Features

Configuring the Logging Feature 9-1
Configuring Log File Management for the J2CA Connector Application 9-1
Configuring Log File Management for Business Services Engine (BSE) 9-8
Configuring the Diagnosibility Feature 9-11
Supporting Protocols 9-12
Configuring the SOA Debugging Feature 9-13
Guidelines for Using the SOA Debugger 9-13
Prerequisite 9-13
Debugging a BPEL Process in Oracle JDeveloper 9-13
Debugging an OSB Process in Oracle JDeveloper 9-27
Exception Filter 9-45

Configuring the Exception Filter 9-45
Credential Mapping for Oracle SOA Suite (BPEL, Mediator, or BPM) 9-56
Configuring Credential Mapping 9-57
Credential Mapping for Oracle Service Bus (OSB) 9-61
Configuring Credential Mapping 9-62

10 Troubleshooting and Error Messages

Troubleshooting 10-1
Application Explorer 10-1
J.D. Edwards One World 10-3
Oracle Adapter J2CA 10-4
BSE Error Messages 10-4
General Error Handling in BSE 10-4
Adapter-Specific Error Handling 10-5

A Configuring J.D. Edwards OneWorld for Outbound and Inbound Processing

Modifying the JDE.INI File for Outbound and Inbound Processing A-1
Using the GenJava Development Tool (Outbound Processing) A-2
Running GenJava A-2
Triggering J.D. Edwards OneWorld Events A-4
Starting the Outbound Scheduler Subsystem Process (R00460) A-6
Verifying the Subsystem Process A-7
Configuring P4210 (Sales Order) to Trigger an Event A-9
Verifying the Configuration Steps A-11

Glossary

Index

Preface

Welcome to *Oracle Fusion Middleware Application Adapter for J.D. Edwards OneWorld User's Guide* for Oracle WebLogic Server. This document provides information on how to integrate with J.D. Edwards OneWorld systems and develop applications.

Audience

This document is intended for system administrators and developers who integrate with J.D. Edwards OneWorld systems and develop applications.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Enterprise Repository 12*c* (12.2.1.2.0) documentation set:

- *Oracle Fusion Middleware Application Adapters Installation Guide for Oracle WebLogic Server*
- *Oracle Fusion Middleware Application Adapter Upgrade Guide for Oracle WebLogic Server*
- *Oracle Fusion Middleware Application Adapter Best Practices Guide for Oracle WebLogic Server*
- Oracle's Unified Method (OUM)

  A wealth of additional Governance information can be found within Oracle's Unified Method (OUM). OUM can be used by Oracle employees, Oracle Partner Network Certified Partners or Certified Advantage Partners, and Clients who either participate in the OUM Customer Program or are engaged on projects where Oracle provides consulting services. OUM is a web-deployed toolkit for planning, executing and controlling software development and implementation projects.

For more information about OUM, see the OUM FAQ at

http://my.oracle.com/portal/page/myo/ROOTCORNER/KNOWLEDGEAREAS1/BUSINESS_PRACTICE/Methods/Learn_about_OUM.html

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

Oracle WebLogic Server connects to a J.D. Edwards OneWorld system through Oracle Application Adapter for J.D. Edwards OneWorld. Oracle Application Adapter for J.D. Edwards OneWorld provides connectivity and carries out interactions on a J.D. Edwards OneWorld system.

Note: Throughout this document, *<ORACLE_HOME>* refers to the 12*c* (12.2.1.0.0) SOA/OSB installed home location.

<ADAPTER_HOME> refers to the following:

- For SOA:

  `<ORACLE_HOME>\soa\soa\thirdparty\ApplicationAdapters`

- For OSB:

  `<ORACLE_HOME>\osb\3rdparty\ApplicationAdapters`

This chapter contains the following sections:

- Section 1.1, "Adapter Features"
- Section 1.2, "J.D. Edwards OneWorld Concepts"
- Section 1.3, "Integration with J.D. Edwards OneWorld"
- Section 1.4, "Adapter Architecture"
- Section 1.5, "BSE Versus Oracle Adapter J2CA Deployment"
- Section 1.6, "Sample Projects"
- Section 1.7, "Quick Start Guide"

1.1 Adapter Features

Oracle Application Adapter for J.D. Edwards OneWorld provides a means to exchange real-time business data between J.D. Edwards systems and other applications, databases, or external business partner systems. The **adapter** enables inbound and outbound processing with J.D. Edwards.

Oracle Application Adapter for J.D. Edwards OneWorld can be deployed as a J2EE Connector Architecture (J2CA) 1.0 resource adapter. This deployment is referred to as Oracle Adapter J2CA. It can also be deployed as a Web services servlet and is referred to as Oracle Adapter Business Services Engine (BSE).

Oracle Application Adapter for J.D. Edwards OneWorld uses XML messages to enable non-J.D. Edwards OneWorld applications to communicate and exchange transactions with J.D. Edwards OneWorld using services and events. Services and events are described as follows:

- Services: Enables applications to initiate a J.D. Edwards OneWorld business event.
- Events: Enables applications to access J.D. Edwards OneWorld data only when a J.D. Edwards OneWorld business event occurs.

To support event functionality, channels are supported. A **channel** represents configured connections to particular instances of back-end or other types of systems.

The channel is the adapter component that receives events in real time from the Enterprise Information System (EIS) application. The channel component can be a File reader, an HTTP listener, a TCP/IP listener, or an FTP listener. A channel is always EIS specific. The adapter supports multiple channels for a particular EIS, which enables the user to choose the optimal channel component based on deployment requirements.

Oracle Application Adapter for J.D. Edwards OneWorld provides:

- XML schemas and WSDLs for the J2CA 1.0 and 1.5 resource adapter.
- Web services for BSE.

See Also: *Oracle Application Server Adapter Concepts Guide*

1.2 J.D. Edwards OneWorld Concepts

You can use Oracle Application Adapter for J.D. Edwards OneWorld to call a J.D. Edwards OneWorld Master Business Function, such as Address Book, Purchase Order, and Sales Order. You can also use the adapter as a part of an integration effort to connect J.D. Edwards OneWorld with non-J.D. Edwards OneWorld systems.

Oracle Application Adapter for J.D. Edwards OneWorld can receive an XML document, or it can run one or more J.D. Edwards OneWorld Master Business Functions (MBF) by passing an XML document into J.D. Edwards OneWorld through the J.D. Edwards OneWorld ThinNet API.

1.3 Integration with J.D. Edwards OneWorld

This section contains the following topics:

- Section 1.3.1, "Propagating Internal Listeners Out of J.D. Edwards OneWorld"
- Section 1.3.2, "J.D. Edwards OneWorld Interoperability Framework"

J.D. Edwards OneWorld supports multiple methods and technologies to provide interoperability. The three supported entry points are:

- Flat files
- Database tables
- Master Business Function (MBF) interactive calls

You configure Oracle AS Adapter to send requests to J.D. Edwards OneWorld. The adapter processes requests for J.D. Edwards OneWorld Master Business Functions (MBF), embedded in XML documents, and forwards them to a back-end J.D. Edwards OneWorld system. The resulting response information is then returned and processed for further routing.

Oracle Application Adapter for J.D. Edwards OneWorld can receive an XML request document from a client and call a specific function in the target Enterprise Information System (EIS). Oracle Application Adapter for J.D. Edwards OneWorld acts as a consumer of request messages and provides a response. An adapter performs the following functions:

- Receives requests from a legacy system, another EIS, or a non-EIS client.
- Transforms the XML request document into the EIS-specific format.

  The request document conforms to a request XML schema.
  The schema is based on metadata in the EIS.
- Calls the underlying function in the EIS and waits for its response.
- Transforms the response from the EIS-specific data format to an XML document.

  The response document conforms to a response XML schema that is generated by the adapter.
  The schema is based on metadata in the EIS.

You can configure a channel for the adapter to receive messages from J.D. Edwards OneWorld. The information the channel receives is used to build an XML record and is forwarded to any specified disposition for further processing.

Channels are consumers of EIS-specific messages and may or may not provide a response. A channel performs the following functions:

- Receives messages from an EIS client
- Transforms the EIS-specific message format into an XML format.

1.3.1 Propagating Internal Listeners Out of J.D. Edwards OneWorld

Integrating a J.D. Edwards OneWorld listener with external systems is similar to the outbound process, except in reverse. The Data Export Control table maintains the determination of whether a transaction must be integrated with an external system. When a transaction must be integrated, the MBF handles logging of all additions, changes, and deletions to the unedited transaction table. After the transaction information writes to the table, a key for that record is sent from the MBF to the subsystem data queue.

The subsystem data queue triggers the processing of the new record by launching an outbound subsystem batch process that is generic and handles all inbound transactions. The J. D. Edwards outbound subsystem then accesses the Data Export Control table to determine the configured external subscriber to run.

1.3.2 J.D. Edwards OneWorld Interoperability Framework

J.D. Edwards OneWorld enables integration with systems through its interoperability framework. The adapter uses the framework and leverages various integration access methods to provide the greatest amount of flexibility and functionality.

Oracle Application Adapter for J.D. Edwards OneWorld supports the following integration access methods:

- J.D. Edwards OneWorld ThinNet API
- J.D. Edwards OneWorld XML
- J.D. Edwards OneWorld unedited transaction tables (Z tables)

Figure 1–1 illustrates the outbound processing framework.

The adapter uses the J.D. Edwards OneWorld ThinNet API to communicate with the J.D. Edwards OneWorld application. Using the ThinNet API, the adapter can run one or more MBF in a single Unit Of Work (UOW). When any of the MBF fail, the entire UOW fails, preventing partial updates. Validation of data, business rules, and communications to the underlying database are handled by the J.D. Edwards OneWorld application because the adapter runs the MBF.

Figure 1–1 J.D. Edwards OneWorld Outbound Processing

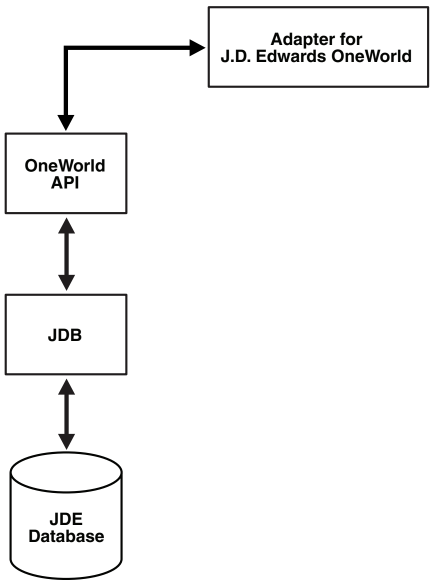
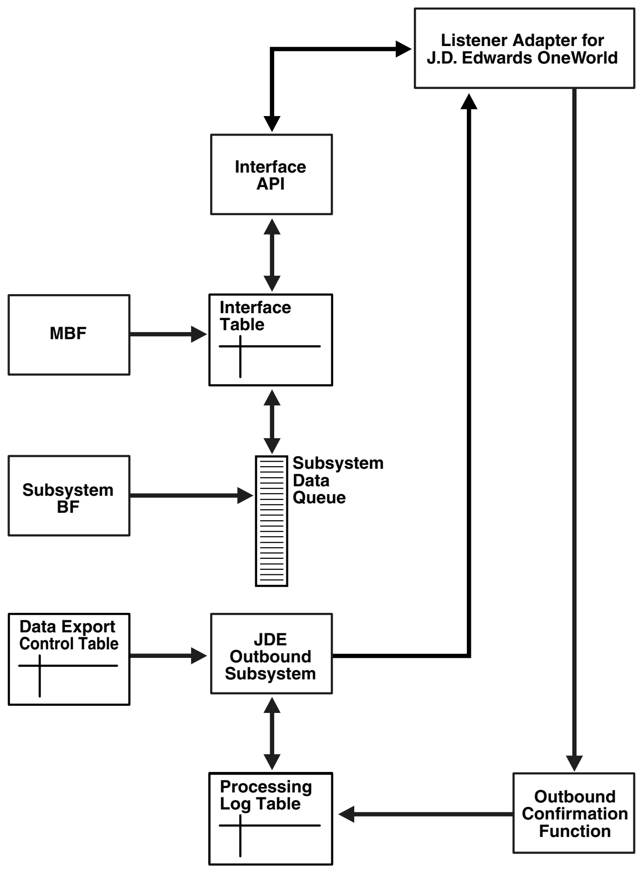


Figure 1–2 illustrates the inbound processing framework.

Figure 1–2 J.D. Edwards OneWorld Inbound Processing



In the outbound process, the event starts when a specific MBF is executed in the J.D. Edwards OneWorld environment. The MBF writes the required information for the event into the appropriate interface table and then notifies the subsystem Batch Function (BF) that an event occurred. The subsystem BF then places an entry about the event on the Subsystem Data Queue.

The J.D. Edwards OneWorld outbound subsystem retrieves the data queue entry and looks in the Data Export Control table for the external processes to notify. The J.D. Edwards OneWorld outbound subsystem then calls the Oracle Application Adapter for J.D. Edwards OneWorld listener with notification. The listener passes the notification to the generator. The generator then uses the J.D. Edwards OneWorld ThinNet API to retrieve the appropriate information from the interface table.

1.4 Adapter Architecture

Oracle Application Adapter for J.D. Edwards OneWorld uses Application Explorer with one of the following components:

- Oracle WebLogic Server Adapter Business Services Engine (BSE)
- Enterprise Connector for J2EE Connector Architecture (J2CA)

This section contains the following topics:

- Section 1.4.1, "Oracle Adapter Application Explorer (Application Explorer)"
- Section 1.4.2, "Resource Adapters"

- Section 1.4.3, "Oracle WebLogic Server Adapter Business Services Engine (BSE) Architecture"
- Section 1.4.4, "Oracale WebLogic Server Adapter Generic J2CA Architecture"
- Section 1.4.5, "Processing Business Functions"

1.4.1 Oracle Adapter Application Explorer (Application Explorer)

Application Explorer is used to configure database connections and create Web services and events. It can be configured to work in a Web services environment with BSE or with the Enterprise Connector for J2EE Connector Architecture (J2CA). When working in a J2CA environment, the connector uses the Common Client Interface (CCI) to provide fast integration services using Adapters instead of using Web services.

Both BSE and the connector for J2CA are deployed to an application server with Application Explorer and the adapters.

Application Explorer uses an explorer metaphor for browsing the J.D. Edwards system for business functions. Application Explorer enables you to create XML schemas and Web services for the associated business function.

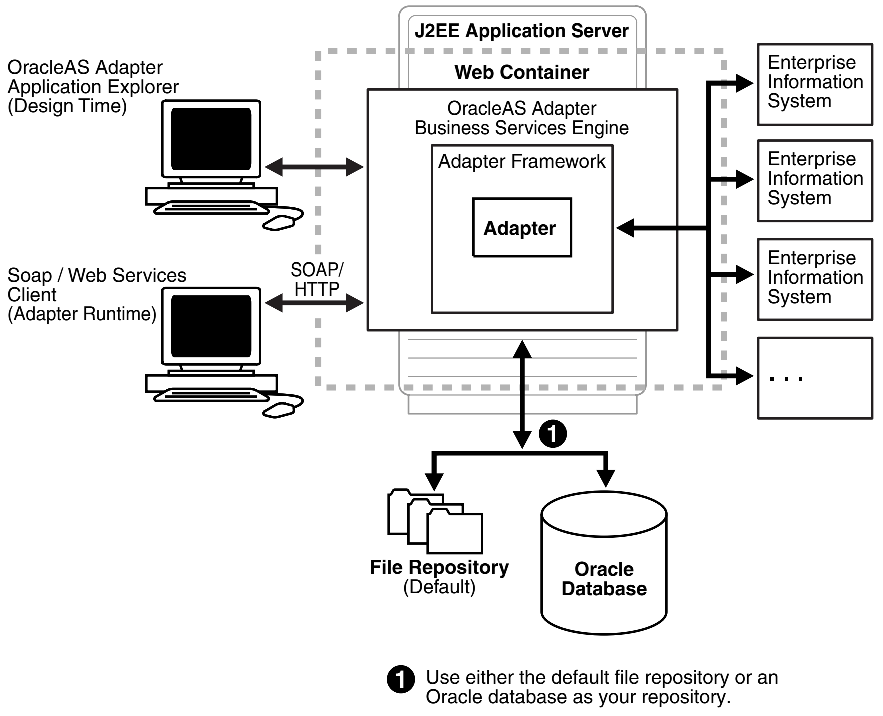
1.4.2 Resource Adapters

Oracle Application Adapter for J.D. Edwards OneWorld is a J2CA-based component also known as resource adapter. Resource adapters connect applications that were not originally designed to communicate with each other. Adapters are bidirectional, that is, they can send requests to an Enterprise Information System (EIS), and receive notification of events occurring in an EIS.

1.4.3 Oracle WebLogic Server Adapter Business Services Engine (BSE) Architecture

Figure 1–3 shows the generic architecture for the Oracle Web service adapter for packaged applications. The adapter works with BSE, as deployed to a Web container in a J2EE application server.

Figure 1–3 Oracle Adapter Business Services Engine (BSE) Architecture


J2EE Application Server
Web Container
OracleAS Adapter Business Services Engine
Adapter Framework
Adapter
OracleAS Adapter Application Explorer (Design Time)
Soap / Web Services Client (Adapter Runtime)
SOAP/ HTTP
Enterprise Information System
Enterprise Information System
Enterprise Information System
. . .
1
File Repository (Default)
Oracle Database
1 Use either the default file repository or an Oracle database as your repository.

Note: Do not use a file repository for BSE in production environments.

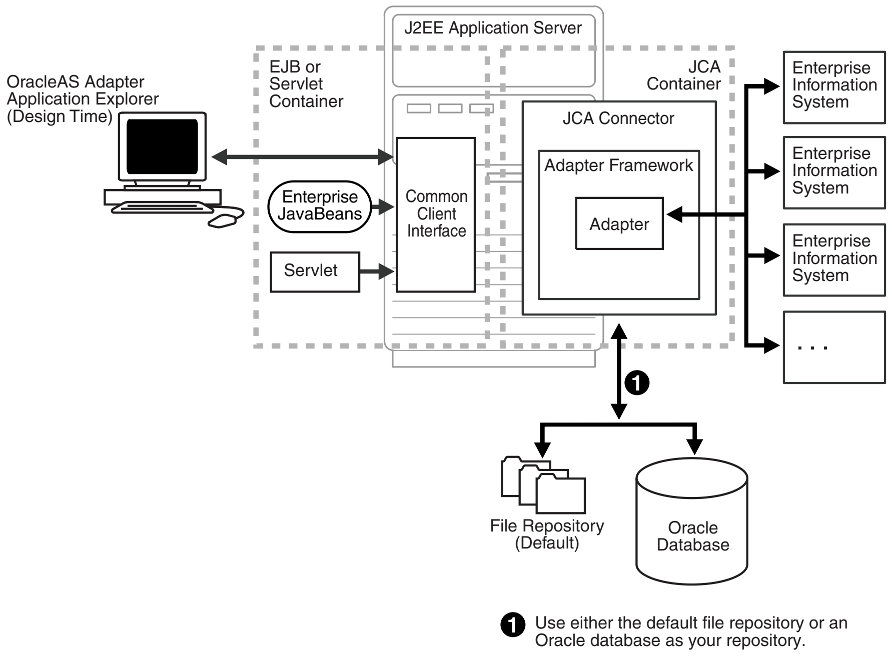
Application Explorer, a design-time tool deployed along with BSE, is used to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. Metadata created while you perform these operations are stored in the repository by BSE.

BSE uses SOAP as a protocol for receiving requests from clients, interacting with the EIS, and sending responses from the EIS back to clients.

1.4.4 Oracale WebLogic Server Adapter Generic J2CA Architecture

Figure 1–4 shows the generic architecture for Oracle J2CA adapter for packaged applications. The J2CA connector is deployed to a standard J2CA Container and serves as host container to the adapters. The connector is configured with a repository.

Figure 1–4 Oracle Adapter Generic J2CA Architecture


J2EE Application Server
OracleAS Adapter Application Explorer (Design Time)
EJB or Servlet Container
JCA Container
Enterprise JavaBeans
Servlet
Common Client Interface
JCA Connector
Adapter Framework
Adapter
Enterprise Information System
Enterprise Information System
Enterprise Information System
. . .
1
File Repository (Default)
Oracle Database
1 Use either the default file repository or an Oracle database as your repository.

Application Explorer, a design tool that works with the connector, is used to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. Metadata created during these operations is stored in the repository by the connector. The repository can be a file system or an Oracle database. It is deployed as a RAR file and has an associated deployment descriptor called `ra.xml`. You can create multiple connector factories by editing the Oracle WebLogic Server deployment descriptor `ra.xml`. For more information, see Chapter 3, "Oracle WebLogic Server Deployment and Integration".

1.4.5 Processing Business Functions

Oracle Application Adapter for J.D. Edwards OneWorld enables the processing of J.D. Edwards OneWorld business functions through the J.D. Edwards ThinNet API. Using the API eliminates the requirement of creating complex and impractical batch processes. In addition, a transport layer, such as IBM MQSeries, is not required because a listener is defined through a HTTP, TCP, or File connection.

External applications that access J.D. Edwards OneWorld through Oracle Application Adapter for J.D. Edwards OneWorld use either XML schemas or Web services to pass data between the external application and the adapter. Chapter 2, "Configuring Oracle Application Adapter for J.D. Edwards OneWorld" describes how to use Application Explorer to create XML schemas and Web services for the J.D. Edwards Master Business Functions (MBF) used with the adapter.

1.5 BSE Versus Oracle Adapter J2CA Deployment

If you are using Oracle Application Adapter for J.D. Edwards OneWorld with Oracle SOA Suite components (for example, BPEL, Mediator, BPM, or OSB), then note that:

- Only Oracle Adapter J2CA deployment supports inbound integration (event notification) with Oracle SOA Suite components.

- Oracle Adapter J2CA and BSE deployments support outbound integration (request-response service) with Oracle SOA Suite components.

The following two factors explain the differences between deploying BSE and Oracle Adapter J2CA. Understanding these factors can help in selecting a deployment option.

1. BSE has the following advantages:
   - Can be deployed in a separate instance of Oracle WebLogic Server.
   - Provides better distribution of load.
   - Conforms more closely to the Service Oriented Architecture (SOA) model for building applications.
2. Oracle Adapter J2CA does provide slightly better performance than BSE.

1.6 Sample Projects

Sample projects for the Oracle Application Adapter for J.D. Edwards OneWorld that demonstrate outbound and inbound integration scenarios using Oracle BPEL, Mediator, BPM, and OSB tools are packaged with the Application Adapters installation. The following table lists the locations of the sample projects:

Sample Project	**Location**
Outbound BPEL Process (J2CA)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\BPEL\J2CA\Outbound_Project`
Inbound BPEL Process (J2CA)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\BPEL\J2CA\Inbound_Project`
Outbound BPEL Process (BSE)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\BPEL\BSE\Outbound_Project`
Outbound Mediator Process (J2CA)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\Mediator\J2CA\Outbound_Project`
Inbound Mediator Process (J2CA)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\Mediator\J2CA\Inbound_Project`
Outbound Mediator Process (BSE)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\Mediator\BSE\Outbound_Project`
Outbound BPM Process (J2CA)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\BPM\J2CA\Outbound_Project`
Inbound BPM Process (J2CA)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\BPM\J2CA\Inbound_Project`
Outbound BPM Process (BSE)	`<`*`ADAPTER_HOME`*`>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\BPM\BSE\Outbound_Project`

Sample Project	Location
Outbound OSB sbconsole Process (J2CA)	`<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\OSB\J2CA\JDEdwards_Sample_J2CA_OSB_Outbound_Project`
Inbound OSB sbconsole Process (J2CA)	`<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\OSB\J2CA\JDEdwards_Sample_J2CA_OSB_Inbound_Project`
Outbound OSB sbsonsole Process (BSE)	`<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\OSB\BSE\JDEdwards_Sample_BSE_OSB_Outbound_Project`
Outbound OSB Jdeveloper Process (J2CA)	`<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\OSB_Jdeveloper\J2CA\JDEdwards_Sample_J2CA_OSB_Outbound_Project`
Inbound OSB Jdeveloper Process (J2CA)	`<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\OSB_Jdeveloper\J2CA\JDEdwards_Sample_J2CA_OSB_Inbound_Project`
Outbound OSB Jdeveloper Process (BSE)	`<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\OSB_Jdeveloper\BSE\JDEdwards_Sample_BSE_OSB_Outbound_Project`

1.7 Quick Start Guide

This section enables you to quickly learn the basic steps to install and configure Oracle Application Adapter for J.D. Edwards OneWorld and to use it immediately. It includes the following topics:

- Section 1.7.1, "Installation"
- Section 1.7.2, "Copying Third-Party Library Files"
- Section 1.7.3, "Configuration"
- Section 1.7.4, "WebLogic Server Deployment and Integration"
- Section 1.7.5, "Creating Configurations, Targets, and Channels in Application Explorer"
- Section 1.7.6, "Working With Service Components in the SOA Suite"
- Section 1.7.7, "Working With Oracle Service Bus"
- Section 1.7.8, "Other Features"

1.7.1 Installation

To install Oracle Application Adapter for J.D. Edwards OneWorld, download the Oracle Fusion Middleware Application Adapters installer and complete the installation for SOA/OSB.

For more information on installing the Oracle Fusion Middleware Application Adapters, see the *Oracle Fusion Middleware Application Adapters Installation Guide for Oracle WebLogic Server*.

1.7.2 Copying Third-Party Library Files

Once the adapter installation is completed, copy the required third-party library files for J.D. Edwards OneWorld to the following directories:

```
<ADAPTER_HOME>\lib

<ORACLE_HOME>\user_projects\domains\base_domain\lib
```

For more information on installing Oracle Fusion Middleware Application Adapters, see the *Oracle Fusion Middleware Application Adapters Installation Guide for Oracle WebLogic Server*.

1.7.3 Configuration

Navigate to *<ADAPTER_HOME>* and make the following changes:

1. Open *`iwafjca.rar\META-INF\ra.xml`* and add the following values under the specified config-property-name parameters, as shown in Table 1–1.

Table 1–1

Config-Property-Name	Config-Property-Value

`IWayHome`	*<ADAPTER_HOME>*
	For example:
	■ **For SOA:**
	`C:\12C_soa\soa\soa\thirdparty\ApplicationAdapters`
	■ **For OSB:**
	`C:\12c_OSB\osb\3rdparty\ApplicationAdapters`

`IWayConfig`	The name of the configuration. For example: `jca_sample`

2. Open *`ibse.war\WEB-INF\web.xml`* and add the following values under the specified param-name parameters, as shown in Table 1–2.

Table 1–2

Param-Name	Param-Value

`ibseroot`	*<ADAPTER_HOME>\ibse.war*
	For example:
	■ **For SOA:**
	`C:\12C_soa\soa\soa\thirdparty\ApplicationAdapters\ibse.war`
	■ **For OSB:**
	`C:\12c_OSB\osb\3rdparty\ApplicationAdapters\ibse.war`

Table 1–2 (Cont.)

Param-Name	Param-Value

IWay.home	*<ADAPTER_HOME>*
	For example:
	■ **For SOA:**
	`C:\12C_soa\soa\soa\thirdparty\ApplicationAdapters`
	■ **For OSB:**
	`C:\12c_OSB\osb\3rdparty\ApplicationAdapters`

Iway.config	The name of the configuration. For example: `IBSE`

Note: These steps are provided only when configuring a File repository. For more information about configuring a database repository and general configuration information, see Chapter 2, "Configuring Oracle Application Adapter for J.D. Edwards OneWorld" and Chapter 3, "Oracle WebLogic Server Deployment and Integration".

1.7.4 WebLogic Server Deployment and Integration

1. Start the WebLogic server and open the WebLogic console.
2. Deploy the adapter components (ibse.war, iwafjca.war, and iwafjca.rar files) and start the deployed adapter components.

For more information on deployment, integration, and target creation, see Chapter 3, "Oracle WebLogic Server Deployment and Integration".

1.7.5 Creating Configurations, Targets, and Channels in Application Explorer

For more information on creating configurations, targets, and channels in Application Explorer, see the following sections in this user's guide:

- Starting Application Explorer: Section 2.1, "Starting Application Explorer"
- Creating a BSE Configuration: Section 2.3.1, "Creating a Configuration for BSE"
- Creating a J2CA Configuration: Section 2.3.2, "Creating a Configuration for J2CA"
- Connecting the Created Configurations: Section 2.3.3, "Connecting to a BSE or J2CA Configuration"
- Creating and Connecting to Targets: Section 2.4, "Establishing a Connection (Target) for J.D. Edwards OneWorld"
- Creating and Testing Web Services: Section 2.7, "Creating and Testing a Web Service (BSE Configurations Only)"
- Generating WSDL Files: Section 2.6, "Generating WSDL (J2CA Configurations Only)"
- Creating and Working With Channels: Section 2.8, "Configuring an Event Adapter"

1.7.6 Working With Service Components in the SOA Suite

Oracle Application Adapter for J.D. Edwards OneWorld integrates with service components in SOA suite such as BPEL, Mediator, and BPM. Required processes are created in JDeveloper and then deployed to the SOA server.

For more information on working with BPEL, Mediator, and BPM service components, see:

- Chapter 4, "Integration With BPEL Service Components in the Oracle SOA Suite"
- Chapter 5, "Integration With Mediator Service Components in the Oracle SOA Suite"
- Chapter 6, "Integration With BPM Service Components in the Oracle SOA Suite"

1.7.7 Working With Oracle Service Bus

Oracle Application Adapter for J.D. Edwards OneWorld integrates with Oracle Service Bus (OSB) to facilitate Web service integration. Required processes are created in the Oracle Service Bus Console. The process can also be created in JDeveloper and then deployed to the SOA server.

For more information on working with OSB Console, see Chapter 7, "Configuring an Outbound and Inbound Process for Oracle Service Bus Using sbconsole".

For more information on working with OSB JDeveloper, see Chapter 8, "Configuring an Outbound and Inbound Process for Oracle Service Bus Using JDeveloper".

1.7.8 Other Features

The following is list of other features and their relevant sections in this user's guide:

- Configuring the Exception Filter: Section 9.4, "Exception Filter"
- Configuring Credential Mapping:
  - Section 9.5, "Credential Mapping for Oracle SOA Suite (BPEL, Mediator, or BPM)"
  - Section 9.6, "Credential Mapping for Oracle Service Bus (OSB)"

2 Configuring Oracle Application Adapter for J.D. Edwards OneWorld

This chapter describes how to use Oracle Adapter Application Explorer (Application Explorer) to define a target to connect to a J.D. Edwards OneWorld system, view system objects, and create XML schemas and Web services. This chapter also explains how to configure an event adapter.

This chapter contains the following sections:

- Section 2.1, "Starting Application Explorer"
- Section 2.2, "Configuring Repository Settings"
- Section 2.3, "Creating a Repository Configuration"
- Section 2.4, "Establishing a Connection (Target) for J.D. Edwards OneWorld"
- Section 2.5, "Creating an XML Schema"
- Section 2.6, "Generating WSDL (J2CA Configurations Only)"
- Section 2.7, "Creating and Testing a Web Service (BSE Configurations Only)"
- Section 2.8, "Configuring an Event Adapter"
- Section 2.9, "Runtime Overview"
- Section 2.10, "Modifying the JDE.INI File for Outbound and Inbound Processing"

2.1 Starting Application Explorer

To start Application Explorer:

1. Ensure that Oracle WebLogic Server is started, which is where Application Explorer is deployed.
2. Open the command prompt.
3. Navigate to the following directory:

   ```
   <ADAPTER_HOME>\user_projects\domains\base_domain\bin
   ```

4. Execute *`setDomainEnv.cmd`* (Windows) or *`. ./setDomainEnv.sh`* (UNIX/Linux).

   This command sets the class path and other environment variables for Application Explorer in the Oracle WebLogic Server environment. In addition, it allows Application Explorer to access the Oracle WebLogic Server APIs to publish WSDL files to the Oracle Service Bus (OSB) Console.

5. Do not close the command prompt.
6. Navigate to the following directory:

   ```
   <ADAPTER_HOME>\tools\iwae\bin
   ```

7. Execute `ae.bat` (Windows) or `iwae.sh` (UNIX/Linux) to start Application Explorer.

Application Explorer starts. You are ready to define new targets to your J.D. Edwards OneWorld system.

Note: Before you run the **iwae.sh** file on UNIX or Linux platforms, the permissions must be changed. For example:

```
chmod +x  iwae.sh
```

2.2 Configuring Repository Settings

A repository holds information about configuration details, adapter targets, channels, and other configuration information. For more information on how to configure BSE and J2CA repository settings, see the *Oracle Fusion Middleware Application Adapters Installation Guide for Oracle WebLogic Server* (Section 2.7.4 "Configuring the Oracle Database Repository").

2.3 Creating a Repository Configuration

Before you use Application Explorer with Oracle Application Adapter for J.D. Edwards OneWorld, you must create a repository configuration. You can create two kinds of repository configurations, Web services and J2CA, depending on the container to which the adapter is deployed. During design time, the repository is used to store metadata created when using Application Explorer to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. The information in the repository is also referenced at run-time.

This section contains the following topics:

- Section 2.3.1, "Creating a Configuration for BSE"
- Section 2.3.2, "Creating a Configuration for J2CA"
- Section 2.3.3, "Connecting to a BSE or J2CA Configuration"

Web services and BSE refer to the same type of deployment. For more information, see "Adapter Features" on page 1-1.

2.3.1 Creating a Configuration for BSE

To create a repository configuration for BSE using Application Explorer, you must first define a new configuration.

This section contains the following topic:

- Section 2.3.1.1, "Defining a New Configuration for BSE"

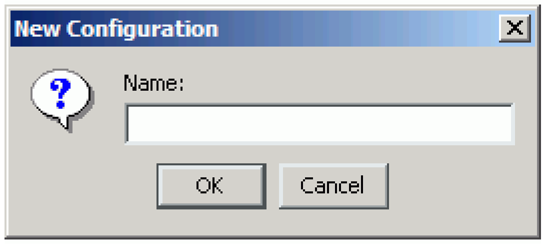
2.3.1.1 Defining a New Configuration for BSE

To define a new configuration for BSE:

1. Right-click **Configurations** and select **New**.

The New Configuration dialog is displayed, as shown in Figure 2–1.

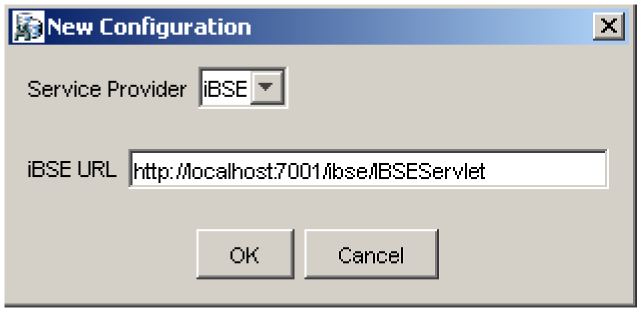
Figure 2–1 New Configuration Dialog



2. Enter a name for the new configuration (for example, myConfig) and click **OK**.

   The New Configuration dialog is displayed, as shown in Figure 2–2.

Figure 2–2 New Configuration Dialog



Service Provider iBSE

iBSE URL http://localhost:7001/ibse/IBSEServlet

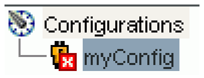
3. From the Service Provider list, select **iBSE**.
4. In the **iBSE URL** field, accept the default URL or replace it with a different URL using the following format:

   ```
   http://host name:port/ibse/IBSEServlet
   ```

   Where *host name* is the system where your Oracle WebLogic Server resides and *port* is the HTTP port for a managed Oracle WebLogic Server (for example, soa_server1).
5. Click **OK**.

   A node representing the new configuration appears beneath the root Configurations node, as shown in Figure 2–3.

Figure 2–3 Configurations Node



2.3.2 Creating a Configuration for J2CA

To create a configuration for J2CA using Application Explorer, you must first define a new configuration.

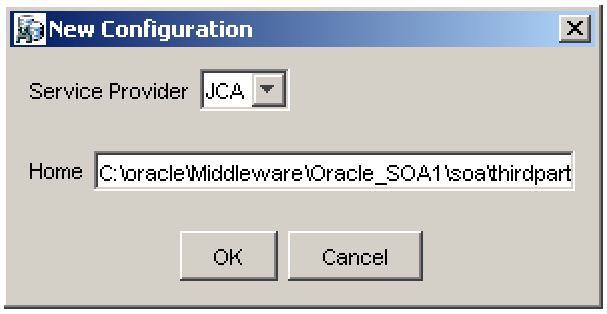
To define a new configuration for J2CA:

1. Right-click **Configurations** and select **New**.

The New Configuration dialog is displayed.

2. Enter a name for the new configuration (for example, myConfig) and click **OK**, as shown in Figure 2–4.

Figure 2–4 New Configuration Dialog



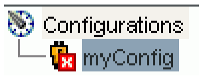
Service Provider: JCA

Home: C:\oracle\Middleware\Oracle_SOA1\soa\thirdpart

3. From the **Service Provider** list, select **JCA**.
4. Click **OK**.

   A node representing the new configuration appears beneath the root Configurations node, as shown in Figure 2–5.

Figure 2–5 Configurations Node



The Oracle Adapter J2CA configuration folder is stored in a location based on your adapter installation:

```
<ADAPTER_HOME>\config\configuration_name
```

The *`configuration_name`* is the name of the configuration you created (for example, SampleConfig).

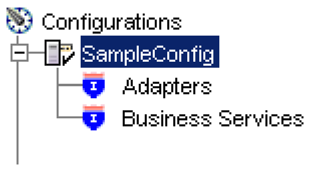
2.3.3 Connecting to a BSE or J2CA Configuration

To connect to a new configuration:

1. Right-click the configuration to which you want to connect, for example, SampleConfig.
2. Select **Connect**.

   Nodes appear for Adapters, Events, and Business Services (also known as Web services). The Business Services node is only available for BSE configurations. If you are connected to a J2CA configuration, then the Business Services node is not shown. As shown in Figure 2–6, the following is an example of a BSE configuration named SampleConfig:

Figure 2–6 The New SampleConfig Configuration That Appears Under The Configurations Node



- Use the **Adapters** node to create inbound interaction with J.D. Edwards OneWorld. For example, you use the J.D. Edwards OneWorld node in the Adapters node to configure a service that updates J.D. Edwards OneWorld.
- Use the **Events** node (available for J2CA configurations only) to configure listeners that listen for events in J.D. Edwards OneWorld.
- Use the **Business Services** node (available for BSE configurations only) to test Web services created in the Adapters node. You can also control security settings for the Web services by using the security features of the Business Services node.

You can now define new targets to J.D. Edwards OneWorld.

2.4 Establishing a Connection (Target) for J.D. Edwards OneWorld

Part of the application definition includes adding a target for the adapter. Setting up the target in Application Explorer requires information which is specific to the target.

This section contains the following topic:

- Section 2.4.1, "Defining a Target to J.D. Edwards OneWorld"

To browse the available Master Business Functions (MBF), you must first define a target to the system you use. After you define the target, it is automatically saved. You must connect to the system every time you start Application Explorer or after you disconnect.

When you launch Application Explorer, the left pane displays (as nodes) the application systems supported by Application Explorer, based on the adapters that are installed.

2.4.1 Defining a Target to J.D. Edwards OneWorld

This section contains the following topics:

- Section 2.4.1.1, "Connecting to a Defined J.D. Edwards OneWorld Target"
- Section 2.4.1.2, "Disconnecting from J.D. Edwards OneWorld"
- Section 2.4.1.3, "Editing a Target"
- Section 2.4.1.4, "Deleting a Target"

To connect to an application system for the first time, you must define a new target.

When you define a target, you must restart the Oracle WebLogic Server to update the repository for run time purposes.

Note: Before you create a new target, you must obtain the required library files for your J.D. Edwards OneWorld system and copy them to the appropriate location where the Oracle Application Adapter for J.D. Edwards OneWorld is deployed. For more information, see the *Oracle Fusion Middleware Application Adapters Installation Guide for Oracle WebLogic Server*.

To define a target:

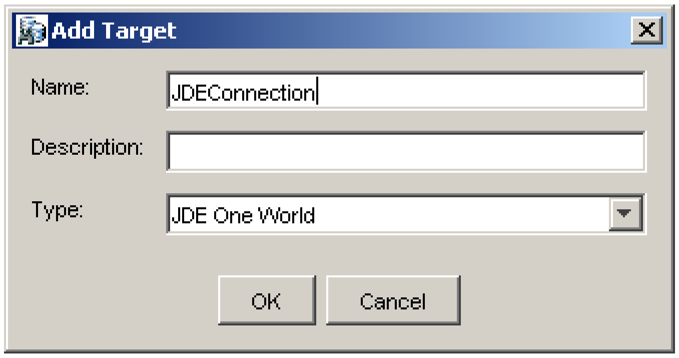
1. In the left pane, expand the **Adapters** node.

   The applications systems supported by Application Explorer appear as nodes based on the adapters that are installed.

2. Right-click the **JDEdwards** node and select **Add Target**.

   The Add Target dialog is displayed, as shown in Figure 2–7.

Figure 2–7 Add Target Dialog



Name: JDEConnection
Description:
Type: JDE One World

   Perform the following steps:

   a. In the **Name** field, enter a descriptive name, for example, JDEConnection.

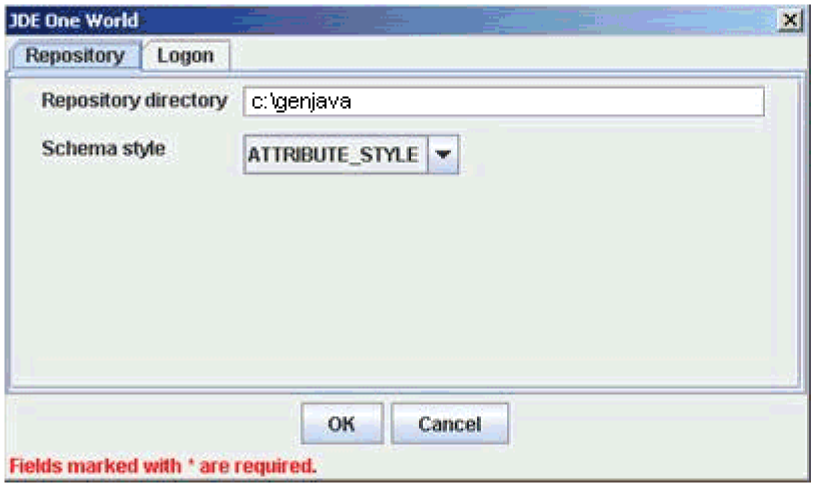
   b. In the **Description** field, enter a description for the target (optional).

   c. From the **Type** list, select **JDE One World**.

3. Click **OK**.

   The JDE One World dialog appears, as shown in Figure 2–8.

Figure 2–8 JDE One World Dialog



Repository directory: c:\genjava

Schema style: ATTRIBUTE_STYLE

Fields marked with * are required.

a. In the **Repository** tab, enter the path to the GenJava repository in the Repository directory field.

   This is the location of the Java wrappers for accessing the J.D. Edwards OneWorld business functions, which are created by the GenJava development tool. Please note that this is a prerequisite step, which must be performed before a new target is created using Application Explorer.

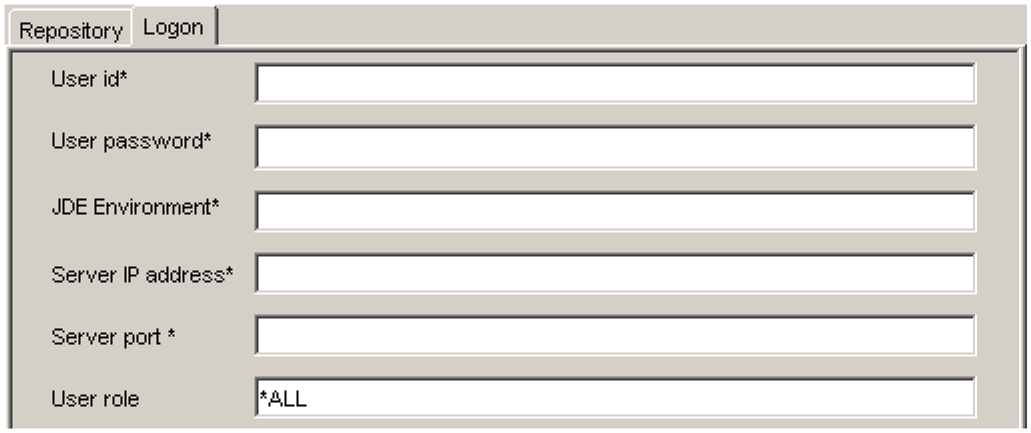
   Note: Generating schemas requires the GenJava repository. For more comprehensive information on building the J.D. Edwards OneWorld Master Business Function repository, see the *J.D. Edwards Interoperability Guide for OneWorld Xe.* For information on how to use the GenJava program, see Using the GenJava Development Tool (Outbound Processing) in Appendix A, "Configuring J.D. Edwards OneWorld for Outbound and Inbound Processing".

   ---

b. From the **Schema style** list, select **ELEMENT_STYLE** or **ATTRIBUTE_STYLE**.

c. Click the **Logon** tab and enter the appropriate information for your target type based on the information in the following table. Fields marked with an asterisk are required, as shown in Figure 2–9.

Figure 2–9 Logon Tab



User id*	
User password*	
JDE Environment*	
Server IP address*	
Server port *	
User role	*ALL

Parameter	Description
User id*	A valid user ID for J.D. Edwards OneWorld.
User password*	The password associated with the user ID.
JDE environment*	The J.D. Edwards OneWorld environment, for example, DU7333. For more information about this parameter, see your J.D. Edwards OneWorld documentation or ask your J.D. Edwards OneWorld system administrator.
Server IP address*	The name of the server on which J.D. Edwards OneWorld is running. This can be the name of the server, for example, JDEOW, or its IP address.
Server Port*	The port number on which the server is listening, for example, 6009.
User role	Specify ***ALL**.

4. Click **OK**.

   The new target, JDEConnection, appears under the JDEdwards node.

2.4.1.1 Connecting to a Defined J.D. Edwards OneWorld Target

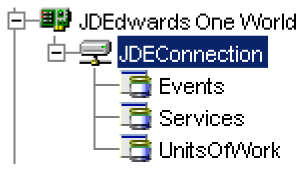
To connect to a target:

1. Expand the **Service Adapters** node.
2. Expand the **JDEdwards** node.
3. Click the target name (for example, JDEConnection) under the JDEdwards node.
4. Click the **Logon** tab on the right.

   The Logon tab displays the values you entered for connection parameters.
5. Verify your connection parameters.
6. Right-click the target name and select **Connect**.

   The x icon disappears, indicating that the node is connected, as shown in Figure 2–10.

Figure 2–10 JDEdwards Target Node



2.4.1.2 Disconnecting from J.D. Edwards OneWorld

To disconnect from a target:

1. Expand the **Adapters** node.
2. Expand the **JDEdwards** node.
3. Right-click the target to which you are connected (for example, JDEConnection), and select **Disconnect**.

   The x icon appears, indicating that the node is disconnected.

2.4.1.3 Editing a Target

To edit a target:

1. In the left pane, ensure that the target you want to edit is disconnected.
2. Right-click the target and select **Edit**.

   A window is displayed that enables you to edit the existing connection parameters.
3. Modify the target information.
4. Click **OK**.

   When you edit a target, you must restart the Oracle WebLogic Server to update the repository for run time purposes.

2.4.1.4 Deleting a Target

You can delete a target, rather than just disconnecting and closing it. When you delete the target, the node disappears from the list of J.D. Edwards OneWorld targets in the left pane of the explorer.

When you delete a connection, you must restart the Oracle WebLogic Server to update the repository for run time purposes.

To delete a target:

1. Expand the **Adapters** node.
2. Expand the **JDEdwards** node.
3. Right-click the target to which you are connected (for example, JDEConnection), and select **Delete**.

   The node disappears from the list of available connections.

For information on how to view application system objects, see *J.D .Edwards Interoperability Guide Release OneWorld XE*.

2.5 Creating an XML Schema

To execute an MBF, the adapter must receive a request document through the J.D. Edwards OneWorld ThinNet API. The agent processes the request and sends an XML response document indicating the result. Application Explorer creates both the XML request schema and the XML response schema.

This section contains the following topic:

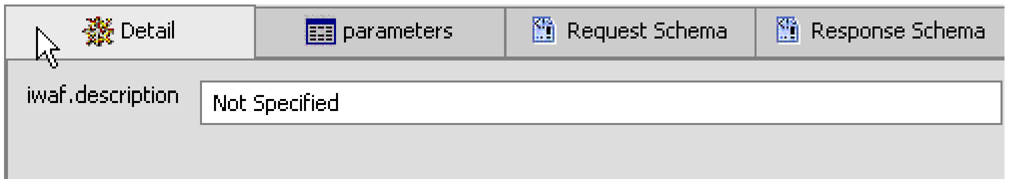
- Section 2.5.1, "Creating a Request and a Response Schema"

2.5.1 Creating a Request and a Response Schema

The following procedure explains how to create request and response schemas for a J.D. Edwards OneWorld business function. Application Explorer enables you to create XML schemas for this function.

1. Connect to a J.D. Edwards OneWorld target as described in "Connecting to a Defined J.D. Edwards OneWorld Target" on page 2-8.
2. Expand the **Services** node.
3. Expand the node of the MBF for which you want to create the schema.
4. Expand and then select the node beneath the MBF, as shown in Figure 2–11.

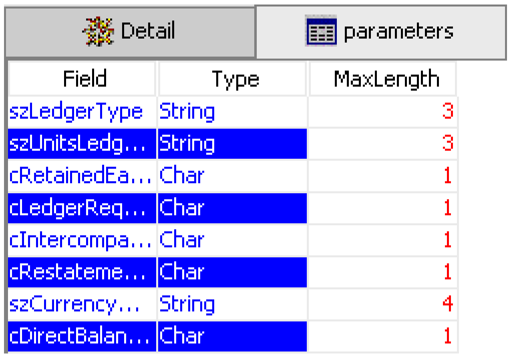
Figure 2–11 Detail Tab



iwaf.description: Not Specified

5. Click the **parameters** tab to view the parameter information Figure 2–12.

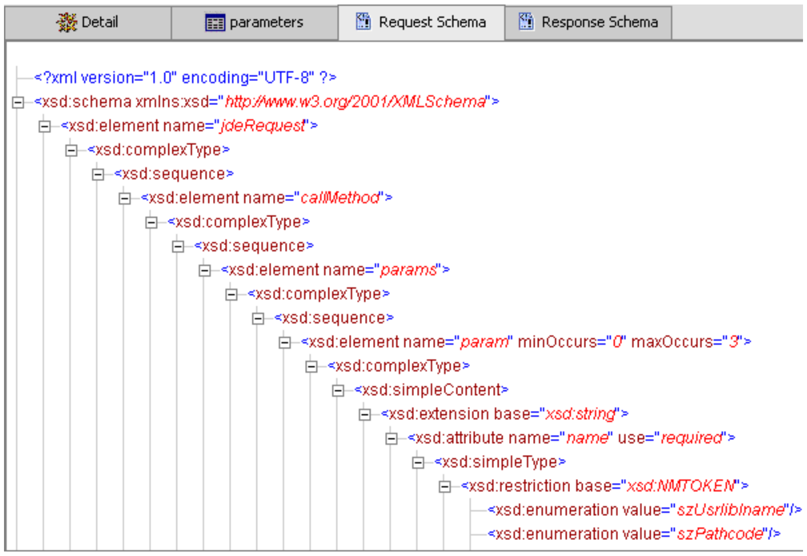
Figure 2–12 Parameters Tab



Field	Type	MaxLength
szLedgerType	String	3
szUnitsLedg...	String	3
cRetainedEa...	Char	1
cLedgerReq...	Char	1
cIntercompa...	Char	1
cRestateme...	Char	1
szCurrency...	String	4
cDirectBalan...	Char	1

6. Click **Request Schema** to view the request schema information, as shown in Figure 2–13.

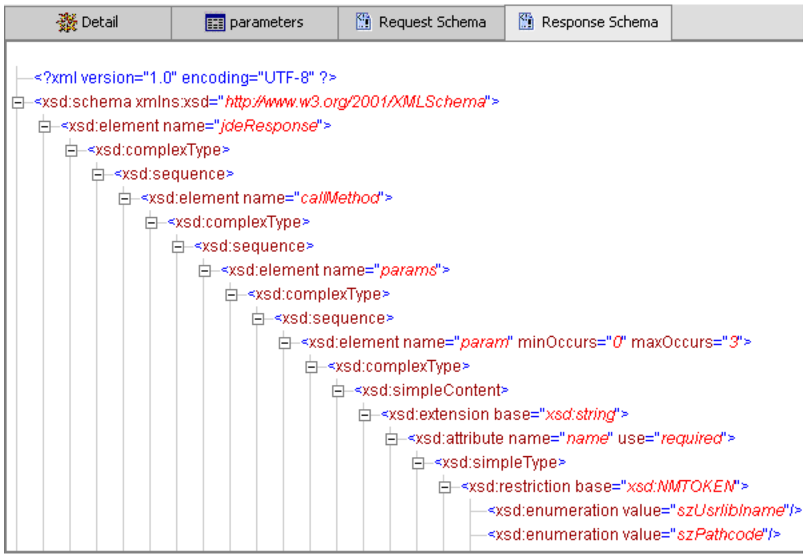
Figure 2–13 Request Schema Tab



```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="jdeRequest">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="callMethod">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="params">
                <xsd:complexType>
                  <xsd:sequence>
                    <xsd:element name="param" minOccurs="0" maxOccurs="3">
                      <xsd:complexType>
                        <xsd:simpleContent>
                          <xsd:extension base="xsd:string">
                            <xsd:attribute name="name" use="required">
                              <xsd:simpleType>
                                <xsd:restriction base="xsd:NMTOKEN">
                                  <xsd:enumeration value="szUsrlibIname"/>
                                  <xsd:enumeration value="szPathcode"/>
```

7. Click **Response Schema** to view the response schema information, as shown in Figure 2–14.

Figure 2–14 Response Schema Tab



```
<?xml version="1.0" encoding="UTF-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="jdeResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="callMethod">
          <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="params">
                <xsd:complexType>
                  <xsd:sequence>
                    <xsd:element name="param" minOccurs="0" maxOccurs="3">
                      <xsd:complexType>
                        <xsd:simpleContent>
                          <xsd:extension base="xsd:string">
                            <xsd:attribute name="name" use="required">
                              <xsd:simpleType>
                                <xsd:restriction base="xsd:NMTOKEN">
                                  <xsd:enumeration value="szUsrlibIname"/>
                                  <xsd:enumeration value="szPathcode"/>
```

2.6 Generating WSDL (J2CA Configurations Only)

The procedure for generating WSDL (Web Service Definition Language) for request-response (outbound) services differs from that of generating WSDL for event notification (inbound) J2CA services of the adapter.

This section contains the following topic:

- Section 2.6.1, "Generating a WSDL for Outbound Interaction"

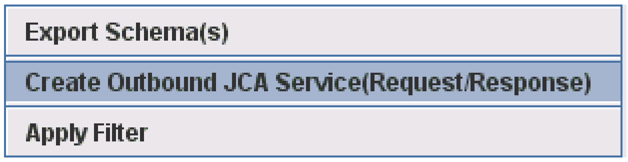
2.6.1 Generating a WSDL for Outbound Interaction

To generate a WSDL file for request-response service:

1. Start Application Explorer and connect to a defined J.D. Edwards OneWorld target.
2. Expand **Services**, **CALLBSFN**, and then **Addressbook**. Select **GetEffectiveAddress**.
3. Right-click GetEffectiveAddress.

   The following menu is displayed, as shown in Figure 2–15.

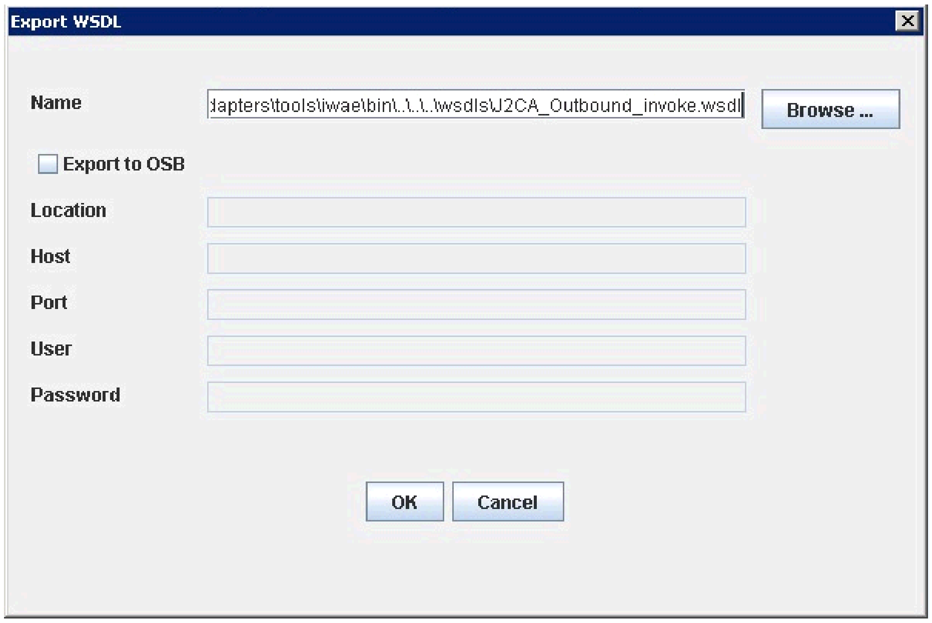
Figure 2–15 Create Outbound JCA Service (Request/Response) Option



4. Select **Create Outbound JCA Service (Request/Response)**.

   The Export WSDL dialog is displayed, as shown in Figure 2–16.

Figure 2–16 Export WSDL Dialog


Export WSDL

Name: dapters\tools\iwae\bin\..\..\..\wsdls\J2CA_Outbound_invoke.wsdl

5. Accept the default name for the file.

   The **.wsdl** file extension is added automatically. By default, the names of WSDL files generated for request-response services end with `_invoke`, while those generated for event notification end with `_receive`.
6. Click **OK**.

   The WSDL file is saved in the specified location.

2.7 Creating and Testing a Web Service (BSE Configurations Only)

You can generate a Web service (also known as a **business service**) using Application Explorer. You can explore the business function repository and generate Web services for the functions you want to use with the adapter.

This section contains the following topics:

- Section 2.7.1, "Creating a Web Service"
- Section 2.7.2, "Testing a Web Serice"
- Section 2.7.3, "Identity Propagation"

The following procedure uses an example called BusinessUnitExistenceCheck.

Note: In a J2EE Connector Architecture (J2CA) implementation, Web services are not available. When the adapters are deployed to use J2CA, the Common Client Interface (CCI) provides integration services.

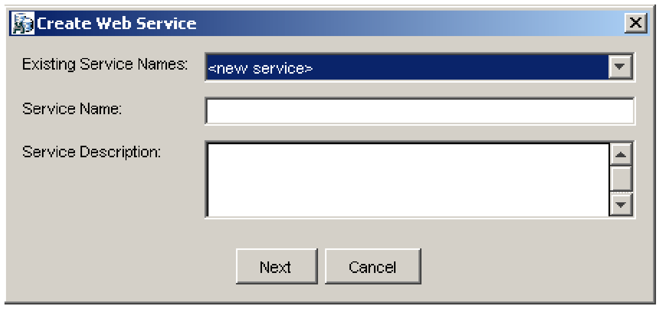
2.7.1 Creating a Web Service

To create a Web service for a business function:

1. Expand the **JDEdwards** node and then the **Services** node.
2. Expand **CALLBSFN** and then **Addressbook**.
3. Right-click **GetEffectiveAddress** and select **Create Web Service**.

   The Create Web Service dialog is displayed, as shown in Figure 2–17.

Figure 2–17 Create Web Service Dialog



   You can add the business function as a method for a new Web service or as a method for an existing one.

   a. From the **Existing Service Names** list, select either **<new service>** or an existing service.

   b. In the **Service Name** field, specify a service name if you are creating a new service. This name identifies the Web service in the list of services under the Business Services node.

   c. Enter a description for the service (optional).

4. Click **Next**.

   Perform the following steps:

   a. In the **License Name** field, select one or more license codes to assign to the Web service.
   b. In the **Method Name** field, leave the default method name.
   c. In the **Description** field, enter a brief description of the method (optional).
   d. In the **DTD Directory** field, specify a location where the Web service are saved. If you want to select a location different than the default, then click **Browse** and navigate to the desired location.

5. Click **OK**.

   Application Explorer switches the view to the **Business Services** node, and the new Web service appears in the left pane.

6. Right-click the new Web service and select **Save WSDL** from the menu.

   The Save dialog is displayed.

7. Provide a name for the WSDL file and a location to save the WSDL file on your file system.
8. Click **Save**.

2.7.2 Testing a Web Serice

After a Web service is created, you can test it to ensure it functions properly. A test tool is provided for testing the Web service.

To test a Web service:

1. Click the **Business Services** node to access your Web services.
2. Expand the **Services** node.
3. Select the name of the business service you want to test.

   The business service name appears as a link in the right pane.

4. In the right pane, click the named business services link.

   The test option appears in the right pane. If you are testing a Web service that requires XML input, then an input field is displayed.

5. Enter the appropriate input.
6. Click **Invoke**.

   Application Explorer displays the results.

2.7.3 Identity Propagation

If you test or execute a Web service using a third party XML editor, for example XMLSPY, then the user name and password values that you specify in the SOAP header must be valid and are used to connect to J.D. Edwards OneWorld. The user name and password values that you provided for J.D. Edwards OneWorld during target creation using Application Explorer are overwritten for this Web service request. The following is a sample SOAP header that is included in the WSDL file for a Web service:

```
<SOAP-ENV:Header>
  <m:ibsinfo xmlns:m="urn:schemas-iwaysoftware-com:iwse">
    <m:service>String</m:service>
    <m:method>String</m:method>
    <m:license>String</m:license>
    <m:disposition>String</m:disposition>
    <m:Username>String</m:Username>
    <m:Password>String</m:Password>
    <m:language>String</m:language>
  </m:ibsinfo>
</SOAP-ENV:Header>
```

You can remove the `<m:disposition>` and `<m:language>` tags from the SOAP header, since they are not required.

2.8 Configuring an Event Adapter

Events are generated by activity in a database or in an application system. You can use events to trigger an action in your application. For example, an update to a database can reflect an update to customer information. If your application must perform when this happens, then your application is a consumer of this event.

This section contains the following topics:

- Section 2.8.1, "Creating and Editing a Channel"
- Section 2.8.2, "The J.D. Edwards OneWorld Event Listener"
- Section 2.8.3, "Configuring the J.D. Edwards OneWorld Event Listener"

After you create a connection to your application system, you can add events using Application Explorer. To create an event, you must create a channel.

Note: If you are using a J2CA configuration, then you must create a new channel for every event object and select this channel when you generate WSDL. Creating a channel is required for J2CA configurations only.

A **channel** represents configured connections to particular instances of back-end systems. For more information, see "Creating and Editing a Channel" on page 2-15.

2.8.1 Creating and Editing a Channel

The following section describes how to create a channel for your event and contains the following topics:

- Section 2.8.1.1, "Creating an HTTP Channel"
- Section 2.8.1.2, "Creating a TCP Channel"
- Section 2.8.1.3, "Creating a File Channel"
- Section 2.8.1.4, "Editing a Channel"
- Section 2.8.1.5, "Deleting a Channel"

When you create, modify, or delete a channel, you must restart the Oracle WebLogic Server to recognize the change and update the repository for run time purposes. After successfully creating the channel and inbound WSDL file, close Application Explorer before you restart the application server.

Note: If you are planning to integrate Oracle Application Adapter for J.D. Edwards OneWorld with BPM, BPEL, Mediator, or OSB inbound process components, then do not start the channel. The channel is managed by the run-time server after the BPM, BPEL, Mediator, or OSB process component is deployed. If you start the channel from Application Explorer for testing and debugging purposes, then stop it before run-time (when working with BPM, BPEL, Mediator, or OSB process components).

Three channel types are available:

- HTTP
- TCP
- File

Note: Channels can be configured only on the system where the Oracle Application Adapter for J.D. Edwards OneWorld is installed.

2.8.1.1 Creating an HTTP Channel

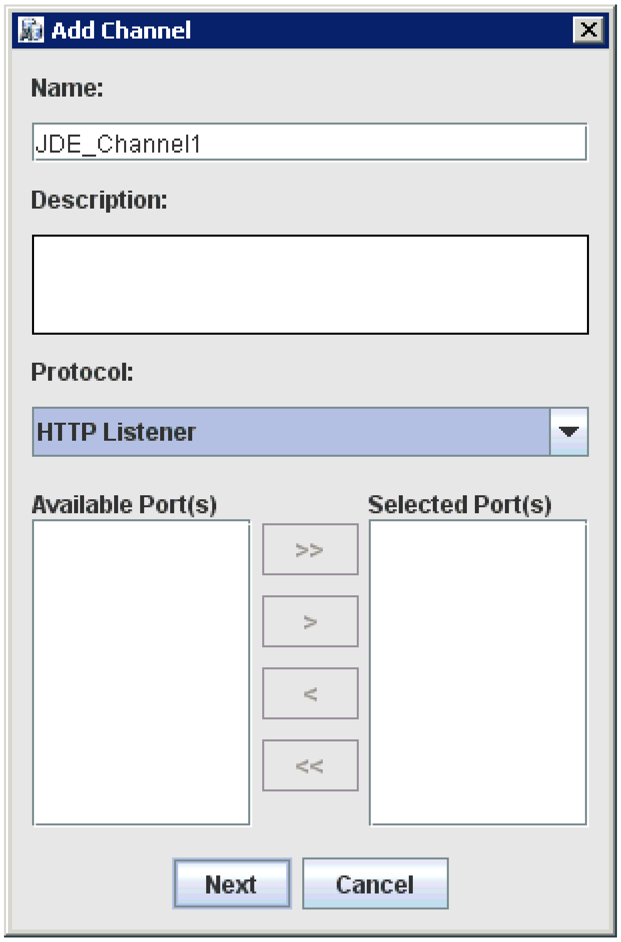
To create an HTTP Channel:

1. Click the **Events** node.
2. Expand the **JDEdwards** node.

   The ports and channels nodes appear in the left pane.
3. Right-click **Channels** and select **Add Channel**.

   The Add Channel dialog is displayed, as shown in Figure 2–18.

Figure 2–18 Add Channel Dialog



JDE_Channel1

HTTP Listener

Provide the following information:

a. Enter a name for the channel, for example, **JDE_Channel1**.

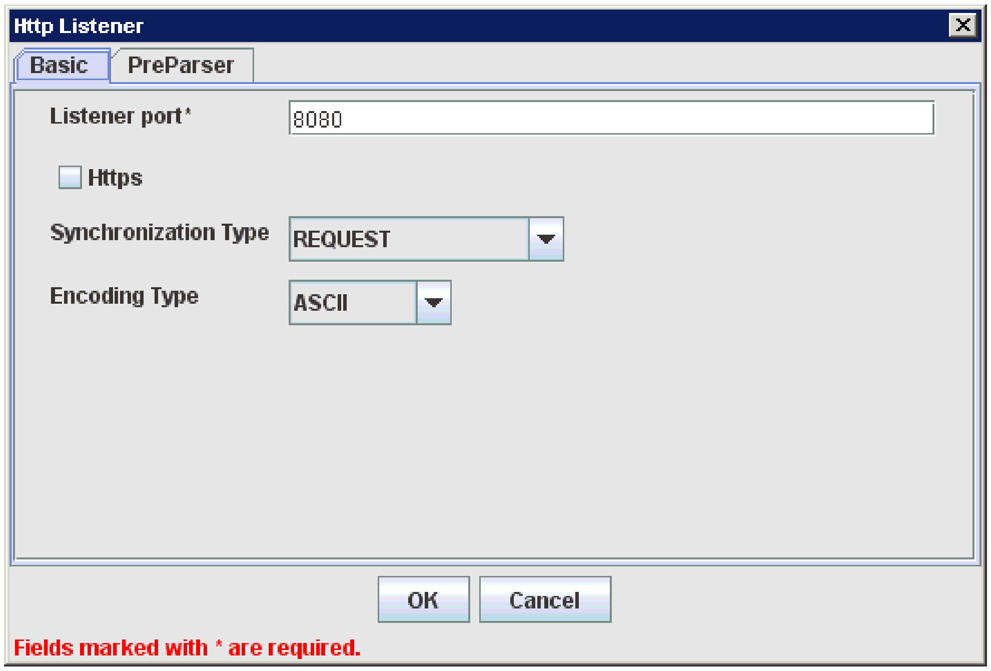
b. Enter a brief description.

c. From the **Protocol** list, select **HTTP Listener**.

4. Click **Next**.

The Http Listener dialog is displayed, as shown in Figure 2–19.

Figure 2–19 Http Listener Dialog Basic Tab



Listener port* 8080

Https

Synchronization Type REQUEST

Encoding Type ASCII

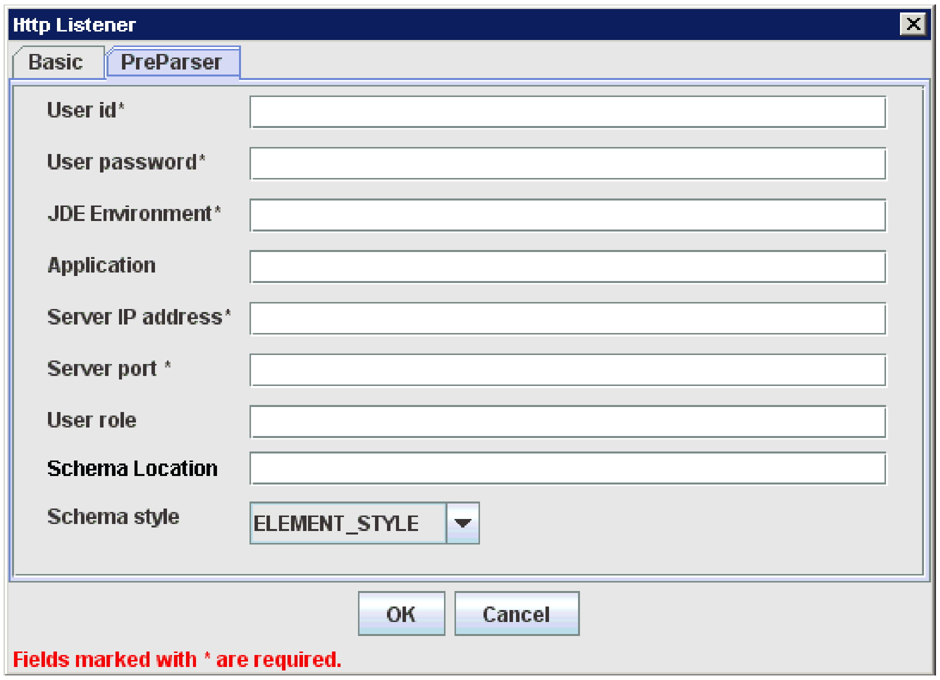
Fields marked with * are required.

5. Enter the system information as specified in the following table:

Parameter	Description
Listener port	Port on which to listen for J.D. Edwards OneWorld event data.
Https	For a secure HTTP connection, select the **Https** check box. This option is currently not supported.
Synchronization Type	Choose from the following synchronization options: ▪ REQUEST_RESPONSE ▪ REQUEST_ACK **Important:** The J.D. Edwards OneWorld channel does not work if the synchronization type is set to REQUEST.
Encoding Type	Choose an encoding type to be used from the list. By default, ASCII is selected.

6. Click the **PreParser** tab, as shown in Figure 2–20.

Figure 2–20 Http Listener Dialog Preparser Tab


Http Listener

Basic | PreParser

User id*	
User password*	
JDE Environment*	
Application	
Server IP address*	
Server port *	
User role	
Schema Location	
Schema style	ELEMENT_STYLE

Fields marked with * are required.

7. Enter the system information as specified in the following table:

Parameter	Description
User id	A valid user ID for J.D. Edwards OneWorld.
User password	The password associated with the J.D. Edwards OneWorld user ID.
JDE Environment	The J.D. Edwards OneWorld environment, for example, DU7333. For more information about this parameter, see your J.D. Edwards OneWorld documentation or ask your J.D. Edwards OneWorld system administrator.
Application	The application that is defined in the J.D. Edwards OneWorld environment.
Server IP address	The name of the server on which J.D. Edwards OneWorld is running. This can be the name of the server, for example, JDEOW, or its IP address.
Server port	The port number on which the server is listening, for example, 6009.
User role	Specify ***ALL**.
Schema Location	The location of the XML schema (.xsd file) that was generated from the event output. For example: `<ADAPTER_HOME>\config\configuration_name\schemas\JDEdwards\target_name\jde-schema.xsd` For more information, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.
Schema Style	Choose from one of the following options: ■ ELEMENT_STYLE (default) ■ ATTRIBUTE_STYLE

8. Click **OK**.

A summary pane is displayed, providing the channel description, channel status, and available ports. All the information is associated with the channel you created.

The channel appears under the channels node in the left pane.

An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.

9. Right-click the channel and select **Start**.

   The channel you created becomes active. The X over the icon in the left pane disappears.

10. To stop the channel, right-click the channel and select **Stop**.

2.8.1.2 Creating a TCP Channel

To create a TCP Channel:

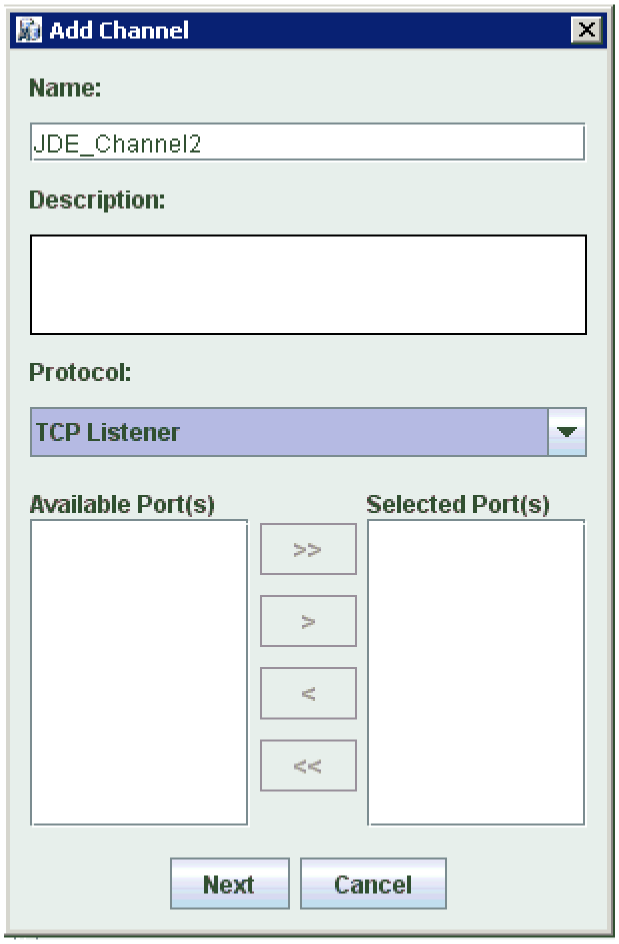
1. Click the **Events** node.
2. Expand the **JDEdwards** node.

   The ports and channels nodes appear in the left pane.

3. Right-click **Channels** and select **Add Channel**.

   The Add Channel dialog is displayed, as shown in Figure 2–21.

Figure 2–21 Add Channel Dialog



Name: JDE_Channel2

Protocol: TCP Listener

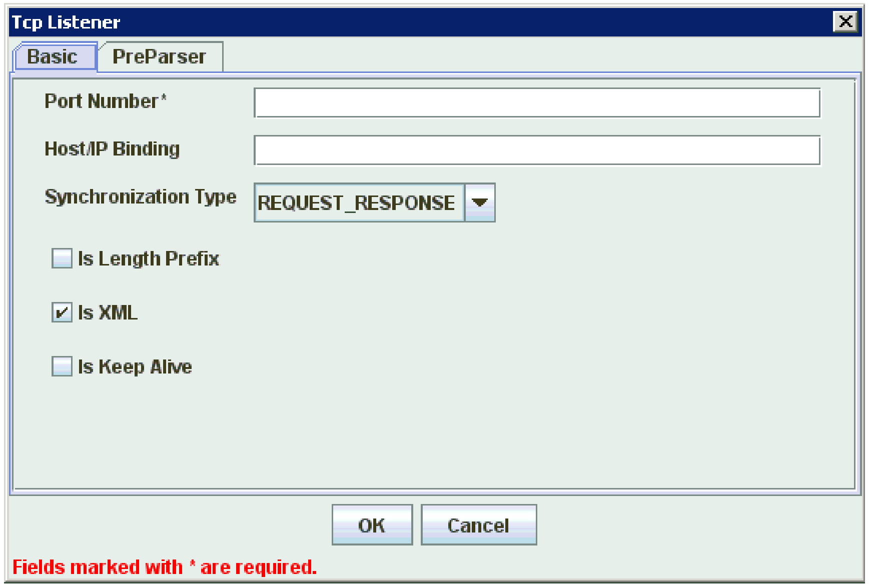
Provide the following information:

   a. Enter a name for the channel, for example, **JDE_Channel2**.
   b. Enter a brief description.
   c. From the **Protocol** list, select **TCP Listener**.

4. Click **Next**.

   The Tcp Listener dialog is displayed, as shown in Figure 2–22.

Figure 2–22 Tcp Listener Dialog Basic Tab



Port Number*

Host/IP Binding

Synchronization Type: REQUEST_RESPONSE

☐ Is Length Prefix

☑ Is XML

☐ Is Keep Alive

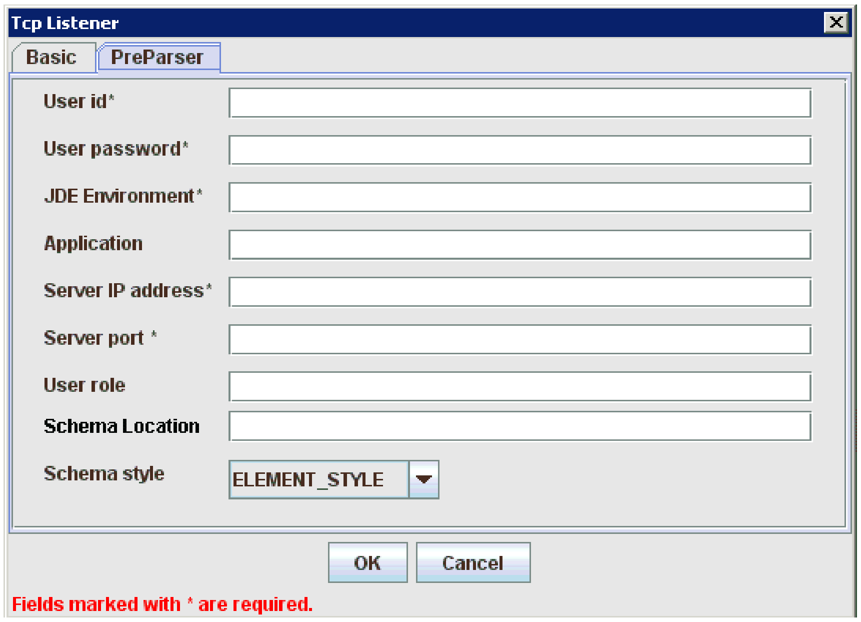
Fields marked with * are required.

5. Enter the system information as specified in the following table:

Parameter	Description
Port Number	Port on which the Host database is listening.
Host/IP Binding	Name or URL of the system where the database resides.
Synchronization Type	Choose from the following synchronization options: ■ REQUEST_RESPONSE ■ REQUEST_ACK **Important:** The J.D. Edwards OneWorld channel does not work if the synchronization type is set to REQUEST.
Is Length Prefix	For J.D. Edwards OneWorld events that send data back that is not in XML format. The TCP/IP event application must prefix the data with a 4-byte binary length field when writing the data to the TCP/IP port.
Is XML	For J.D. Edwards OneWorld events that send data back in XML format. No preparser is required.
Is Keep Alive	Maintains continuous communication between the event transaction and the channel.

6. Click the **PreParser** tab, as shown in Figure 2–23.

Figure 2–23 Tcp Listener Dialog Preparser Tab



User id*

User password*

JDE Environment*

Application

Server IP address*

Server port *

User role

Schema Location

Schema style: ELEMENT_STYLE

Fields marked with * are required.

7. Enter the system information as specified in the following table:

Parameter	Description
User id	A valid user ID for J.D. Edwards OneWorld.
User password	The password associated with the J.D. Edwards OneWorld user ID.
JDE Environment	The J.D. Edwards OneWorld environment, for example, DU7333. For more information about this parameter, see your J.D. Edwards OneWorld documentation or ask your J.D. Edwards OneWorld system administrator.
Application	The application that is defined in the J.D. Edwards OneWorld environment.
Server IP address	The name of the server on which J.D. Edwards OneWorld is running. This can be the name of the server, for example, JDEOW, or its IP address.
Server port	The port number on which the server is listening, for example, 6009.
User role	Specify ***ALL**.
Schema Location	The location of the XML schema (.xsd file) that was generated from the event output. For example: `<ADAPTER_HOME>\config\configuration_name\schemas\JDEdwards\target_name\jde-schema.xsd` For more information, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.
Schema Style	Choose from one of the following options: ■ ELEMENT_STYLE (default) ■ ATTRIBUTE_STYLE

8. Click **OK**.

   A summary pane is displayed, providing the channel description, channel status, and available ports. All the information is associated with the channel you created.

   The channel appears under the channels node in the left pane.

An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.

9. Right-click the channel and select **Start**.

   The channel you created becomes active. The X over the icon in the left pane disappears.

10. To stop the channel, right-click the channel and select **Stop**.

2.8.1.3 Creating a File Channel

To create a File Channel:

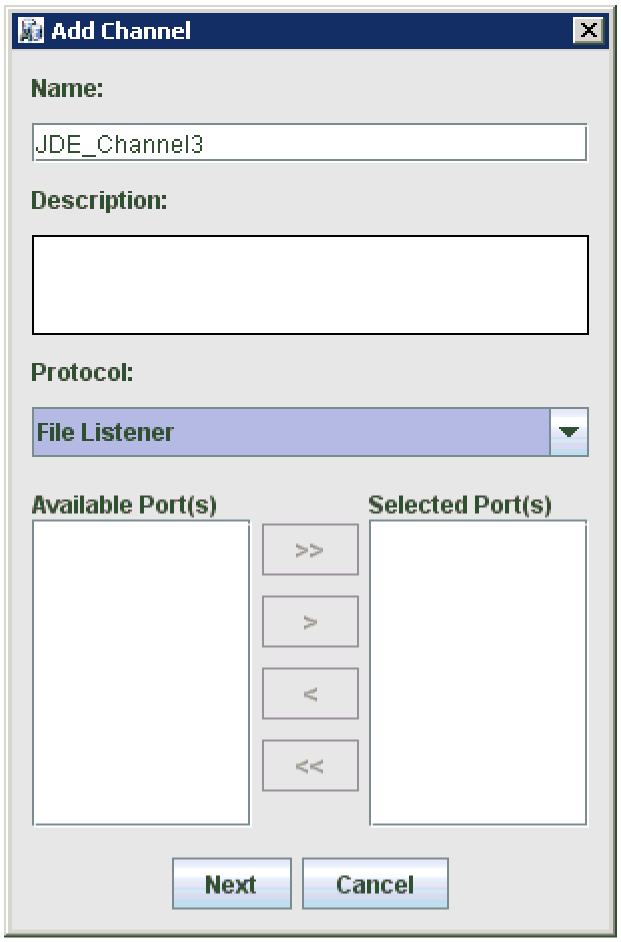
1. Click the **Events** node.
2. Expand the **JDEdwards** node.

   The ports and channels nodes appear in the left pane.

3. Right-click **Channels** and select **Add Channel**.

   The Add Channel dialog is displayed, as shown in Figure 2–24.

Figure 2–24 Add Channel Dialog



Name: JDE_Channel3

Description:

Protocol: File Listener

Provide the following information:

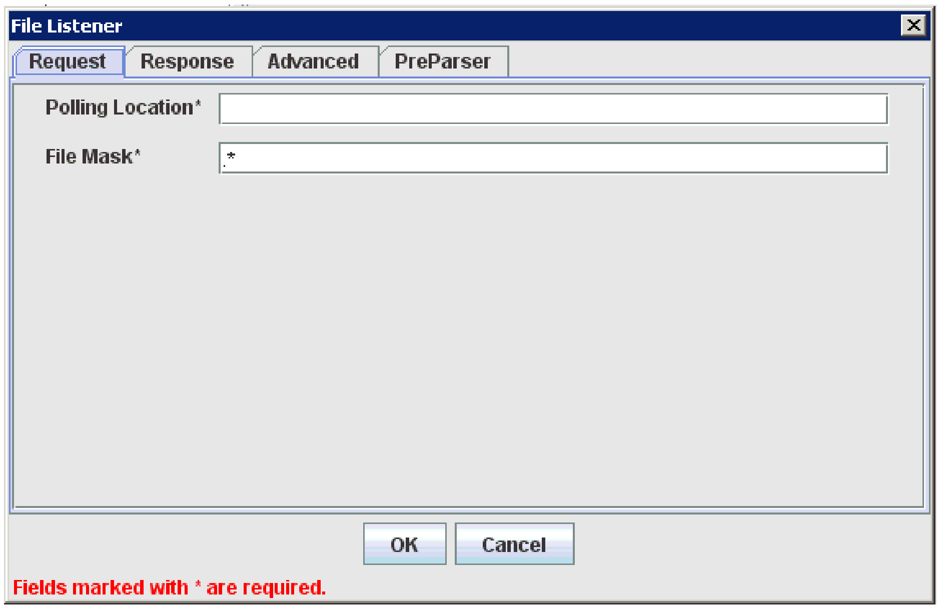
a. Enter a name for the channel, for example, **JDE_Channel3**.

   b. Enter a brief description.
   c. From the **Protocol** list, select **File Listener**.

4. Click **Next**.

   The File Listener dialog is displayed, as shown in Figure 2–25.

Figure 2–25 File Listener Dialog



Polling Location*

File Mask* .*

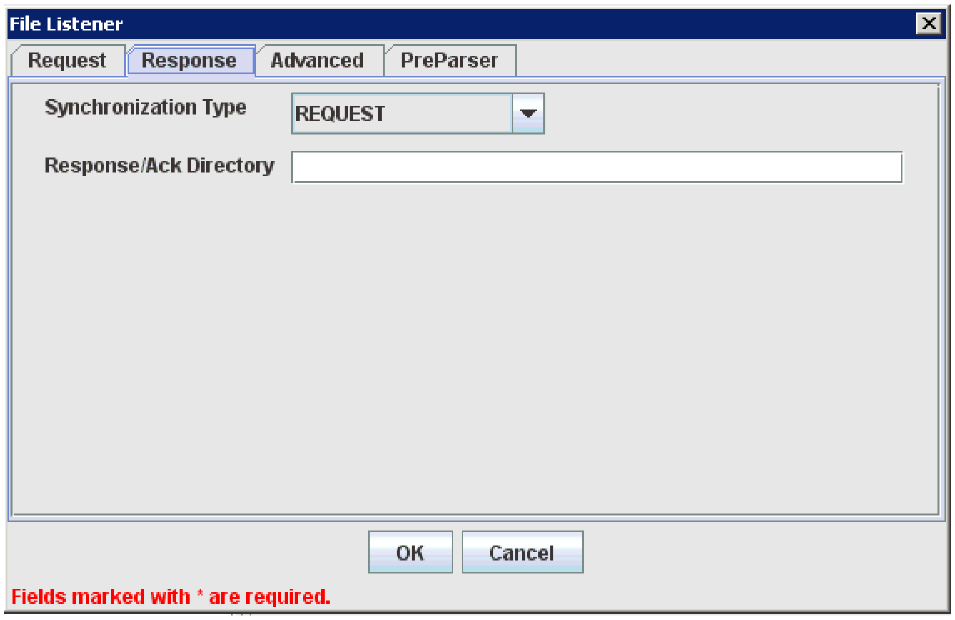
Fields marked with * are required.

5. Enter the system information in the Request tab as specified in the following table:

Parameter	Description
Polling Location	Target file system location for the J.D. Edwards OneWorld XML file.
File Mask	File name to be used for the output file generated by the operation.

6. Click the **Response** tab, as shown in Figure 2–26.

Figure 2–26 File Listener Dialog Response Tab



Synchronization Type: REQUEST

Response/Ack Directory:

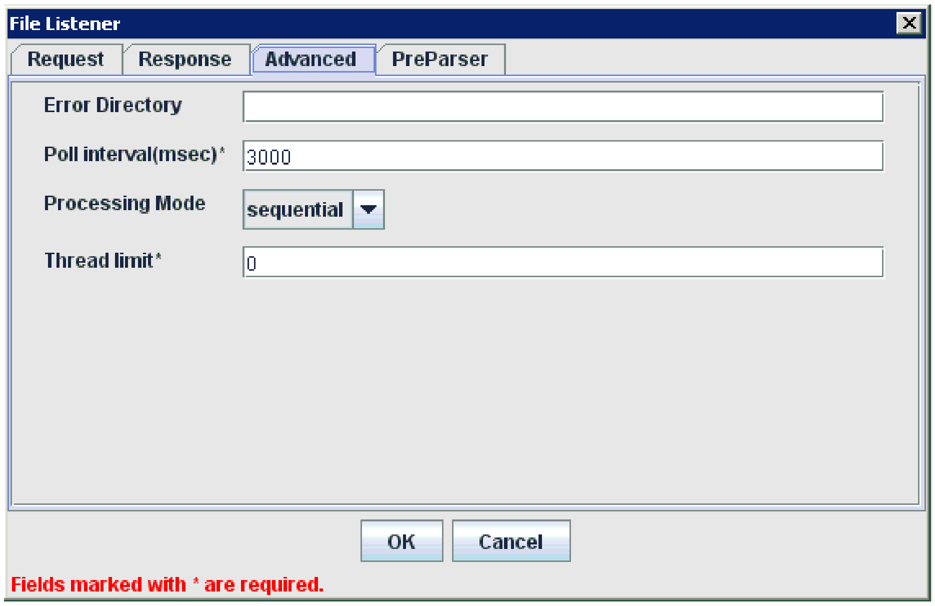
Fields marked with * are required.

7. Enter the system information in the Response tab as specified in the following table:

Parameter	Description
Synchronization Type	Choose from the following synchronization options: ▪ REQUEST_RESPONSE ▪ REQUEST_ACK **Important:** The J.D. Edwards OneWorld channel does not work if the synchronization type is set to REQUEST.
Response/Ack Directory	Target file system location for the J.D. Edwards OneWorld XML file.

8. Click the **Advanced** tab, as shown in Figure 2–27.

Figure 2–27 File Listener Dialog Advanced Tab



Error Directory

Poll interval(msec)* 3000

Processing Mode sequential

Thread limit* 0

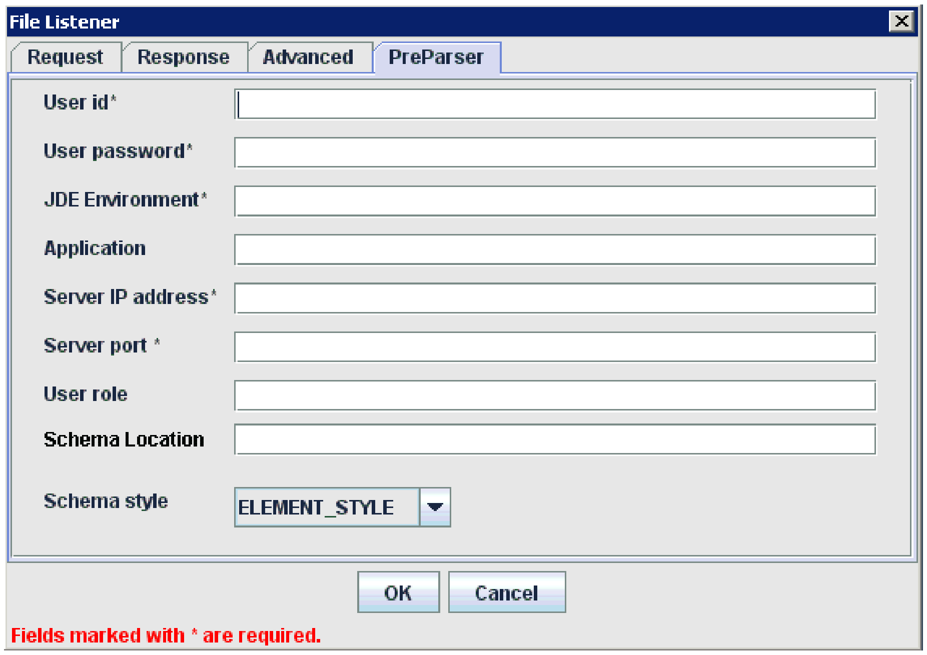
Fields marked with * are required.

9. Enter the system information in the Advanced tab as specified in the following table:

Parameter	Description
Error directory	Directory to which documents with errors are written.
Poll interval (msec)	Interval (in milliseconds) when to check for new input. The default is three seconds. Optional.
Processing Mode	**Sequential** indicates single processing of requests. **Threaded** indicates processing of multiple requests simultaneously.
Thread limit	If you selected threaded processing, then indicate the maximum number of requests that can be processed simultaneously.

10. Click the **PreParser** tab, as shown in Figure 2–28.

Figure 2–28 File Listener Dialog Preparser Tab



User id*

User password*

JDE Environment*

Application

Server IP address*

Server port *

User role

Schema Location

Schema style: ELEMENT_STYLE

Fields marked with * are required.

11. Enter the system information as specified in the following table:

Parameter	Description
User id	A valid user ID for J.D. Edwards OneWorld.
User password	The password associated with the J.D. Edwards OneWorld user ID.
JDE Environment	The J.D. Edwards OneWorld environment, for example, DU7333. For more information about this parameter, see your J.D. Edwards OneWorld documentation or ask your J.D. Edwards OneWorld system administrator.
Application	The application that is defined in the J.D. Edwards OneWorld environment.
Server IP address	The name of the server on which J.D. Edwards OneWorld is running. This can be the name of the server, for example, JDEOW, or its IP address.
Server port	The port number on which the server is listening, for example, 6009.
User role	Specify ***ALL**.
Schema Location	The location of the XML schema (.xsd file) that was generated from the event output. For example: `<ADAPTER_HOME>\config\configuration_name\schemas\JDEdwards\target_name\jde-schema.xsd` For more information, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.
Schema Style	Choose from one of the following options: ■ ELEMENT_STYLE (default) ■ ATTRIBUTE_STYLE

12. Click **OK**.

A summary pane is displayed, providing the channel description, channel status, and available ports. All the information is associated with the channel you created.

The channel appears under the channels node in the left pane.

An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.

13. Right-click the channel and select **Start**.

    The channel you created becomes active. The X over the icon in the left pane disappears.

14. To stop the channel, right-click the channel and select **Stop**.

2.8.1.4 Editing a Channel

To edit a channel:

1. In the left pane, locate the channel you want to edit.
2. Right-click the channel and select **Edit**.

   The Edit channels pane is displayed.

3. Make the required changes to the channel configuration and click **Finish**.

2.8.1.5 Deleting a Channel

To delete a channel:

1. In the left pane, locate the channel you want to delete.
2. Right-click the channel and select **Delete**.

   A confirmation dialog is displayed.

3. To delete the channel you selected, click **OK**.

   The channel disappears from the list in the left pane.

2.8.2 The J.D. Edwards OneWorld Event Listener

Oracle Application Adapter for J.D. Edwards OneWorld Event Listener is designed specifically to provide J.D. Edwards OneWorld approved access to your business events. The J.D. Edwards OneWorld Event Listener refers to a specialized application that runs with J.D. Edwards OneWorld business functions and is called by the J.D. Edwards OneWorld application system.

The J.D. Edwards OneWorld application system provides the Event Listener with the information required to retrieve the event information for only the desired events. For information about configuring the J.D. Edwards OneWorld environment, see the *J.D. Edwards Interoperability Guide for OneWorld.*

The J.D. Edwards OneWorld Event Listener is called directly from the J.D. Edwards OneWorld application and is passed a Z-file record identifier. This identifier then generates a request document that is passed to the server for processing. The server retrieves the event information from the J.D. Edwards OneWorld system and propagates the information for integration with other application systems.

2.8.3 Configuring the J.D. Edwards OneWorld Event Listener

The J.D. Edwards OneWorld Event Listener is installed as part of the basic installation. The J.D. Edwards OneWorld Adapter is automatically installed in the appropriate directory. If the integration server is not installed on the same computer as the J.D. Edwards OneWorld application server, then you must configure the J.D. Edwards OneWorld Event Listener.

The J.D. Edwards OneWorld Event Listener is invoked by J.D. Edwards OneWorld for specific transactions as configured in the J.D. Edwards OneWorld environment.

The J.D. Edwards OneWorld Event Listener includes the following components:

- The listener event stub, `(IWOEvent.dll)`, is located in the `\etc\jde` directory. For example:

  ```
  <ADAPTER_HOME>\etc\jde\iwoevent.dll
  ```

  The file extension varies depending on your operating system:

  – For **Windows**, the event stub is `iwoevent.dll`.

  – For **Sun Solaris**, the event stub is `libiwoevent.so`.

  – For **HP-UX**, the event stub is `libiwoevent.sl`.

  – For **AS/400**, the event stub is `iwaysav.sav`.

  – For **IBM AIX**, the event stub is `libiwoevent.so`.

- The listener configuration file `(iwoevent.cfg)`, which must be created by the user.

The J.D. Edwards OneWorld Event listener exit is the function that passes the key fields for a record in the J.D. Edwards OneWorld outbound transaction tables to the integration server for processing by the inbound Oracle Application Adapter for J.D. Edwards OneWorld. The J.D. Edwards OneWorld Event listener is deployed under the J.D. Edwards OneWorld Enterprise Server. The Java class for the J.D. Edwards OneWorld Event listener is called IWOEvent (the file extension depends on the operating system) and is case-sensitive.

1. Create a folder called Outbound under the JDE structure on the JDE Enterprise Sever, for example:

   ```
   \\JDEdwards\E812\DDP\Outbound
   ```

2. Copy the `iwoevent.dll` file in the new Outbound folder.

3. Create an environment variable, *`IWOEVENT_HOME`*, to point to the directory containing the `iwoevent.dll` file.

   - On Windows: Add *`IWOEVENT_HOME`* to the system environment variables.
   - On UNIX: Add the following command to your start-up script:

     ```
     export IWOEVENT_HOME =/directory_name
     ```

4. On the J.D. Edwards OneWorld Server, create an `iwoevent.cfg` file in the defined directory, *`IWOEVENT_HOME`*.

   The J.D. Edwards OneWorld Event listener requires connection information for the associated adapter to initiate events properly. This information is contained in the `iwoevent.cfg` file. You must create this file and add the connection information to it. The J.D. Edwards OneWorld Event Listener requires connection information for the associated integration server to function properly. This information is contained in the `iwoevent.cfg` file. The `iwoevent.cfg` file has three distinct sections:

   - **Common**

     The common section of the configuration file contains basic configuration options. Currently, only the trace option is supported.

     To set the trace option, select **on** or **off**.

```
common.trace=on|off
```

Where `on` sets the tracing to on and `off` sets the tracing to off. Off is the default value.

- **Alias**

  The alias section of the configuration file contains the connection information required to send transactions to specific servers. Currently, the Oracle Application Adapter for J.D. Edwards OneWorld supports 100 entries (alias names) in the configuration file.

  The alias values to these entries are as follows:

  ```
  Alias.aliasname={ipaddress|dsn}:port, trace={on|off}
  ```

  Where:

  `aliasname` is the symbolic name given to the connection.

  `ipaddress|dsn` is the IP address or DSN name for the server containing Oracle Application Adapter for J.D. Edwards OneWorld (required).

  `port` is the port defined for Oracle Application Adapter for J.D. Edwards OneWorld in the TCP channel configuration (required).

  `trace={on|off}` sets the tracing to on for the particular alias.

- **Trans**

  The trans section of the configuration file contains transaction information required to route J.D. Edwards OneWorld transactions to specified servers.

  If a particular J.D. Edwards OneWorld transaction is not defined to an alias, then it is sent to all aliases. The trans values to these entries are as follows:

  ```
  trans.jdeTransactionName=alias1,alias2,aliasn
  ```

  Where `jdeTransactionName` is the JDE-defined name for the outbound transaction and `alias1,alias2,aliasn` is the list of aliases to which the transactions are sent.

The following is a sample entry for `iwoevent.cfg` that supplies connection information:

```
common.trace=on

alias.edamcs1=172.1.1.1:3694
alias.edamcs1t=172.1.1.1:3694, trace=on
alias.edamcs2=222.2.2.2:1234

trans.JDESOOUT=edamcs1t,edamcs2
trans.JDEPOOUT=edamcs1
```

5. Create a folder using the alias names that are specified in the iwoevent.cfg file under the defined directory, *`IWOEVENT_HOME`*. For example:

```
\\JDEdwards\E812\DDP\Outbound\edamcs1
```

2.9 Runtime Overview

After J.D. Edwards OneWorld starts the J.D. Edwards OneWorld Event listener, the listener accesses the configuration file, called `iwoevent.cfg` (case-sensitive). Based on the information in the configuration file, the listener sends the event notification to the integration server. All log information is saved in a file called `iwoevent.log`. The `iwoevent.log` file is created in the outbound folder where the `iwoevent.dll` and `iwoevent.cfg` files are located.

2.10 Modifying the JDE.INI File for Outbound and Inbound Processing

This section describes the settings that are required in the JDE.INI file for the XML call object kernel (outbound and inbound processing).

Open the JDE.INI file and modify the **[JDENET_KERNEL_DEF6]** and **[JDENET_KERNEL_DEF15]** sections as follows:

```
[JDENET_KERNEL_DEF6]
krnlName=CALL OBJECT KERNEL
dispatchDLLName=XMLCallObj.dll
dispatchDLLFunction=_XMLTransactionDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1

[JDENET_KERNEL_DEF15]
krnlName=XML TRANSACTION KERNEL
dispatchDLLName=XMLTransactions.dll
dispatchDLLFunction=_XMLTransactionDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1
```

The parameters containing an underscore (_) and @28 are for Windows NT operating systems only. For other operating systems, replace the parameters with the values in the following table:

Operating System	Call Object dispatch DLLName	XML Trans dispatch DLLName
AS400	XMLCALLOBJ	XMLTRANS
HP9000B	libxmlcallojb.sl	libxmltransactions.lo
Sun or RS6000	libxmlcallojb.so	Libxmltransactions.so

Note: The J.D. Edwards OneWorld installation for version B7333(XE) does not include **[JDENET_KERNEL_DEF15]**. As a result, if you are using version B7333(XE), you must manually add it to the jde.ini file. For all other J.D. Edwards OneWorld versions, **[JDENET_KERNEL_DEF15]** is included with the installation.

3

Oracle WebLogic Server Deployment and Integration

This chapter describes Oracle WebLogic Server (OracleWLS) deployment and integration with Oracle Application Adapter for J.D. Edwards OneWorld. It contains the following topics:

- Section 3.1, "Adapter Integration with Oracle WebLogic Server"
- Section 3.2, "Deployment of Adapter"
- Section 3.3, "Updating Adapter Configuration"

See Also:

- *Oracle Application Server Adapter Concepts Guide*

3.1 Adapter Integration with Oracle WebLogic Server

Oracle Application Adapter for J.D. Edwards OneWorld is deployed within an OracleWLS container during installation. All client applications run within the OracleWLS environment. In J2CA deployment, the Common Client Interface (CCI) integrates an OracleWLS client application with a resource adapter.

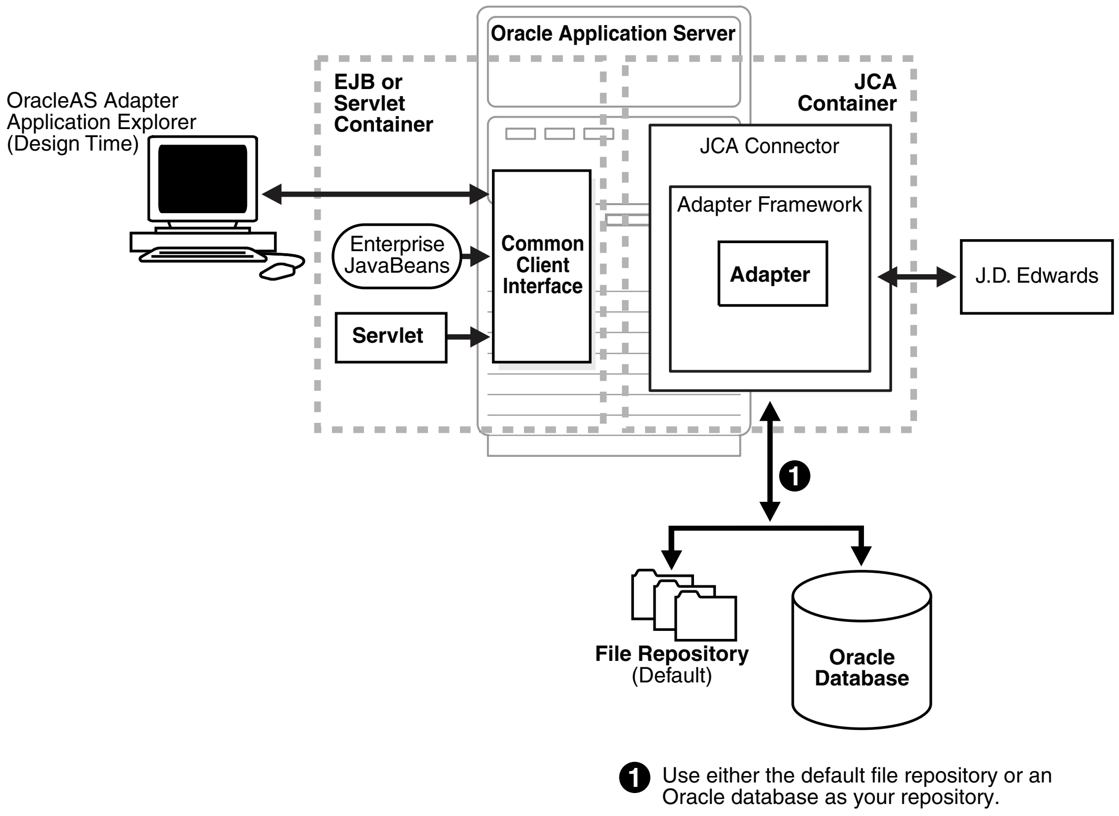
See Also:

- *Oracle Application Server Adapter Concepts Guide*

3.2 Deployment of Adapter

Figure 3–1 shows deployment of the J2CA Connector to the Oracle Application Server. In a run-time service scenario, an Enterprise Java Bean, servlet, or Java program client makes CCI calls to J2CA resource adapters. The adapters process the calls as requests and send them to the EIS. The EIS response is then sent back to the client.

Figure 3–1 Oracle Application Server J2CA Architecture


Oracle Application Server
OracleAS Adapter Application Explorer (Design Time)
EJB or Servlet Container
JCA Container
JCA Connector
Adapter Framework
Adapter
Enterprise JavaBeans
Common Client Interface
Servlet
J.D. Edwards
1
File Repository (Default)
Oracle Database
1 Use either the default file repository or an Oracle database as your repository.

See Also:

- *Oracle Application Server Adapter Concepts Guide*

3.3 Updating Adapter Configuration

This section contains the following topics:

- Section 3.3.1, "Creating a Managed Connector Factory Object"
- Section 3.3.2, "Creating Multiple Managed Connector Factory Objects"
- Section 3.3.3, "Modifying WSDL Files for Additional Connection Factory Values"

During the J2CA deployment of Oracle Application Adapter for J.D. Edwards OneWorld, OracleWLS generates a deployment descriptor called `ra.xml`, located in:

```
<ADAPTER_HOME>\iwafjca.rar\META-INF
```

Your installation contains more than one file named `ra.xml`. The OracleWLS deployment descriptor that is described in this section is located in the directory specified above.

Note: Multiple managed connection factories are supported only for outbound processing (services).

3.3.1 Creating a Managed Connector Factory Object

The `ra.xml` descriptor provides OracleWLS-specific deployment information for resource adapters. For example, the default `jca_sample` configuration in Application Explorer is represented in the `ra.xml` file as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE connector PUBLIC '-//Sun Microsystems, Inc.//DTD Connector 1.0//EN'
'http://java.sun.com/dtd/connector_1_0.dtd'>
<connector>
  <display-name>IWAFJCA10</display-name>
  <vendor-name>IWAY Software</vendor-name>
  <spec-version>1.0</spec-version>
  <eis-type>IWAF</eis-type>
  <version>1.0</version>
  <license>
    <license-required>false</license-required>
  </license>
  <resourceadapter>

<managedconnectionfactory-class>com.ibi.afjca.spi.IWAFManagedConnectionFactory</ma
nagedconnectionfactory-class>

<connectionfactory-interface>javax.resource.cci.ConnectionFactory</connectionfacto
ry-interface>

<connectionfactory-impl-class>com.ibi.afjca.cci.IWAFConnectionFactory</connectionf
actory-impl-class>
    <connection-interface>javax.resource.cci.Connection</connection-interface>

<connection-impl-class>com.ibi.afjca.cci.IWAFConnection</connection-impl-class>
    <transaction-support>NoTransaction</transaction-support>
    <config-property>
      <config-property-name>AdapterName</config-property-name>
      <config-property-type>java.lang.String</config-property-type>
      <config-property-value></config-property-value>
    </config-property>
    <config-property>
      <config-property-name>Config</config-property-name>
      <config-property-type>java.lang.String</config-property-type>
      <config-property-value></config-property-value>
    </config-property>
    <config-property>
      <config-property-name>IWayHome</config-property-name>
      <config-property-type>java.lang.String</config-property-type>
      <config-property-value>C:\oracle\Middleware\Oracle_
SOA1\soa\thirdparty\ApplicationAdapters</config-property-value>
    </config-property>
    <config-property>
      <config-property-name>IWayConfig</config-property-name>
      <config-property-type>java.lang.String</config-property-type>
      <config-property-value>jca_sample</config-property-value>
    </config-property>
    <config-property>
      <config-property-name>IWayRepoDriver</config-property-name>
      <config-property-type>java.lang.String</config-property-type>
      <config-property-value></config-property-value>
    </config-property>
    <config-property>
      <config-property-name>IWayRepoURL</config-property-name>
      <config-property-type>java.lang.String</config-property-type>
```

```
        <config-property-value></config-property-value>
      </config-property>
      <config-property>
        <config-property-name>IWayRepoUser</config-property-name>
        <config-property-type>java.lang.String</config-property-type>
        <config-property-value></config-property-value>
      </config-property>
      <config-property>
        <config-property-name>IWayRepoPassword</config-property-name>
        <config-property-type>java.lang.String</config-property-type>
        <config-property-value></config-property-value>
      </config-property>
      <config-property>
        <config-property-name>LogLevel</config-property-name>
        <config-property-type>java.lang.String</config-property-type>
        <config-property-value>DEBUG</config-property-value>
      </config-property>
      <authentication-mechanism>
        <authentication-mechanism-type>BasicPassword</authentication-mechanism-type>

<credential-interface>javax.resource.spi.security.PasswordCredential</credential-i
nterface>
      </authentication-mechanism>
      <reauthentication-support>true</reauthentication-support>
  </resourceadapter>
</connector>
```

The parameters defined in the ra.xml file are described in the following table:

Parameter Name	Description
IWayHome	The base installation directory for the OracleWLS packaged application adapter.
IWayConfig	The adapter configuration name as defined in Application Explorer. For example, Oracle Application Adapter for J.D. Edwards OneWorld has a preconfigured `jca_sample` configuration in Application Explorer.
IWayRepoURL	The URL to use when opening a connection to the database. This is necessary only when using an Oracle database as the repository.
IWayRepoUser	User name to use when connecting to the database. This is necessary only when using an Oracle database as the repository.
IWayRepoPassword	Password. If provided, then it overwrites configuration. This is necessary only when using an Oracle database as the repository.
loglevel	It overwrites the level set by the ManagedConnectorFactory property.

3.3.2 Creating Multiple Managed Connector Factory Objects

To establish multiple managed connector factory objects, you must edit the `weblogic-ra.xml` file and add more `<connection-instance>` nodes. This file is located in:

<ADAPTER_HOME>`\iwafjca.rar\META-INF`

For example, the first jca_configuration in Application Explorer is represented in the `weblogic-ra.xml` file as follows:

```
<?xml version="1.0"?>
<weblogic-connector xmlns="http://www.bea.com/ns/weblogic/90">
    <enable-access-outside-app>true</enable-access-outside-app>
    <enable-global-access-to-classes>true</enable-global-access-to-classes>
    <outbound-resource-adapter>
        <default-connection-properties>
        <pool-params>
        <initial-capacity>0</initial-capacity>
        </pool-params>
        <transaction-support>LocalTransaction</transaction-support>
        </default-connection-properties>
        <connection-definition-group>

<connection-factory-interface>javax.resource.cci.ConnectionFactory</connection-fac
tory-interface>
            <connection-instance>
                <jndi-name>eis/OracleJCAAdapter/DefaultConnection</jndi-name>
            </connection-instance>
        </connection-definition-group>
    </outbound-resource-adapter>
</weblogic-connector>
```

To create multiple managed connector factory objects, you must add new `<connection-instance>` nodes in the file. For example:

```
<?xml version="1.0"?>
<weblogic-connector xmlns="http://www.bea.com/ns/weblogic/90">

    <enable-access-outside-app>true</enable-access-outside-app>
    <enable-global-access-to-classes>true</enable-global-access-to-classes>

    <outbound-resource-adapter>
        <default-connection-properties>
        <pool-params>
        <initial-capacity>0</initial-capacity>
        </pool-params>
        <transaction-support>LocalTransaction</transaction-support>
        </default-connection-properties>
        <connection-definition-group>

<connection-factory-interface>javax.resource.cci.ConnectionFactory</connection-fac
tory-interface>
            <connection-instance>
                <jndi-name>eis/OracleJCAAdapter/DefaultConnection</jndi-name>
            </connection-instance>
            <connection-instance>
                <jndi-name>eis/OracleJCAAdapter/DefaultConnection1</jndi-name>
                <connection-properties>
                <properties>
                <property>
<name>IWayHome</name>
<value>C:\oracle\Middleware\Oracle_SOA1\soa\thirdparty\ApplicationAdapters</value>
                </property>
                <property>
                <name>IWayConfig</name>
                <value>jca_sample2</value>
                </property>
            <property>
    <name>IWayRepoURL</name>
    <value></value>
            </property>
            <property>
      <name>IWayRepoUser</name>
      <value></value>
            </property>
            <property>
      <name>IWayRepoPassword</name>
      <value></value>
            </property>
            <property>
      <name>LogLevel</name>
      <value>Debug</value>
            </property>
            </properties>
            </connection-properties>
         </connection-instance>
      </connection-definition-group>
    </outbound-resource-adapter>
</weblogic-connector>
```

If you do not specify a `<property>` element in the `<connection-instance>` section, then the value is taken from the `ra.xml` file. You can specify the default properties in the `ra.xml` file and then override them as required in the `weblogic-ra.xml` file. In addition, note that the J2CA configuration (for example, jca_sample2) must already be created in Application Explorer.

Note: When you modify the `ra.xml` and `weblogic-ra.xml` files, the application server must be restarted. If the application server is already running, then stop the application server and then restart it.

In addition, the `iwafjca.rar` file must be redeployed in the Oracle WebLogic Administration Console to activate these changes.

3.3.3 Modifying WSDL Files for Additional Connection Factory Values

Application Explorer generates the J2CA properties file using the default connection factory name `eis/OracleJCAAdapter/DefaultConnection`. If you created additional connection factories, then the WSDLs generated for the additional configuration and connection factory should be changed to reflect the location field of the `jca:address` section in the J2CA properties file. The default J2CA properties file for the Oracle Application Adapter for J.D. Edwards OneWorld with a configuration of `isdsrv2_conn2` is shown in the following example.

Notice that the J2CA properties file has the following default connection factory: `eis/OracleJCAAdapter/DefaultConnection`

```
<jca:address location="eis/OracleJCAAdapter/DefaultConnection"
        ConnectionSpec="com.ibi.afjca.cci.IWAFConnectionSpec"
        cs.AdapterName="JDEdwards" cs.Config="isdsrv2_conn2"
UIConnectionName="Connection1"/>
```

The connection factory value must be changed to the following: `eis/OracleJCAAdapter/DefaultConnection1`

For example:

```
<jca:address location="eis/OracleJCAAdapter/DefaultConnection1"
                       ConnectionSpec="com.ibi.afjca.cci.IWAFConnectionSpec"
                       cs.AdapterName="JDEdwards" cs.Config="isdsrv2_conn2"
UIConnectionName="Connection1"/>
```

Note that only the value for the location field in the `jca:address` section should be modified. Do not modify any other field or section.

4 Integration With BPEL Service Components in the Oracle SOA Suite

Oracle Application Adapter for J.D. Edwards OneWorld integrates seamlessly with Business Process Execution Language (BPEL) Process Manager to facilitate Web service integration. Oracle BPEL Process Manager is based on the Service-Oriented Architecture (SOA). It consumes adapter services exposed as Web Service Definition Language (WSDL) documents.

This chapter contains the following topics:

- Section 4.1, "Overview"
- Section 4.2, "Deployment of Adapter"
- Section 4.3, "Configuring a New Application Server Connection"
- Section 4.4, "Designing an Outbound BPEL Process for Service Integration (J2CA Configuration)"
- Section 4.5, "Designing an Inbound BPEL Process for Event Integration (J2CA Configuration)"
- Section 4.6, "Designing an Outbound BPEL Process for Service Integration (BSE Configuration)"

4.1 Overview

To integrate with Oracle BPEL Process Manager, Oracle Application Adapter for J.D. Edwards OneWorld must be deployed in the same WLS container as Oracle BPEL Process Manager. The underlying adapter services must be exposed as WSDL files, which are generated during design time in Oracle Application Adapter Application Explorer (Application Explorer) for both request-response (outbound) and event notification (inbound) services of the adapter. For more information, see Chapter 2, "Configuring Oracle Application Adapter for J.D. Edwards OneWorld".

The generated WSDL files are used to design the appropriate BPEL processes for inbound or outbound adapter services. A completed BPEL process must be successfully compiled in Oracle JDeveloper and deployed to a BPEL server. Upon deployment to the BPEL server, every newly built process is automatically deployed to the Oracle Enterprise Manager console, where you run, monitor, administer BPEL processes, and listen to adapter events.

4.2 Deployment of Adapter

During installation, Oracle Application Adapter for J.D. Edwards OneWorld is deployed as a J2CA 1.0 resource adapter within the WLS container. The adapter must be deployed in the same WLS container as Oracle BPEL Process Manager.

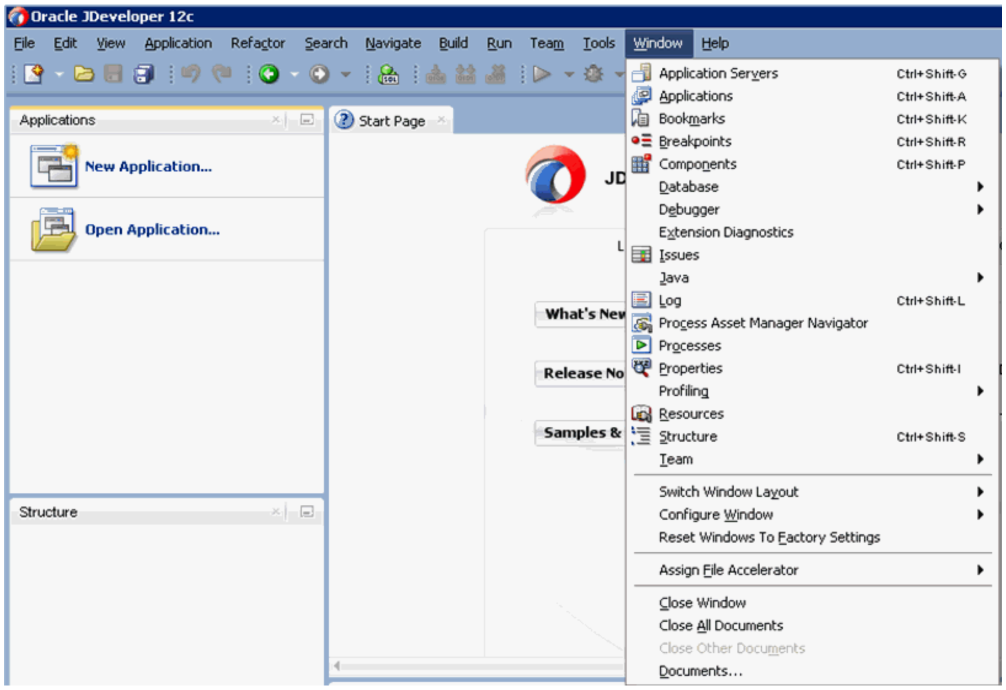
See Also: *Oracle Application Server Adapter Concepts Guide*

4.3 Configuring a New Application Server Connection

To configure a new Application Server connection in Oracle JDeveloper:

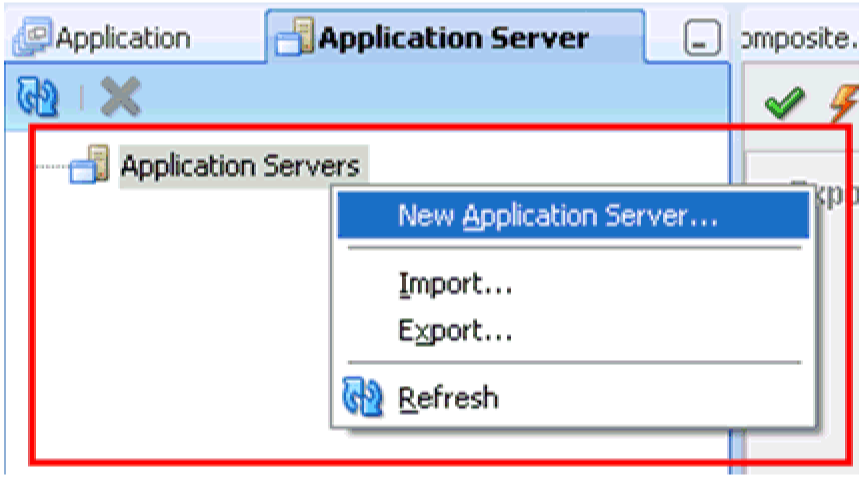
1. Open **Oracle JDeveloper** on your system.
2. From the menu bar, click **Window** and select **Application Server Navigator**, as shown in Figure 4–1.

Figure 4–1 Application Server Navigator



The Application Server tab is displayed, as shown in Figure 4–2.

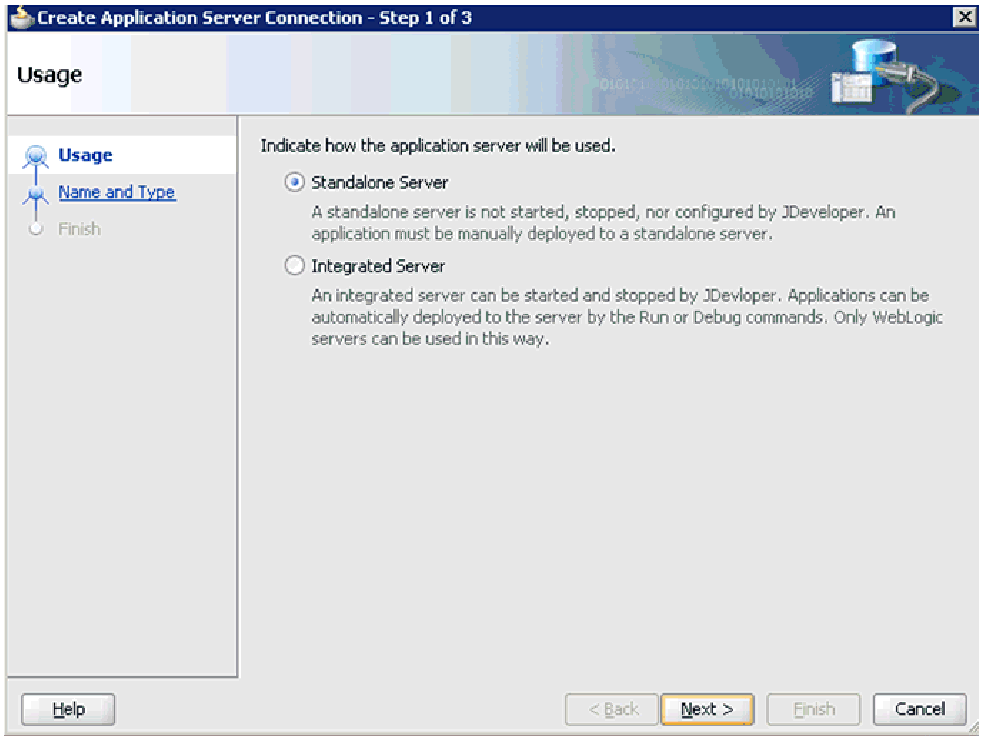
Figure 4–2 Application Server Tab



3. Right-click **Application Servers** and select **New Application Server**.

   The Create Application Server Connection Wizard is displayed, as shown in Figure 4–3.

Figure 4–3 Create Application Server Connection Wizard



Indicate how the application server will be used.

(●) Standalone Server

A standalone server is not started, stopped, nor configured by JDeveloper. An application must be manually deployed to a standalone server.

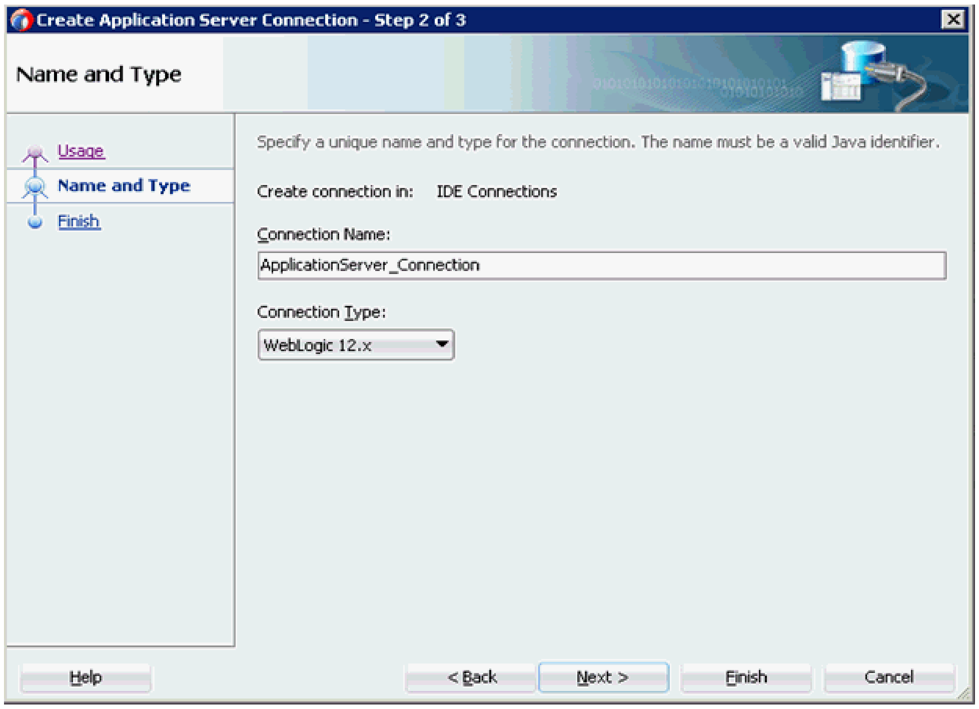
( ) Integrated Server

An integrated server can be started and stopped by JDevloper. Applications can be automatically deployed to the server by the Run or Debug commands. Only WebLogic servers can be used in this way.

4. Accept the default selection (Standalone Server) and click **Next**.

   The Name and Type page is displayed, as shown in Figure 4–4.

Figure 4–4 Name and Type Page


Create Application Server Connection - Step 2 of 3

Name and Type

Specify a unique name and type for the connection. The name must be a valid Java identifier.

Create connection in: IDE Connections

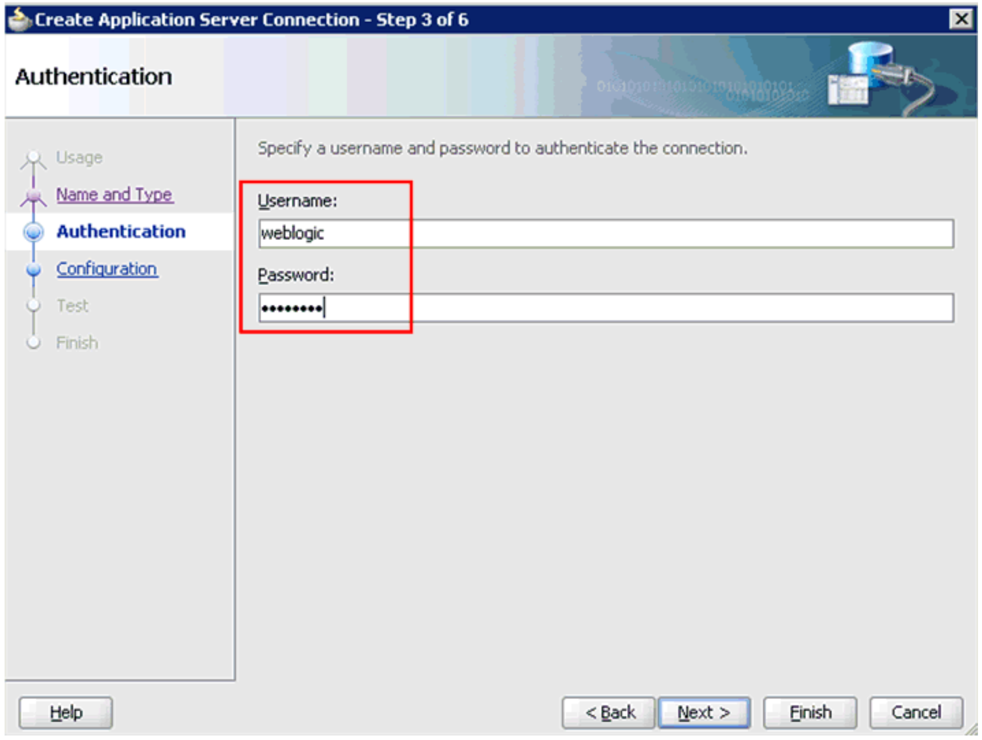
Connection Name: ApplicationServer_Connection

Connection Type: WebLogic 12.x

5. Specify a new name for the Application Server connection and click **Next**.

   The Authentication page is displayed, as shown in Figure 4–5.

Figure 4–5 Authentication Page


Create Application Server Connection - Step 3 of 6

Authentication

Specify a username and password to authenticate the connection.

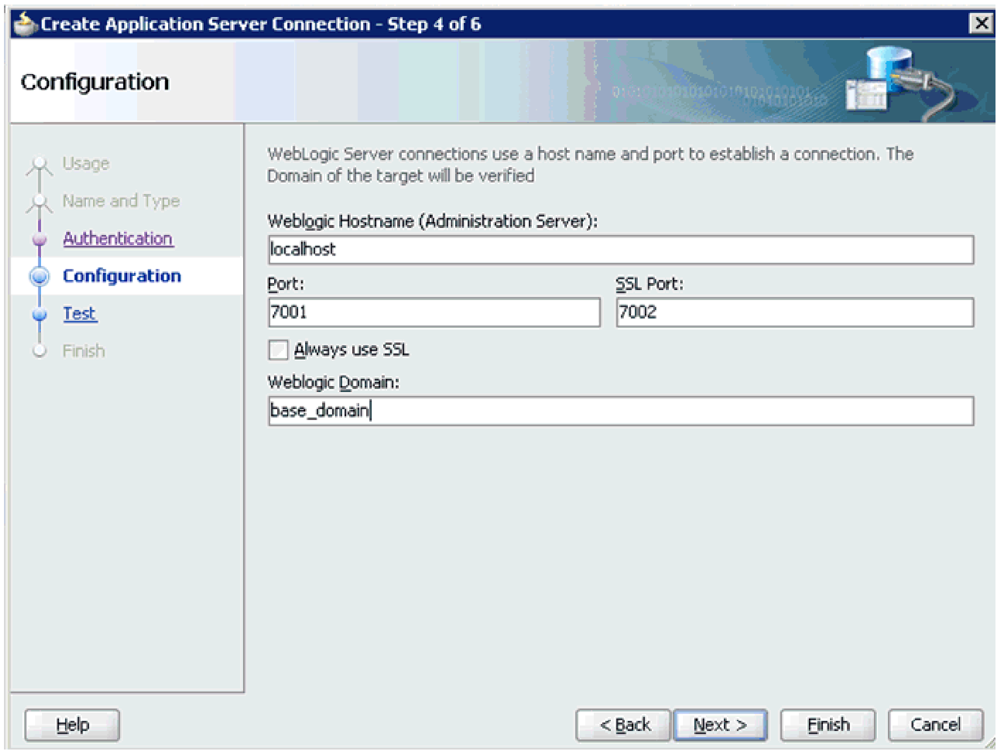
Username: weblogic

Password: ••••••••

6. Specify a valid user name (for example, weblogic) and a password (for example, welcome1) for your new connection.
7. Click **Next**.

   The Configuration page is displayed, as shown in Figure 4–6.

Figure 4–6 Configuration Page



Create Application Server Connection - Step 4 of 6

Configuration

WebLogic Server connections use a host name and port to establish a connection. The Domain of the target will be verified

Weblogic Hostname (Administration Server): localhost

Port: 7001

SSL Port: 7002

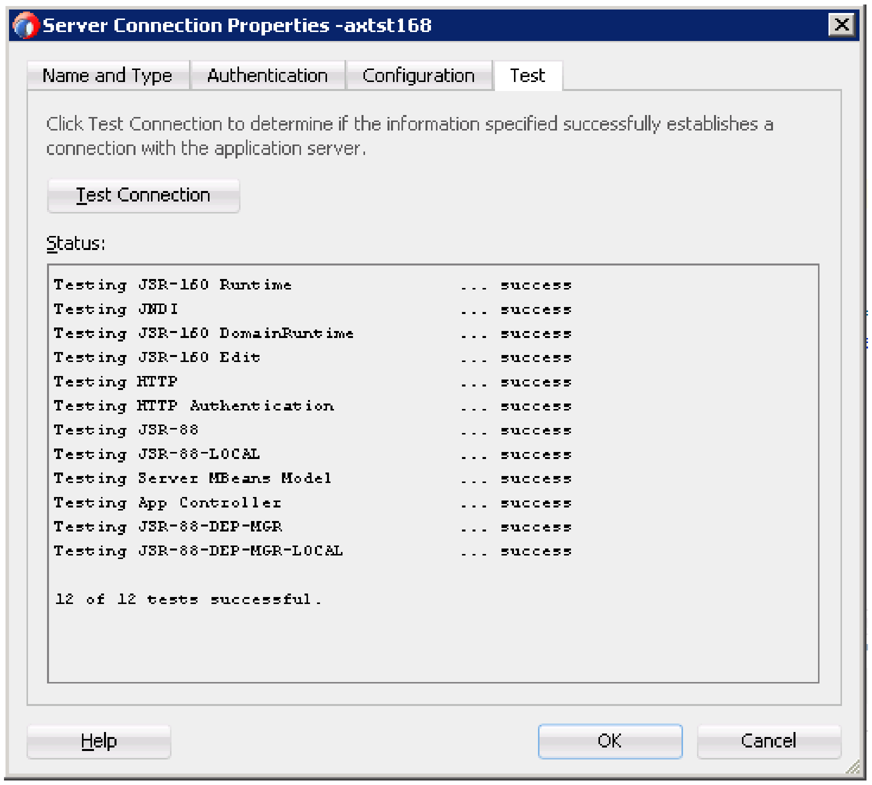
Always use SSL

Weblogic Domain: base_domain

8. Specify the Oracle WebLogic host name (for example, localhost), which is the system IP where the process must deploy and Oracle WebLogic domain (for example, base_domain).
9. Click **Next**.

   The Test page is displayed, as shown in Figure 4–7.

Figure 4–7 Test Page



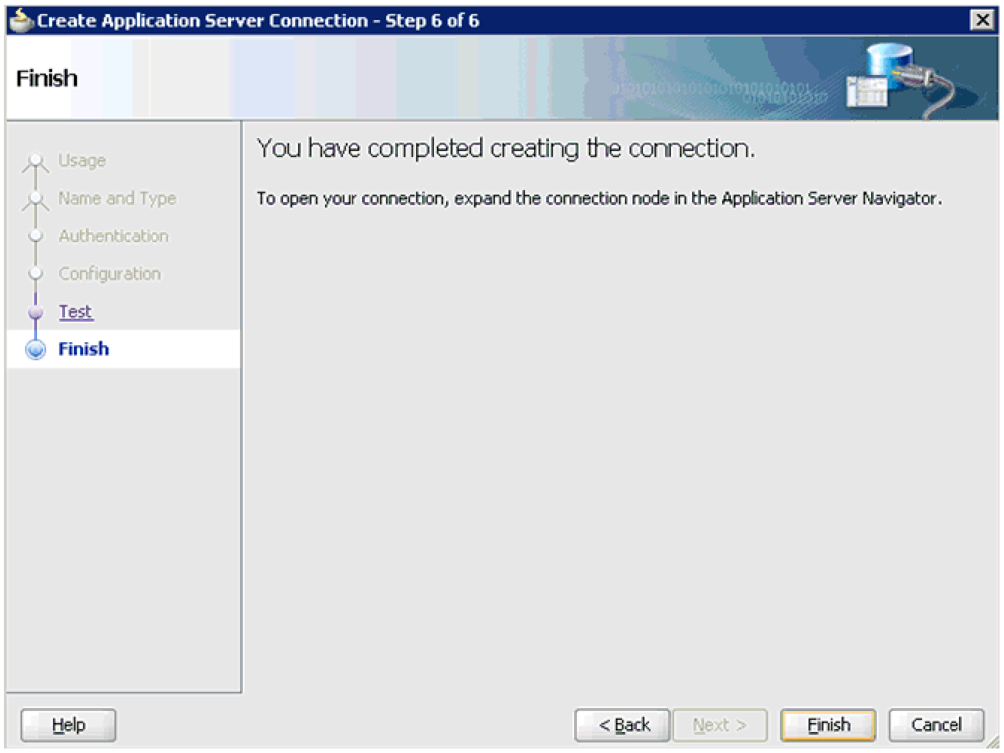
Click Test Connection to determine if the information specified successfully establishes a connection with the application server.

```
Testing JSR-160 Runtime                    ... success
Testing JNDI                               ... success
Testing JSR-160 DomainRuntime              ... success
Testing JSR-160 Edit                       ... success
Testing HTTP                               ... success
Testing HTTP Authentication                ... success
Testing JSR-88                             ... success
Testing JSR-88-LOCAL                       ... success
Testing Server MBeans Model                ... success
Testing App Controller                     ... success
Testing JSR-88-DEP-MGR                     ... success
Testing JSR-88-DEP-MGR-LOCAL               ... success

12 of 12 tests successful.
```

10. Click **Test Connection**.
11. Make sure that the test status is successful.
12. Click **Next**.

    The Finish page is displayed, as shown in Figure 4–8.

Figure 4–8 Finish Page



You have completed creating the connection.

To open your connection, expand the connection node in the Application Server Navigator.

13. Click **Finish**.

    The new Application Server connection is listed in the left pane (Application Server tab).

4.4 Designing an Outbound BPEL Process for Service Integration (J2CA Configuration)

This section describes how to design an outbound BPEL process for service integration.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\BPEL\J2CA\Outbound_Project
```

The following tools are required to complete your adapter design-time configuration:

- Oracle Adapter Application Explorer (Application Explorer)
- Oracle JDeveloper BPEL Designer (JDeveloper)

Note: The examples in this chapter demonstrate the use of Oracle JDeveloper.

This section contains the following topics:

- Section 4.4.1, "Generating WSDL for Request/Response Service"
- Section 4.4.2, "Creating an Empty Composite for SOA"

- Section 4.4.3, "Defining a BPEL Outbound Process"
- Section 4.4.4, "Deploying the BPEL Outbound Process"
- Section 4.4.5, "Invoking the Input XML Document in the Oracle Enterprise Manager Console"
- Section 4.4.6, "Testing Outbound BPEL and Mediator Processes"

Before you design a BPEL process, you must generate WSDL using Application Explorer. For more information, see "Generating WSDL for Request/Response Service" on page 4-8. The WSDL generated in Application Explorer is used during the BPEL process configuration.

4.4.1 Generating WSDL for Request/Response Service

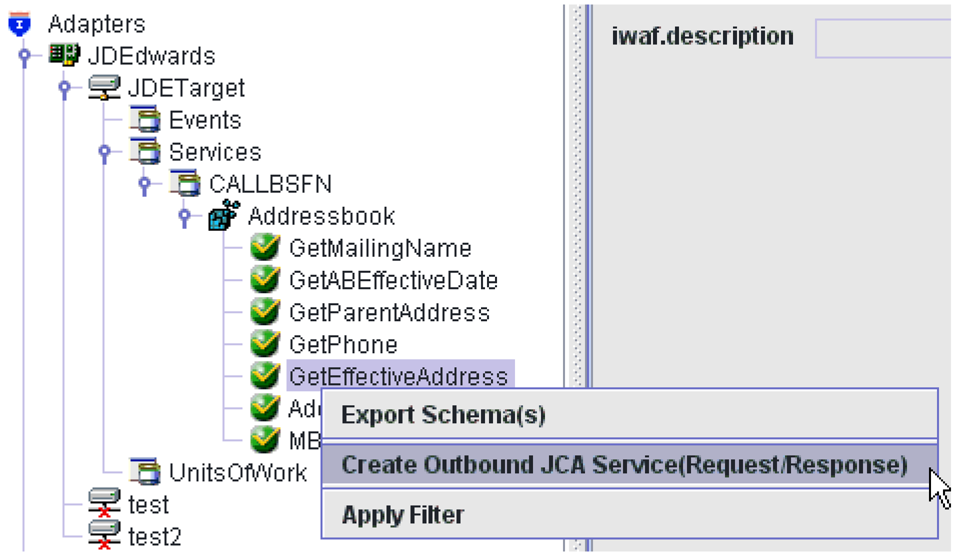
Perform the following steps to generate a WSDL for request/response service:

1. Start Application Explorer and connect to a defined J.D. Edwards OneWorld target.

   For more information, see "Defining a Target to J.D. Edwards OneWorld" on page 2-5.

2. Expand **Services**, **CALLBSFN**, and **Addressbook**.

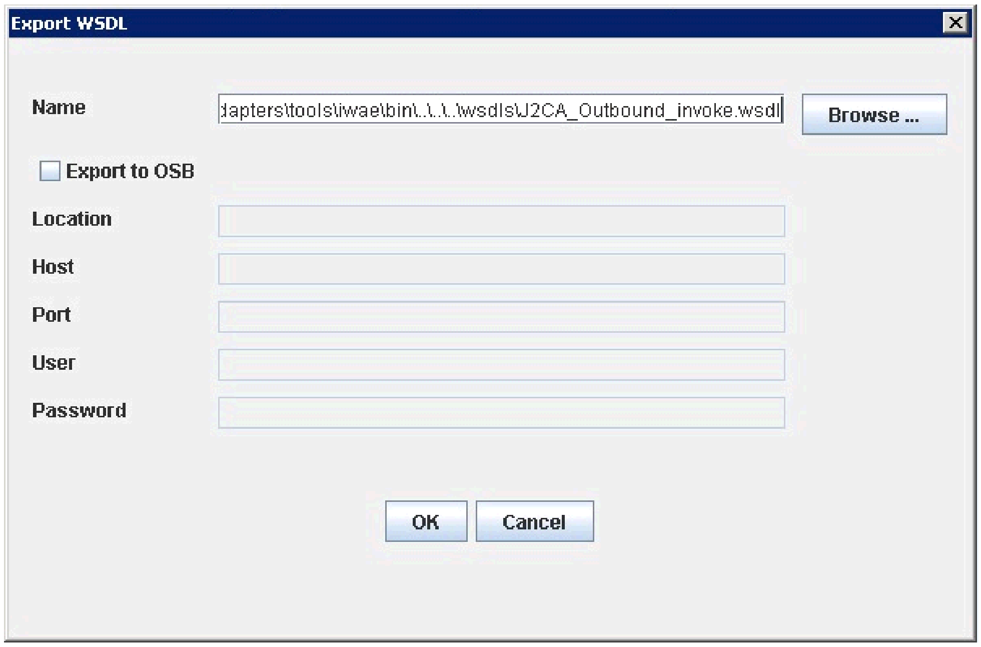
Figure 4–9 Create Outbound JCA Service(Request/Response) Option



3. Right-click **GetEffectiveAddress**, and then select **Create Outbound JCA Service (Request/Response)**, as shown in Figure 4–9.

   The Export WSDL dialog is displayed, as shown in Figure 4–10.

Figure 4–10 Export WSDL Dialog


Export WSDL

Name: dapters\tools\iwae\bin\..\..\..\wsdls\J2CA_Outbound_invoke.wsdl

4. Accept the default name for the file.

   The **.wsdl** file extension is added automatically. By default, the names of WSDL files generated for request-response services end with **_invoke**.

5. Click **OK**.

You can now create a new SOA application, which is the first step that is required to define a BPEL outbound process in Oracle JDeveloper.

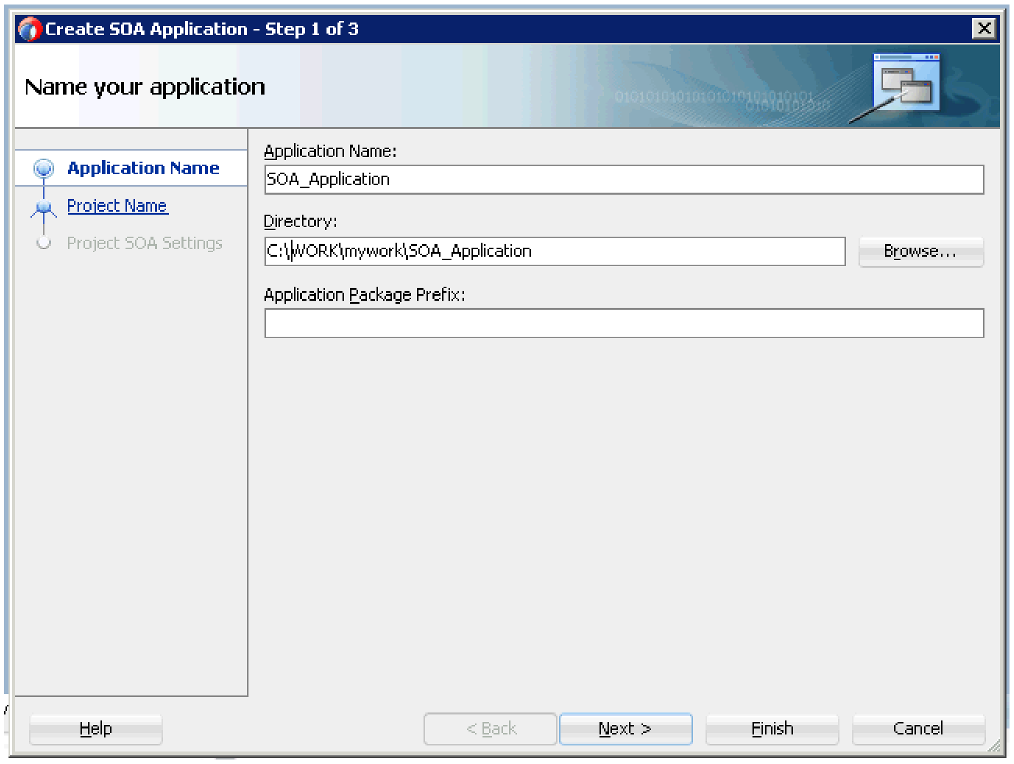
4.4.2 Creating an Empty Composite for SOA

Perform the following steps to create an empty composite for SOA:

1. Create a new SOA application.

2. Enter a name for the new SOA Application and click **Next**, as shown in Figure 4–11.

Figure 4–11 Name Your Application Page



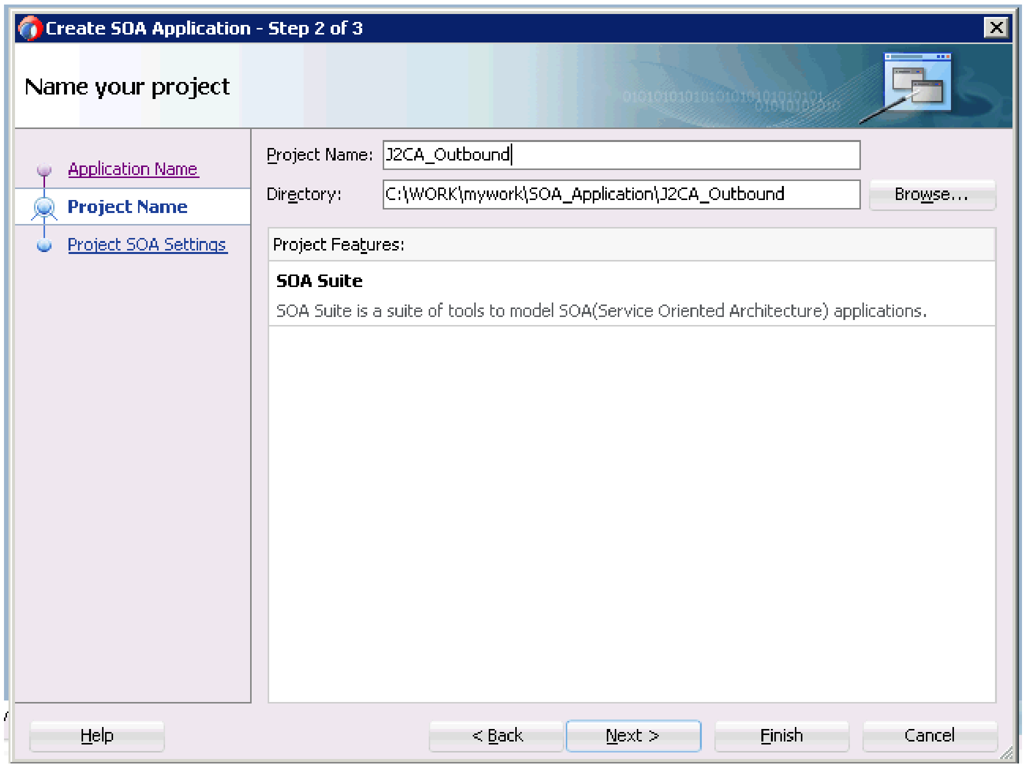
Application Name: SOA_Application

Directory: C:\WORK\mywork\SOA_Application

Application Package Prefix:

The Name your project page is displayed, as shown in Figure 4–12.

Figure 4–12 Name Your Project Page



Project Name: J2CA_Outbound

Directory: C:\WORK\mywork\SOA_Application\J2CA_Outbound

Project Features:

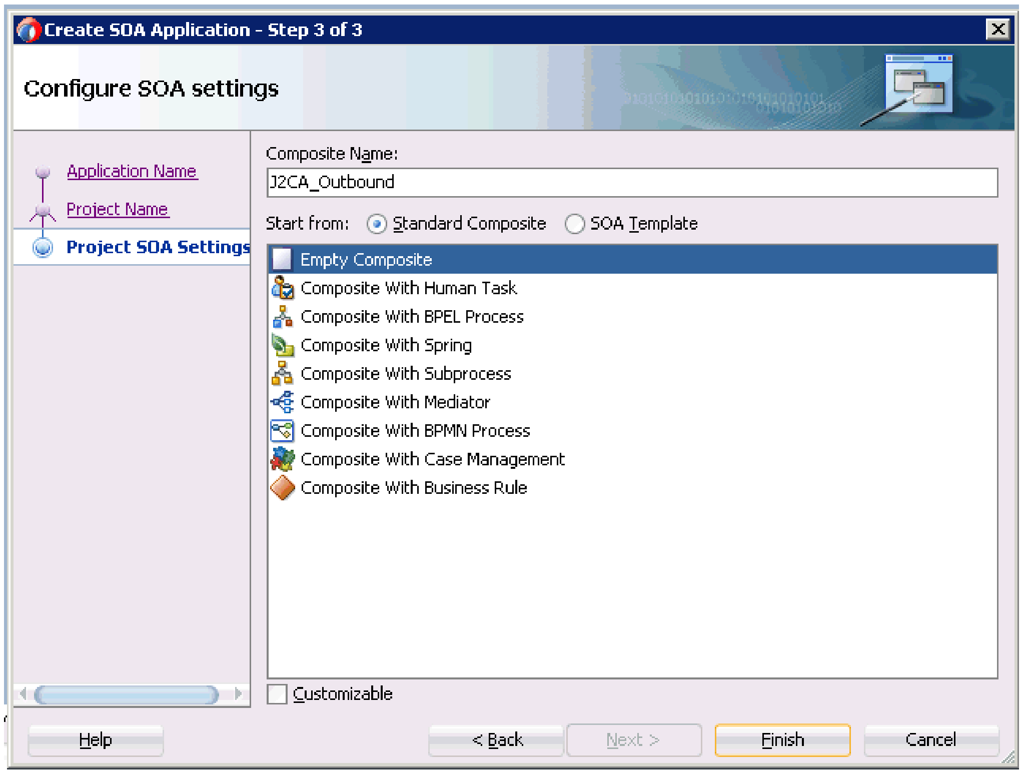
SOA Suite

SOA Suite is a suite of tools to model SOA(Service Oriented Architecture) applications.

3. Enter a project name and click **Next**.

   The Configure SOA settings page is displayed, as shown in Figure 4–13.

Figure 4–13 Configure SOA Settings Page



Create SOA Application - Step 3 of 3

Configure SOA settings

Application Name
Project Name
Project SOA Settings

Composite Name: J2CA_Outbound

Start from: [✓ Standard Composite] SOA Template

Empty Composite
Composite With Human Task
Composite With BPEL Process
Composite With Spring
Composite With Subprocess
Composite With Mediator
Composite With BPMN Process
Composite With Case Management
Composite With Business Rule

Customizable

4. From the Composite Template list, select **Empty Composite** and click **Finish**.

4.4.3 Defining a BPEL Outbound Process

This section describes how to define a BPEL outbound process, which consists of the following topics:

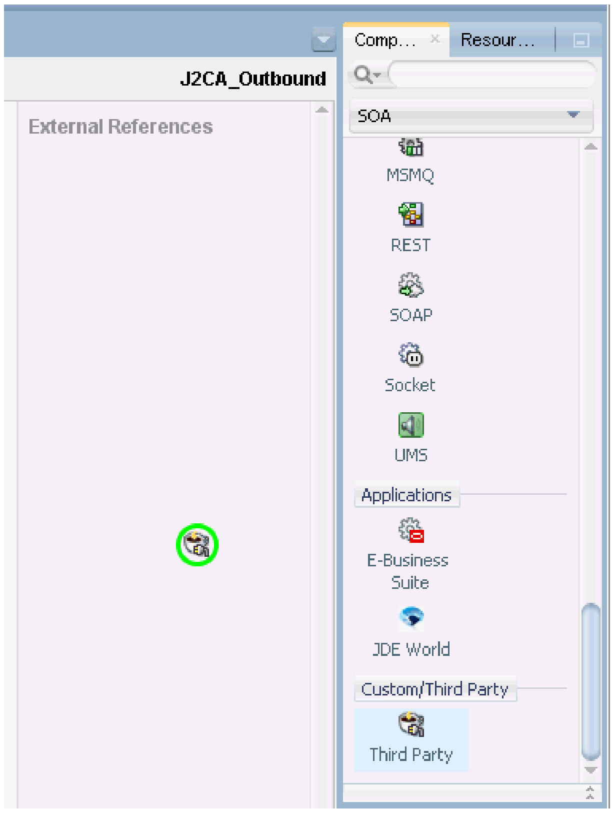
- Section 4.4.3.1, "Configuring a Third Party Adapter Service Component"
- Section 4.4.3.2, "Configuring an Outbound BPEL Process Component"
- Section 4.4.3.3, "Adjusting for Known Deployment Issues With 12c"

4.4.3.1 Configuring a Third Party Adapter Service Component

Perform the following steps to create a third party adapter service component:

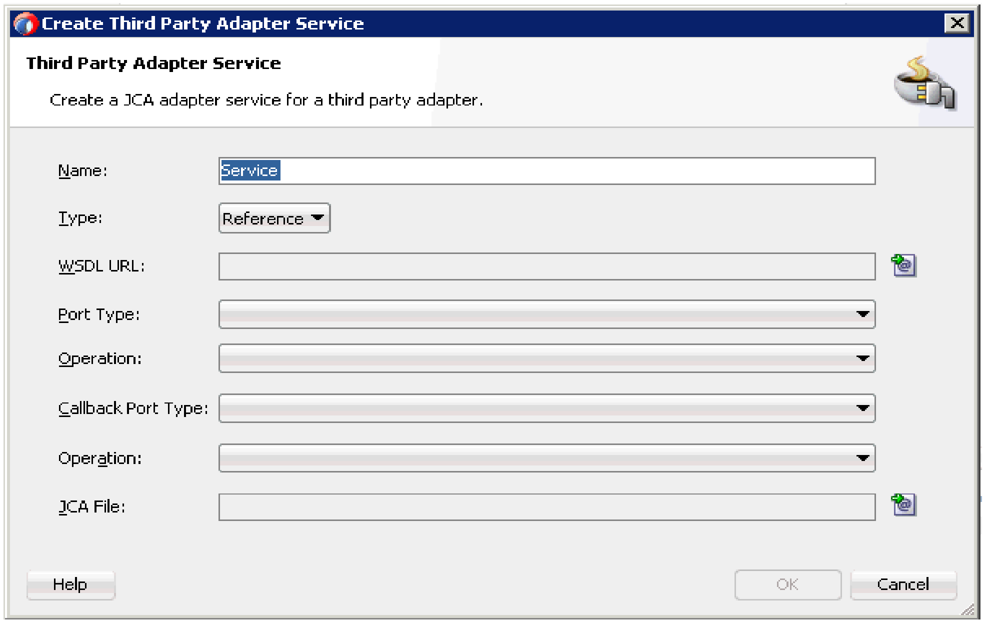
1. Drag and drop the **Third Party Adapter** component from the Service Adapters pane to the External References pane, as shown in Figure 4–14.

Figure 4–14 Third Party Adapter Component



The Create Third Party Adapter Service dialog is displayed, as shown in Figure 4–15.

Figure 4–15 Create Third Party Adapter Service Dialog



Third Party Adapter Service

Create a JCA adapter service for a third party adapter.

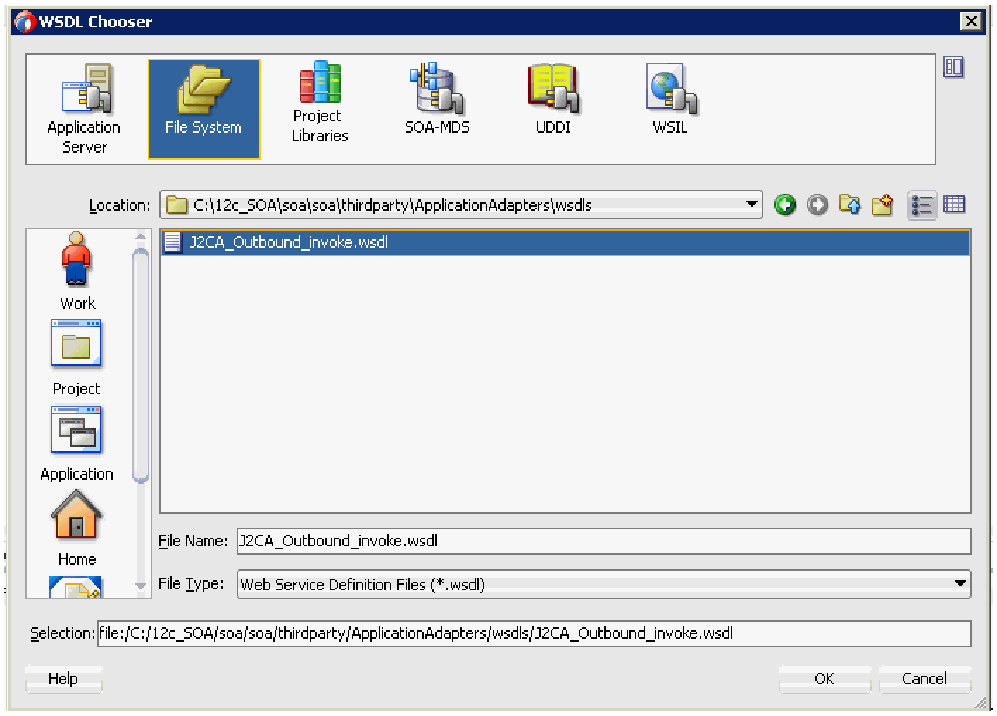
Name:	Service
Type:	Reference
WSDL URL:	
Port Type:	
Operation:	
Callback Port Type:	
Operation:	
JCA File:	

2. Ensure that **Reference** is selected from the Type list (default).

3. Click the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.

   The WSDL Chooser dialog is displayed, as shown in Figure 4–16.

Figure 4–16 WSDL Chooser Dialog



Location: C:\12c_SOA\soa\soa\thirdparty\ApplicationAdapters\wsdls

J2CA_Outbound_invoke.wsdl

File Name: J2CA_Outbound_invoke.wsdl

File Type: Web Service Definition Files (*.wsdl)

Selection: file:/C:/12c_SOA/soa/soa/thirdparty/ApplicationAdapters/wsdls/J2CA_Outbound_invoke.wsdl

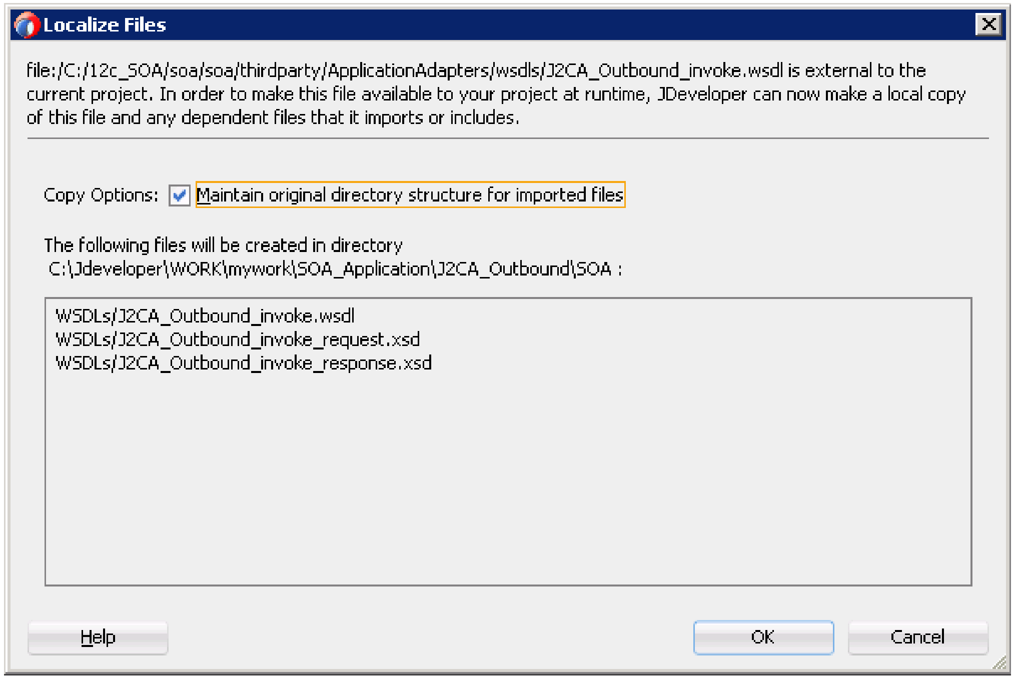
4. Browse and select an outbound WSDL file from the following directory:

   `<ADAPTER_HOME>\wsdls`

5. Click **OK**.

   The Localize Files dialog is displayed, as shown in Figure 4–17.

Figure 4–17 Localize Files Dialog


Localize Files

file:/C:/12c_SOA/soa/soa/thirdparty/ApplicationAdapters/wsdls/J2CA_Outbound_invoke.wsdl is external to the current project. In order to make this file available to your project at runtime, JDeveloper can now make a local copy of this file and any dependent files that it imports or includes.

Copy Options: [✓] Maintain original directory structure for imported files

The following files will be created in directory
C:\Jdeveloper\WORK\mywork\SOA_Application\J2CA_Outbound\SOA :

WSDLs/J2CA_Outbound_invoke.wsdl
WSDLs/J2CA_Outbound_invoke_request.xsd
WSDLs/J2CA_Outbound_invoke_response.xsd

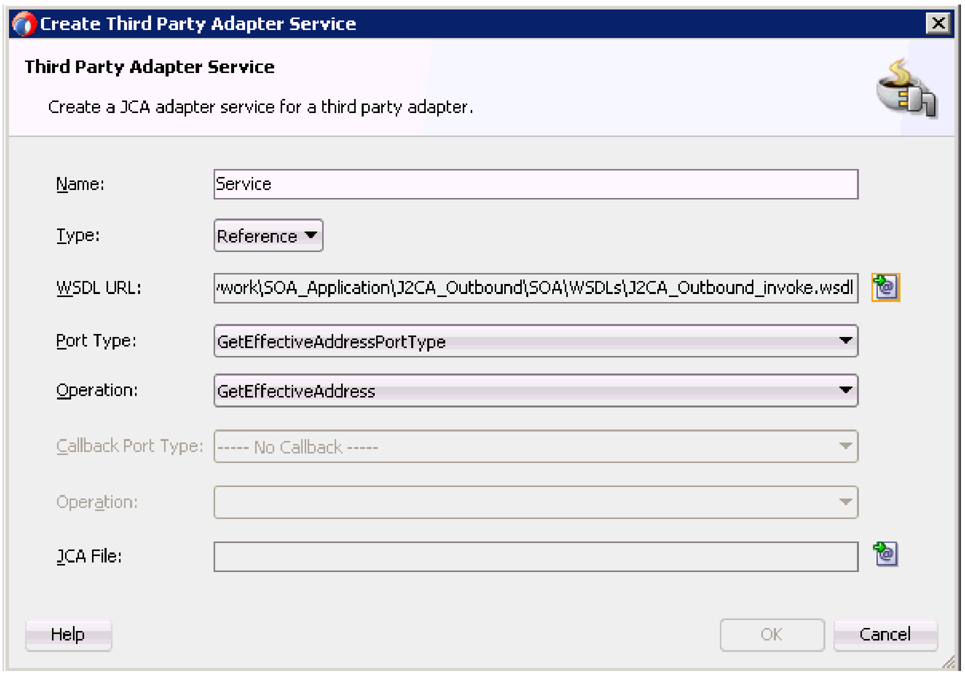
Help | OK | Cancel

6. Click **OK**.

   The outbound WSDL file and associated request and response XML schema files (.xsd) are imported to the project folder that has been created.

   You are returned to the Create Third Party Adapter Service dialog, as shown in Figure 4–18.

Figure 4–18 Create Third Party Adapter Service Dialog


Create Third Party Adapter Service

Third Party Adapter Service

Create a JCA adapter service for a third party adapter.

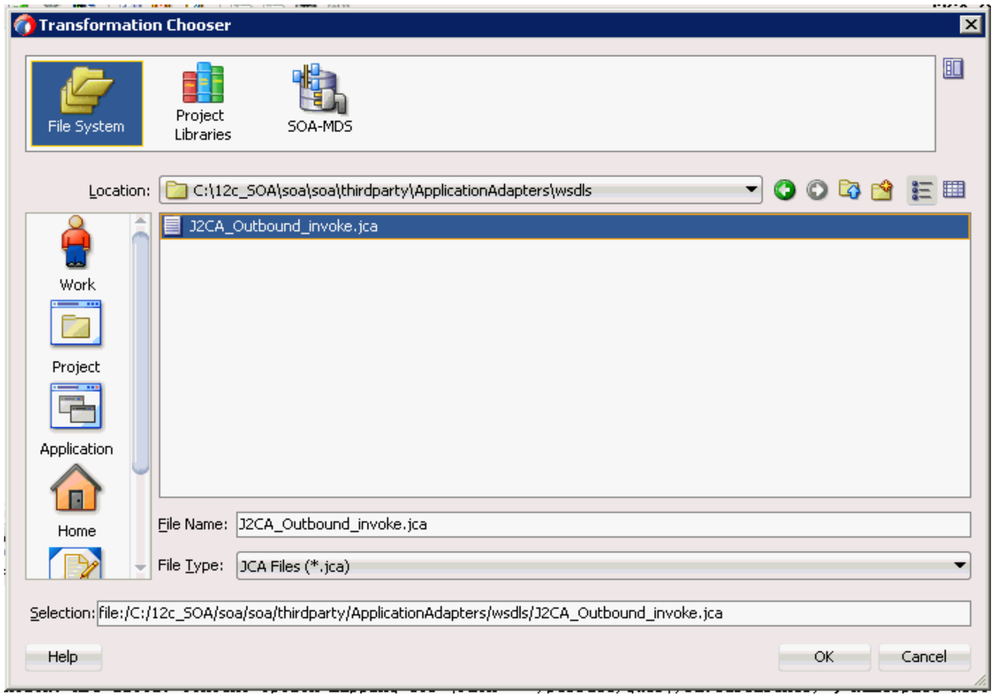
Name: Service
Type: Reference
WSDL URL: work\SOA_Application\J2CA_Outbound\SOA\WSDLs\J2CA_Outbound_invoke.wsdl
Port Type: GetEffectiveAddressPortType
Operation: GetEffectiveAddress
Callback Port Type: ----- No Callback -----
Operation:
JCA File:

Help | OK | Cancel

7. Click the **Find JCA file** icon, which is located to the right of the JCA File field.

The Transformation Chooser dialog is displayed, as shown in Figure 4–19.

Figure 4–19 Transformation Chooser Dialog



Location: C:\12c_SOA\soa\soa\thirdparty\ApplicationAdapters\wsdls

J2CA_Outbound_invoke.jca

File Name: J2CA_Outbound_invoke.jca

File Type: JCA Files (*.jca)

Selection: file:/C:/12c_SOA/soa/soa/thirdparty/ApplicationAdapters/wsdls/J2CA_Outbound_invoke.jca

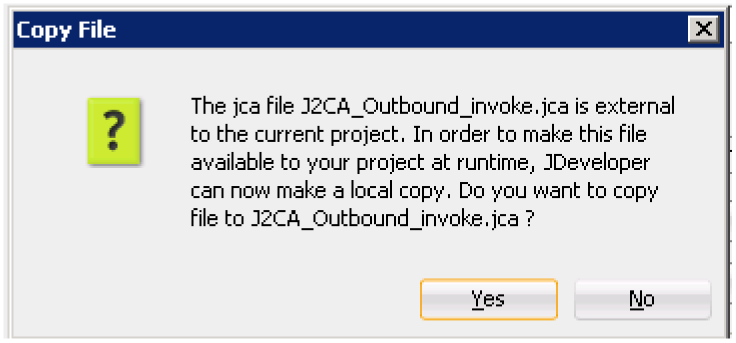
8. Browse and select the JCA properties file from the following directory:

   `<ADAPTER_HOME>\wsdls`

9. Click **OK**.

   The Copy File message is displayed, as shown in Figure 4–20.

Figure 4–20 Copy File Message



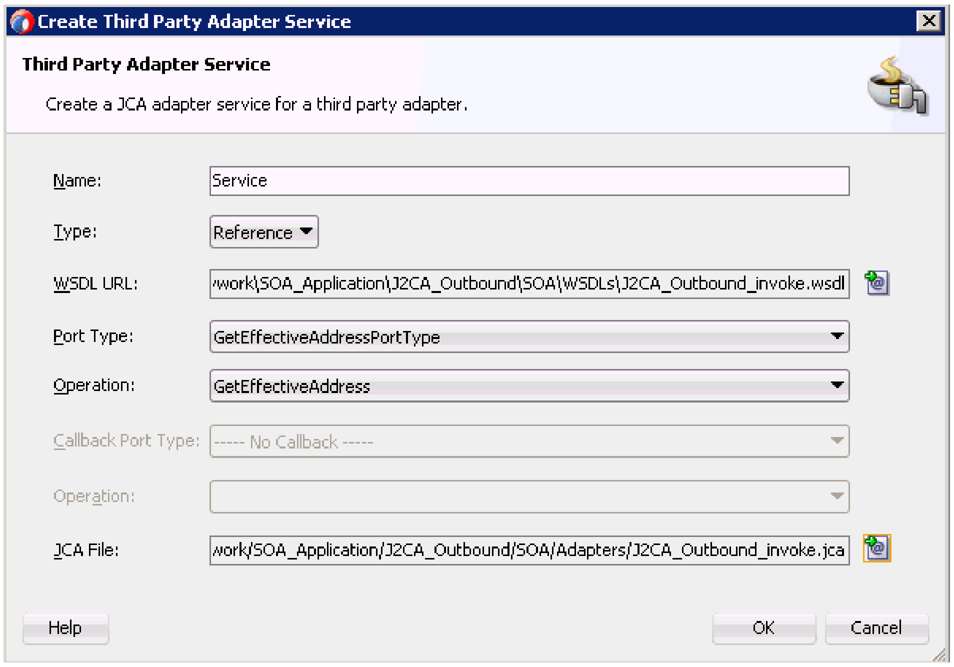
The jca file J2CA_Outbound_invoke.jca is external to the current project. In order to make this file available to your project at runtime, JDeveloper can now make a local copy. Do you want to copy file to J2CA_Outbound_invoke.jca ?

10. Click **Yes**.

    A copy of the JCA properties file is made in the project folder.

    You are returned to the Create Third Party Adapter Service dialog, as shown in Figure 4–21.

Figure 4–21 Create Third Party Adapter Service Dialog


Create Third Party Adapter Service

Third Party Adapter Service

Create a JCA adapter service for a third party adapter.

Name: Service

Type: Reference

WSDL URL: /work\SOA_Application\J2CA_Outbound\SOA\WSDLs\J2CA_Outbound_invoke.wsdl

Port Type: GetEffectiveAddressPortType

Operation: GetEffectiveAddress

Callback Port Type: ----- No Callback -----

Operation:

JCA File: work/SOA_Application/J2CA_Outbound/SOA/Adapters/J2CA_Outbound_invoke.jca

11. Click **OK**.

    The third party adapter service component is created and displayed in the External References pane.

    You are now ready to configure an outbound BPEL process component.

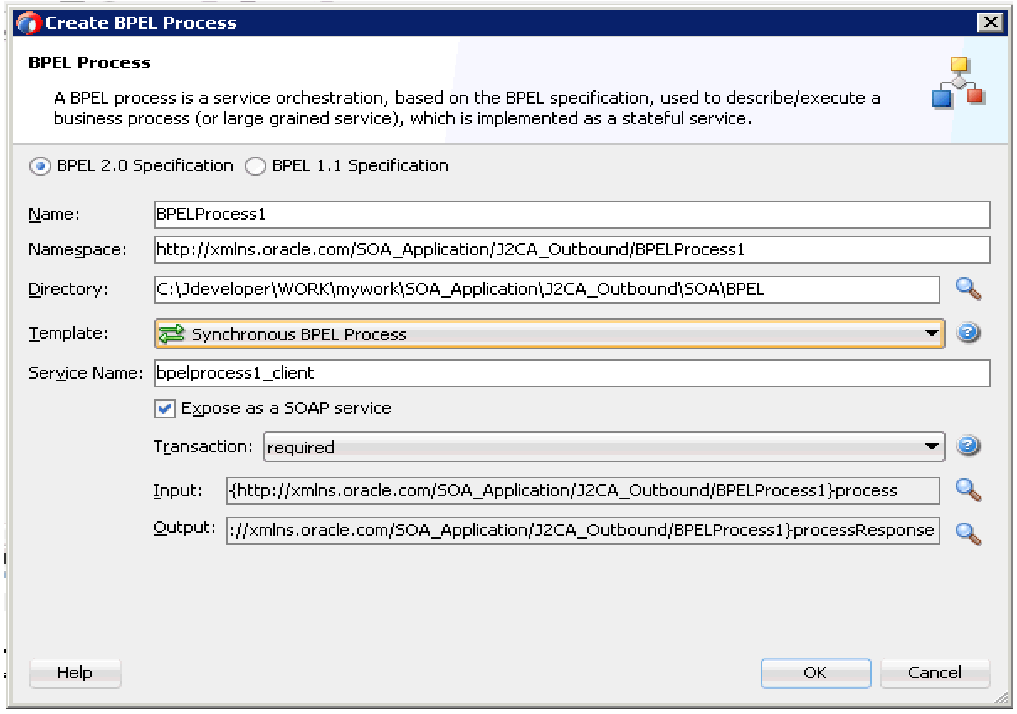
4.4.3.2 Configuring an Outbound BPEL Process Component

Perform the following steps to configure an outbound BPEL process component:

1. Drag and drop the **BPEL Process** component from the Components pane to the Components pane.

   The Create BPEL Process dialog is displayed, as shown in Figure 4–22.

Figure 4–22 Create BPEL Process Dialog



BPEL Process

A BPEL process is a service orchestration, based on the BPEL specification, used to describe/execute a business process (or large grained service), which is implemented as a stateful service.

[✓ BPEL 2.0 Specification] BPEL 1.1 Specification

Name:	BPELProcess1
Namespace:	http://xmlns.oracle.com/SOA_Application/J2CA_Outbound/BPELProcess1
Directory:	C:\Jdeveloper\WORK\mywork\SOA_Application\J2CA_Outbound\SOA\BPEL
Template:	Synchronous BPEL Process
Service Name:	bpelprocess1_client

[✓] Expose as a SOAP service

Transaction:	required
Input:	{http://xmlns.oracle.com/SOA_Application/J2CA_Outbound/BPELProcess1}process
Output:	://xmlns.oracle.com/SOA_Application/J2CA_Outbound/BPELProcess1}processResponse

2. In the Name field, enter a name to identify the new outbound BPEL process component or leave it to the default value.

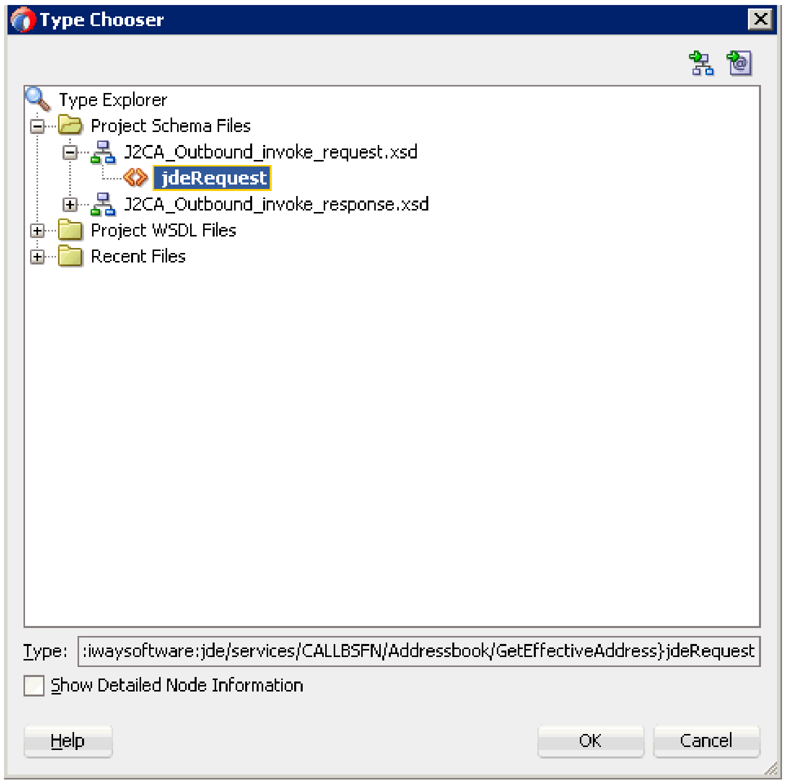
   By default, the BPEL 2.0 Specification option is selected.

3. From the Template list, select **Synchronous BPEL Process**.

4. Click the **Browse** icon, which is located to the right of the Input field to select the associated XML request schema file.

   The Type Chooser dialog is displayed, as shown in Figure 4–23.

Figure 4–23 Type Chooser Dialog



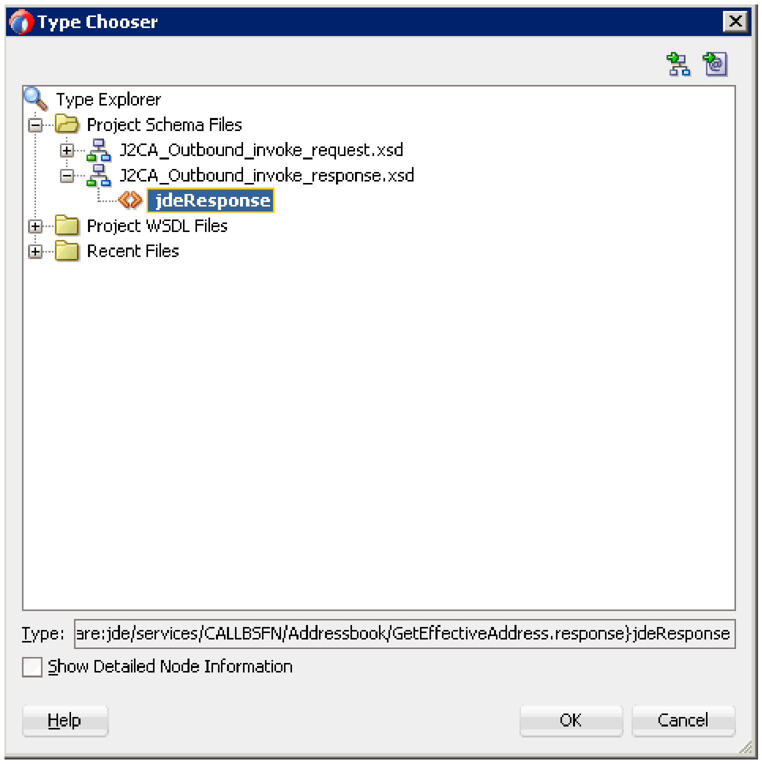
Type: :iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress}jdeRequest

5. Expand **Project Schema Files**, **J2CA_Outbound_invoke_request.xsd**, and select **jdeRequest**.
6. Click **OK**.

   You are returned to the Create BPEL Process dialog.
7. Click the **Browse** icon, which is located to the right of the Output field to select the associated XML response schema file.

   The Type Chooser dialog is displayed, as shown in Figure 4–24.

Figure 4–24 Type Chooser Dialog



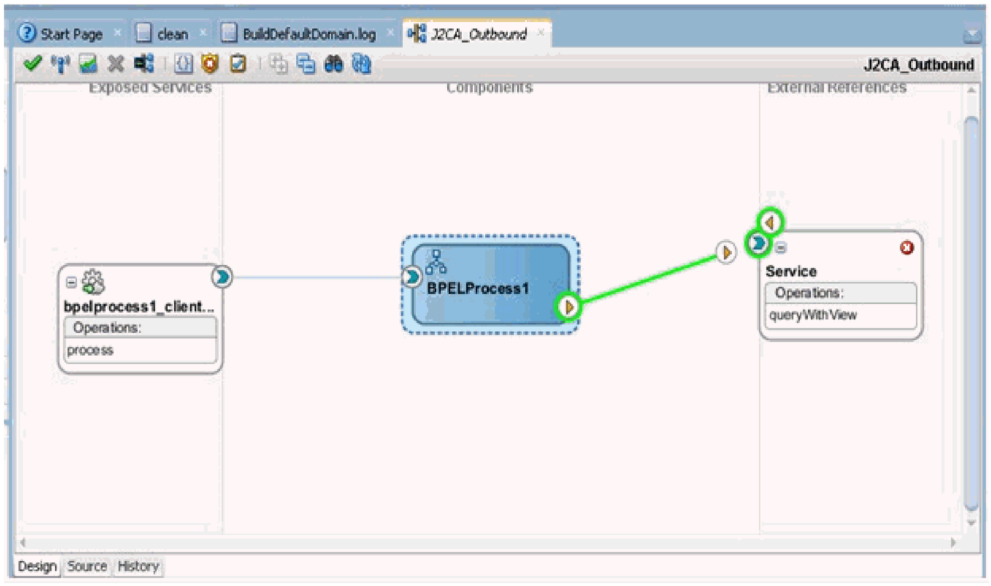
Type: are:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress.response}jdeResponse

☐ Show Detailed Node Information

8. Expand **Project Schema Files**, **J2CA_Outbound_invoke_response.xsd**, and select **jdeResponse**.
9. Click **OK**.

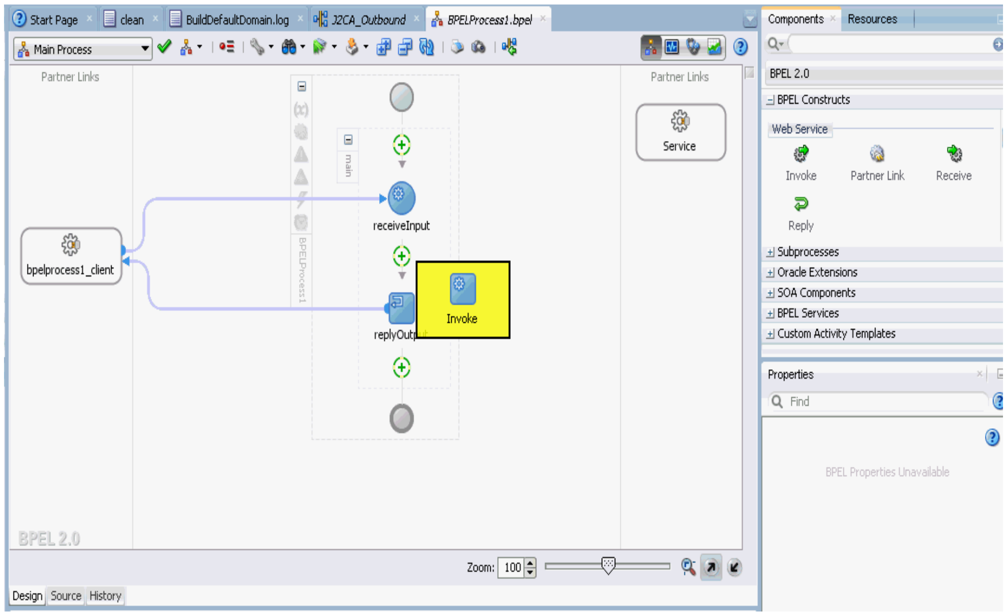
   You are returned to the Create BPEL Process dialog.
10. Click **OK**.
11. Create a connection between the outbound BPEL process component and the third party adapter service component, as shown in Figure 4–25.

Figure 4–25 Created Connection



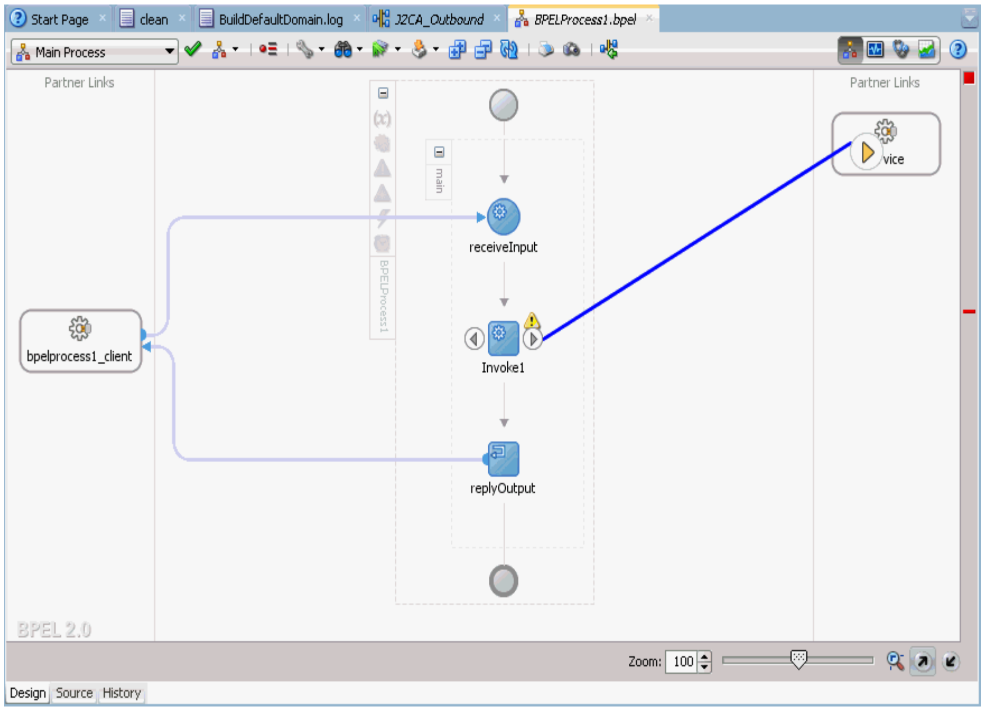
12. Double-click the outbound BPEL process component in the Components pane.
13. Drag and drop the **Invoke** activity component under BPEL Constructs - Web Service, to the Components pane and place it between the **receiveInput** activity component and the **replyOutput** activity component, as shown in Figure 4–26.

Figure 4–26 Invoke Activity Component



14. Create a connection between the new Invoke activity component Service and the third party adapter service component (Service), as shown in Figure 4–27.

Figure 4–27 Created Connection



The Edit Invoke dialog is displayed.

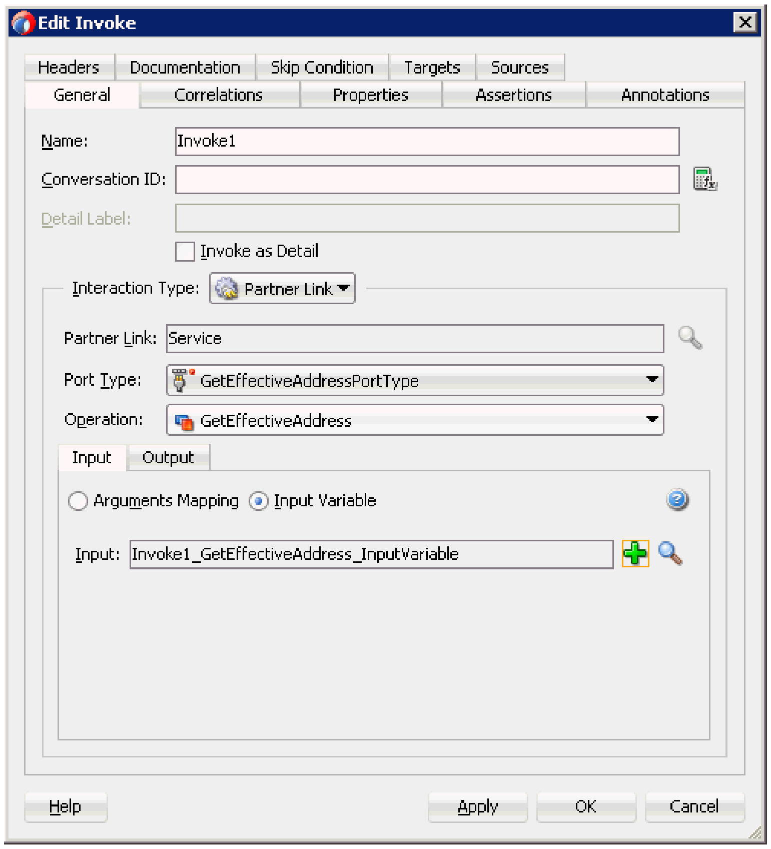
15. Click the **Plus sign** icon, which is located to the right of the Input field to configure a new input variable.

    The Create Variable dialog is displayed.

16. Accept the default values that are provided for the new input variable and click **OK**.

    You are returned to the Edit Invoke dialog, as shown in Figure 4–28.

Figure 4–28 Edit Invoke Dialog



Name: Invoke1
Conversation ID:
Detail Label:
Invoke as Detail
Interaction Type: Partner Link
Partner Link: Service
Port Type: GetEffectiveAddressPortType
Operation: GetEffectiveAddress

Arguments Mapping [✓ Input Variable]
Input: Invoke1_GetEffectiveAddress_InputVariable

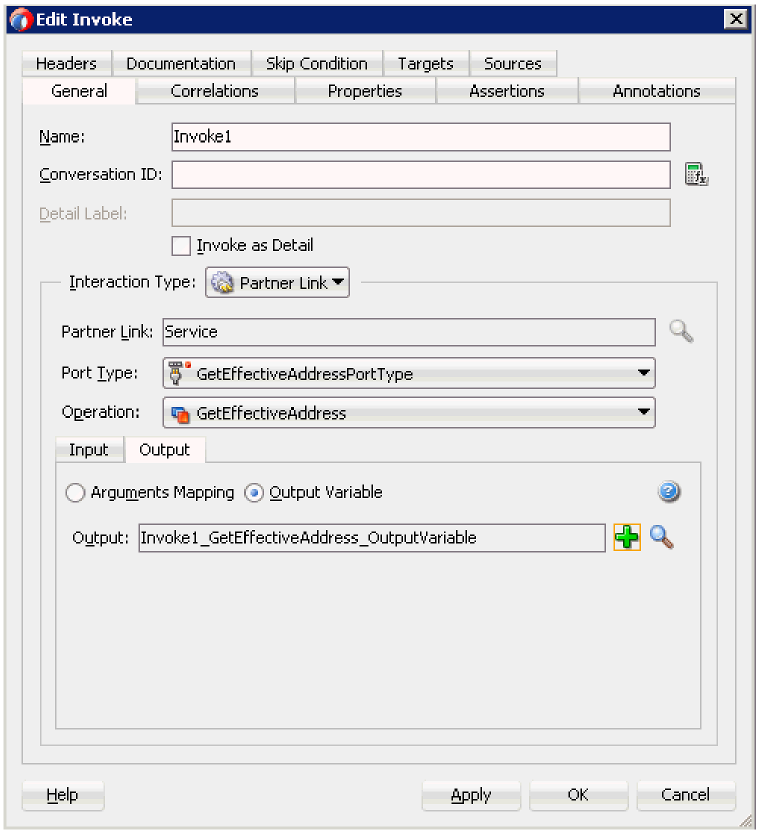
17. Select the **Output** tab and click the **Plus sign** icon, which is located to the right of the Output field to configure a new output variable.

    The Create Variable dialog is displayed.

18. Accept the default values that are provided for the new output variable and click **OK**.

    You are returned to the Edit Invoke dialog, as shown in Figure 4–29.

Figure 4–29 Edit Invoke Dialog


Edit Invoke

Name: Invoke1

Conversation ID:

Detail Label:

[ ] Invoke as Detail

Interaction Type: Partner Link

Partner Link: Service

Port Type: GetEffectiveAddressPortType

Operation: GetEffectiveAddress

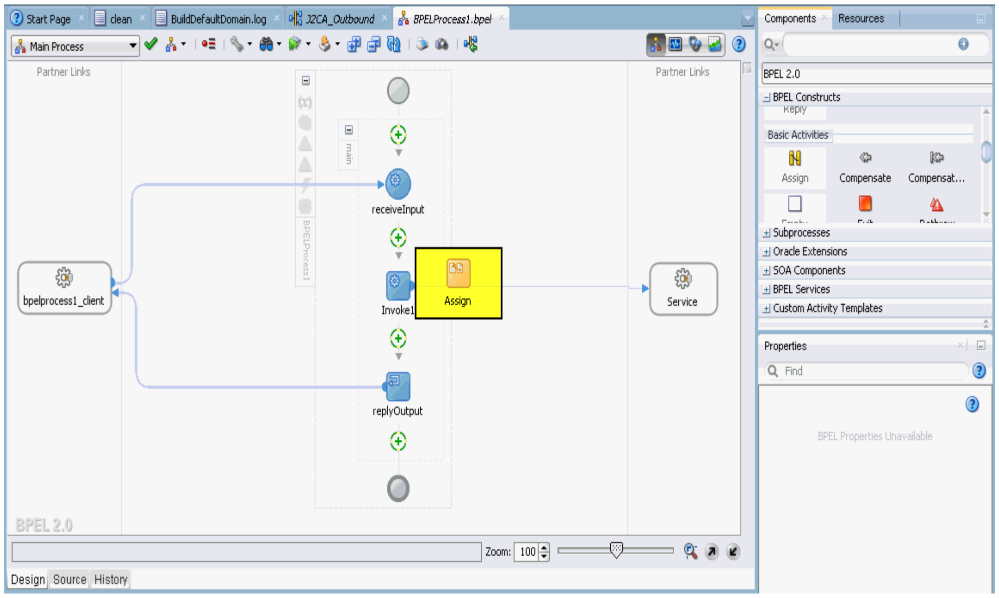
( ) Arguments Mapping [✓ Output Variable]

Output: Invoke1_GetEffectiveAddress_OutputVariable

19. Click **Apply** and then **OK**.

20. Drag and drop the **Assign** activity under BPEL Constructs - Basic Activities component, to the Components pane and place it between the Receive activity component (receiveInput) and the Invoke activity component (Invoke1), as shown in Figure 4–30.

Figure 4–30 Assign Activity Component



21. Double-click the new Assign activity component (**Assign1**).

    The Edit Assign dialog is displayed.

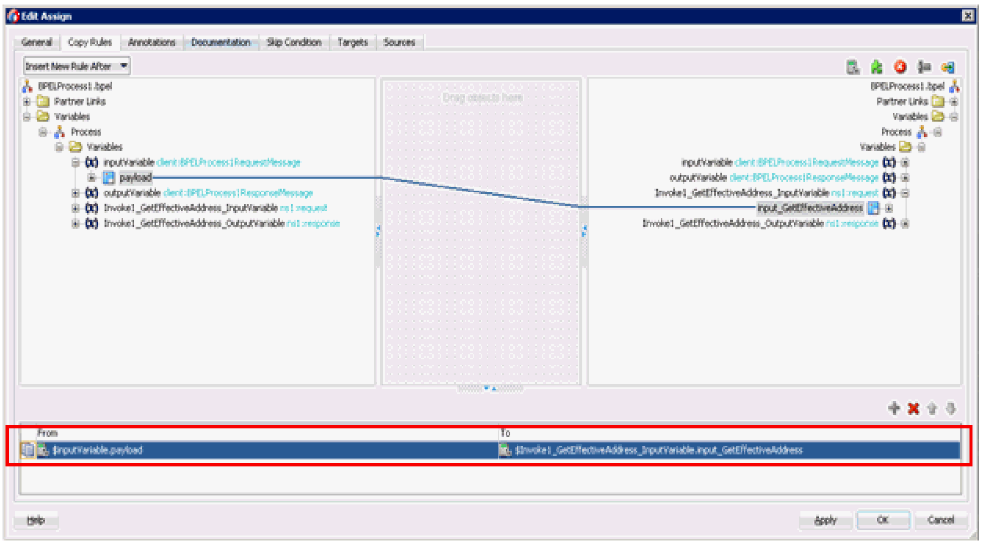
22. In the left pane, under Variables, expand **InputVariable**, and then select **payload**.

23. In the right pane, under Variables, expand **Invoke1_GetEffectiveAddress_InputVariable**, and then select **input_GetEffectiveAddress**.

24. Drag and map the **payload** variable to the **input_GetEffectiveAddress** variable.

    The mapped variables are populated in the highlighted area as shown in Figure 4–31.

Figure 4–31 Edit Assign Dialog


Edit Assign
General
Copy Rules
Annotations
Documentation
Skip Condition
Targets
Sources
Insert New Rule After
BPELProcess1.bpel
Partner Links
Variables
Process
Variables
inputVariable client:BPELProcess1RequestMessage
payload
outputVariable client:BPELProcess1ResponseMessage
Invoke1_GetEffectiveAddress_InputVariable ns1:request
Invoke1_GetEffectiveAddress_OutputVariable ns1:response
Drag objects here
BPELProcess1.bpel
Partner Links
Variables
Process
Variables
inputVariable client:BPELProcess1RequestMessage
outputVariable client:BPELProcess1ResponseMessage
Invoke1_GetEffectiveAddress_InputVariable ns1:request
input_GetEffectiveAddress
Invoke1_GetEffectiveAddress_OutputVariable ns1:response
Help
Apply
OK
Cancel

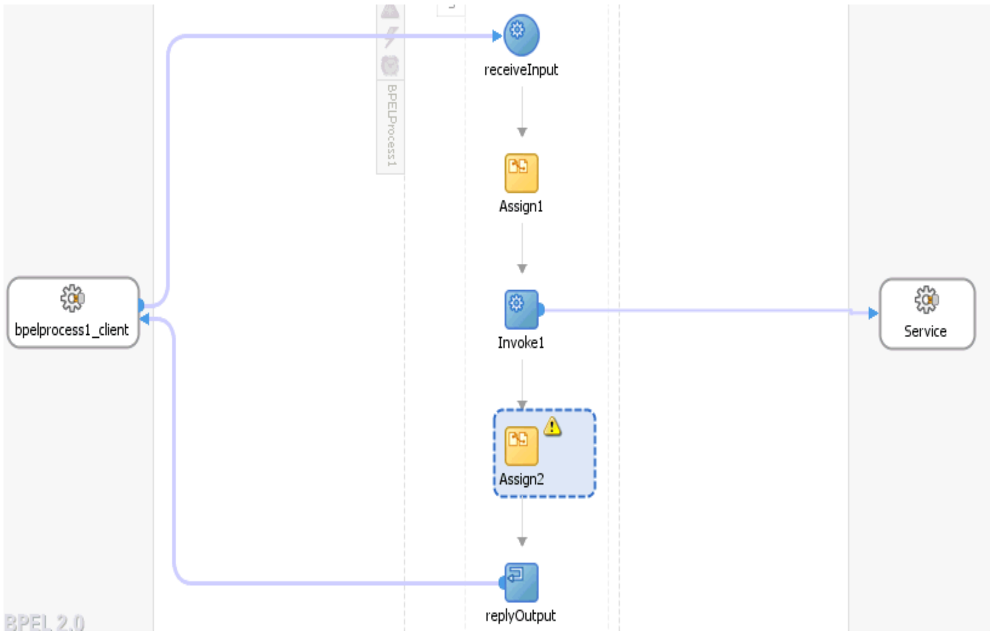
From	To
$inputVariable.payload	$Invoke1_GetEffectiveAddress_InputVariable.input_GetEffectiveAddress

25. Click **Apply** and then **OK**.

26. Drag and drop the **Assign** activity component to the Components pane and place it between the Invoke activity (Invoke1) and the Reply activity (replyOutput).

27. Double-click the new Assign activity component (**Assign2**), as shown in Figure 4–32.

Figure 4–32 New Assign Activity Component



The Edit Assign dialog is displayed.

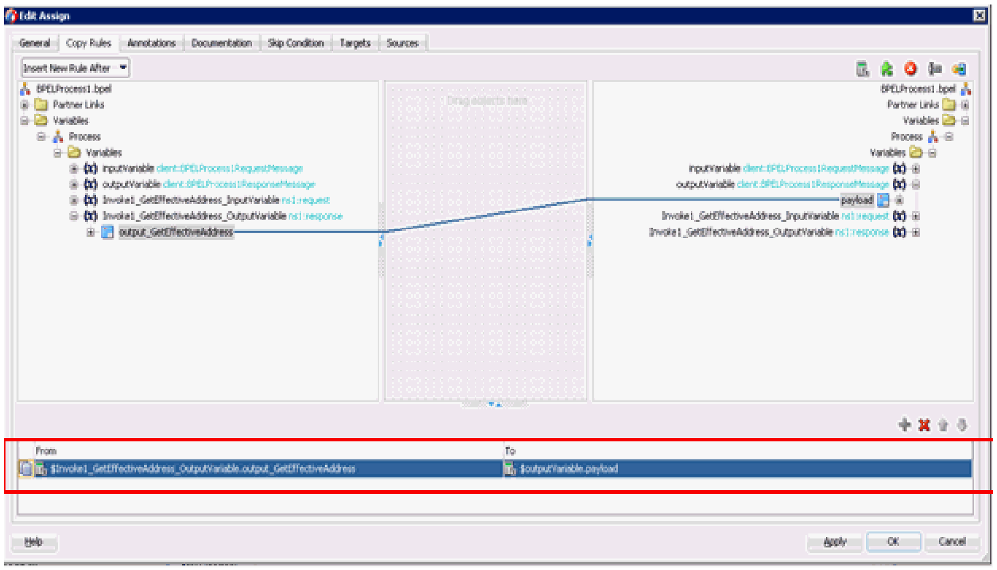
28. In the left pane, under Variables, expand **Invoke1_GetEffectiveAddress_OutputVariable**, and then select **output_GetEffectiveAddress**.

29. In the right pane, under Variables, expand **outputVariable** and select **payload**.

30. Drag and map the **output_GetEffectiveAddress** variable to the **payload** variable.

    The mapped variables are populated in the highlighted area as shown in Figure 4–33.

Figure 4–33 Edit Assign Dialog

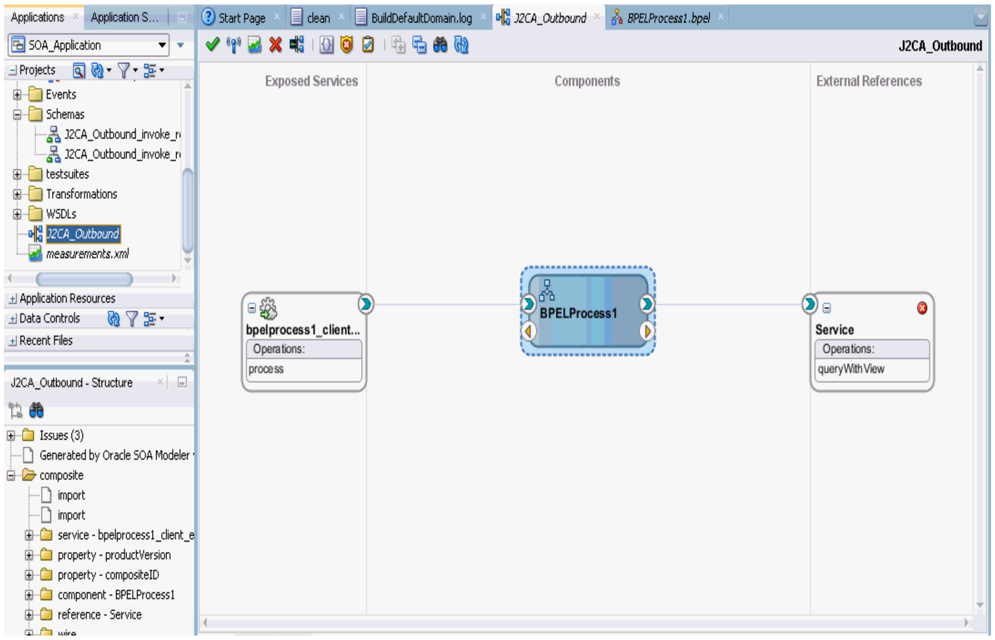

Edit Assign
General
Copy Rules
Annotations
Documentation
Skip Condition
Targets
Sources
Insert New Rule After
BPELProcess1.bpel
Partner Links
Variables
Process
Variables
inputVariable client:BPELProcess1RequestMessage
outputVariable client:BPELProcess1ResponseMessage
Invoke1_GetEffectiveAddress_InputVariable ns1:request
Invoke1_GetEffectiveAddress_OutputVariable ns1:response
output_GetEffectiveAddress
Drag objects here
BPELProcess1.bpel
Partner Links
Variables
Process
Variables
inputVariable client:BPELProcess1RequestMessage
outputVariable client:BPELProcess1ResponseMessage
payload
Invoke1_GetEffectiveAddress_InputVariable ns1:request
Invoke1_GetEffectiveAddress_OutputVariable ns1:response
Help
Apply
OK
Cancel

From	To
$Invoke1_GetEffectiveAddress_OutputVariable.output_GetEffectiveAddress	$outputVariable.payload

31. Click **Apply** and then **OK**.

    You are returned to the Activity component pane, as shown in Figure 4–34.

Figure 4–34 Activity Component Pane



32. Click the **Save All** icon in the menu bar to save the new outbound BPEL process component that was configured.

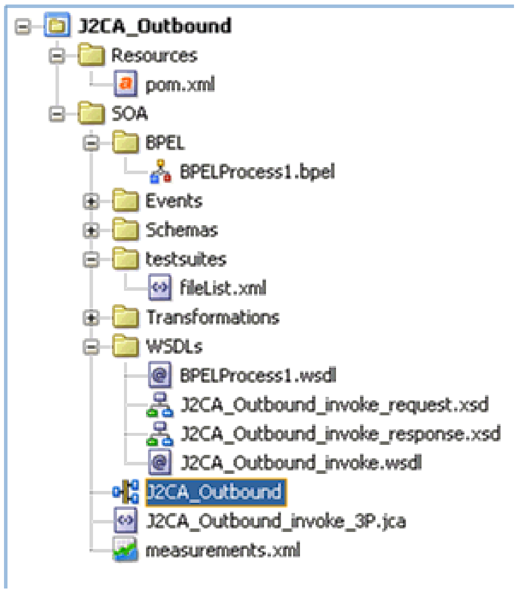
    You are now ready to deploy the BPEL outbound process.

4.4.3.3 Adjusting for Known Deployment Issues With 12c

Perform the following steps to adjust for known deployment issues with 12c.

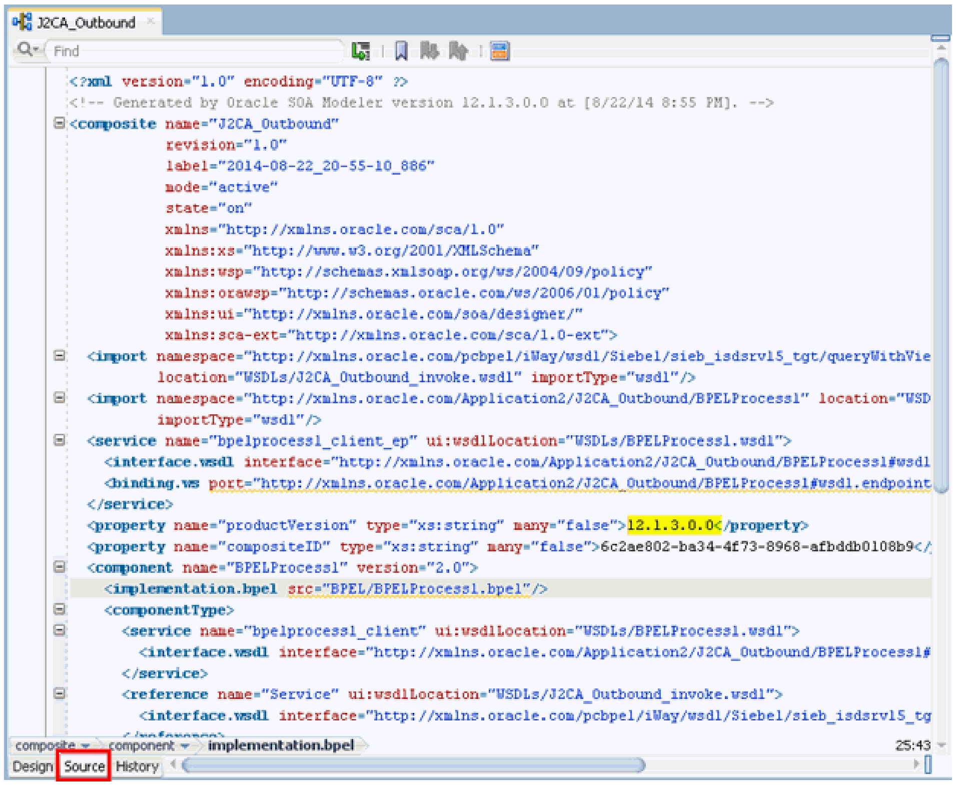
1. Double-click **J2CA_Outbound** (created BPEL process) of the created process, as shown in Figure 4–35.

Figure 4–35 J2CA_Outbound Node



2. Click the **Source** tab below the opened process, as shown in Figure 4–36.

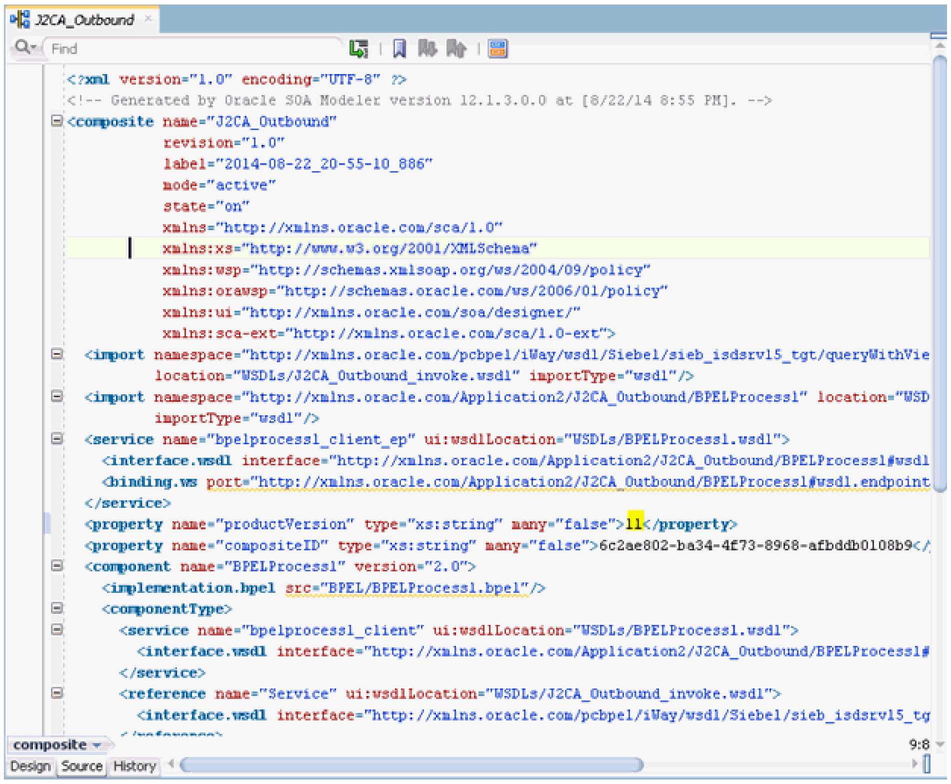
Figure 4–36 Source Tab



```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 12.1.3.0.0 at [8/22/14 8:55 PM]. -->
<composite name="J2CA_Outbound"
           revision="1.0"
           label="2014-08-22_20-55-10_886"
           mode="active"
           state="on"
           xmlns="http://xmlns.oracle.com/sca/1.0"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
           xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
           xmlns:ui="http://xmlns.oracle.com/soa/designer/"
           xmlns:sca-ext="http://xmlns.oracle.com/sca/1.0-ext">
  <import namespace="http://xmlns.oracle.com/pcbpel/iWay/wsdl/Siebel/sieb_isdsrv15_tgt/queryWithVie
          location="WSDLs/J2CA_Outbound_invoke.wsdl" importType="wsdl"/>
  <import namespace="http://xmlns.oracle.com/Application2/J2CA_Outbound/BPELProcess1" location="WSD
          importType="wsdl"/>
  <service name="bpelprocess1_client_ep" ui:wsdlLocation="WSDLs/BPELProcess1.wsdl">
    <interface.wsdl interface="http://xmlns.oracle.com/Application2/J2CA_Outbound/BPELProcess1#wsdl
    <binding.ws port="http://xmlns.oracle.com/Application2/J2CA_Outbound/BPELProcess1#wsdl.endpoint
  </service>
  <property name="productVersion" type="xs:string" many="false">12.1.3.0.0</property>
  <property name="compositeID" type="xs:string" many="false">6c2ae802-ba34-4f73-8968-afbddb0108b9</
  <component name="BPELProcess1" version="2.0">
    <implementation.bpel src="BPEL/BPELProcess1.bpel"/>
    <componentType>
      <service name="bpelprocess1_client" ui:wsdlLocation="WSDLs/BPELProcess1.wsdl">
        <interface.wsdl interface="http://xmlns.oracle.com/Application2/J2CA_Outbound/BPELProcess1#
      </service>
      <reference name="Service" ui:wsdlLocation="WSDLs/J2CA_Outbound_invoke.wsdl">
        <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/iWay/wsdl/Siebel/sieb_isdsrv15_tg
```

3. Change the productVersion property value from `12.1.3.0.0` to `11`, as shown in Figure 4–37.

Figure 4–37 Property Value



```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Generated by Oracle SOA Modeler version 12.1.3.0.0 at [8/22/14 8:55 PM]. -->
<composite name="J2CA_Outbound"
           revision="1.0"
           label="2014-08-22_20-55-10_886"
           mode="active"
           state="on"
           xmlns="http://xmlns.oracle.com/sca/1.0"
           xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"
           xmlns:orawsp="http://schemas.oracle.com/ws/2006/01/policy"
           xmlns:ui="http://xmlns.oracle.com/soa/designer/"
           xmlns:sca-ext="http://xmlns.oracle.com/sca/1.0-ext">
  <import namespace="http://xmlns.oracle.com/pcbpel/iWay/wsdl/Siebel/sieb_isdsrv15_tgt/queryWithVie
          location="WSDLs/J2CA_Outbound_invoke.wsdl" importType="wsdl"/>
  <import namespace="http://xmlns.oracle.com/Application2/J2CA_Outbound/BPELProcess1" location="WSD
          importType="wsdl"/>
  <service name="bpelprocess1_client_ep" ui:wsdlLocation="WSDLs/BPELProcess1.wsdl">
    <interface.wsdl interface="http://xmlns.oracle.com/Application2/J2CA_Outbound/BPELProcess1#wsdl
    <binding.ws port="http://xmlns.oracle.com/Application2/J2CA_Outbound/BPELProcess1#wsdl.endpoint
  </service>
  <property name="productVersion" type="xs:string" many="false">11</property>
  <property name="compositeID" type="xs:string" many="false">6c2ae802-ba34-4f73-8968-afbddb0108b9</
  <component name="BPELProcess1" version="2.0">
    <implementation.bpel src="BPEL/BPELProcess1.bpel"/>
    <componentType>
      <service name="bpelprocess1_client" ui:wsdlLocation="WSDLs/BPELProcess1.wsdl">
        <interface.wsdl interface="http://xmlns.oracle.com/Application2/J2CA_Outbound/BPELProcess1#
      </service>
      <reference name="Service" ui:wsdlLocation="WSDLs/J2CA_Outbound_invoke.wsdl">
        <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/iWay/wsdl/Siebel/sieb_isdsrv15_tg
```

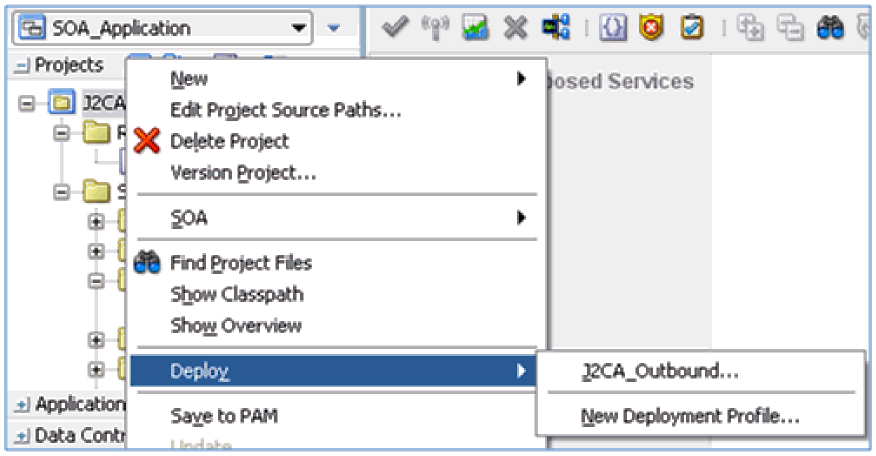
4. Save the changes and proceed to deploy the project.

4.4.4 Deploying the BPEL Outbound Process

Perform the following steps to deploy the BPEL outbound process.

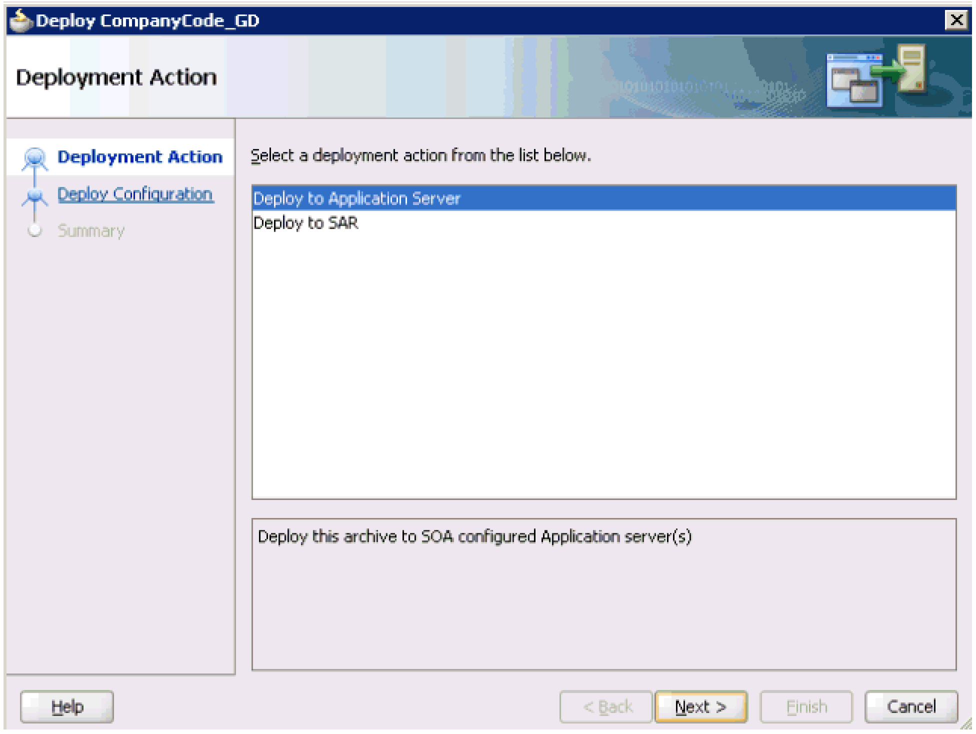
1. Right-click the project name in the left pane, select **Deploy**, and then click **J2CA_ Outbound**, as shown in Figure 4–38.

Figure 4–38 J2CA_Outbound Option



The Deployment Action page is displayed, as shown in Figure 4–39.

Figure 4–39 Deployment Action Page


Deploy CompanyCode_GD

Deployment Action

Select a deployment action from the list below.

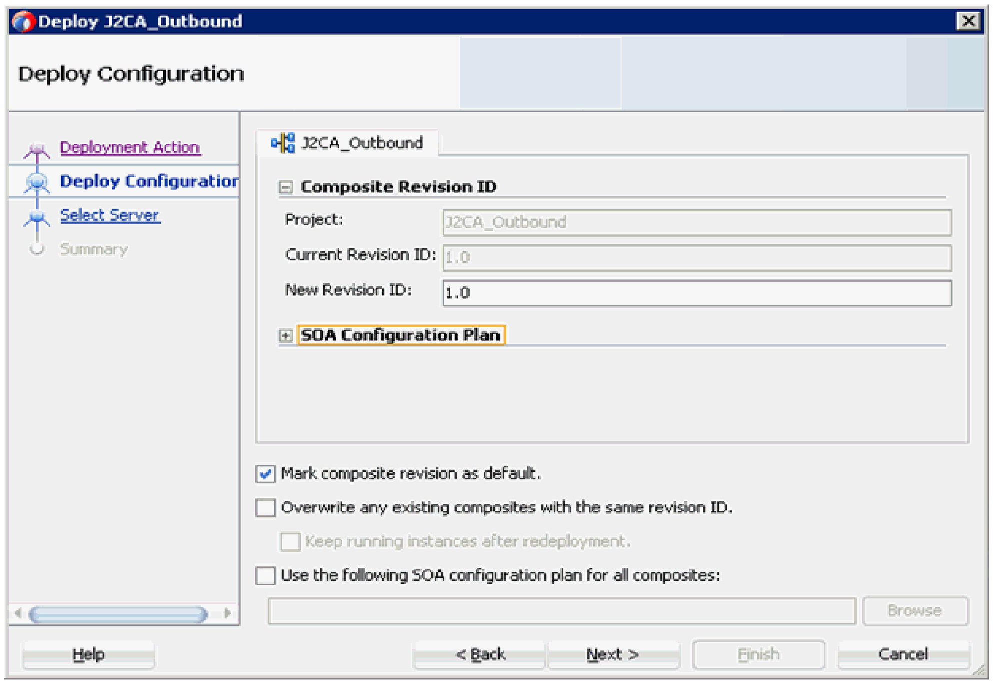
Deploy to Application Server
Deploy to SAR

Deploy this archive to SOA configured Application server(s)

2. Ensure that **Deploy to Application Server** is selected.
3. Click **Next**.

   The Deploy Configuration page is displayed, as shown in Figure 4–40.

Figure 4–40 Deploy Configurations Page


Deploy J2CA_Outbound

Deploy Configuration

J2CA_Outbound

Composite Revision ID

Project: J2CA_Outbound
Current Revision ID: 1.0
New Revision ID: 1.0

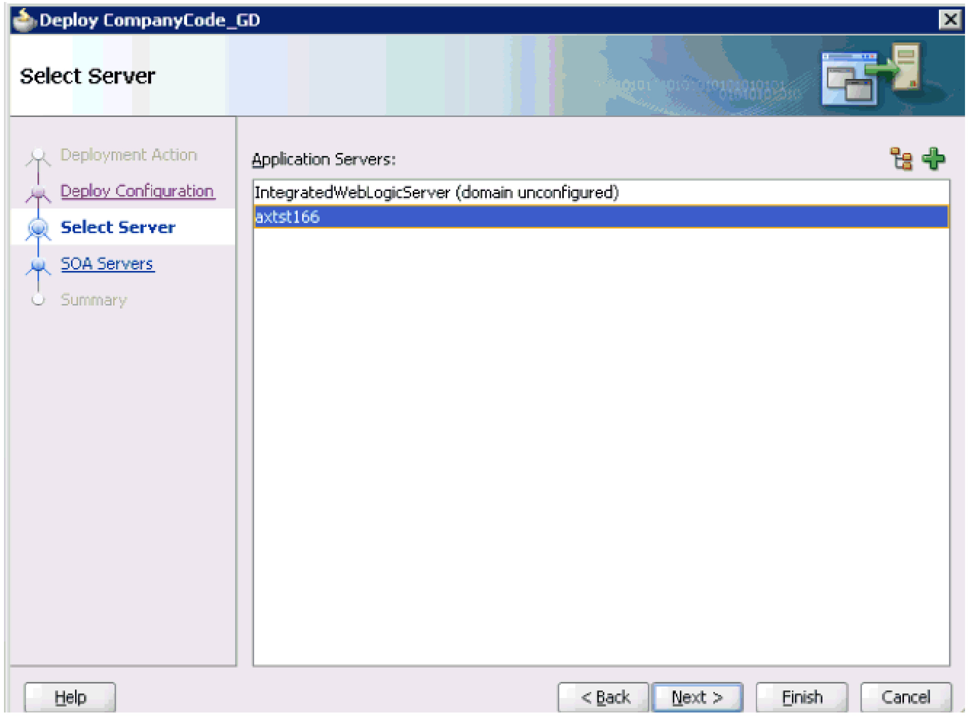
SOA Configuration Plan

[✓] Mark composite revision as default.
[ ] Overwrite any existing composites with the same revision ID.
[ ] Keep running instances after redeployment.
[ ] Use the following SOA configuration plan for all composites:

4. Leave the default values selected and click **Next**.

The Select Server page is displayed, as shown in Figure 4–41.

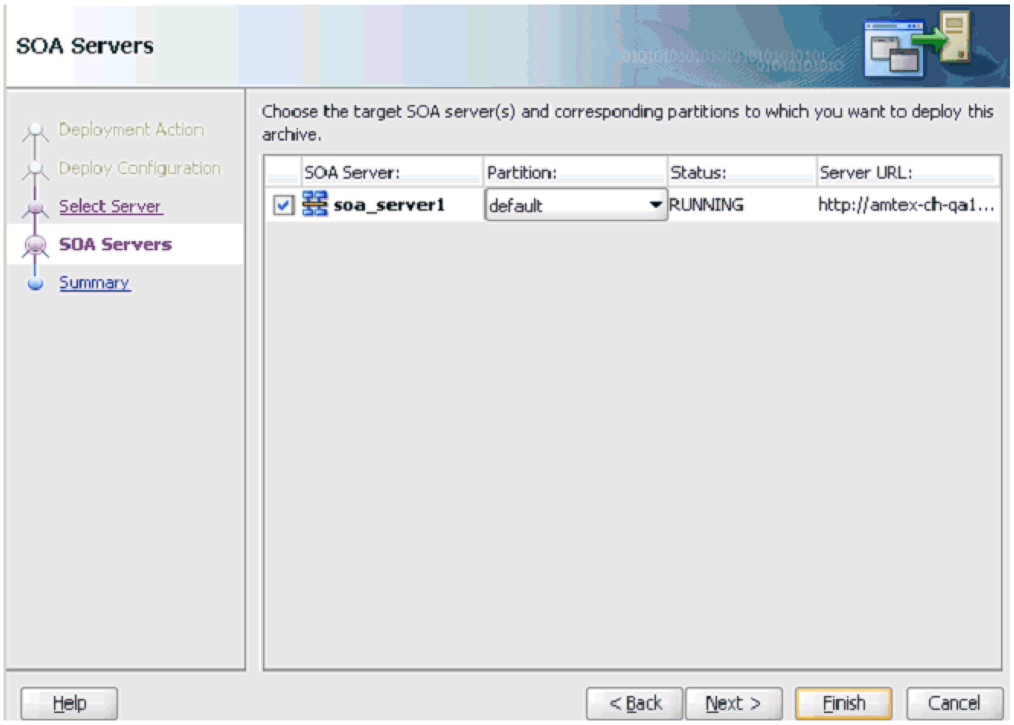
Figure 4–41 Select Server Page



5. Select an available application server that was configured and click **Next**.

   The SOA Servers page is displayed, as shown in Figure 4–42.

Figure 4–42 SOA Servers Page



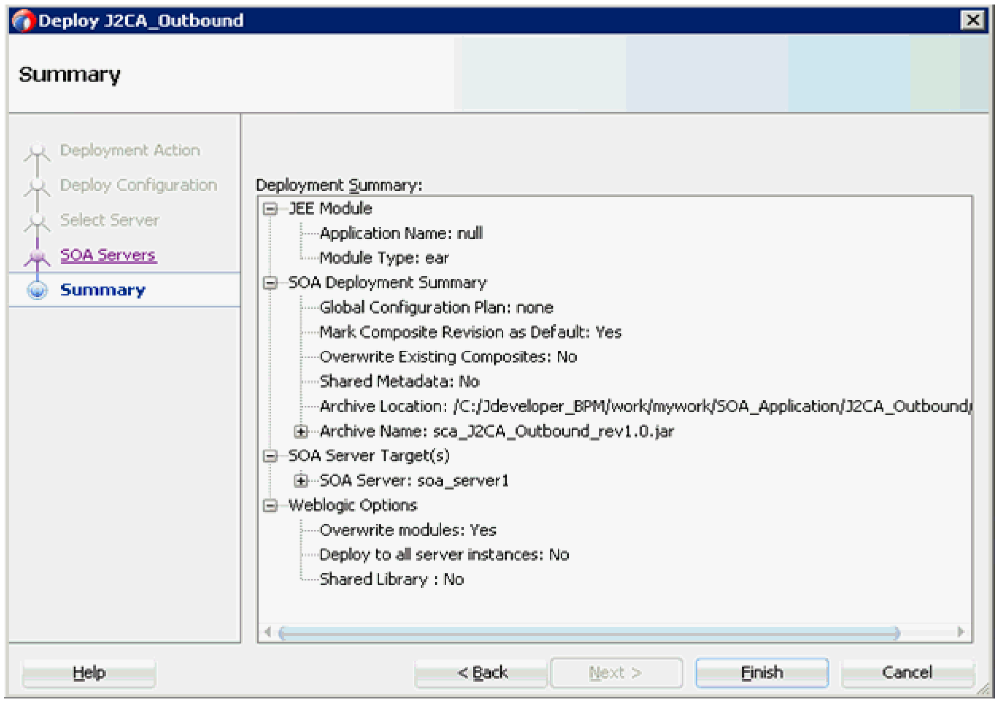
Choose the target SOA server(s) and corresponding partitions to which you want to deploy this archive.

	SOA Server:	Partition:	Status:	Server URL:
✓	soa_server1	default	RUNNING	http://amtex-ch-qa1...

6. Select a target SOA server and click **Next**.

   The Summary page is displayed, as shown in Figure 4–43.

Figure 4–43 Summary Page

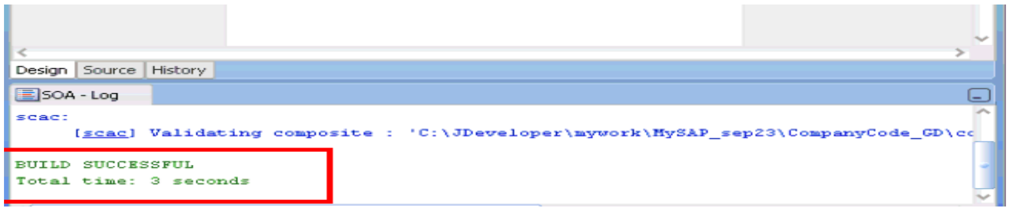


Deployment Summary:
- JEE Module
  - Application Name: null
  - Module Type: ear
- SOA Deployment Summary
  - Global Configuration Plan: none
  - Mark Composite Revision as Default: Yes
  - Overwrite Existing Composites: No
  - Shared Metadata: No
  - Archive Location: /C:/Jdeveloper_BPM/work/mywork/SOA_Application/J2CA_Outbound/
  - Archive Name: sca_J2CA_Outbound_rev1.0.jar
- SOA Server Target(s)
  - SOA Server: soa_server1
- Weblogic Options
  - Overwrite modules: Yes
  - Deploy to all server instances: No
  - Shared Library : No

7. Review and verify all the available deployment information for your project and click **Finish**.

   The process is deployed successfully, as shown in Figure 4–44.

Figure 4–44 Successful Deployment Message



```
scac:
     [scac] Validating composite : 'C:\JDeveloper\mywork\MySAP_sep23\CompanyCode_GD\cc
BUILD SUCCESSFUL
Total time: 3 seconds
```

4.4.5 Invoking the Input XML Document in the Oracle Enterprise Manager Console

Perform the following steps to invoke the input XML document in the Oracle Enterprise Manager console.

1. Logon to the Oracle Enterprise Manager console.

Note: For customers using 12*c* (12.2.1.1.0) and 12c (12.2.1.2.0), perform the following steps:

2. Click **Target Navigation** in the left pane, expand **SOA**, and then select **soa-infra (soa_server1)**.

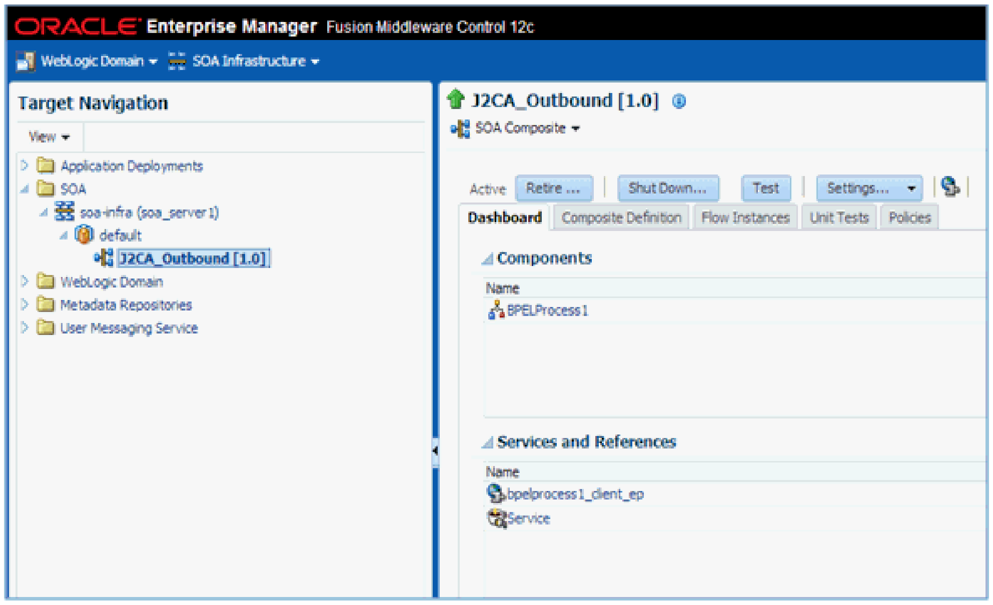
3. Click the **Deployed Composites** tab, which will list all of the deployed composites. Click on the available project (for example, J2CA_Outbound).

Skip to **Step 4** in this procedure.

---

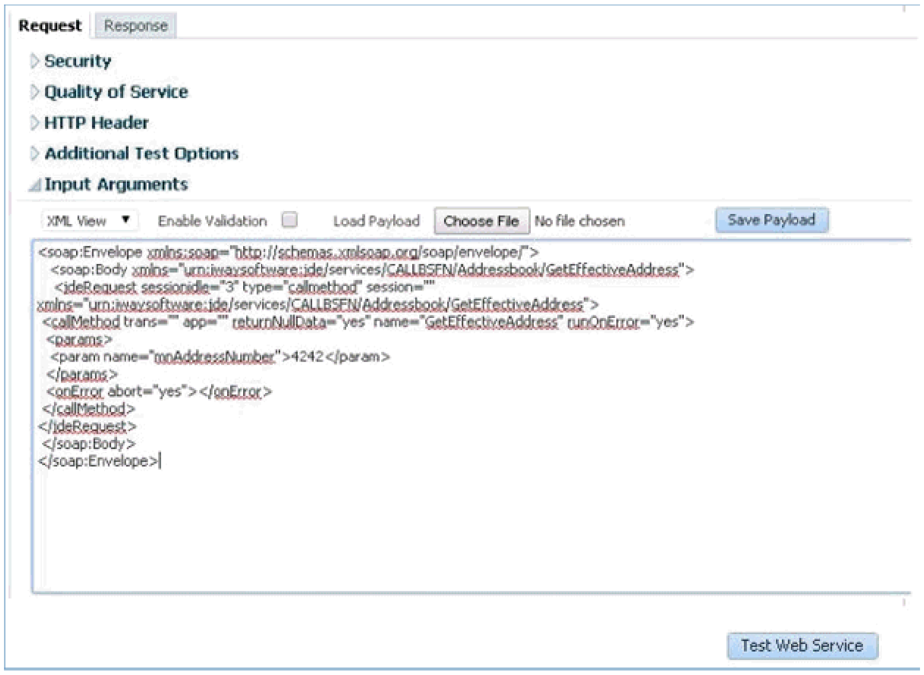
2. Expand **SOA**, select **soa-infra (soa_server1)**, and then click **Default**.
3. Select an available project (for example, J2CA_Outbound) and click **Test** as shown in Figure 4–45.

Figure 4–45 Test Button



4. Click the **Request** tab.
5. Select **XML View** from the list, as shown in Figure 4–46.

Figure 4–46 Input Arguments List

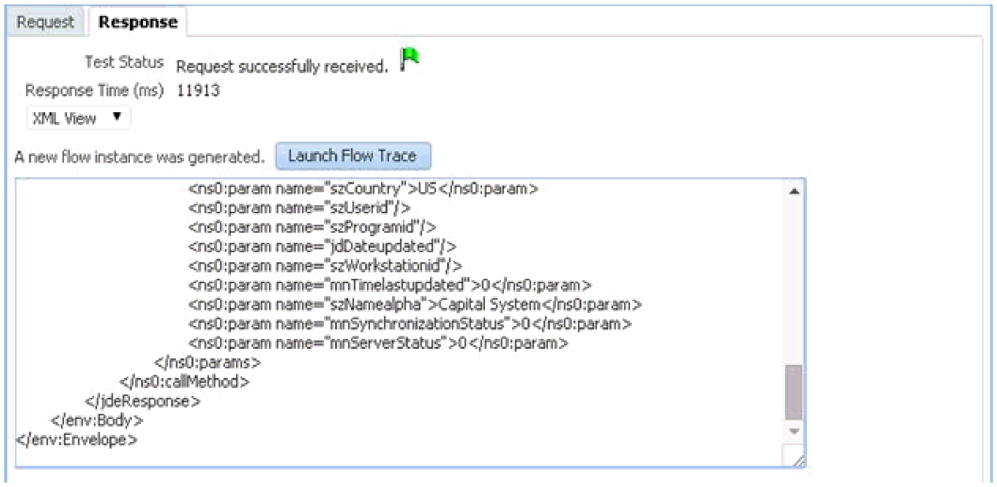


```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body xmlns="urn:iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress">
   <jdeRequest sessionidle="3" type="callmethod" session=""
xmlns="urn:iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress">
 <callMethod trans="" app="" returnNullData="yes" name="GetEffectiveAddress" runOnError="yes">
  <params>
   <param name="mnAddressNumber">4242</param>
  </params>
  <onError abort="yes"></onError>
 </callMethod>
</jdeRequest>
 </soap:Body>
</soap:Envelope>
```

6. Provide an appropriate input XML document in the Input Arguments area and click **Test Web Service**.

The output response is received in the Oracle Enterprise Manager console, as shown in Figure 4–47.

Figure 4–47 Received Output Response



Test Status Request successfully received.
Response Time (ms) 11913

A new flow instance was generated.

```
                    <ns0:param name="szCountry">US</ns0:param>
                    <ns0:param name="szUserid"/>
                    <ns0:param name="szProgramid"/>
                    <ns0:param name="jdDateupdated"/>
                    <ns0:param name="szWorkstationid"/>
                    <ns0:param name="mnTimelastupdated">0</ns0:param>
                    <ns0:param name="szNamealpha">Capital System</ns0:param>
                    <ns0:param name="mnSynchronizationStatus">0</ns0:param>
                    <ns0:param name="mnServerStatus">0</ns0:param>
                </ns0:params>
            </ns0:callMethod>
        </jdeResponse>
    </env:Body>
</env:Envelope>
```

4.4.6 Testing Outbound BPEL and Mediator Processes

When testing an outbound BPEL process or an outbound Mediator process from the Oracle Enterprise Manager console, do not use the XML envelopes that are generated by these consoles. Instead, remove them and use the XML payloads that are generated from the schemas, which conform to the WSDLs for namespace qualifications.

The Mediator data flows can be tested using the Enterprise Manager console. When creating a Mediator data flow and interactions, the Web services are created and registered with the Oracle Application Server. For more information on creating a Mediator outbound process, see Chapter 5, "Integration With Mediator Service Components in the Oracle SOA Suite".

4.5 Designing an Inbound BPEL Process for Event Integration (J2CA Configuration)

This section illustrates how Oracle Application Adapter for J.D. Edwards OneWorld integrates with J.D. Edwards OneWorld to receive event data. The design-time and run-time configuration procedures are outlined in the following sections.

A sample project has been provided for this inbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\BPEL\J2CA\Inbound_Project
```

The following tools are required to complete your adapter design-time configuration:

- Oracle Adapter Application Explorer (Application Explorer)
- Oracle JDeveloper BPEL Designer (JDeveloper)

Note: The examples in this chapter demonstrate the use of Oracle JDeveloper.

This section contains the following topics:

- Section 4.5.1, "Generating WSDL for Event Integration"
- Section 4.5.2, "Creating an Empty Composite for SOA"
- Section 4.5.3, "Defining a BPEL Inbound Process"
- Section 4.5.4, "Deploying the BPEL Inbound Process"
- Section 4.5.5, "Triggering an Event in J.D. Edwards OneWorld"

Before you design a BPEL process, you must generate the respective WSDL file using Application Explorer. For more information, see "Generating WSDL for Event Integration" on page 4-34.

4.5.1 Generating WSDL for Event Integration

Before you design a BPEL process using Oracle JDeveloper, you must create a separate channel for every J2CA event and select that channel when you generate WSDL for inbound interaction using Application Explorer.

Note: If two or more events share the same channel, then event messages may not be delivered to the right BPEL process.

This section contains the following topics:

- Section 4.5.1.1, "Creating a Channel in Application Explorer"
- Section 4.5.1.2, "Generating WSDL for Event Notification (Command Prompt Only)"

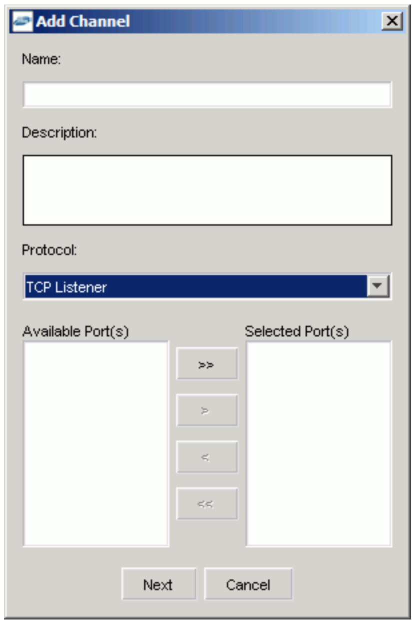
4.5.1.1 Creating a Channel in Application Explorer

To create a channel:

1. In Application Explorer, expand the **JDEdwards** node.
2. Right-click the **Channels** node, and select **Add Channels**.

   The Add Channel dialog is displayed, as shown in Figure 4–48.

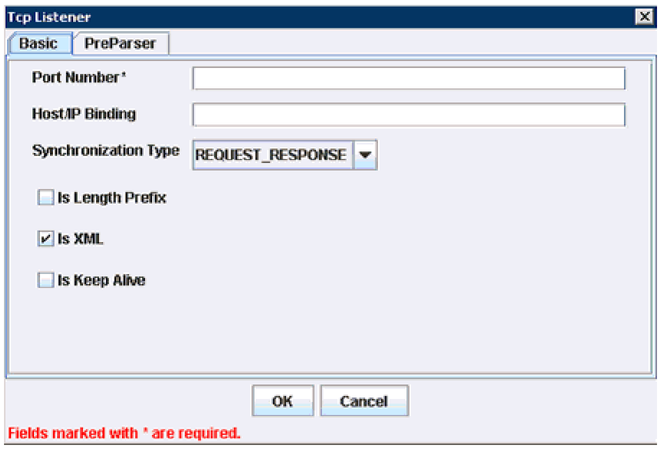
Figure 4–48 Add Channel Dialog



3. In the **Name** field, enter a descriptive name for the channel.
4. In the **Description** field, enter a description (optional).
5. From the **Protocol** list, choose a protocol for your channel.
6. Click **Next**.

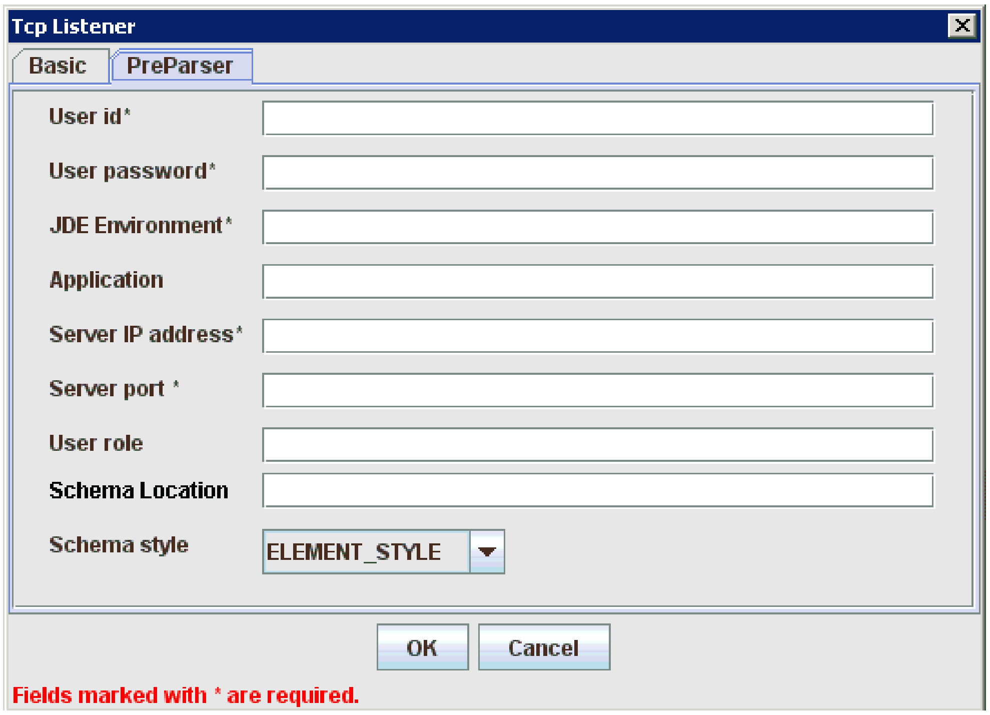
   The dialog is displayed for the selected listener, as shown in Figure 4–49.

Figure 4–49 TCP Listener Dialog



7. Enter the port number of the channel in the **Port Number** field.
8. Enter the location of the server in the **Host/IP Binding** field.
9. Select the Synchronization type from the **Synchronization Type** list.
10. Select **Is Length Prefix** for events that send data which is not in XML format. The TCP/IP event application must prefix the data with a 4-byte binary length field when writing the data to the TCP/IP port.
11. Select **Is XML** for events that send data back in XML format. No preparser is required.
12. Select **Is Keep Alive** to maintain a continuous communication between the event transaction and the channel.
13. Click the **PreParser** tab, as shown in Figure 4–50.

Figure 4–50 PreParser Tab



ELEMENT_STYLE

Fields marked with * are required.

Enter values based on the table.

Parameter	Description
User id*	A valid user ID for J.D. Edwards OneWorld.
User password*	The password associated with the user ID.
JDE environment*	Your J.D. Edwards OneWorld environment. For more information about this parameter, see your J.D. Edwards OneWorld documentation or ask your OneWorld system administrator.
Application	XMLInterop or the application name in J.D. Edwards OneWorld. Optional.
Server IP address*	The name of the server on which J.D. Edwards OneWorld is running. This can be the name of the server, for example, JDEOW, or its IP address, for example, 123.45.67.89.

Parameter	Description
Server Port*	The port number on which the server is listening, for example, 6009.
User Role	Define a user role according to your requirements.
Schema Location	The location of the XML schema (.xsd file) that was generated from the event output. For example: `<ADAPTER_HOME>\config\configuration_name\schemas\JDEdwards\target_name\jde-schema.xsd` For more information, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.
Schema style	Choose from one of the following options: ■ ELEMENT_STYLE (default) ■ ATTRIBUTE_STYLE

Click **OK**.

The channel is created and displayed under the Channels node. An X over the icon indicates that the channel is currently disconnected.

Note: The channel you created in Application Explorer is managed by BPEL PM Server. If you start the channel for testing and debugging purposes, then stop it before run-time.

4.5.1.2 Generating WSDL for Event Notification (Command Prompt Only)

You cannot generate WSDL for J.D. Edwards OneWorld event notification using Application Explorer. To generate WSDL from the command prompt, you must perform the following steps.

You can create inbound J2CA service only if the node you have selected supports events.

Note: The schema validation options (Root, Namespace, Schema) are not applicable for the Oracle Application Adapter for J.D. Edwards OneWorld.

To generate a WSDL file for J.D. Edwards OneWorld event notification:

1. Create a channel using Application Explorer under the J.D. Edwards Events node.
2. Start the channel.

   Do not restart Oracle WebLogic Server after the channel is started.
3. Send an inbound message from J.D. Edwards OneWorld.
4. Capture the inbound message payload in the log file, which is located in the following directory based on your adapter installation:

   ```
   <ADAPTER_HOME>\config\configuration_name\log\iwaf_jca1500.log
   ```

   Alternatively, you can create a port using the File protocol under the Events node in Application Explorer, which disposes the event message to the file system.

5. Use a third party tool (for example, XMLSpy) to create the XML schema (.xsd file) using the XML payload that was captured in the previous step.
6. In the generated XML schema (.xsd file) perform the following modifications:
   a. Search for `Schemas-jdedwards-com` and replace it with `iwaysoftware`.

```
<xs:schema
targetNamespace="urn:Schemas-jdedwards-com:trans.response.JDESOOUT"
      xmlns="urn:Schemas-jdedwards-com:trans.response.JDESOOUT"
 xmlns:xs=http://www.w3.org/2001/XMLSchema elementFormDefault="qualified">
```

   to:

```
<xs:schema
      targetNamespace="urn:iwaysoftware:trans.response.JDESOOUT"
      xmlns="urn:iwaysoftware:trans.response.JDESOOUT"
xmlns:xs=http://www.w3.org/2001/XMLSchema elementFormDefault="qualified">
```

   b. Cut the following syntax:

```
<xs:element name="jdeResponse">
<xs:complexType>
</xs:complexType>
</xs:element>
```

   c. Paste it before the following line:

```
<xs:element name="transaction">
```

7. Copy the XML schema (.xsd file) to the following directory based on your adapter installation:

```
<ADAPTER_HOME>\config\configuration_name\schemas\JDEdwards\target_name\
```

Note: Edit the created channel by providing the location of the schema (.xsd) file (as mentioned in step 7) in the PreParser tab of Application Explorer. For example:

```
<ADAPTER_HOME>\config\configuration_name\schemas\JDEdwards\target_
name\jde-schema.xsd
```

8. Open a command prompt and navigate to the following base domain directory:

```
<ADAPTER_HOME>\user_projects\domains\base_domain\bin
```

9. Execute **setDomainEnv.cmd** (Windows) or **../setDomainEnv.sh** (UNIX/Linux).
10. In the same command prompt, navigate to the following directory:

```
<ADAPTER_HOME>\tools\iwae\bin
```

11. Execute the *obadapter.bat* file to set the environment.
12. Based on your adapter installation, navigate to the following directory where the XML schema (.xsd file) is copied:

```
<ADAPTER_HOME>\config\configuration_name\schemas\JDEdwards\target_name
```

13. Enter the following command to generate a WSDL:

```
java -Diway.oem=oracle12c
com.iwaysoftware.af.container.tools.wsdl.IWayWSILBrowser adapterhome adapter
target channel schemaPrefix wsdlFileName
```

where:

adapterhome is the path to your ApplicationAdapters home. For example:

For SOA:

```
<ORACLE_HOME>\soa\soa\thirdparty\ApplicationAdapters
```

For OSB:

```
<ORACLE_HOME>\osb\3rdparty\ApplicationAdapters
```

adapter is the name of the adapter. For example, JDEdwards.

target is the name of the adapter target you created in Application Explorer.

channel is the name of the channel you created in Application Explorer.

schemaPrefix is the prefix for the XSD schema. The schema file must be in the same directory where the Java command is executed.

Execute the following command to generate the inbound WSDL.

```
java -Diway.oem=oracle12c
com.iwaysoftware.af.container.tools.wsdl.IWayWSILBrowser
C:\12c_SOA\soa\soa\thirdparty\ApplicationAdapters\
JDEdwards JDEConnection JDEchannel jde-schema J2CA_Inbound_receive.wsdl
```

Once the command is executed, the following is displayed in the command window:

```
Running Inbound WSDL generation tool...
  -> Generating WSDL...
  -> Generating files for OEM oracle12c
  -> Done.
```

Note: It is good practice to append **_receive** to the names of WSDL files that are generated for event notification services. This allows you to easily distinguish between them and those generated for request-response services.

14. Stop the channel in Application Explorer.

You can now create a new SOA application, which is the first step that is required to define a BPEL inbound process in Oracle JDeveloper.

4.5.2 Creating an Empty Composite for SOA

Perform the following steps to create an empty composite for SOA:

1. Create a new SOA application.
2. Enter a name for the new SOA Application and click **Next**.

   The Name your project page is displayed.
3. Enter a project name and click **Next**.

The Configure SOA settings page is displayed.

4. From the Composite Template list, select **Empty Composite** and click **Finish**.

   For more information, see Section 4.4.2, "Creating an Empty Composite for SOA" on page 4-9.

4.5.3 Defining a BPEL Inbound Process

This section describes how to define a BPEL inbound process, which consists of the following topics:

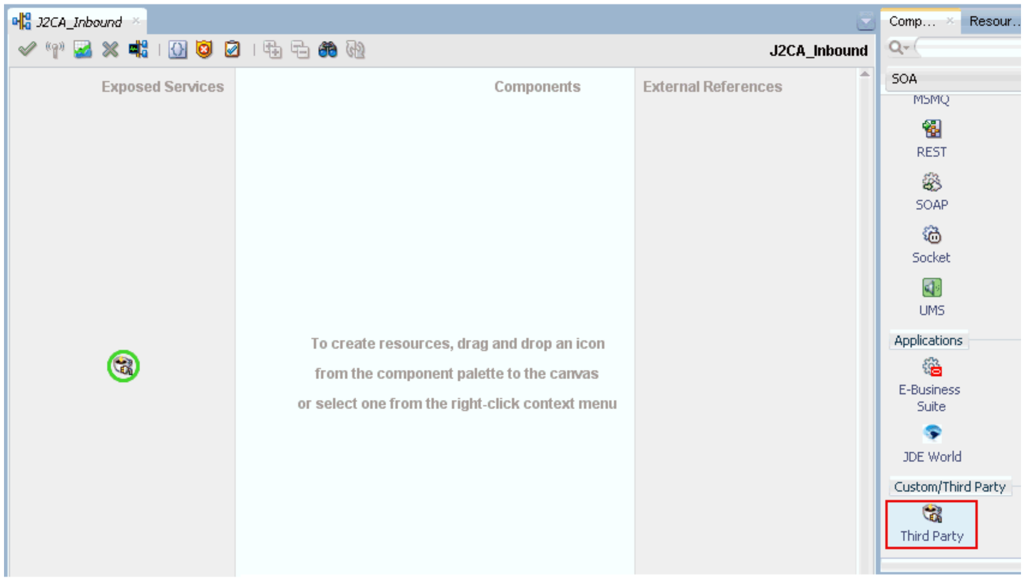
- Section 4.5.3.1, "Creating a Third Party Adapter Service Component"
- Section 4.5.3.2, "Creating an Inbound BPEL Process Component"
- Section 4.5.3.3, "Adjusting for Known Deployment Issues With 12c"

4.5.3.1 Creating a Third Party Adapter Service Component

Perform the following steps to create a third party adapter service component:

1. Drag and drop the **Third Party Adapter** component from the Service Adapters pane to the Exposed Services pane, as shown in Figure 4–51.

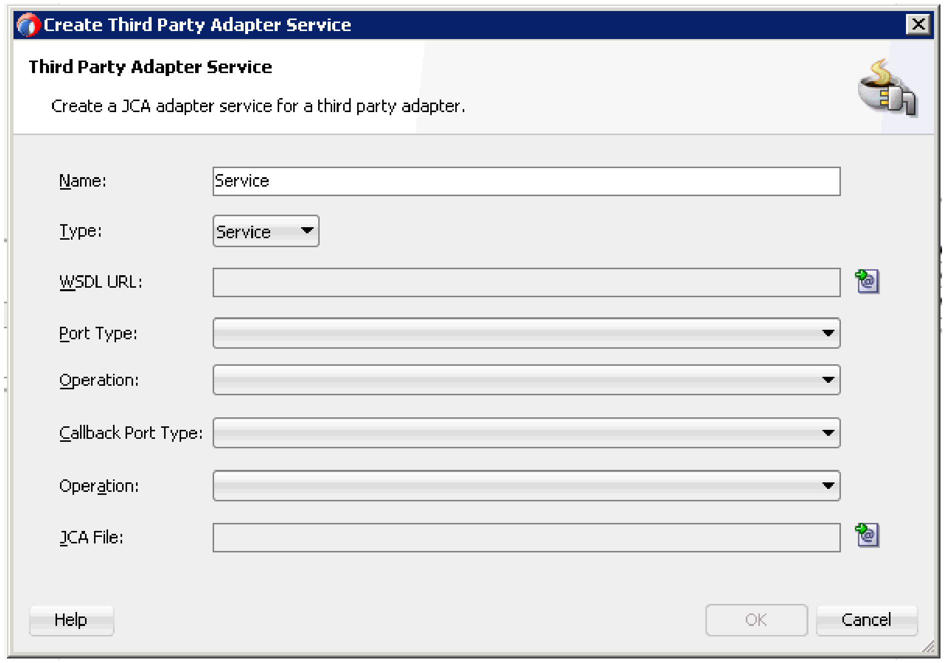
Figure 4–51 Third Party Adapter Component



To create resources, drag and drop an icon from the component palette to the canvas or select one from the right-click context menu

The Create Third Party Adapter Service dialog is displayed, as shown in Figure 4–52.

Figure 4–52 Create Third Party Adapter Service Dialog



Third Party Adapter Service

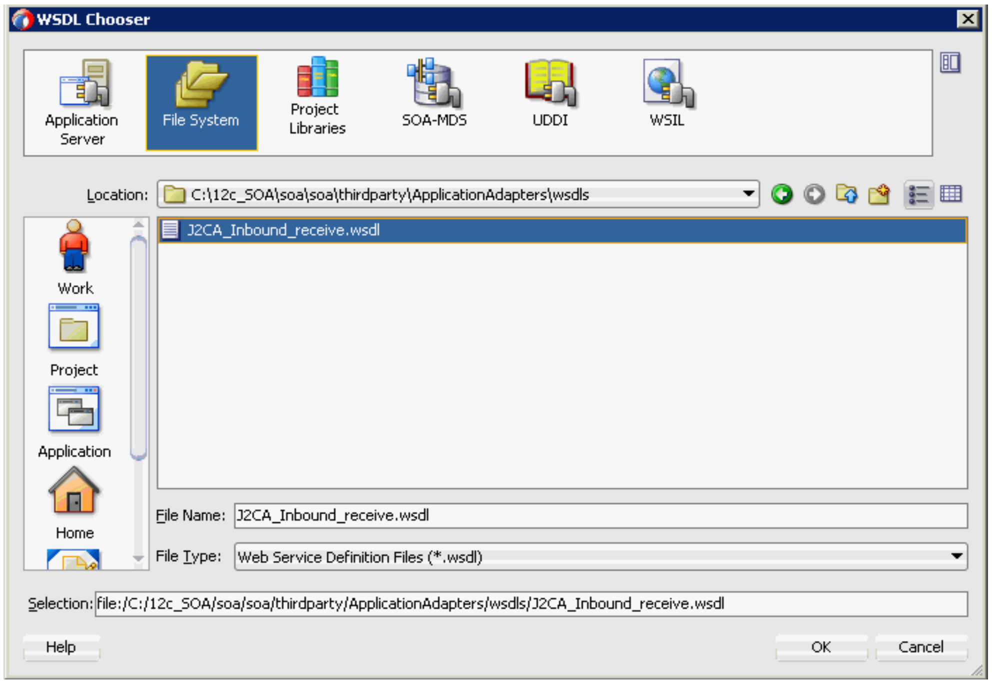
Create a JCA adapter service for a third party adapter.

Name:	Service
Type:	Service
WSDL URL:	
Port Type:	
Operation:	
Callback Port Type:	
Operation:	
JCA File:	

2. Ensure that **Service** is selected from the Type list (default).
3. Click the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.

   The WSDL Chooser dialog is displayed, as shown in Figure 4–53.

Figure 4–53 WSDL Chooser Dialog



Location: C:\12c_SOA\soa\soa\thirdparty\ApplicationAdapters\wsdls

J2CA_Inbound_receive.wsdl

File Name: J2CA_Inbound_receive.wsdl

File Type: Web Service Definition Files (*.wsdl)

Selection: file:/C:/12c_SOA/soa/soa/thirdparty/ApplicationAdapters/wsdls/J2CA_Inbound_receive.wsdl

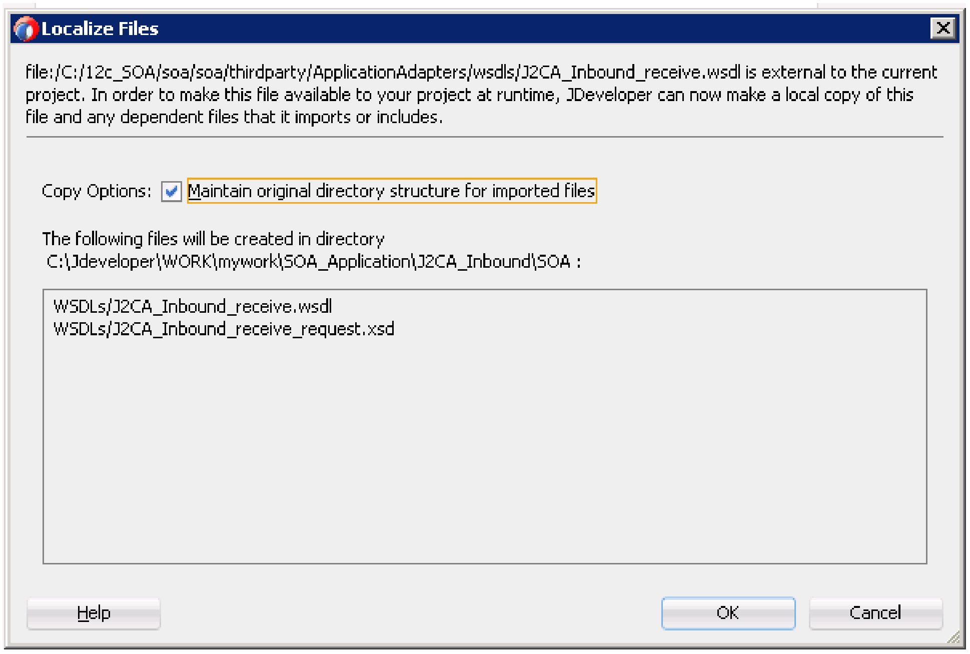
4. Browse and select an inbound WSDL file from the following directory:

```
<ADAPTER_HOME>\wsdls
```

5. Click **OK**.

   The Localize Files dialog is displayed, as shown in Figure 4–54.

Figure 4–54 Localize Files Dialog



Localize Files

file:/C:/12c_SOA/soa/soa/thirdparty/ApplicationAdapters/wsdls/J2CA_Inbound_receive.wsdl is external to the current project. In order to make this file available to your project at runtime, JDeveloper can now make a local copy of this file and any dependent files that it imports or includes.

Copy Options: [✓] Maintain original directory structure for imported files

The following files will be created in directory
C:\Jdeveloper\WORK\mywork\SOA_Application\J2CA_Inbound\SOA :

```
WSDLs/J2CA_Inbound_receive.wsdl
WSDLs/J2CA_Inbound_receive_request.xsd
```

6. Click **OK**.

   The inbound WSDL file and associated receive/request XML schema file (.xsd) are imported to the project folder that has been created.

   You are returned to the Create Third Party Adapter Service dialog.

7. Click the **Find JCA file** icon, which is located to the right of the JCA File field.

   The Transformation Chooser dialog is displayed.

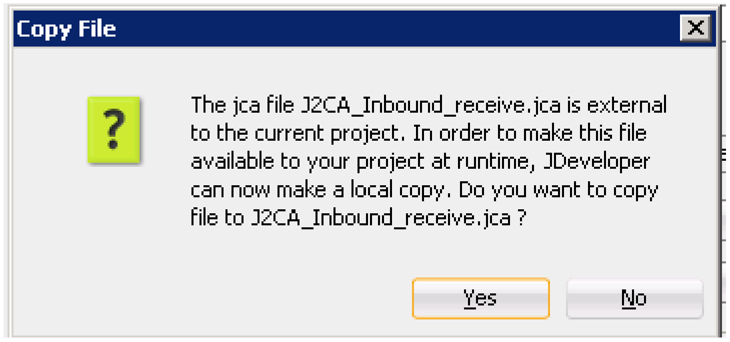
8. Browse and select the JCA properties file from the following directory:

   ```
   <ADAPTER_HOME>\wsdls
   ```

9. Click **OK**.

   A Copy File message is displayed, as shown in Figure 4–55.

Figure 4–55 Copy File Confirmation Message



Copy File

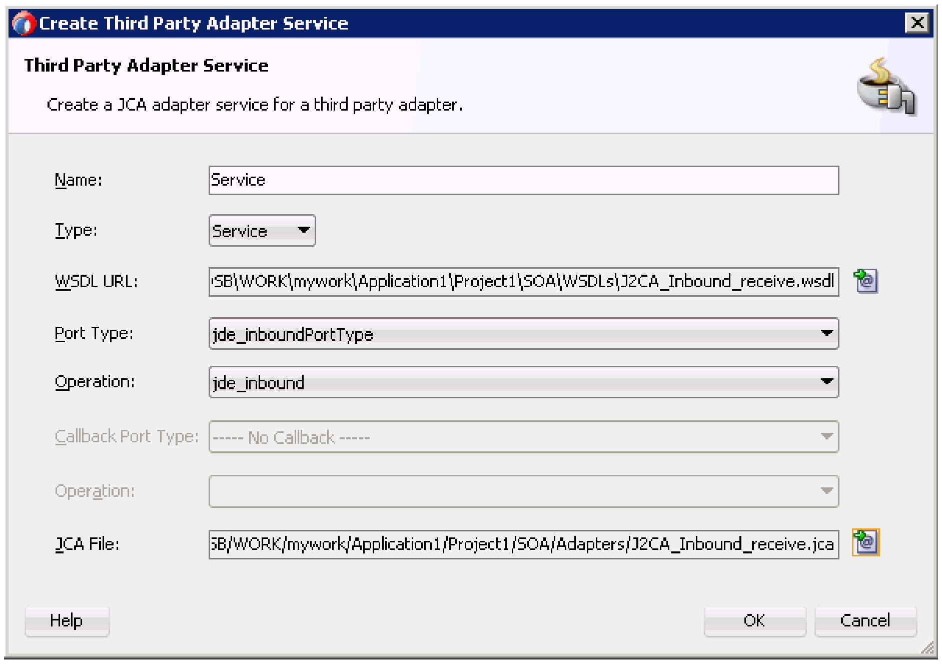
The jca file J2CA_Inbound_receive.jca is external to the current project. In order to make this file available to your project at runtime, JDeveloper can now make a local copy. Do you want to copy file to J2CA_Inbound_receive.jca ?

10. Click **Yes**.

A copy of the JCA properties file is made in the project folder.

You are returned to the Create Third Party Adapter Service dialog, as shown in Figure 4–56.

Figure 4–56 Create Third Party Adapter Service Dialog



Create Third Party Adapter Service

Third Party Adapter Service

Create a JCA adapter service for a third party adapter.

Name: Service

Type: Service

WSDL URL: SB\WORK\mywork\Application1\Project1\SOA\WSDLs\J2CA_Inbound_receive.wsdl

Port Type: jde_inboundPortType

Operation: jde_inbound

Callback Port Type: ----- No Callback -----

Operation:

JCA File: SB/WORK/mywork/Application1/Project1/SOA/Adapters/J2CA_Inbound_receive.jca

11. Click **OK**.

The third party adapter service component is created and displayed in the Exposed Services pane.

You are now ready to configure an inbound BPEL process component.

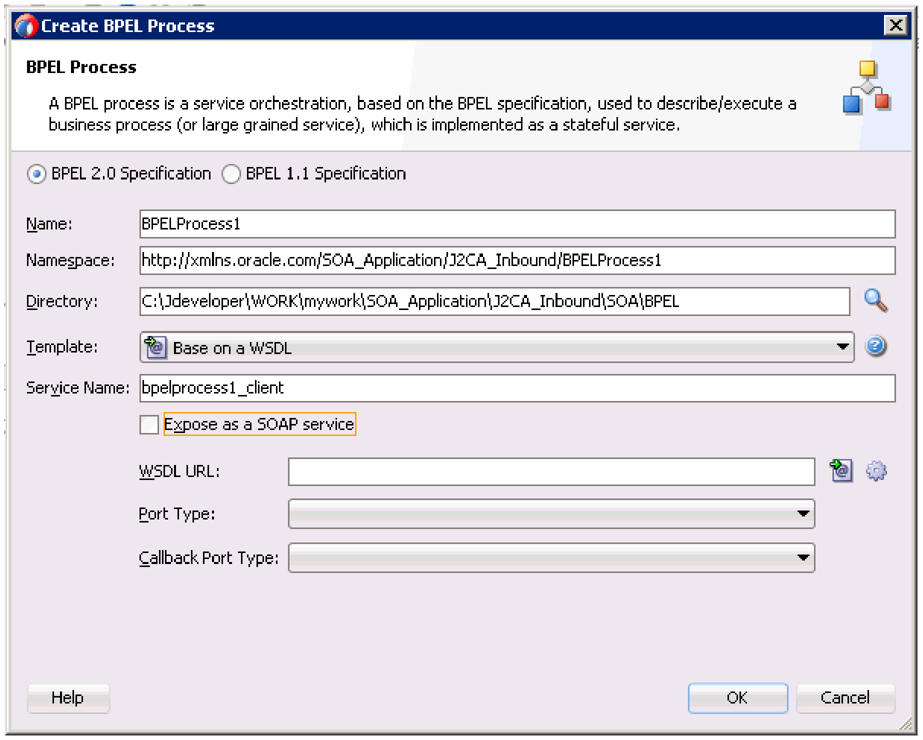
4.5.3.2 Creating an Inbound BPEL Process Component

Perform the following steps to create an inbound BPEL process component:

1. Drag and drop the **BPEL Process** component from the Service Components pane to the Components pane.

The Create BPEL Process dialog is displayed, as shown in Figure 4–57.

Figure 4–57 Create BPEL Process Dialog


Create BPEL Process

BPEL Process

A BPEL process is a service orchestration, based on the BPEL specification, used to describe/execute a business process (or large grained service), which is implemented as a stateful service.

[✓ BPEL 2.0 Specification] BPEL 1.1 Specification

Name:	BPELProcess1
Namespace:	http://xmlns.oracle.com/SOA_Application/J2CA_Inbound/BPELProcess1
Directory:	C:\Jdeveloper\WORK\mywork\SOA_Application\J2CA_Inbound\SOA\BPEL
Template:	Base on a WSDL
Service Name:	bpelprocess1_client

☐ Expose as a SOAP service

WSDL URL:	
Port Type:	
Callback Port Type:	

2. In the Name field, enter a name to identify the new inbound BPEL process component or leave to default.

   By default, the BPEL 2.0 Specification option is selected.

3. From the Template list, select **Base on a WSDL**.

4. Uncheck the **Expose as SOAP service** check box.

5. Click the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.

   The WSDL Chooser dialog is displayed.

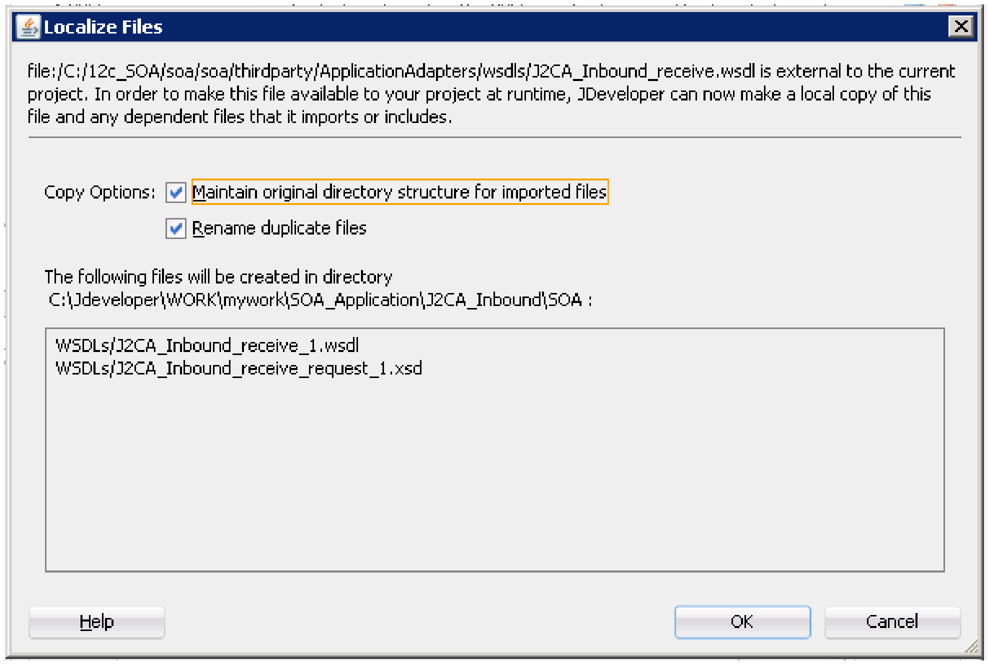
6. Select an inbound WSDL file from the following directory:

   ```
   <ADAPTER_HOME>\wsdls
   ```

7. Click **OK**.

   The Localize Files dialog is displayed, as shown in Figure 4–58.

Figure 4–58 Localize Files Dialog



file:/C:/12c_SOA/soa/soa/thirdparty/ApplicationAdapters/wsdls/J2CA_Inbound_receive.wsdl is external to the current project. In order to make this file available to your project at runtime, JDeveloper can now make a local copy of this file and any dependent files that it imports or includes.

Copy Options: ✓ Maintain original directory structure for imported files

✓ Rename duplicate files

The following files will be created in directory C:\Jdeveloper\WORK\mywork\SOA_Application\J2CA_Inbound\SOA :

WSDLs/J2CA_Inbound_receive_1.wsdl
WSDLs/J2CA_Inbound_receive_request_1.xsd

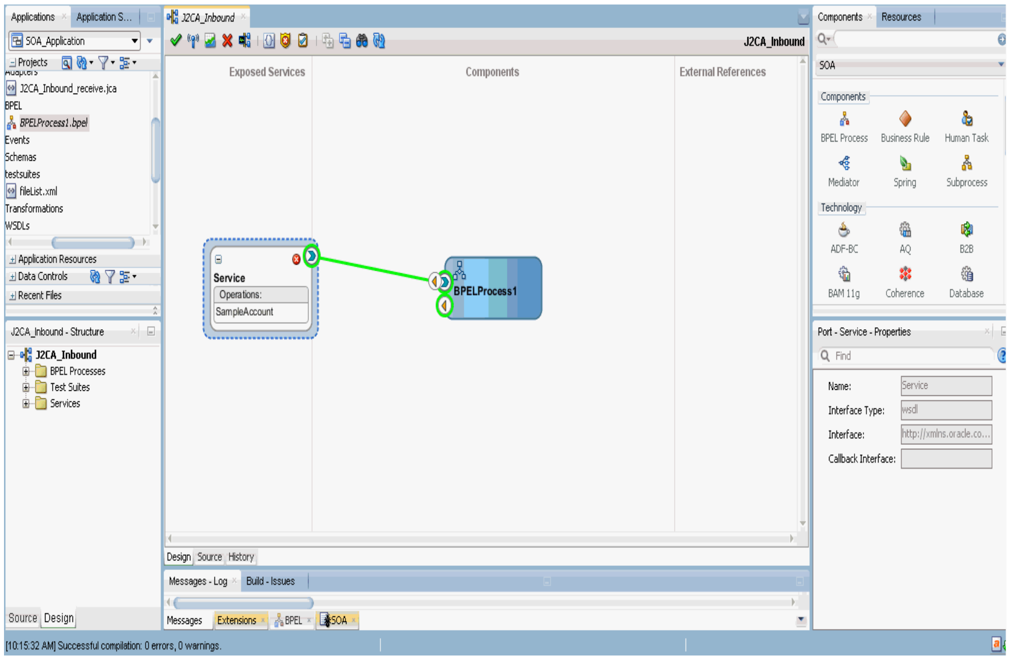
8. Uncheck the **Rename duplicate files** option.

9. Click **OK**.

   You are returned to the Create BPEL Process dialog.

10. Click **OK**.

Figure 4–59 Created Connection



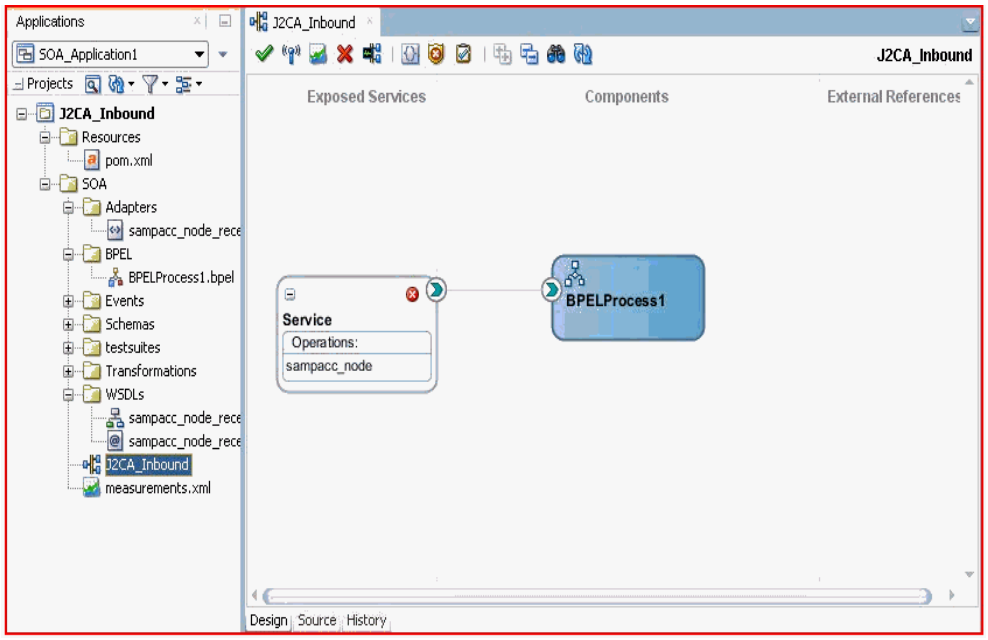
Name: Service
Interface Type: wsdl
Interface: http://xmlns.oracle.co...
Callback Interface:

[10:15:32 AM] Successful compilation: 0 errors, 0 warnings.

11. Create a connection between the third party adapter service component and the inbound BPEL process component, as shown in Figure 4–59.

12. Double-click **J2CA_Outbound** in the left pane.

Figure 4–60 Save All Icon



13. Click the **Save All** icon in the menu bar to save the new inbound BPEL process component that was configured, as shown in Figure 4–60.

    You are now ready to deploy the BPEL inbound process.

4.5.3.3 Adjusting for Known Deployment Issues With 12c

For more information on how to adjust for known deployment issues with 12c, see Section 4.4.3.3, "Adjusting for Known Deployment Issues With 12c" on page 4-26.

4.5.4 Deploying the BPEL Inbound Process

Perform the following steps to deploy the BPEL inbound process.

1. Right-click the project name in the left pane, select **Deploy**, and click **J2CA_ Inbound**.

   The Deployment Action page is displayed.

2. Ensure that **Deploy to Application Server** is selected.
3. Click **Next**.

   The Deploy Configuration page is displayed.

4. Leave the default values selected and click **Next**.

   The Select Server page is displayed.

5. Select an available application server that was configured and click **Next**.

   The SOA Servers page is displayed.

6. Select a target SOA server and click **Next**.

   The Summary page is displayed.

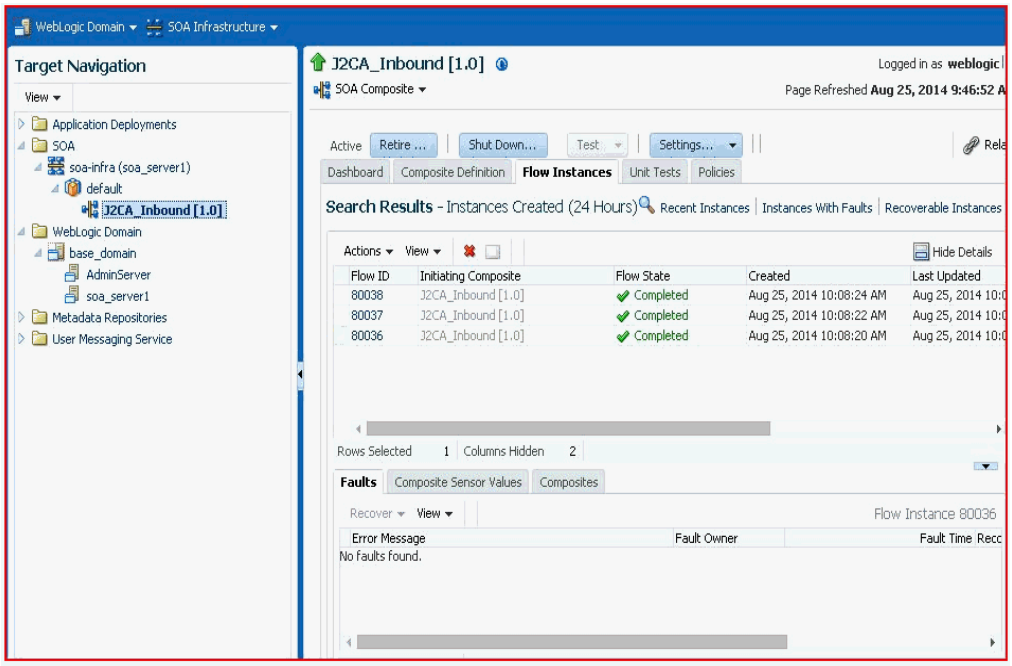
7. Review and verify all the available deployment information for your project and click **Finish**.

   The process is deployed successfully.

   For more information, see Section 4.4.4, "Deploying the BPEL Outbound Process" on page 4-28.

   Once event messages are triggered through J.D. Edwards OneWorld, successful instances are received in the Oracle Enterprise Manager console, as shown in Figure 4–61.

Figure 4–61 Received Instances



Flow ID	Initiating Composite	Flow State	Created	Last Updated
80038	J2CA_Inbound [1.0]	✓ Completed	Aug 25, 2014 10:08:24 AM	Aug 25, 2014 10:0
80037	J2CA_Inbound [1.0]	✓ Completed	Aug 25, 2014 10:08:22 AM	Aug 25, 2014 10:0
80036	J2CA_Inbound [1.0]	✓ Completed	Aug 25, 2014 10:08:20 AM	Aug 25, 2014 10:0

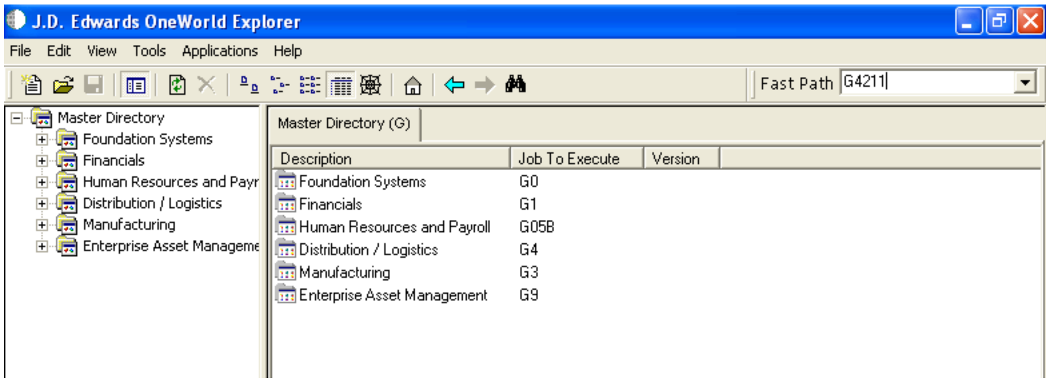
Error Message	Fault Owner	Fault Time	Recc
No faults found.			

4.5.5 Triggering an Event in J.D. Edwards OneWorld

Events are generated by activity in a database or in an application system. You can use events to trigger an action in your application. To trigger an event in J.D. Edwards OneWorld:

1. Log in to your J.D. Edwards OneWorld system.
2. In the **Fast Path** field of the J.D. Edwards OneWorld Explorer window, type **G4211** and press **Enter**, as shown in Figure 4–62.

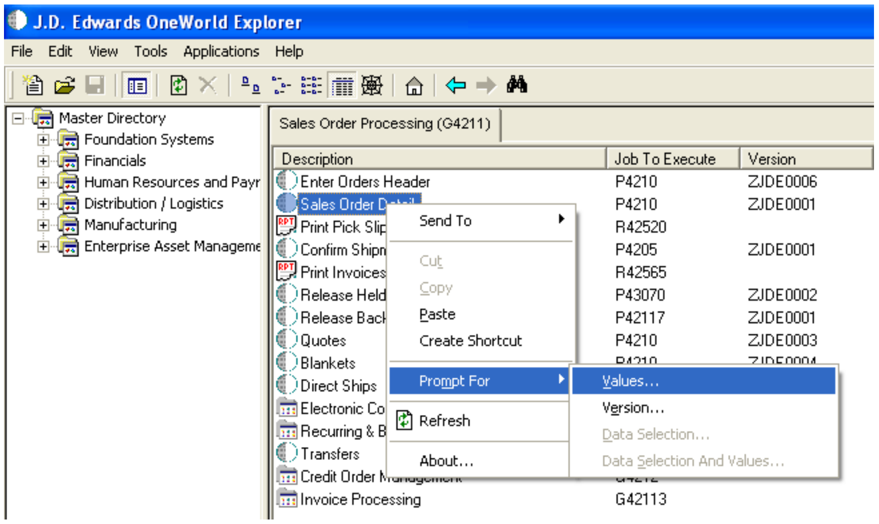
Figure 4–62 J.D. Edwards OneWorld Explorer Window



Description	Job To Execute	Version
Foundation Systems	G0	
Financials	G1	
Human Resources and Payroll	G05B	
Distribution / Logistics	G4	
Manufacturing	G3	
Enterprise Asset Management	G9	

3. Right-click **Sales Order Detail** (P4210).

Figure 4–63 Sales Order Detail Menu

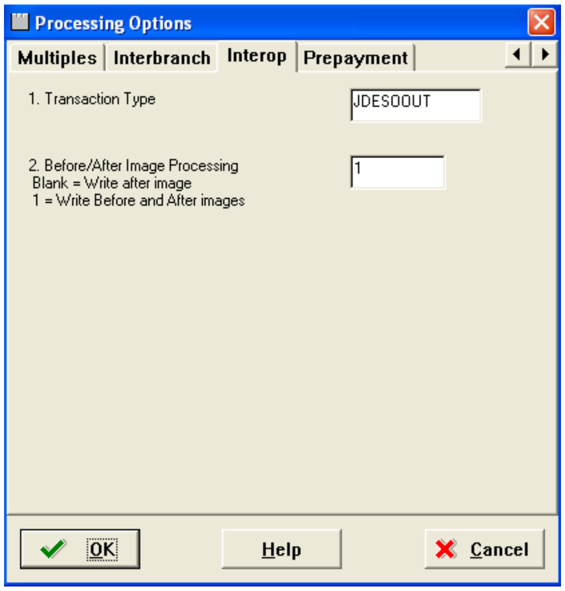


Description	Job To Execute	Version
Enter Orders Header	P4210	ZJDE0006
Sales Order Detail	P4210	ZJDE0001
Print Pick Slip	R42520	
Confirm Shipn	P4205	ZJDE0001
Print Invoices	R42565	
Release Held	P43070	ZJDE0002
Release Back	P42117	ZJDE0001
Quotes	P4210	ZJDE0003
Blankets	P4210	ZJDE0004
Direct Ships		
Electronic Co		
Recurring & B		
Transfers		
Credit Order Management	G4212	
Invoice Processing	G42113	

4. Select **Prompt for**, and then **Values**, as shown in Figure 4–63.

   The Processing Options dialog is displayed, as shown in Figure 4–64.

Figure 4–64 Processing Options Dialog



1. Transaction Type: JDESOOUT

2. Before/After Image Processing
Blank = Write after image
1 = Write Before and After images: 1

Perform the following steps:

a. Click the **Interop** tab.

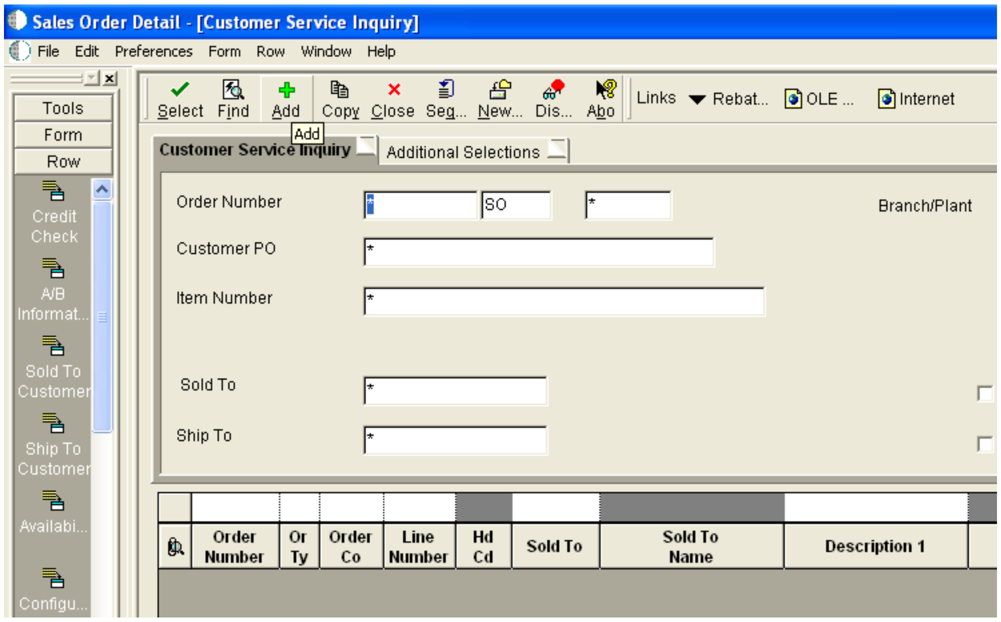
b. In the **Transaction Type** field, type **JDESOOUT**.

c. Verify that the value in the **Before/After Image Processing Blank** field is 1.

5. Click **OK**.

The **Sales Order Detail - (Customer Service Inquiry)** window is displayed, as shown in Figure 4–65.

Figure 4–65 Sales Order Detail Window



Order Number: * SO *

Customer PO: *

Item Number: *

Sold To: *

Ship To: *

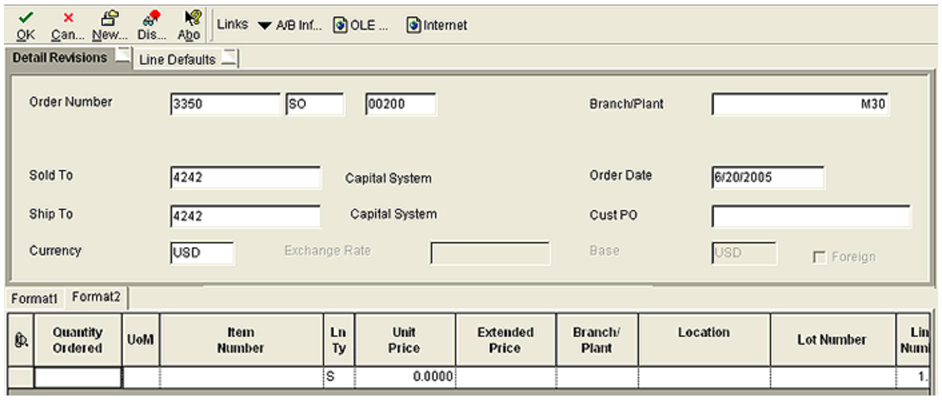
Order Number	Or Ty	Order Co	Line Number	Hd Cd	Sold To	Sold To Name	Description 1

6. Click the **Add** icon (third icon from left).

7. Enter the values as shown in Figure 4–66.

   To move to a different field, use the **Tab** key on your keyboard.

Figure 4–66 Values


OK Can... New... Dis... Abo Links A/B Inf... OLE ... Internet

Detail Revisions | Line Defaults

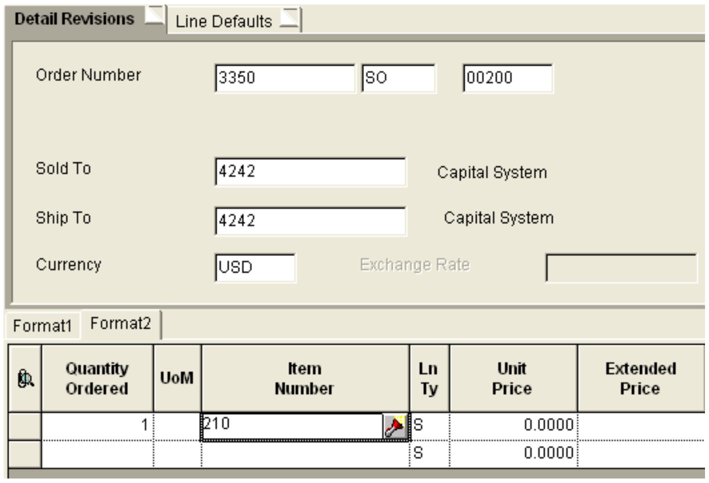
Order Number	3350	SO	00200	Branch/Plant	M30
Sold To	4242	Capital System		Order Date	6/20/2005
Ship To	4242	Capital System		Cust PO	
Currency	USD	Exchange Rate		Base	USD Foreign

Format1 | Format2

	Quantity Ordered	UoM	Item Number	Ln Ty	Unit Price	Extended Price	Branch/ Plant	Location	Lot Number	Lin Num
				S	0.0000					1.

8. Enter a value for **Quantity Ordered** and **Item Number**, as shown in Figure 4–67.

Figure 4–67 Sample Values


Detail Revisions | Line Defaults

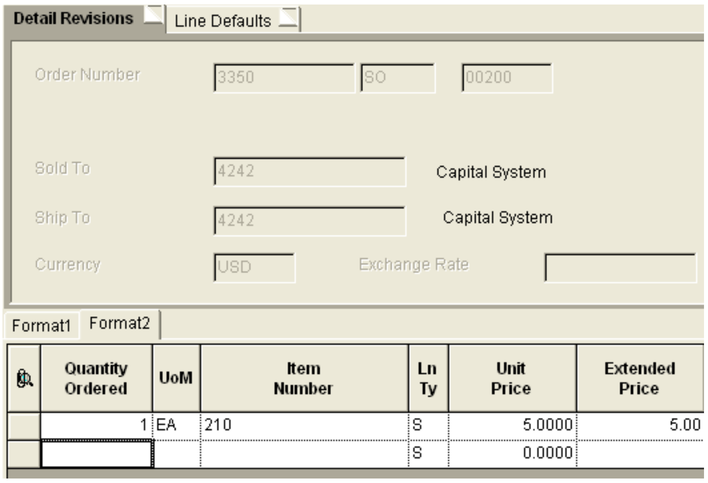
Order Number	3350	SO	00200
Sold To	4242	Capital System	
Ship To	4242	Capital System	
Currency	USD	Exchange Rate	

Format1 | Format2

	Quantity Ordered	UoM	Item Number	Ln Ty	Unit Price	Extended Price
	1		210	S	0.0000	
				S	0.0000	

9. Click the first field in the second row and allow a few seconds for processing, as shown in Figure 4–68.

Figure 4–68 Sample Values



Detail Revisions | Line Defaults

Order Number	3350	SO	00200	
Sold To	4242	Capital System		
Ship To	4242	Capital System		
Currency	USD	Exchange Rate		

Format1 | Format2

	Quantity Ordered	UoM	Item Number	Ln Ty	Unit Price	Extended Price
	1	EA	210	S	5.0000	5.00
				S	0.0000	

10. Click **OK**.

An event is triggered in the J.D. Edwards OneWorld system.

Verifying the Results

To verify your results:

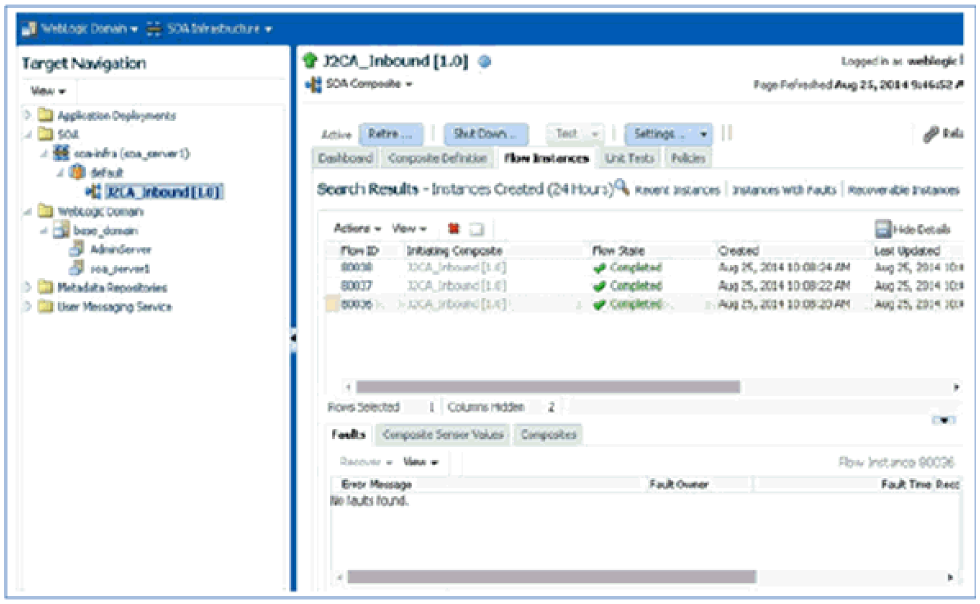
1. Log in to the Oracle Enterprise Manager console by using the following URL:

   ```
   http://localhost:7001/em
   ```

2. Click **SOA**, select **soa-infra (soa_server1)**, **default**, and then click **J2CA_Inbound**.
3. Click **Flow Instances**.

   Instances will be received, as shown in Figure 4–69.

Figure 4–69 Flow Instances



Flow ID	Initiating Composite	Flow State	Created	Last Updated
80038	J2CA_Inbound [1.0]	✓ Completed	Aug 25, 2014 10:08:24 AM	Aug 25, 2014 10:
80037	J2CA_Inbound [1.0]	✓ Completed	Aug 25, 2014 10:08:22 AM	Aug 25, 2014 10:
80036	J2CA_Inbound [1.0]	✓ Completed	Aug 25, 2014 10:08:20 AM	Aug 25, 2014 10:

Rows Selected 1 | Columns Hidden 2

Flow Instance 80036

Error Message	Fault Owner	Fault Time Rec
No faults found.		

4.6 Designing an Outbound BPEL Process for Service Integration (BSE Configuration)

This section describes how to design an outbound BPEL process for service integration.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\BPEL\BSE\Outbound_Project
```

The following tools are required to complete your adapter design-time configuration:

- Oracle Adapter Application Explorer (Application Explorer)
- Oracle JDeveloper BPEL Designer (JDeveloper)

This section includes the following topics:

- Section 4.6.1, "Generating a WSDL File for Request and Response Services Using a Web Service"
- Section 4.6.2, "Creating an Empty Composite for SOA"
- Section 4.6.3, "Defining a BPEL Outbound Process"

Before you design a BPEL process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.6.1, "Generating a WSDL File for Request and Response Services Using a Web Service".

4.6.1 Generating a WSDL File for Request and Response Services Using a Web Service

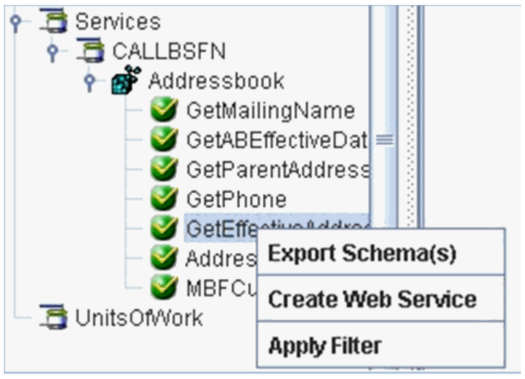
To generate a WSDL file for request and response services using a Web service:

1. Start Application Explorer and connect to a defined J.D. Edwards target (BSE configuration).

   For more information on defining a target and connecting to J.D. Edwards, see Section 2.4.1, "Defining a Target to J.D. Edwards OneWorld".

2. Expand the J.D. Edwards target to which you are connected.
3. Expand **Services**, **CALLBSFN**, and **Addressbook**.
4. Right-click **GetEffectiveAddress**, and then select **Create Web Service** from the menu, as shown in Figure 4–70.

Figure 4–70 Create Web Service Option



The Create Web Service dialog is displayed.

5. Enter a service name, and click **Next**.

6. Click **OK** on the next dialog that is displayed.

   Application Explorer switches the view to the Business Services node, and the new Web service is displayed in the left pane.

7. Right-click the new Web service and select **Save WSDL** from the menu.

8. Save the WSDL in the wsdls folder and click **Save**.

   You can now create an empty composite for SOA, which is the first step that is required to define a BPEL outbound process in JDeveloper.

4.6.2 Creating an Empty Composite for SOA

To create an empty composite for SOA:

1. Create a new SOA application.

2. Enter a name for the SOA Application and click **Next**.

   The Name your project page is displayed.

3. Enter a project name and click **Next**.

   The Configure SOA settings page is displayed.

4. From the Composite Template list, select **Empty Composite** and click **Finish**.

   For more information, see Section 4.4.2, "Creating an Empty Composite for SOA," on page 4-9.

4.6.3 Defining a BPEL Outbound Process

This section describes how to configure a BPEL outbound process component.

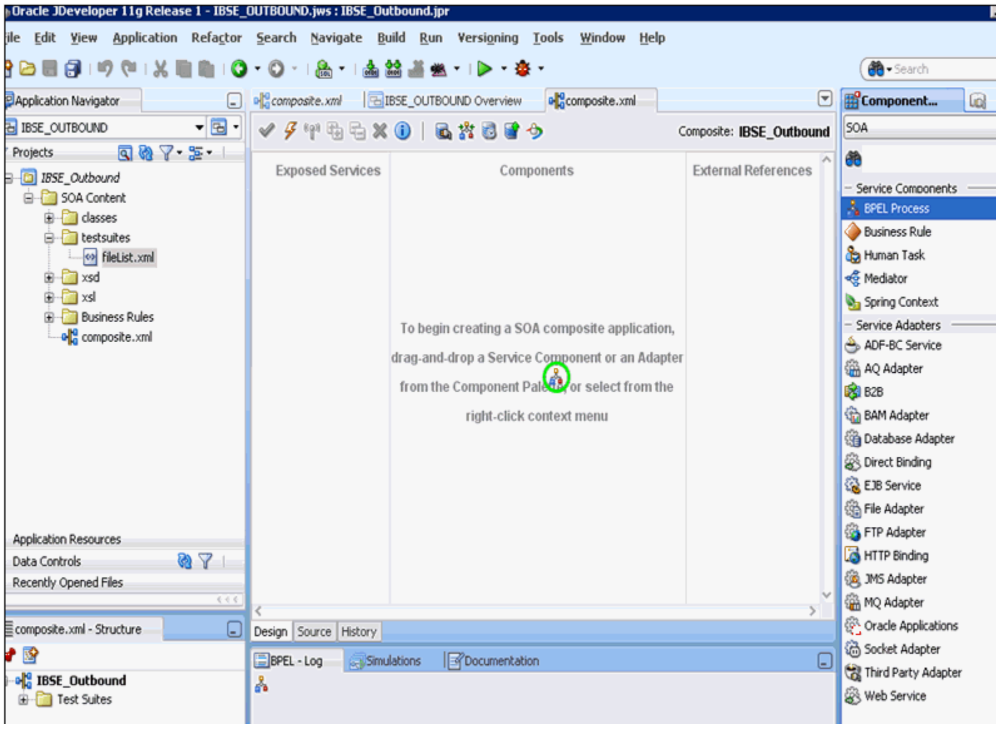
This section includes the following topics:

- Section 4.6.3.1, "Creating a Partner Link"
- Section 4.6.3.2, "Creating BPEL Activities and Mappings With the Created Partner Link"

To define a BPEL outbound process:

1. Drag and drop the **BPEL Process** component from the Service Components pane to the Components pane, as shown in Figure 4–71.

Figure 4–71 BPEL Process Component



To begin creating a SOA composite application, drag-and-drop a Service Component or an Adapter from the Component Palette or select from the right-click context menu

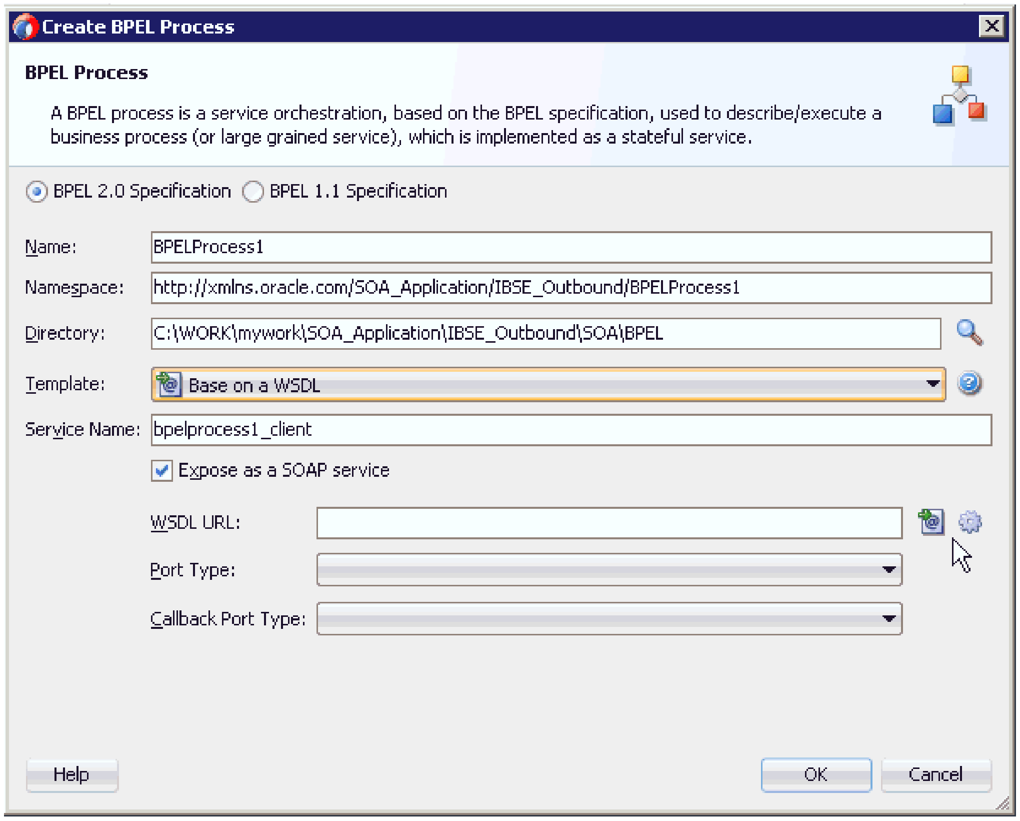
2. In the Name field, enter a name to identify the new outbound BPEL process component or leave it to the default value.

   By default, the BPEL 2.0 Specification option is selected.

3. From the Template drop-down list, select **Base on a WSDL**.

4. Click the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field, as shown in Figure 4–72.

Figure 4–72 Find Existing WSDLs Icon


Create BPEL Process

BPEL Process

A BPEL process is a service orchestration, based on the BPEL specification, used to describe/execute a business process (or large grained service), which is implemented as a stateful service.

[✓ BPEL 2.0 Specification] BPEL 1.1 Specification

Name:	BPELProcess1
Namespace:	http://xmlns.oracle.com/SOA_Application/IBSE_Outbound/BPELProcess1
Directory:	C:\WORK\mywork\SOA_Application\IBSE_Outbound\SOA\BPEL
Template:	Base on a WSDL
Service Name:	bpelprocess1_client

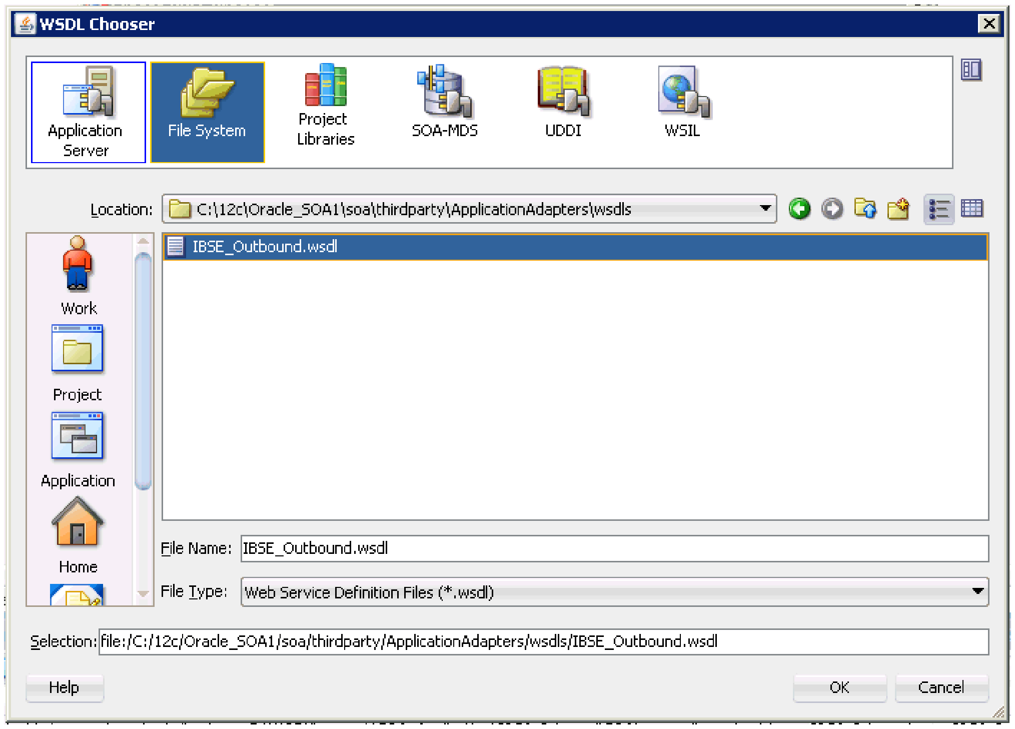
[✓] Expose as a SOAP service

WSDL URL:	
Port Type:	
Callback Port Type:	

The WSDL Chooser dialog is displayed.

5. Navigate to the location where the WSDL is exported from Application Explorer, select the WSDL, and click **OK**, as shown in Figure 4–73.

Figure 4–73 WSDL Chooser Dialog



Location: C:\12c\Oracle_SOA1\soa\thirdparty\ApplicationAdapters\wsdls

IBSE_Outbound.wsdl

File Name: IBSE_Outbound.wsdl

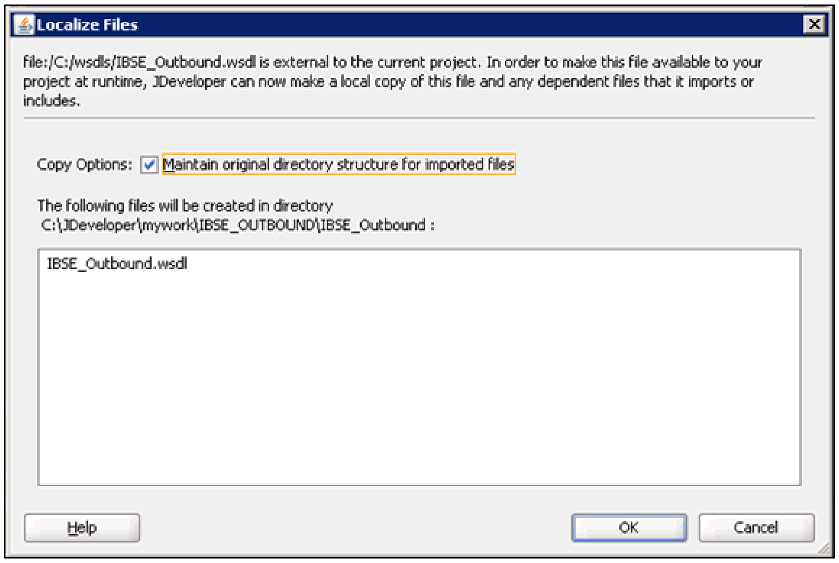
File Type: Web Service Definition Files (*.wsdl)

Selection: file:/C:/12c/Oracle_SOA1/soa/thirdparty/ApplicationAdapters/wsdls/IBSE_Outbound.wsdl

The Localize Files window is displayed.

6. In the displayed Localize Files window, click **OK**. This imports the WSDL file to the project folder, as shown in Figure 4–74.

Figure 4–74 Localize Files Window



file:/C:/wsdls/IBSE_Outbound.wsdl is external to the current project. In order to make this file available to your project at runtime, JDeveloper can now make a local copy of this file and any dependent files that it imports or includes.

Copy Options: ☑ Maintain original directory structure for imported files

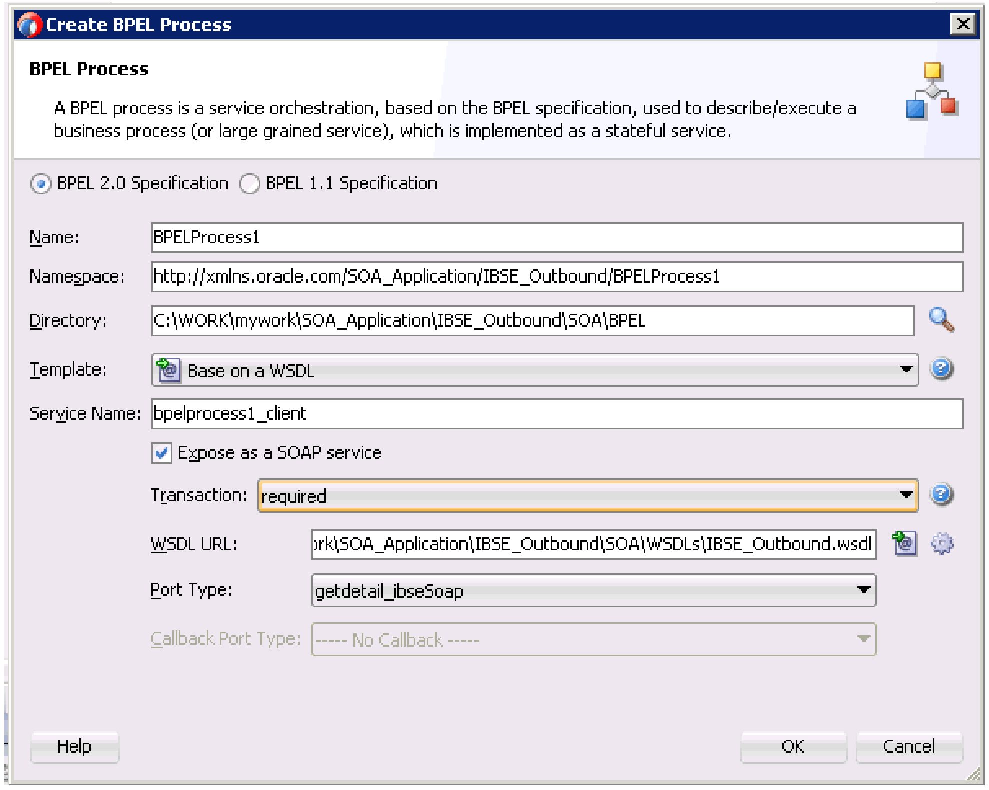
The following files will be created in directory C:\JDeveloper\mywork\IBSE_OUTBOUND\IBSE_Outbound :

IBSE_Outbound.wsdl

The Create BPEL Process window is displayed.

7. In the BPEL Process pane, click **OK**, as shown in Figure 4–75.

Figure 4–75 BPEL Process Pane



BPEL Process

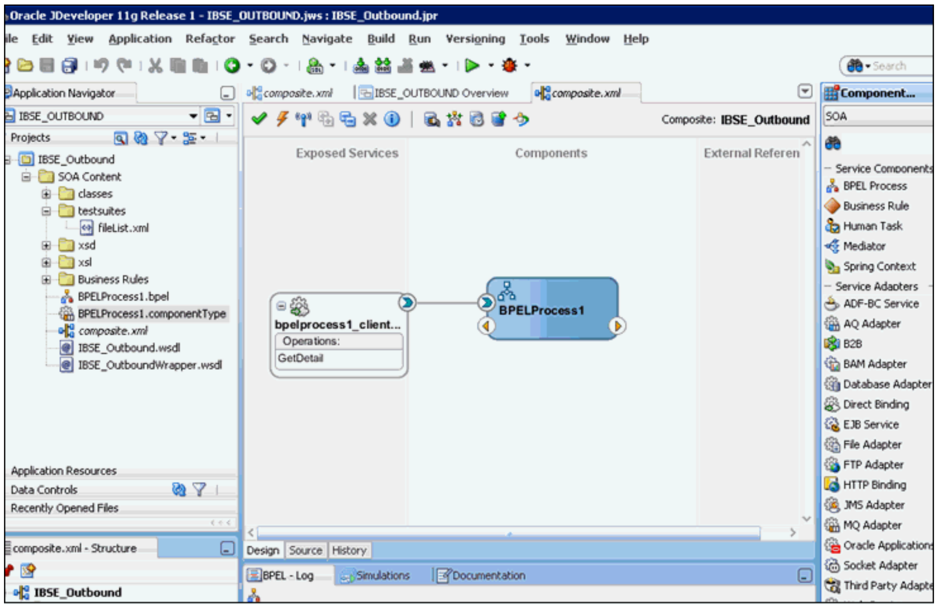
A BPEL process is a service orchestration, based on the BPEL specification, used to describe/execute a business process (or large grained service), which is implemented as a stateful service.

[✓ BPEL 2.0 Specification] BPEL 1.1 Specification

Name:	BPELProcess1
Namespace:	http://xmlns.oracle.com/SOA_Application/IBSE_Outbound/BPELProcess1
Directory:	C:\WORK\mywork\SOA_Application\IBSE_Outbound\SOA\BPEL
Template:	Base on a WSDL
Service Name:	bpelprocess1_client
✓ Expose as a SOAP service	
Transaction:	required
WSDL URL:	›rk\SOA_Application\IBSE_Outbound\SOA\WSDLs\IBSE_Outbound.wsdl
Port Type:	getdetail_ibseSoap
Callback Port Type:	----- No Callback -----

The BPEL Process component is created and displayed, as shown in Figure 4–76.

Figure 4–76 BPEL Process Component



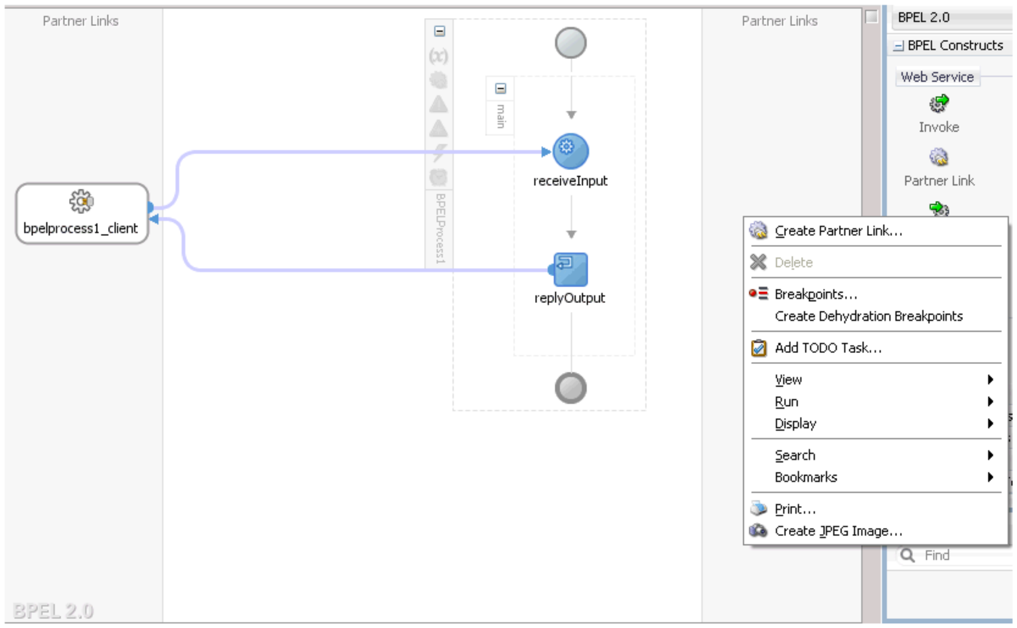
4.6.3.1 Creating a Partner Link

This section describes how to create a partner link.

To create a partner link:

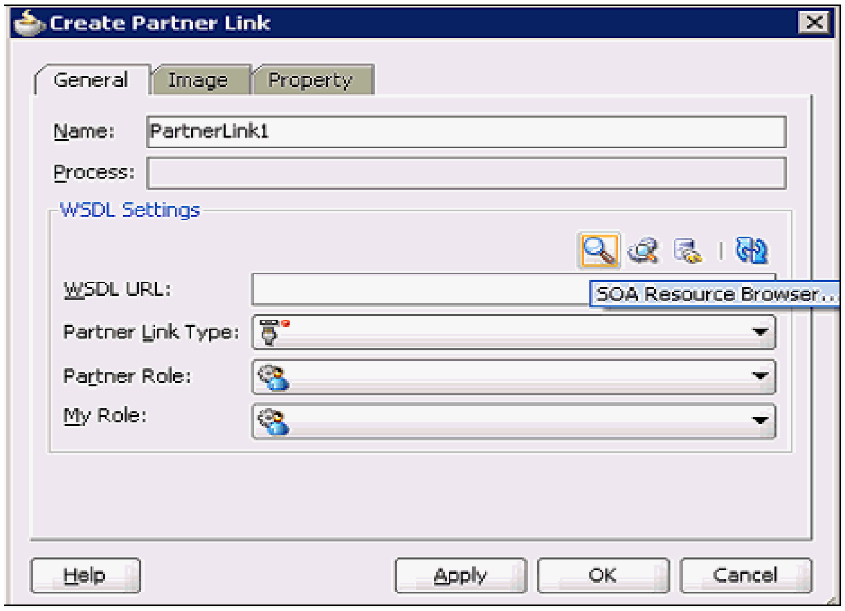
1. Double-click the outbound BPEL process component in the Components pane.
2. Right-click on the **Partner Links** pane and select **Create Partner Link**, as shown in Figure 4–77.

Figure 4–77 Create Partner Link



3. In the displayed Create Partner Link window, provide an appropriate name and click on the SOA Resource Browser tool, as shown in Figure 4–78.

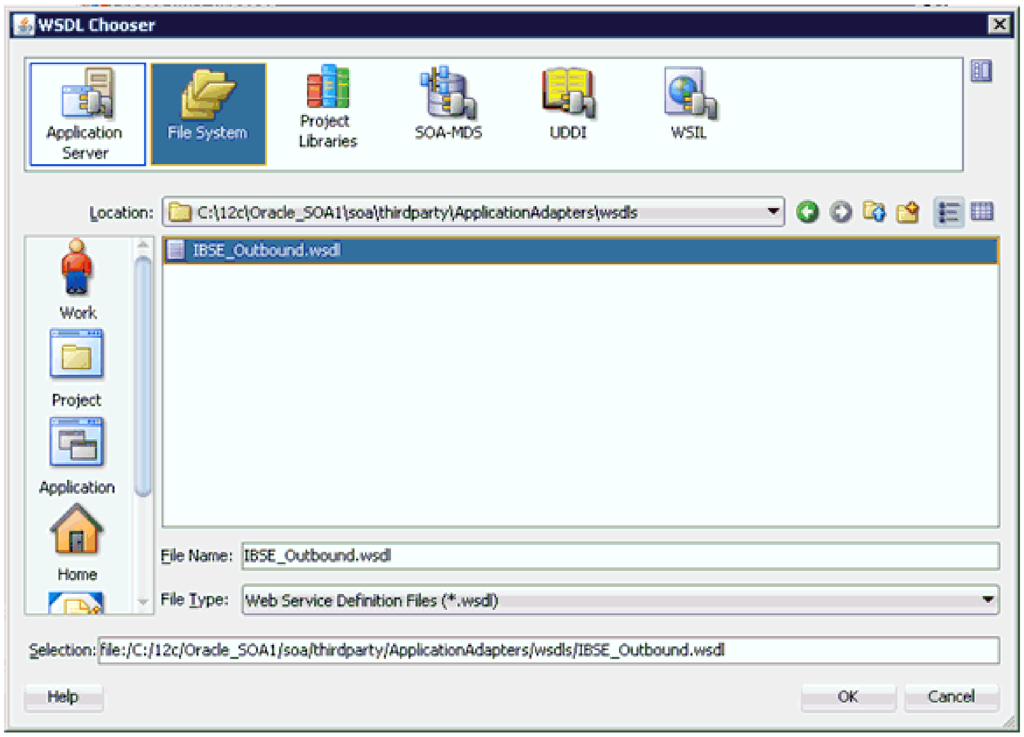
Figure 4–78 SOA Resource Browser Tool



Name: PartnerLink1
Process:
WSDL URL:
Partner Link Type:
Partner Role:
My Role:

4. In the WSDL Chooser dialog that is displayed, navigate to the location where the WSDL is exported from Application Explorer, select the WSDL, and click **OK**, as shown in Figure 4–79.

Figure 4–79 WSDL Chooser Dialog



Location: C:\12c\Oracle_SOA1\soa\thirdparty\ApplicationAdapters\wsdls

IBSE_Outbound.wsdl

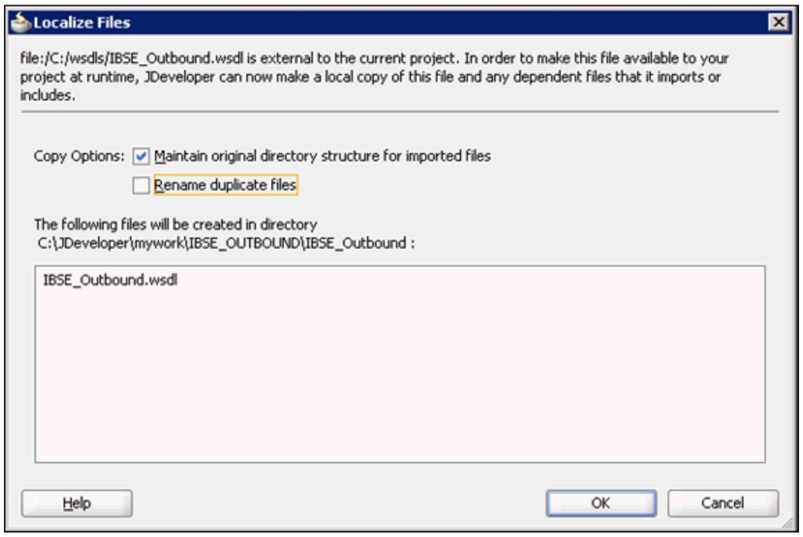
File Name: IBSE_Outbound.wsdl

File Type: Web Service Definition Files (*.wsdl)

Selection: file:/C:/12c/Oracle_SOA1/soa/thirdparty/ApplicationAdapters/wsdls/IBSE_Outbound.wsdl

5. In the displayed Localize Files window, uncheck the **Rename duplicate files** check box and click **OK**, as shown inFigure 4–80.

Figure 4–80 Localize Files Window



file:/C:/wsdls/IBSE_Outbound.wsdl is external to the current project. In order to make this file available to your project at runtime, JDeveloper can now make a local copy of this file and any dependent files that it imports or includes.

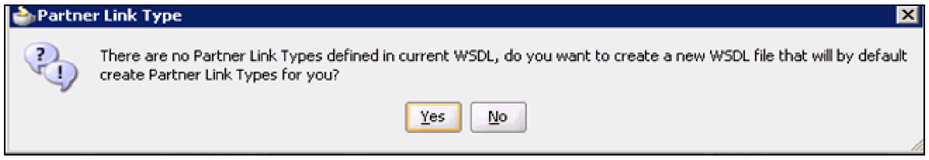
Copy Options: [✓] Maintain original directory structure for imported files
[ ] Rename duplicate files

The following files will be created in directory C:\JDeveloper\mywork\IBSE_OUTBOUND\IBSE_Outbound :

IBSE_Outbound.wsdl

6. Click **Yes** in the displayed Partner Link Type window, as shown in Figure 4–81.

Figure 4–81 Partner Link Type

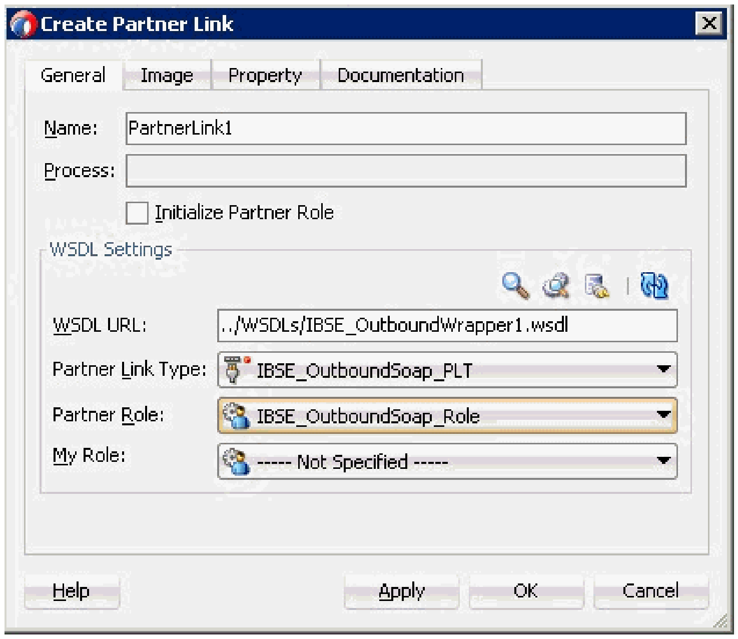


Partner Link Type

There are no Partner Link Types defined in current WSDL, do you want to create a new WSDL file that will by default create Partner Link Types for you?

7. In the displayed Create Partner Link window, expand the **Partner Role** drop-down list and select the available partner role.
8. Click **Apply**, and then **OK**, as shown in Figure 4–82.

Figure 4–82 Create Partner Link



Name: PartnerLink1

Process:

Initialize Partner Role

WSDL Settings

WSDL URL: ../WSDLs/IBSE_OutboundWrapper1.wsdl

Partner Link Type: IBSE_OutboundSoap_PLT

Partner Role: IBSE_OutboundSoap_Role

My Role: ----- Not Specified -----

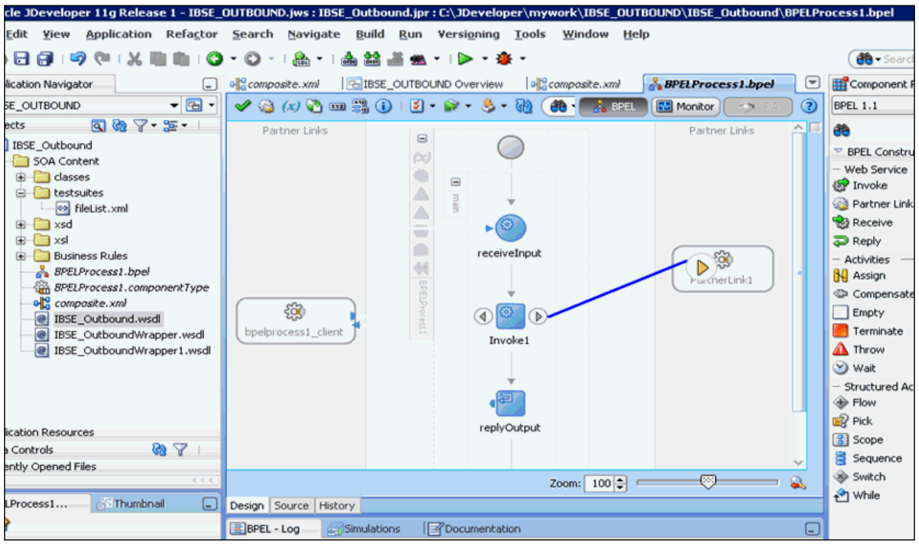
4.6.3.2 Creating BPEL Activities and Mappings With the Created Partner Link

This section describes how to create BPEL activities and mappings with the created partner link.

To create BPEL Activities and map with the created partner link:

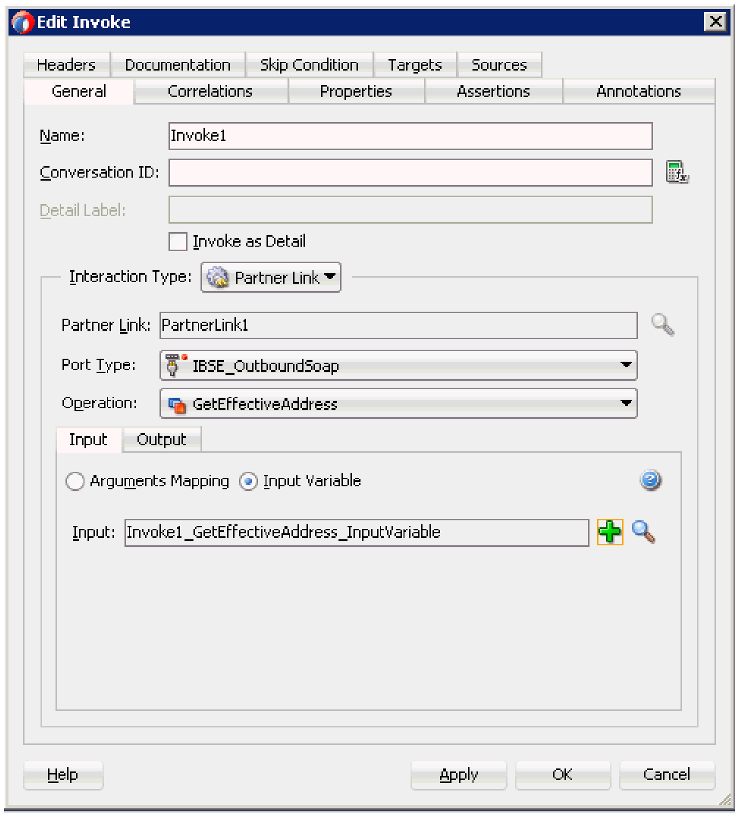
1. Drag and drop the **Invoke** activity component from BPEL Constructs to the Components pane. Place it between the **receiveInput** activity component and the **replyOutput** activity component.
2. Create a connection between the new **Invoke** activity component (Invoke1) and the **Partner Link** component (Partner link1), as shown in Figure 4–83.

Figure 4–83 Partner Link Component



3. In the displayed Edit Invoke window, click the Plus (**+**) icon, located to the right of the Input field, to configure a new input variable.
4. Accept the default values that are provided for the new input variable and click **OK**.
5. Click the Plus (**+**) icon, which is located to the right of the Output field, to configure a new output variable, as shown in Figure 4–84.

Figure 4–84 Edit Invoke Window


Edit Invoke

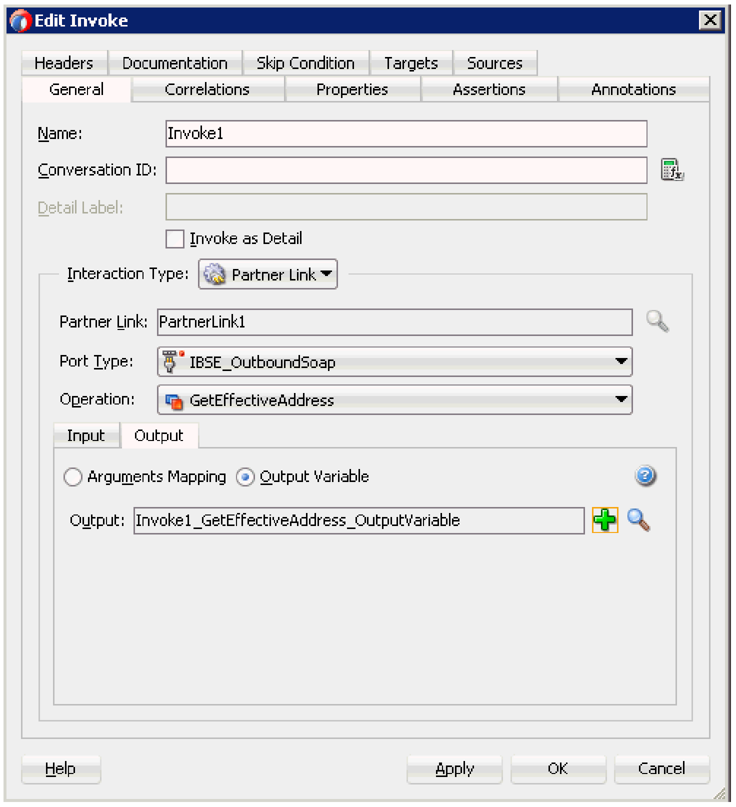
Headers | Documentation | Skip Condition | Targets | Sources
General | Correlations | Properties | Assertions | Annotations

Name: Invoke1
Conversation ID:
Detail Label:
[ ] Invoke as Detail
Interaction Type: Partner Link
Partner Link: PartnerLink1
Port Type: IBSE_OutboundSoap
Operation: GetEffectiveAddress

Input | Output
( ) Arguments Mapping [✓ Input Variable]
Input: Invoke1_GetEffectiveAddress_InputVariable

6. Accept the default values that are provided for the new output variable and click **OK**.
7. Click **Apply** and then **OK**, as shown in Figure 4–85.

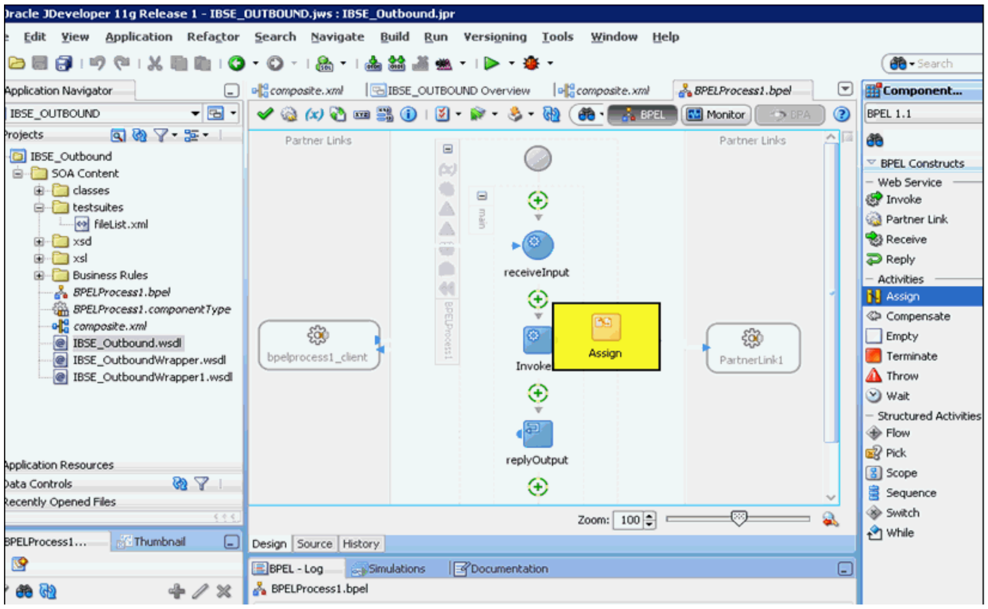
Figure 4–85 Edit Invoke Window



Name: Invoke1
Conversation ID:
Detail Label:
[ ] Invoke as Detail
Interaction Type: Partner Link
Partner Link: PartnerLink1
Port Type: IBSE_OutboundSoap
Operation: GetEffectiveAddress
Input | Output
( ) Arguments Mapping (●) Output Variable
Output: Invoke1_GetEffectiveAddress_OutputVariable

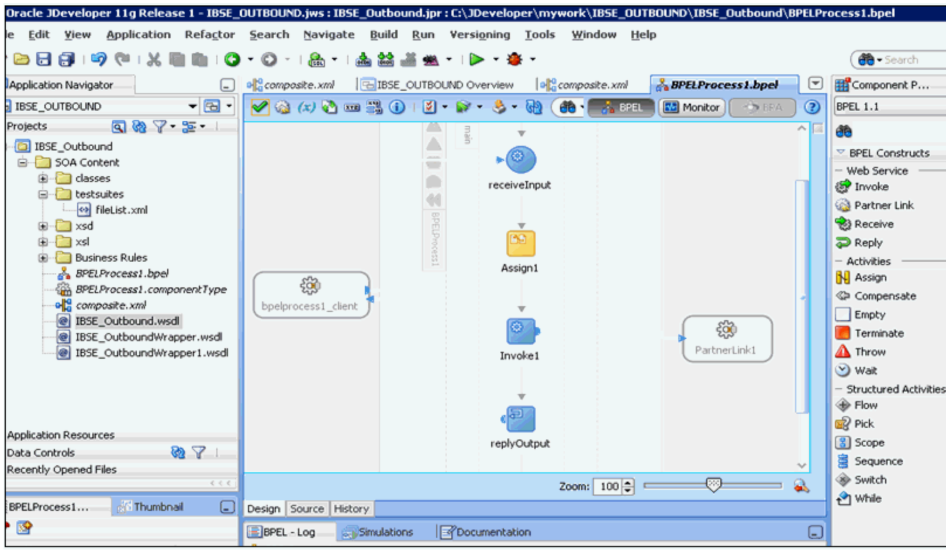
8. Drag and drop the **Assign** activity component from BPEL Constructs to the Components pane. Place it between the **Receive** activity component (receiveInput) and the **Invoke** activity component (Invoke1), as shown in Figure 4–86.

Figure 4–86 Assign Activity Component



9. Double-click the new **Assign** activity component (Assign1), as shown in Figure 4–87.

Figure 4–87 Assign Activity Component

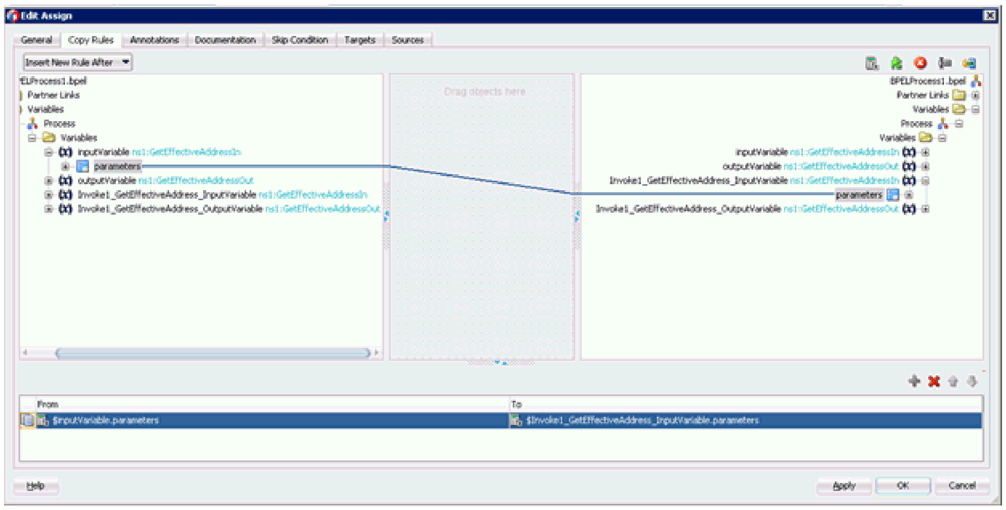

Oracle JDeveloper 11g Release 1 - IBSE_OUTBOUND.jws : IBSE_Outbound.jpr : C:\JDeveloper\mywork\IBSE_OUTBOUND\IBSE_Outbound\BPELProcess1.bpel
Application Navigator
IBSE_OUTBOUND
Projects
receiveInput
Assign1
Invoke1
replyOutput
bpelprocess1_client
PartnerLink1
Zoom: 100
BPEL Constructs
Assign

10. In the left pane, under Variables, expand **InputVariable**, and then select **parameters**.

11. In the right pane, under Variables, expand **Invoke1_GetEffectiveAddress_InputVariable**, and then select **parameters**.

12. Drag and map the **InputVariable** parameters to the **Invoke1_GetEffectiveAddress_InputVariable** parameters, as shown in Figure 4–88.

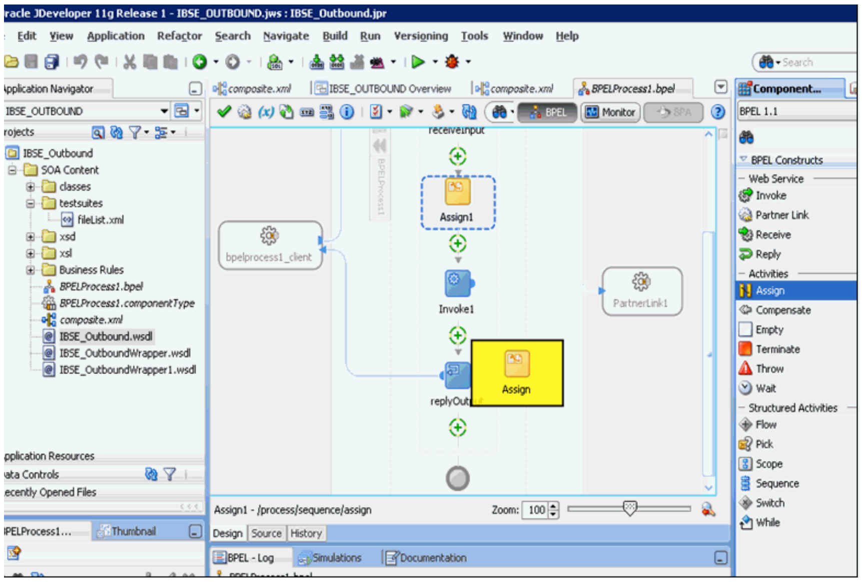
Figure 4–88 InputVariable Parameters



13. Click **Apply** and then **OK**.

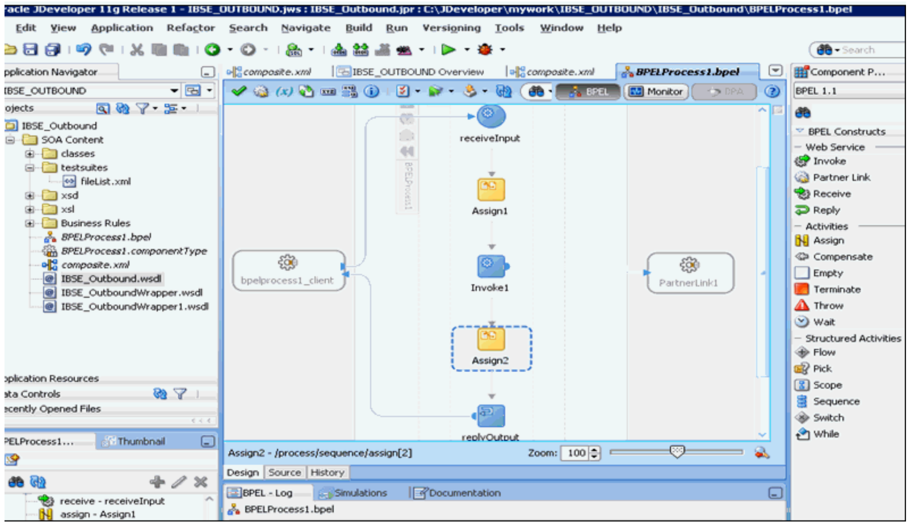
14. Drag and drop the **Assign** activity component to the Components pane and place it between the **Invoke** activity (Invoke1) and the **Reply** activity (replyOutput), as shown in Figure 4–89.

Figure 4–89 Assign Activity Component



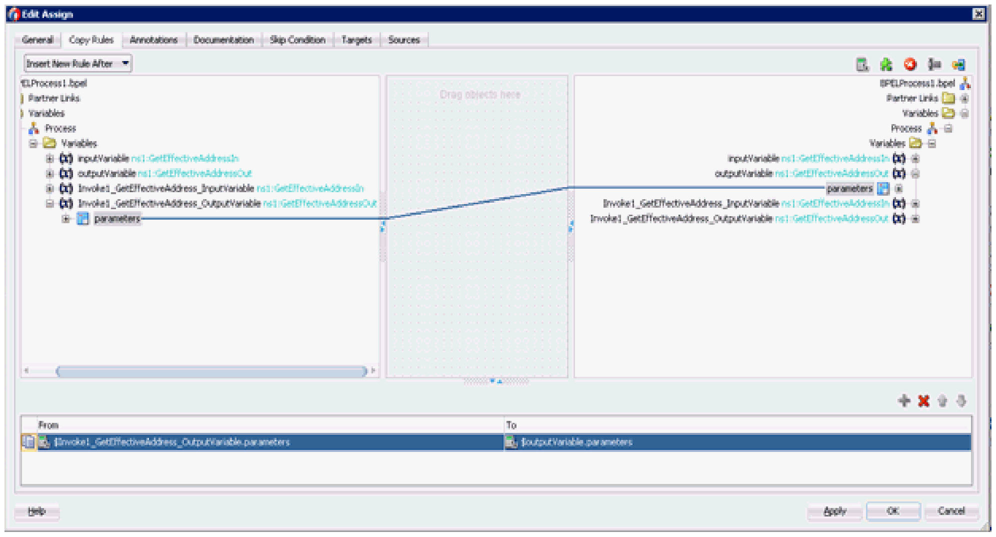
15. Double-click the new **Assign** activity component (Assign2), as shown in Figure 4–90.

Figure 4–90   New Assign Activity Component



16. In the left pane, under Variables, expand **Invoke1_GetEffectiveAddress_OutputVariable**, and then select **parameters**.
17. In the right pane, under Variables, expand **outputVariable**, and then select **parameters**.
18. Drag and map the **Invoke1_GetEffectiveAddress_OutputVariable** parameters to the **outputVariable** parameters, as shown in Figure 4–91.

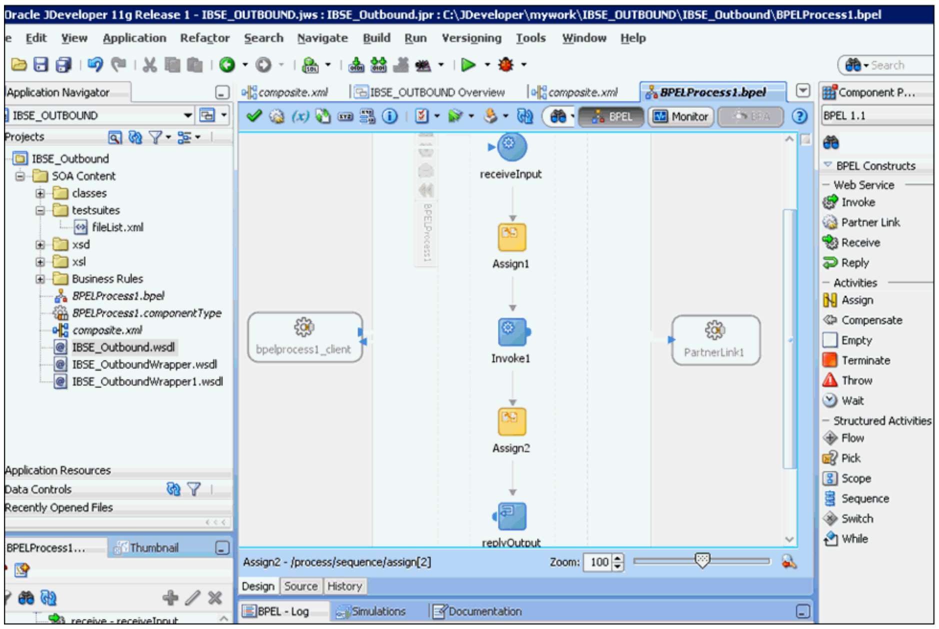
Figure 4–91   outputVariable Parameters



19. Click **Apply** and then **OK**.

    You are returned to the component pane, as shown in Figure 4–92.

Figure 4–92 Component Pane



20. Click the **Save All** icon in the menu bar to save the new outbound BPEL process component that was configured.

    You are now ready to deploy the BPEL Outbound process. You can follow the same procedure as in Section 4.4.4, "Deploying the BPEL Outbound Process" on page 4-28.

    Once deployed you can invoke the input XML, as defined in Section 4.4.5, "Invoking the Input XML Document in the Oracle Enterprise Manager Console" on page 4-31.

5 Integration With Mediator Service Components in the Oracle SOA Suite

This chapter describes integration with Mediator service components in the Oracle SOA Suite. It contains the following sections:

- Section 5.1, "Configuring a New Application Server Connection"
- Section 5.2, "Configuring a Mediator Outbound Process (J2CA Configuration)"
- Section 5.3, "Configuring a Mediator Inbound Process (J2CA Configuration)"
- Section 5.4, "Configuring a Mediator Outbound Process (BSE Configuration)"

The scenarios shown in this chapter require the following prerequisites.

Prerequisites

The following are installation and configuration requirements:

- Oracle Application Adapter for J.D. Edwards OneWorld must be installed on Oracle WebLogic Server.
- J.D. Edwards OneWorld must be configured for inbound and outbound processing.

See Also: *Oracle Application Server Adapter Concepts Guide*

The examples in this chapter present the configuration steps necessary for demonstrating service and event integration with J.D. Edwards OneWorld. Prior to using this material, you must be familiar with the following:

- How to configure Oracle Application Adapter for J.D. Edwards OneWorld for services and events. For more information, see Chapter 2, "Configuring Oracle Application Adapter for J.D. Edwards OneWorld".
- How to configure Oracle JDeveloper. For more information, see Chapter 4, "Integration With BPEL Service Components in the Oracle SOA Suite".

Overview of Mediator Integration

Mediator provides a comprehensive application integration framework. Oracle Application Adapter for J.D. Edwards OneWorld used with Mediator enables you to seamlessly integrate enterprise software, eliminating the need to write custom code. Functional modeling, as opposed to custom coding solutions, allows for software reuse and reduces the complexity and management challenges that arise over the software lifecycle. This integration model consists of two components--high-level integration logic and low-level platform services.

Adapter integration with Oracle WebLogic Server, Mediator is a two-step process:

1. **Design Time:** Oracle Application Adapter for J.D. Edwards OneWorld is configured in Application Explorer for services and events, as described in Chapter 2, "Configuring Oracle Application Adapter for J.D. Edwards OneWorld". Integration logic is modeled in iStudio. Metadata are stored in repositories.
2. **Runtime:** The underlying platform treats this metadata as run-time instructions to enable the communication between participating applications.

5.1 Configuring a New Application Server Connection

For more information on how to configure a new Application Server connection in Oracle JDeveloper, see Section 4.3, "Configuring a New Application Server Connection" on page 4-2.

5.2 Configuring a Mediator Outbound Process (J2CA Configuration)

This section describes how to configure a Mediator outbound process to your J.D. Edwards OneWorld system, using a Mediator project in Oracle JDeveloper.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\Mediator\J2CA\Outbound_Project
```

This section contains the following topics:

- Section 5.2.1, "Creating an Empty Composite for SOA"
- Section 5.2.2, "Defining a Mediator Outbound Process"
- Section 5.2.3, "Deploying the Mediator Outbound Process"
- Section 5.2.4, "Invoking the Input XML Document in the Oracle Enterprise Manager Console"

Prerequisites

Before you design a Mediator outbound process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.4.1, "Generating WSDL for Request/Response Service" on page 4-8.

5.2.1 Creating an Empty Composite for SOA

Perform the following steps to create an empty composite for SOA:

1. Create a new SOA application.
2. Enter a name for the new SOA Application and click **Next**.

   The Name your project page is displayed.
3. Enter a project name and click **Next**.

   The Configure SOA settings page is displayed.
4. From the Composite Template list, select **Empty Composite** and click **Finish**.

For more information, see Section 4.4.2, "Creating an Empty Composite for SOA" on page 4-9.

5.2.2 Defining a Mediator Outbound Process

This section describes how to define a Mediator outbound process, which consists of the following topics:

- Section 5.2.2.1, "Configuring a Third Party Adapter Service Component"
- Section 5.2.2.2, "Configuring an Outbound Mediator Process Component"
- Section 5.2.2.3, "Configuring the Routing Rules"
- Section 5.2.2.4, "Adjusting for Known Deployment Issues With 12c"

5.2.2.1 Configuring a Third Party Adapter Service Component

Perform the following steps to create a third party adapter service component:

1. Drag and drop the **Third Party Adapter** component from the Service Adapters pane to the External References pane.
2. Enter a name for the third party adapter service.
3. Ensure that **Reference** is selected from the Type drop-down list (default).
4. Click the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.
5. Browse and select an outbound WSDL file from the following directory:

   ```
   <ADAPTER_HOME>\wsdls
   ```

6. Click **OK**.
7. Click **OK**.

   The outbound WSDL file and associated request and response XML schema files (.xsd) are imported to the project folder that has been created.
8. Click the **Find JCA file** icon, which is located to the right of the JCA File field.
9. Browse and select the JCA properties file from the following directory:

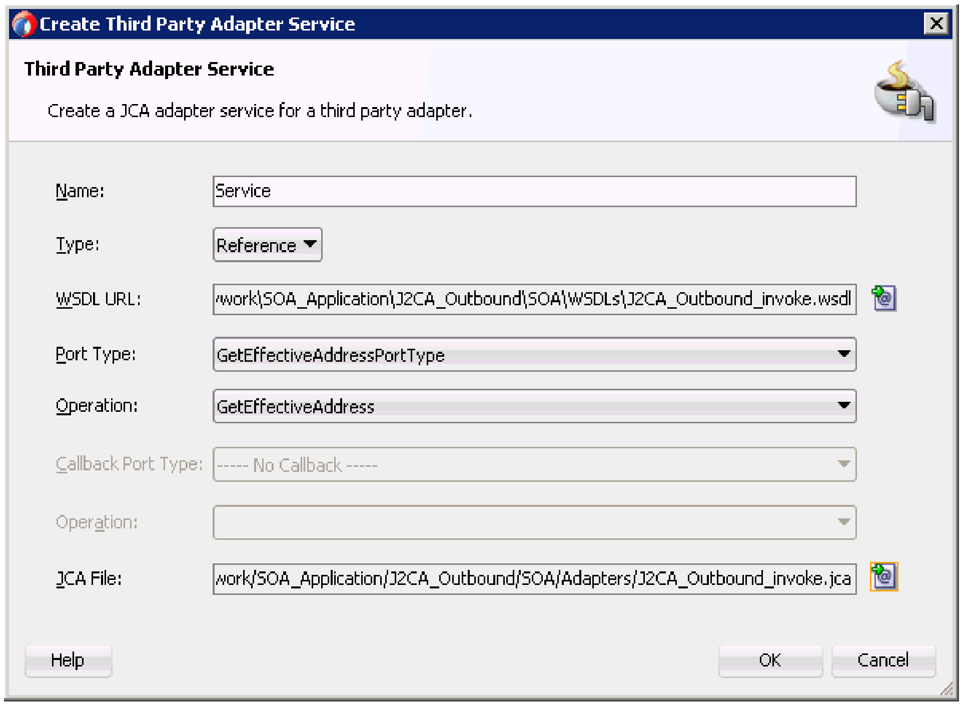
   ```
   <ADAPTER_HOME>\wsdls
   ```

10. Click **OK**.

    A Copy File confirmation message is displayed.
11. Click **Yes**.

    A copy of the JCA properties file is made in the project folder.

Figure 5–1 Create Third Party Adapter Service Dialog


Create Third Party Adapter Service

Third Party Adapter Service

Create a JCA adapter service for a third party adapter.

Name: Service

Type: Reference

WSDL URL: work\SOA_Application\J2CA_Outbound\SOA\WSDLs\J2CA_Outbound_invoke.wsdl

Port Type: GetEffectiveAddressPortType

Operation: GetEffectiveAddress

Callback Port Type: ----- No Callback -----

Operation:

JCA File: work/SOA_Application/J2CA_Outbound/SOA/Adapters/J2CA_Outbound_invoke.jca

12. Click **OK**.

    The third party adapter service component is created in the External References pane.

    You are now ready to configure an outbound Mediator process component.

    For more information, see Section 4.5.3.1, "Creating a Third Party Adapter Service Component" on page 4-40.

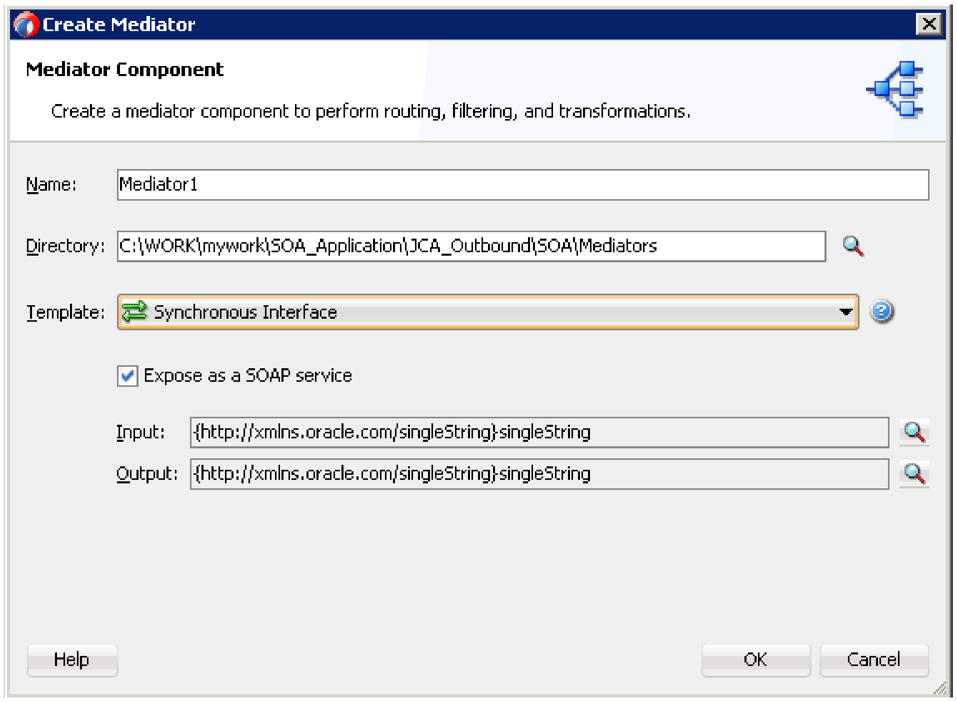
5.2.2.2 Configuring an Outbound Mediator Process Component

Perform the following steps to configure an outbound Mediator process component:

1. Drag and drop the **Mediator Process** component from the Components pane to the Components pane.

    The Create Mediator dialog is displayed, as shown in Figure 5–2.

Figure 5–2 Create Mediator Dialog



Create Mediator

Mediator Component

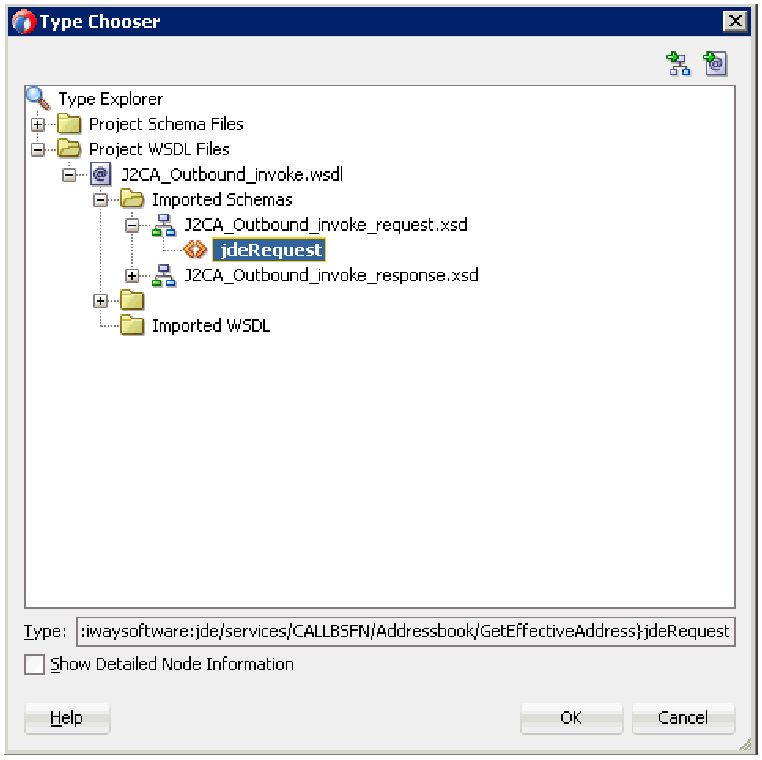
Create a mediator component to perform routing, filtering, and transformations.

Name:	Mediator1
Directory:	C:\WORK\mywork\SOA_Application\JCA_Outbound\SOA\Mediators
Template:	Synchronous Interface
	✓ Expose as a SOAP service
Input:	{http://xmlns.oracle.com/singleString}singleString
Output:	{http://xmlns.oracle.com/singleString}singleString

2. In the Name field, enter a name to identify the new outbound Mediator process component or leave it to the default value.
3. From the Template drop-down list, select **Synchronous Interface**.
4. Click the **Browse** icon, which is located to the right of the Input field to select the associated XML request schema file.

   The Type Chooser dialog is displayed, as shown in Figure 5–3.

Figure 5–3 Type Chooser Dialog


Type Chooser
Type Explorer
Project Schema Files
Project WSDL Files
J2CA_Outbound_invoke.wsdl
Imported Schemas
J2CA_Outbound_invoke_request.xsd
jdeRequest
J2CA_Outbound_invoke_response.xsd
Imported WSDL
Show Detailed Node Information
Help
OK
Cancel

Type: :iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress}jdeRequest

5. Expand **Project WSDL Files**, **J2CA_Outbound_invoke.wsdl**, **Imported Schemas**, **J2CA_Outbound_invoke_request.xsd**, and select **jdeRequest**.

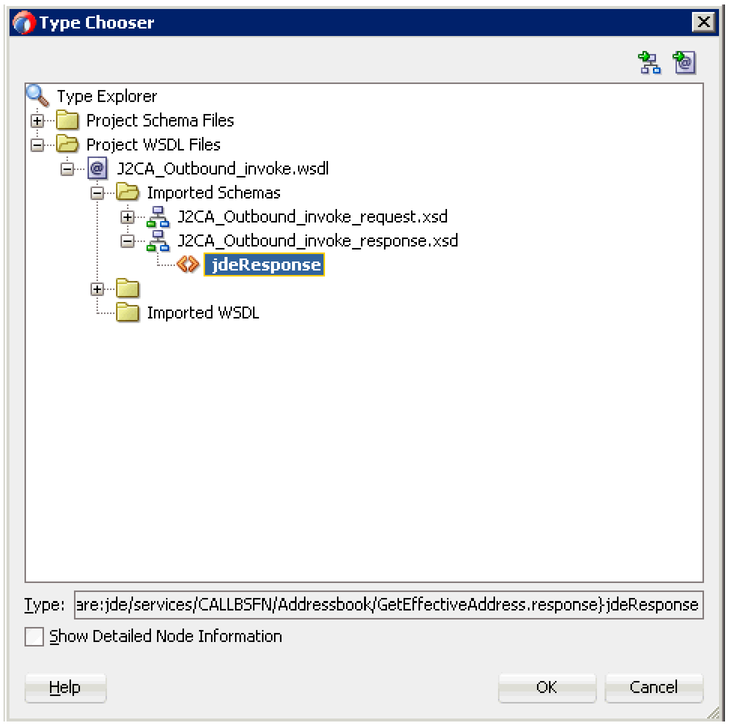
6. Click **OK**.

   You are returned to the Create Mediator dialog.

7. Click the **Browse** icon, which is located to the right of the Output field to select the associated XML response schema file.

   The Type Chooser dialog is displayed, as shown in Figure 5–4.

Figure 5–4 Type Chooser Dialog



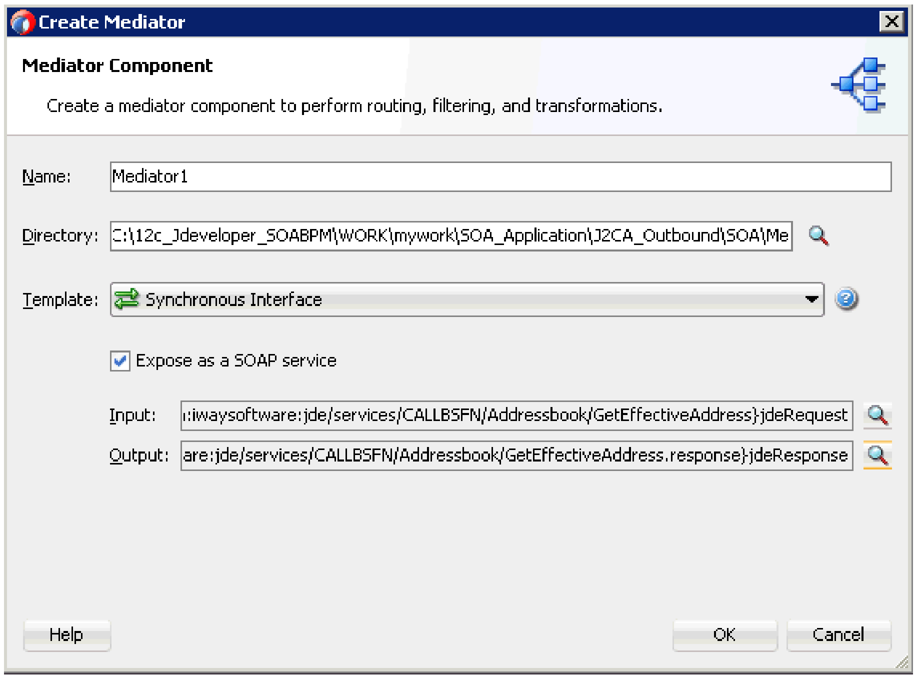
Type: are:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress.response}jdeResponse

☐ Show Detailed Node Information

8. Expand **Project WSDL Files**, **J2CA_Outbound_invoke.wsdl**, **Imported Schemas**, **J2CA_Outbound_invoke_response.xsd**, and select **jdeResponse**.
9. Click **OK**.

   You are returned to the Create Mediator dialog, as shown in Figure 5–5.

Figure 5–5 Create Mediator Dialog



Create a mediator component to perform routing, filtering, and transformations.

Name: Mediator1

Directory: C:\12c_Jdeveloper_SOABPM\WORK\mywork\SOA_Application\J2CA_Outbound\SOA\Me

Template: Synchronous Interface

☑ Expose as a SOAP service

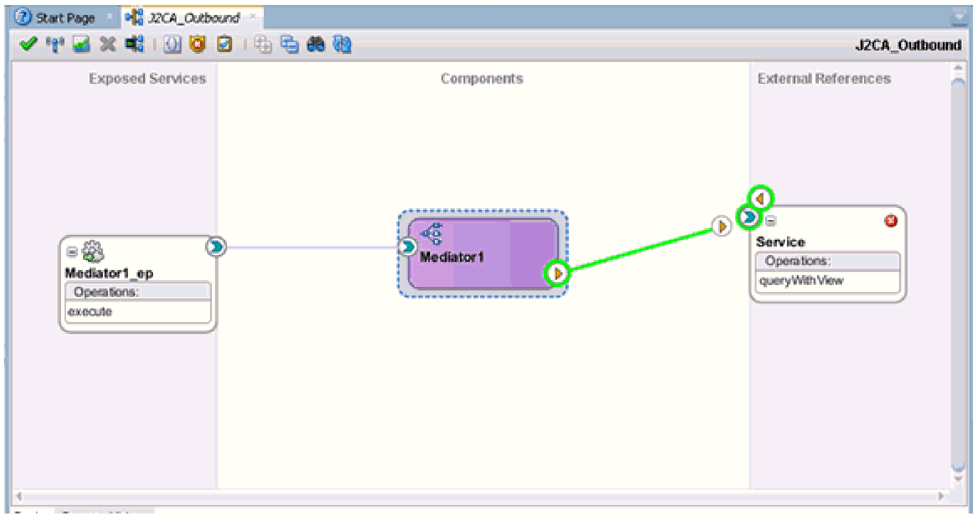
Input: ::iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress}jdeRequest

Output: are:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress.response}jdeResponse

10. Click **OK**.

11. Create a connection between the outbound Mediator process component and the third party adapter service component, as shown in Figure 5–6.

Figure 5–6 Created Connection



You are now ready to configure the routing rules.

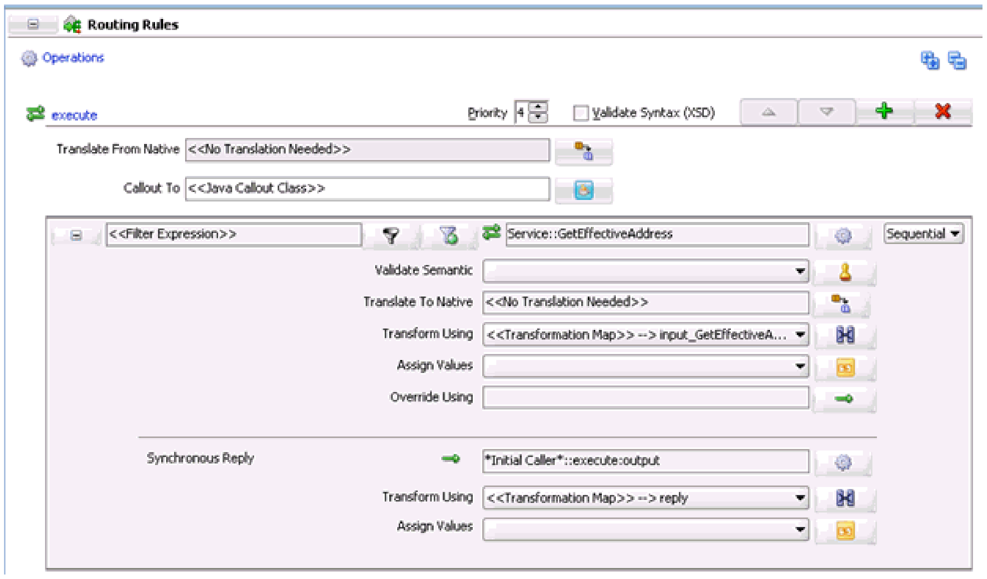
5.2.2.3 Configuring the Routing Rules

Perform the following steps to configure routing rules for the Mediator outbound process component:

1. Double-click the outbound Mediator process component in the Components pane.

   The Routing Rules dialog is displayed, as shown in Figure 5–7.

Figure 5–7 Routing Rules Dialog


Routing Rules

Operations

execute   Priority 4   Validate Syntax (XSD)

Translate From Native <<No Translation Needed>>

Callout To <<Java Callout Class>>

<<Filter Expression>>   Service::GetEffectiveAddress   Sequential

Validate Semantic

Translate To Native <<No Translation Needed>>

Transform Using <<Transformation Map>> --> input_GetEffectiveA...

Assign Values

Override Using

Synchronous Reply   *Initial Caller*::execute:output

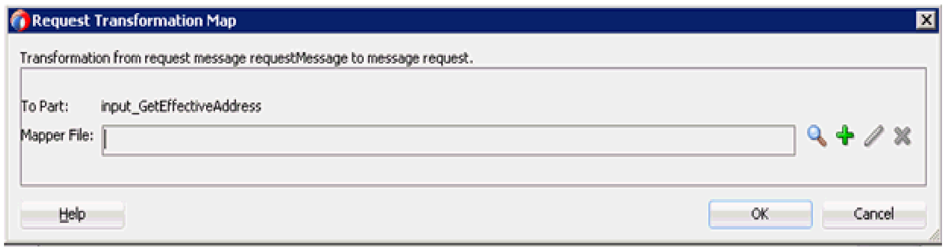
Transform Using <<Transformation Map>> --> reply

Assign Values

2. In the <<Filter Expression>> area, click the icon to the right of the Transform Using field.

   The Request Transformation Map dialog is displayed, as shown in Figure 5–8.

Figure 5–8 Request Transformation Map Dialog



Request Transformation Map

Transformation from request message requestMessage to message request.

To Part: input_GetEffectiveAddress

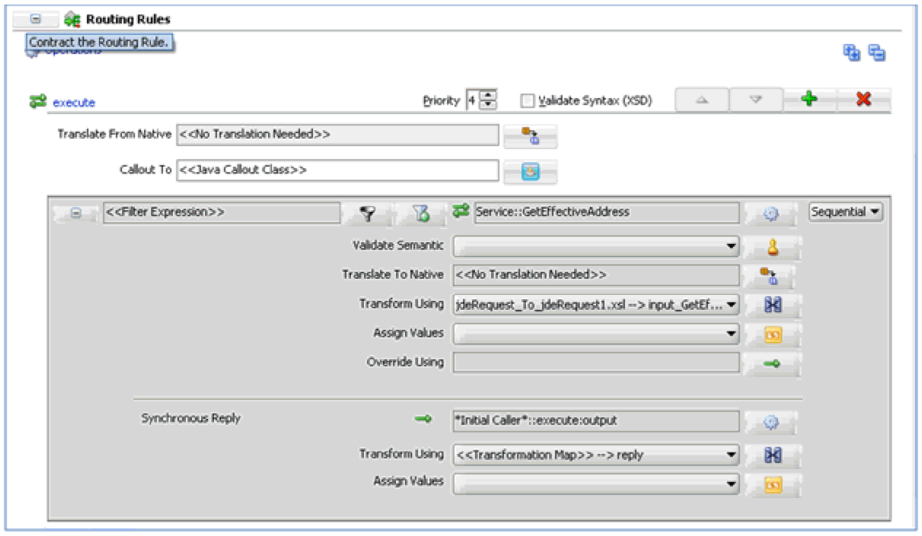
Mapper File:

3. Click the Add (+) icon.

   The Create Transformation Map page is displayed.
4. Make sure the Type is selected as **XSLT** and click **OK**.
5. Click **OK**.
6. Map the **ns0:jdeRequest** source element to the **ns0:jdeRequest** target element.

   The Auto Map Preferences dialog is displayed.
7. Retain the default values and click **OK**.
8. Return to the Routing Rules dialog, as shown in Figure 5–9.

Figure 5–9 Routing Rules Dialog



Routing Rules

Contract the Routing Rule.

execute

Priority 4

Validate Syntax (XSD)

Translate From Native <<No Translation Needed>>

Callout To <<Java Callout Class>>

<<Filter Expression>>

Service::GetEffectiveAddress

Sequential

Validate Semantic

Translate To Native <<No Translation Needed>>

Transform Using jdeRequest_To_jdeRequest1.xsl --> input_GetEf...

Assign Values

Override Using

Synchronous Reply

Initial Caller::execute:output

Transform Using <<Transformation Map>> --> reply

Assign Values

9. In the Synchronous Reply area, click the icon to the right of the Transform Using field.

   The Reply Transformation Map dialog is displayed.
10. Click the Add (+) icon.

    The create Transformation Page is displayed.
11. Make sure the type is selected as **XSLT** and click **OK**.

A mapping page is displayed.

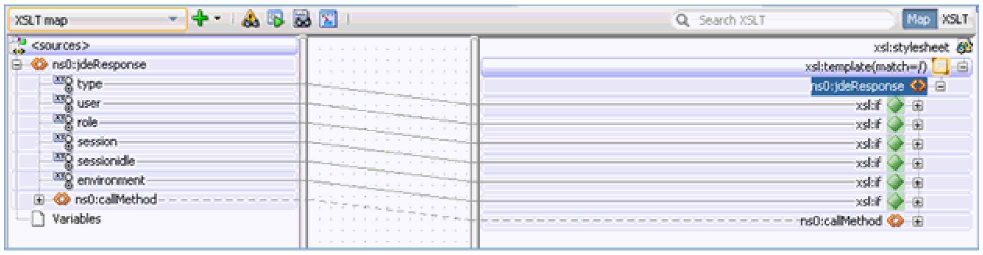
12. Click **OK**.
13. Map the **ns0:jdeResponse** source element to the **ns0:jdeResponse** target element.

    The Auto Map Preferences dialog is displayed.

14. Retain the default values and click **OK**.

    The mapping is completed, as shown in Figure 5–10.

Figure 5–10 Completed Mapping



15. Click the **Save All** icon in the menu bar to save the new outbound Mediator process component that was configured.

5.2.2.4 Adjusting for Known Deployment Issues With 12c

For more information on how to adjust for known deployment issues with 12c, see Section 4.4.3.3, "Adjusting for Known Deployment Issues With 12c" on page 4-26.

5.2.3 Deploying the Mediator Outbound Process

Perform the following steps to deploy the Mediator outbound process.

1. Right-click the project name in the left pane, select **Deploy**, and then click **J2CA_Outbound**.

   The Deployment Action page is displayed.

2. Ensure that **Deploy to Application Server** is selected.
3. Click **Next**.

   The Deploy Configuration page is displayed.

4. Leave the default values selected and click **Next**.

   The Select Server page is displayed.

5. Select an available application server that was configured and click **Next**.

   The SOA Servers page is displayed.

6. Select a target SOA server and click **Next**.

   The Summary page is displayed.

7. Review and verify all the available deployment information for your project and click **Finish**.

   For more information, see Section 4.4.4, "Deploying the BPEL Outbound Process" on page 4-28.

5.2.4 Invoking the Input XML Document in the Oracle Enterprise Manager Console

For more information, see Section 4.4.5, "Invoking the Input XML Document in the Oracle Enterprise Manager Console" on page 4-31.

5.3 Configuring a Mediator Inbound Process (J2CA Configuration)

This section describes how to configure a Mediator inbound process to your J.D. Edwards OneWorld system, using a Mediator project in Oracle JDeveloper.

A sample project has been provided for this inbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\Mediator\J2CA\Inbound_Project
```

This section contains the following topics:

- Section 5.3.1, "Creating an Empty Composite for SOA"
- Section 5.3.2, "Defining a Mediator Inbound Process"

Prerequisites

Before you design a Mediator inbound process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.

5.3.1 Creating an Empty Composite for SOA

Perform the following steps to create an empty composite for SOA:

1. Create a new SOA application.
2. Enter a name for the new SOA Application and click **Next**.

   The Name your project page is displayed.
3. Enter a project name and click **Next**.

   The Configure SOA settings page is displayed.
4. From the Composite Template list, select **Empty Composite** and click **Finish**.

   For more information, see Section 4.4.2, "Creating an Empty Composite for SOA" on page 4-9.

5.3.2 Defining a Mediator Inbound Process

This section describes how to define a Mediator inbound process, which contains the following topics:

- Section 5.3.2.1, "Configuring a Third Party Adapter Service Component"
- Section 5.2.2.2, "Configuring an Outbound Mediator Process Component"
- Section 5.2.2.3, "Configuring the Routing Rules"
- Section 5.3.2.4, "Adjusting for Known Deployment Issues With 12c"

5.3.2.1 Configuring a Third Party Adapter Service Component

Perform the following steps to create a third party adapter service component:

1. Drag and drop the **Third Party Adapter** component from the Service Adapters pane to the Exposed Services pane.

   The Create Third Party Adapter Service dialog is displayed.
2. Enter a name for the third party adapter service.
3. Ensure that **Service** is selected from the Type drop-down list (default).
4. Click the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.

   The WSDL Chooser dialog is displayed.
5. Browse and select an inbound WSDL file from the following directory:

   ```
   <ADAPTER_HOME>\wsdls
   ```
6. Click **OK**.

   The Localize Files dialog is displayed.
7. Click **OK**.

   The inbound WSDL file and associated receive/request schema file (.xsd) are imported to the project folder that has been created.

   You are returned to the Create Third Party Adapter Service dialog.
8. Click the **Find JCA file** icon, which is located to the right of the JCA File field.

   The Transformation Chooser dialog is displayed.
9. Browse and select the JCA properties file from the following directory:

   ```
   <ADAPTER_HOME>\wsdls
   ```
10. Click **OK**.

    The Copy File Confirmation message is displayed.
11. Click **Yes**.

    A copy of the JCA properties file is made in the project folder.

    You are returned to the Create Third Party Adapter Service dialog.
12. Click **OK**.

    The third party adapter service component is created in the Exposed Services pane.

    You are now ready to configure an inbound Mediator process component.

    For more information, see Section 6.5.3.1, "Creating a Third Party Adapter Service Component" on page 6-45.

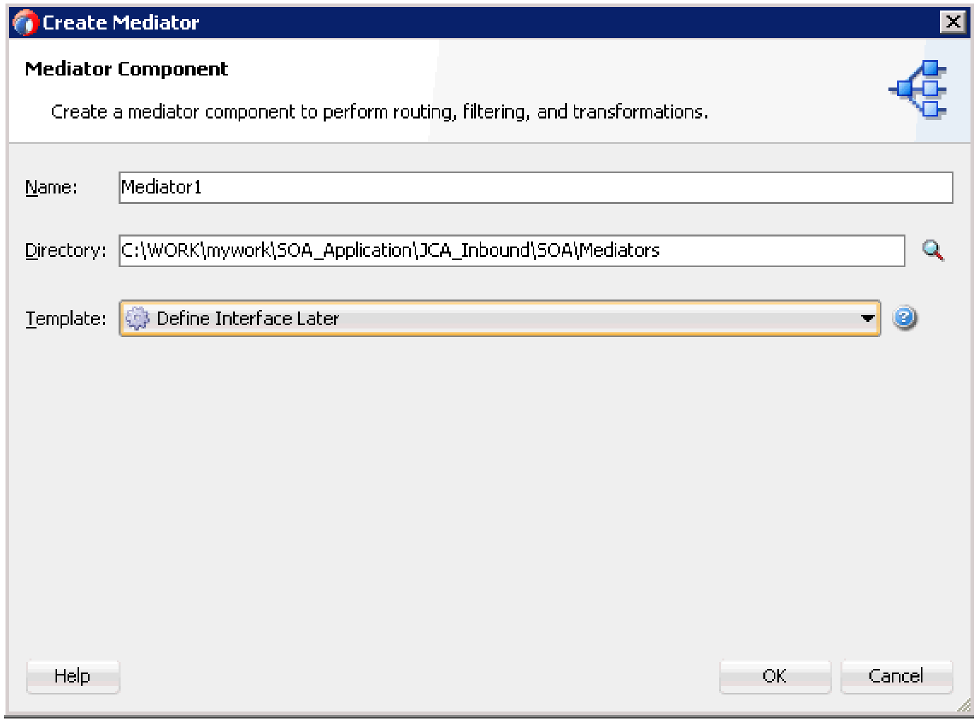
5.3.2.2 Configuring an Inbound Mediator Process Component With a File Adapter

Perform the following steps to configure an inbound Mediator process component with a File adapter.

1. Drag and drop the **Mediator Process** component from the Service Components pane to the Components pane.

   The Create Mediator dialog is displayed, as shown in Figure 5–11.

Figure 5–11 Create Mediator Dialog



Mediator Component

Create a mediator component to perform routing, filtering, and transformations.

Name: Mediator1

Directory: C:\WORK\mywork\SOA_Application\JCA_Inbound\SOA\Mediators

Template: Define Interface Later

2. In the Name field, enter a name to identify the new inbound Mediator process component.
3. From the Template drop-down list, select **Define Interface Later**.
4. Click the **OK**.

   The new Mediator process component is added to the Components pane.
5. Drag and drop the **File** component from the Technology Adapters pane to the External References pane.

   The File Adapter Configuration Wizard is displayed.
6. Type a name for the new File adapter and click **Next**.

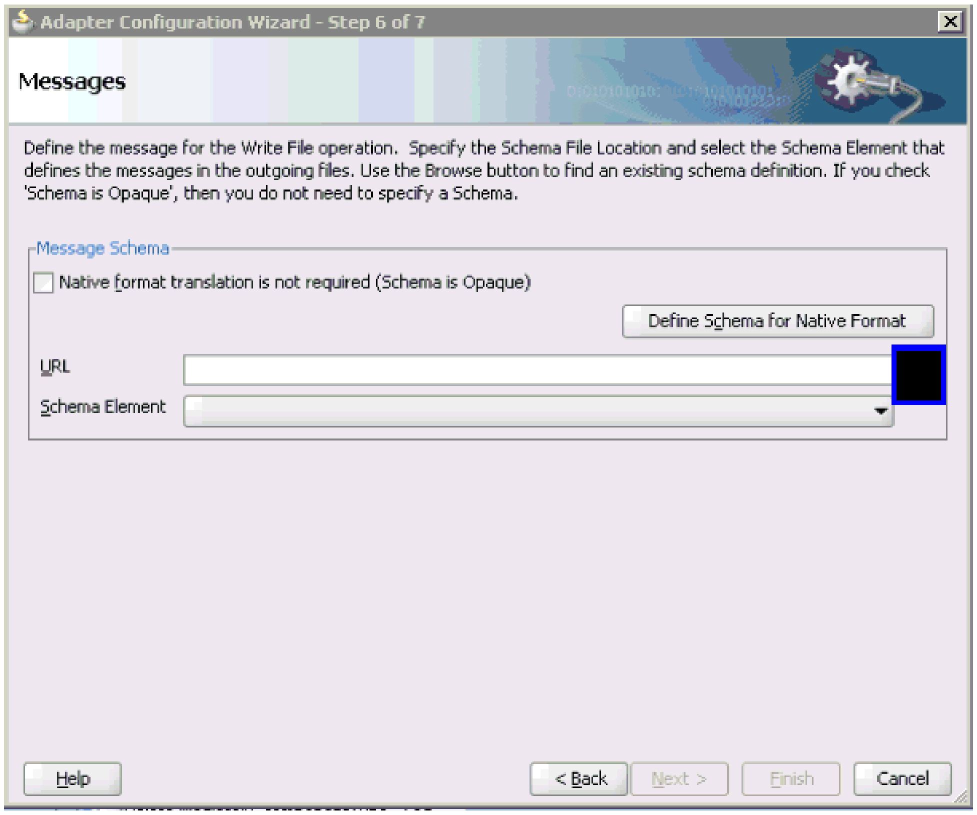
   The Adapter Interface page is displayed.
7. Ensure that the **Define from operation and schema (specified later)** option is selected.
8. Click **Next**.

   The Operation page is displayed.
9. Click **Next**.
10. Select **Write File** from the list of Operation Type options and specify an Operation Name (for example, Write).
11. Click **Next**.

    The File Configuration page is displayed.
12. Specify a location on your file system where the output file is written.
13. In the File Naming Convention field, specify a name for the output file.
14. Click **Next**.

The Messages page is displayed, as shown in Figure 5–12.

Figure 5–12 Messages Page


Adapter Configuration Wizard - Step 6 of 7

Messages

Define the message for the Write File operation. Specify the Schema File Location and select the Schema Element that defines the messages in the outgoing files. Use the Browse button to find an existing schema definition. If you check 'Schema is Opaque', then you do not need to specify a Schema.

Message Schema

Native format translation is not required (Schema is Opaque)

Define Schema for Native Format

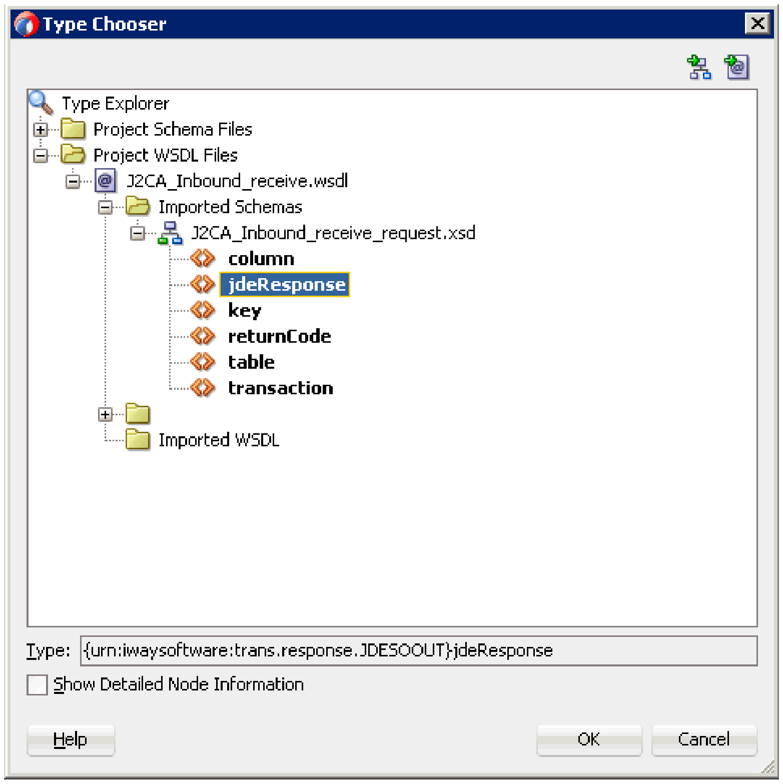
URL

Schema Element

15. Click **Browse**, which is located to the right of the URL field.

    The Type Chooser dialog is displayed, as shown in Figure 5–13.

Figure 5–13 Type Chooser Dialog



Type: {urn:iwaysoftware:trans.response.JDESOOUT}jdeResponse

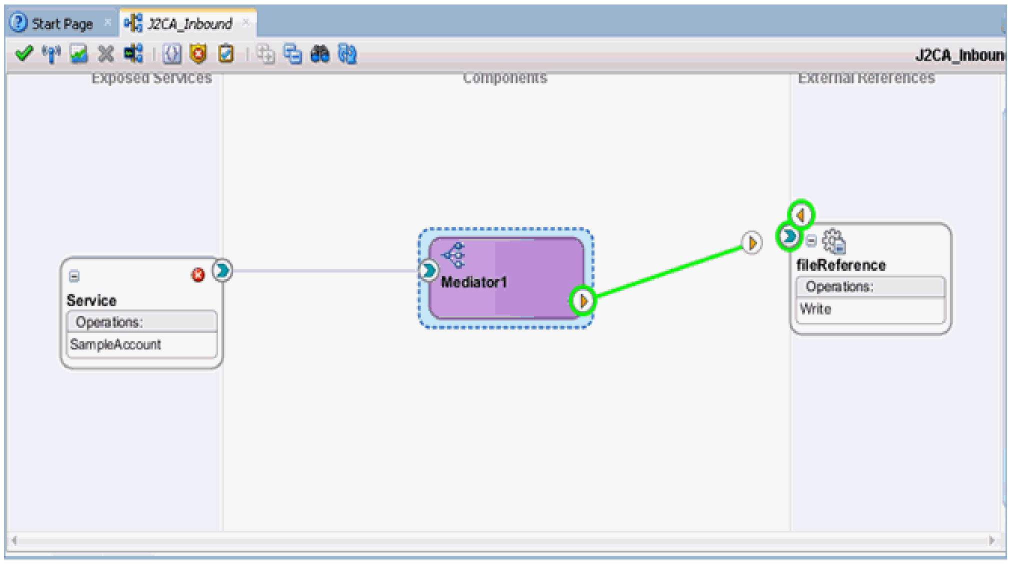
Show Detailed Node Information

16. Expand **Project WSDL Files**, **J2CA_Inbound_receive.wsdl**, **Imported Schemas**, **J2CA_Inbound_receive_request.xsd**, and select **jdeResponse**.
17. Click **OK**.

    You are returned to the Messages page.
18. Click **Next**.

    The Finish page is displayed.
19. Click **Finish**.
20. Create a connection between the inbound Mediator process component and the third party adapter service component.
21. Create a connection between the inbound Mediator process component and the File adapter component, as shown in Figure 5–14.

Figure 5–14 Created Connection



You are now ready to configure the routing rules.

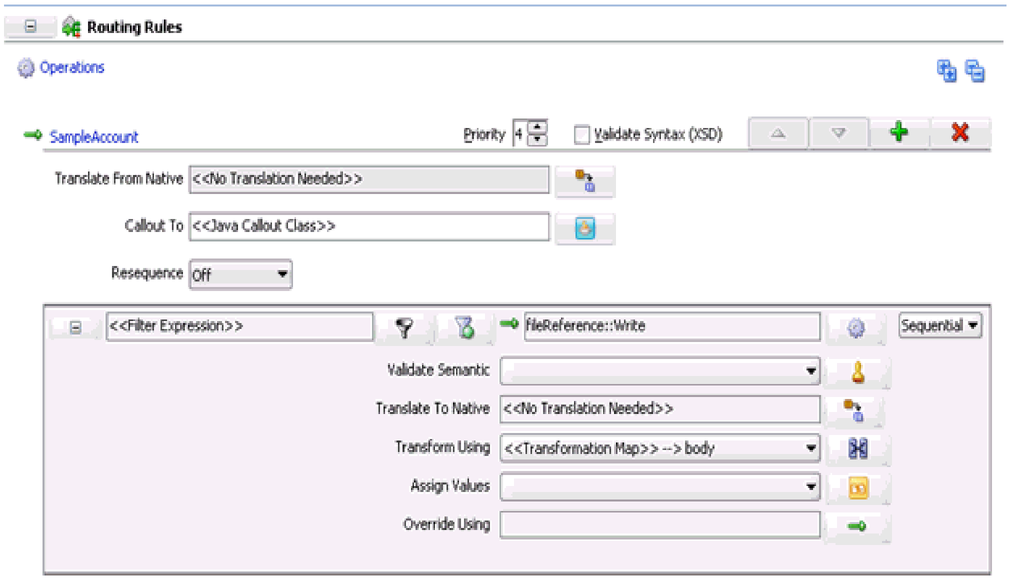
5.3.2.3 Configuring the Routing Rules

Perform the following steps to configure routing rules for the Mediator inbound process component:

1. Double-click the inbound Mediator process component in the Components page.

   The Routing Rules dialog is displayed, as shown in Figure 5–15.

Figure 5–15 Routing Rules Dialog



Priority 4

Translate From Native <<No Translation Needed>>

Callout To <<Java Callout Class>>

Resequence Off

<<Filter Expression>> fileReference::Write Sequential

Validate Semantic

Translate To Native <<No Translation Needed>>

Transform Using <<Transformation Map>> --> body

Assign Values

Override Using

2. In the <<Filter Expression>> area, click the icon to the right of the Transform Using field.

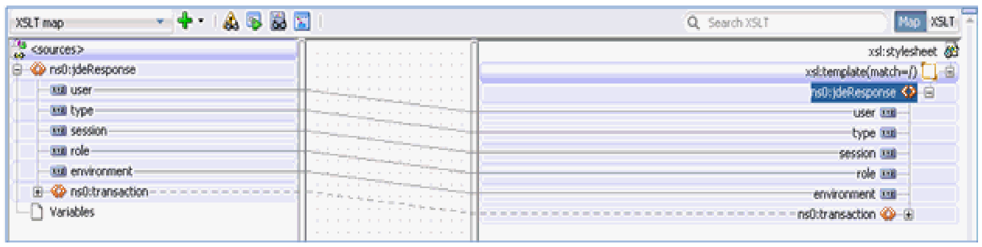
   The Request Transformation Map dialog is displayed.

3. Click the Add (+) icon and ensure that the selected Type is **XSLT**, then click **OK**.

4. Click **OK**.

   The mapping page is displayed, as shown in Figure 5–16.

Figure 5–16 Mapping Page



5. Click **OK**.
6. Map the **ns0:jdeResponse** source element to the **ns0:jdeResponse** target element.

   The Auto Map Preferences dialog is displayed.
7. Retain the default values and click **OK**.

   The mapping is now complete.
8. Click the **Save All** icon in the menu bar to save the new inbound Mediator process component that was configured.

5.3.2.4 Adjusting for Known Deployment Issues With 12c

For more information on how to adjust for known deployment issues with 12c, see Section 4.4.3.3, "Adjusting for Known Deployment Issues With 12c" on page 4-26.

You are now ready to deploy the Mediator inbound process. You can follow the same procedure in Section 4.5.4, "Deploying the BPEL Inbound Process" on page 4-46.

Once event messages are triggered through J.D. Edwards OneWorld, output XML is received in the location that was specified for the File adapter component. For more information on triggering events in J.D. Edwards OneWorld, see Section 4.5.5, "Triggering an Event in J.D. Edwards OneWorld" on page 4-47.

5.4 Configuring a Mediator Outbound Process (BSE Configuration)

This section describes how to configure a Mediator outbound process to your J.D. Edwards OneWorld system, using a Mediator project in Oracle JDeveloper.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\Mediator\BSE\Outbound_Project
```

This section contains the following topics:

- Section 5.4.1, "Creating an Empty Composite for SOA"
- Section 5.4.2, "Defining a Mediator Outbound Process"

  Prerequisites

  Before you design a Mediator outbound process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.6.1, "Generating a WSDL File for Request and Response Services Using a Web Service" on page 4-52.

5.4.1 Creating an Empty Composite for SOA

Perform the following steps to create an empty composite for SOA:

1. Create a new SOA application.
2. Enter a name for the SOA Application and click **Next**.
3. Enter a project name (for example, IBSE_Outbound), and click **Next**.
4. From the Composite Template list, select **Empty Composite** and click **Finish**.

For more information, see Section 4.6.2, "Creating an Empty Composite for SOA" on page 4-53.

5.4.2 Defining a Mediator Outbound Process

This section describes how to define a Mediator outbound process. The following topics are included:

- Section 5.4.2.1, "Configuring a SOAP Service"
- Section 5.4.2.2, "Creating a Mediator Component"
- Section 5.4.2.3, "Configuring the Routing Rules"

5.4.2.1 Configuring a SOAP Service

Perform the following steps to configure a SOAP Service:

1. Drag and drop the **SOAP** node from the Technology Adapters pane to the External References pane.
2. Enter an appropriate name for the SOAP Service and click on the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.
3. In the displayed SOA Resource Browser window, select the File system tab and navigate to the location where the WSDL is exported from the Application Explorer, select the WSDL, and click **OK**.
4. In the Create Web Service Window, click **OK**.
5. In the displayed Localize Files window, click **OK**. This imports the WSDL file to the project folder.

   The Web Service is created and displayed.

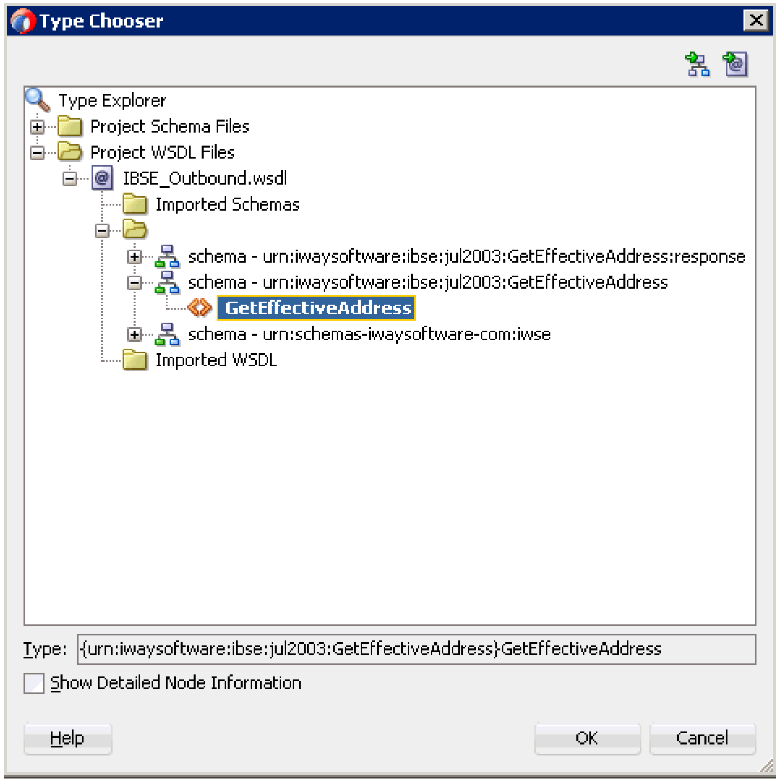
5.4.2.2 Creating a Mediator Component

Perform the following steps to create a Mediator component:

1. Drag and drop the **Mediator** component from the Components pane in to the Components pane.
2. In the Name field, enter a name to identify the new outbound Mediator process component.
3. From the Template drop-down list, select **Synchronous Interface**.
4. Click the **Browse** icon, which is located to the right of the Input field, to select the associated XML request schema file.

5. In the Type Chooser dialog, expand **Project WSDL Files**, select **IBSE_Outbound.wsdl**, and click **GetEffectiveAddress**, as shown in Figure 5–17.

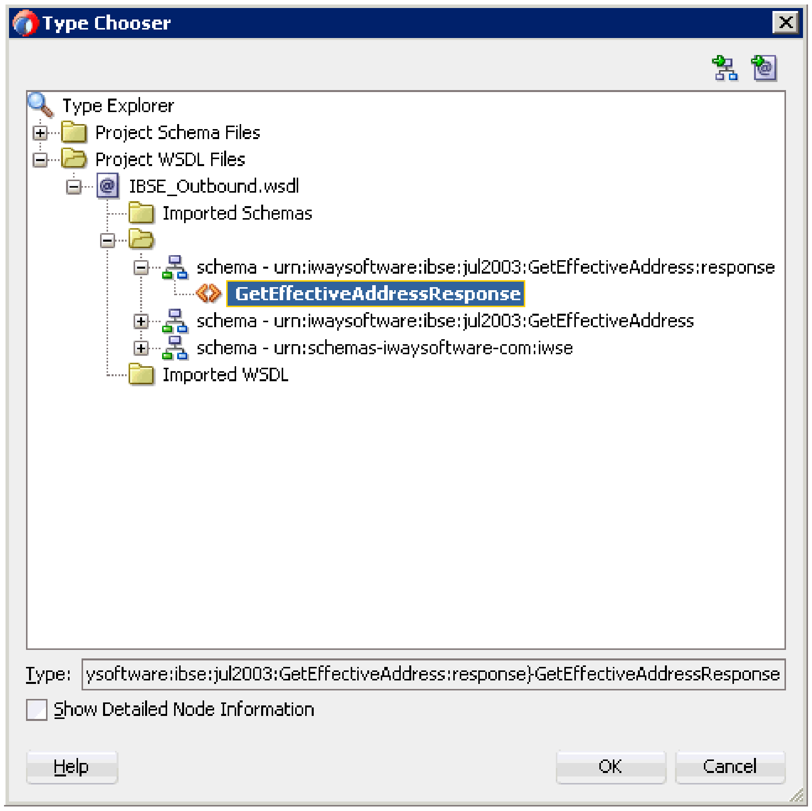
Figure 5–17 Type Chooser Dialog


Type Chooser
Type Explorer
Project Schema Files
Project WSDL Files
IBSE_Outbound.wsdl
Imported Schemas
schema - urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress:response
schema - urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress
GetEffectiveAddress
schema - urn:schemas-iwaysoftware-com:iwse
Imported WSDL
Show Detailed Node Information
Help
OK
Cancel

Type: {urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress}GetEffectiveAddress

6. Click **OK**.
7. Click the **Browse** icon, which is located to the right of the Output field, to select the associated XML response schema file.
8. In The Type Chooser dialog, expand **Project WSDL Files**, select **IBSE_Outbound.wsdl**, and click **GetEffectiveAddressResponse**, as shown in Figure 5–18.

Figure 5–18 Type Chooser Dialog


Type Chooser

Type Explorer
Project Schema Files
Project WSDL Files
IBSE_Outbound.wsdl
Imported Schemas
schema - urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress:response
GetEffectiveAddressResponse
schema - urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress
schema - urn:schemas-iwaysoftware-com:iwse
Imported WSDL

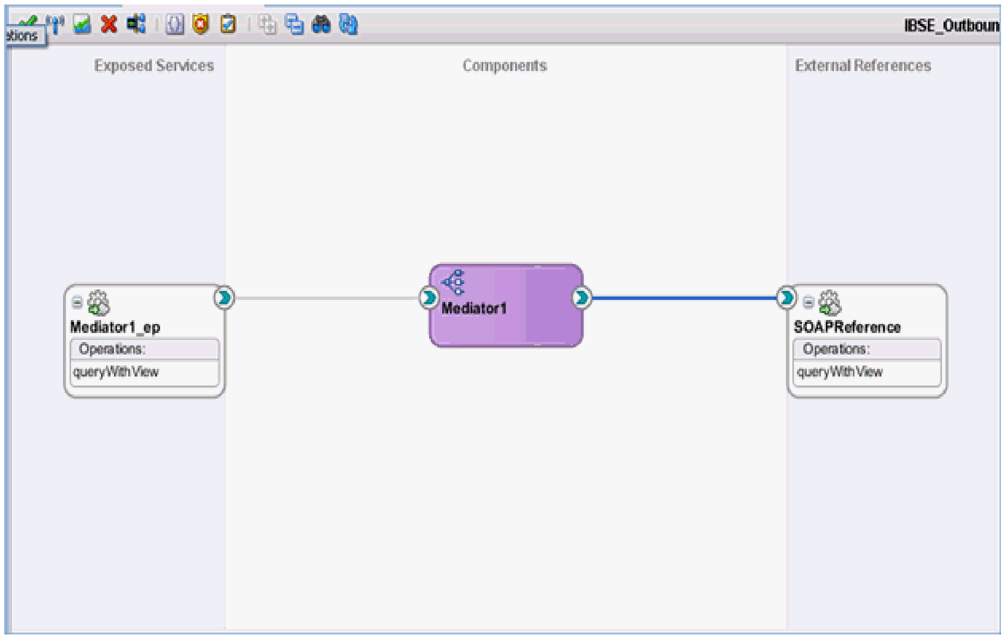
Type: ysoftware:ibse:jul2003:GetEffectiveAddress:response}GetEffectiveAddressResponse

Show Detailed Node Information

9. Click **OK**.
10. Click **OK**.

    The Mediator component is created and displayed.
11. Create a connection between the **Mediator** component and the **SOAP service** component, as shown in Figure 5–19.

Figure 5–19 Created Connection

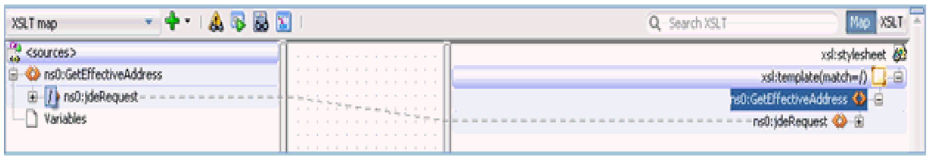


5.4.2.3 Configuring the Routing Rules

Perform the following steps to configure the routing rules:

1. Double-click the **Mediator** component in the Components pane.
2. In the <<Filter Expression>> area of the Static Routing section, click the icon to the right of the Transform Using field.
3. In the displayed Request Transformation Map window, click the Add (+) icon and make sure the selected Type is **XSLT** in the Create Transformation Map dialog box and click **OK**.
4. Click **OK**.
5. Map the **ns0:GetEffectiveAddress** source element to the **ns0:GetEffectiveAddress** target element, as shown in Figure 5–20.

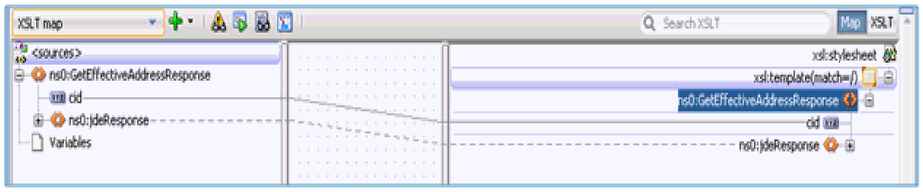
Figure 5–20 Source and Target Elements



6. In the displayed Auto Map Preferences window, retain the default values and click **OK**.
7. In the Synchronous Reply area, click the icon to the right of the Transform Using field.
8. In the displayed Reply Transformation Map window, click the Add (+) icon and make sure the Type is selected as **XSLT** in the Create Transformation Map dialog box, and then click **OK**.

9. Map the **ns0:GetEffectiveAddressResponse** source element to the **ns0:GetEffectiveAddressResponse** target element, as shown in Figure 5–21.

Figure 5–21 Source and Target Elements

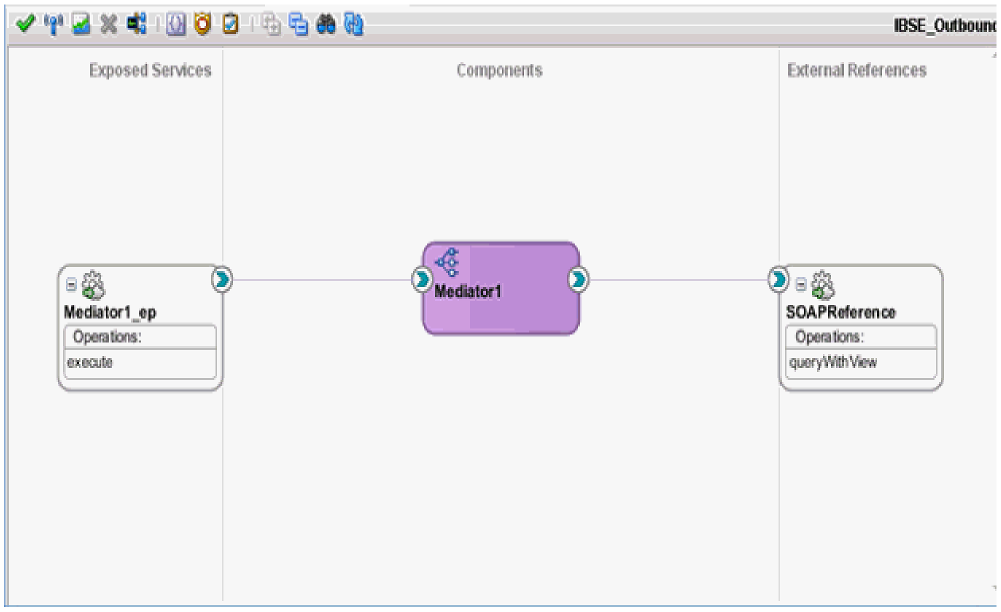


10. In the displayed Auto Map Preferences window, retain the default values and click **OK**.

11. Double-click **composite.xml** in the left pane.

12. Click the **Save All** icon in the menu bar to save the new outbound Mediator component that was configured, as shown in Figure 5–22.

Figure 5–22 Save All Icon



You are now ready to deploy the Mediator IBSE outbound process. You can follow the same procedure found in Section 5.2.3, "Deploying the Mediator Outbound Process" on page 5-10.

Once deployed, you can invoke the input XML, as defined in Section 5.2.4, "Invoking the Input XML Document in the Oracle Enterprise Manager Console" on page 5-11.

6 Integration With BPM Service Components in the Oracle SOA Suite

Oracle Application Adapter for J.D. Edwards OneWorld integrates seamlessly with Oracle Business Process Management (BPM) to facilitate Web service integration. Oracle BPM is based on the Service-Oriented Architecture (SOA). It consumes adapter services exposed as Web Service Definition Language (WSDL) documents.

This chapter contains the following topics:

- Section 6.1, "Overview"
- Section 6.2, "Deployment of Adapter"
- Section 6.3, "Configuring a New Application Server Connection"
- Section 6.4, "Designing an Outbound BPM Process Using Transformations for Service Integration (J2CA Configuration)"
- Section 6.5, "Designing an Inbound BPM Process Using Transformations for Event Integration (J2CA Configuration)"
- Section 6.6, "Designing an Outbound BPM Process Using Transformations for Service Integration (BSE Configuration)"

6.1 Overview

To integrate with Oracle BPM, Oracle Application Adapter for J.D. Edwards OneWorld must be deployed in the same WLS container as Oracle BPM. The underlying adapter services must be exposed as WSDL files, which are generated during design time in Oracle Adapter Application Explorer (Application Explorer) for both request-response (outbound) and event notification (inbound) services of the adapter. For more information, see "Generating WSDL (J2CA Configurations Only)" on page 2-11.

The generated WSDL files are used to design the appropriate BPM processes for inbound or outbound adapter services. A completed BPM process must be successfully compiled in JDeveloper and deployed to a BPM server. Upon deployment to the BPM server, every newly built process is automatically deployed to the Oracle Enterprise Manager console, where you run, monitor, and administer BPM processes, and listen to adapter events.

6.2 Deployment of Adapter

During installation, Oracle Application Adapter for J.D. Edwards OneWorld is deployed as a J2CA 1.0 resource adapter within the WLS container. The adapter must be deployed in the same WLS container as Oracle BPM.

6.3 Configuring a New Application Server Connection

For more information on how to configure a new Application Server connection in Oracle JDeveloper, see Section 4.3, "Configuring a New Application Server Connection" on page 4-2.

6.4 Designing an Outbound BPM Process Using Transformations for Service Integration (J2CA Configuration)

This section describes how to design an outbound BPM process using transformations for service integration.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\BPM\J2CA\Outbound_Project
```

The following tools are required to complete your outbound design-time configuration:

- Oracle Adapter Application Explorer (Application Explorer)
- Oracle JDeveloper BPM Designer (JDeveloper)

Note: The examples in this chapter demonstrate the use of JDeveloper.

This section contains the following topics:

- Section 6.4.1, "Creating an Empty Composite for BPM"
- Section 6.4.2, "Defining a BPM Outbound Process"
- Section 6.4.3, "Adjusting for Known Deployment Issues With 12c"
- Section 6.4.4, "Deploying the BPM Outbound Process"
- Section 6.4.5, "Invoking the Input XML Document in the Oracle Enterprise Manager Console"

Before you design a BPM process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.4.1, "Generating WSDL for Request/Response Service" on page 4-8.

6.4.1 Creating an Empty Composite for BPM

Perform the following steps to create an empty composite for BPM:

1. Create a new BPM application.
2. Enter a name for the new BPM application and click **Next**.

   The Name your project page is displayed.
3. Enter a project name, in the project features select **BPM**, and then click **Next**.

   The Configure SOA settings page is displayed.
4. From the Composite Template list, select **Empty Composite** and click **Finish**.

6.4.2 Defining a BPM Outbound Process

This section describes how to define a BPM outbound process, which contains the following topics:

- Section 6.4.2.1, "Configuring a Third Party Adapter Service Component"
- Section 6.4.2.2, "Configuring an Outbound BPM Process Component"
- Section 6.4.2.3, "Creating a File Adapter for the Write Operation"

6.4.2.1 Configuring a Third Party Adapter Service Component

Perform the following steps to create a third party adapter service component:

1. Double-click the created project to load the components.
2. Drag and drop the **Third Party Adapter** component from the Custom/Thirdparty pane to the External References pane.

   The Create Third Party Adapter Service dialog is displayed.
3. Enter a name for the third party adapter service.
4. Ensure that **Reference** is selected from the Type list (default).
5. Click the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.

   The WSDL Chooser dialog is displayed.
6. Browse and select an outbound WSDL file from the following directory:

   ```
   <ADAPTER_HOME>\wsdls
   ```
7. Click **OK**.

   The Localize Files dialog is displayed.
8. Click **OK**.

   The outbound WSDL file and associated request and response XML schema files (.xsd) are imported to the project folder that has been created.

   You are returned to the Create Third Party Adapter Service dialog.
9. Click the **Find JCA file** icon, which is located to the right of the JCA File field.

   The Transformation Map dialog is displayed.
10. Browse and select the JCA properties file from the following directory:

    ```
    <ADAPTER_HOME>\wsdls
    ```
11. Click **OK**.

    The Copy File message is displayed.
12. Click **Yes**.

    A copy of the JCA properties file is made in the project folder.

    You are returned to the Create Third Party Adapter Service dialog.
13. Click **OK**.

    The third party adapter service component is created and displayed in the External References pane.

    You are now ready to configure an outbound BPM process component.

For more detailed information, including screen shots, see Section 4.4.3.1, "Configuring a Third Party Adapter Service Component" on page 4-11.

6.4.2.2 Configuring an Outbound BPM Process Component

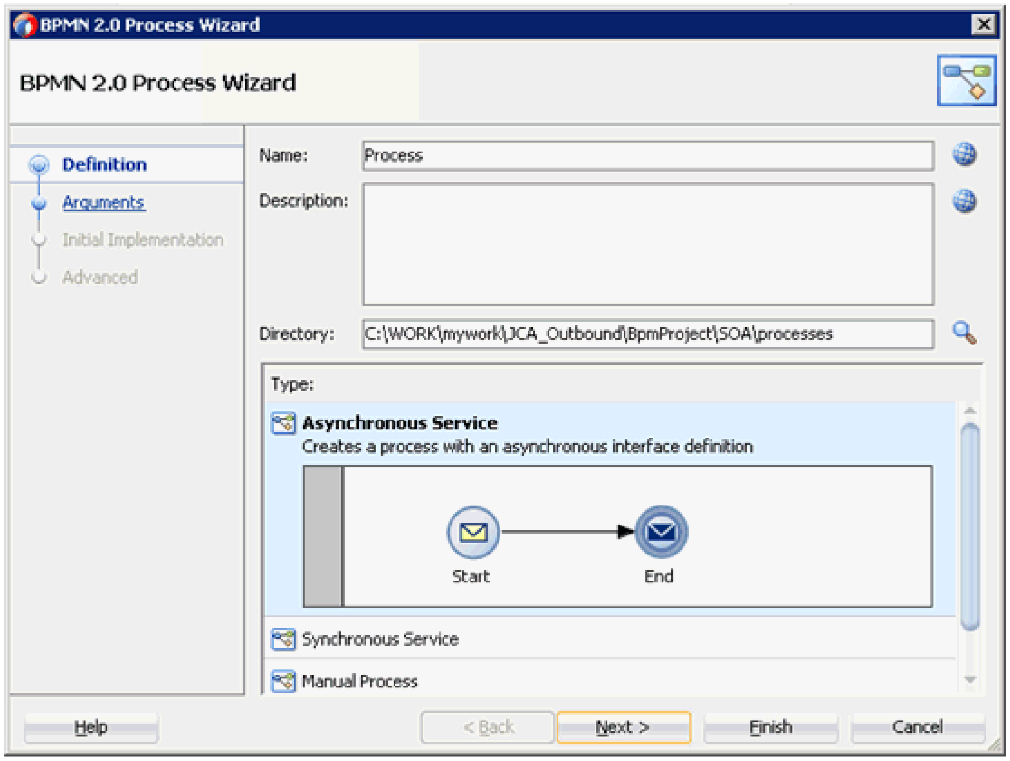
This section describes how to configure an outbound BPM process component.

Perform the following steps to configure an outbound BPM process component:

1. Drag and drop the **BPMN Process** component from the Components pane to the Components pane.

   The Create BPMN Process dialog is displayed, as shown in Figure 6–1.

Figure 6–1 Create BPMN Process Dialog



BPMN 2.0 Process Wizard

BPMN 2.0 Process Wizard

Name: Process

Description:

Directory: C:\WORK\mywork\JCA_Outbound\BpmProject\SOA\processes

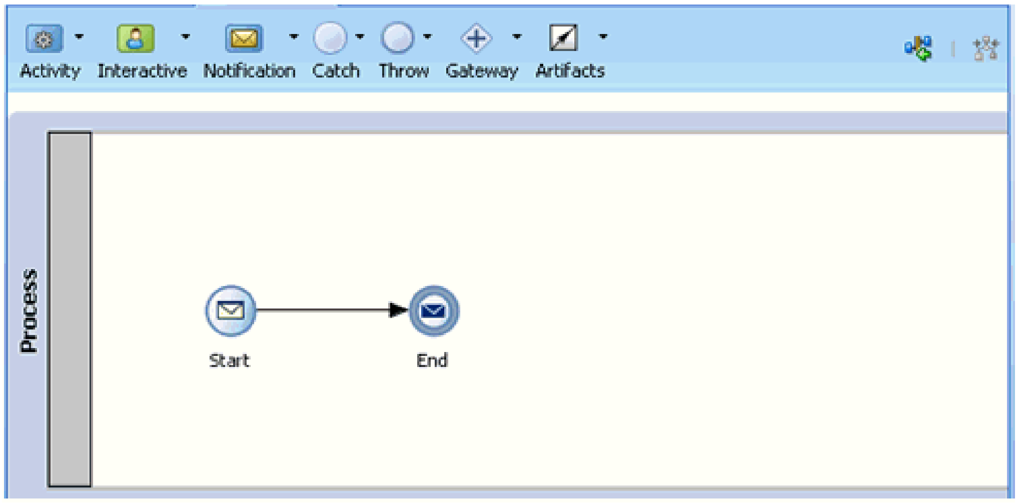
Type:

Asynchronous Service
Creates a process with an asynchronous interface definition

2. Accept the default option that is selected under the Type area (Asynchronous Service) and click **Finish**.

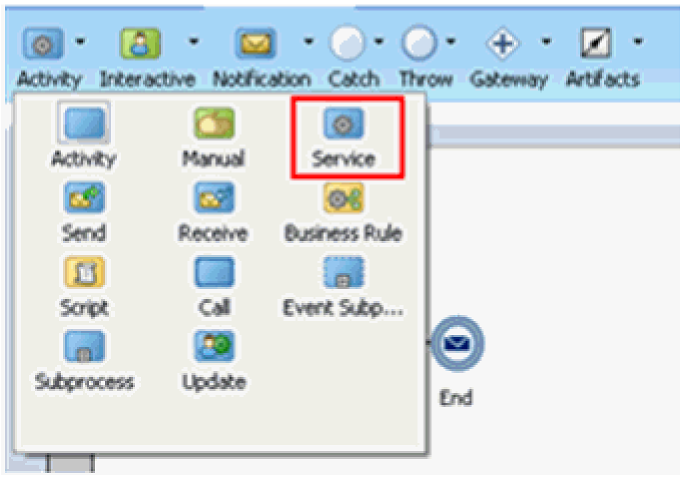
   The BPMN process is displayed, as shown in Figure 6–2.

Figure 6–2 BPMN Process



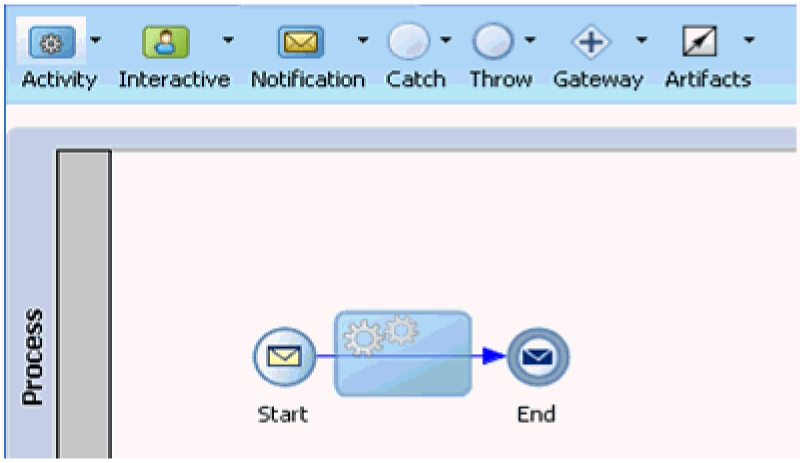
3. Click the **Activity** drop-down menu and select **Service**, as shown in Figure 6–3.

Figure 6–3 Activity Drop-down Menu



4. Drop the Service icon on the wire between the Start and End event components, as shown in Figure 6–4.

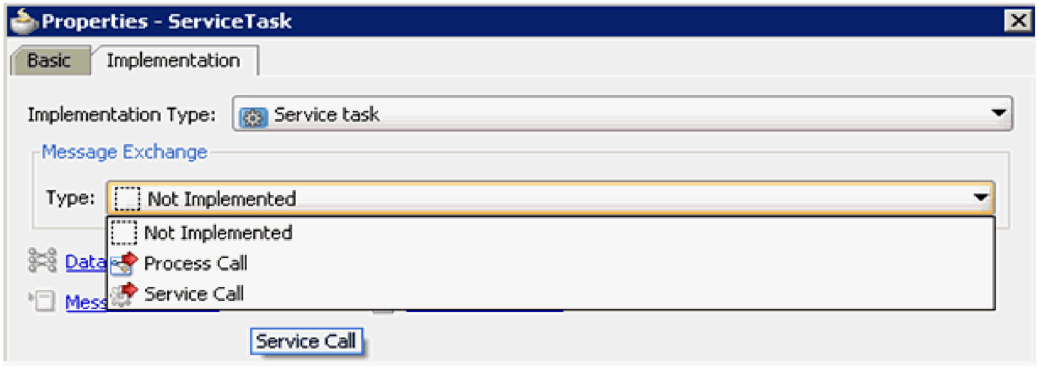
Figure 6–4 Activity Icon



The Properties - ServiceTask window is displayed.

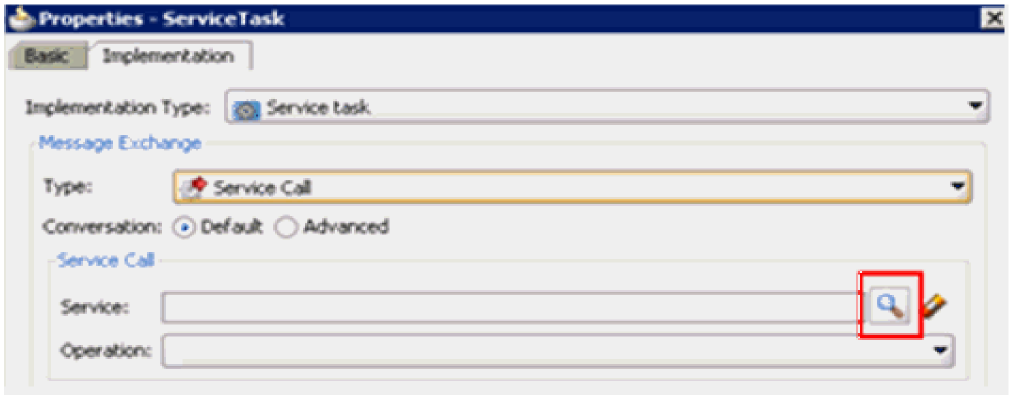
5. Click the **Implementation** tab.
6. Select **Service Call** from the Message Exchange Type list, as shown in Figure 6–5.

Figure 6–5 Service Call



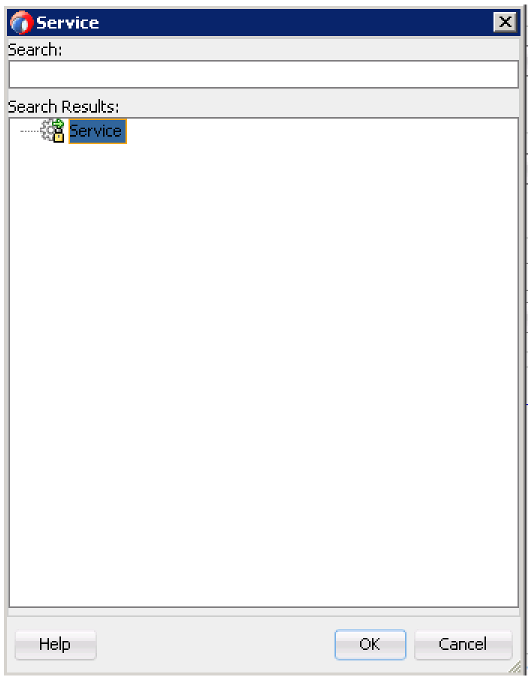
7. Click the Browse icon to the right of the Service field, as shown in Figure 6–6.

Figure 6–6 Browse Icon



The Service dialog is displayed, as shown in Figure 6–7.

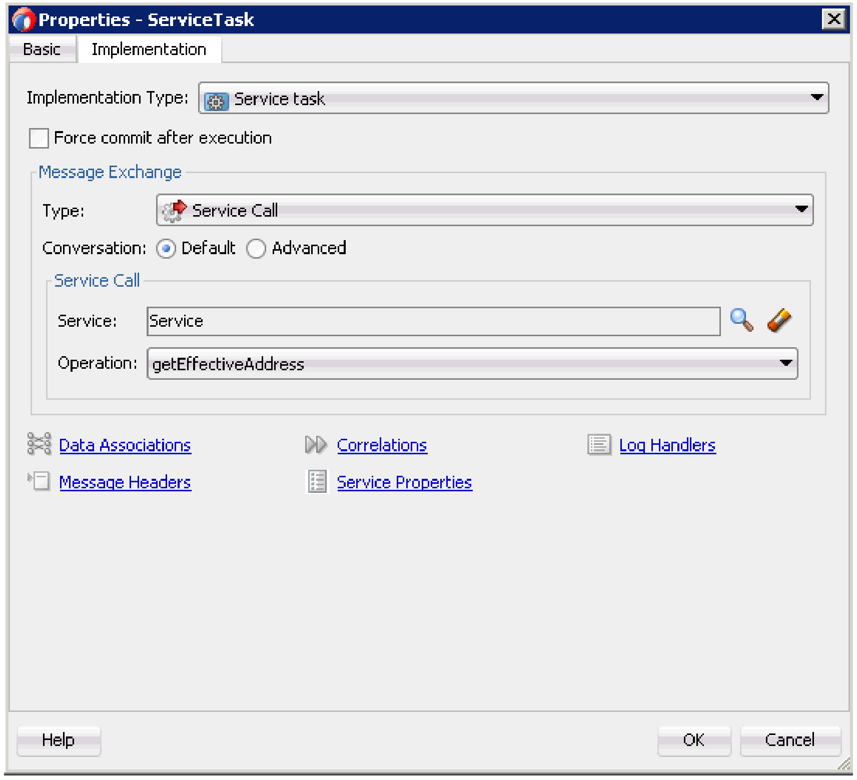
Figure 6–7 Service Dialog



8. Select the Third Party Service that has been created and click **OK**.

You are returned to the Properties - ServiceTask dialog, as shown in Figure 6–8.

Figure 6–8 Properties - ServiceTask Dialog



Implementation Type: Service task

Force commit after execution

Type: Service Call

Conversation: [✓ Default] Advanced

Service: Service

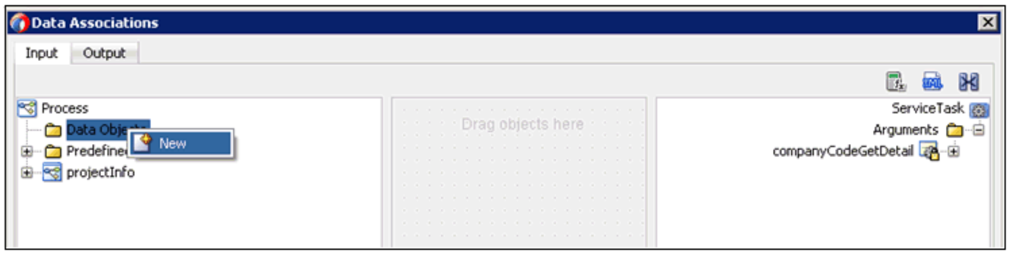
Operation: getEffectiveAddress

9. Click the **Data Associations** hyperlink.

The Data Associations dialog is displayed.

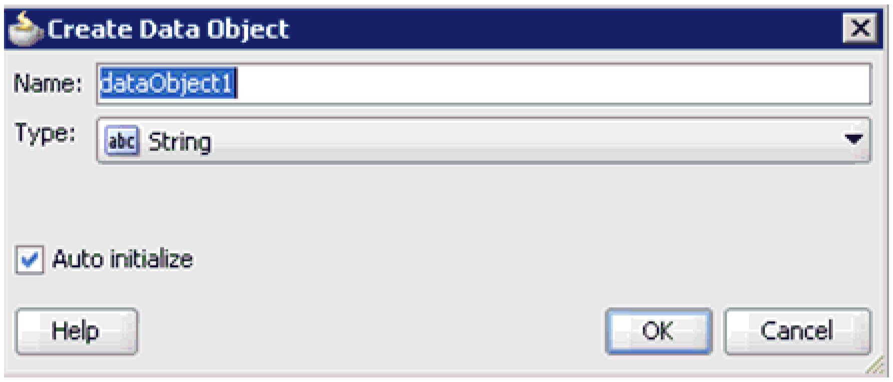
10. Right-click the **Data Objects** node in the left pane under Process, and select **New** as shown in Figure 6–9.

Figure 6–9 New Option



The Create Data Object dialog is displayed, as shown in Figure 6–10.

Figure 6–10 Create Data Object Dialog



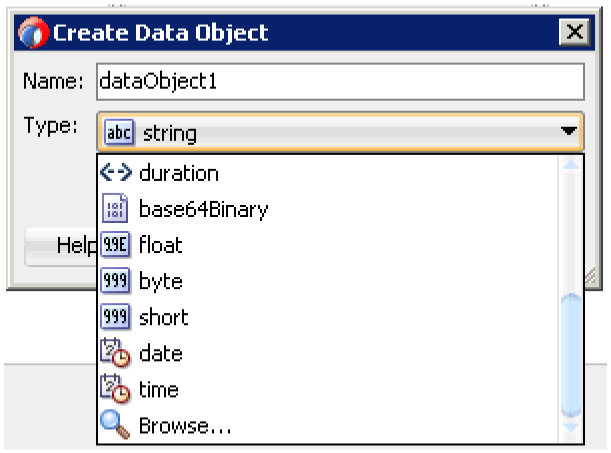
Name: dataObject1

Type: String

Auto initialize [✓]

11. Enter a name in the Name field (for example, Request), click the drop-down button in the Type field, and select **Browse** from the list, as shown in Figure 6–11.

Figure 6–11 Create Data Object Dialog

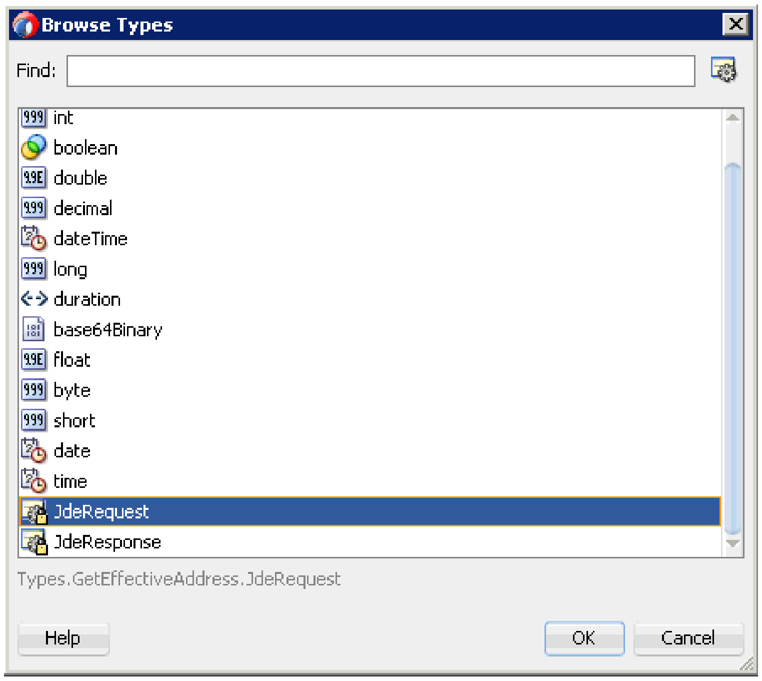


Name: dataObject1

Type: string

The Browse Types dialog is displayed, as shown in Figure 6–12.

Figure 6–12 Browse Types Dialog



12. Select the first component (for example, JdeRequest) and click **OK**.

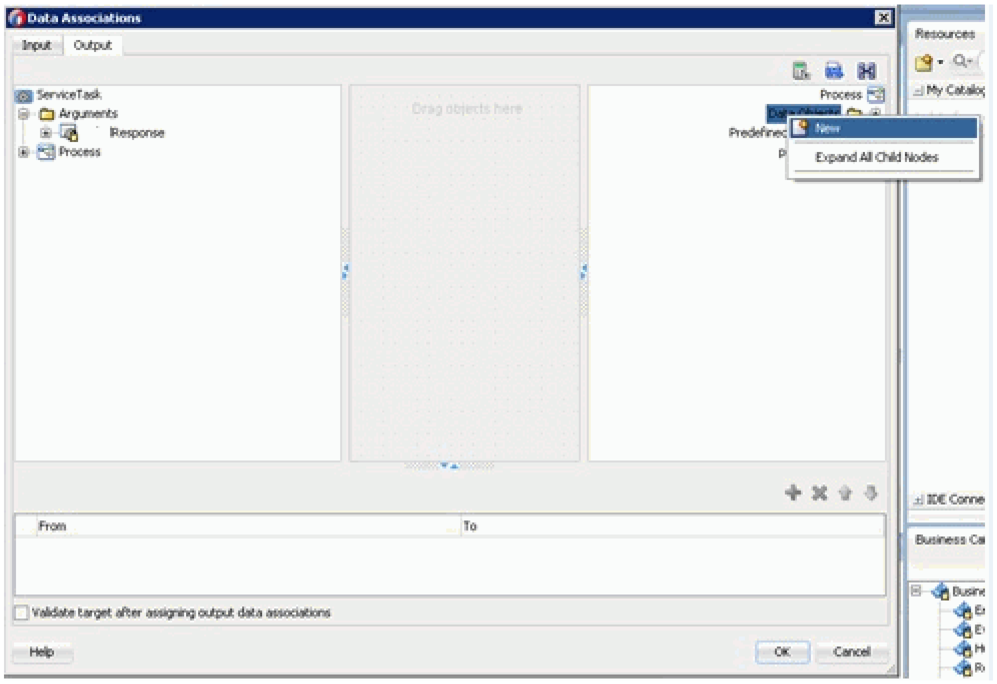
    You are returned to the Create Data Object dialog.

13. Click **OK**.

    The Data Object (for example, Request) that has been created is displayed under the Data Objects node in the Data Associations dialog.

14. Create another Data Object by right-clicking the **Data Objects** node in the right pane of the Output tab and selecting **New**, as shown in Figure 6–13.

Figure 6–13 Data Associations Dialog

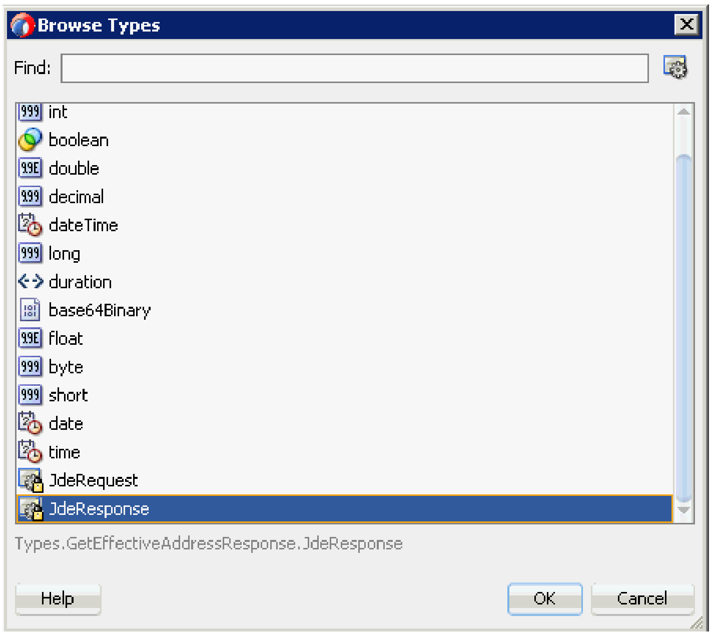


The Create Data Object dialog is displayed.

15. Enter a name in the Name field (for example, Response), and then click the drop-down button in the Type field and select **Browse** from the list.

    The Browse Types dialog is displayed, as shown in Figure 6–14.

Figure 6–14 Browse Types Dialog



16. Select the second component (for example, jdeResponse) and click **OK**.

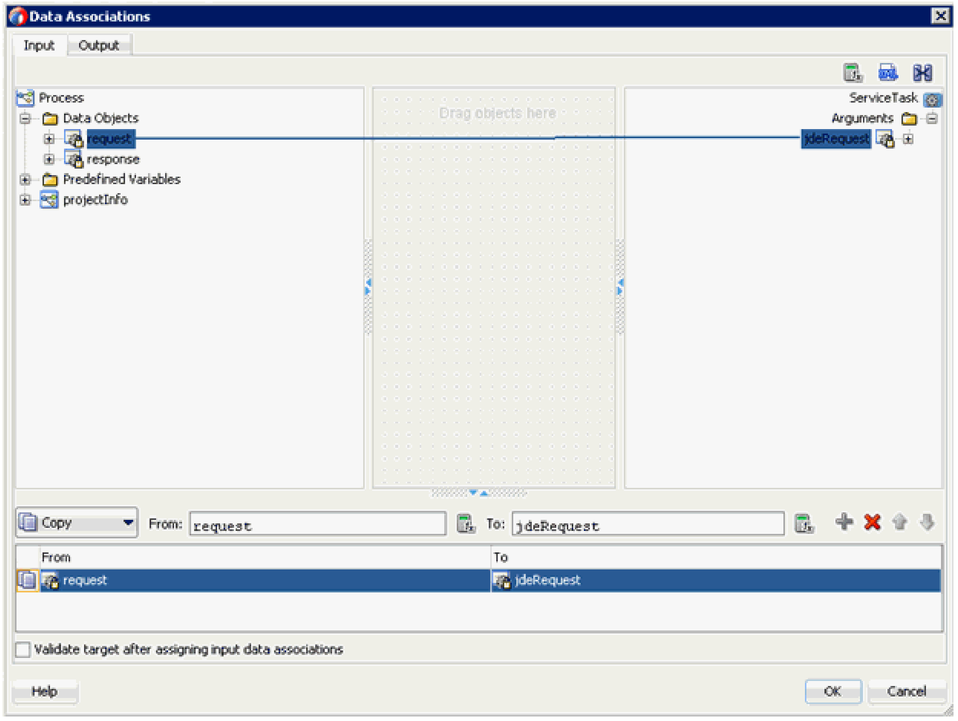
    You are returned to the Create Data Object dialog.

17. Click **OK**.

    The Data Object (for example, Response) that has been created is displayed under the Process node in the Data Associations dialog.

18. Select the **Request** Data Object under the Data Objects node in the left pane of the Input tab and drag and connect it to JdeRequest under the Arguments node in the right pane, as shown in Figure 6–15.

Figure 6–15 Request Data Object



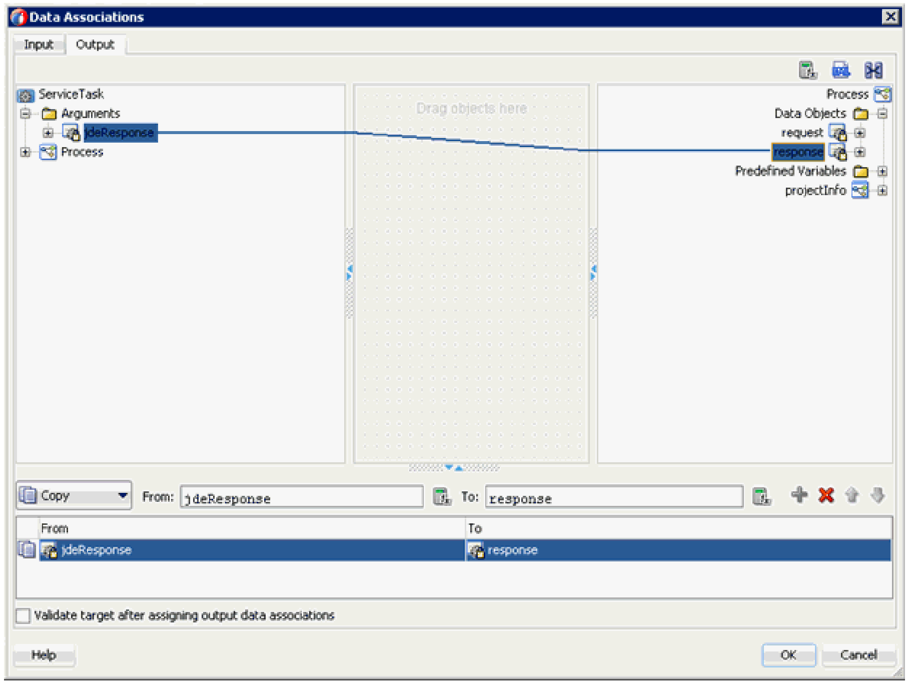
From: request
To: jdeRequest

From	To
request	jdeRequest

Validate target after assigning input data associations

19. Click on the **Output** tab and select **jdeResponse** under the Arguments node in the left pane and drag and connect it to the Response Data Object under the Data Objects node, as shown in Figure 6–16.

Figure 6–16 Response Data Object



From: jdeResponse To: response

From	To
jdeResponse	response

Validate target after assigning output data associations

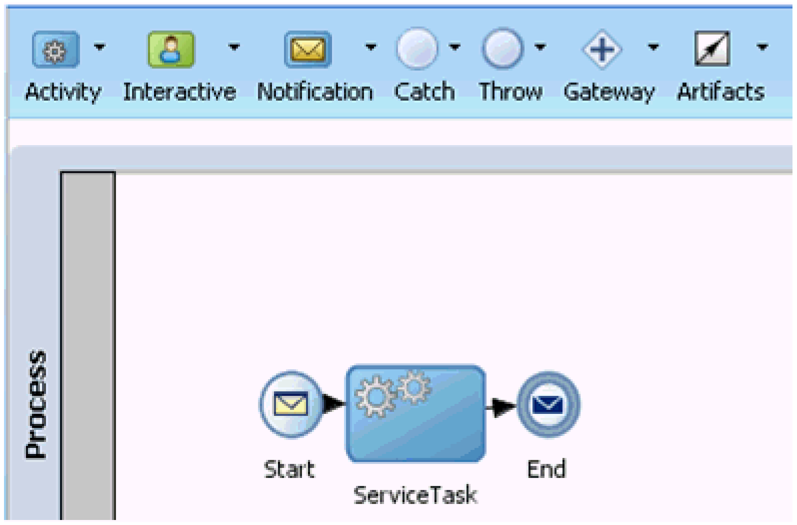
20. Click **OK**.

You are returned to the Properties - ServiceTask dialog.

21. Click **OK**.

The Service Task is created between the Start and End Event components, as shown in Figure 6–17.

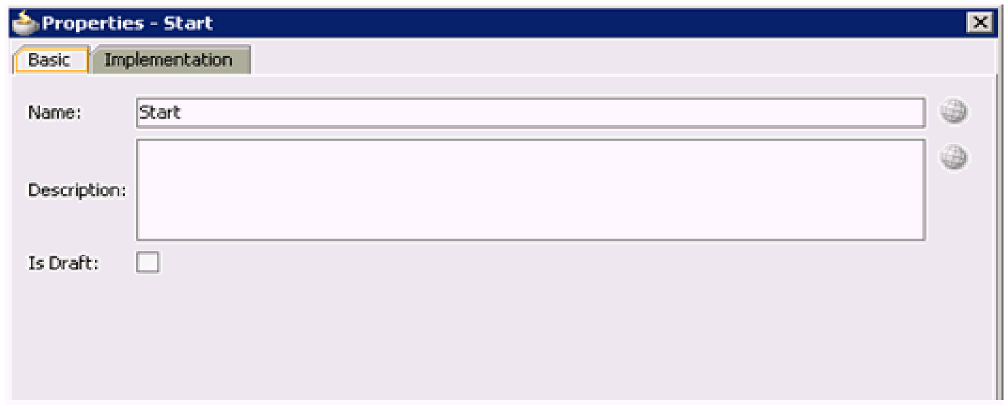
Figure 6–17 Service Task



22. Save the process and double-click the Start event component.

The Properties - Start dialog is displayed, as shown in Figure 6–18.

Figure 6–18 Properties - Start Dialog



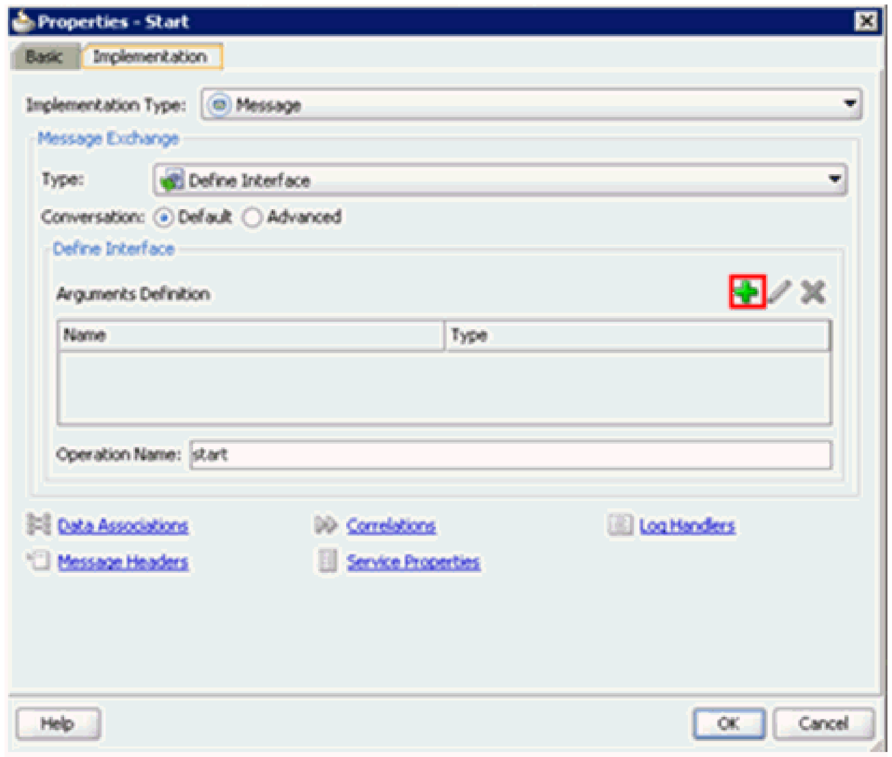
Name: Start

Description:

Is Draft:

23. Click the **Implementation** tab, as shown in Figure 6–19.

Figure 6–19 Implementation Tab



Implementation Type: Message

Type: Define Interface

Conversation: (● Default) ○ Advanced

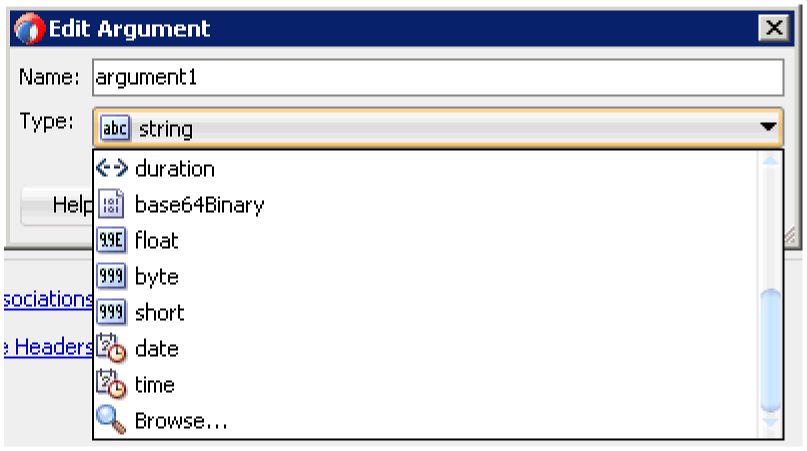
Operation Name: start

24. Click the **Plus** icon to the right of the Arguments Definition field.

    The Create Argument dialog is displayed.

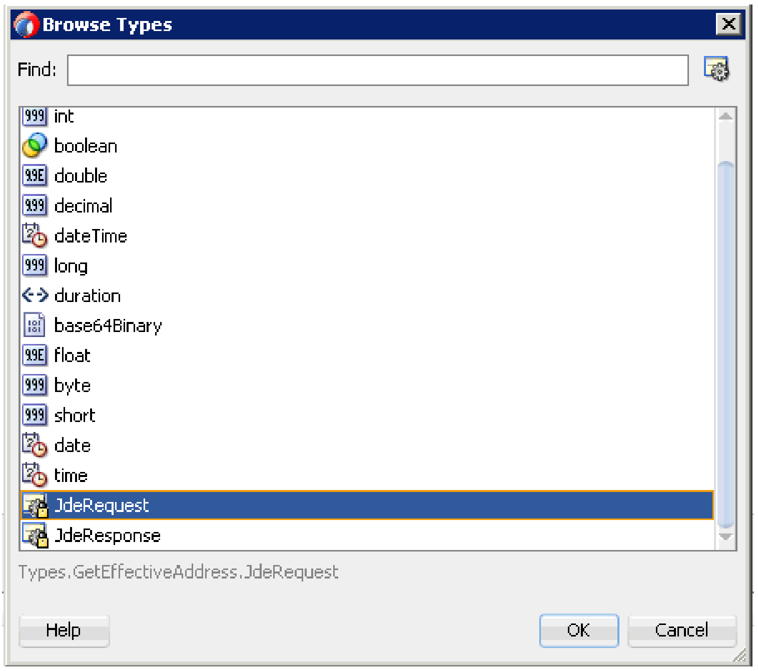
25. Enter a name in the Name field (by default, argument1), and then click the drop-down button in the Type field and select **Browse** from the list, as shown in Figure 6–20.

Figure 6–20 Create Argument Dialog



The Browse Types dialog is displayed, as shown in Figure 6–21.

Figure 6–21 Browse Types Dialog



26. Select the first component (for example, JdeRequest) and click **OK**.

You are returned to the Create Argument dialog.

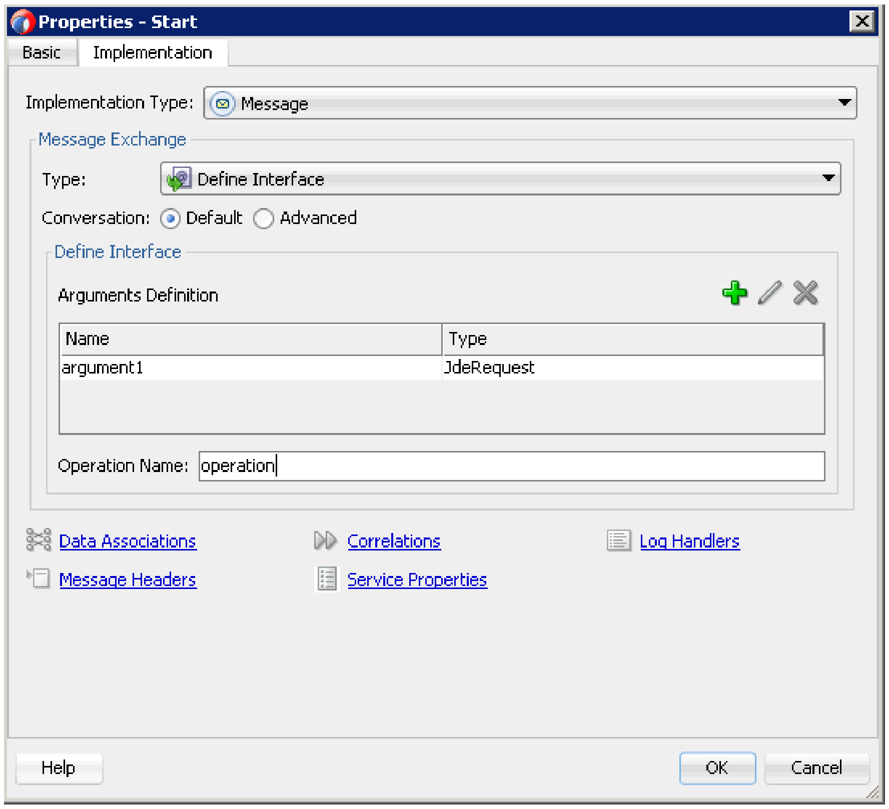
27. Click **OK**.

You are returned to the Properties - Start dialog.

28. In the Operation Name field, change **start** (default) to **operation** as shown in Figure 6–22.

Note: This change is necessary to work with old BPM payloads.

Figure 6–22 Operation Name Field


Properties - Start

Basic | Implementation

Implementation Type: Message

Message Exchange

Type: Define Interface

Conversation: [✓ Default] Advanced

Define Interface

Arguments Definition

Name	Type
argument1	JdeRequest

Operation Name: operation

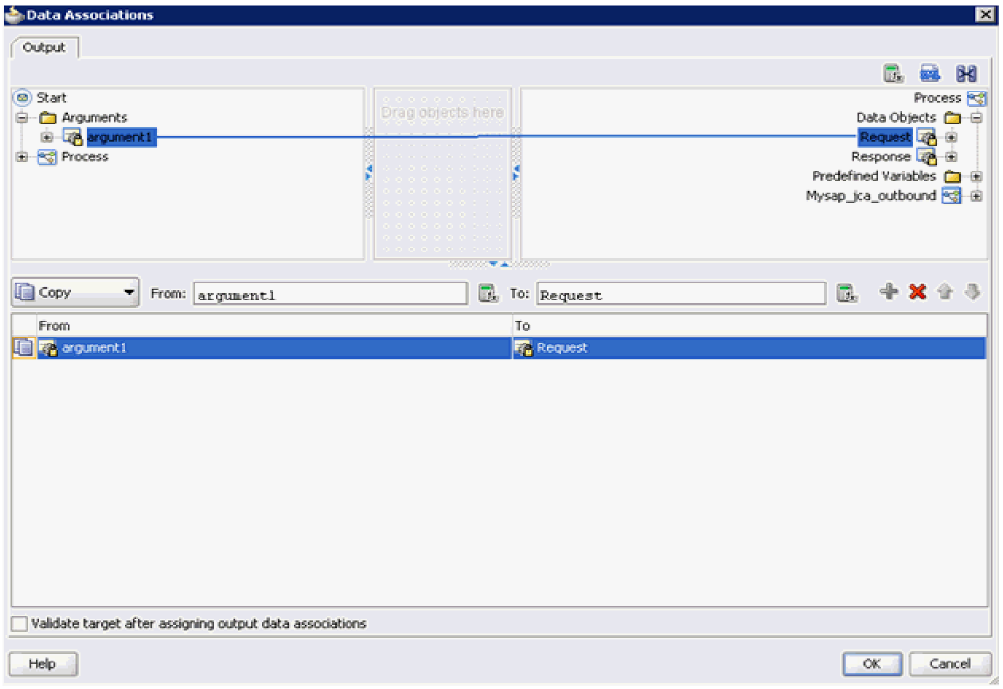
29. Click the **Data Associations** hyperlink.

    The Data Associations dialog is displayed.

30. Select **arguments1** under the Arguments node in the left pane and drag and connect it to the **Request** Data Object under Data Objects in the right pane.

31. Click **OK** as shown in Figure 6–23.

Figure 6–23 OK Button

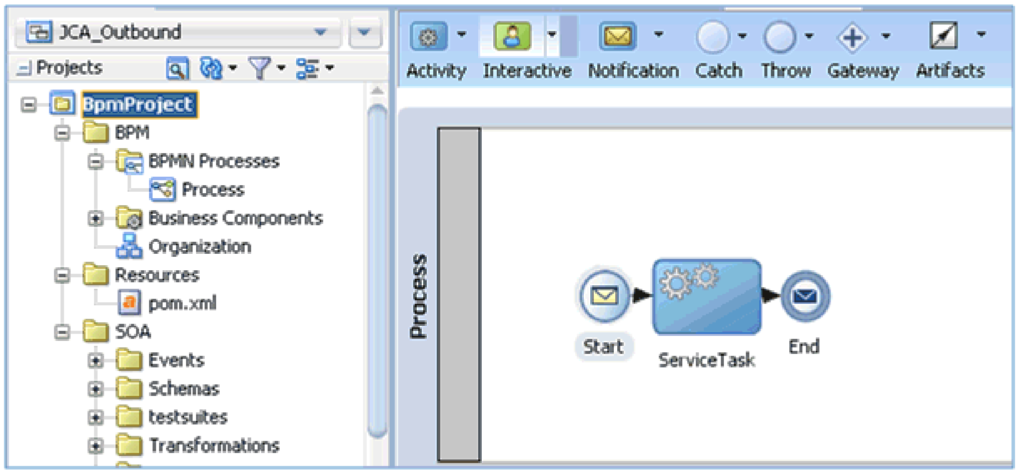


You are returned to the Properties - Start dialog.

32. Click **OK**.

You are returned to the Process workspace area, as shown in Figure 6–24.

Figure 6–24 Process Workspace Area



33. Double-click the created project to load the components.

34. Click the **Save All** icon in the menu bar to save the new outbound BPM process component that was configured.

You are now ready to create a File adapter for the write operation.

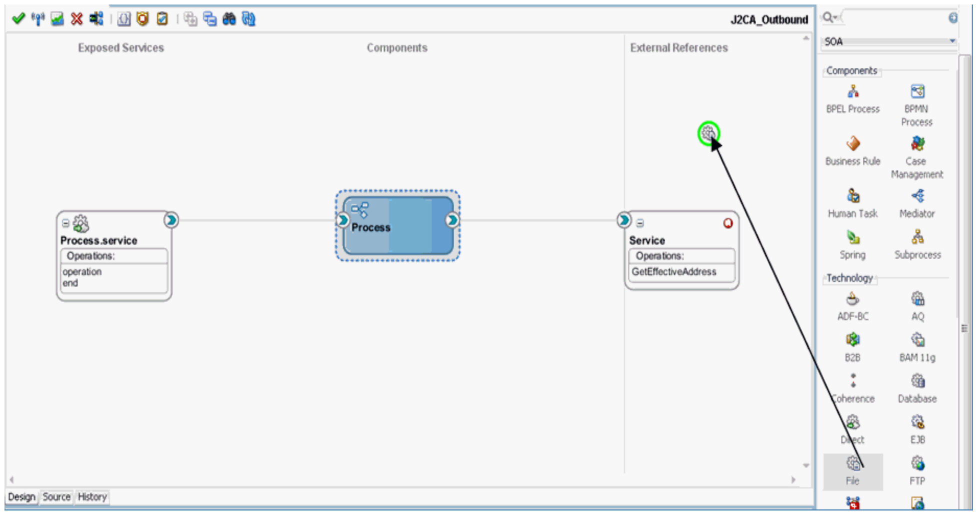
6.4.2.3 Creating a File Adapter for the Write Operation

This section describes how to create a File adapter for the write operation.

Perform the following steps to create a File adapter for the write operation:

1. Drag and drop the **File Adapter** component from the Technology Adapters pane to the External References pane, as shown in Figure 6–25.

Figure 6–25 File Adapter Component



   The Adapter Configuration Wizard is displayed.

2. Provide a Reference Name (for example, FileWrite).
3. Click **Next**.

   The Adapter Interface page is displayed.

4. Ensure that the **Define from operation and schema (specified later)** option is selected.
5. Click **Next**.

   The File Server Connection page is displayed.

6. Click **Next**.

   The Operation page is displayed.

7. Select **Write File** from the list of Operation Type options and specify an Operation Name (for example, Write).
8. Click **Next**.

   The File Configuration page is displayed.

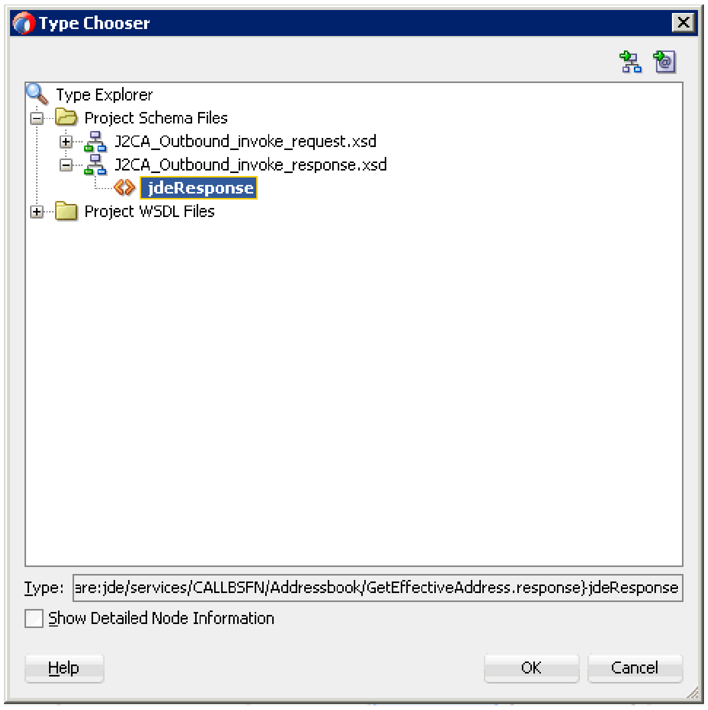
9. Specify a location on your file system where the output file is written.
10. In the File Naming Convention field, specify a name for the output file.
11. Click **Next**.

    The Messages page is displayed.

12. Click **Browse**, which is located to the right of the URL field.

    The Type Chooser dialog is displayed, as shown in Figure 6–26.

Figure 6–26 Type Chooser Dialog


Type Chooser
Type Explorer
Project Schema Files
J2CA_Outbound_invoke_request.xsd
J2CA_Outbound_invoke_response.xsd
jdeResponse
Project WSDL Files
Show Detailed Node Information
Help
OK
Cancel

Type: are:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress.response}jdeResponse

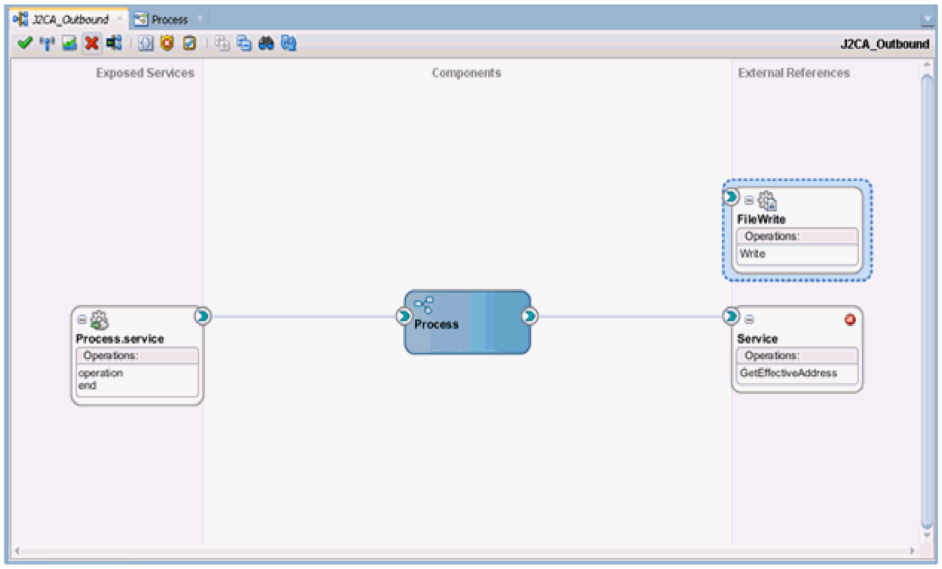
13. Expand **Project Schema Files** and **J2CA_Outbound_invoke_response.xsd**.
14. Select the available schema (for example, jdeResponse).
15. Click **OK**.

    You are returned to the Messages page.
16. Click **Next**.

    The Finish page is displayed.
17. Click **Finish**.

    The File Adapter service is created in the External References pane, as shown in Figure 6–27.

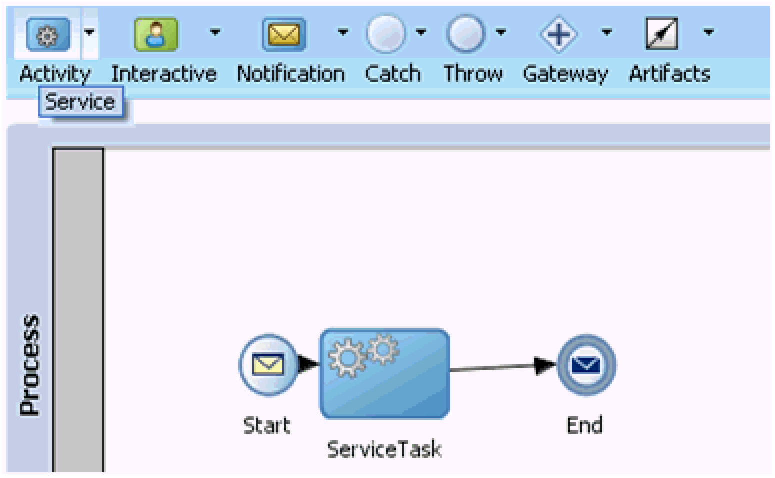
Figure 6–27 File Adapter Service



18. Double-click the BPMN Process component.

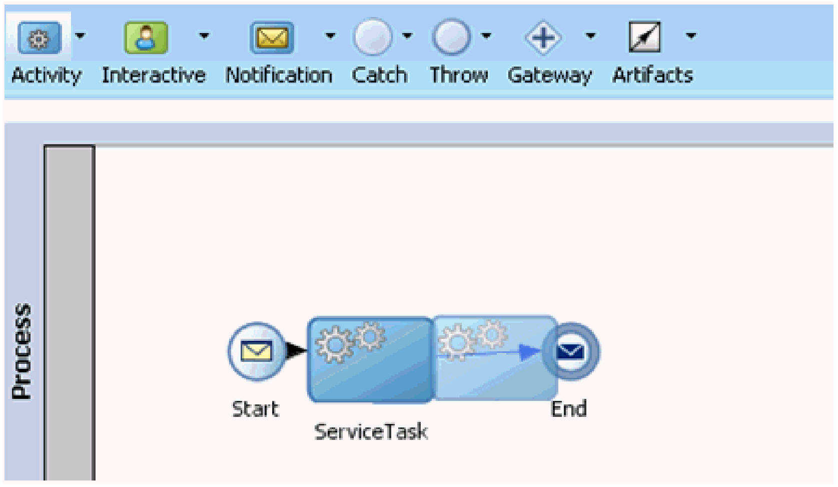
    The BPMN process is displayed, as shown in Figure 6–28.

Figure 6–28 BPMN Process



19. Click the **Activity** icon, and select **Service**.
20. Drop the Service icon on the wire between the Service Task and End event components, as shown in Figure 6–29.

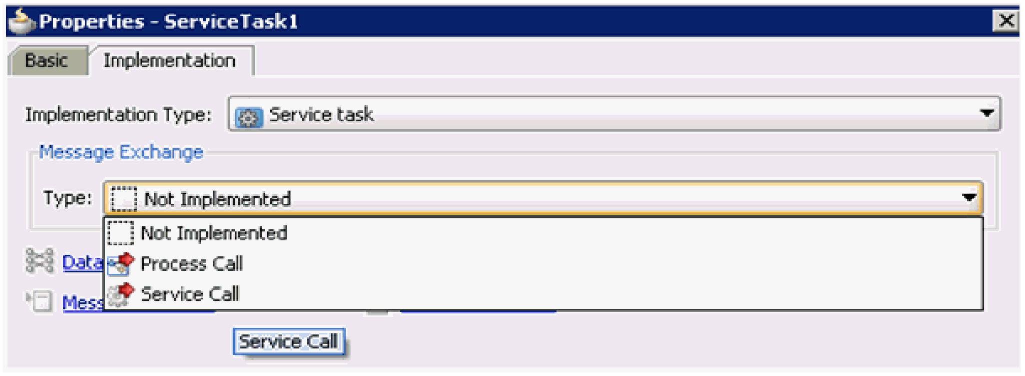
Figure 6–29 Activity Icon



The Properties - ServiceTask1 dialog is displayed.

21. Click the **Implementation** tab.
22. Select **Service Call** from the Type drop-down list in the Message Exchange section, as shown in Figure 6–30.

Figure 6–30 Service Call



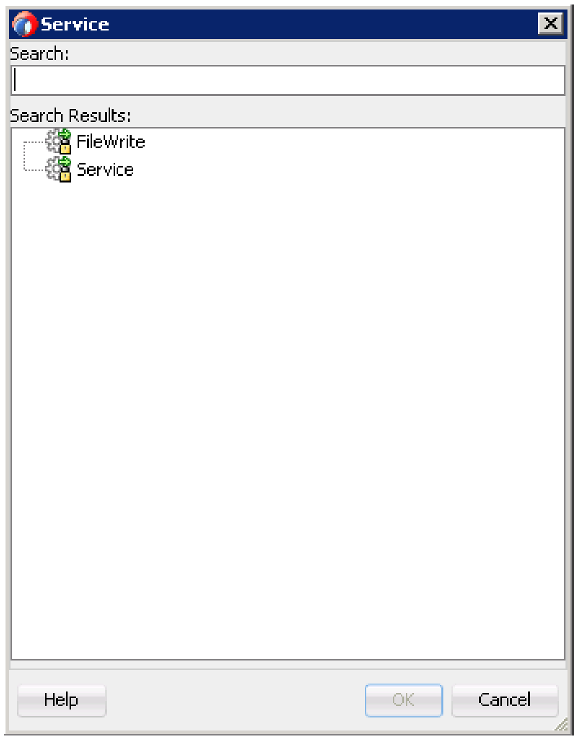
Implementation Type: Service task

Type: Not Implemented

23. Click the **Browse** icon to the right of the Service field.

The Service dialog is displayed, as shown in Figure 6–31.

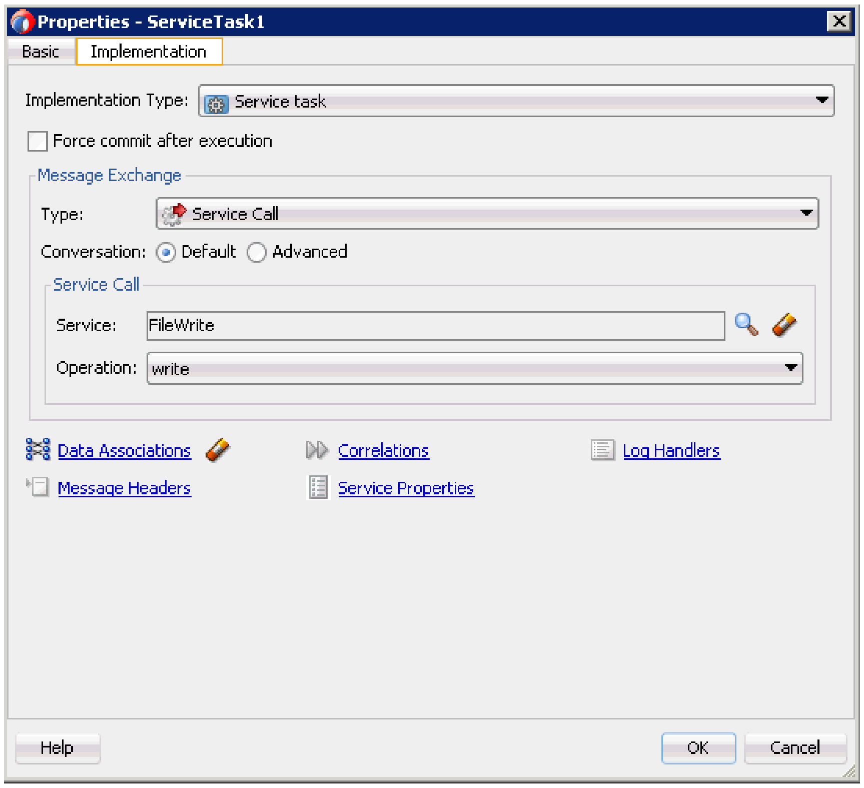
Figure 6–31 Service Dialog



24. Select the service for write operation that has been created (for example, FileWrite) and click **OK**.

    You are returned to the Properties - ServiceTask1 dialog, as shown in Figure 6–32.

Figure 6–32 Properties - ServiceTask1 Dialog


Properties - ServiceTask1

Basic | Implementation

Implementation Type: Service task

☐ Force commit after execution

Message Exchange

Type: Service Call

Conversation: [✓ Default] Advanced

Service Call

Service: FileWrite

Operation: write

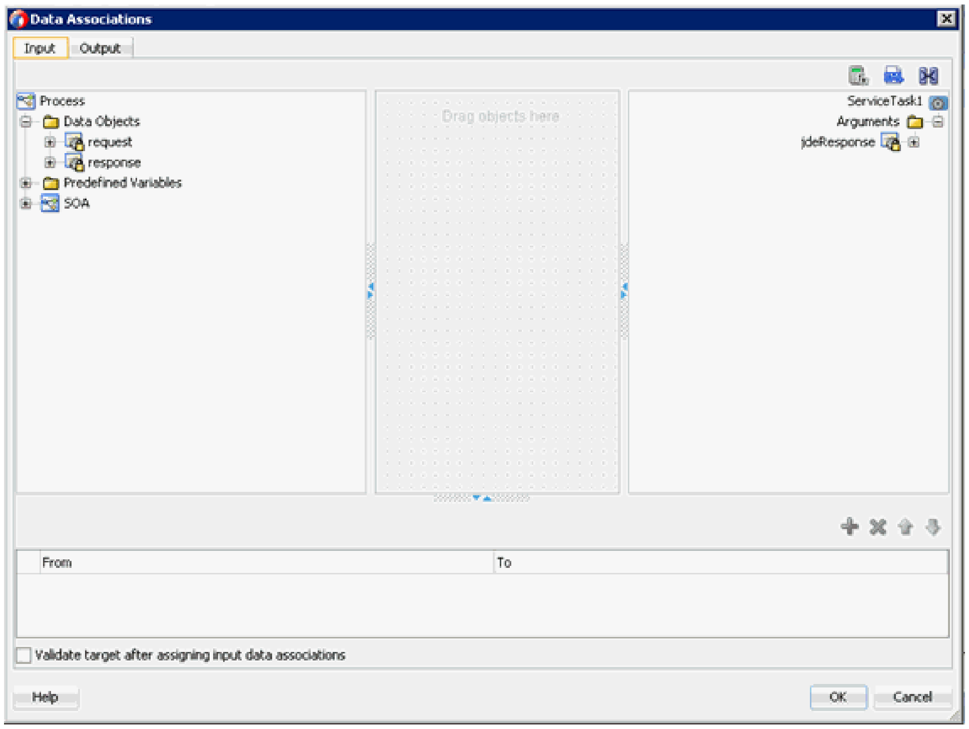
Data Associations
Correlations
Log Handlers
Message Headers
Service Properties

Help | OK | Cancel

25. Click the **Data Associations** hyperlink.

The Data Associations dialog is displayed, as shown in Figure 6–33.

Figure 6–33 Data Associations Dialog


Data Associations

Input | Output

Process
Data Objects
request
response
Predefined Variables
SOA

Drag objects here

ServiceTask1
Arguments
jdeResponse

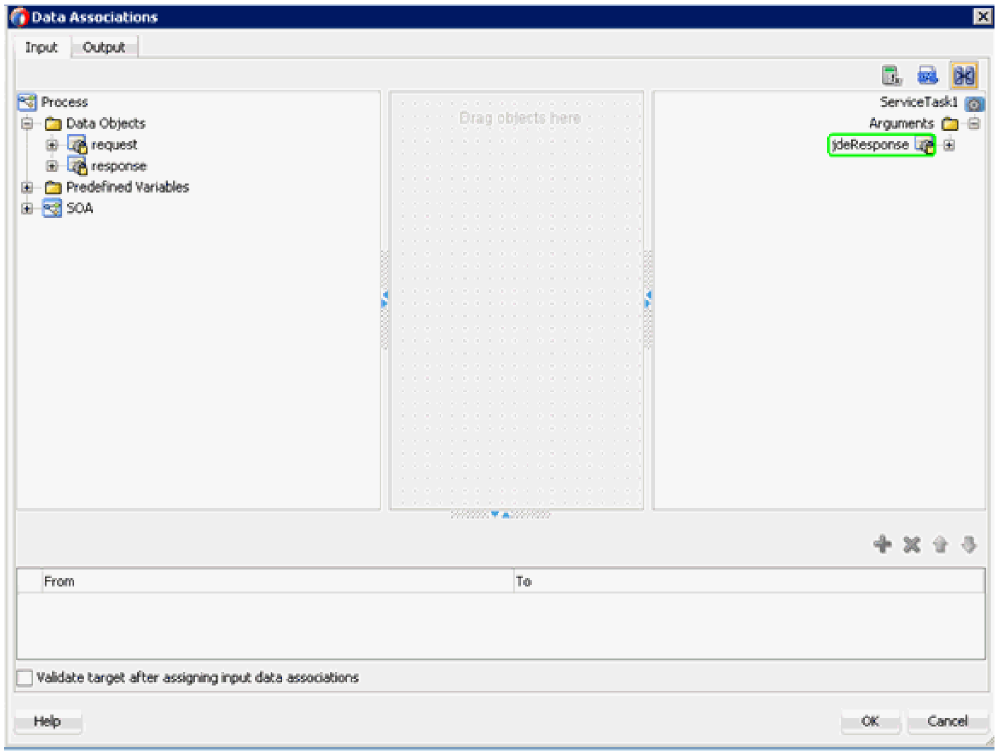
From	To

☐ Validate target after assigning input data associations

Help | OK | Cancel

26. In the Input tab, click the XSL Transformation icon in the top right pane.
27. Drag and drop the XSL Transformation icon to the **jdeResponse** node, as shown in Figure 6–34.

Figure 6–34 CompanyCodeJDEResponse Node

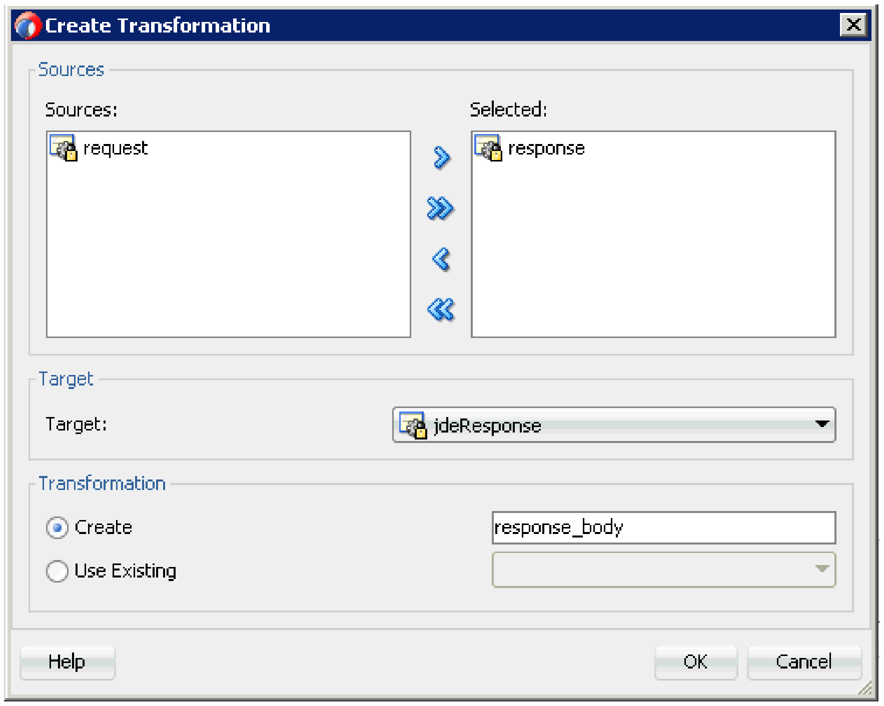


The Create Transformation dialog is displayed.

28. Select **Response** in the Sources section and click the right arrow symbol.

    The Response object is added to the Selected elements area as shown in Figure 6–35.

Figure 6–35 Response Object


Create Transformation

Sources

Sources: request

Selected: response

Target

Target: jdeResponse

Transformation

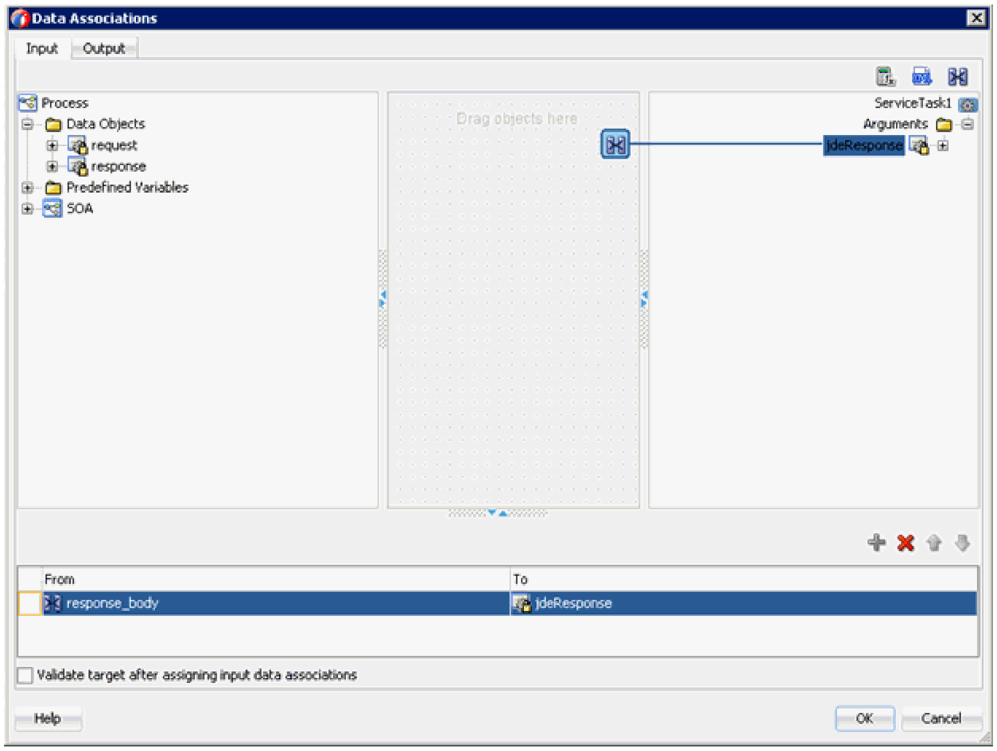
[✓ Create] response_body

Use Existing

29. Accept the default value selected in the Target drop-down list and the default name in the Create field by clicking **OK**.

    You are returned to the Data Associations dialog window with the XSL transformation created, as shown in Figure 6–36.

Figure 6–36 Data Associations Dialog


Data Associations

From	To
response_body	jdeResponse

Validate target after assigning input data associations

30. Click **OK**.

    You are returned to the Properties - ServiceTask1 dialog.

31. Click **OK**.

    The Response_body.xsl tab is displayed.

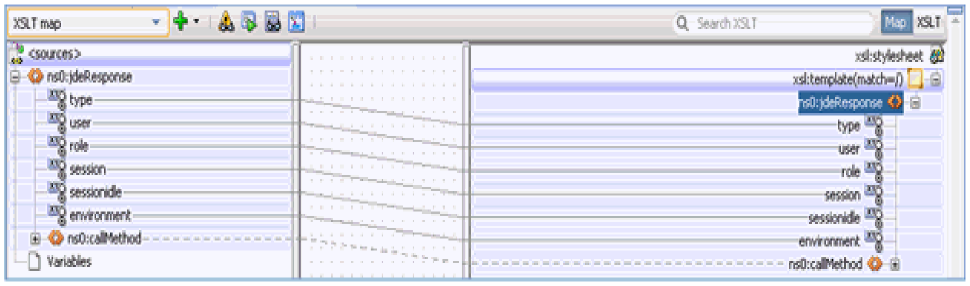
32. Automap the Source and Target elements.

    The Auto Map Preferences dialog is displayed.

33. Accept the default values and click **OK**.

    The transformation is completed, as shown in Figure 6–37.

Figure 6–37 Completed Transformation



34. Save the transformation.

35. Return to the Process workspace area.

    The ServiceTask1 component is created between the ServiceTask component and the End event component.

36. Click the **Save All** icon in the menu bar to save the new outbound BPM process component that was configured.

    You are now ready to deploy the outbound BPM process.

6.4.3 Adjusting for Known Deployment Issues With 12c

For more information on how to adjust for known deployment issues with 12c, see Section 4.4.3.3, "Adjusting for Known Deployment Issues With 12c" on page 4-26.

6.4.4 Deploying the BPM Outbound Process

Perform the following steps to deploy the Mediator outbound process.

1. Right-click the project name in the left pane, select **Deploy**, and then click **J2CA_Outbound**.

   The Deployment Action page is displayed.

2. Ensure that **Deploy to Application Server** is selected.

3. Click **Next**.

   The Deploy Configuration page is displayed.

4. Leave the default values selected and click **Next**.

   The Select Server page is displayed.

5. Select an available application server that was configured and click **Next**.

The SOA Servers page is displayed.

6. Select a target SOA server and click **Next**.

   The Summary page is displayed.

7. Review and verify all the available deployment information for your project and click **Finish**.

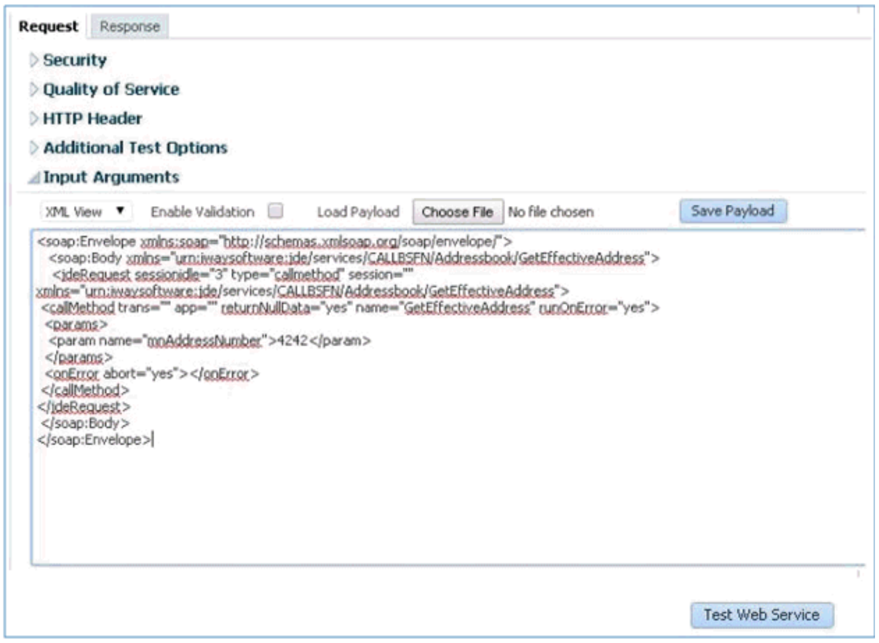
   For more information, see Section 4.4.4, "Deploying the BPEL Outbound Process" on page 4-28.

6.4.5 Invoking the Input XML Document in the Oracle Enterprise Manager Console

Perform the following steps to invoke the input XML document in the Oracle Enterprise Manager console.

1. Logon to the Oracle Enterprise Manager console.
2. Expand your domain in the left pane followed by the **SOA** folder.
3. Select an available project (for example, J2CA_Outbound).
4. Click **Test**.
5. Click the **Request** tab.

Figure 6–38 Request Tab

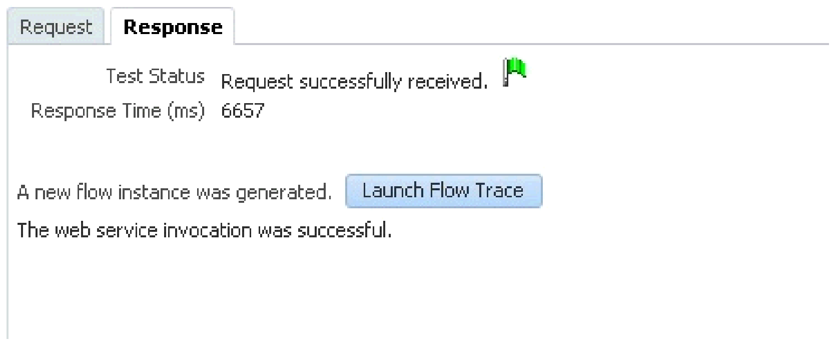


```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body xmlns="urn:iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress">
   <jdeRequest sessionidle="3" type="callmethod" session=""
xmlns="urn:iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress">
 <callMethod trans="" app="" returnNullData="yes" name="GetEffectiveAddress" runOnError="yes">
  <params>
   <param name="mnAddressNumber">4242</param>
  </params>
  <onError abort="yes"></onError>
 </callMethod>
</jdeRequest>
 </soap:Body>
</soap:Envelope>
```

6. Provide an appropriate input value in the Value field and click **Test Web Service**, as shown in Figure 6–38.

   A response is received in the Response tab to indicate that invocation was successful in the Oracle Enterprise Manager console, as shown in Figure 6–39.

Figure 6–39 Received Response



Test Status Request successfully received.
Response Time (ms) 6657

A new flow instance was generated.

The web service invocation was successful.

7. Navigate to the defined output directory on your file system and open the XML response document that was received.

   The XML response document contains the generated output with values.

6.5 Designing an Inbound BPM Process Using Transformations for Event Integration (J2CA Configuration)

This section demonstrates how Oracle Application Adapter for J.D. Edwards OneWorld integrates with J.D. Edwards OneWorld to receive event data.

A sample project has been provided for this inbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\BPM\J2CA\Inbound_Project
```

The following tools are required to complete your adapter design-time configuration:

- Oracle Adapter Application Explorer (Application Explorer)
- Oracle JDeveloper BPM Designer (JDeveloper)

Note: The examples in this chapter demonstrate the use of JDeveloper.

This section contains the following topics:

- Section 6.5.1, "Creating an Empty Composite for BPM"
- Section 6.5.2, "Defining a BPM Inbound Process"

Before you design a BPM process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.

6.5.1 Creating an Empty Composite for BPM

For more information on how to configure a new Application Server connection in Oracle JDeveloper, see Section 4.3, "Configuring a New Application Server Connection" on page 4-2.

6.5.2 Defining a BPM Inbound Process

This section describes how to define a BPM inbound process, which contains the following topics:

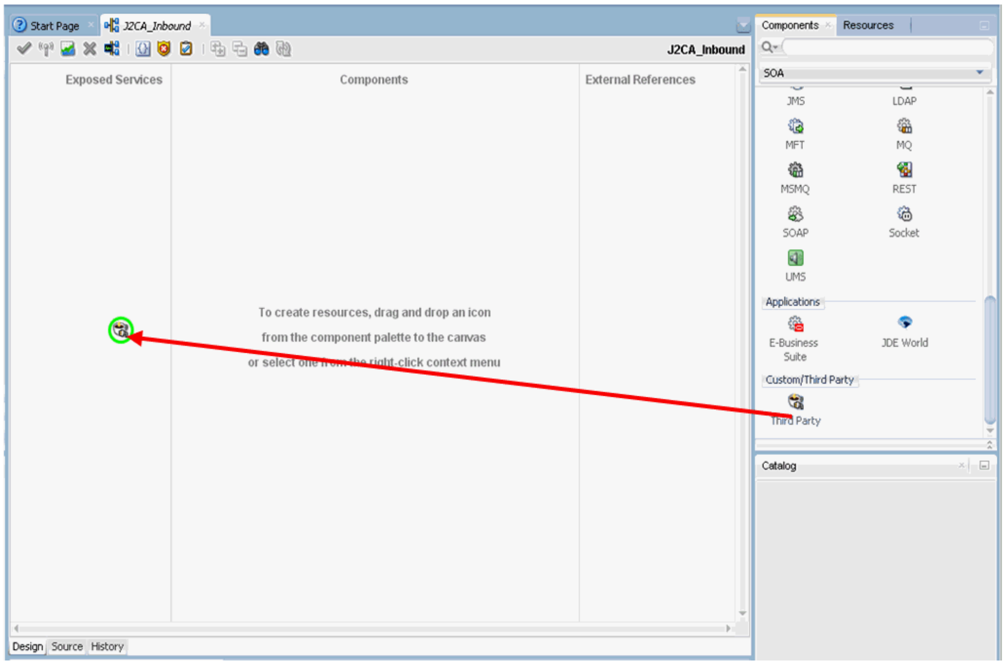
- Section 6.5.2.1, "Configuring a Third Party Adapter Service Component"
- Section 6.5.2.2, "Configuring an Inbound BPM Process Component"
- Section 6.5.2.3, "Creating a File Adapter for the Write Operation"
- Section 6.5.2.4, "Adjusting for Known Deployment Issues With 12c"

6.5.2.1 Configuring a Third Party Adapter Service Component

Perform the following steps to create a third party adapter service component:

1. Double-click the created project to load the components.
2. Drag and drop the **Third Party Adapter** component from the Custom/Thirdparty pane to the Exposed References pane, as shown in Figure 6–40.

Figure 6–40 Third Party Adapter Component



   The Create Third Party Adapter Service dialog is displayed.

3. Enter a name for the third party adapter service.
4. Ensure that **Service** is selected from the Type list (default).
5. Click the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.

   The WSDL Chooser dialog is displayed.

6. Select **File System**, and then browse and select an inbound WSDL file from the following directory:

   ```
   <ADAPTER_HOME>\wsdls
   ```

7. Click **OK**.

The Localize Files dialog is displayed.

8. Click **OK**.

   The inbound WSDL file and associated receive_request XML schema file (.xsd) are imported to the project folder that has been created.

   You are returned to the Create Third Party Adapter Service dialog.

9. Click the **Find JCA file** icon, which is located to the right of the JCA File field.

   The Transformation Chooser dialog is displayed.

10. Select **File System**, and then browse and select the JCA properties file from the following directory:

    ```
    <ADAPTER_HOME>\wsdls
    ```

11. Click **OK**.

    The Copy File message is displayed.

12. Click **Yes**.

    A copy of the JCA properties file is made in the project folder.

    You are returned to the Create Third Party Adapter Service dialog.

13. Click **OK**.

    The third party adapter service component (matmas) is created in the Exposed References pane.

    You are now ready to configure an inbound BPM process component.

    For more information, see Section 4.4.3.1, "Configuring a Third Party Adapter Service Component" on page 4-11.

6.5.2.2 Configuring an Inbound BPM Process Component

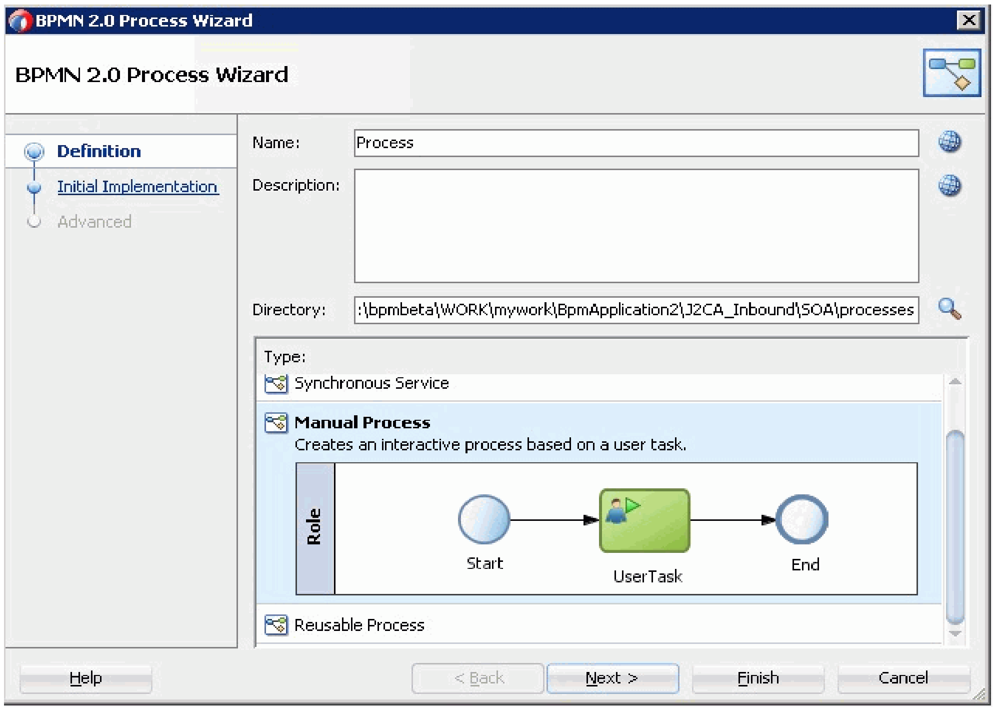
This section describes how to configure an inbound BPM process component.

Perform the following steps to configure an inbound BPM process component:

1. Drag and drop the **BPMN Process** component from the Components pane to the Components pane.

   The Create BPMN Process dialog is displayed, as shown in Figure 6–41.

Figure 6–41 Create BPMN Process Dialog



Name: Process

Description:

Directory: :\bpmbeta\WORK\mywork\BpmApplication2\J2CA_Inbound\SOA\processes

Type:

Synchronous Service

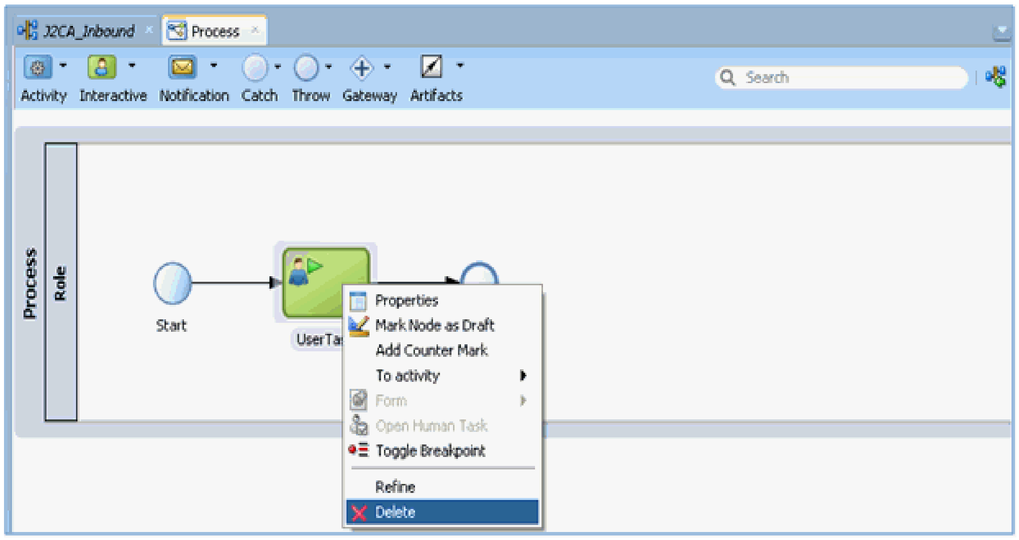
Manual Process
Creates an interactive process based on a user task.

Reusable Process

2. Select **Manual Process** in the Type section.
3. Click **Finish**.

   The BPMN process is displayed, as shown in Figure 6–42.

Figure 6–42 BPMN Process

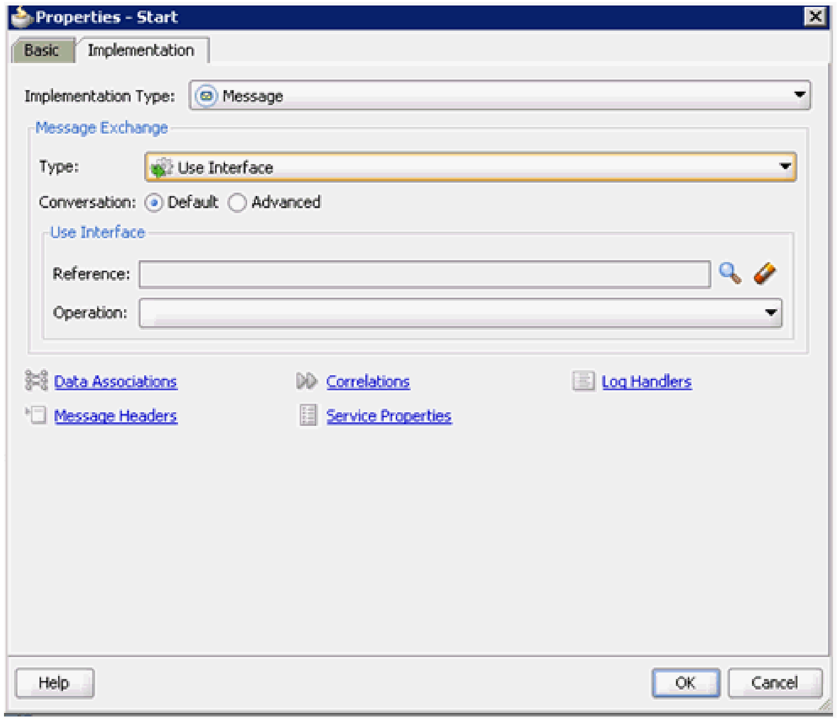


4. Right-click **UserTask** and select **Delete** from the menu.
5. Double-click the **Start** event component.

   The Properties - Start dialog is displayed.
6. Click the **Implementation** tab.

7. Select **Message** from the Implementation Type list.
8. Select **Use Interface** from the Message Exchange Type drop-down list.
9. Click the **Browse** icon to the right of the Reference field, as shown in Figure 6–43.

Figure 6–43 Browse Icon



Implementation Type: Message

Message Exchange

Type: Use Interface

Conversation: (●) Default ( ) Advanced

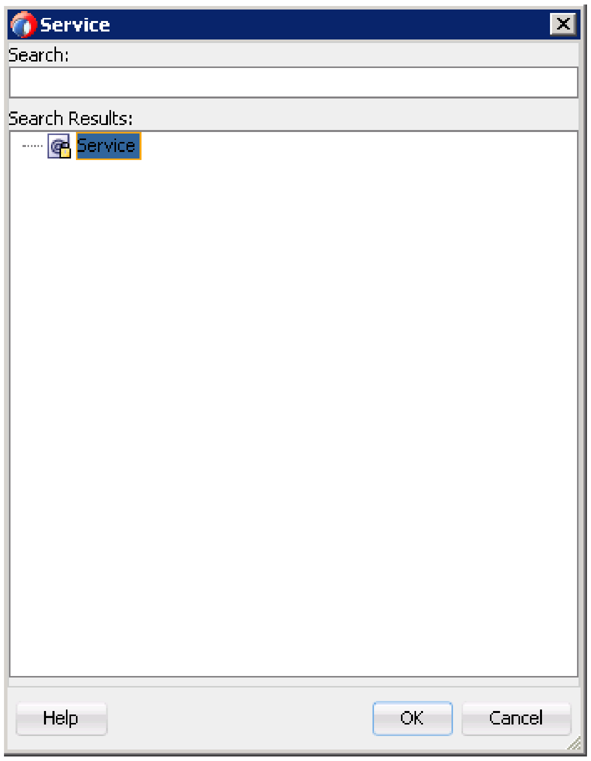
Use Interface

Reference:

Operation:

The Service dialog is displayed, as shown in Figure 6–44.

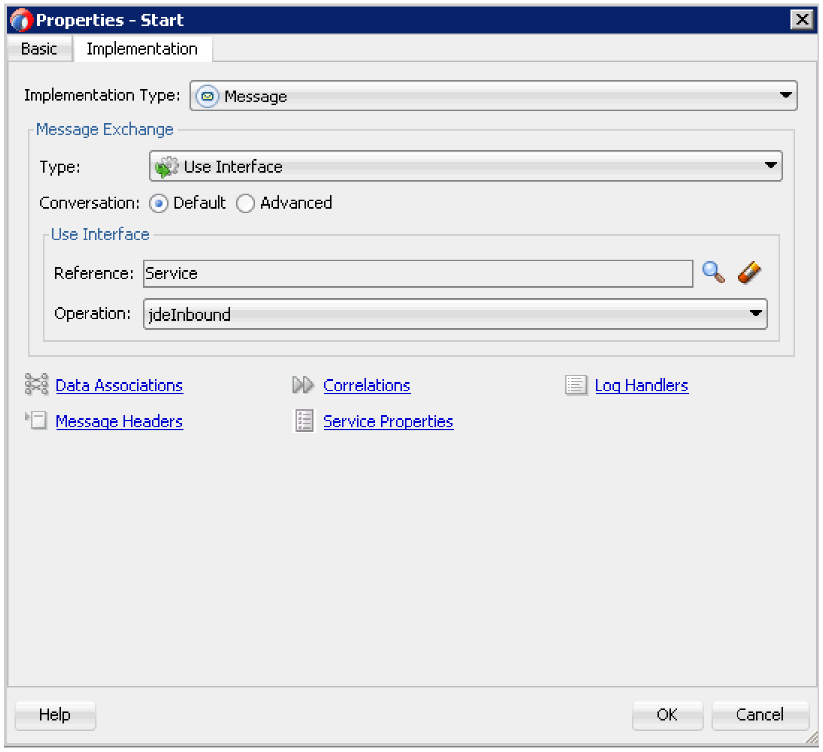
Figure 6–44 Service Dialog



10. Select the Third Party Service that has been created and click **OK**.

    You are returned to the Properties - Start dialog, as shown in Figure 6–45.

Figure 6–45 Properties - Start Dialog



Implementation Type: Message

Type: Use Interface

Conversation: [✓ Default] Advanced

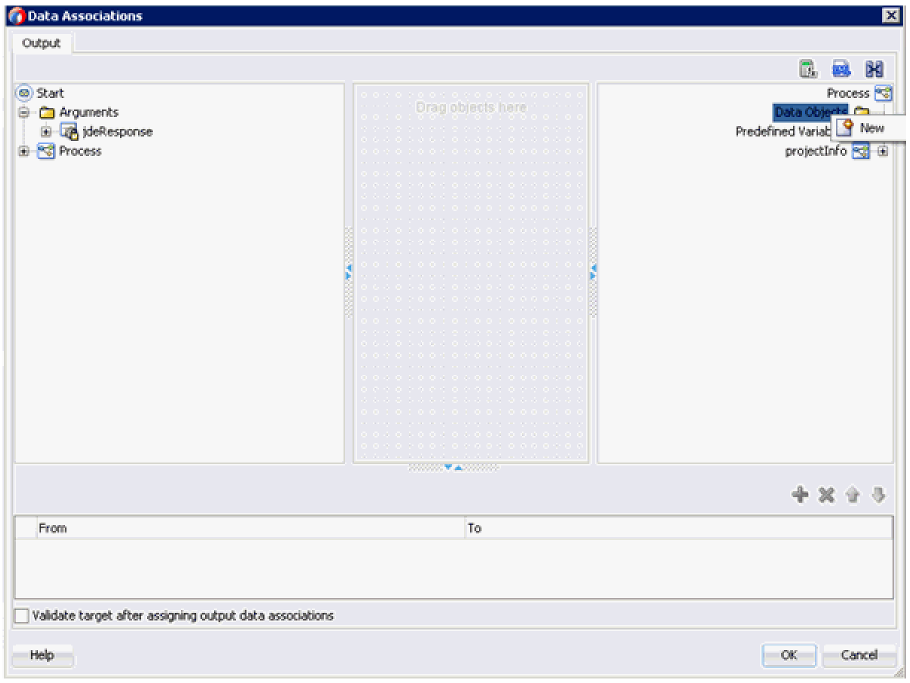
Reference: Service

Operation: jdeInbound

11. Click the **Data Associations** icon.

    The Data Associations dialog is displayed, as shown in Figure 6–46.

Figure 6–46 Data Associations Dialog

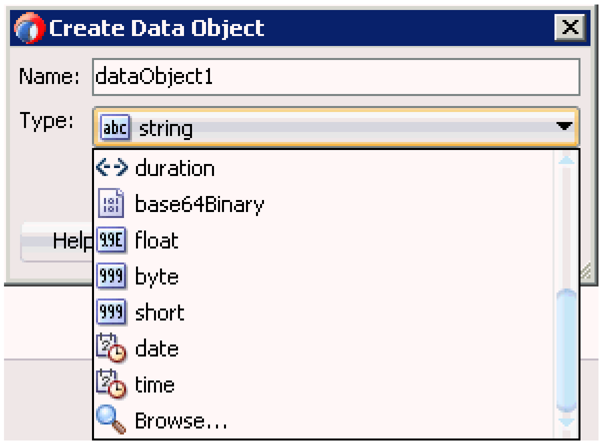


12. Right-click the **Data Object** node in the right pane and select **New**.

    The Create Data Object dialog is displayed.

13. Enter a name in the Name field, and then click the drop-down button in the Type field and select **Browse** from the list, as shown in Figure 6–47.

Figure 6–47 Create Data Object Dialog



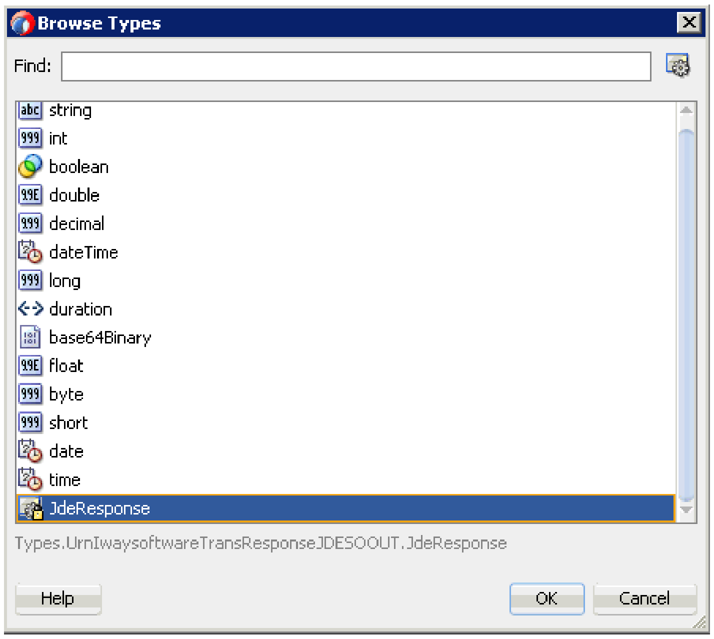
Create Data Object

Name: dataObject1

Type: string

The Browse Types dialog is displayed, as shown in Figure 6–48.

Figure 6–48 Browse Types Dialog



Types.UrnIwaysoftwareTransResponseJDESOOUT.JdeResponse

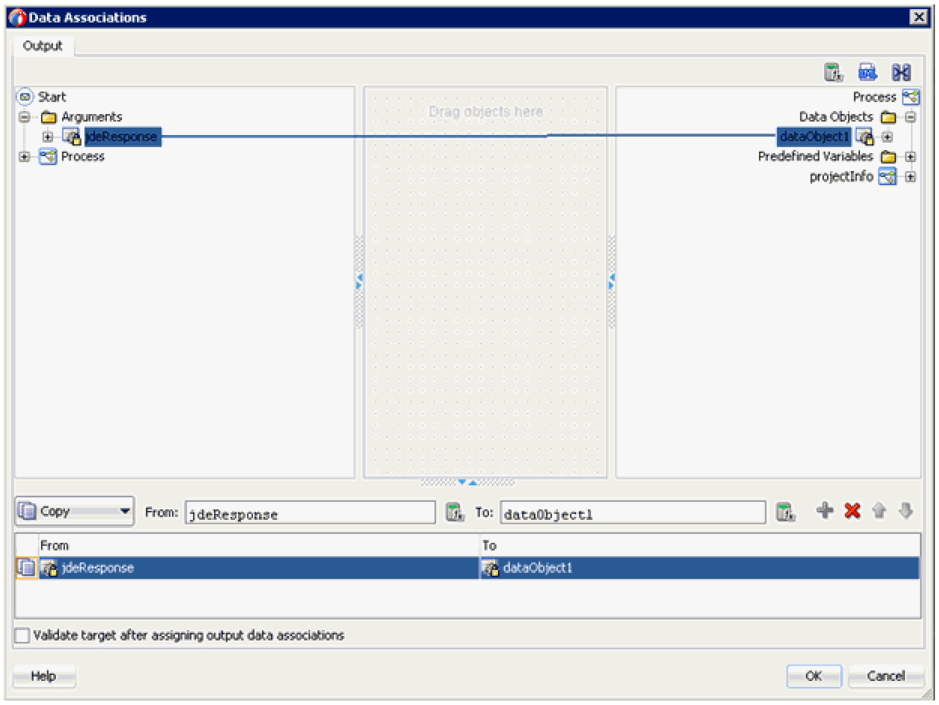
14. Select the component and click **OK**.

    You are returned to the Create Data Object dialog.

15. Click **OK**.

    The Data Object that has been created is displayed under the Data Objects node in the Data Associations dialog, as shown in Figure 6–49.

Figure 6–49 Data Associations Dialog



From: jdeResponse To: dataObject1

From	To
jdeResponse	dataObject1

16. Select and drag the **jdeResponse** Argument under the Start node in the left pane and drag it to the Data Object in the right pane.
17. Click **OK**.

    You are returned to the Properties - Start dialog.
18. Click **OK**.

    You are returned to the Process workspace area.
19. Double-click the created project to load the components.
20. Click the **Save All** icon in the menu bar to save the new inbound BPM process component that was configured.

    You are now ready to create a File adapter for the write operation.

6.5.2.3 Creating a File Adapter for the Write Operation

This section describes how to create a File adapter for the write operation.

Perform the following steps to create a File adapter for the write operation:

1. Drag and drop the **File Adapter** component from the Technology Adapters pane to the External References pane.

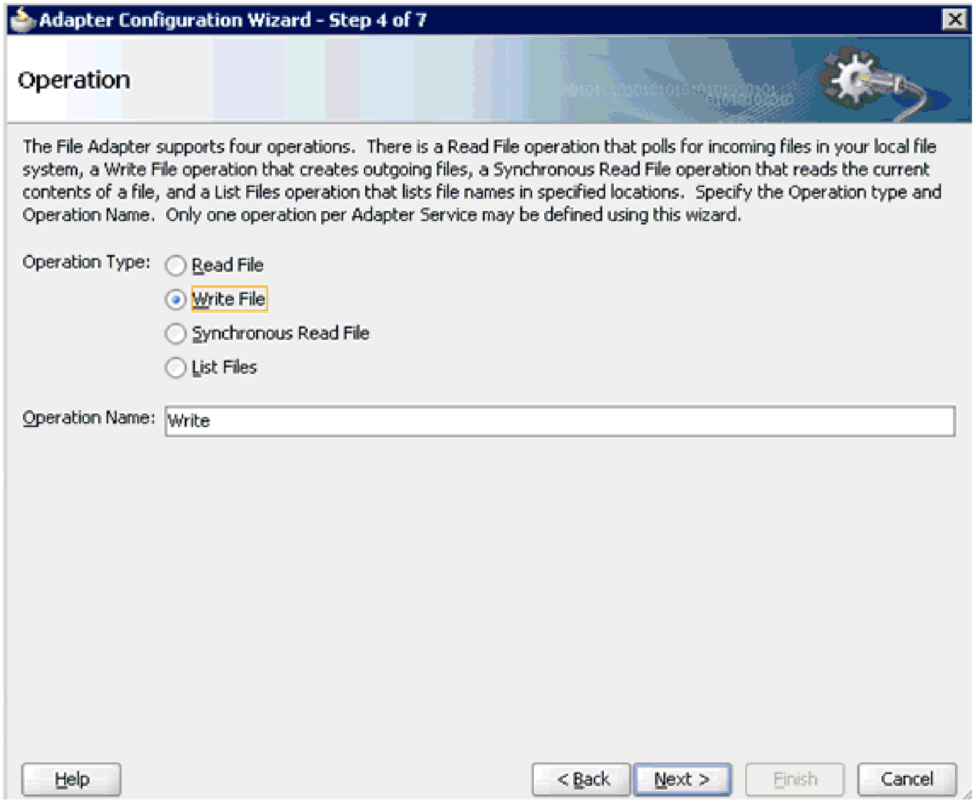
   The Adapter Configuration Wizard is displayed.
2. Type a name for the new File adapter in the Name field and click **Next**.

   The Adapter Interface page is displayed.
3. Ensure that the **Define from operation and schema (specified later)** option is selected.
4. Click **Next**.

   The File Server Connection page is displayed.
5. Click **Next**.

   The Operation page is displayed, as shown in Figure 6–50.

Figure 6–50 Operation Page



The File Adapter supports four operations. There is a Read File operation that polls for incoming files in your local file system, a Write File operation that creates outgoing files, a Synchronous Read File operation that reads the current contents of a file, and a List Files operation that lists file names in specified locations. Specify the Operation type and Operation Name. Only one operation per Adapter Service may be defined using this wizard.

Operation Type: [ ] Read File [✓ Write File] [ ] Synchronous Read File [ ] List Files

Operation Name: Write

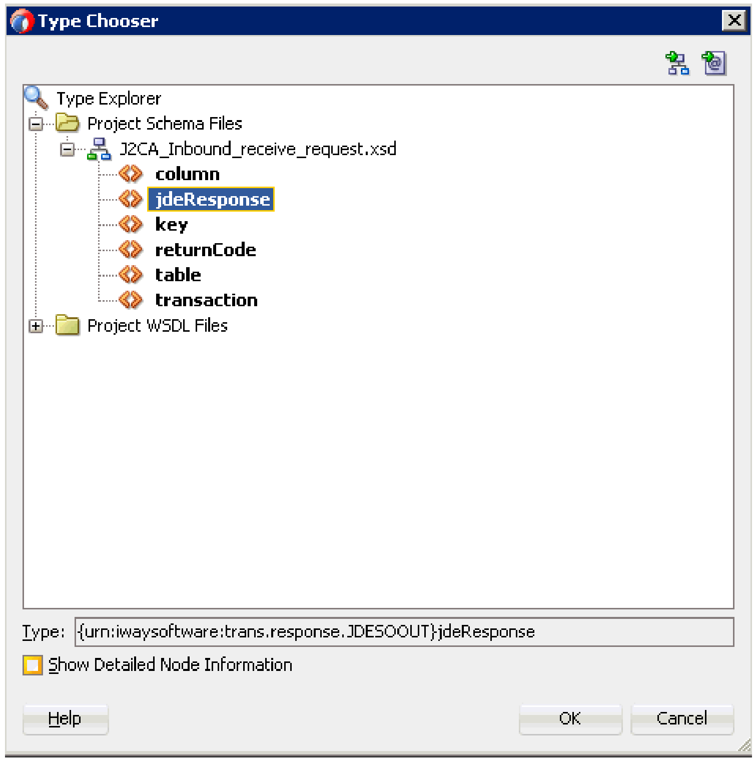
6. Select **Write File** from the list of Operation Type options and specify an Operation Name (for example, Write).
7. Click **Next**.

   The File Configuration page is displayed.
8. Specify a location on your file system where the output file is written.
9. In the File Naming Convention field, specify a name for the output file.
10. Click **Next**.

    The Messages page is displayed.
11. Click **Browse**, which is located to the right of the URL field.

    The Type Chooser dialog is displayed, as shown in Figure 6–51.

Figure 6–51 Type Chooser Dialog



Type: {urn:iwaysoftware:trans.response.JDESOOUT}jdeResponse

12. Expand **Project Schema Files** and **J2CA_Inbound_receive_request.xsd**.
13. Select the available schema.
14. Click **OK**.

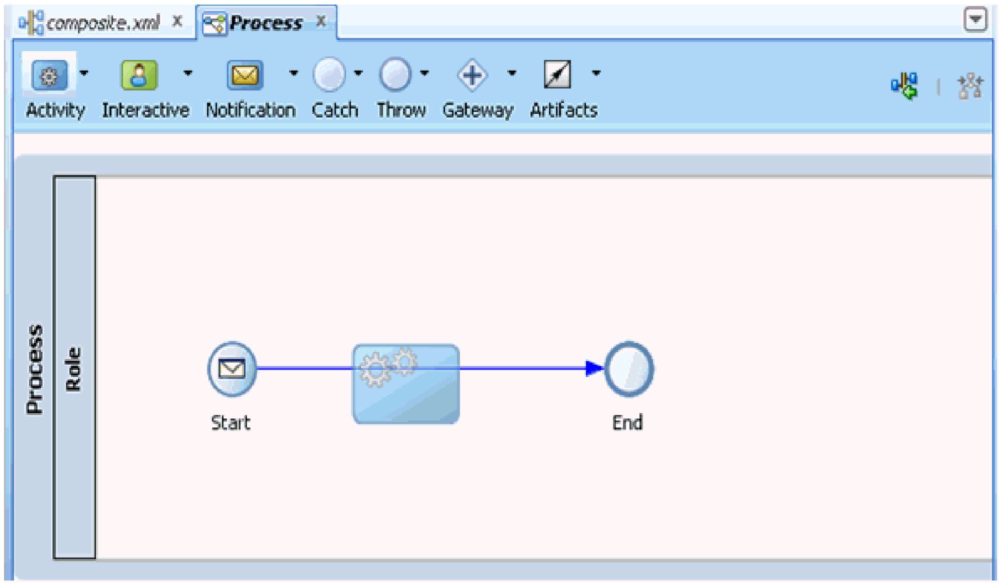
    You are returned to the Messages page.
15. Click **Next**.

    The Finish page is displayed.
16. Click **Finish**.

    The File Adapter service is created in the External References pane.
17. Double-click the BPMN Process component.

    The BPMN process is displayed.
18. Click the **Activity** icon, and select **Service**.
19. Drop the Service icon on the wire between the Start and End event components, as shown in Figure 6–52.

Figure 6–52 Activity Icon

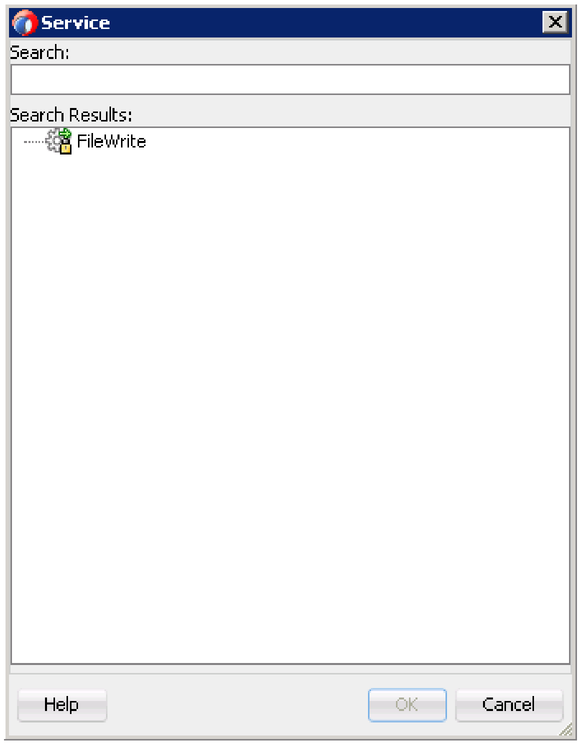


The Properties - ServiceTask dialog is displayed.

20. Click the **Implementation** tab.
21. Select **Service Task** from the Implementation Type list.
22. Select **Service Call** from the Message Exchange Type list.
23. Click the **Browse** icon to the right of the Service field.

    The Type dialog is displayed, as shown in Figure 6–53.

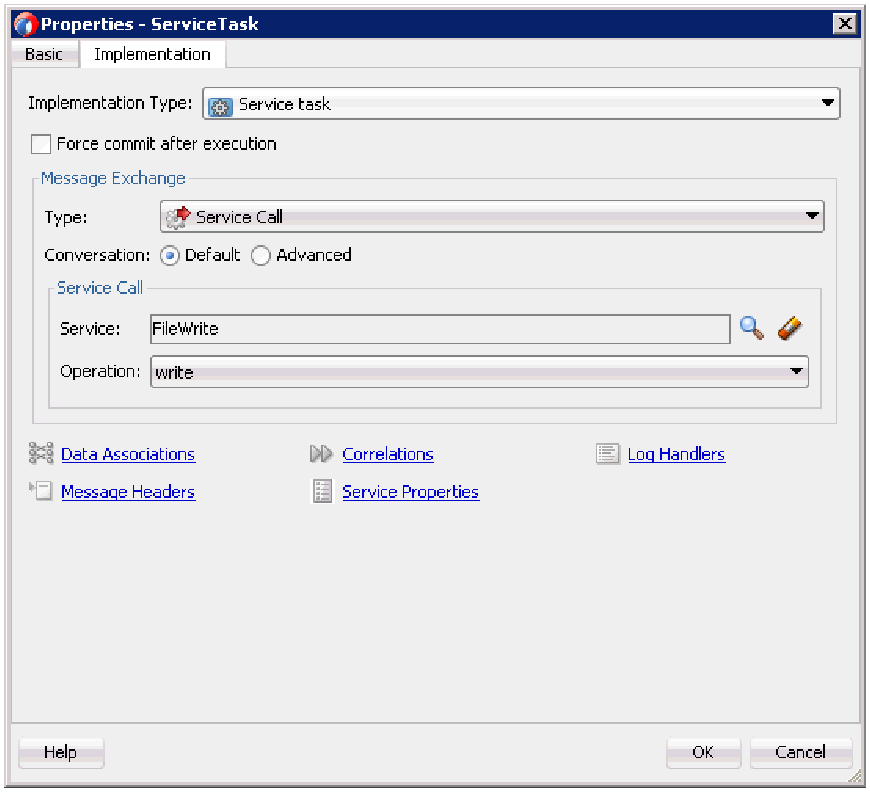
Figure 6–53 Type Dialog



24. Select the service for write operation that has been created and click **OK**.

You are returned to the Properties - ServiceTask dialog, as shown in Figure 6–54.

Figure 6–54 Properties - ServiceTask Dialog


Properties - ServiceTask

Implementation Type: Service task

Force commit after execution

Message Exchange

Type: Service Call

Conversation: [✓ Default] Advanced

Service Call

Service: FileWrite

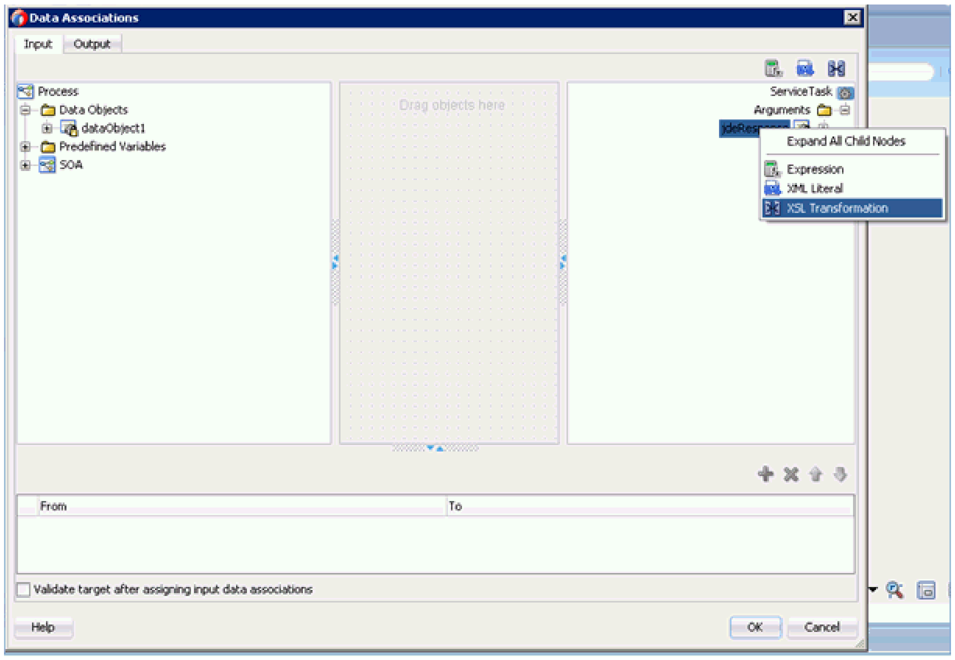
Operation: write

25. Click the **Data Associations** hyperlink.

The Data Associations dialog is displayed.

26. Right-click the **jdeResponse** argument on the right pane and select **XSL Transformation**, as shown in Figure 6–55.

Figure 6–55 XSL Transformation



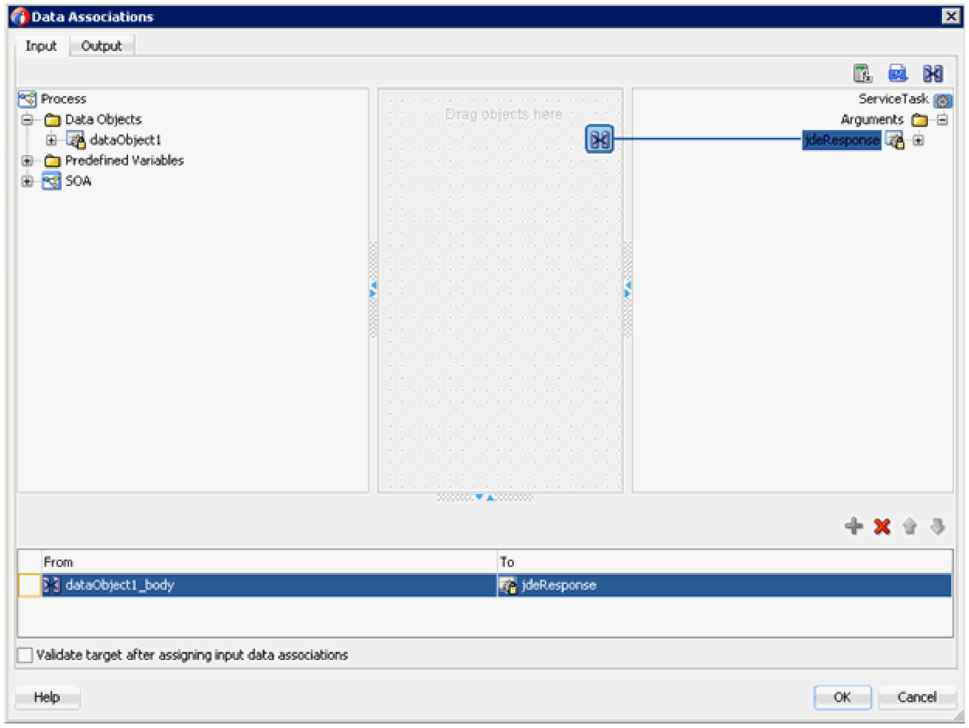
The Create Transformation dialog is displayed.

27. Select the created data object in the Sources area and click the right arrow icon so that the created data object is added to the Selected elements area.

28. Click **OK**.

You are returned to the Data Associations dialog, as shown in Figure 6–56.

Figure 6–56 Data Associations Dialog



From	To
dataObject1_body	jdeResponse

Validate target after assigning input data associations

29. Click **OK**.

    You are returned to the Properties - ServiceTask dialog.

30. Click **OK**.

    The dataobject1_body.xsl tab is displayed.

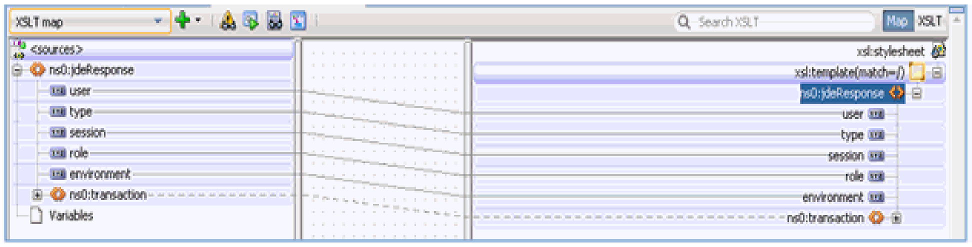
31. Automap the Source and Target elements.

    The Auto Map Preferences dialog is displayed.

32. Accept the default values and click **OK**.

    The transformation is completed, as shown in Figure 6–57.

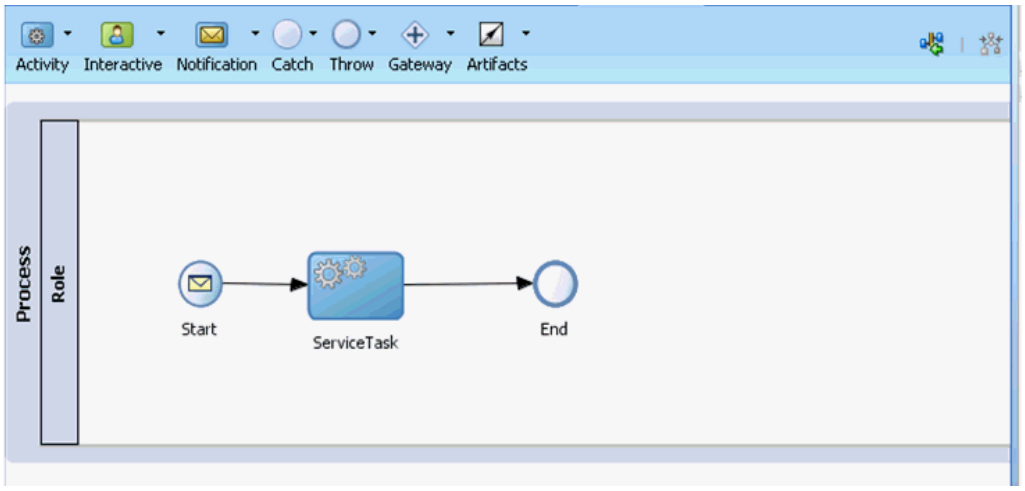
Figure 6–57 Completed Transformation



33. Save the transformation.

34. Return to the Process workspace area, as shown in Figure 6–58.

Figure 6–58 Process Workspace Area



The ServiceTask component is created between the Start event component and the End event component.

35. Click the **Save All** icon in the menu bar to save the new inbound BPM process component that was configured.

6.5.2.4 Adjusting for Known Deployment Issues With 12c

For more information on how to adjust for known deployment issues with 12c, see Section 4.4.3.3, "Adjusting for Known Deployment Issues With 12c" on page 4-26.

You are now ready to deploy the inbound BPM process. You can follow the same procedure that is described in Section 4.5.4, "Deploying the BPEL Inbound Process" on page 4-46.

For more information on how to trigger events in J.D. Edwards OneWorld, see Section 4.5.5, "Triggering an Event in J.D. Edwards OneWorld" on page 4-47.

6.6 Designing an Outbound BPM Process Using Transformations for Service Integration (BSE Configuration)

This section describes how to configure a BPM outbound process to your J.D. Edwards OneWorld system, using a BPM project in Oracle JDeveloper.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\BPM\BSE\Outbound_Project
```

The following tools are required to complete your outbound design-time configuration:

- Oracle Adapter Application Explorer (Application Explorer)
- Oracle JDeveloper BPM Designer (JDeveloper)

This section contains the following topics:

- Section 6.6.1, "Creating an Empty Composite for BPM"
- Section 6.6.2, "Defining a BPM Outbound Process"

Prerequisites

Before you design a BPM outbound process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.6.1, "Generating a WSDL File for Request and Response Services Using a Web Service" on page 4-52.

6.6.1 Creating an Empty Composite for BPM

Perform the following steps to create an empty composite for SOA:

1. Create a new BPM application.
2. Enter a name for the BPM Application, and click **Next**.
3. Enter a name in the Project Name field, and click **Next**.
4. From the Composite Template list, select **Empty Composite** and click **Finish**.

   For more information, see Section 4.5.2, "Creating an Empty Composite for SOA" on page 4-39.

6.6.2 Defining a BPM Outbound Process

This section describes how to define a BPM outbound process. It contains the following topics:

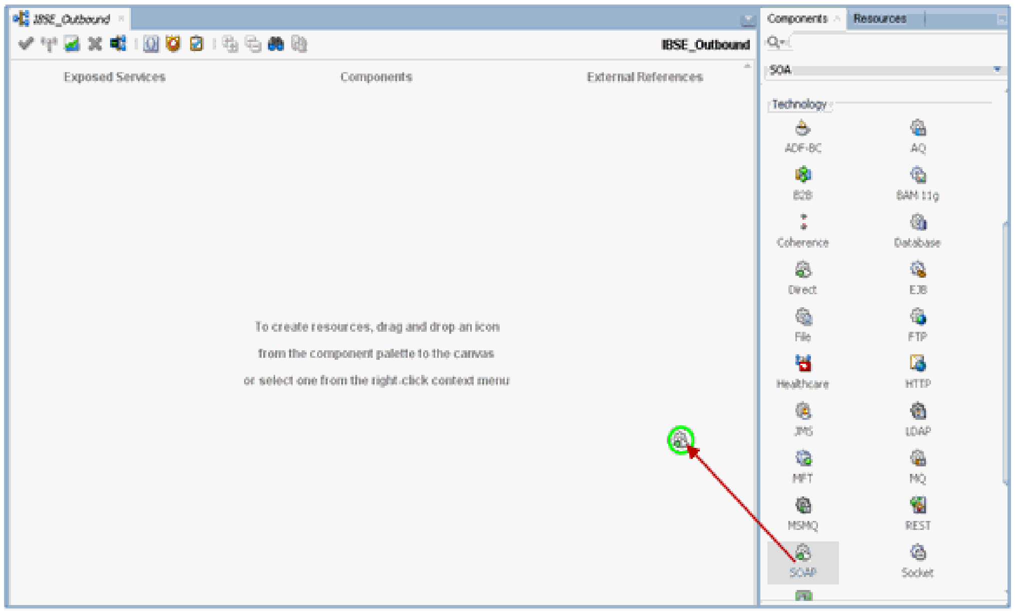
- Section 6.6.2.1, "Configuring a Web Service Component"
- Section 6.6.2.2, "Configuring a BPM Process Component"
- Section 6.6.2.3, "Creating a File Adapter for the Write Operation"

6.6.2.1 Configuring a Web Service Component

Perform the following steps to configure a Web Service component:

1. Double-click the created project to load the components.
2. Drag and drop the **Web Service** node from the Technology Adapters pane to the External References pane, as shown in Figure 6–59.

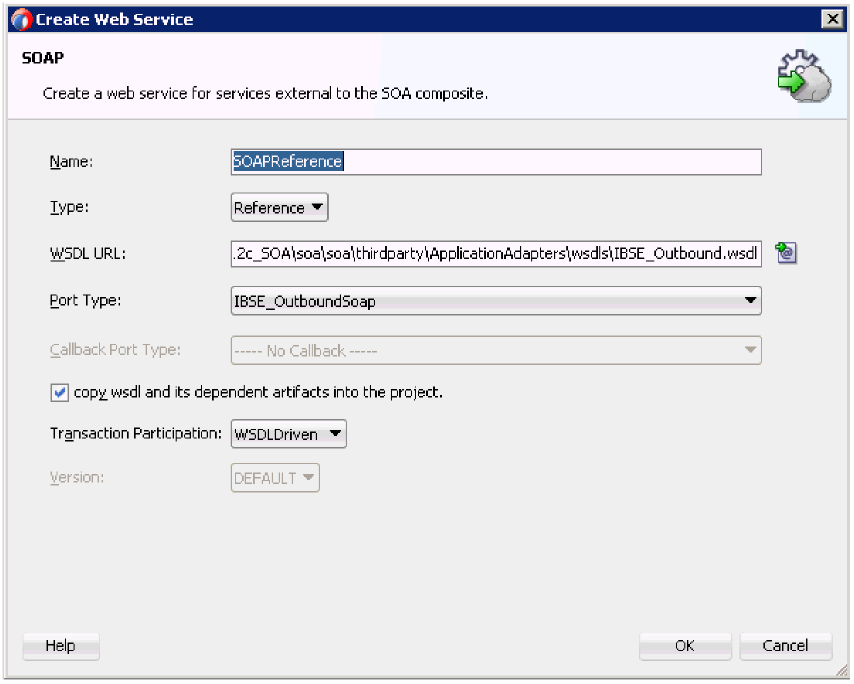
Figure 6–59 Web Service Node



To create resources, drag and drop an icon
from the component palette to the canvas
or select one from the right-click context menu

3. Enter an appropriate name for the Web Service and click on the **Find existing WSDLs** icon, which is located to the right of the WSDL URL field.
4. In the displayed WSDL Chooser window, navigate to the location where the WSDL is exported from the Application Explorer, and select the WSDL.
5. Click **OK**.
6. In the Web Service pane, click **OK**, as shown in Figure 6–60.

Figure 6–60 Web Service Pane


Create Web Service

SOAP

Create a web service for services external to the SOA composite.

Name: SOAPReference

Type: Reference

WSDL URL: .2c_SOA\soa\soa\thirdparty\ApplicationAdapters\wsdls\IBSE_Outbound.wsdl

Port Type: IBSE_OutboundSoap

Callback Port Type: ----- No Callback -----

☑ copy wsdl and its dependent artifacts into the project.

Transaction Participation: WSDLDriven

Version: DEFAULT

7. In the displayed Localize Files window, click **OK**.

   This will import the WSDL file to the project folder

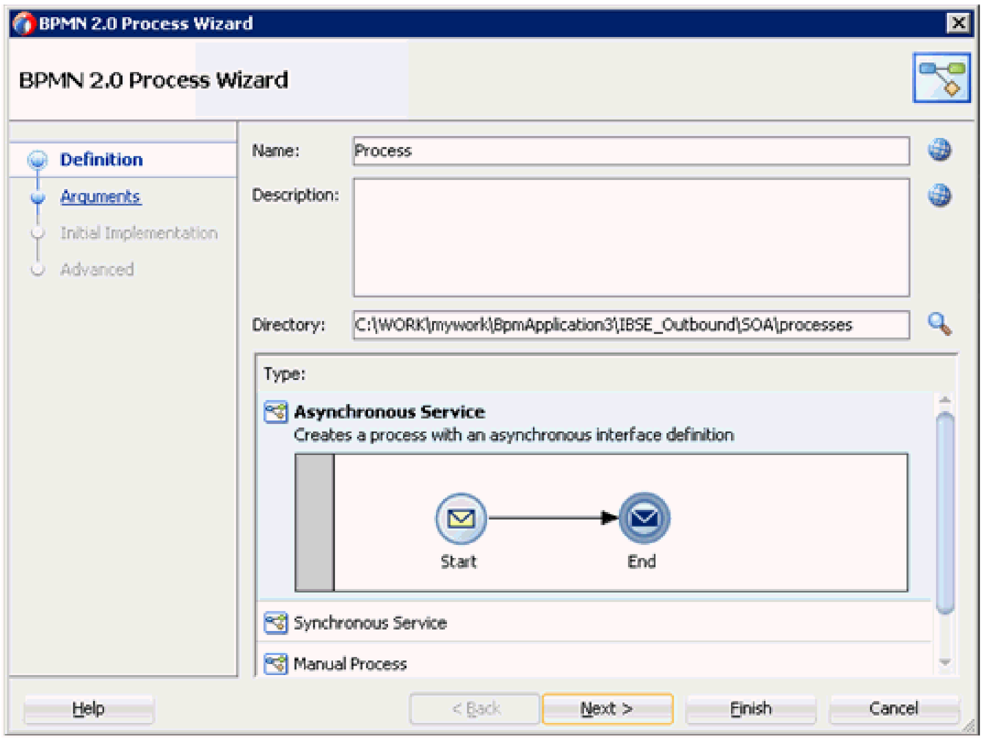
6.6.2.2 Configuring a BPM Process Component

This section describes how to configure an outbound BPM process component.

Perform the following steps to configure a BPM Component:

1. Drag and drop the **BPMN Process** component from the Components pane in to the Components pane.
2. Accept the default option that is selected under the Type area (Asynchronous Service) and click **Finish**, as shown in Figure 6–61.

Figure 6–61 Type Area



Name: Process

Description:

Directory: C:\WORK\mywork\BpmApplication3\IBSE_Outbound\SOA\processes

Type:

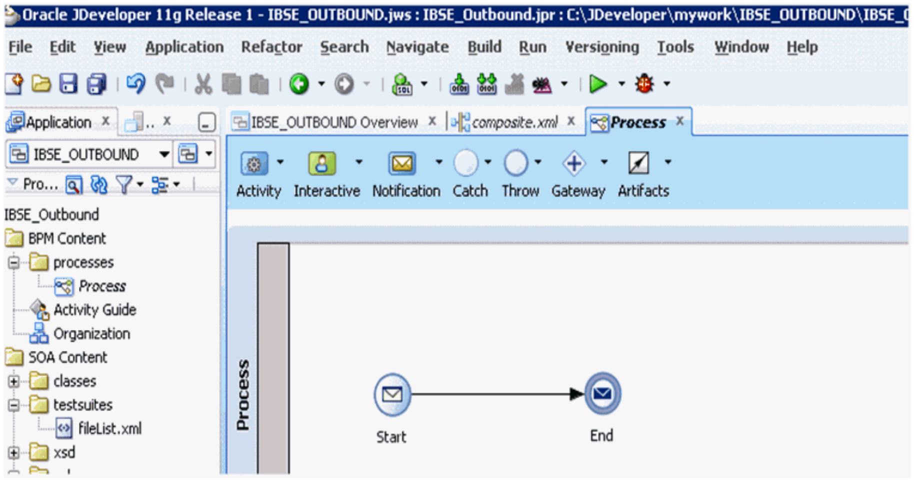
Asynchronous Service
Creates a process with an asynchronous interface definition

Synchronous Service

Manual Process

3. Double click on the Start Event component, as shown in Figure 6–62.

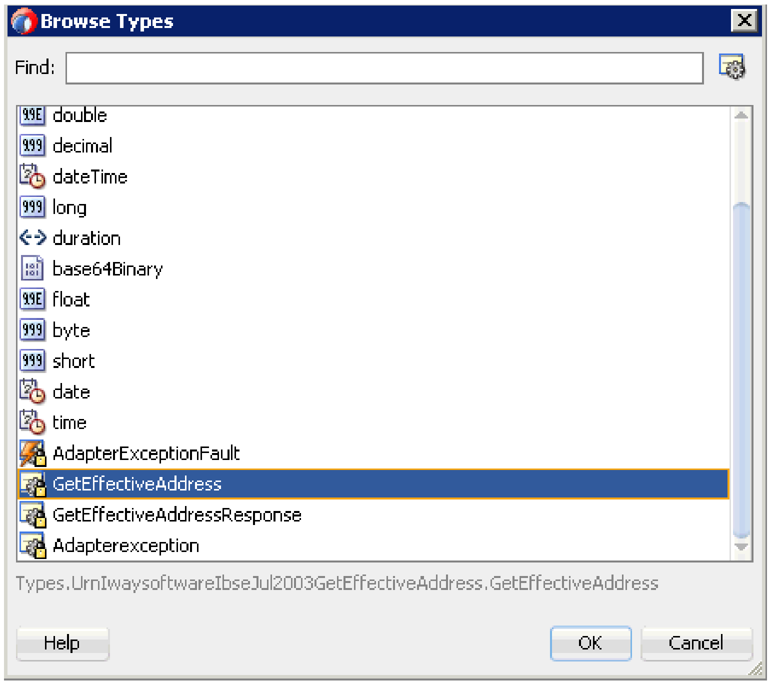
Figure 6–62 Start Event Component



4. In the displayed Properties-start window, click the **Implementation** tab.
5. Click the Plus (+) icon to the right of the Arguments Definition field.

   The Edit Argument window is displayed.
6. Enter a name in the Name field, and then click the Type drop-down list and select **Browse**.
7. Select the **Request** component (for example, GetEffectiveAddress), and click **OK**, as shown in Figure 6–63.

Figure 6–63 Request Component



double
decimal
dateTime
long
duration
base64Binary
float
byte
short
date
time
AdapterExceptionFault
GetEffectiveAddress
GetEffectiveAddressResponse
Adapterexception

Types.UrnIwaysoftwareIbseJul2003GetEffectiveAddress.GetEffectiveAddress

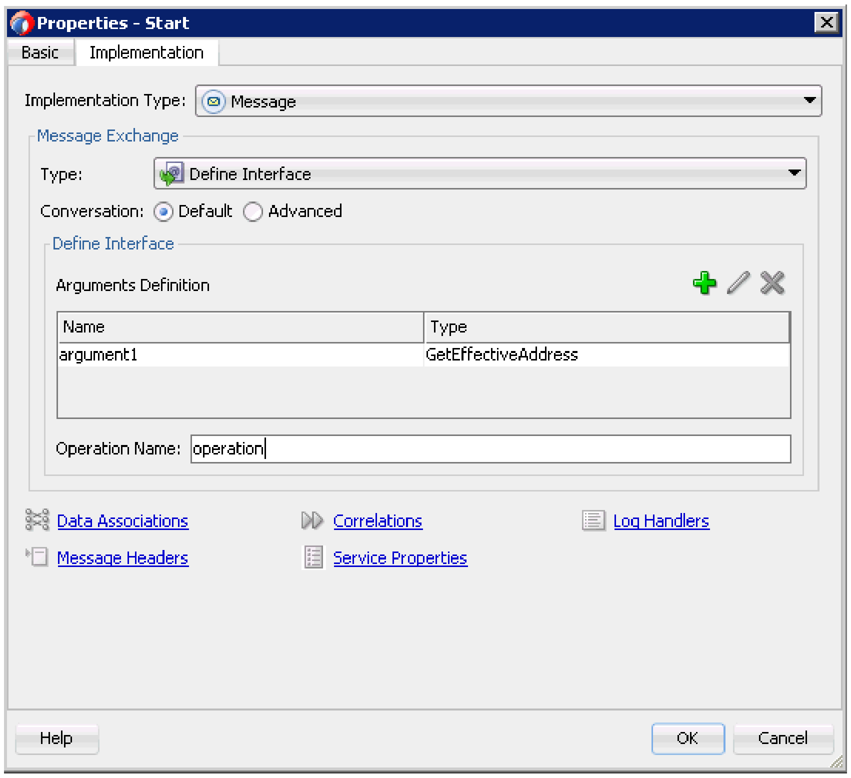
8. In the Edit Argument window that is displayed, click **OK**.

   The Properties - Start window is displayed.

9. In the Operation Name field, change the default entry from **start** to **operation**.

10. Click the **Data Associations** hyperlink, as shown in Figure 6–64.

Figure 6–64 Properties - Start Window



Properties - Start

Basic | Implementation

Implementation Type: Message

Message Exchange

Type: Define Interface

Conversation: (● Default) ○ Advanced

Define Interface

Arguments Definition

Name	Type
argument1	GetEffectiveAddress

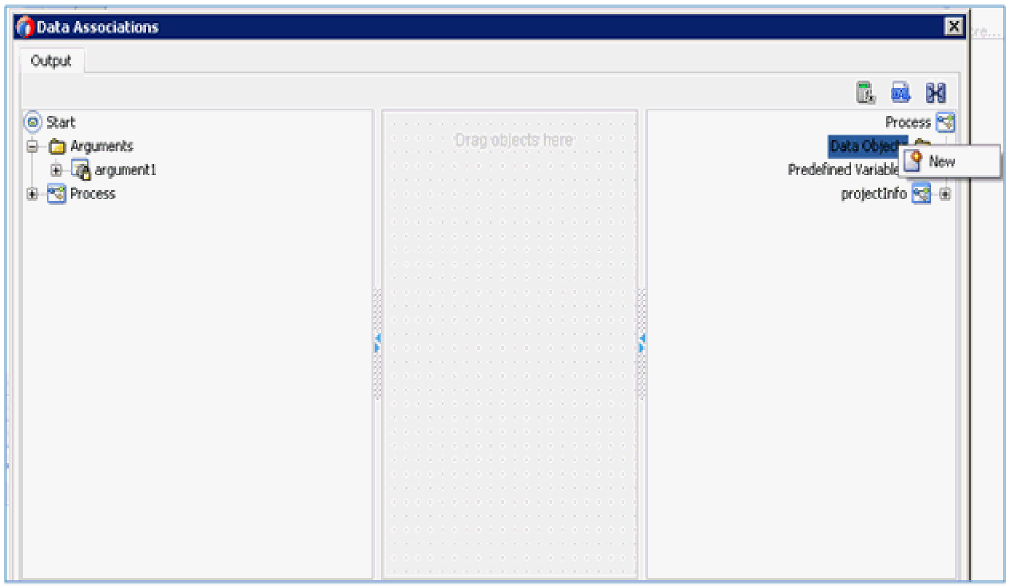
Operation Name: operation

Data Associations · Correlations · Log Handlers · Message Headers · Service Properties

Help · OK · Cancel

11. Right-click the **Data Objects** node in the right pane, under Process, and select **New**, as shown in Figure 6–65.

Figure 6–65 Data Objects Node

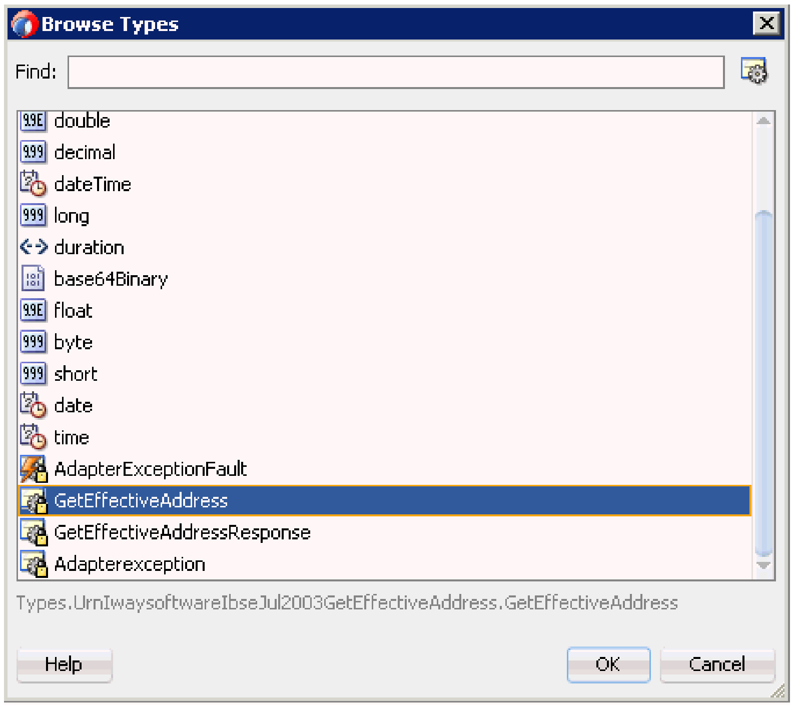


The Create Data Object window is displayed.

12. Enter a name in the Name field, click the Type drop-down list, and select **Browse**.

13. Select the **Request** component (for example, GetEffectiveAddress) and click **OK**, as shown in Figure 6–66.

Figure 6–66 Request Component



Types.UrnIwaysoftwareIbseJul2003GetEffectiveAddress.GetEffectiveAddress

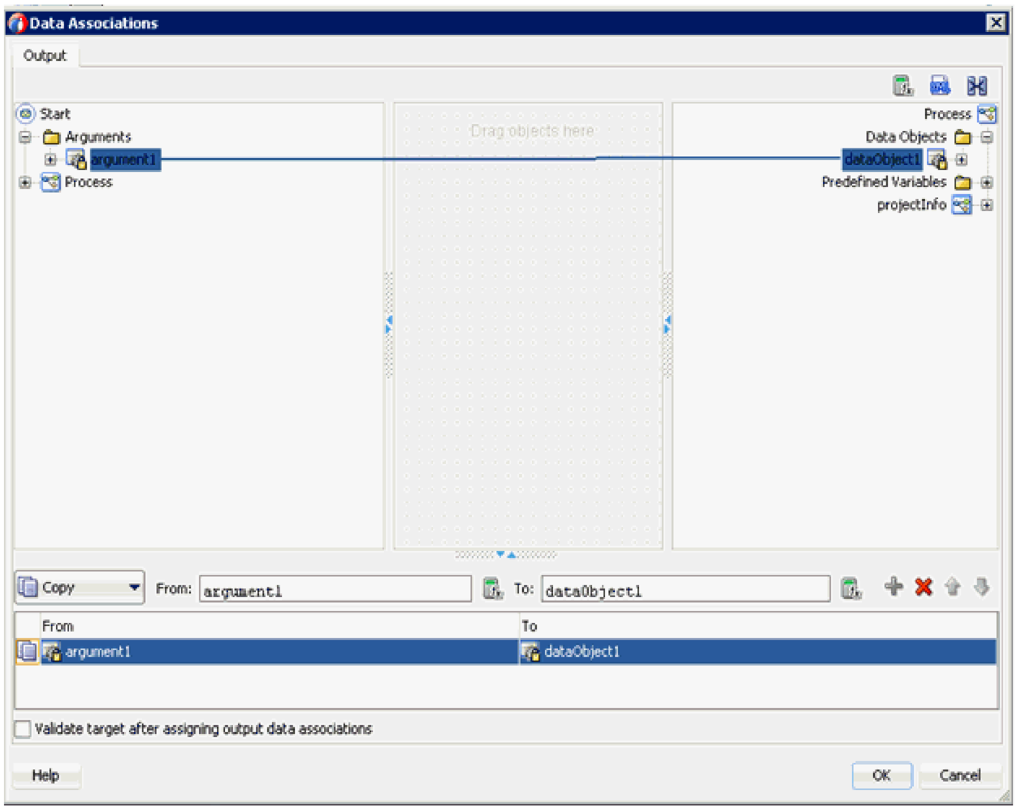
14. In the Create Data Object window, click **OK**.

    The Data Associations window is displayed.

15. Select **argument1** under the Arguments node in the left pane and drag and connect it to **dataObject1**, under Data Objects, in the right pane.

16. Click **OK**, as shown in Figure 6–67.

Figure 6–67 Data Associations



Copy From: argument1 To: dataObject1

From	To
argument1	dataObject1

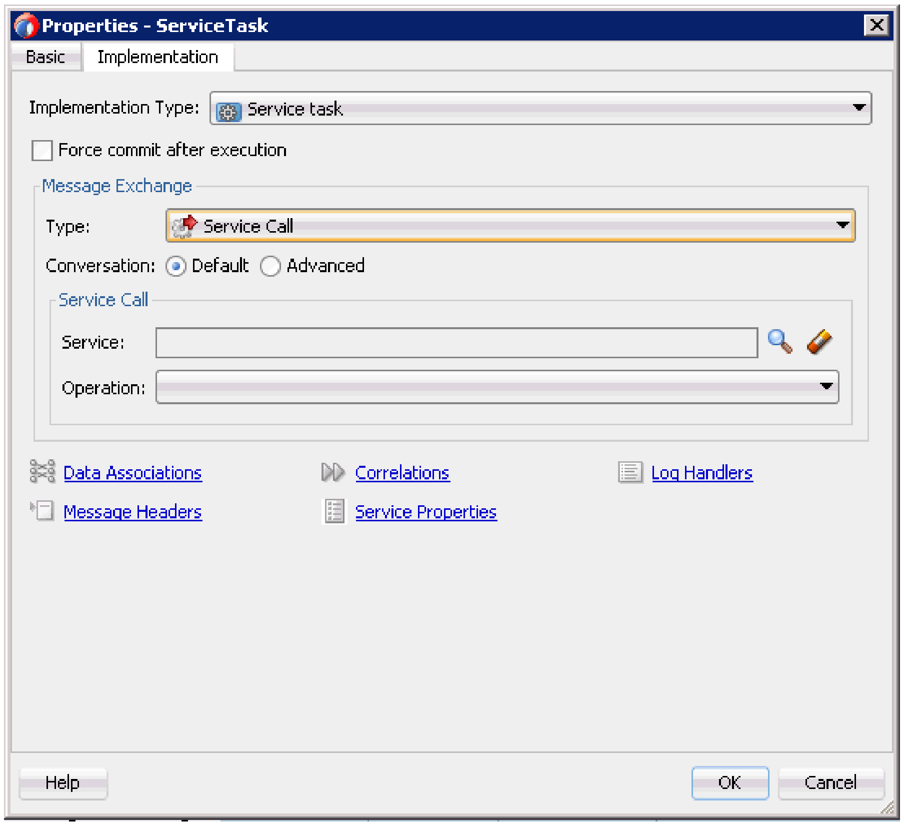
Validate target after assigning output data associations

17. In the Properties - Start window that is displayed, click **OK**.

    You are returned to the Process workspace area.

18. Click the **Activity** drop-down menu and select **Service**.
19. Drop the **Service** icon on the wire between the **Start** and **End** event components.
20. In the displayed Properties - ServiceTask window, click the **Implementation** tab.
21. Select **Service Call** from the Message Exchange Type list.
22. Click the **Browse** icon to the right of the Service field, as shown in Figure 6–68.

Figure 6–68 Browse Icon


Properties - ServiceTask

Basic | Implementation

Implementation Type: Service task

Force commit after execution

Message Exchange

Type: Service Call

Conversation: (●) Default ( ) Advanced

Service Call

Service:

Operation:

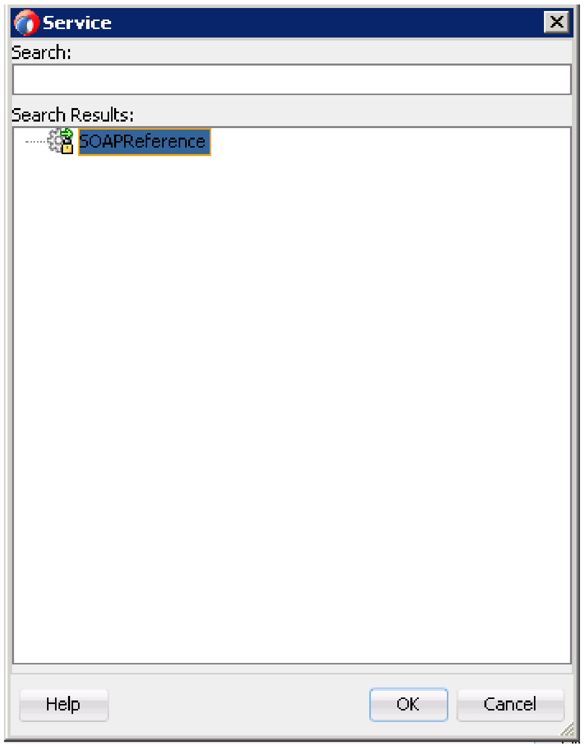
Data Associations
Correlations
Log Handlers
Message Headers
Service Properties

Help
OK
Cancel

The Service window is displayed.

23. Select the Web Service that has been created and click **OK**, as shown in Figure 6–69.

Figure 6–69 Created Web Service

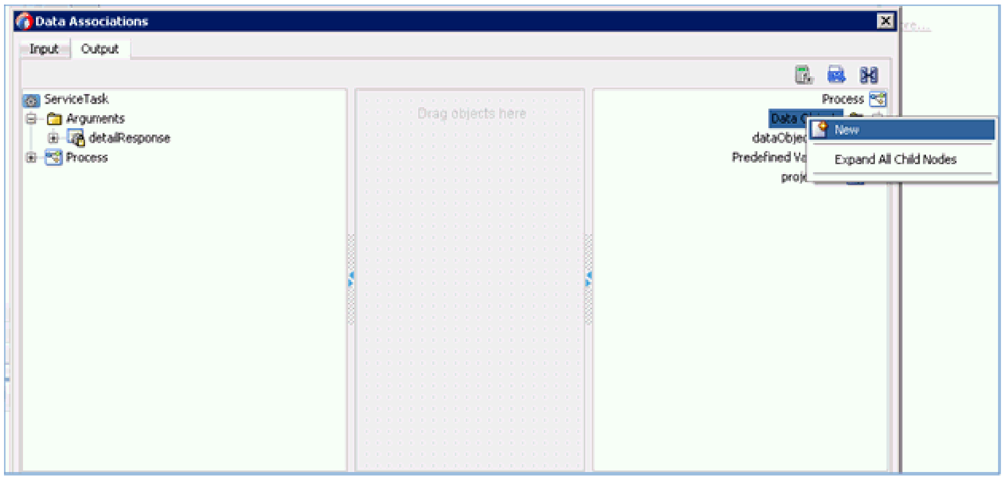


24. In the Properties - ServiceTask window that is displayed, click the **Data Associations** hyperlink.

    The Data Associations window is displayed.

25. Create response Data Object by right-clicking the **Data Objects** node in the right pane of the Output tab and selecting **New**, as shown in Figure 6–70.

Figure 6–70 Data Objects Node

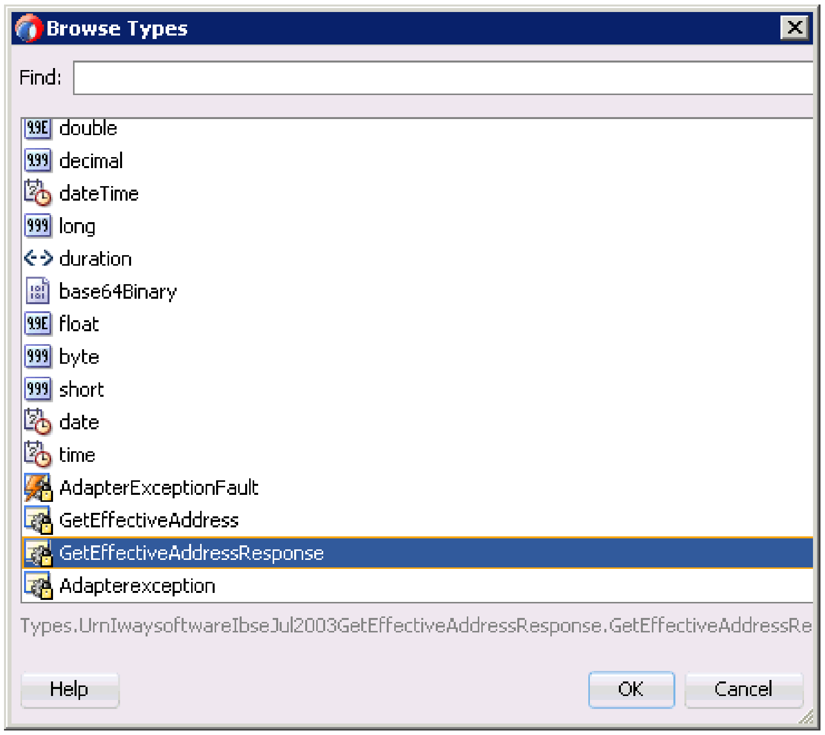


    The Create Data Object window is displayed.

26. Enter a name in the Name field, click the Type drop-down list, and select **Browse**.

27. Select the Response component (for example, GetEffectiveAddressResponse) and click **OK**, as shown in Figure 6–71.

Figure 6–71 Response Component



double
decimal
dateTime
long
duration
base64Binary
float
byte
short
date
time
AdapterExceptionFault
GetEffectiveAddress
GetEffectiveAddressResponse
Adapterexception

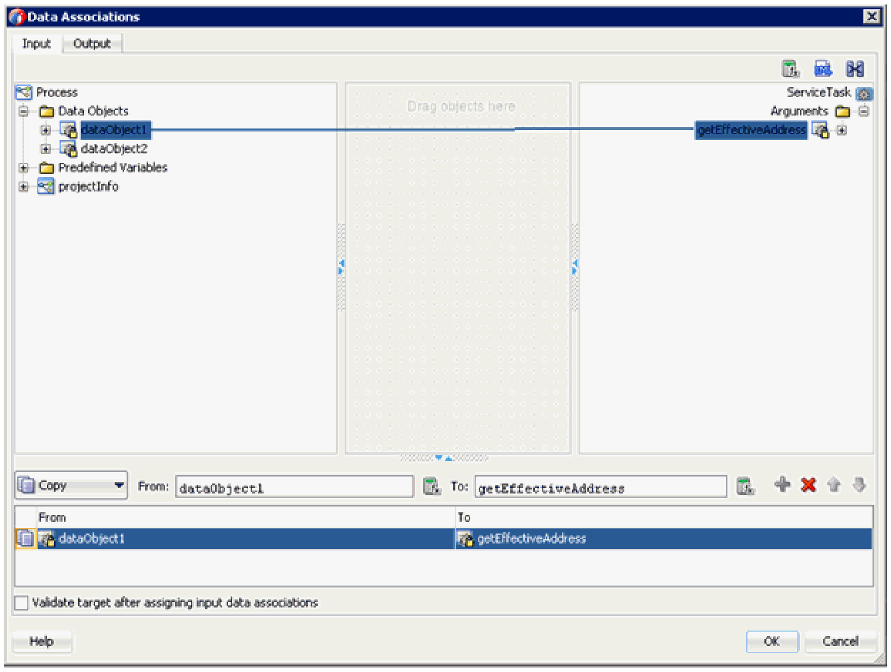
Types.UrnIwaysoftwareIbseJul2003GetEffectiveAddressResponse.GetEffectiveAddressRe

28. In the Create Data Object window, click **OK**.

The Data Associations window is displayed.

29. Select **dataObject1**, under the Data Objects node in the left pane of the Input tab, and drag and connect it to the **getEffectiveAddress** node, under the Arguments node in the right pane, as shown in Figure 6–72.

Figure 6–72 Data Associations



Process
Data Objects
dataObject1
dataObject2
Predefined Variables
projectInfo

ServiceTask
Arguments
getEffectiveAddress

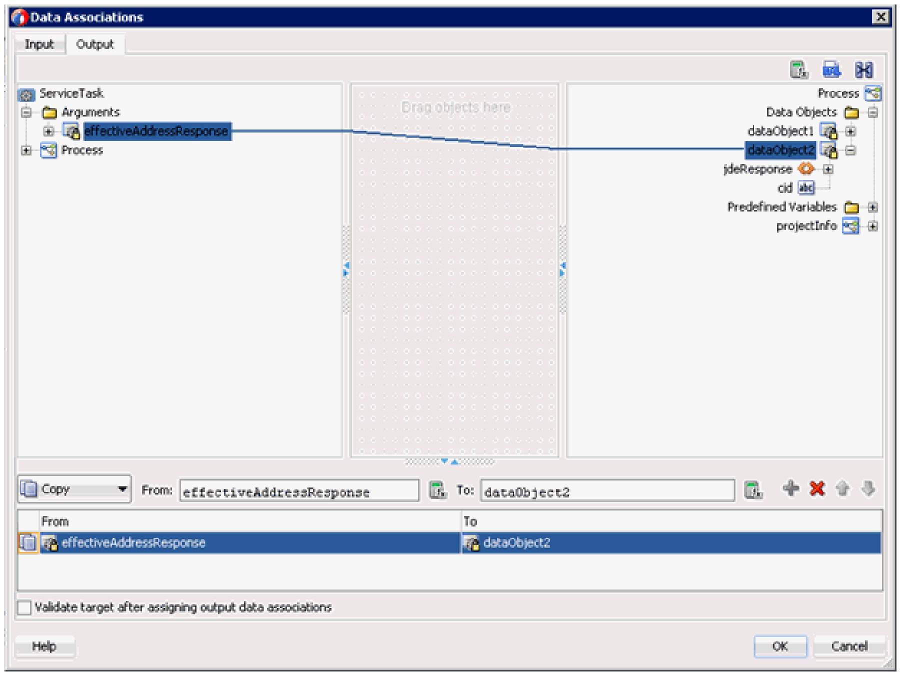
Copy From: dataObject1 To: getEffectiveAddress

From	To
dataObject1	getEffectiveAddress

Validate target after assigning input data associations

30. Click on the **Output** tab and select **GetEffectiveAddressResponse** under the Arguments node in the left pane and drag and connect it to **dataObject2** under the Data Objects node.
31. Click **OK**, as shown in Figure 6–73.

Figure 6–73 Output Tab



From: effectiveAddressResponse To: dataObject2

From	To
effectiveAddressResponse	dataObject2

Validate target after assigning output data associations

32. In the Properties - ServiceTask window that is displayed, click **OK**.
33. Click the **Save All** icon in the menu bar to save the new outbound BPM process component that was configured.
34. Double-click the **composite.xml** node in the left pane.

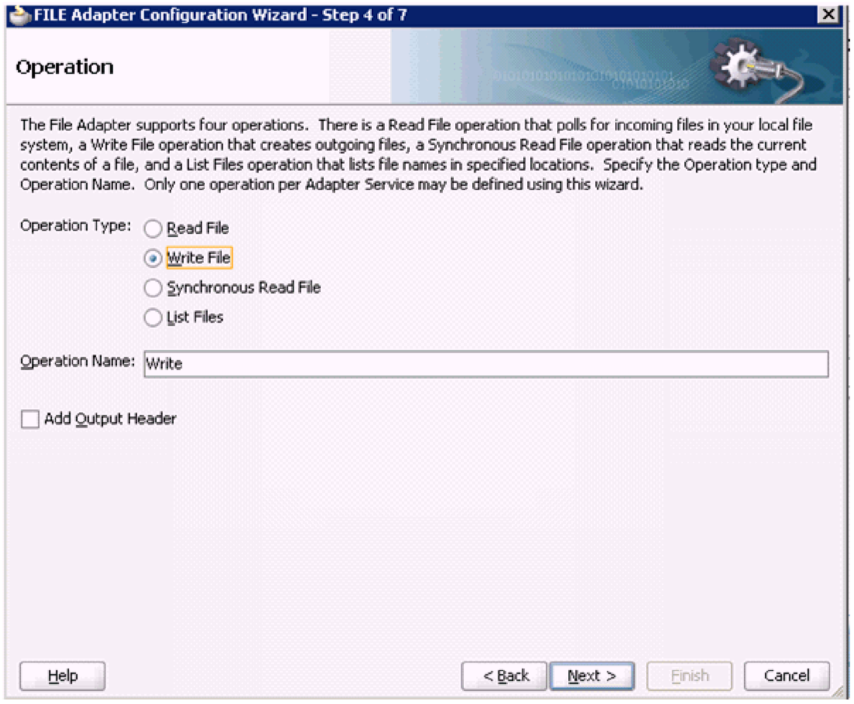
6.6.2.3 Creating a File Adapter for the Write Operation

This section describes how to create a File adapter for the write operation.

Perform the following steps to create a File adapter for the write operation:

1. Drag and drop the **File Adapter** component from the Technology Adapters pane to the External References pane, and provide a name for the File Adapter.
2. In the Adapter Interface pane that is displayed, ensure that the **Define from operation and schema (specified later)** option is selected, and click **Next**.
3. Click **Next**.
4. In the Operation pane that is displayed, select **Write File** from the list of Operation Type options, and click **Next**, as shown in Figure 6–74.

Figure 6–74 Operation Pane


FILE Adapter Configuration Wizard - Step 4 of 7

Operation

The File Adapter supports four operations. There is a Read File operation that polls for incoming files in your local file system, a Write File operation that creates outgoing files, a Synchronous Read File operation that reads the current contents of a file, and a List Files operation that lists file names in specified locations. Specify the Operation type and Operation Name. Only one operation per Adapter Service may be defined using this wizard.

Operation Type: ( ) Read File [✓ Write File] ( ) Synchronous Read File ( ) List Files

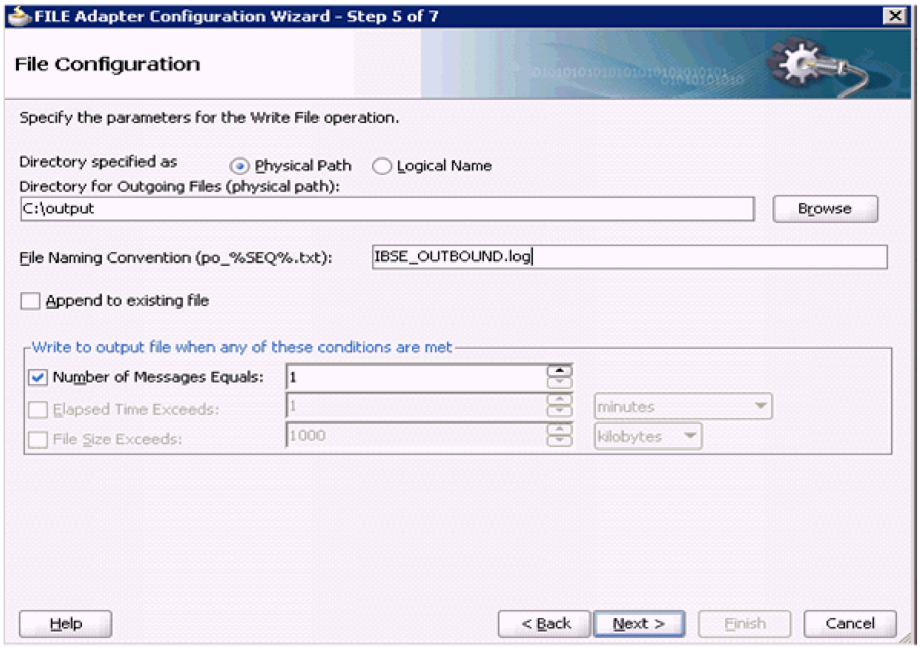
Operation Name: Write

☐ Add Output Header

The File Configuration pane is displayed.

5. In the Directory for Outgoing Files (physical path) field, specify a location on your file system where the output file is written.
6. In the File Naming Convention field, specify a name for the output file.
7. Click **Next**, as shown in Figure 6–75.

Figure 6–75 File Configuration Pane


FILE Adapter Configuration Wizard - Step 5 of 7

File Configuration

Specify the parameters for the Write File operation.

Directory specified as [✓ Physical Path] ( ) Logical Name

Directory for Outgoing Files (physical path): C:\output

File Naming Convention (po_%SEQ%.txt): IBSE_OUTBOUND.log

☐ Append to existing file

Write to output file when any of these conditions are met

☑ Number of Messages Equals: 1

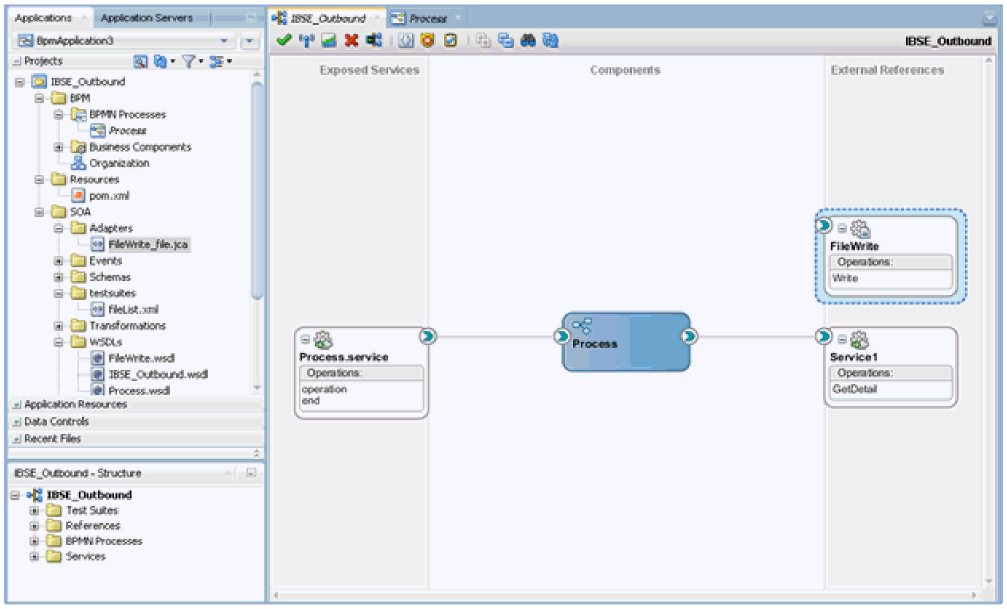
☐ Elapsed Time Exceeds: 1 minutes

☐ File Size Exceeds: 1000 kilobytes

The Messages pane is displayed.

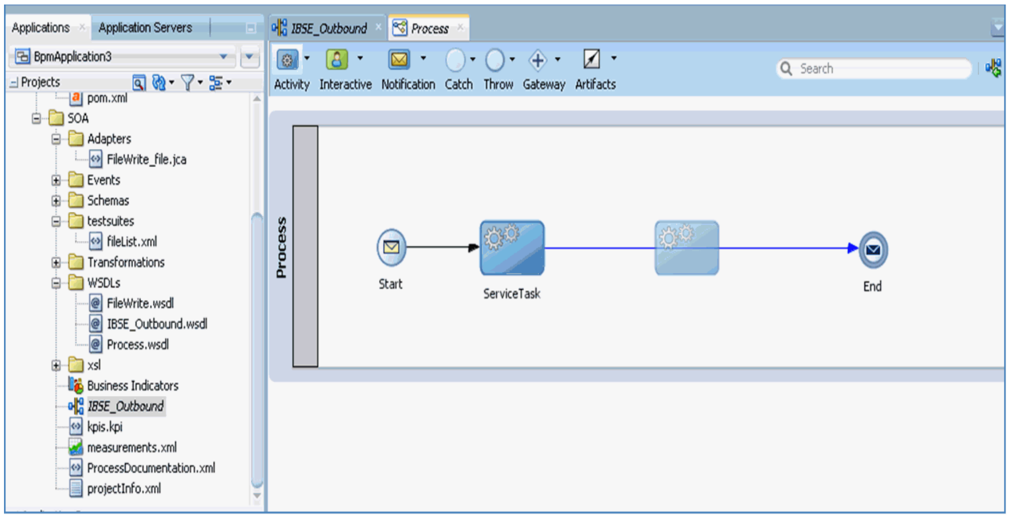
8. Click the **Browse**, which is located to the right of the URL field.
9. In the displayed Type Chooser window, expand **Project WSDL Files**, **IBSE_ Outbound.wsdl**, **Inline Schemas** and then select **GetEffectiveAddressResponse**.
10. Click **OK**.
11. In the Messages pane, click **Next**.
12. In the Finish pane that is displayed, click **Finish**.
13. Double-click the **BPMN Process** component, as shown in Figure 6–76.

Figure 6–76 Composite.xml Tab



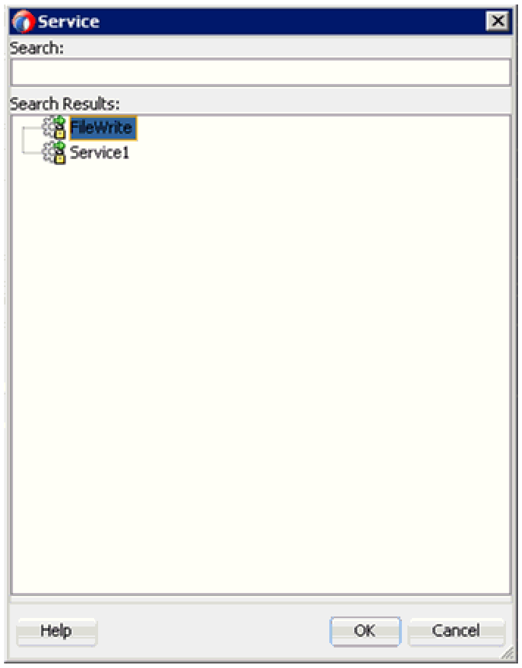
14. Click the **Activity** icon.
15. Drop the **Activity** icon on the wire between the **Service Task** and **End** event components, as shown in Figure 6–77.

Figure 6–77 Activity Icon



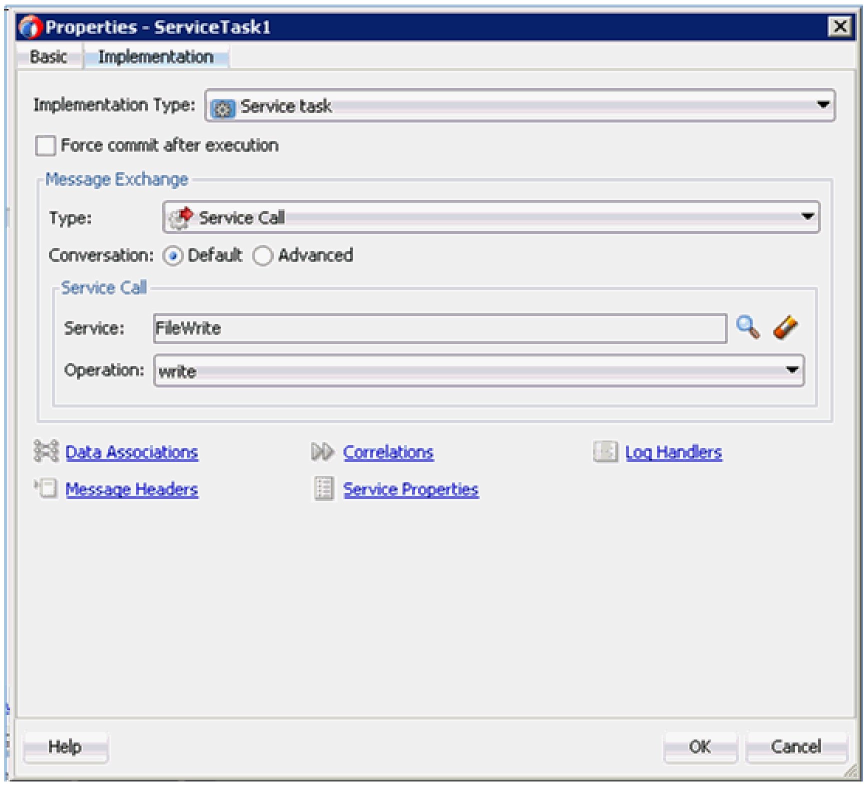
16. In the displayed Properties-ServiceTask1 window, click the **Implementation** tab
17. Select **Service Call** from the Type drop-down list in the Message Exchange section.
18. Click the **Browse** icon to the right of the Service field.
19. Select the service for write operation that has been created and click **OK**, as shown in Figure 6–78.

Figure 6–78 Service Window



20. In the Properties - ServiceTask1 window, click the **Data Associations** hyperlink, as shown in Figure 6–79.

Figure 6–79 Data Associations


Properties - ServiceTask1

Implementation Type: Service task

Force commit after execution: ☐

Message Exchange

Type: Service Call

Conversation: [✓ Default] Advanced

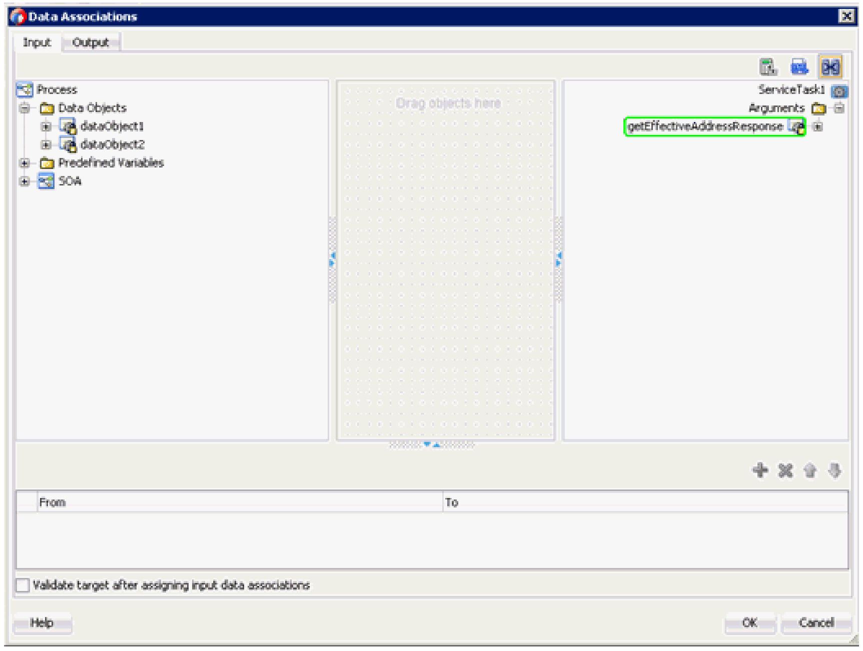
Service Call

Service: FileWrite

Operation: write

21. In the Input tab, click the **XSL Transformation** icon in the top right pane.
22. Drag and drop the **XSL Transformation** icon to the **GetEffectiveAddressResponse** node, as shown in Figure 6–80.

Figure 6–80 GetEffectiveAddressResponse Node


Data Associations

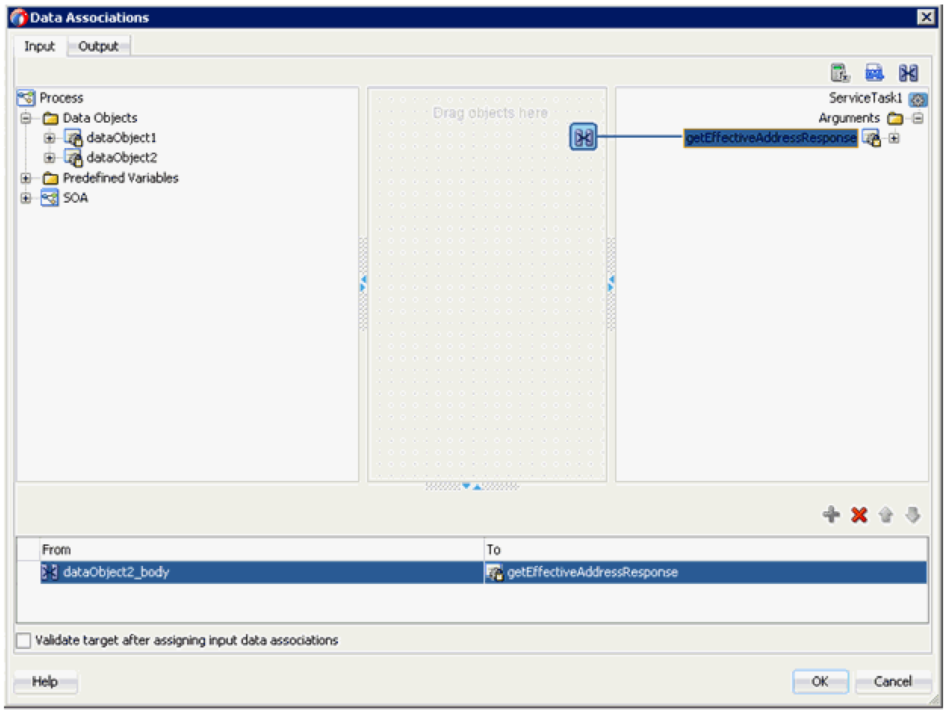
Process
- Data Objects
  - dataObject1
  - dataObject2
- Predefined Variables
- SOA

ServiceTask1
- Arguments
  - getEffectiveAddressResponse

Validate target after assigning input data associations: ☐

23. In the displayed Create Transformation window, select **dataObject2** in the Sources section and click the right arrow symbol.
24. Accept the default value selected in the Target drop-down list and the default name in the Create field by clicking **OK**.
25. In the Data Associations window, click **OK**, as shown in Figure 6–81.

Figure 6–81 Data Associations Window

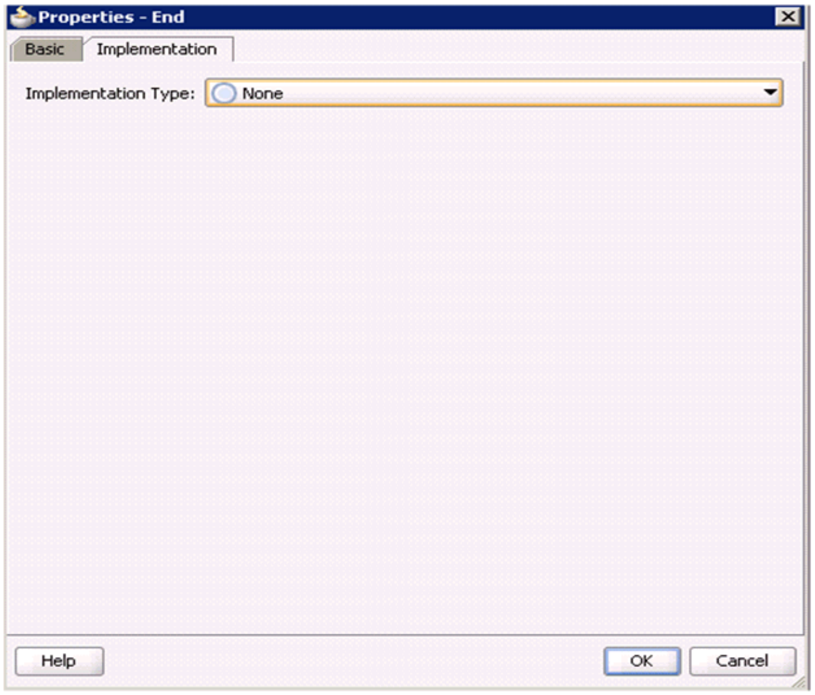


From	To
dataObject2_body	getEffectiveAddressResponse

Validate target after assigning input data associations

26. In the Properties - ServiceTask1 window, click **OK**.
27. In the response_body.xsl tab, map the **ns0:GetEffectiveAddressResponse** source element to the **ns0:GetEffectiveAddressResponse** target element.
28. In the displayed Auto Map Preferences window, retain the default values and click **OK**.
29. Return to the Process workspace area and double-click the **End** event component.
30. In the displayed Properties - End window, click the **Implementation** tab.
31. Select **None** from the Implementation Type drop-down list.
32. Click **OK**, as shown in Figure 6–82.

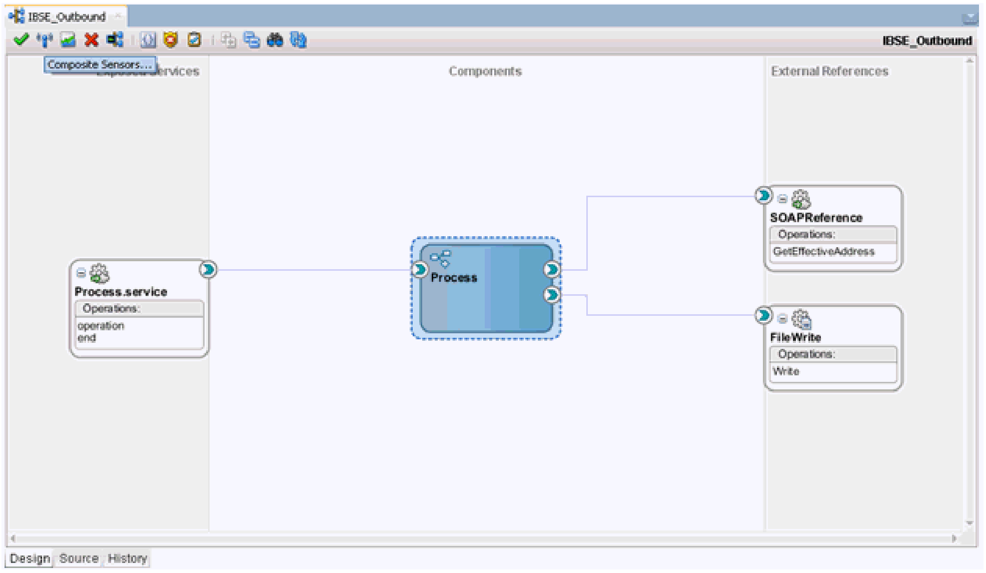
Figure 6–82 Implementation Tab



Implementation Type: None

33. Click the **Save All** icon in the menu bar to save the new outbound BPM component that was configured, as shown in Figure 6–83.

Figure 6–83 Save All Icon



You are now ready to deploy the BPM BSE Outbound process. You can follow the same procedure as Section 6.4.4, "Deploying the BPM Outbound Process" on page 6-25.

Once deployed, you can invoke the input XML as defined in Section 6.4.5, "Invoking the Input XML Document in the Oracle Enterprise Manager Console" on page 6-26.

7 Configuring an Outbound and Inbound Process for Oracle Service Bus Using sbconsole

---

Note: With Release 12*c* (12.2.1.0.0) configuring an outbound and inbound process for Oracle Service Bus using sbconsole has changed.

If you want to create a process for Oracle Service Bus using sbconsole, see *Chapter 2, Configuring an Outbound and Inbound Process for Oracle Service Bus Using sbconsole* in the *Oracle Fusion Middleware Application Adapters Release Notes for 12c (12.2.1.0.0)*.

Oracle Application Adapter for J.D. Edwards OneWorld integrates seamlessly with Oracle Service Bus (OSB) to facilitate Web service integration. OSB is based on the Service-Oriented Architecture (SOA). It consumes adapter services exposed as Web Service Definition Language (WSDL) documents.

This chapter contains the following topics:

- Section 7.1, "Overview of Application Adapter Integration with Oracle Service Bus"
- Section 7.2, "Configuring an Outbound Process Using Sbconsole (J2CA Configuration)"
- Section 7.3, "Configuring an Inbound Process Using sbconsole (J2CA Configuration)"
- Section 7.4, "Configuring an Outbound Process Using Sbconsole (BSE Configuration)"
- Section 7.5, "Configuring JMS Proxy Services Using Oracle Service Bus (J2CA Configuration)"
- Section 7.6, "Configuring HTTP Proxy Services Using Oracle Service Bus (J2CA Configuration)"

7.1 Overview of Application Adapter Integration with Oracle Service Bus

To integrate with Oracle Service Bus (OSB), Oracle Application Adapter for J.D. Edwards OneWorld must be deployed in the same Oracle WebLogic Server as OSB. The underlying adapter services must be exposed as WSDL files, which are generated during design time in Oracle Adapter Application Explorer (Application

Explorer) for both request-response (outbound) and event notification (inbound) services of the adapter.

7.2 Configuring an Outbound Process Using Sbconsole (J2CA Configuration)

This section describes how to configure an outbound process using sbconsole for J2CA configurations.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\OSB\J2CA\JDEdwards_Sample_J2CA_OSB_Outbound_Project
```

This section includes the following topics:

- Section 7.2.1, "Starting Oracle Service Bus and Creating Project Folders"
- Section 7.2.2, "Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus"
- Section 7.2.3, "Publishing a WSDL From Application Explorer to Oracle Service Bus"
- Section 7.2.4, "Configuring a WSDL-based Business Service"
- Section 7.2.5, "Configuring a File Type Business Service"
- Section 7.2.6, "Configuring a Pipeline With Proxy Service"

7.2.1 Starting Oracle Service Bus and Creating Project Folders

This section describes how to start Oracle Service Bus (OSB) and create project folders.

Perform the following steps to start Oracle Service Bus and create project folders:

1. Start the Oracle WebLogic Server for the Oracle WebLogic Server domain that you have configured.
2. Open the Oracle Service Bus Console in a Web browser by entering the following URL:

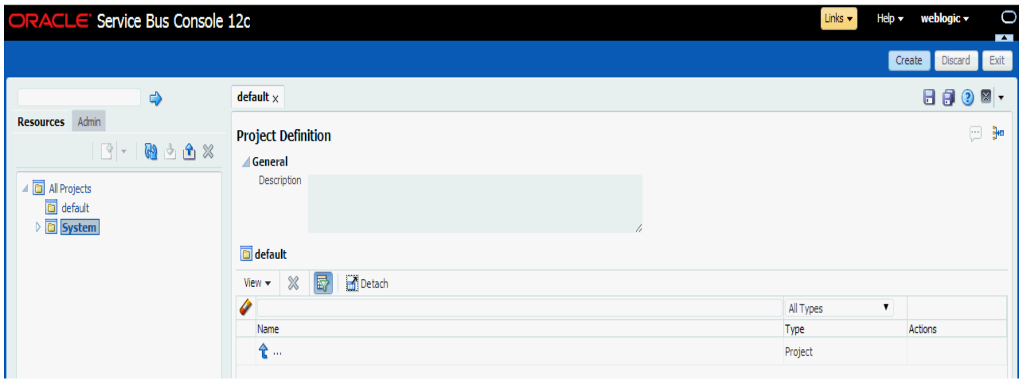
   `http://hostname:port/sbconsole`

   Where *hostname* is the name of the machine where Oracle WebLogic Server is running and *port* is the port for the domain you are using.

   The Oracle Service Bus Console logon page is displayed.
3. Log on to the Oracle Service Bus Console using a valid user name and password.

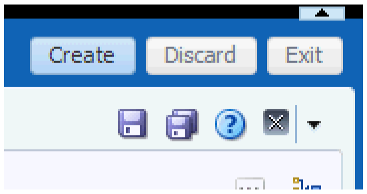
   The Oracle Service Bus Console home page is displayed, as shown in Figure 7–1.

Figure 7–1 Oracle Service Bus Console Home Page



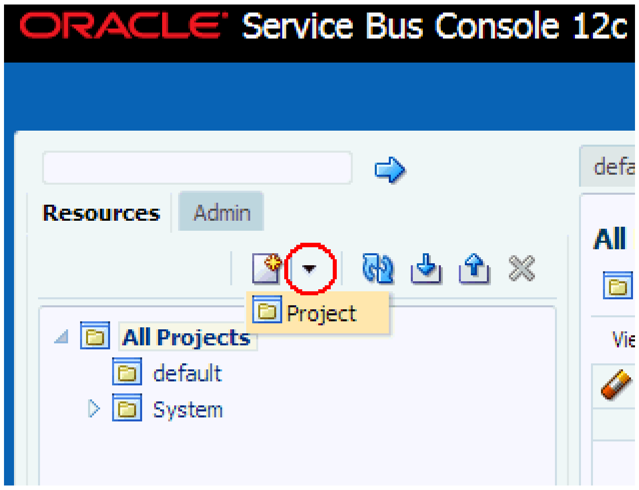
4. Click **Create** in the right pane of the Oracle Service Bus session, as shown in Figure 7–2.

Figure 7–2 Oracle Service Bus Session



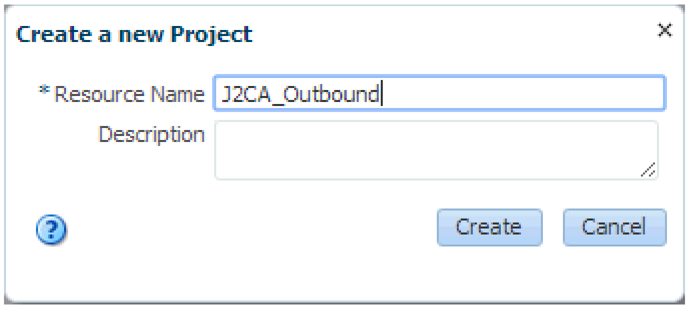
5. Select **All Projects**, click the down arrow in the left pane, and select **Project**, as shown in Figure 7–3.

Figure 7–3 All Projects Folder



The Create a new Project window is displayed, as shown in Figure 7–4.

Figure 7–4 Create New Project Window


Create a new Project

* Resource Name: J2CA_Outbound

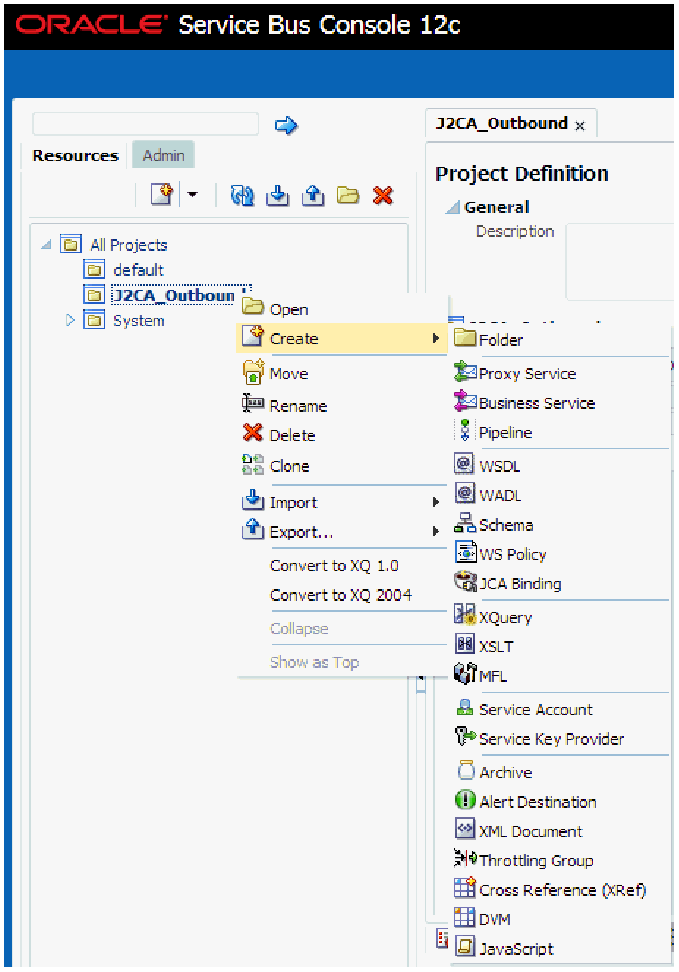
Description:

6. Provide a valid name for the new project (for example, J2CA_Outbound) in the Resource Name field, and click **Create**.

   The new project is successfully created and listed.

7. Right-click the newly created project, select **Create**, and click **Folder**, as shown in Figure 7–5.

Figure 7–5 Create Option



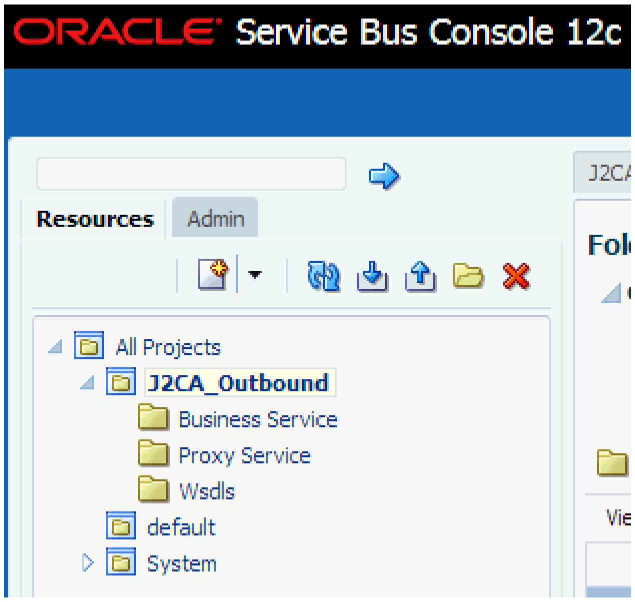
The Create a new Folder window is displayed.

8. In the Resource Name field, type **Business Service** and click **Create**.

9. Repeat steps 7 and 8 to create folders with the names **Proxy Service** and **Wsdls**.

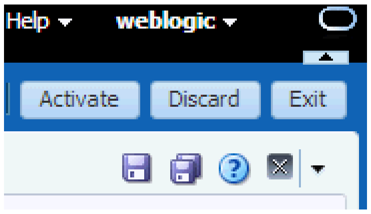
   The Business Service, Proxy Service, and Wsdls folders are listed in the left pane below the project node, as shown in Figure 7–6.

Figure 7–6 Project Node



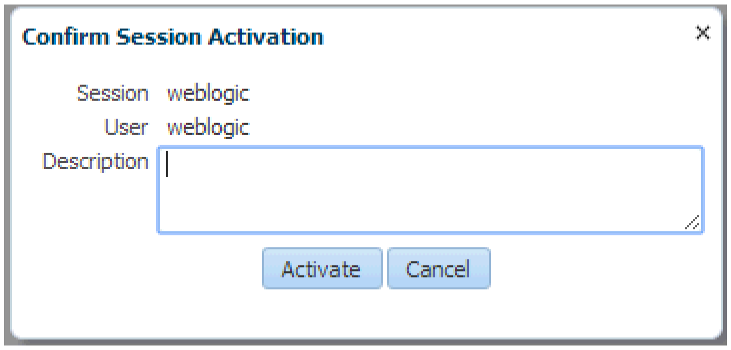
10. Click **Activate** in the right pane of the Oracle Service Bus session, as shown in Figure 7–7.

Figure 7–7 Activate Button



11. In the Confirm Session Activation page, click **Activate** to save the changes, as shown in Figure 7–8.

Figure 7–8 Confirm Session Activation Window


Confirm Session Activation

Session weblogic

User weblogic

Description

7.2.2 Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus

Before starting and using Application Explorer to publish a WSDL directly to the Oracle Service Bus (OSB) Console (project/folder), OSB users must perform the following steps:

1. Open the command prompt window.
2. Navigate to the following directory:

   ```
   <ORACLE_HOME>\user_projects\domains\base_domain\bin
   ```

3. Execute **setDomainEnv.cmd** (Windows) or **. ./setDomainEnv.sh** (UNIX/Linux).

   This command sets the class path for Application Explorer to access the Oracle WebLogic Server APIs to publish the WSDLs to the OSB Console.

4. Do not close the command prompt window.
5. Navigate to the following directory:

   ```
   <ADAPTER_HOME>\tools\iwae\bin
   ```

6. Execute **ae.bat** (Windows) or **iwae.sh** (UNIX/Linux) to start Application Explorer.

   You are now ready to publish WSDLs from Application Explorer to the OSB Console.

7.2.3 Publishing a WSDL From Application Explorer to Oracle Service Bus

Perform the following steps to publish a WSDL from Application Explorer to Oracle Service Bus:

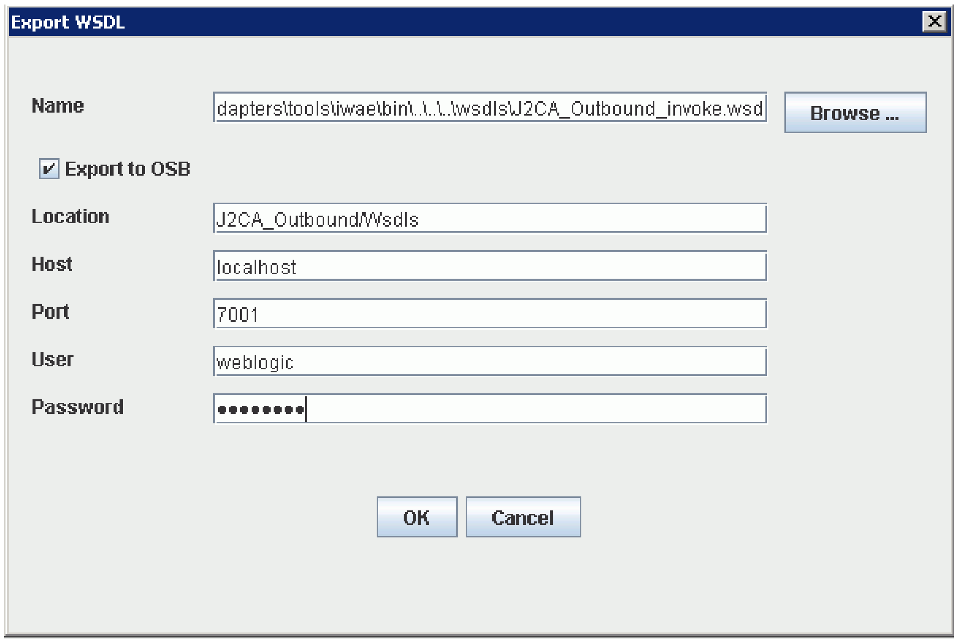
1. Start Application Explorer, connect to a J2CA configuration, and connect to a J.D. Edwards OneWorld target.

   For more information, see Chapter 2, "Configuring Oracle Application Adapter for J.D. Edwards OneWorld".

2. Expand the J.D. Edwards OneWorld target to which you are connected.
3. Right-click a method and then select **Create Outbound JCA Service (Request/Response)** from the menu.

   The Export WSDL dialog is displayed, as shown in Figure 7–9.

Figure 7–9 Export WSDL Dialog



Name: dapters\tools\iwae\bin\..\..\..\wsdls\J2CA_Outbound_invoke.wsd

☑ Export to OSB

Location: J2CA_Outbound/Wsdls

Host: localhost

Port: 7001

User: weblogic

Password: ••••••••

4. In the Name field, a default file name for the WSDL file is provided. You can accept the default or provide your own.
5. Select the **Export to OSB** option.
6. In the Location field, enter the folder name in Oracle Service Bus where you want to publish the WSDL document.

   The location is composed of an Oracle Service Bus project name and optionally, one or more folder names. The project name and any folder names must be separated by a forward slash character "/".
7. In the Host field, enter the name of the machine where Oracle Service Bus is installed.
8. In the Port field, enter the port that is being used by Oracle Service Bus.
9. In the User field, enter your username to access Oracle Service Bus.
10. In the Password field, enter your password to access Oracle Service Bus.
11. Click **OK**.

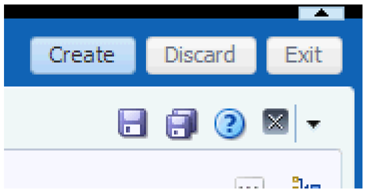
    The WSDL is published to the location specified in the Export WSDL dialog and is now available for use with a Business Service or Proxy Service in Oracle Service Bus.

7.2.4 Configuring a WSDL-based Business Service

Perform the following steps to configure a WSDL-based Proxy Service:

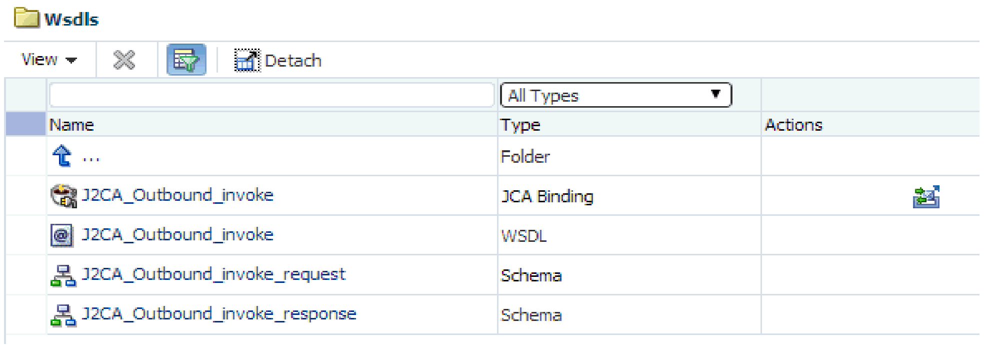
1. Open the Oracle Service Bus Console and click **Create** in the right pane of the Oracle Service Bus session, as shown in Figure 7–10.

Figure 7–10 Create Button



2. Double-click the created WSDL folder in the left pane (for example, Wsdls) and ensure that the exported WSDL is listed in the right pane, as shown in Figure 7–11.

Figure 7–11 Wsdls Folder

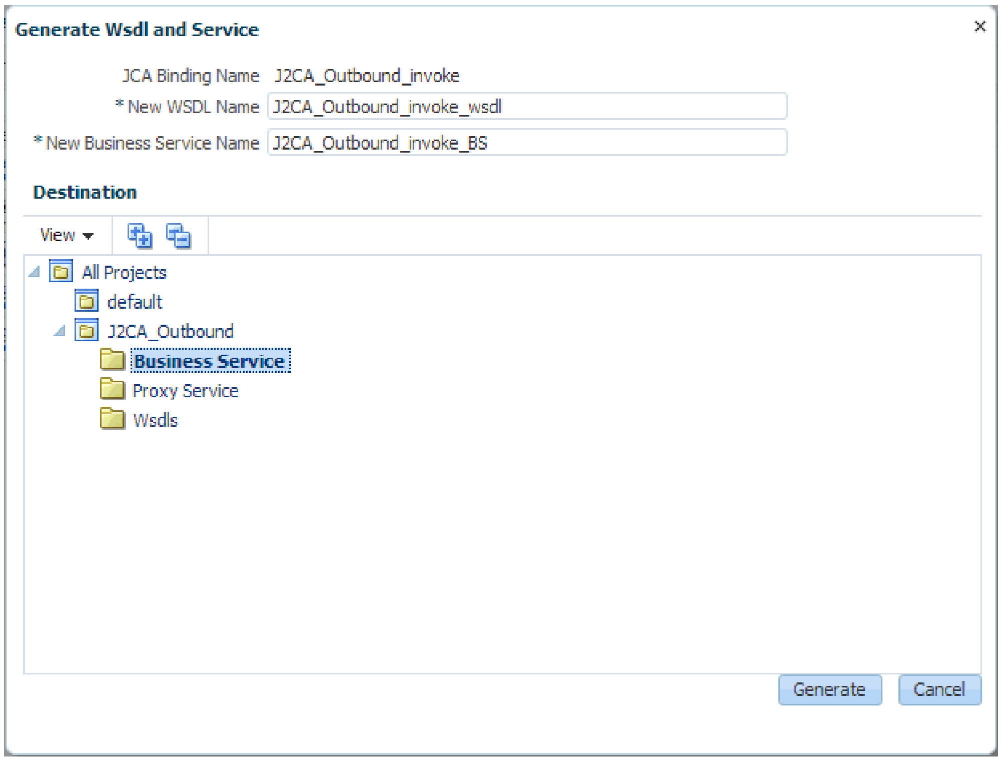


Name	Type	Actions
...	Folder	
J2CA_Outbound_invoke	JCA Binding	
J2CA_Outbound_invoke	WSDL	
J2CA_Outbound_invoke_request	Schema	
J2CA_Outbound_invoke_response	Schema	

3. Click the icon that corresponds to the JCA Binding in the Actions column.

   The Generate WSDL and Service window is displayed, as shown in Figure 7–12.

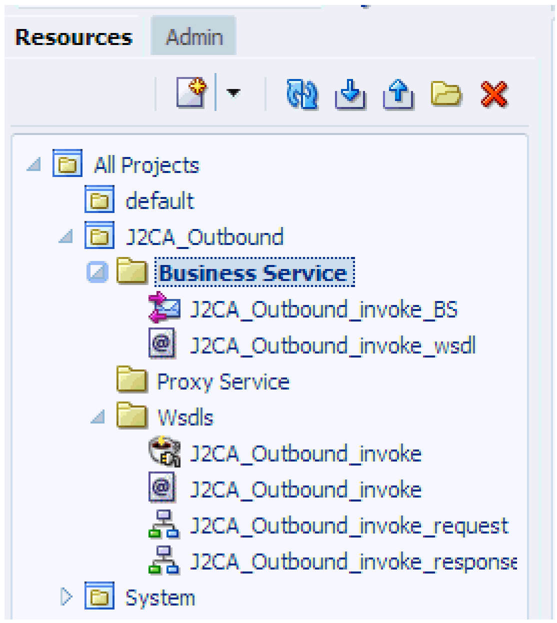
Figure 7–12 Generate WSDL and Service Window


Generate Wsdl and Service

JCA Binding Name	J2CA_Outbound_invoke
* New WSDL Name	J2CA_Outbound_invoke_wsdl
* New Business Service Name	J2CA_Outbound_invoke_BS

4. Provide a new WSDL name and a new Business Service name in the corresponding fields.
5. In the Destination area, select an available project and the sub-folder that is designated for Business Services.
6. Click **Generate**.
7. Expand **Business Service** under the project folder and check if the generated WSDL and Business Service are listed, as shown in Figure 7–13.

Figure 7–13 Business Service Folder

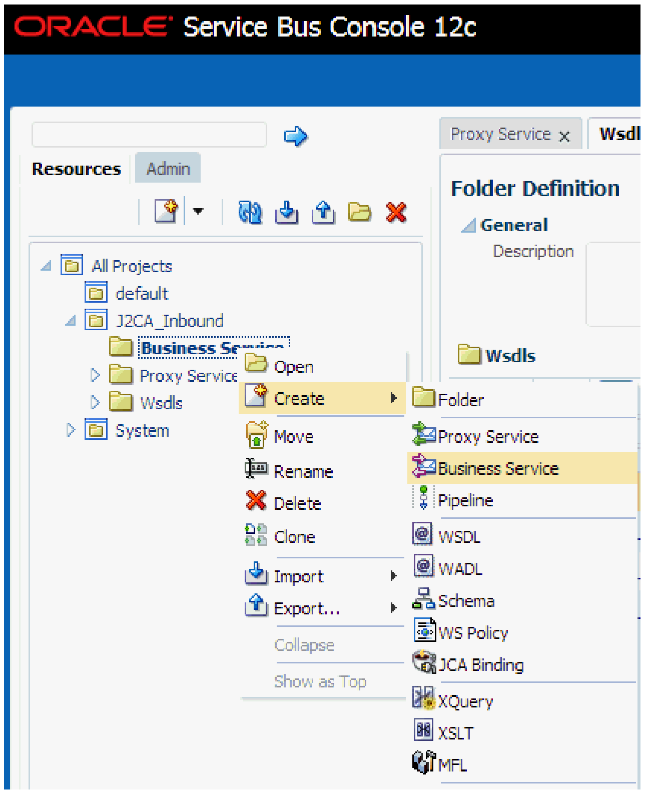


7.2.5 Configuring a File Type Business Service

Perform the following steps to configure a File type Business Service:

1. Right-click the **Business Service** folder you created in the left pane, select **Create**, and click **Business Service** as shown in Figure 7–14.

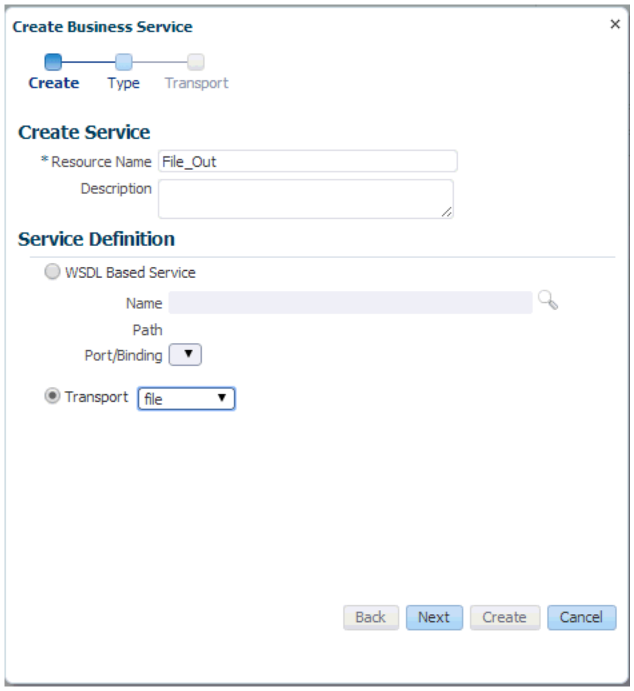
Figure 7–14 Business Service Folder



The Create Business Service window is displayed.

2. In the Resource Name field, provide a name for the Business Service, select the **File** option in the Transport section under Service Definition, and click **Next**, as shown in Figure 7–15.

Figure 7–15 Service Definition



Create Service

* Resource Name: File_Out

Description:

Service Definition

( ) WSDL Based Service

Name:

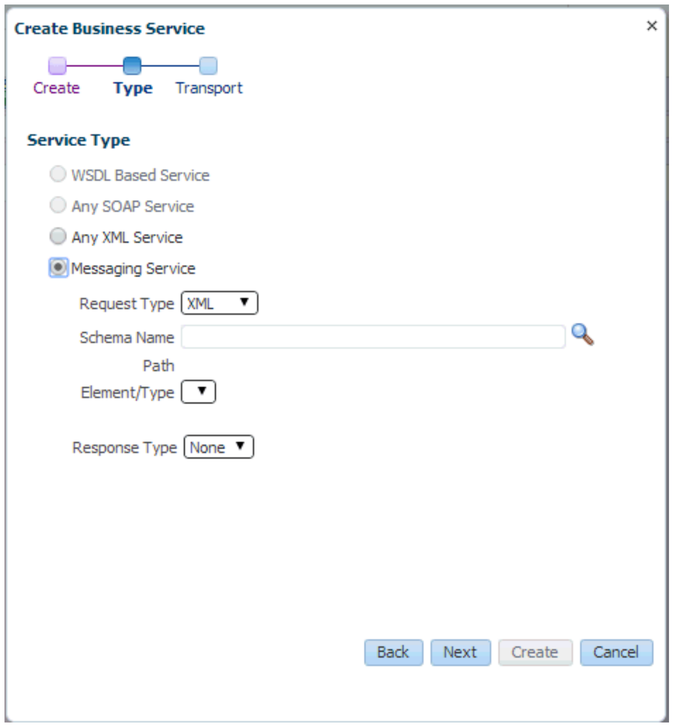
Path:

Port/Binding:

(●) Transport: file

3. In the Service Type section, select **Messaging Service**. By default, the Request Type is set to XML, and the Response Type is set to None. Then click **Next**, as shown in Figure 7–16.

Figure 7–16 Service Type Configuration Page



Service Type

( ) WSDL Based Service

( ) Any SOAP Service

( ) Any XML Service

(●) Messaging Service

Request Type: XML

Schema Name:

Path:

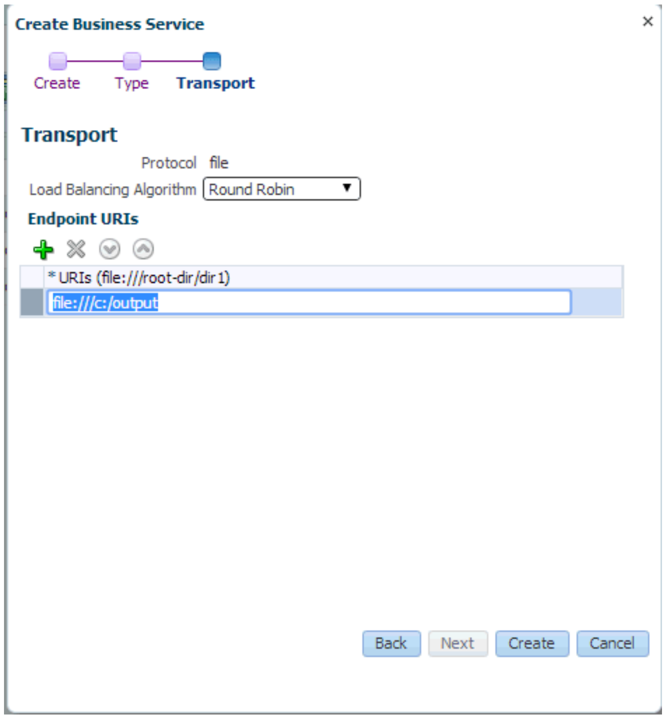
Element/Type:

Response Type: None

4. Enter the path to a destination folder on your file system in the Endpoint URI field.

5. Click **Create**, as shown in Figure 7–17.

Figure 7–17 Transport Page



Transport

Protocol file

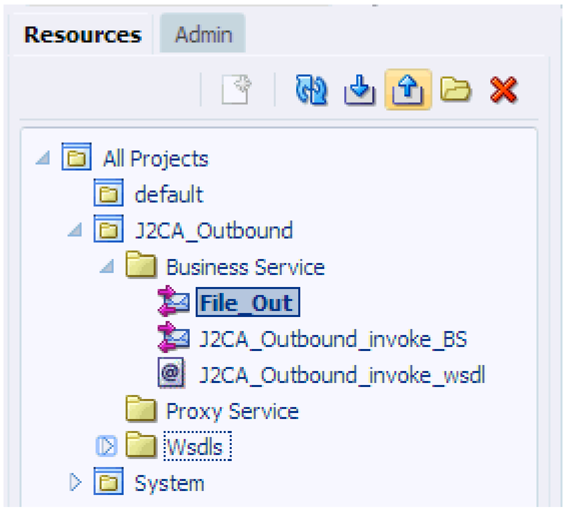
Load Balancing Algorithm Round Robin

*URIs (file:///root-dir/dir1)

file:///c:/output

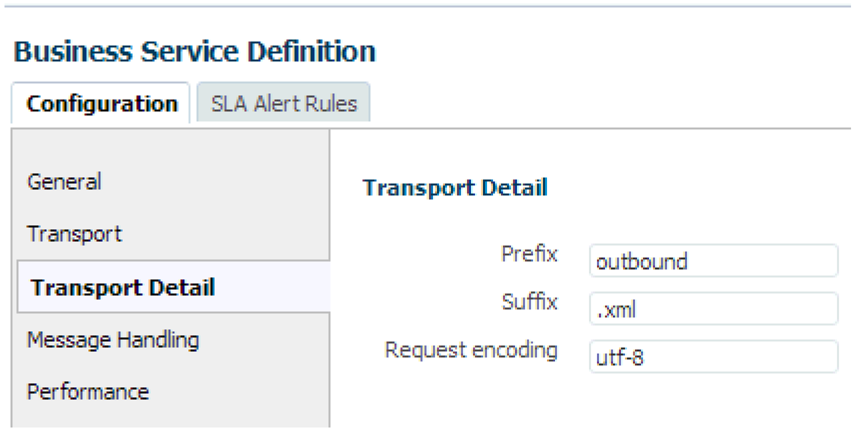
The Business Service **File_Out** is created and listed under Business Service, as shown in Figure 7–18.

Figure 7–18 File_Out Business Service



6. Double-click **File_Out**, click **Transport Detail** in the left pane, and enter the prefix and suffix for the output file to be received, as shown in Figure 7–19.

Figure 7–19 Transport Detail

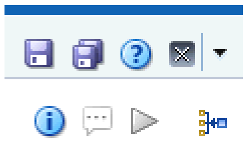


Transport Detail

Prefix: outbound
Suffix: .xml
Request encoding: utf-8

7. Click the **Save** or **Save All** icon in the right corner, as shown in Figure 7–20.

Figure 7–20 Save/Save All Icons

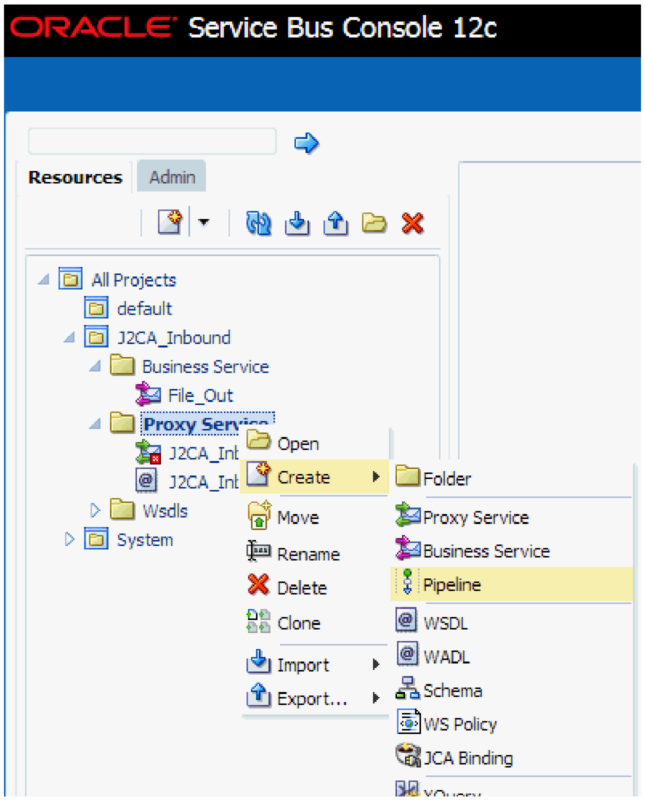


7.2.6 Configuring a Pipeline With Proxy Service

Perform the following steps to configure a Pipeline:

1. Right-click the Proxy Service folder, select **Create** and click **Pipeline**, as shown in Figure 7–21.

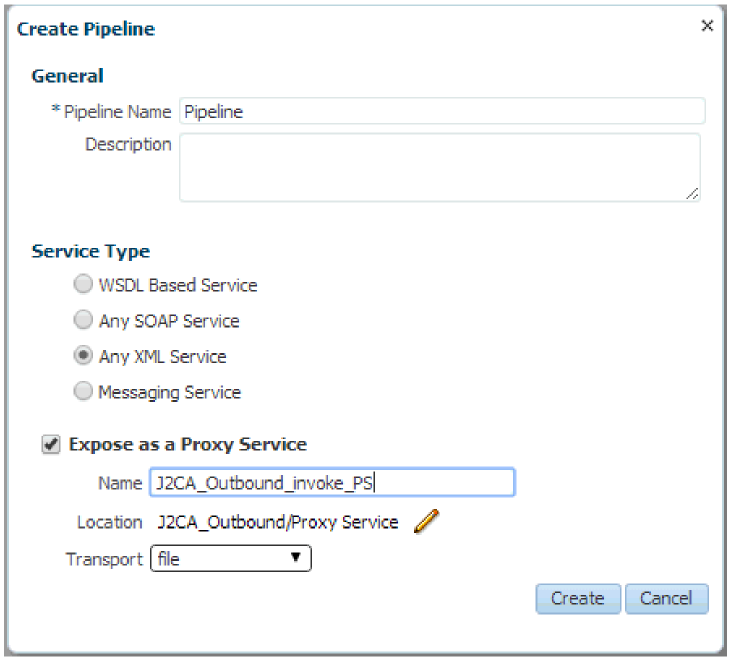
Figure 7–21 Pipeline Option



The Create Pipeline window is displayed.

2. Enter a name in the Pipeline Name field. By default, **Expose as a Proxy Service** is selected. If you wish to change the Proxy Service Name, change it and set Transport as **file**, and click **Create** as shown in Figure 7–22.

Figure 7–22 Create Pipeline Window


Create Pipeline

General

* Pipeline Name: Pipeline

Description:

Service Type

- WSDL Based Service
- Any SOAP Service
- [✓ Any XML Service]
- Messaging Service

[✓] Expose as a Proxy Service

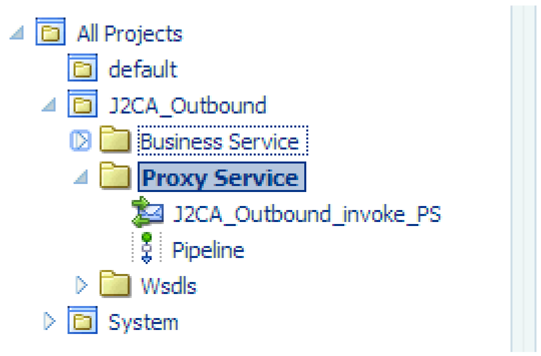
Name: J2CA_Outbound_invoke_PS

Location: J2CA_Outbound/Proxy Service

Transport: file

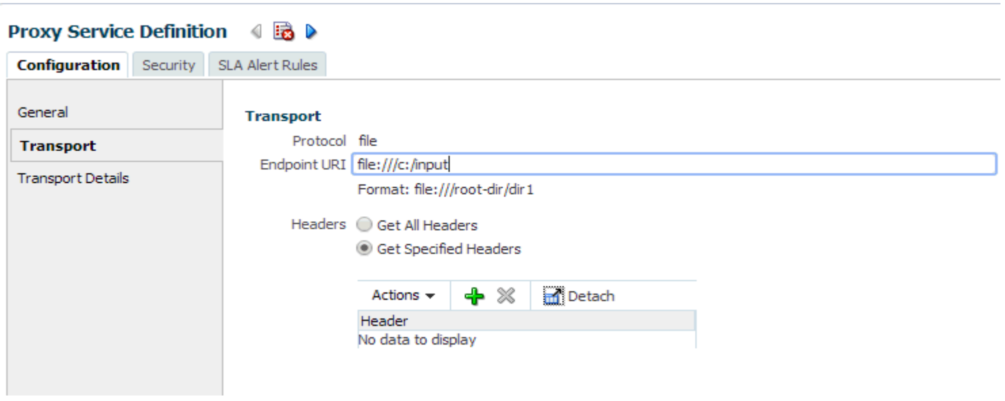
The created Pipeline and the Proxy Service is listed under Proxy Service, as shown in Figure 7–23.

Figure 7–23 Pipeline Node



3. Double-click the created proxy service and click **Transport** in the left pane. Provide the input location in the Endpoint URI field, as shown in Figure 7–24.

Figure 7–24 Transport



Transport

Protocol file

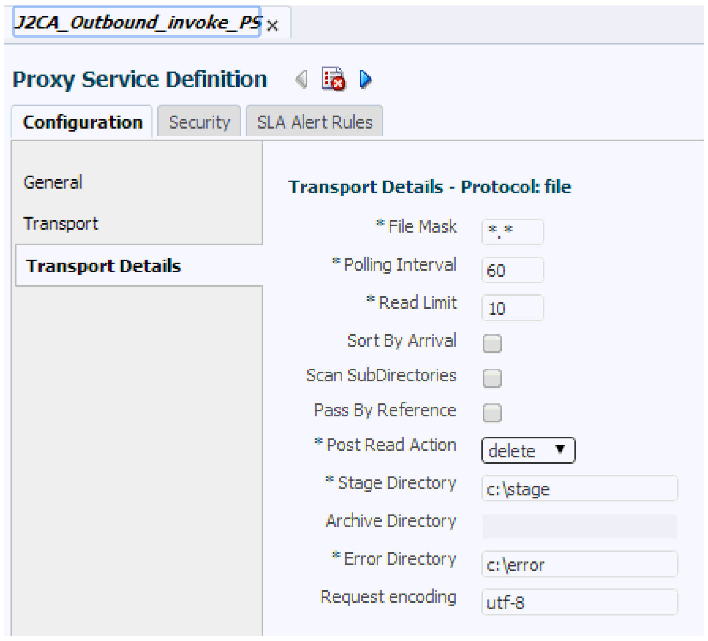
Endpoint URI file:///c:/input

Format: file:///root-dir/dir1

Headers ○ Get All Headers [✓ ● Get Specified Headers]

4. Click **Transport Details** in the left pane and provide the location for the Stage Directory and the Error Directory fields, as shown in Figure 7–25.

Figure 7–25 Transport Details


J2CA_Outbound_invoke_PS

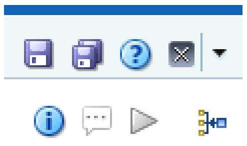
Proxy Service Definition

Transport Details - Protocol: file

* File Mask	*.*
* Polling Interval	60
* Read Limit	10
Sort By Arrival	☐
Scan SubDirectories	☐
Pass By Reference	☐
* Post Read Action	delete
* Stage Directory	c:\stage
Archive Directory	
* Error Directory	c:\error
Request encoding	utf-8

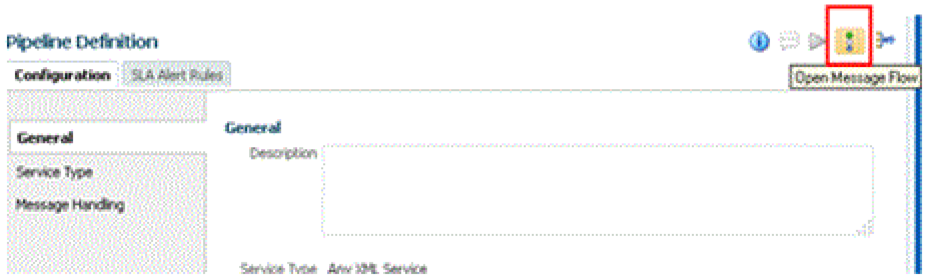
5. Click the **Save All** icon in the right corner, as shown in Figure 7–26.

Figure 7–26 Save All Icon



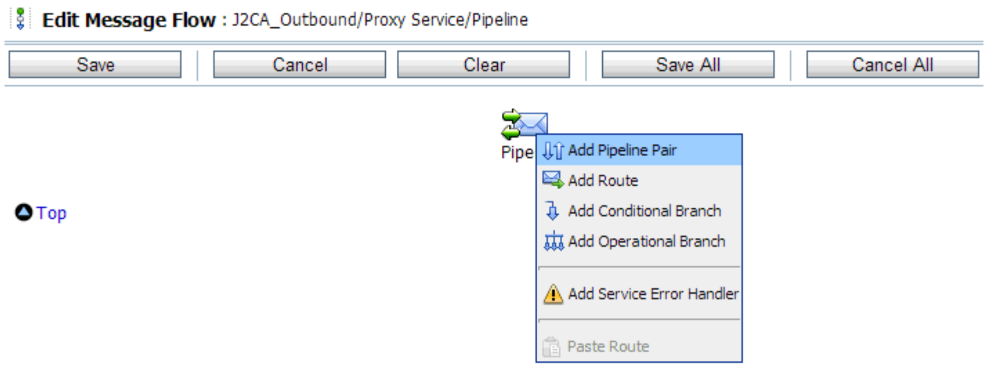
6. Double-click the **Pipeline** node and click the **Open Message Flow** icon on the right pane to open the message flow, as shown in Figure 7–27.

Figure 7–27 Open Message Flow Icon



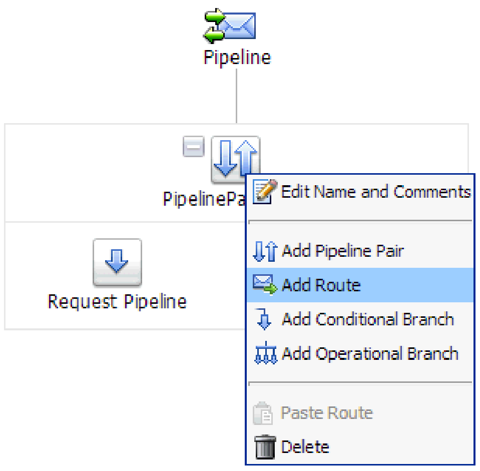
7. Click the Proxy Service icon and select **Add Pipeline Pair** from the menu, as shown in Figure 7–28.

Figure 7–28 Add Pipeline Pair Option



8. Click the **PipelinePairNode1** icon and select **Add Route** from the menu, as shown in Figure 7–29.

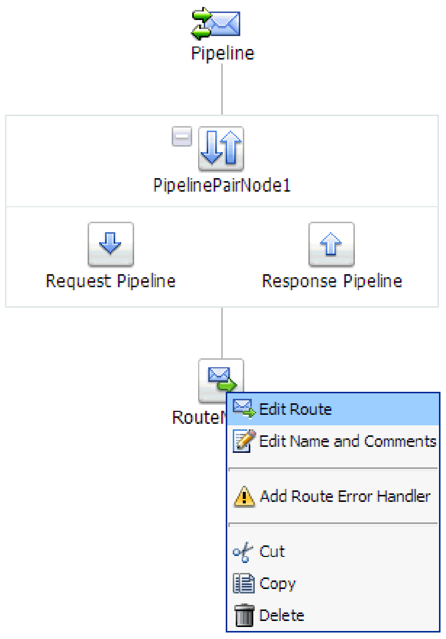
Figure 7–29 Add Route Option



The RouteNode1 icon is added below the PipelinePairNode1 icon.

9. Click the RouteNode1 icon and select **Edit Route** from the menu, as shown in Figure 7–30.

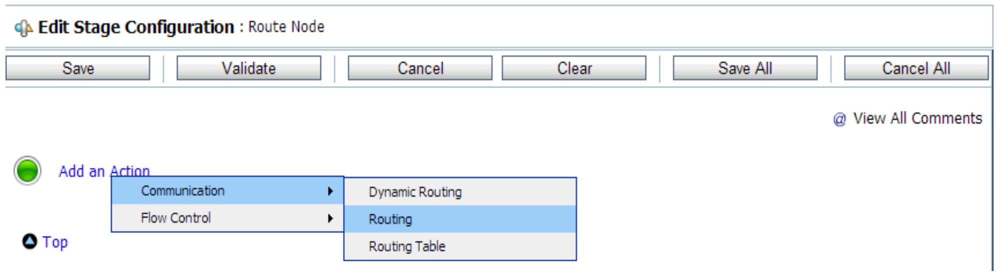
Figure 7–30 Edit Route Option



The Edit Stage Configuration workspace area is displayed.

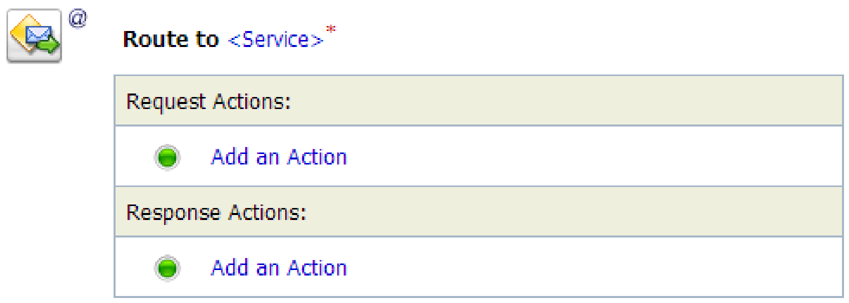
10. Click **Add an Action**, select **Communication** and click **Routing**, as shown in Figure 7–31.

Figure 7–31 Edit Stage Configuration Workspace Area



11. Click **<Service>**, as shown in Figure 7–32.

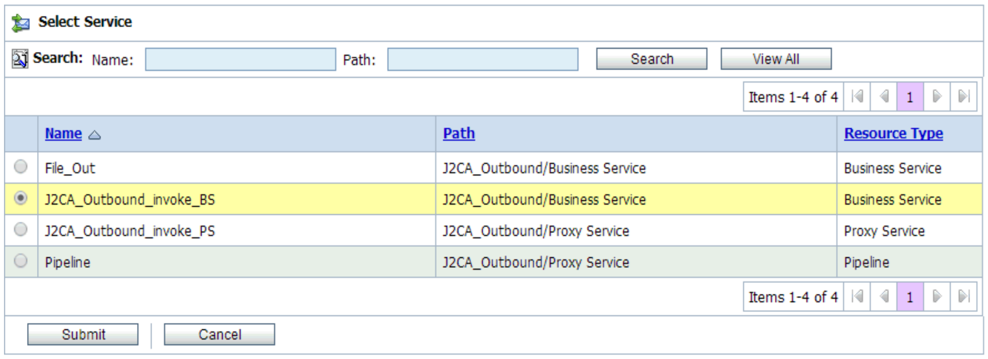
Figure 7–32 Actions



The Select Service dialog is displayed.

12. Select the WSDL type Business Service configured for J.D. Edwards OneWorld and click on **Submit**, as shown in Figure 7–33.

Figure 7–33 Select Service Dialog



Select Service

Search: Name: Path: Search View All

Items 1-4 of 4 1

	Name	Path	Resource Type
○	File_Out	J2CA_Outbound/Business Service	Business Service
●	J2CA_Outbound_invoke_BS	J2CA_Outbound/Business Service	Business Service
○	J2CA_Outbound_invoke_PS	J2CA_Outbound/Proxy Service	Proxy Service
○	Pipeline	J2CA_Outbound/Proxy Service	Pipeline

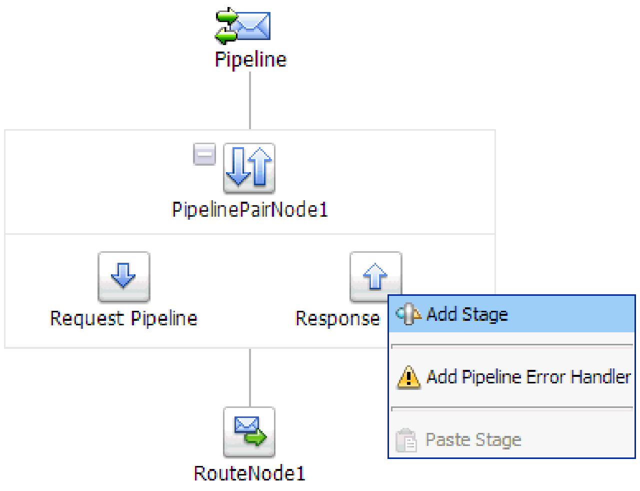
Items 1-4 of 4 1

Submit Cancel

13. Select the name of the J.D. Edwards OneWorld business object as the operational attribute from the list, and click **Save**.

14. Click the Response Pipeline icon and select **Add Stage** from the menu, as shown in Figure 7–34.

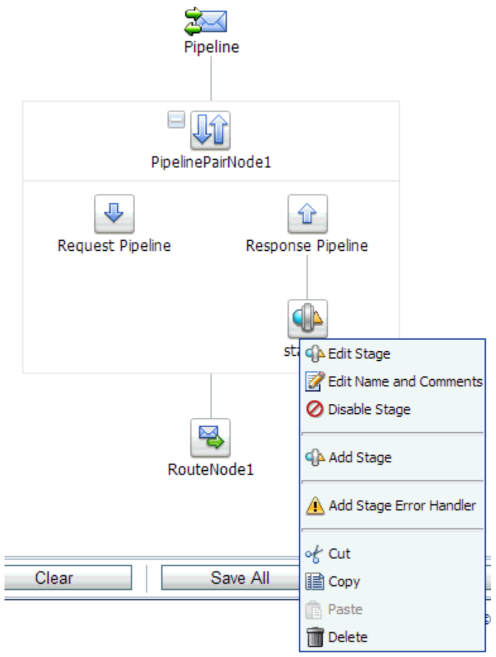
Figure 7–34 Response Pipeline Icon



The Stage1 icon is added below the Response Pipeline icon.

15. Click the Stage1 icon and select **Edit Stage** from the menu, as shown in Figure 7–35.

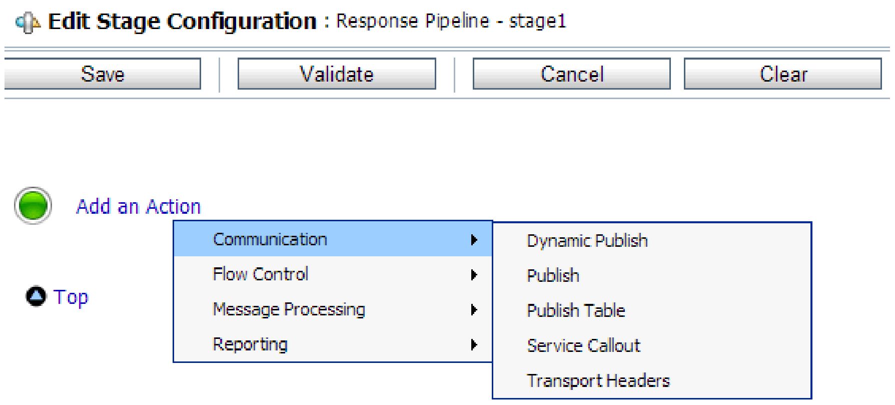
Figure 7–35 Edit Stage Option



The Edit Stage Configuration workspace area is displayed.

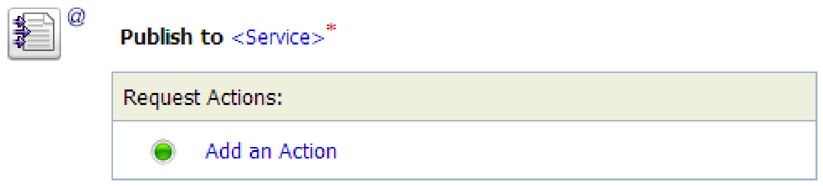
16. Click **Add an Action**, select **Communication**, and then click **Publish**, as shown in Figure 7–36.

Figure 7–36 Edit Stage Configuration Workspace Area



17. Click **<Service>**, as shown in Figure 7–37.

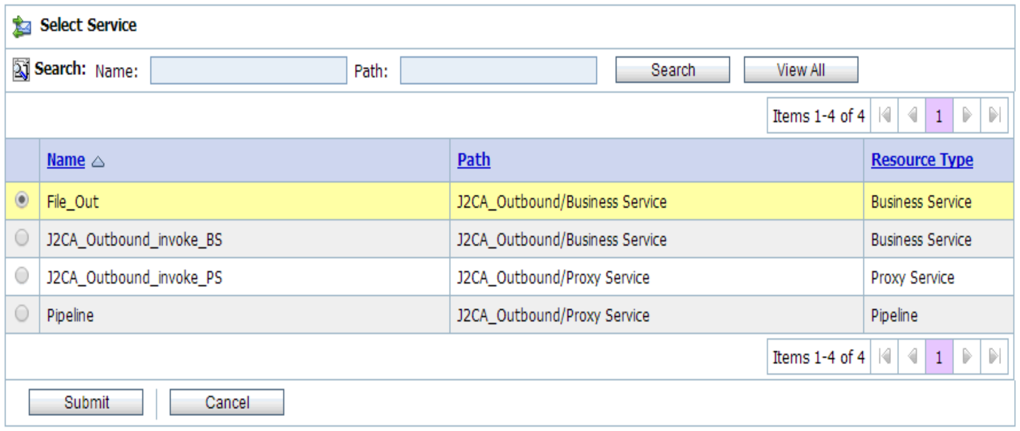
Figure 7–37 <Service> Action


Publish to <Service>*

Request Actions:
● Add an Action

18. In the Select Service dialog, select a File type Business Service and click **Submit**, as shown in Figure 7–38.

Figure 7–38 Select Service Dialog


Select Service

Search: Name: Path: Search View All

Items 1-4 of 4 1

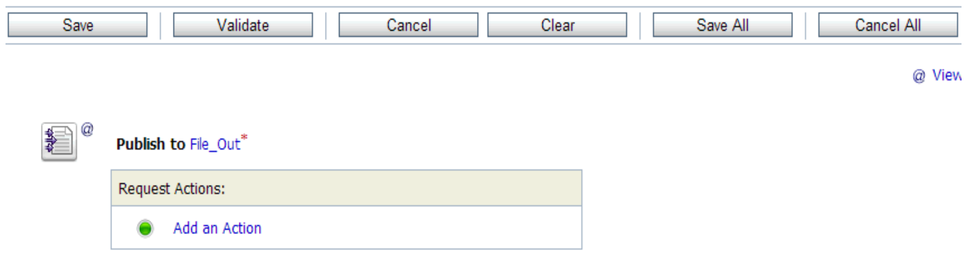
	Name	Path	Resource Type
●	File_Out	J2CA_Outbound/Business Service	Business Service
○	J2CA_Outbound_invoke_BS	J2CA_Outbound/Business Service	Business Service
○	J2CA_Outbound_invoke_PS	J2CA_Outbound/Proxy Service	Proxy Service
○	Pipeline	J2CA_Outbound/Proxy Service	Pipeline

Items 1-4 of 4 1

Submit Cancel

19. Click **Save All**, as shown in Figure 7–39.

Figure 7–39 Save All Button

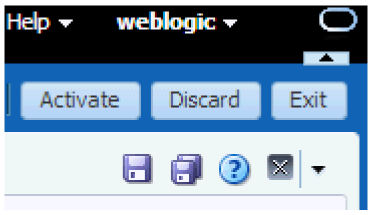


Publish to File_Out*

Request Actions:
● Add an Action

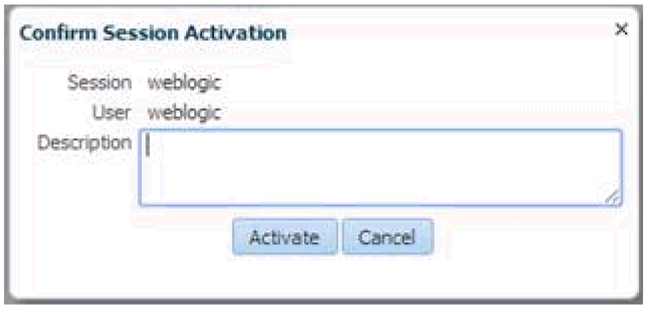
20. Click **Activate** in the right pane of the Oracle Service Bus session, as shown in Figure 7–40.

Figure 7–40 Activate Button



21. Click **Activate** to save the changes, as shown in Figure 7–41.

Figure 7–41 Confirm Session Activation



22. Copy and paste an input XML file in the input folder you have configured (for example, C:\input). Output is received in the configured output location (for example, C:\output).

7.3 Configuring an Inbound Process Using sbconsole (J2CA Configuration)

This section describes how to configure an inbound process using sbconsole for J2CA configurations.

A sample project has been provided for this inbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\OSB\J2CA\JDEdwards_Sample_J2CA_OSB_Inbound_Project
```

This section includes the following topics:

- Section 7.3.1, "Starting Oracle Service Bus and Creating Project Folders"
- Section 7.3.2, "Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus"
- Section 7.3.3, "Generating WSDL for Event Integration"
- Section 7.3.4, "Configuring a WSDL-based Proxy Service"
- Section 7.3.5, "Configuring a File Type Business Service"
- Section 7.3.6, "Configuring a Pipeline"

7.3.1 Starting Oracle Service Bus and Creating Project Folders

For more information on starting Oracle Service Bus and creating project folders, see Section 7.2.1, "Starting Oracle Service Bus and Creating Project Folders" on page 7-2.

7.3.2 Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus

For more information on setting the class path for Application Explorer to integrate with Oracle Service Bus, see Section 7.2.2, "Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus" on page 7-6.

7.3.3 Generating WSDL for Event Integration

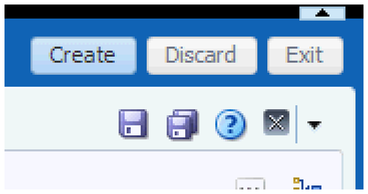
You cannot publish inbound WSDL for J.D. Edwards OneWorld event notification using Application Explorer. To generate WSDL from the command prompt, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.

7.3.4 Configuring a WSDL-based Proxy Service

Perform the following steps to select the inbound WSDL from the File system and configure a WSDL-based Proxy Service:

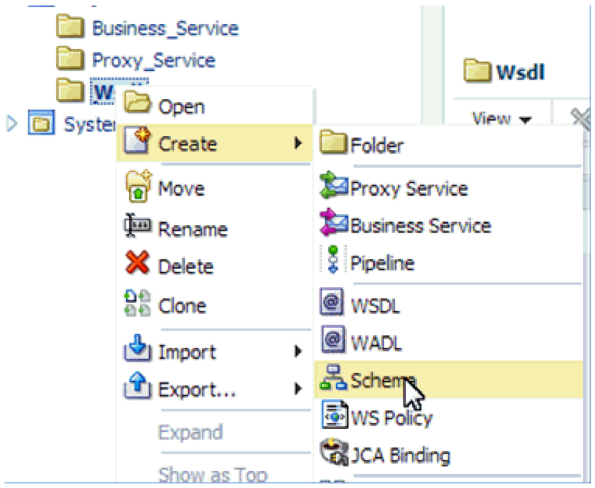
1. Open the Oracle Service Bus Console and click **Create** in the right pane of the Oracle Service Bus session, as shown in Figure 7–42.

Figure 7–42 Create Button



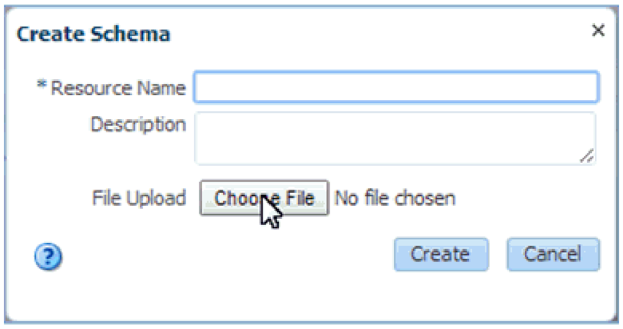
2. Right-click the WSDL folder, select **Create**, and click **Schema**, as shown in Figure 7–43.

Figure 7–43 Schema Option



3. In the displayed window, click **Choose File** and select the available schema file (for example, J2CA_Inbound_receive_request.xsd), as shown in Figure 7–44.

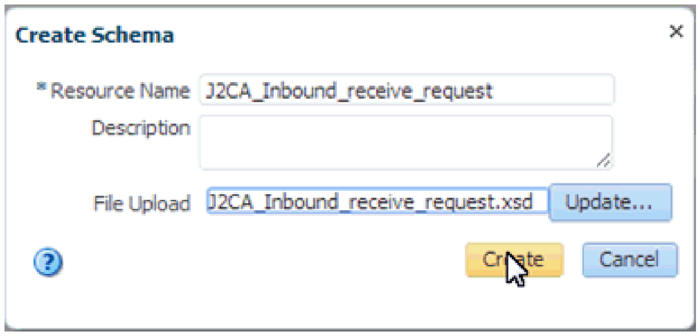
Figure 7–44 Choose File Button



The Resource Name will be added by default.

4. Click **Create**, as shown in Figure 7–45.

Figure 7–45 Create Button

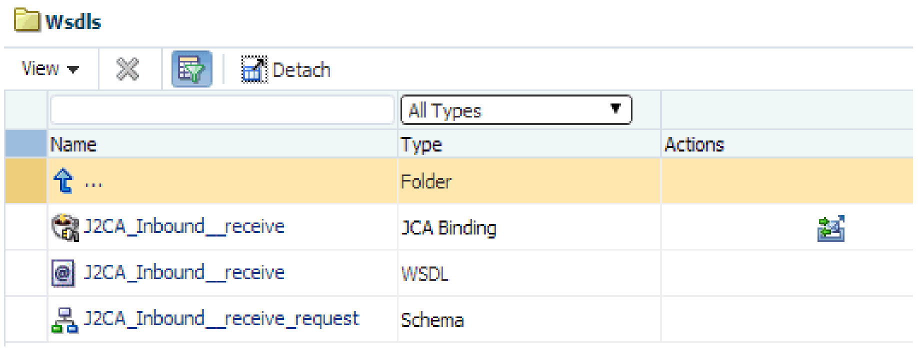


Resource Name: J2CA_Inbound_receive_request

File Upload: J2CA_Inbound_receive_request.xsd

5. Right-click the WSDL folder, select **Create**, and click **WSDL**. Repeat the steps and select the WSDL file.
6. Right-click the WSDL folder, select **Create**, and click **JCA Binding**. Repeat the steps and select the JCA file.
7. Double-click the created WSDL folder in the left pane (for example, Wsdls), and ensure that the WSDL is listed in the right pane, as shown in Figure 7–46.

Figure 7–46 Exported WSDL

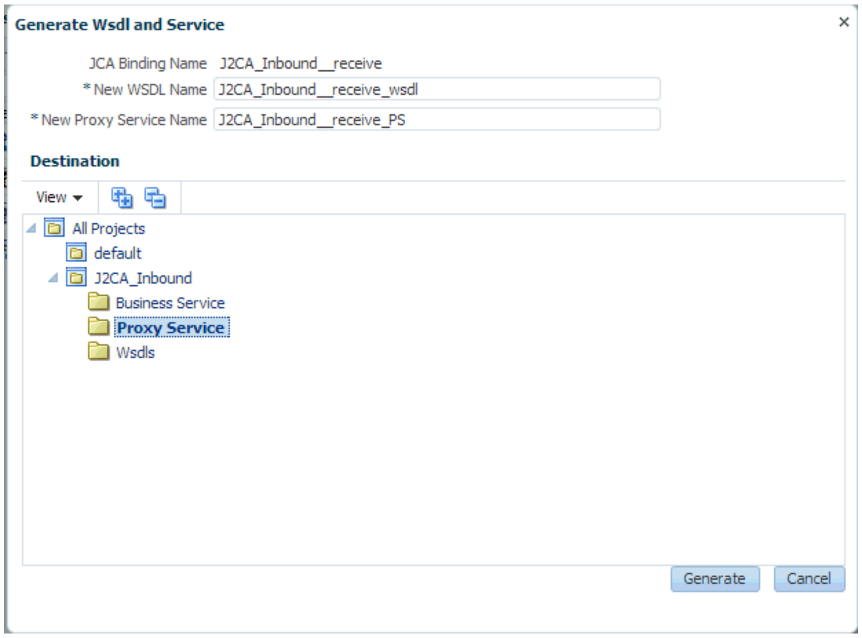


Name	Type	Actions
...	Folder	
J2CA_Inbound__receive	JCA Binding	
J2CA_Inbound__receive	WSDL	
J2CA_Inbound__receive_request	Schema	

8. Click the icon that corresponds to the JCA Binding in the Actions column.

   The Generate WSDL and Service page is displayed, as shown in Figure 7–47.

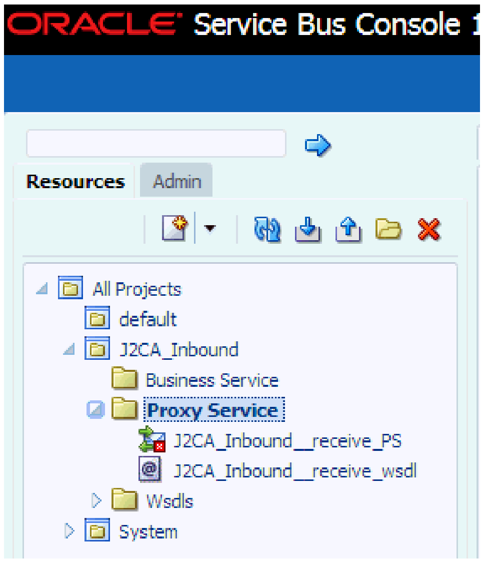
Figure 7–47 Generate WSDL and Service Page



JCA Binding Name J2CA_Inbound__receive
* New WSDL Name J2CA_Inbound__receive_wsdl
* New Proxy Service Name J2CA_Inbound__receive_PS

9. Provide a new WSDL name and a new Proxy Service name in the corresponding fields.
10. In the Destination area, select an available project and the sub-folder that is designated for Proxy Services.
11. Click **Generate**.
12. Expand **Proxy Service** under Project Explorer and check if the generated WSDL and Proxy Service are listed, as shown in Figure 7–48.

Figure 7–48 Generated WSDL

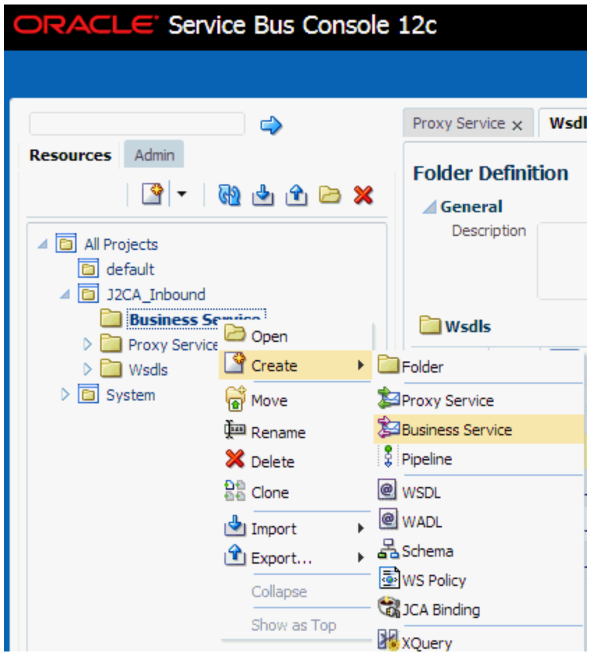


7.3.5 Configuring a File Type Business Service

Perform the following steps to configure a File type Business Service:

1. Right-click the Business Service folder you created in the left pane, select **Create**, and click **Business Service**, as shown in Figure 7–49.

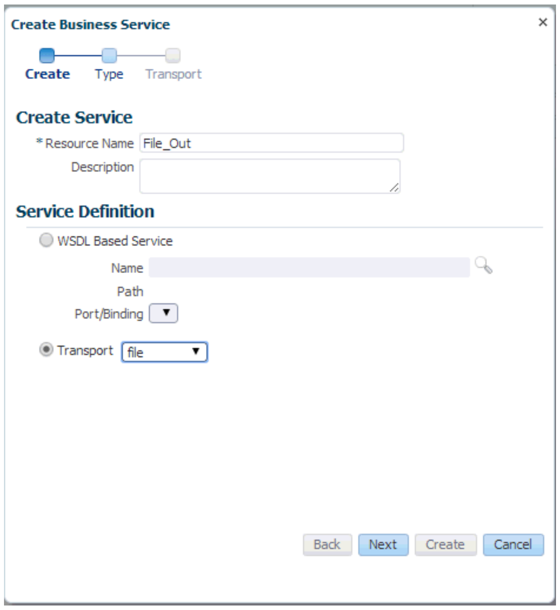
Figure 7–49 Business Service Folder



The Create Business Service window is displayed.

2. In the Resource Name field, provide a name for the Business Service and select the **File** option from the Transport drop-down list in the Service Definition area, as shown in Figure 7–50.

Figure 7–50 Create Business Service Window



Create Service

* Resource Name: File_Out

Description:

Service Definition

WSDL Based Service

Name

Path

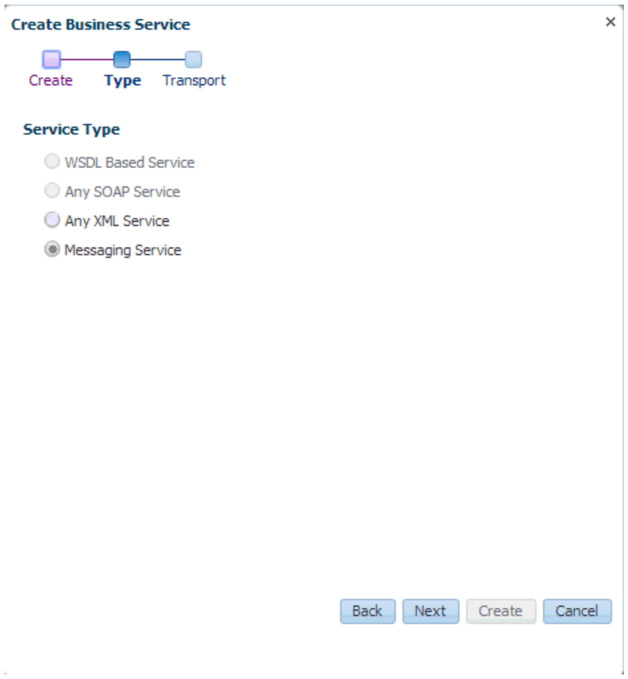
Port/Binding

● Transport: file

3. Click **Next**.

4. In the Service Type area, select **Messaging Service** as the service type, as shown in Figure 7–51.

Figure 7–51 Service Type Area



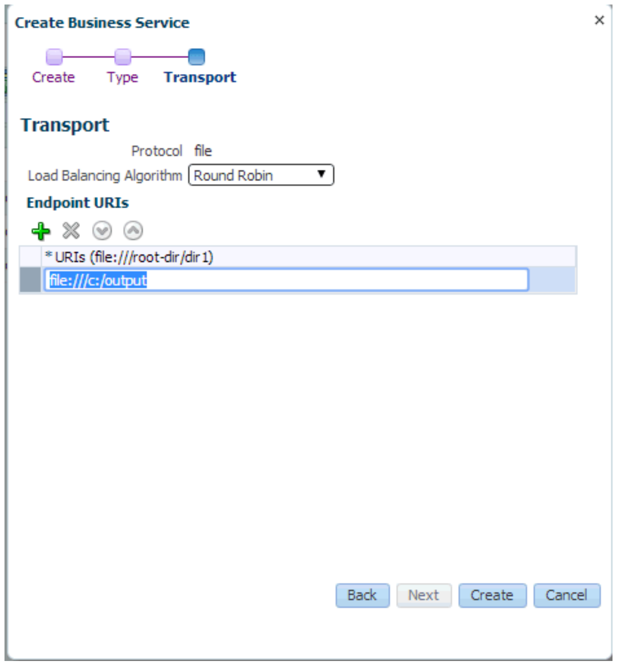
Service Type

WSDL Based Service
Any SOAP Service
Any XML Service
[✓ Messaging Service]

5. Click **Next**.

   The Transport page is displayed, as shown in Figure 7–52.

Figure 7–52 Transport Page



Transport

Protocol file

Load Balancing Algorithm Round Robin

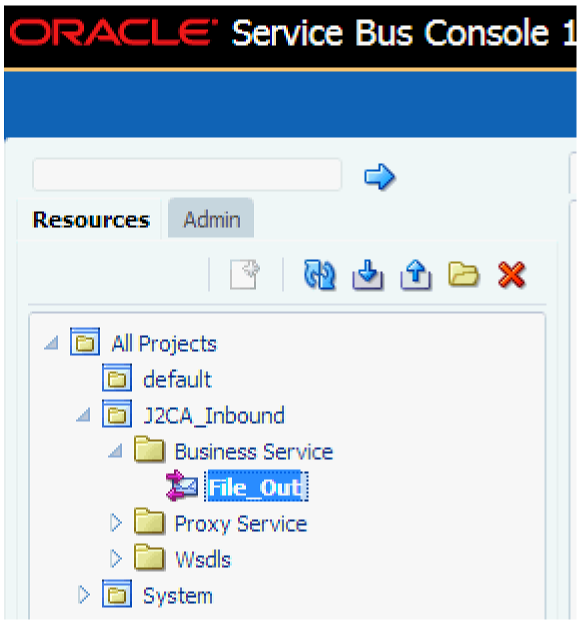
Endpoint URIs

* URIs (file:///root-dir/dir1)
file:///c:/output

6. Enter the path to a destination folder on your file system in the Endpoint URI field and click **Create**.

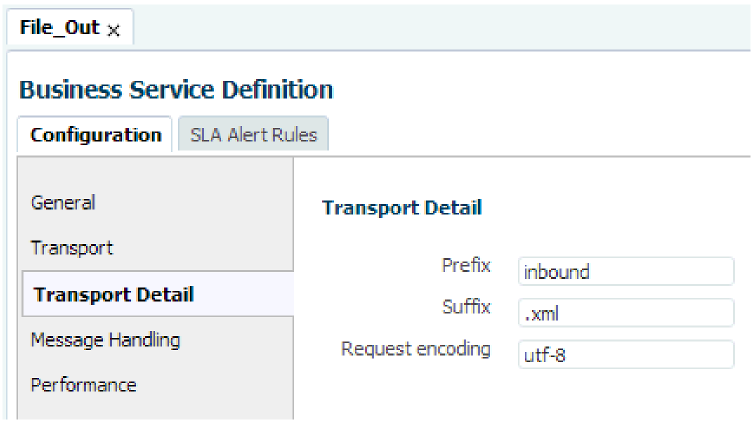
   The Business Service File_Out is created and listed under Business Service, as shown in Figure 7–53.

Figure 7–53 File_Out Business Service



7. Double-click **File_Out**, click **Transport Detail** in the left pane, and enter the prefix and suffix for the output file to be received, as shown in Figure 7–54.

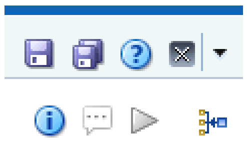
Figure 7–54 Transport Detail Page



Prefix: inbound
Suffix: .xml
Request encoding: utf-8

8. Click the Save or Save All icon in the right corner, as shown in Figure 7–55.

Figure 7–55 Save Icons

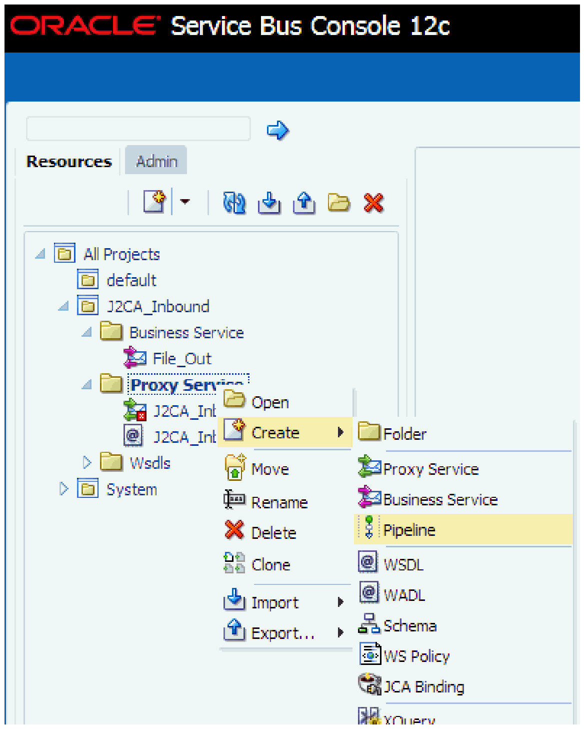


7.3.6 Configuring a Pipeline

Perform the following steps to configure a Pipeline:

1. Right-click the proxy service you created and select **Create**, and then click **Pipeline**, as shown in Figure 7–56.

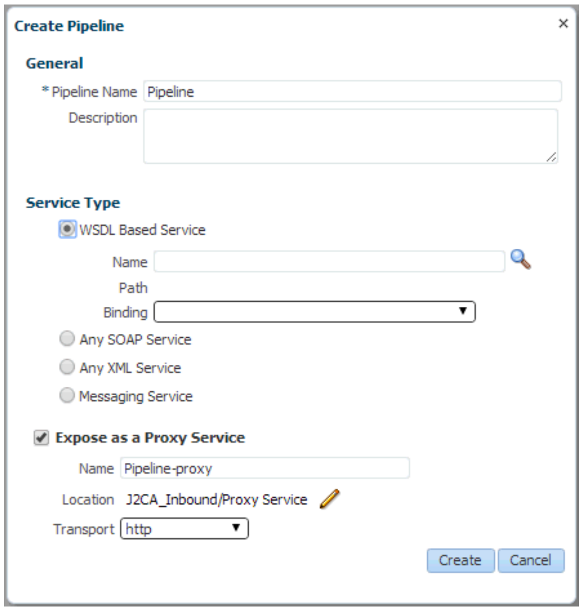
Figure 7–56 Pipeline Option



The Create Pipeline window is displayed.

2. In the Pipeline Name field, enter a name and select the Service Type as **WSDL Based Service**, as shown in Figure 7–57.

Figure 7–57 Create Pipeline Window


Create Pipeline

General

* Pipeline Name: Pipeline

Description:

Service Type

(●) WSDL Based Service

Name:

Path

Binding:

( ) Any SOAP Service

( ) Any XML Service

( ) Messaging Service

[✓] Expose as a Proxy Service

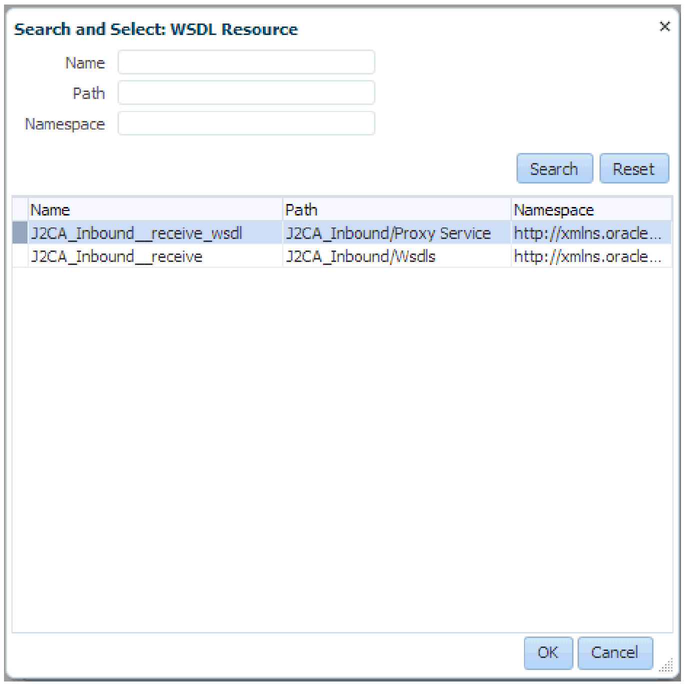
Name: Pipeline-proxy

Location: J2CA_Inbound/Proxy Service

Transport: http

3. Click the Search icon, and in the displayed Search and Select: WSDL Resource window, select **J2CA_Inbound_receive_wsdl**, and click **OK**, as shown in Figure 7–58.

Figure 7–58 Search and Select: WSDL Resource Window


Search and Select: WSDL Resource

Name:

Path:

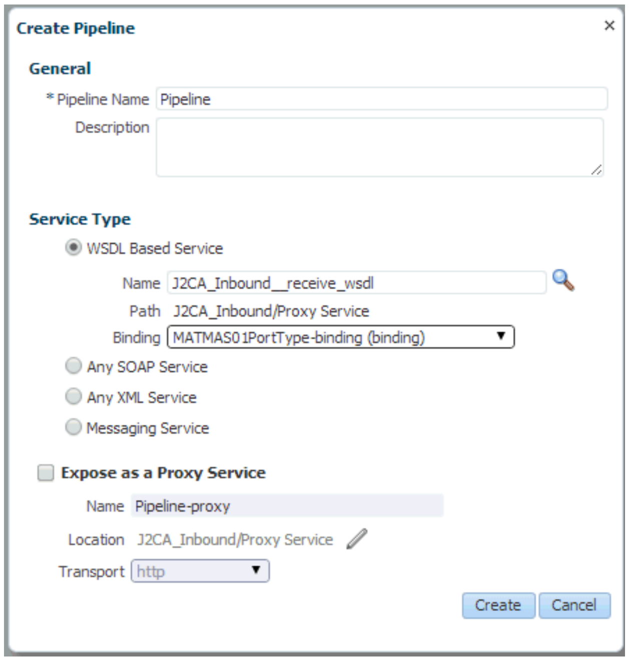
Namespace:

Name	Path	Namespace
J2CA_Inbound__receive_wsdl	J2CA_Inbound/Proxy Service	http://xmlns.oracle...
J2CA_Inbound__receive	J2CA_Inbound/Wsdls	http://xmlns.oracle...

The Create Pipeline window opens.

4. Clear the check box for **Expose as a Proxy Service**, and click **Create**, as shown in Figure 7–59.

Figure 7–59 Create Pipeline Window


Create Pipeline

General

* Pipeline Name: Pipeline

Description:

Service Type

(●) WSDL Based Service

Name: J2CA_Inbound__receive_wsdl

Path: J2CA_Inbound/Proxy Service

Binding: MATMAS01PortType-binding (binding)

( ) Any SOAP Service

( ) Any XML Service

( ) Messaging Service

☐ Expose as a Proxy Service

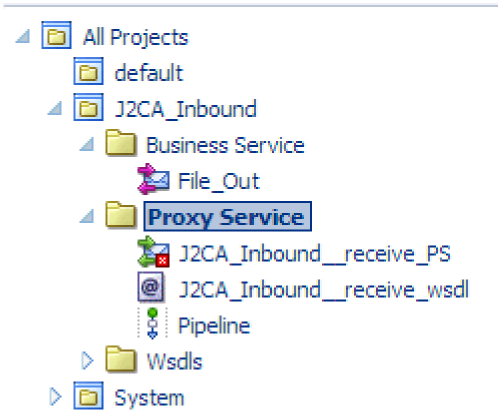
Name: Pipeline-proxy

Location: J2CA_Inbound/Proxy Service

Transport: http

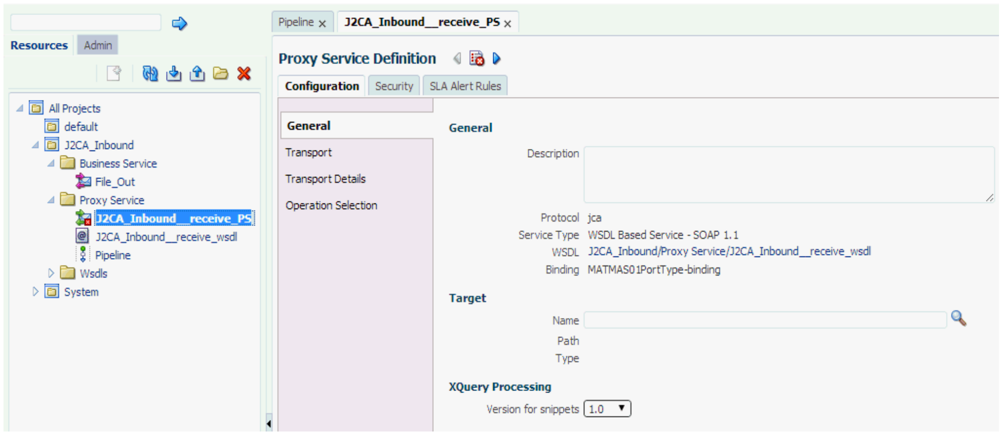
The pipeline is created and listed under Proxy Service, as shown in Figure 7–60.

Figure 7–60 Proxy Service Pipeline



5. Double-click the **J2CA_Inbound_receive_PS** node under Proxy Service in the left pane and click the **Search** icon in the Target area in right pane, as shown in Figure 7–61.

Figure 7–61 Proxy Service Definition Window



General

Description

Protocol jca
Service Type WSDL Based Service - SOAP 1.1
WSDL J2CA_Inbound/Proxy Service/J2CA_Inbound__receive_wsdl
Binding MATMAS01PortType-binding

Target

Name
Path
Type

XQuery Processing

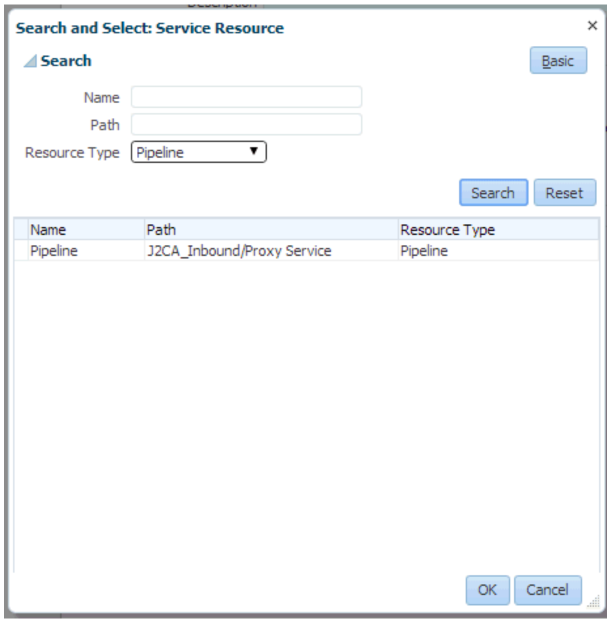
Version for snippets 1.0

The Search and Select: Service Resource window appears.

6. From the Resource Type drop-down list, select **Pipeline** and then click the **Search** button.

   The Pipeline is listed, as shown in Figure 7–62.

Figure 7–62 Search and Select: Service Resource Window



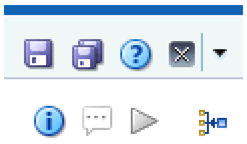
Name
Path
Resource Type Pipeline

Name	Path	Resource Type
Pipeline	J2CA_Inbound/Proxy Service	Pipeline

7. Select the Pipeline and click **OK**.

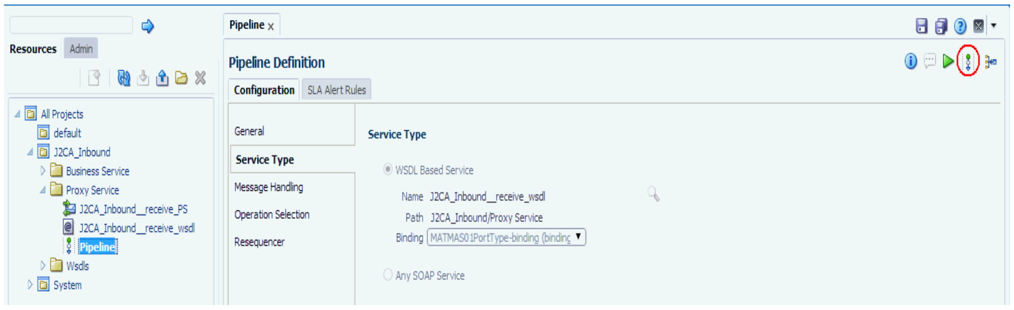
8. Click the Save or Save All icon in the right corner, as shown in Figure 7–63.

Figure 7–63 Save and Save All Icons



9. In the left pane, double-click **Pipeline** under the Proxy Service folder and click the down-pointing icon on the right pane to open the message flow, as shown in Figure 7–64.

Figure 7–64 Message Flow



Service Type

WSDL Based Service

Name J2CA_Inbound__receive_wsdl

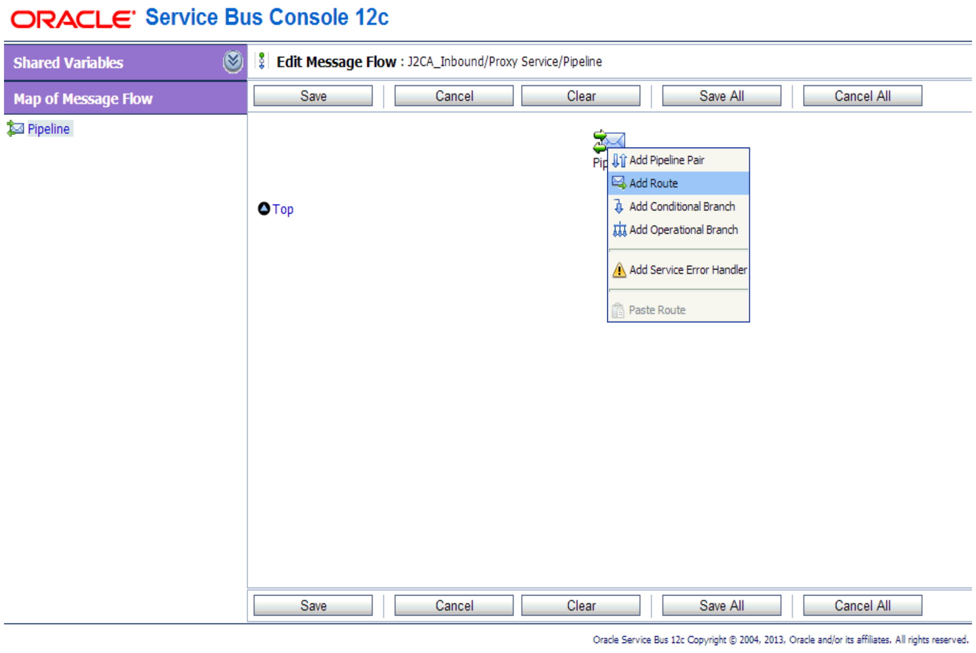
Path J2CA_Inbound/Proxy Service

Binding MATMAS01PortType-binding (binding

Any SOAP Service

10. Click the displayed Proxy service icon and select **Add Route** from the menu, as shown in Figure 7–65.

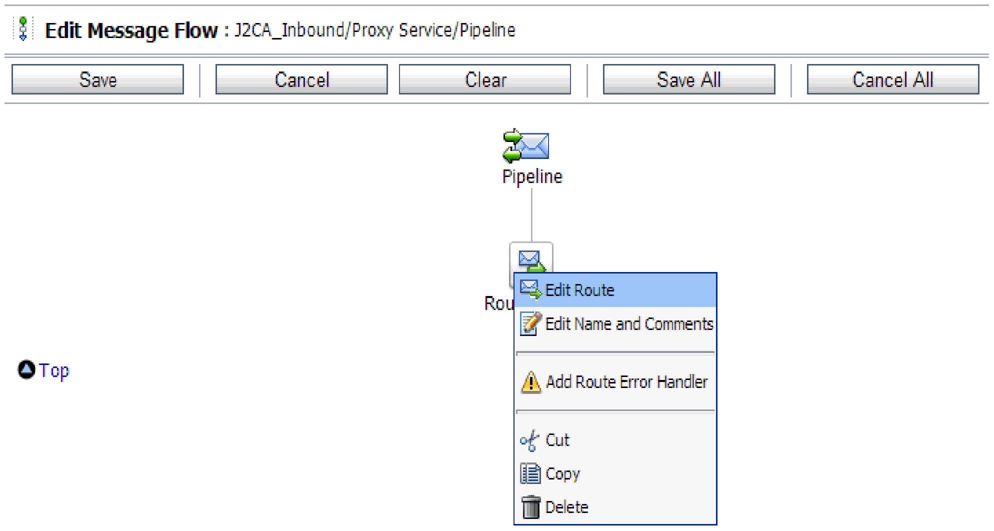
Figure 7–65 Add Route Option


ORACLE Service Bus Console 12c
Shared Variables
Map of Message Flow
Pipeline
Edit Message Flow : J2CA_Inbound/Proxy Service/Pipeline
Save
Cancel
Clear
Save All
Cancel All
Top
Pip
Add Pipeline Pair
Add Route
Add Conditional Branch
Add Operational Branch
Add Service Error Handler
Paste Route
Oracle Service Bus 12c Copyright © 2004, 2013, Oracle and/or its affiliates. All rights reserved.

The RouteNode1 icon is added.

11. Click the RouteNode1 icon and select **Edit Route** from the menu, as shown in Figure 7–66.

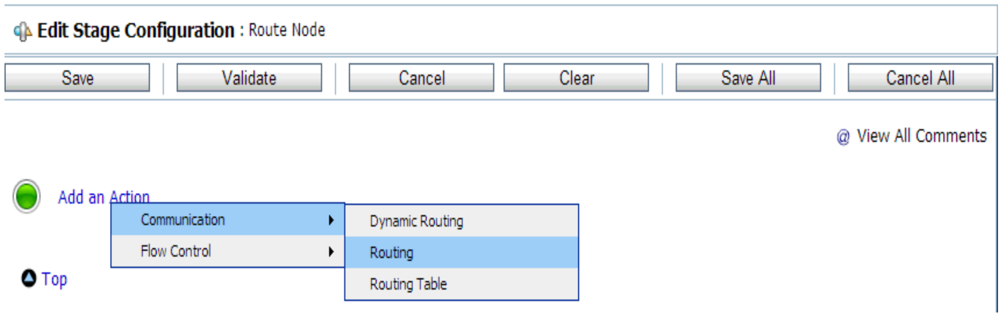
Figure 7–66 Edit Route Option



The Edit Stage Configuration workspace area is displayed.

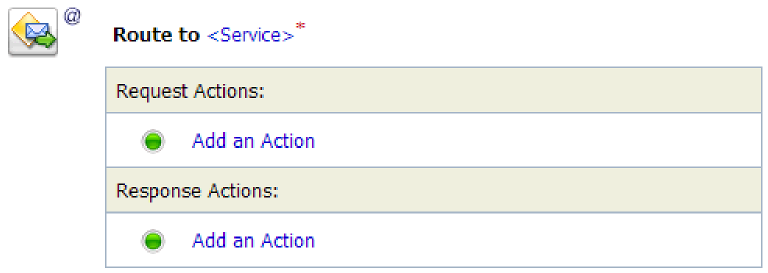
12. Click **Add an Action**, select **Communication** from the menu, and then click **Routing**, as shown in Figure 7–67.

Figure 7–67 Edit Stage Configuration Workspace



13. Click **<Service>**, as shown in Figure 7–68.

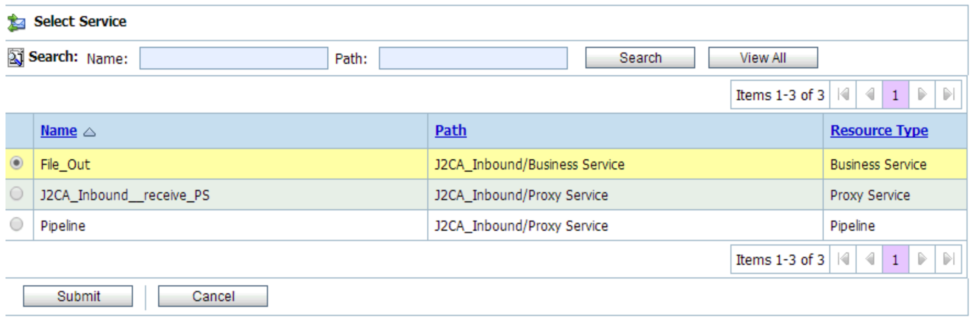
Figure 7–68 Service Route Actions



The Select Service dialog is displayed.

14. Select the **File_Out** Business service and click **Submit** as shown in Figure 7–69.

Figure 7–69 Select Service Dialog



Select Service

Search: Name: Path: Search View All

Items 1-3 of 3

	Name	Path	Resource Type
●	File_Out	J2CA_Inbound/Business Service	Business Service
○	J2CA_Inbound__receive_PS	J2CA_Inbound/Proxy Service	Proxy Service
○	Pipeline	J2CA_Inbound/Proxy Service	Pipeline

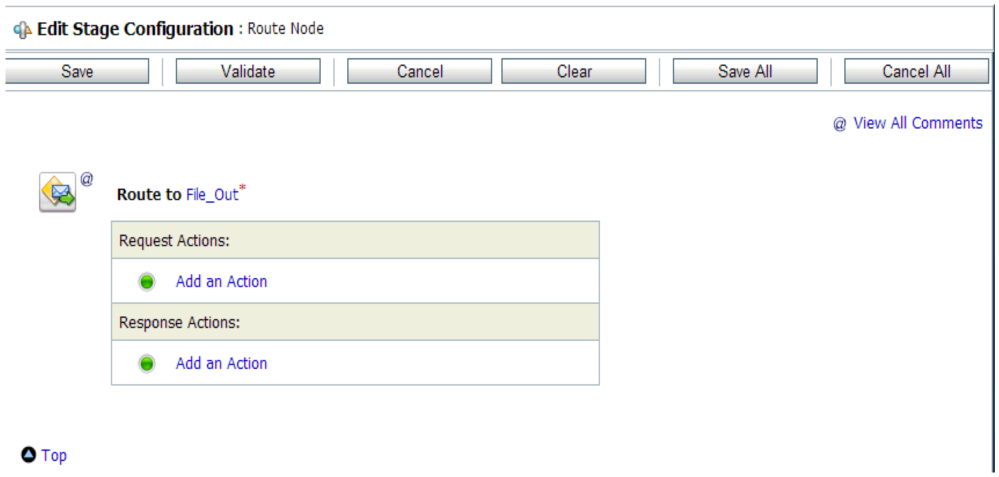
Items 1-3 of 3

Submit Cancel

You are returned to the Edit Stage Configuration workspace area.

15. Click Save All, as shown inFigure 7–70.

Figure 7–70 Edit Stage Configuration Workspace Area



Edit Stage Configuration : Route Node

Save Validate Cancel Clear Save All Cancel All

View All Comments

Route to File_Out*

Request Actions:

Add an Action

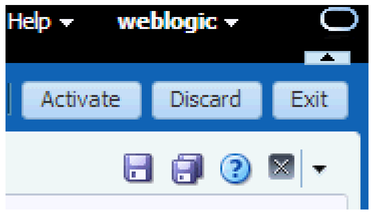
Response Actions:

Add an Action

Top

16. Click **Activate** in the right pane of the Oracle Service Bus session, as shown in Figure 7–71.

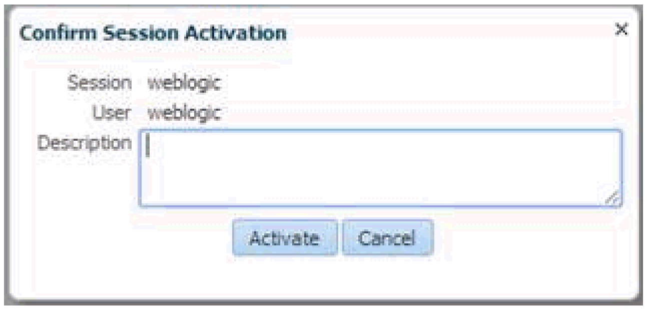
Figure 7–71 Activate Button



The Confirm Session Activation window appears.

17. Click **Activate** to save the changes, as shown in Figure 7–72.

Figure 7–72 Confirm Session Activation Window



18. Trigger an event from the J.D. Edwards OneWorld system and check if the output is received in the configured output location.

    For more information on triggering an event, see Section 4.5.5, "Triggering an Event in J.D. Edwards OneWorld" on page 4-47.

7.4 Configuring an Outbound Process Using Sbconsole (BSE Configuration)

This section describes how to configure an outbound process using sbconsole for BSE configurations.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\OSB\BSE\JDEdwards_Sample_BSE_OSB_Outbound_Project
```

This section includes the following topics:

- Section 7.4.1, "Starting Oracle Service Bus and Creating Project Folders"
- Section 7.4.2, "Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus"
- Section 7.4.3, "Publishing a WSDL From Application Explorer to Oracle Service Bus"
- Section 7.4.4, "Configuring a File Type Business Service"
- Section 7.4.5, "Configuring a WSDL-based Business Service"
- Section 7.4.6, "Configuring a Pipeline With Proxy Service"

7.4.1 Starting Oracle Service Bus and Creating Project Folders

For more information on starting Oracle Service Bus and creating project folders, see Section 7.2.1, "Starting Oracle Service Bus and Creating Project Folders" on page 7-2.

7.4.2 Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus

For more information on setting the class path for Application Explorer to integrate with Oracle Service Bus, see Section 7.2.2, "Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus" on page 7-6.

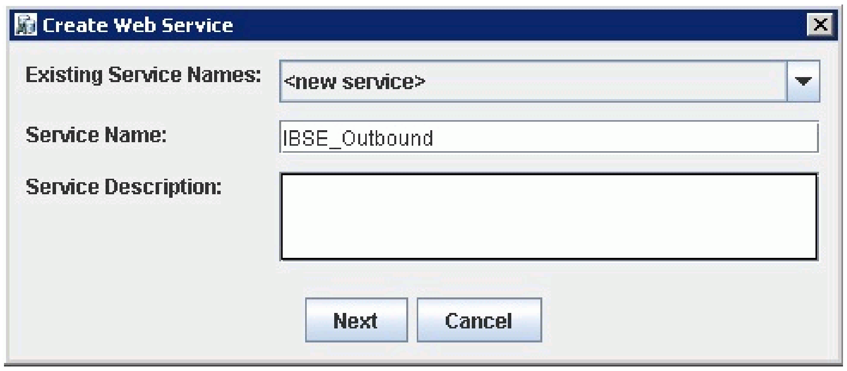
7.4.3 Publishing a WSDL From Application Explorer to Oracle Service Bus

This section describes how to publish a WSDL from Application Explorer (BSE configuration) to Oracle Service Bus.

1. Start Application Explorer, connect to a BSE configuration, and connect to a J.D. Edwards OneWorld target.
2. Expand **Services**, **CALLBSFN**, and then **AddressBook** business object.
3. Right-click **GetEffectiveAddress** and select **Create Web Service** from the menu.

   The Create Web Service dialog is displayed, as shown in Figure 7–73.

Figure 7–73 Create Web Service Dialog



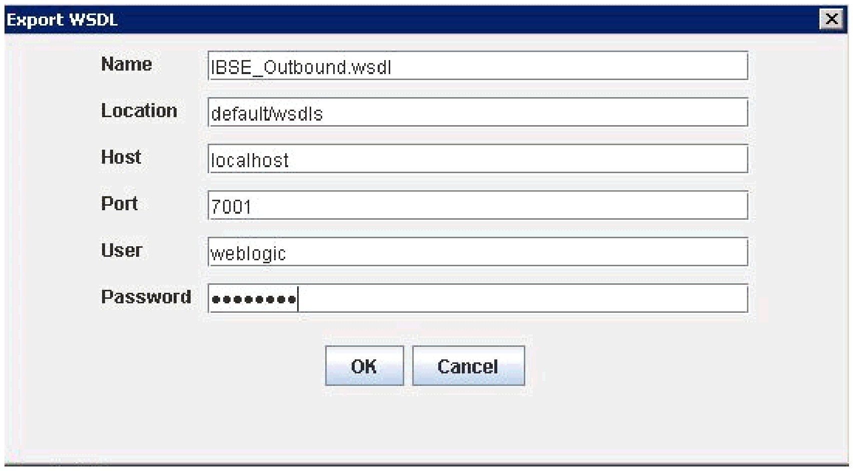
Existing Service Names: <new service>
Service Name: IBSE_Outbound
Service Description:

4. Enter a service name and click **Next**.
5. Click **OK** on the next dialog that is displayed.

   Application Explorer switches the view to the Business Services node, and the new Web service appears in the left pane.
6. Right-click the new Web service and select **Export WSDL** from the menu.

   The Export WSDL dialog is displayed, as shown in Figure 7–74.

Figure 7–74 Export WSDL Dialog



Name: IBSE_Outbound.wsdl
Location: default\wsdls
Host: localhost
Port: 7001
User: weblogic
Password: ••••••••

7. In the Name field, a default file name for the WSDL file is provided. You can accept the default or provide your own.

8. In the Location field, enter the location where you want to publish the WSDL document.

   The location is composed of an Oracle Service Bus project name and optionally, one or more folder names. The project name and any folder names must be separated by a forward slash character "/".

9. In the Host field, enter the name of the machine where Oracle WebLogic Server is running.

10. In the Port field, enter the port for the domain you are using.

11. In the User field, enter your username to access Oracle Service Bus.

12. In the Password field, enter your password to access Oracle Service Bus.

13. Click **OK**.

    The WSDL is published to the location specified in the Export WSDL dialog and is now available for use with a Business Service or Proxy Service in Oracle Service Bus.

7.4.4 Configuring a File Type Business Service

For more information on configuring a file type business service, see Section 7.2.5, "Configuring a File Type Business Service" on page 7-9.

7.4.5 Configuring a WSDL-based Business Service

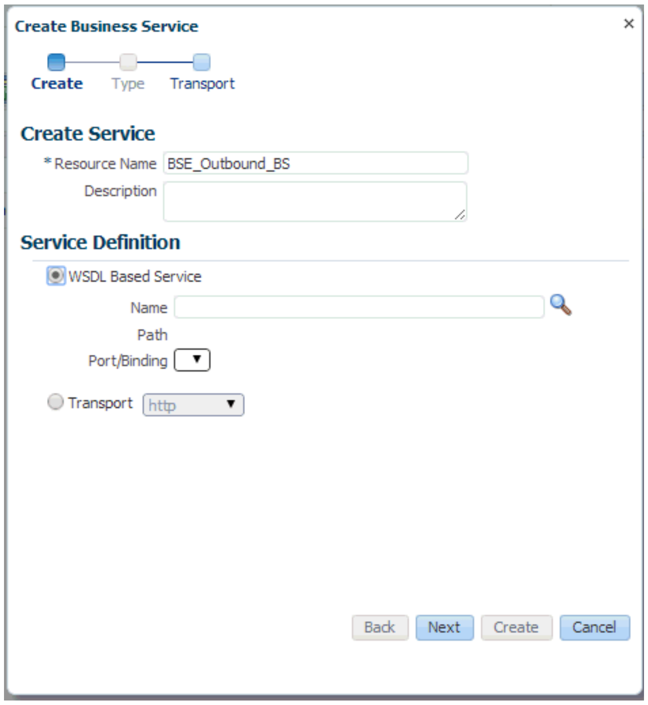
This section describes how to configure a WSDL type Business Service using the Oracle Service Bus Console.

Perform the following steps to configure a WSDL-based Proxy Service:

1. Right-click on the Business Service folder in the left pane and select **Business Service**.

   The Create Business Service window is displayed, as shown in Figure 7–75.

Figure 7–75 Create Business Service Window


Create Business Service

Create Service

* Resource Name: BSE_Outbound_BS

Description:

Service Definition

(●) WSDL Based Service

Name:

Path:

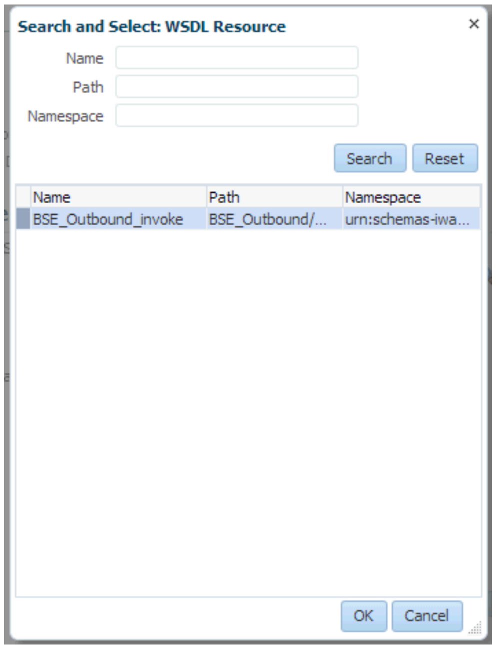
Port/Binding:

( ) Transport: http

2. Provide a name for the Business Service, and in Service Definition area, select the WSDL Based Service option and click the search icon.

   The Search and Select: WSDL Resource window is displayed, as shown in Figure 7–76.

Figure 7–76 Search and Select: WSDL Resource Window


Search and Select: WSDL Resource

Name:

Path:

Namespace:

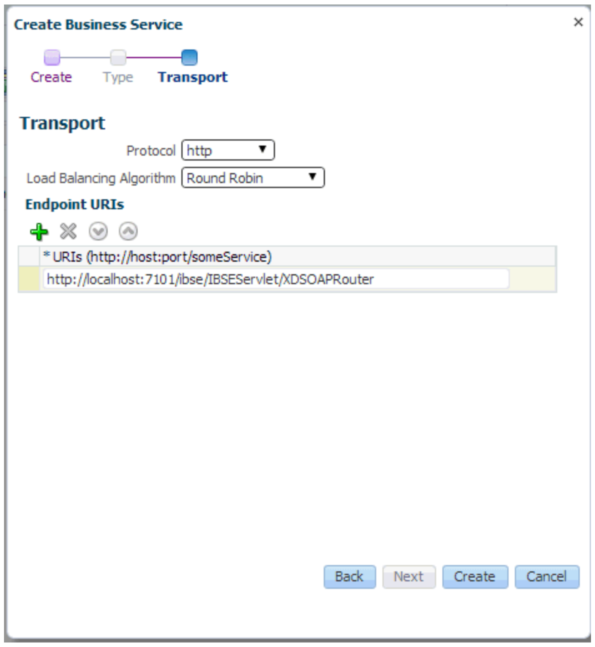
Name	Path	Namespace
BSE_Outbound_invoke	BSE_Outbound/...	urn:schemas-iwa...

3. Click the **Search** button, select the BSE Outbound WSDL, and click **OK**.

   You are returned to the Create Business Service window.

4. Click **Next**.
5. Accept the default values and click the **Create** button, as shown in Figure 7–77.

Figure 7–77 Create Business Service Window



Transport

Protocol: http

Load Balancing Algorithm: Round Robin

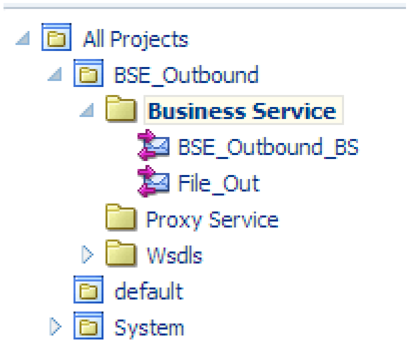
Endpoint URIs

* URIs (http://host:port/someService)

http://localhost:7101/ibse/IBSEServlet/XDSOAPRouter

The created WSDL-based Business Service is listed under the Business Service folder, as shown in Figure 7–78.

Figure 7–78 WSDL-based Business Service

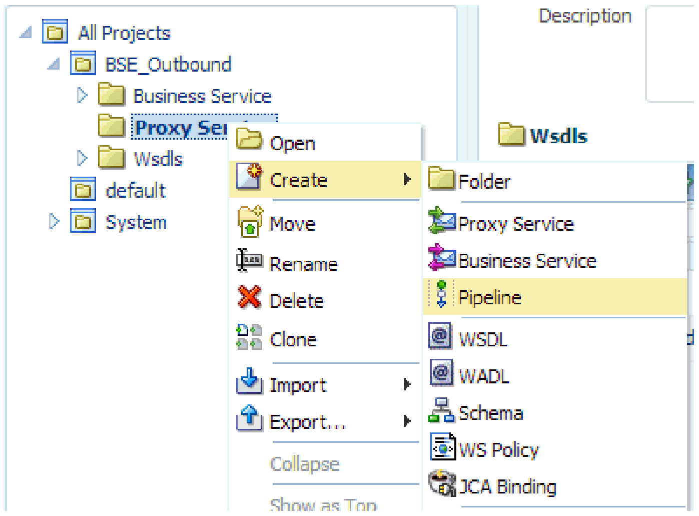


7.4.6 Configuring a Pipeline With Proxy Service

This section describes how to configure a Proxy Service using the Oracle Service Bus Console.

1. Right-click the Proxy Service folder, select **Create** and click **Pipeline**, as shown in Figure 7–79.

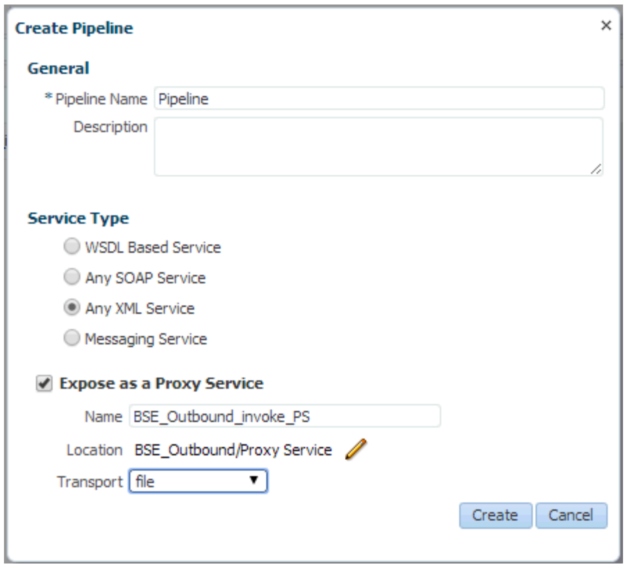
Figure 7–79 Pipeline Option



The Create Pipeline window is displayed.

2. Enter a name in the Pipeline Name field. By default, **Expose as a Proxy Service** is selected. If you wish to change the Proxy Service Name, change it and set Transport to **file**, and click **Create** as shown in Figure 7–80.

Figure 7–80 Create Pipeline Window


Create Pipeline

General

* Pipeline Name: Pipeline

Description:

Service Type

- WSDL Based Service
- Any SOAP Service
- [✓ Any XML Service]
- Messaging Service

[✓] **Expose as a Proxy Service**

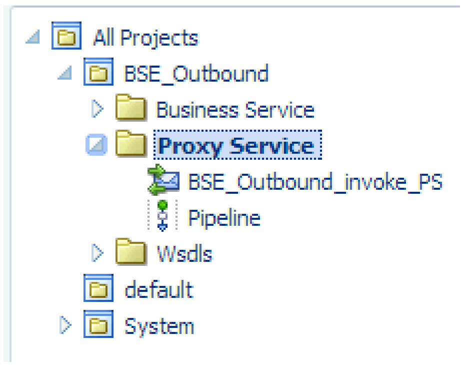
Name: BSE_Outbound_invoke_PS

Location: BSE_Outbound/Proxy Service

Transport: file

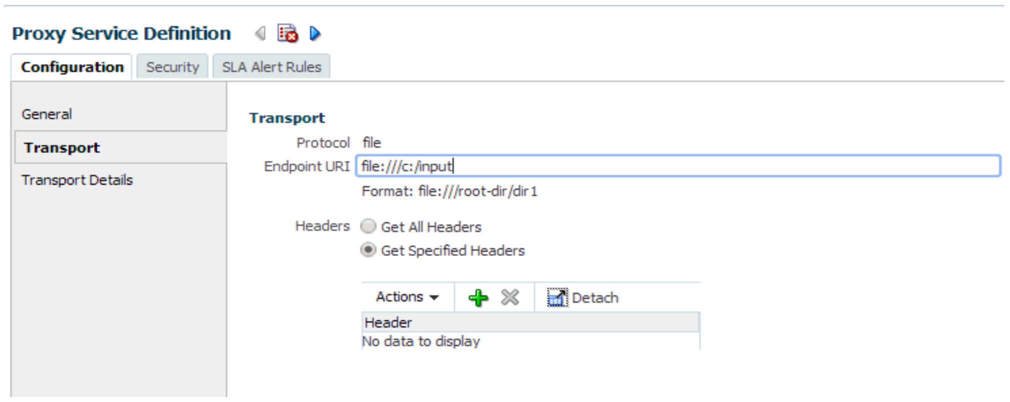
The created Pipeline and the Proxy Service is listed under Proxy Service, as shown in Figure 7–81.

Figure 7–81 Pipeline Node



3. Double-click the created proxy service and click **Transport** in the left pane. Provide the input location in the Endpoint URI field, as shown in Figure 7–82.

Figure 7–82 Transport



Transport

Protocol file

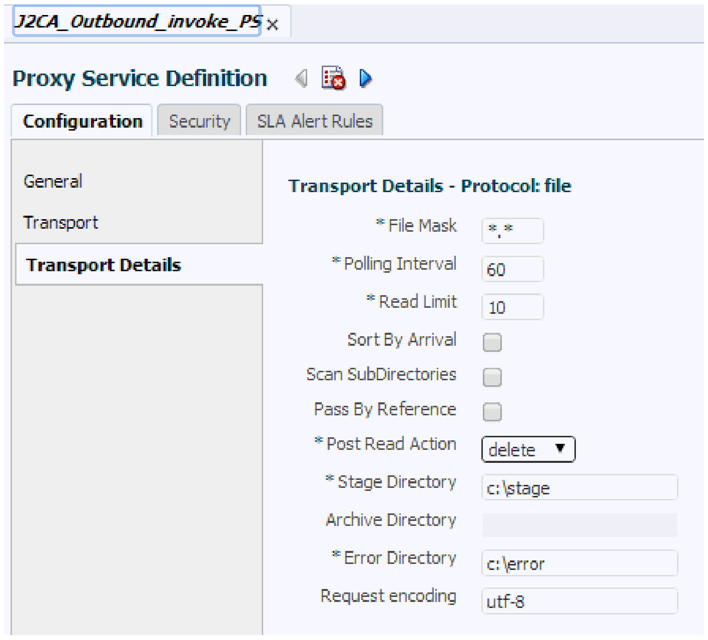
Endpoint URI file:///c:/input

Format: file:///root-dir/dir1

Headers ( ) Get All Headers [✓ Get Specified Headers]

4. Click **Transport Details** in the left pane and provide the location for the Stage Directory and the Error Directory fields, as shown in Figure 7–83.

Figure 7–83 Transport Details


J2CA_Outbound_invoke_PS

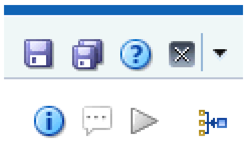
Proxy Service Definition

Transport Details - Protocol: file

* File Mask	*.*
* Polling Interval	60
* Read Limit	10
Sort By Arrival	☐
Scan SubDirectories	☐
Pass By Reference	☐
* Post Read Action	delete
* Stage Directory	c:\stage
Archive Directory	
* Error Directory	c:\error
Request encoding	utf-8

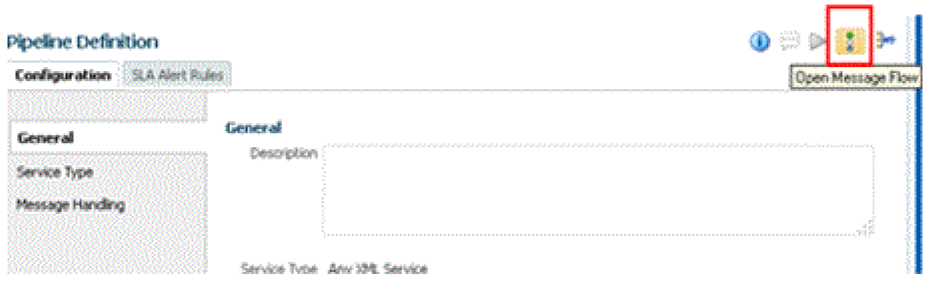
5. Click the **Save All** icon in the right corner, as shown in Figure 7–84.

Figure 7–84 Save All Icon



6. Double-click the **Pipeline** node and click the **Open Message Flow** icon on the right pane to open the message flow, as shown in Figure 7–85.

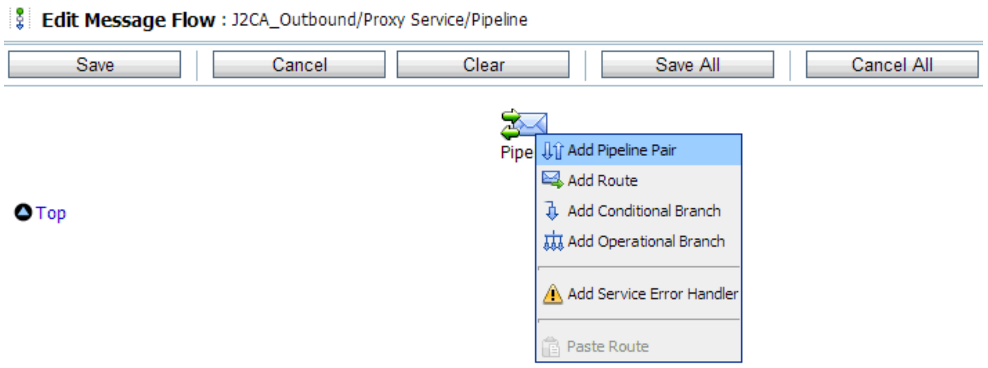
Figure 7–85 Open Message Flow Icon



Service Type Any XML Service

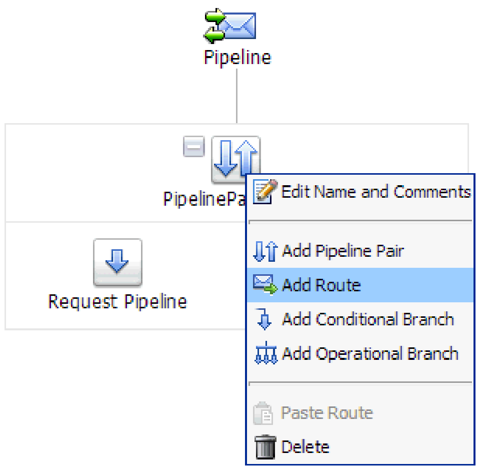
7. Click the Proxy Service icon and select **Add Pipeline Pair** from the menu, as shown in Figure 7–86.

Figure 7–86 Add Pipeline Pair Option



8. Click the **PipelinePairNode1** icon and select **Add Route** from the menu, as shown in Figure 7–87.

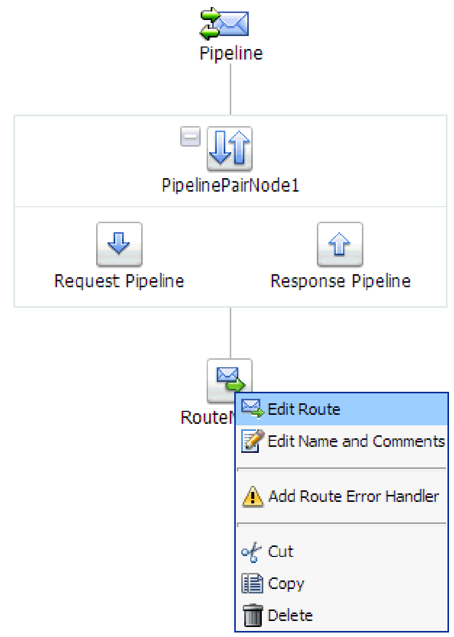
Figure 7–87 Add Route Option



The RouteNode1 icon is added below the PipelinePairNode1 icon.

9. Click the RouteNode1 icon and select **Edit Route** from the menu, as shown in Figure 7–88.

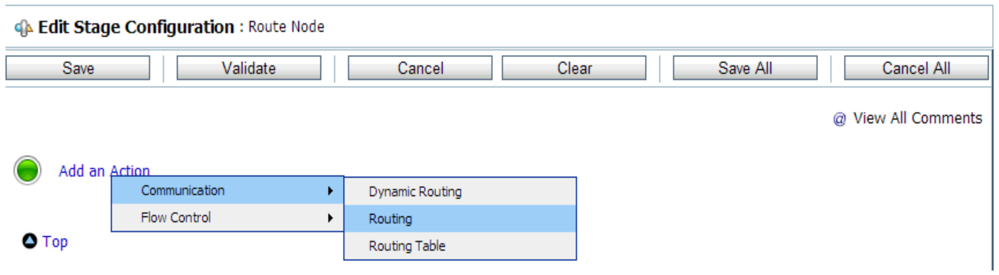
Figure 7–88 Edit Route Option



The Edit Stage Configuration workspace area is displayed.

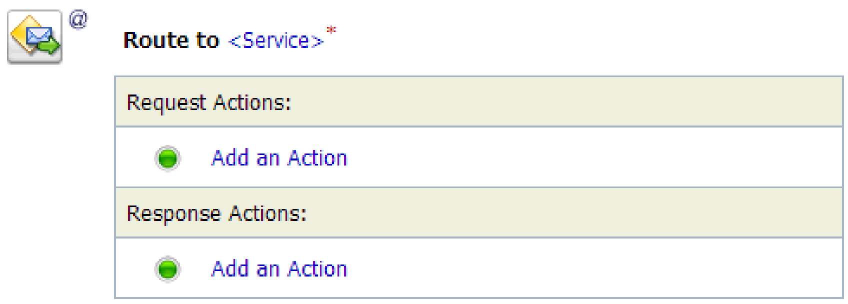
10. Click **Add an Action**, select **Communication** and click **Routing**, as shown in Figure 7–89.

Figure 7–89 Edit Stage Configuration Workspace Area



11. Click **<Service>**, as shown in Figure 7–90.

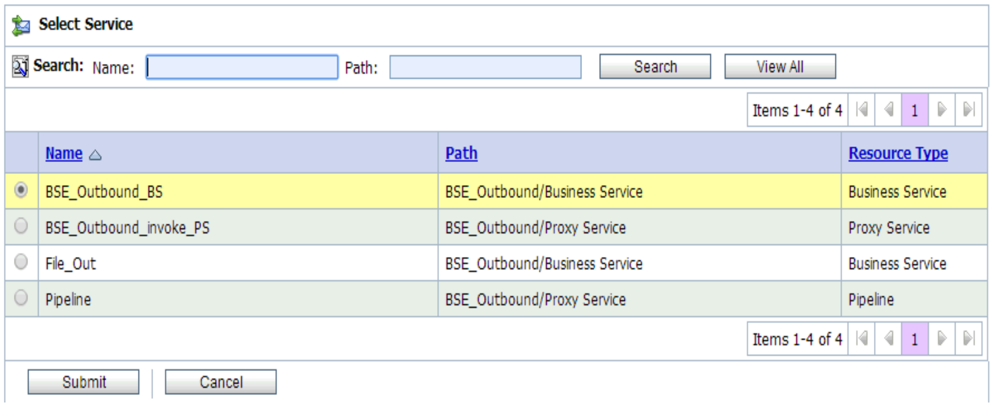
Figure 7–90 Actions



The Select Service dialog is displayed.

12. Select the WSDL type Business Service configured for J.D. Edwards OneWorld and click on **Submit**, as shown in Figure 7–91.

Figure 7–91 Select Service Dialog



Select Service

Search: Name: Path: Search View All

Items 1-4 of 4 1

	Name	Path	Resource Type
●	BSE_Outbound_BS	BSE_Outbound/Business Service	Business Service
○	BSE_Outbound_invoke_PS	BSE_Outbound/Proxy Service	Proxy Service
○	File_Out	BSE_Outbound/Business Service	Business Service
○	Pipeline	BSE_Outbound/Proxy Service	Pipeline

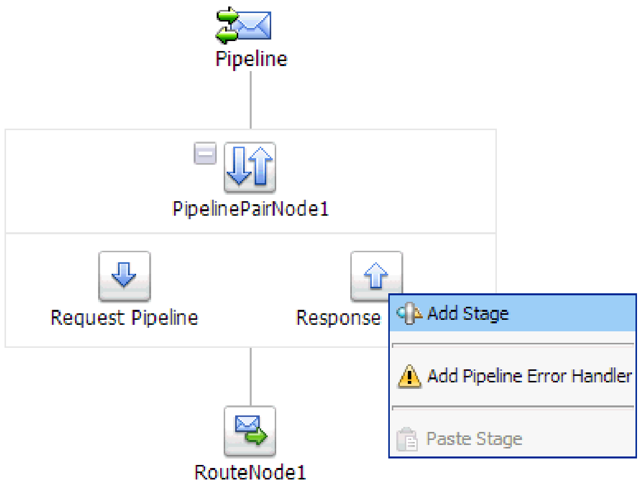
Items 1-4 of 4 1

Submit Cancel

13. Select the name of the J.D. Edwards OneWorld business object as the operational attribute from the list, and click **Save**.

14. Click the Response Pipeline icon and select **Add Stage** from the menu, as shown in Figure 7–92.

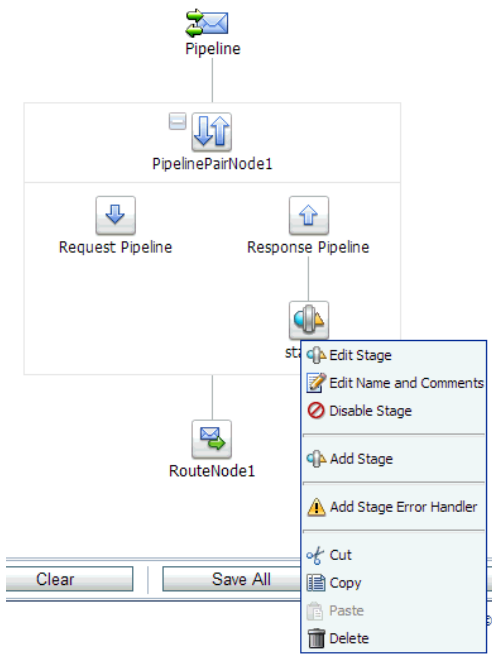
Figure 7–92 Response Pipeline Icon



The Stage1 icon is added below the Response Pipeline icon.

15. Click the Stage1 icon and select **Edit Stage** from the menu, as shown in Figure 7–93.

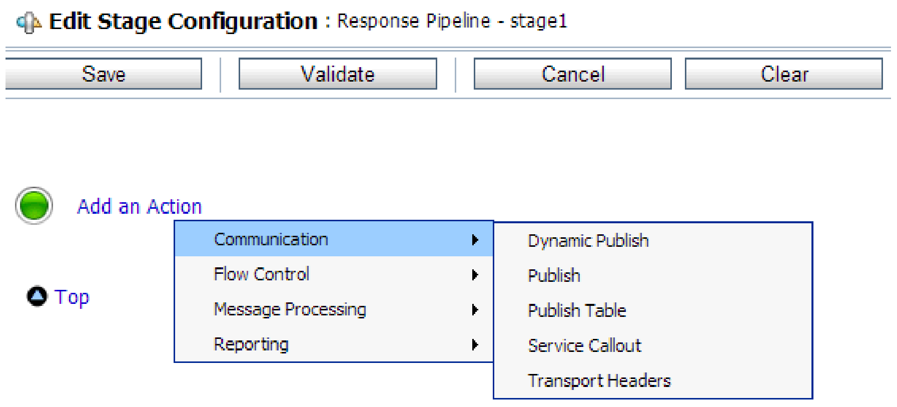
Figure 7–93 Edit Stage Option



The Edit Stage Configuration workspace area is displayed.

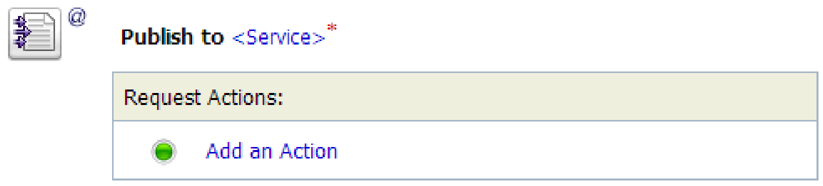
16. Click **Add an Action**, select **Communication**, and then click **Publish**, as shown in Figure 7–94.

Figure 7–94 Edit Stage Configuration Workspace Area



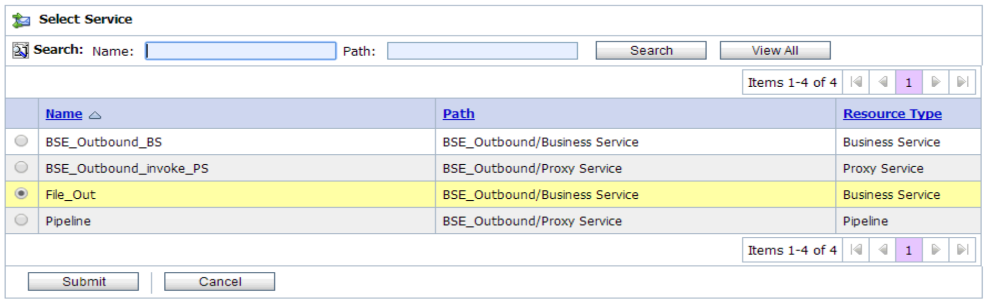
17. Click **<Service>**, as shown in Figure 7–95.

Figure 7–95 <Service> Action



18. In the Select Service dialog, select a File type Business Service and click **Submit**, as shown in Figure 7–96.

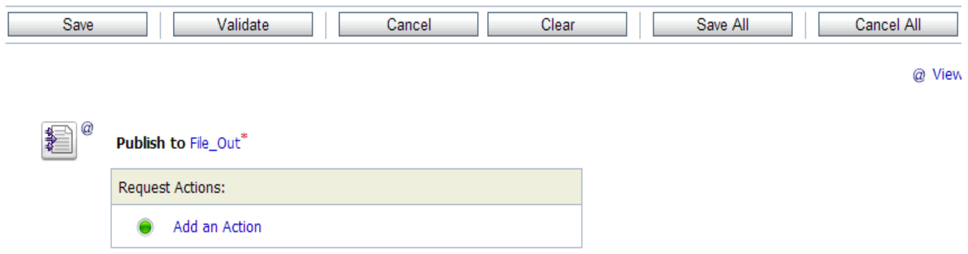
Figure 7–96 Select Service Dialog



	Name	Path	Resource Type
○	BSE_Outbound_BS	BSE_Outbound/Business Service	Business Service
○	BSE_Outbound_invoke_PS	BSE_Outbound/Proxy Service	Proxy Service
◉	File_Out	BSE_Outbound/Business Service	Business Service
○	Pipeline	BSE_Outbound/Proxy Service	Pipeline

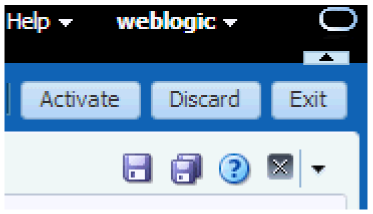
19. Click **Save All**, as shown in Figure 7–97.

Figure 7–97 Save All Button



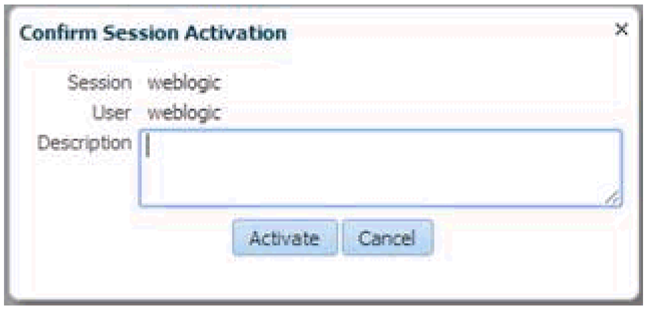
20. Click **Activate** in the right pane of the Oracle Service Bus session, as shown in Figure 7–98.

Figure 7–98 Activate Button



21. Click **Activate** to save the changes, as shown in Figure 7–99.

Figure 7–99 Confirm Session Activation


Confirm Session Activation

Session weblogic
User weblogic
Description

22. Copy and paste an input XML file in the input folder you have configured (for example, C:\input).

    Output is received in the configured output location (for example, C:\output).

7.5 Configuring JMS Proxy Services Using Oracle Service Bus (J2CA Configuration)

This section describes how to configure JMS Proxy Services using Oracle Service Bus for a J2CA configuration.

1. Start Oracle Service Bus and create the required project folder.

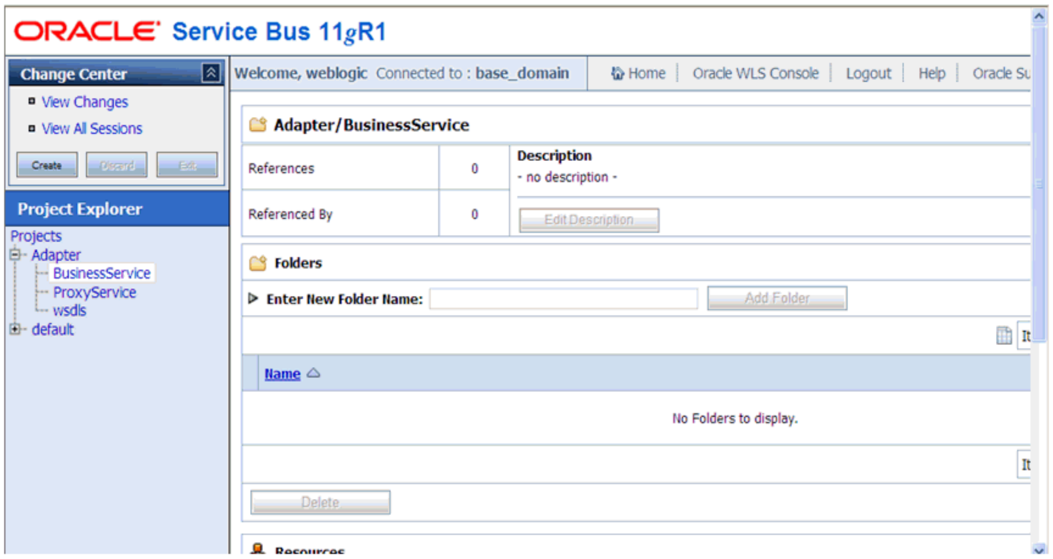
   For more information, see Section 7.2.1, "Starting Oracle Service Bus and Creating Project Folders" on page 7-2.

2. Generate and publish the WSDL from Application Explorer to the created project folder. Using the published WSDL, create a Business Service.

   For more information, see Section 7.2.3, "Publishing a WSDL From Application Explorer to Oracle Service Bus" on page 7-6.

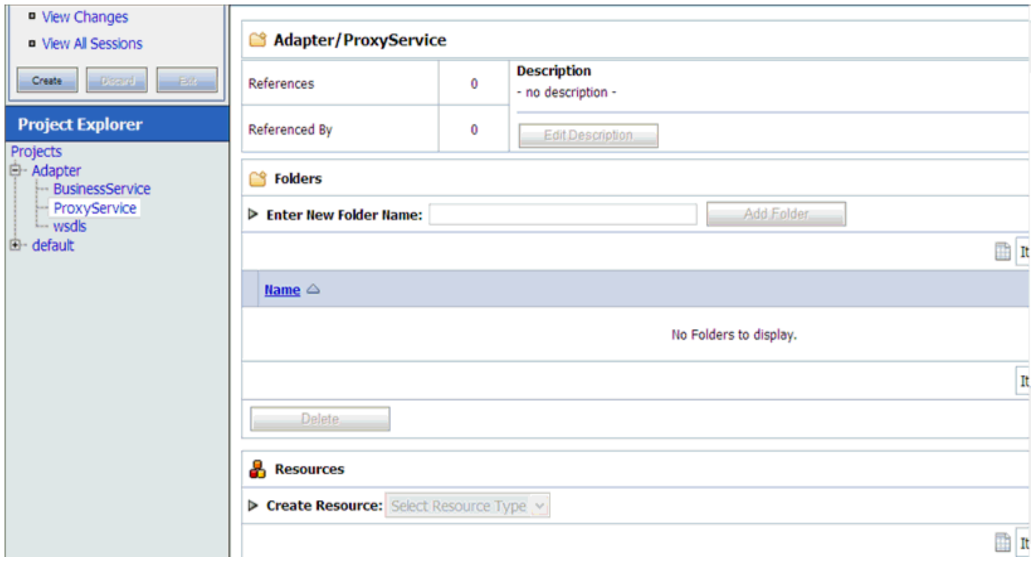
3. Open the Service Bus Console page, as shown in Figure 7–100.

Figure 7–100 Service Bus Console



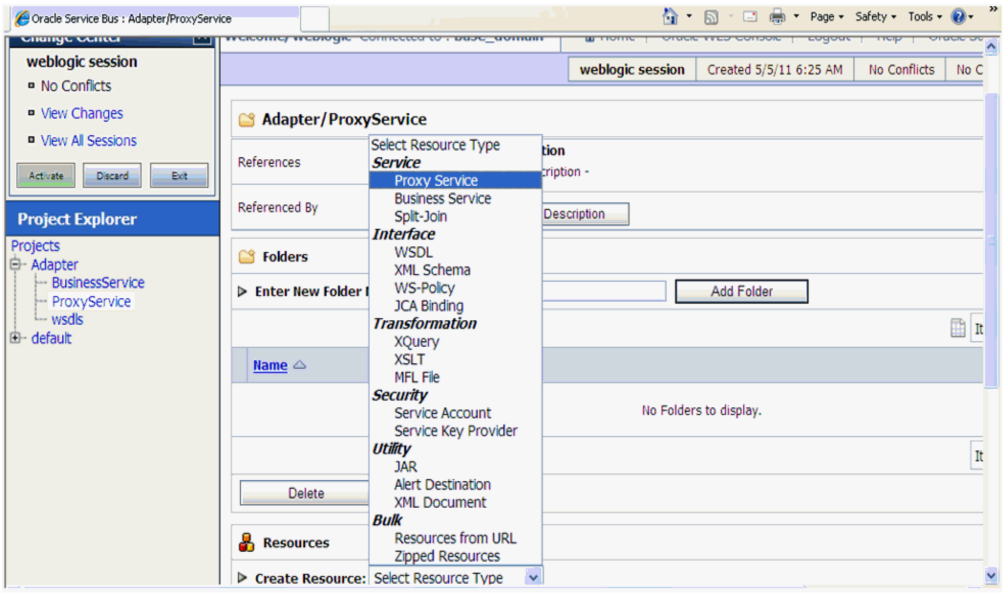
4. Select the ProxyService project folder in the left pane, and click **Create**, as shown in Figure 7–101.

Figure 7–101 Proxy Service



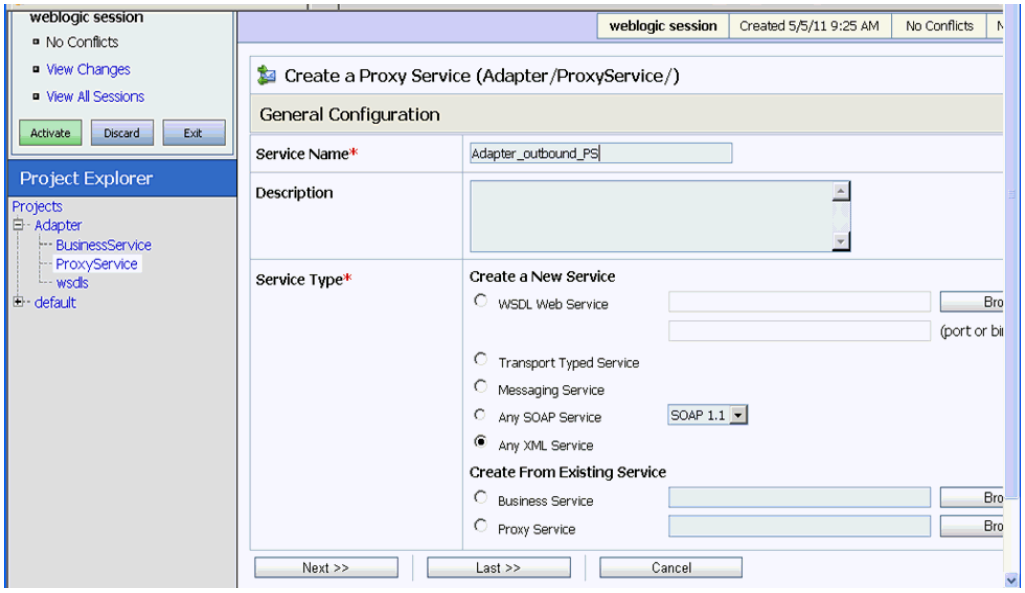
5. In the right pane, select **Proxy Service** from the Create Resource list, as shown in Figure 7–102.

Figure 7–102 Create Resource Menu



6. Enter an appropriate name in the **Service Name** field, as shown in Figure 7–103.

Figure 7–103 Service Name



weblogic session
- No Conflicts
- View Changes
- View All Sessions

Activate | Discard | Exit

Project Explorer

Projects
- Adapter
  - BusinessService
  - ProxyService
  - wsdls
- default

weblogic session | Created 5/5/11 9:25 AM | No Conflicts

Create a Proxy Service (Adapter/ProxyService/)

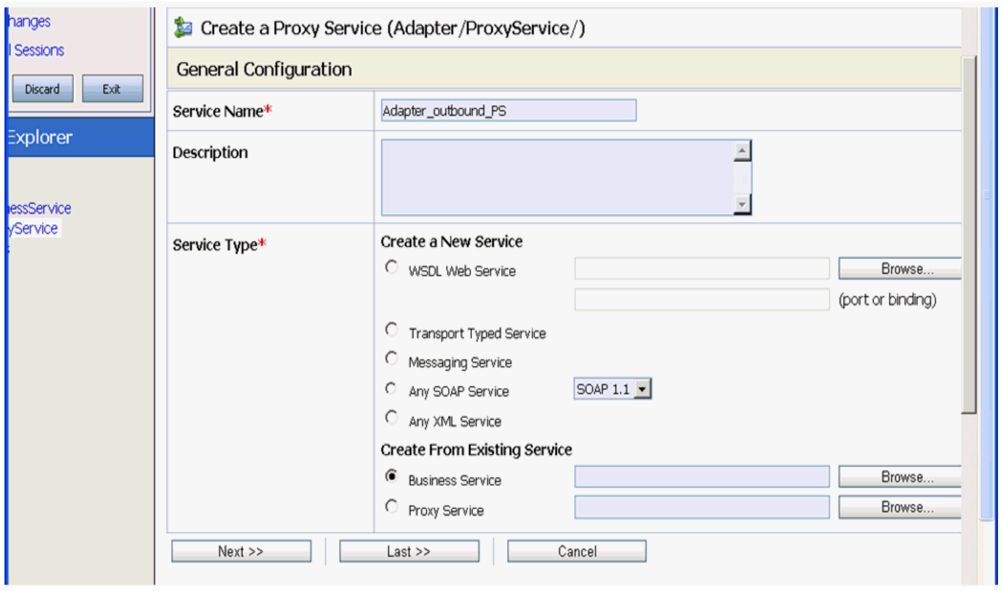
General Configuration

Service Name*	Adapter_outbound_PS
Description	
Service Type*	Create a New Service
	( ) WSDL Web Service
	( ) Transport Typed Service
	( ) Messaging Service
	( ) Any SOAP Service — SOAP 1.1
	[✓ Any XML Service]
	Create From Existing Service
	( ) Business Service
	( ) Proxy Service

Next >> | Last >> | Cancel

7. In the Service Type section, under Create From Existing Service, select the **Business Service** radio button and click **Browse**, as shown in Figure 7–104.

Figure 7–104 Business Service



Create a Proxy Service (Adapter/ProxyService/)

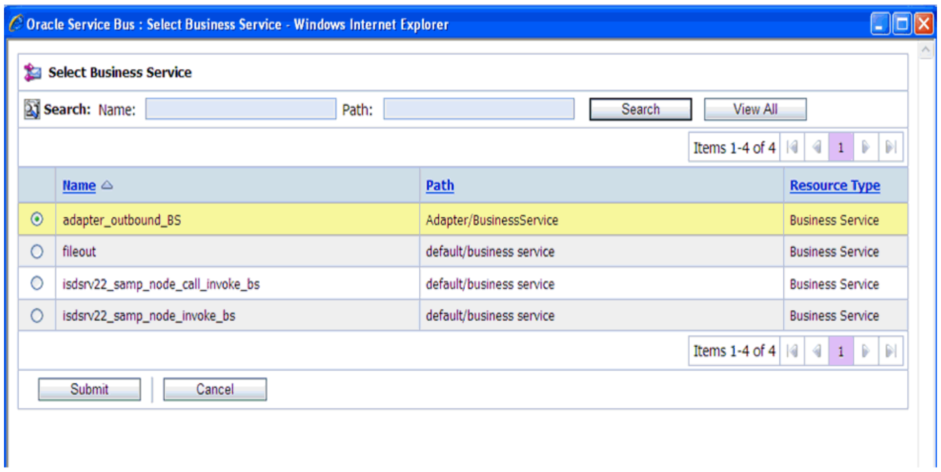
General Configuration

Service Name*	Adapter_outbound_PS
Description	
Service Type*	Create a New Service
	( ) WSDL Web Service — Browse... (port or binding)
	( ) Transport Typed Service
	( ) Messaging Service
	( ) Any SOAP Service — SOAP 1.1
	( ) Any XML Service
	Create From Existing Service
	[✓ Business Service] — Browse...
	( ) Proxy Service — Browse...

Next >> | Last >> | Cancel

8. Select the existing business service and click **Submit**, as shown in Figure 7–105.

Figure 7–105 Existing Business Service



Oracle Service Bus : Select Business Service - Windows Internet Explorer

Select Business Service

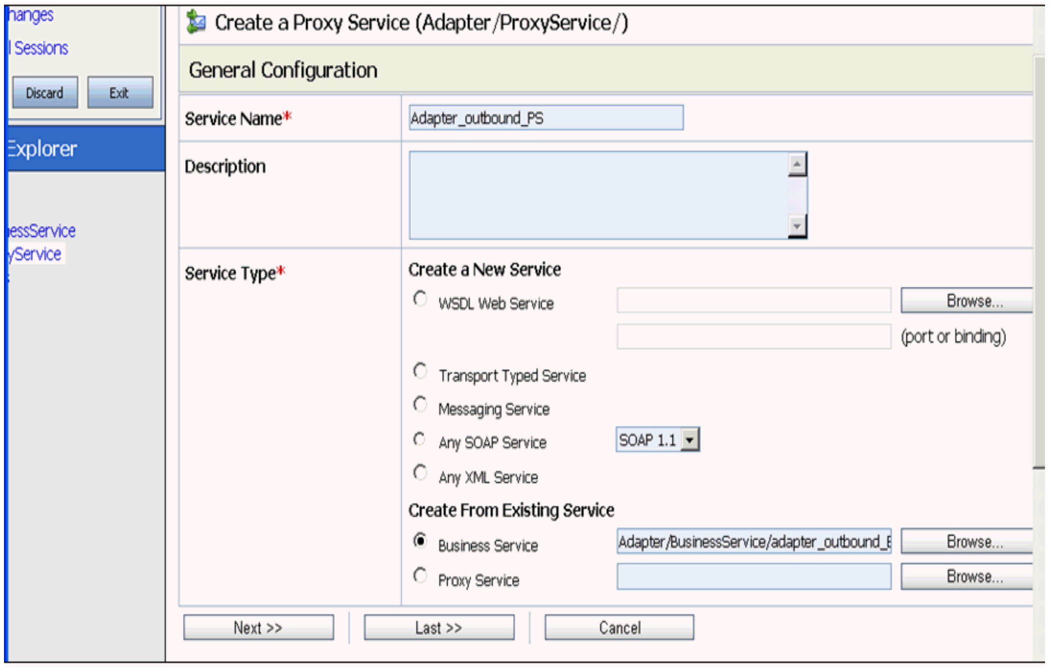
Search: Name: Path:

Items 1-4 of 4

	Name	Path	Resource Type
●	adapter_outbound_BS	Adapter/BusinessService	Business Service
○	fileout	default/business service	Business Service
○	isdsrv22_samp_node_call_invoke_bs	default/business service	Business Service
○	isdsrv22_samp_node_invoke_bs	default/business service	Business Service

9. Click **Next**, as shown in Figure 7–106.

Figure 7–106 Next



Create a Proxy Service (Adapter/ProxyService/)

General Configuration

Service Name*: Adapter_outbound_PS

Description:

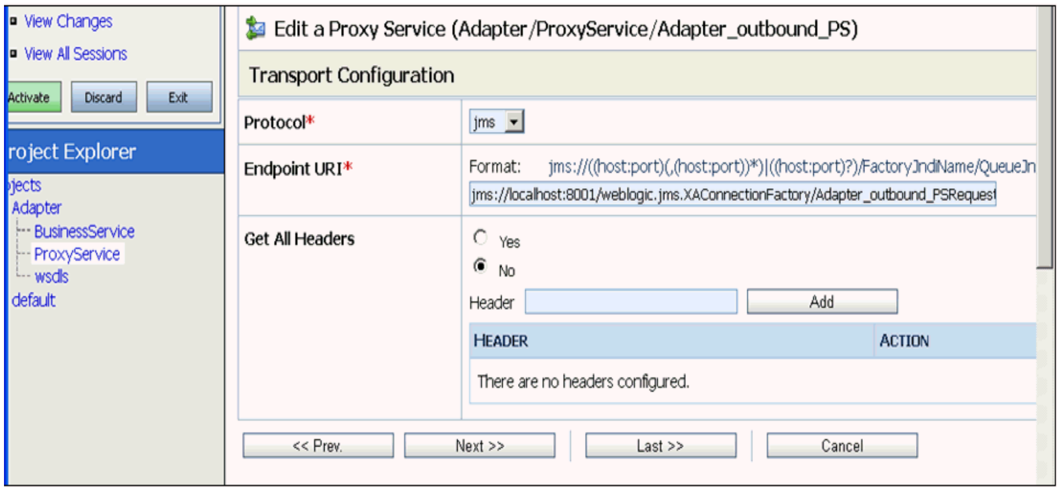
Service Type*:

Create a New Service
- ○ WSDL Web Service (port or binding)
- ○ Transport Typed Service
- ○ Messaging Service
- ○ Any SOAP Service SOAP 1.1
- ○ Any XML Service

Create From Existing Service
- ● Business Service Adapter/BusinessService/adapter_outbound_E
- ○ Proxy Service

10. Select **jms** from the Protocol list and click **Next**, as shown in Figure 7–107.

Figure 7–107 Protocol List



Edit a Proxy Service (Adapter/ProxyService/Adapter_outbound_PS)

Transport Configuration

Protocol*	jms
Endpoint URI*	Format: jms://((host:port)(,(host:port))*)|((host:port)?)/FactoryJndiName/QueueJn jms://localhost:8001/weblogic.jms.XAConnectionFactory/Adapter_outbound_PSRequest
Get All Headers	Yes [✓ No] Header

HEADER	ACTION
There are no headers configured.	

11. Provide the following parameters, as shown in Figure 7–108.

    a. Select **Queue** in the Destination Type section.

    b. Enable the **Is Response Required** check box.

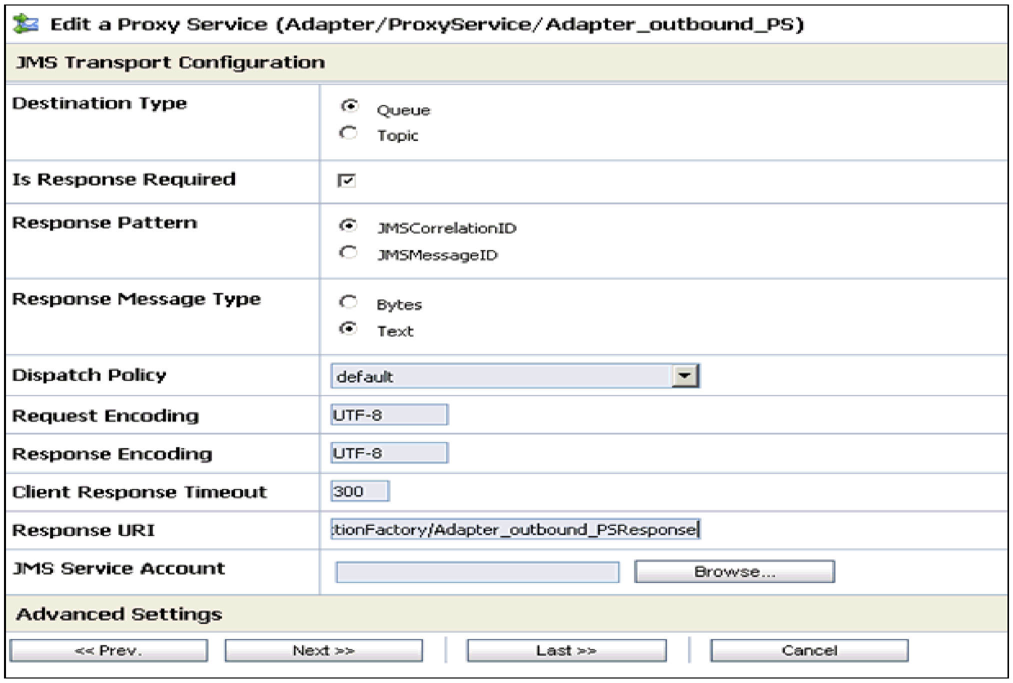
    c. Select **Text** in the Response Message Type section.

    d. In the Response URI field, provide the Endpoint URI used in the Transport Configuration and change `Request` to `Response`.

    For example:

    ```
    jms://localhost:8001/weblogic.jms.XAConnectionFactory/Adap
    ter_outbound_PSResponse
    ```

Figure 7–108 Edit a Proxy Service



Edit a Proxy Service (Adapter/ProxyService/Adapter_outbound_PS)

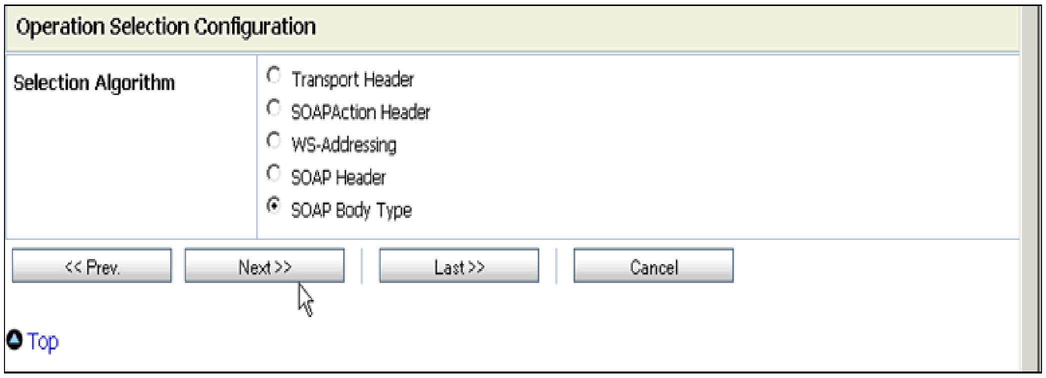
JMS Transport Configuration

Destination Type	[✓ Queue] Topic
Is Response Required	✓
Response Pattern	[✓ JMSCorrelationID] JMSMessageID
Response Message Type	Bytes [✓ Text]
Dispatch Policy	default
Request Encoding	UTF-8
Response Encoding	UTF-8
Client Response Timeout	300
Response URI	tionFactory/Adapter_outbound_PSResponse
JMS Service Account	

12. Click **Next**.

    The Operation Selection Configuration pane appears, as shown in Figure 7–109.

Figure 7–109 Operation Selection Configuration Pane

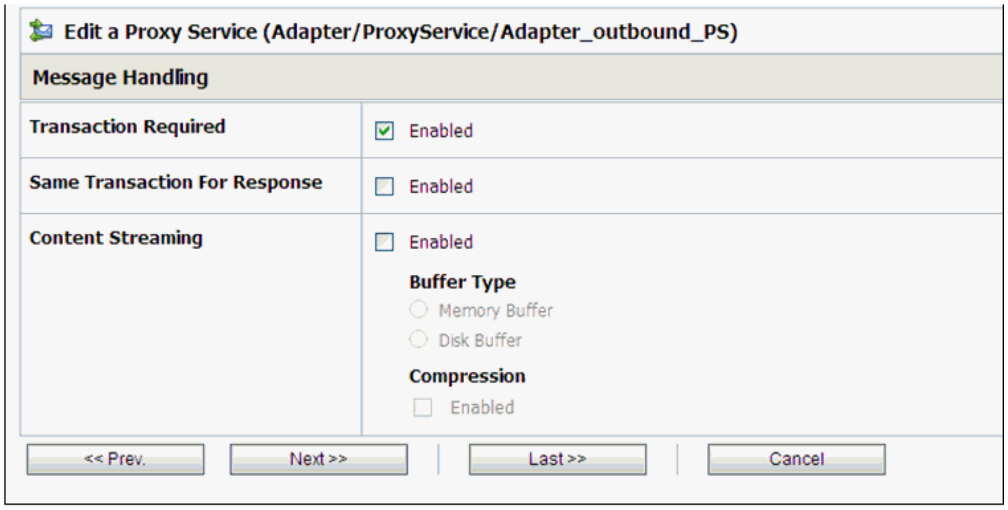


Operation Selection Configuration

Selection Algorithm
- ( ) Transport Header
- ( ) SOAPAction Header
- ( ) WS-Addressing
- ( ) SOAP Header
- (●) SOAP Body Type

13. Ensure the **SOAP Body Type** is selected and click **Next**.
14. Enable the **Transaction Required** box and click **Next**, as shown in Figure 7–110.

Figure 7–110 Message Handling



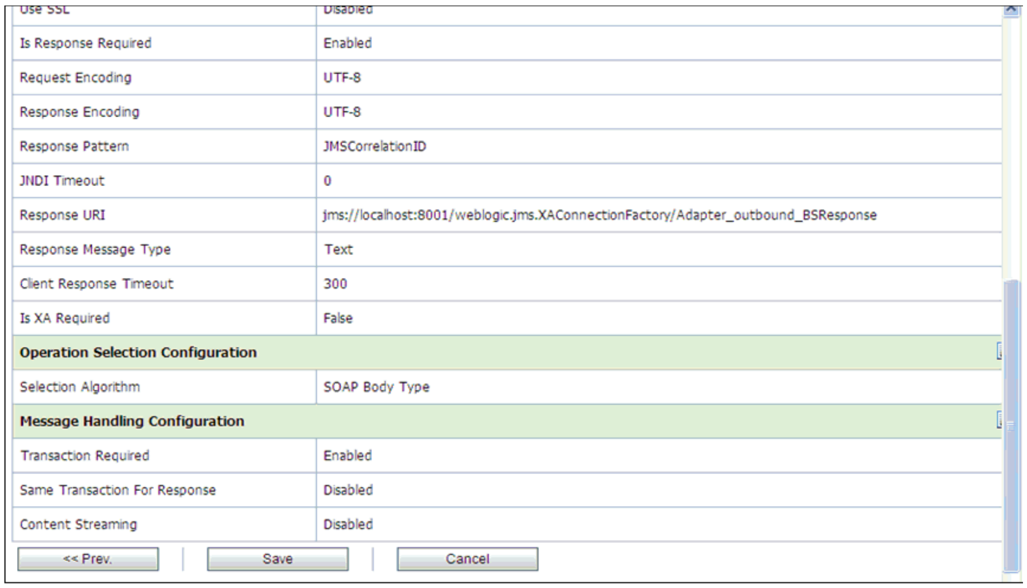
Edit a Proxy Service (Adapter/ProxyService/Adapter_outbound_PS)

Message Handling

Transaction Required	✓ Enabled
Same Transaction For Response	☐ Enabled
Content Streaming	☐ Enabled Buffer Type ( ) Memory Buffer ( ) Disk Buffer Compression ☐ Enabled

15. Click **Save**, as shown in Figure 7–111.

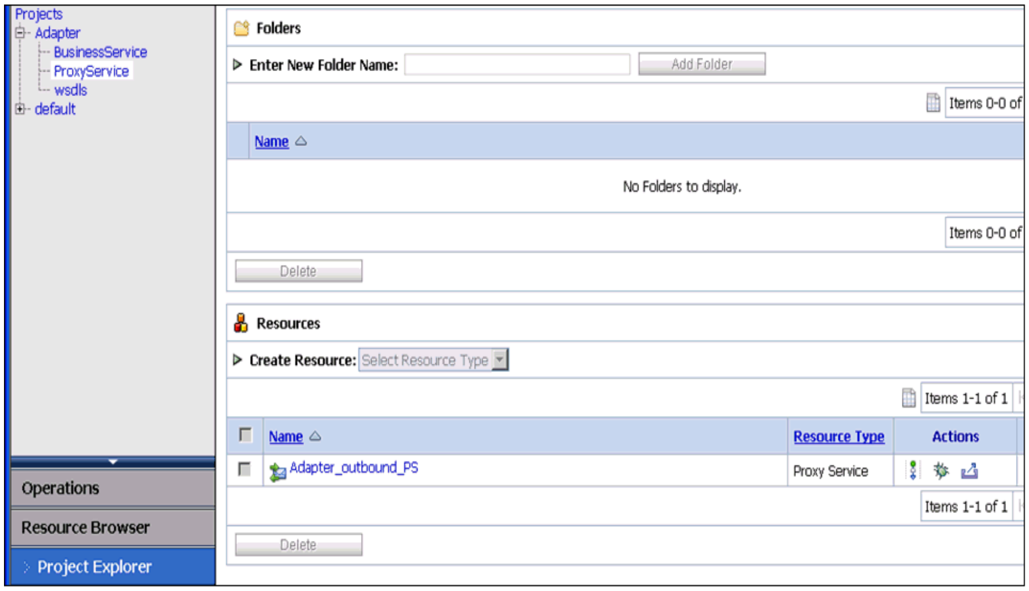
Figure 7–111 Save



Use SSL	Disabled
Is Response Required	Enabled
Request Encoding	UTF-8
Response Encoding	UTF-8
Response Pattern	JMSCorrelationID
JNDI Timeout	0
Response URI	jms://localhost:8001/weblogic.jms.XAConnectionFactory/Adapter_outbound_BSResponse
Response Message Type	Text
Client Response Timeout	300
Is XA Required	False
Operation Selection Configuration	
Selection Algorithm	SOAP Body Type
Message Handling Configuration	
Transaction Required	Enabled
Same Transaction For Response	Disabled
Content Streaming	Disabled

The created Proxy Service is saved, as shown in Figure 7–112.

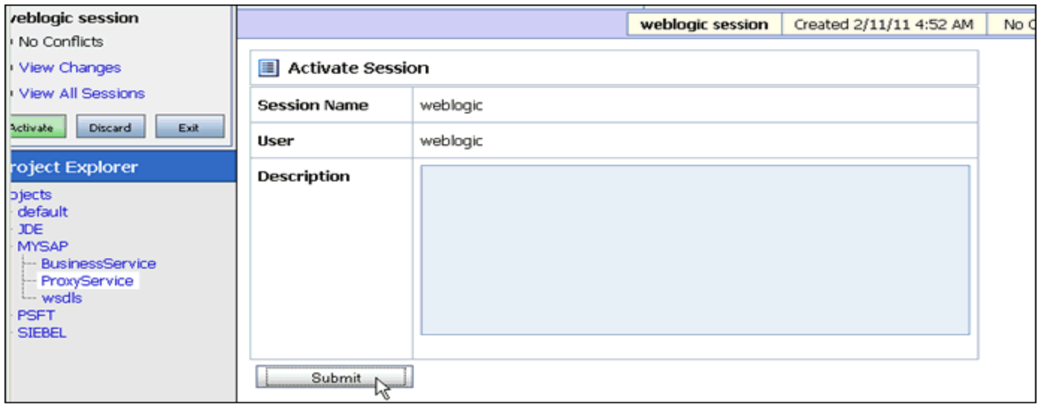
Figure 7–112 Proxy Service



	Name	Resource Type	Actions
	Adapter_outbound_PS	Proxy Service	

16. In the left pane, click **Activate**, and then **Submit**, as shown in Figure 7–113.

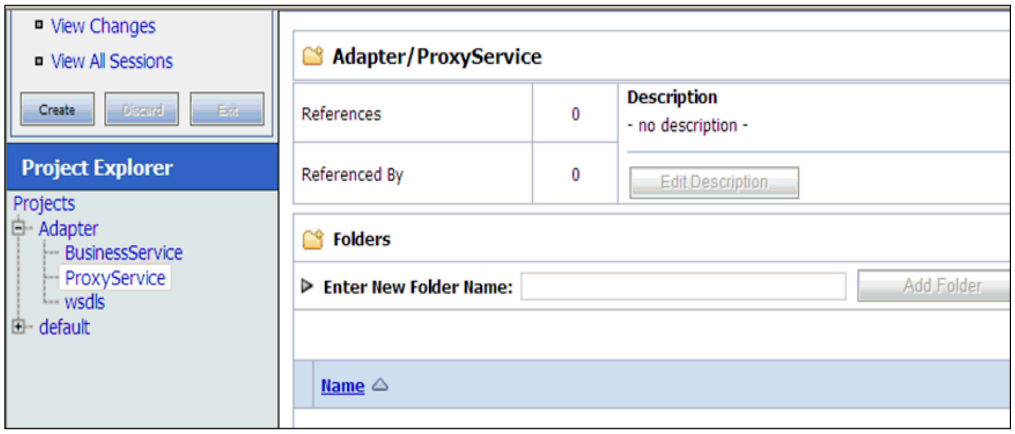
Figure 7–113 Activate Session



Session Name	weblogic
User	weblogic
Description	

17. In the left pane, click **ProxyService** under the Projects folder, as shown in Figure 7–114.

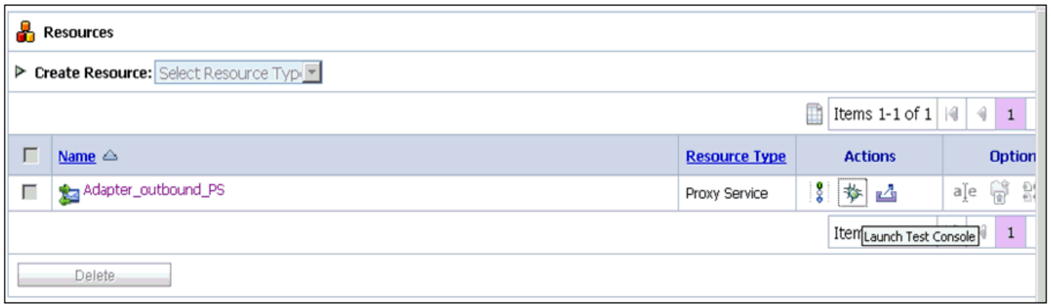
Figure 7–114 Adapter/ProxyService



References	0	Description - no description -
Referenced By	0	

18. Click the **Launch Test Console** icon for the created Proxy Service, as shown in Figure 7–115.

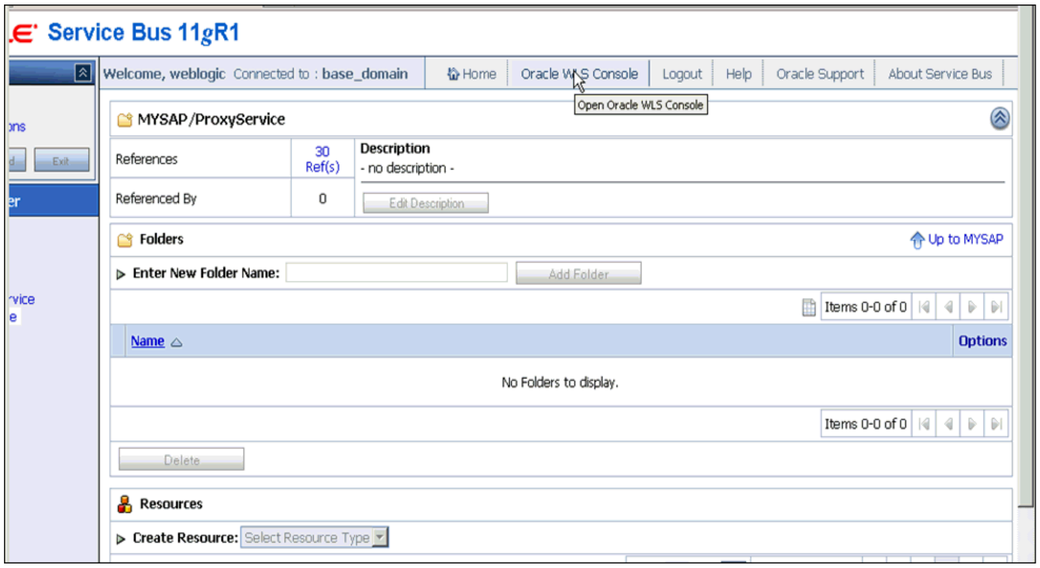
Figure 7–115 Launch Test Console Icon



Name	Resource Type	Actions	Option
Adapter_outbound_PS	Proxy Service		

19. Provide the input values for **Payload**, uncheck the **Direct Call** box, and click **Execute**.
20. Review the Response document, and then click **Close**.
21. Click the **Oracle WLS Console** tab, as shown in Figure 7–116.

Figure 7–116 ProxyService

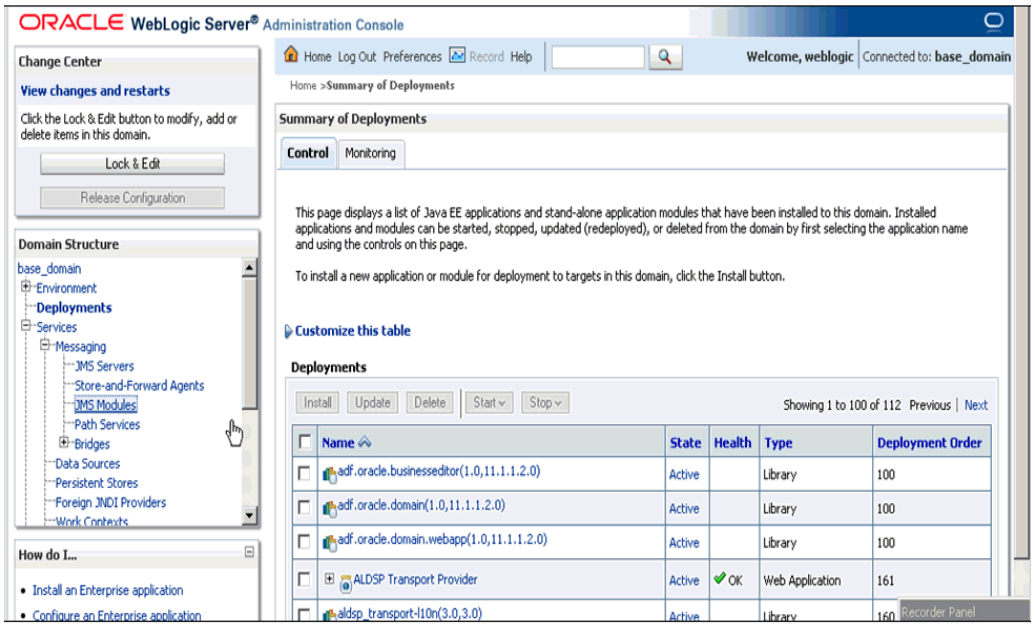


References	30 Ref(s)	Description - no description -
Referenced By	0	

No Folders to display.

22. In the Oracle WLS Console, expand **Services**, expand **Messaging**, and click **JMS Modules**, as shown in Figure 7–117.

Figure 7–117 Oracle WLS Console



Click the Lock & Edit button to modify, add or delete items in this domain.

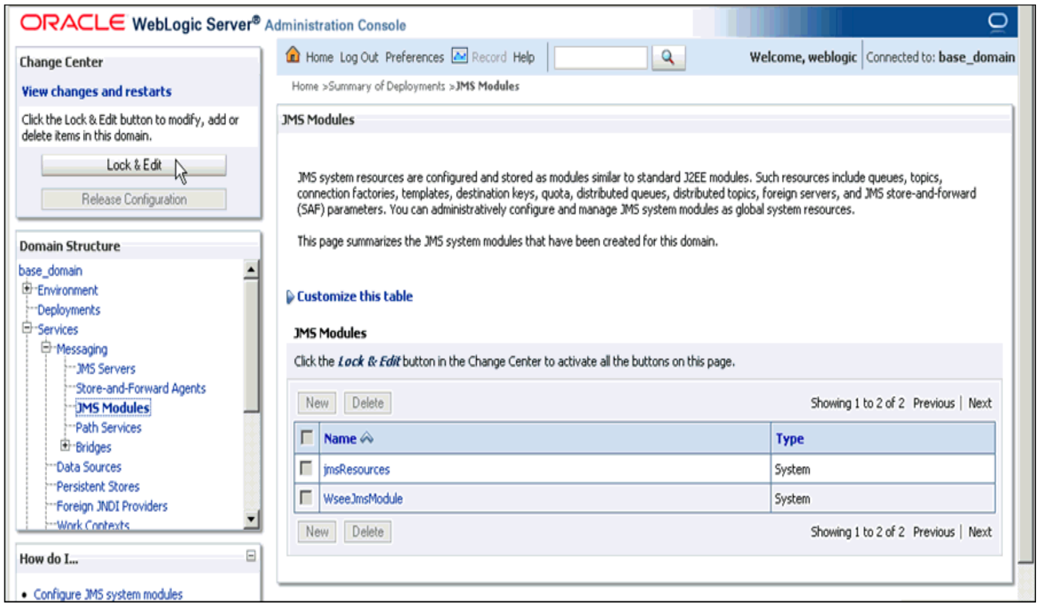
This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

Name	State	Health	Type	Deployment Order
adf.oracle.businesseditor(1.0,11.1.1.2.0)	Active		Library	100
adf.oracle.domain(1.0,11.1.1.2.0)	Active		Library	100
adf.oracle.domain.webapp(1.0,11.1.1.2.0)	Active		Library	100
ALDSP Transport Provider	Active	✓ OK	Web Application	161
aldsp_transport-l10n(3.0,3.0)	Active		Library	160

23. Click **jmsResources**, as shown in Figure 7–118.

Figure 7–118 JMS Modules



ORACLE WebLogic Server® Administration Console

Home Log Out Preferences Record Help | Welcome, weblogic | Connected to: base_domain

Home >Summary of Deployments >JMS Modules

Change Center

View changes and restarts

Click the Lock & Edit button to modify, add or delete items in this domain.

Lock & Edit

Release Configuration

Domain Structure

base_domain
- Environment
- Deployments
- Services
  - Messaging
    - JMS Servers
    - Store-and-Forward Agents
    - JMS Modules
    - Path Services
    - Bridges
  - Data Sources
  - Persistent Stores
  - Foreign JNDI Providers
  - Work Contexts

How do I...

- Configure JMS system modules

JMS Modules

JMS system resources are configured and stored as modules similar to standard J2EE modules. Such resources include queues, topics, connection factories, templates, destination keys, quota, distributed queues, distributed topics, foreign servers, and JMS store-and-forward (SAF) parameters. You can administratively configure and manage JMS system modules as global system resources.

This page summarizes the JMS system modules that have been created for this domain.

Customize this table

JMS Modules

Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

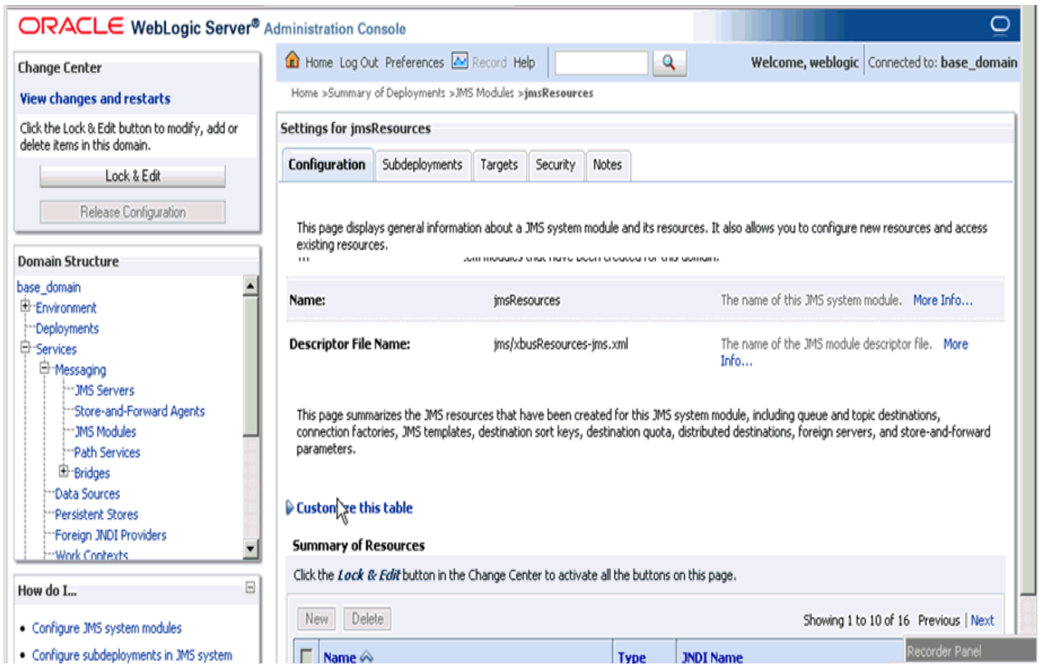
New Delete Showing 1 to 2 of 2 Previous | Next

Name	Type
jmsResources	System
WseeJmsModule	System

New Delete Showing 1 to 2 of 2 Previous | Next

24. Click **Lock & Edit**, as shown in Figure 7–119.

Figure 7–119 Configuration Settings



ORACLE WebLogic Server® Administration Console

Home Log Out Preferences Record Help | Welcome, weblogic | Connected to: base_domain

Home >Summary of Deployments >JMS Modules >jmsResources

Change Center

View changes and restarts

Click the Lock & Edit button to modify, add or delete items in this domain.

Lock & Edit

Release Configuration

Domain Structure

base_domain
- Environment
- Deployments
- Services
  - Messaging
    - JMS Servers
    - Store-and-Forward Agents
    - JMS Modules
    - Path Services
    - Bridges
  - Data Sources
  - Persistent Stores
  - Foreign JNDI Providers
  - Work Contexts

How do I...

- Configure JMS system modules
- Configure subdeployments in JMS system

Settings for jmsResources

Configuration | Subdeployments | Targets | Security | Notes

This page displays general information about a JMS system module and its resources. It also allows you to configure new resources and access existing resources.

Name:	jmsResources	The name of this JMS system module. More Info...
Descriptor File Name:	jms/xbusResources-jms.xml	The name of the JMS module descriptor file. More Info...

This page summarizes the JMS resources that have been created for this JMS system module, including queue and topic destinations, connection factories, JMS templates, destination sort keys, destination quota, distributed destinations, foreign servers, and store-and-forward parameters.

Customize this table

Summary of Resources

Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

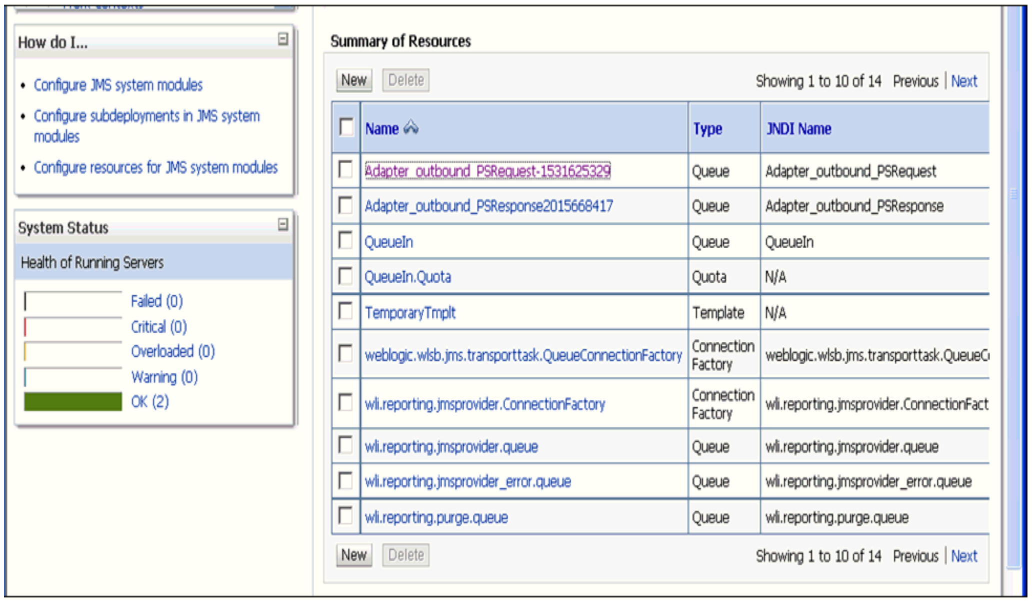
New Delete Showing 1 to 10 of 16 Previous | Next

Name	Type	JNDI Name

Recorder Panel

25. Click the appropriate request link, for example, **Adapter_outbound_PSRequest**, as shown in Figure 7–120.

Figure 7–120 Adapter_outbound_PSRequest



How do I...

- Configure JMS system modules
- Configure subdeployments in JMS system modules
- Configure resources for JMS system modules

System Status

Health of Running Servers

	Failed (0)
	Critical (0)
	Overloaded (0)
	Warning (0)
🟩	OK (2)

Summary of Resources

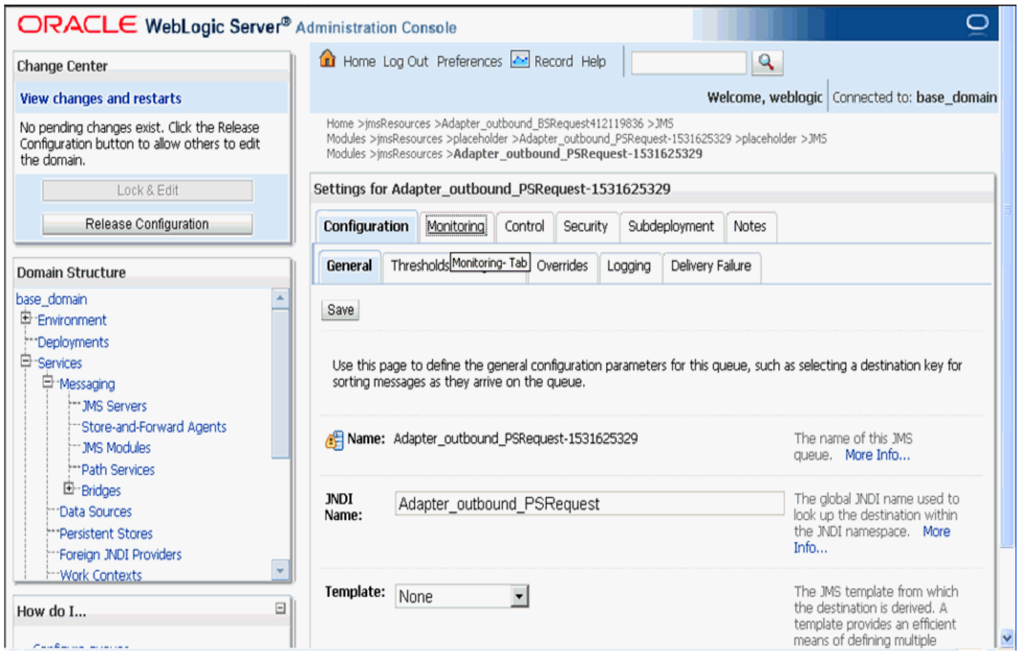
New Delete Showing 1 to 10 of 14 Previous | Next

Name	Type	JNDI Name
Adapter_outbound_PSRequest-1531625329	Queue	Adapter_outbound_PSRequest
Adapter_outbound_PSResponse2015668417	Queue	Adapter_outbound_PSResponse
QueueIn	Queue	QueueIn
QueueIn.Quota	Quota	N/A
TemporaryTmplt	Template	N/A
weblogic.wlsb.jms.transporttask.QueueConnectionFactory	Connection Factory	weblogic.wlsb.jms.transporttask.QueueC
wli.reporting.jmsprovider.ConnectionFactory	Connection Factory	wli.reporting.jmsprovider.ConnectionFact
wli.reporting.jmsprovider.queue	Queue	wli.reporting.jmsprovider.queue
wli.reporting.jmsprovider_error.queue	Queue	wli.reporting.jmsprovider_error.queue
wli.reporting.purge.queue	Queue	wli.reporting.purge.queue

New Delete Showing 1 to 10 of 14 Previous | Next

26. Click the **Monitoring** tab, as shown in Figure 7–121.

Figure 7–121 Monitoring Tab



ORACLE WebLogic Server® Administration Console

Change Center

View changes and restarts

No pending changes exist. Click the Release Configuration button to allow others to edit the domain.

Lock & Edit

Release Configuration

Domain Structure

base_domain
- Environment
- Deployments
- Services
  - Messaging
    - JMS Servers
    - Store-and-Forward Agents
    - JMS Modules
    - Path Services
    - Bridges
  - Data Sources
  - Persistent Stores
  - Foreign JNDI Providers
  - Work Contexts

How do I...

Home Log Out Preferences Record Help

Welcome, weblogic | Connected to: base_domain

Home >jmsResources >Adapter_outbound_BSRequest412119836 >JMS Modules >jmsResources >placeholder >Adapter_outbound_PSRequest-1531625329 >placeholder >JMS Modules >jmsResources >**Adapter_outbound_PSRequest-1531625329**

Settings for Adapter_outbound_PSRequest-1531625329

Configuration | Monitoring | Control | Security | Subdeployment | Notes

General | Thresholds | Overrides | Logging | Delivery Failure

Monitoring- Tab

Save

Use this page to define the general configuration parameters for this queue, such as selecting a destination key for sorting messages as they arrive on the queue.

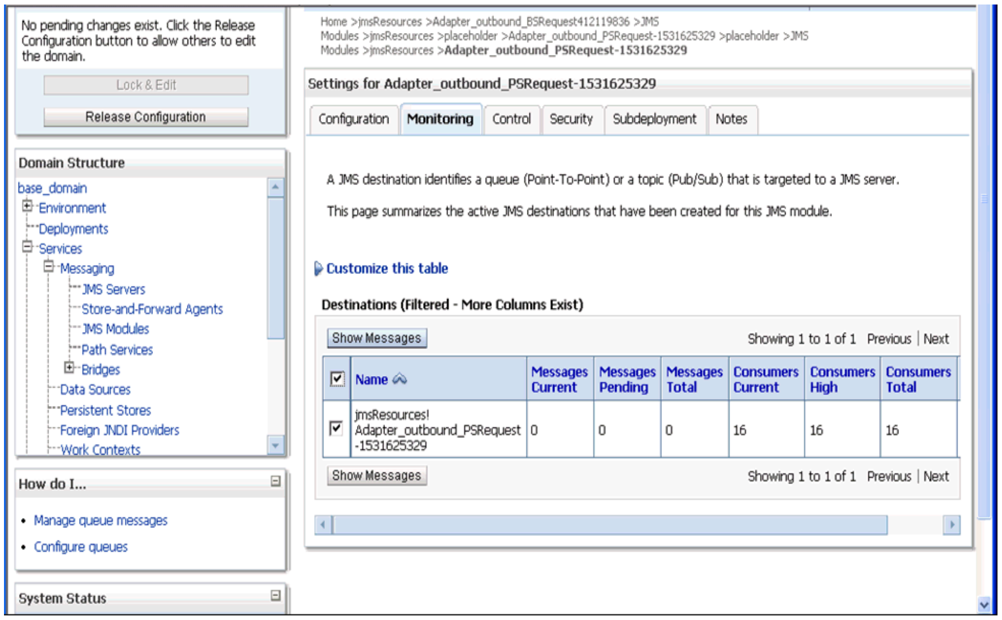
Name: Adapter_outbound_PSRequest-1531625329 — The name of this JMS queue. More Info...

JNDI Name: Adapter_outbound_PSRequest — The global JNDI name used to look up the destination within the JNDI namespace. More Info...

Template: None — The JMS template from which the destination is derived. A template provides an efficient means of defining multiple

27. Enable the check box and click **Show Messages**, as shown in Figure 7–122.

Figure 7–122 Adapter Settings


No pending changes exist. Click the Release Configuration button to allow others to edit the domain.

Home >jmsResources >Adapter_outbound_BSRequest412119836 >JMS Modules >jmsResources >placeholder >Adapter_outbound_PSRequest-1531625329 >placeholder >JMS Modules >jmsResources >**Adapter_outbound_PSRequest-1531625329**

Settings for Adapter_outbound_PSRequest-1531625329

A JMS destination identifies a queue (Point-To-Point) or a topic (Pub/Sub) that is targeted to a JMS server.

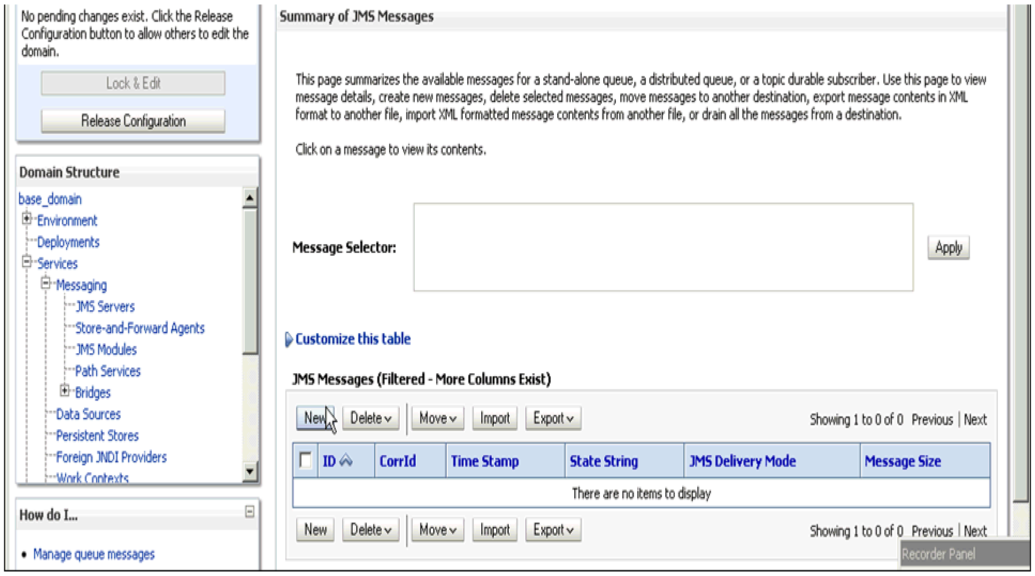
This page summarizes the active JMS destinations that have been created for this JMS module.

Destinations (Filtered - More Columns Exist)

☑	Name	Messages Current	Messages Pending	Messages Total	Consumers Current	Consumers High	Consumers Total
☑	jmsResources!Adapter_outbound_PSRequest-1531625329	0	0	0	16	16	16

28. Click **New**, as shown in Figure 7–123.

Figure 7–123 JMS Messages


No pending changes exist. Click the Release Configuration button to allow others to edit the domain.

Summary of JMS Messages

This page summarizes the available messages for a stand-alone queue, a distributed queue, or a topic durable subscriber. Use this page to view message details, create new messages, delete selected messages, move messages to another destination, export message contents in XML format to another file, import XML formatted message contents from another file, or drain all the messages from a destination.

Click on a message to view its contents.

Message Selector:

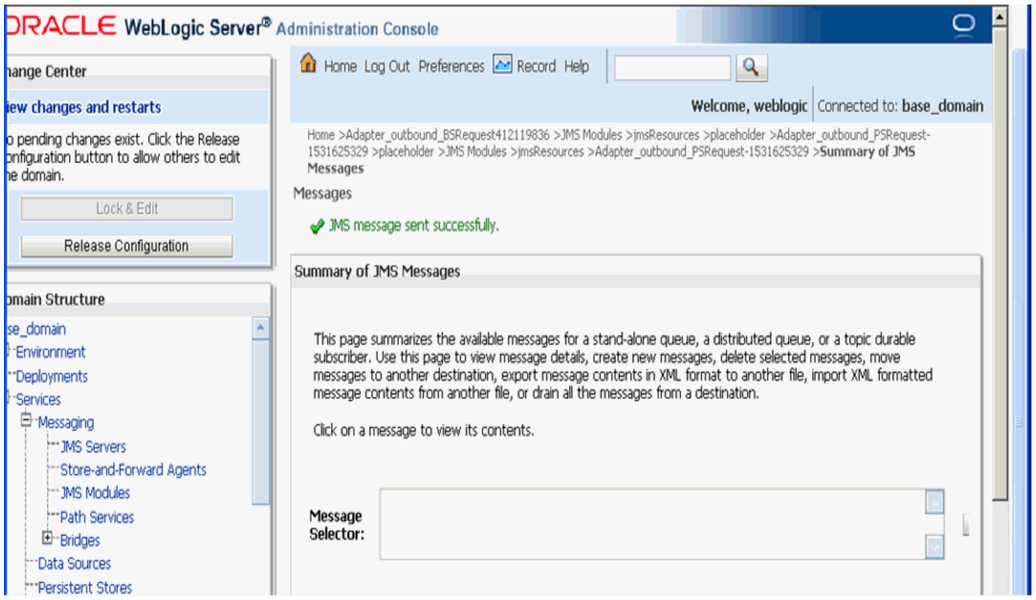
JMS Messages (Filtered - More Columns Exist)

☐	ID	CorrId	Time Stamp	State String	JMS Delivery Mode	Message Size
There are no items to display						

29. Provide the input payload in the **Body** field and click **OK**.

A Success message appears, as shown in Figure 7–124.

Figure 7–124 JMS Success Message



o pending changes exist. Click the Release
onfiguration button to allow others to edit
e domain.

Home >Adapter_outbound_BSRequest412119836 >JMS Modules >jmsResources >placeholder >Adapter_outbound_PSRequest-1531625329 >placeholder >JMS Modules >jmsResources >Adapter_outbound_PSRequest-1531625329 >Summary of JMS Messages

Messages

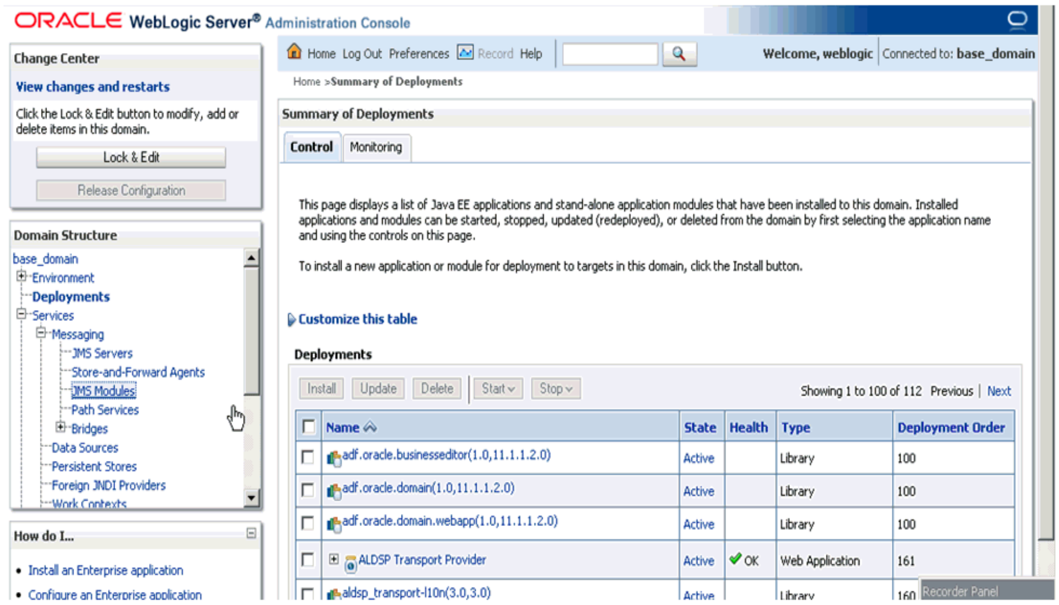
✓ JMS message sent successfully.

This page summarizes the available messages for a stand-alone queue, a distributed queue, or a topic durable subscriber. Use this page to view message details, create new messages, delete selected messages, move messages to another destination, export message contents in XML format to another file, import XML formatted message contents from another file, or drain all the messages from a destination.

Click on a message to view its contents.

30. In the Oracle WLS Console, expand **Services**, expand **Messaging**, and click **JMS Modules**, as shown in Figure 7–125.

Figure 7–125 JMS Modules



Click the Lock & Edit button to modify, add or delete items in this domain.

- Install an Enterprise application
- Configure an Enterprise application

Home >Summary of Deployments

Summary of Deployments

This page displays a list of Java EE applications and stand-alone application modules that have been installed to this domain. Installed applications and modules can be started, stopped, updated (redeployed), or deleted from the domain by first selecting the application name and using the controls on this page.

To install a new application or module for deployment to targets in this domain, click the Install button.

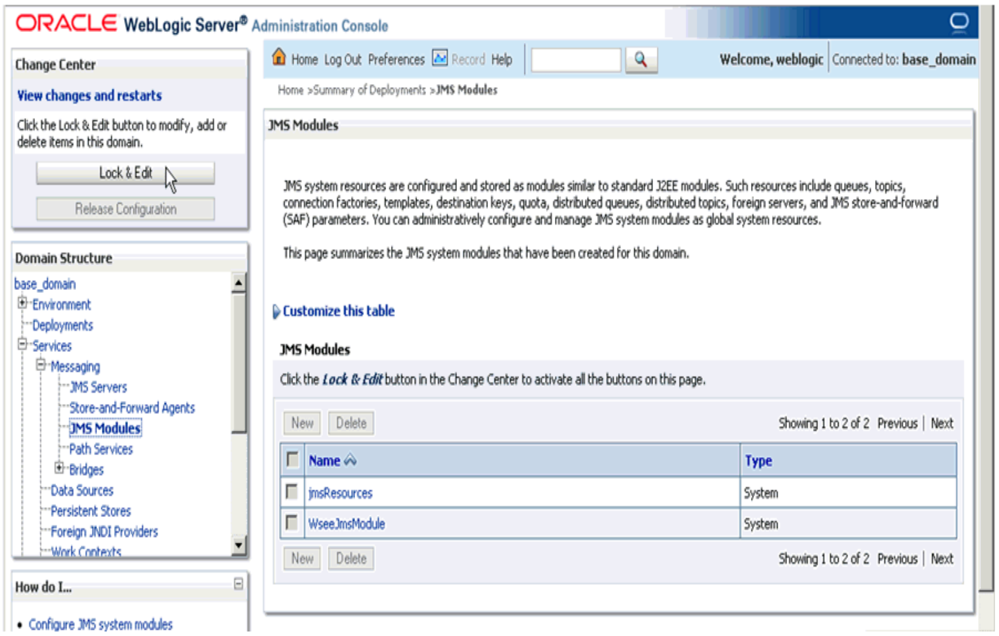
Deployments

Showing 1 to 100 of 112

Name	State	Health	Type	Deployment Order
adf.oracle.businesseditor(1.0,11.1.1.2.0)	Active		Library	100
adf.oracle.domain(1.0,11.1.1.2.0)	Active		Library	100
adf.oracle.domain.webapp(1.0,11.1.1.2.0)	Active		Library	100
ALDSP Transport Provider	Active	✓ OK	Web Application	161
aldsp_transport-l10n(3.0,3.0)	Active		Library	160

31. Click **jmsResources**, as shown in Figure 7–126.

Figure 7–126 jmsResources



Click the Lock & Edit button to modify, add or delete items in this domain.

JMS Modules

JMS system resources are configured and stored as modules similar to standard J2EE modules. Such resources include queues, topics, connection factories, templates, destination keys, quota, distributed queues, distributed topics, foreign servers, and JMS store-and-forward (SAF) parameters. You can administratively configure and manage JMS system modules as global system resources.

This page summarizes the JMS system modules that have been created for this domain.

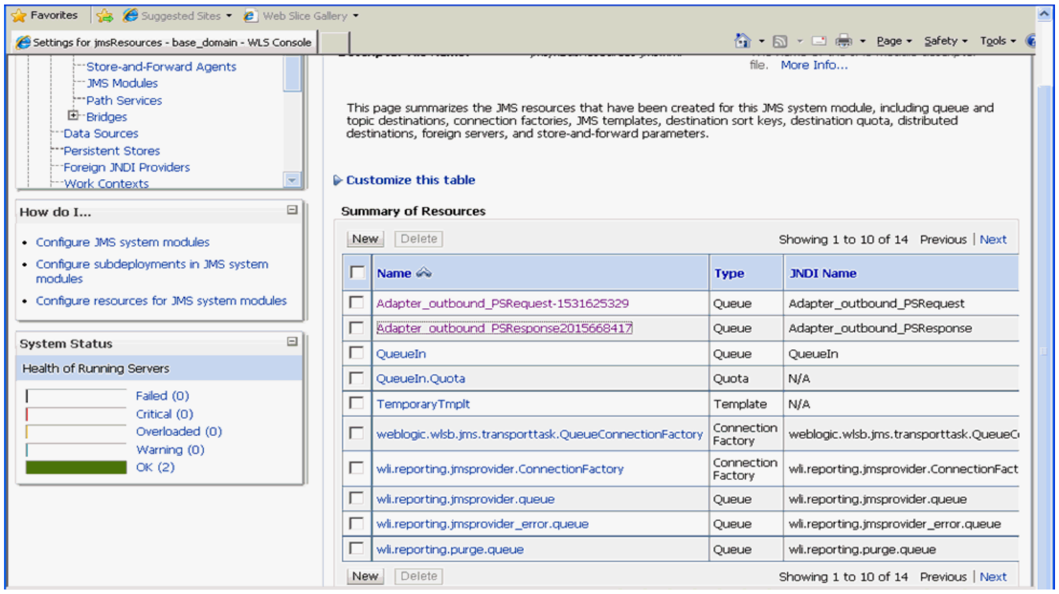
JMS Modules

Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

Name	Type
jmsResources	System
WseeJmsModule	System

32. Click the appropriate response link, for example, **Adapter_outbound_PSResponse**, as shown in Figure 7–127.

Figure 7–127 Summary of Resources



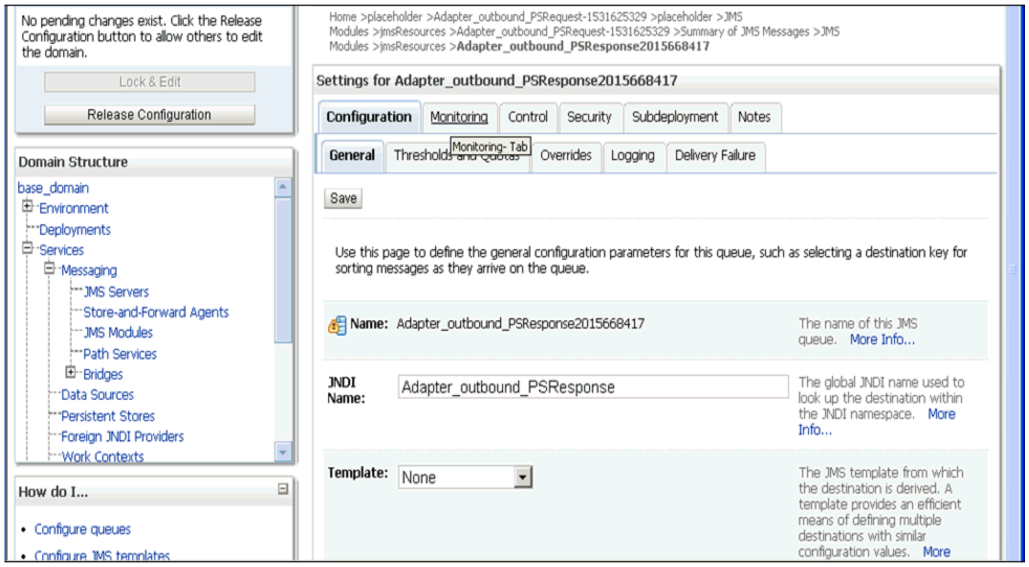
This page summarizes the JMS resources that have been created for this JMS system module, including queue and topic destinations, connection factories, JMS templates, destination sort keys, destination quota, distributed destinations, foreign servers, and store-and-forward parameters.

Summary of Resources

Name	Type	JNDI Name
Adapter_outbound_PSRequest-1531625329	Queue	Adapter_outbound_PSRequest
Adapter_outbound_PSResponse2015668417	Queue	Adapter_outbound_PSResponse
QueueIn	Queue	QueueIn
QueueIn.Quota	Quota	N/A
TemporaryTmplt	Template	N/A
weblogic.wlsb.jms.transporttask.QueueConnectionFactory	Connection Factory	weblogic.wlsb.jms.transporttask.QueueC
wli.reporting.jmsprovider.ConnectionFactory	Connection Factory	wli.reporting.jmsprovider.ConnectionFact
wli.reporting.jmsprovider.queue	Queue	wli.reporting.jmsprovider.queue
wli.reporting.jmsprovider_error.queue	Queue	wli.reporting.jmsprovider_error.queue
wli.reporting.purge.queue	Queue	wli.reporting.purge.queue

33. Click the **Monitoring** tab, as shown in Figure 7–128.

Figure 7–128 Monitoring Tab



No pending changes exist. Click the Release Configuration button to allow others to edit the domain.

Lock & Edit

Release Configuration

Domain Structure

base_domain
- Environment
- Deployments
- Services
  - Messaging
    - JMS Servers
    - Store-and-Forward Agents
    - JMS Modules
    - Path Services
    - Bridges
  - Data Sources
  - Persistent Stores
  - Foreign JNDI Providers
  - Work Contexts

How do I...

- Configure queues
- Configure JMS templates

Home >placeholder >Adapter_outbound_PSRequest-1531625329 >placeholder >JMS Modules >jmsResources >Adapter_outbound_PSRequest-1531625329 >Summary of JMS Messages >JMS Modules >jmsResources >**Adapter_outbound_PSResponse2015668417**

Settings for Adapter_outbound_PSResponse2015668417

Configuration | Monitoring | Control | Security | Subdeployment | Notes

General | Thresholds and Quotas | Overrides | Logging | Delivery Failure

Monitoring- Tab

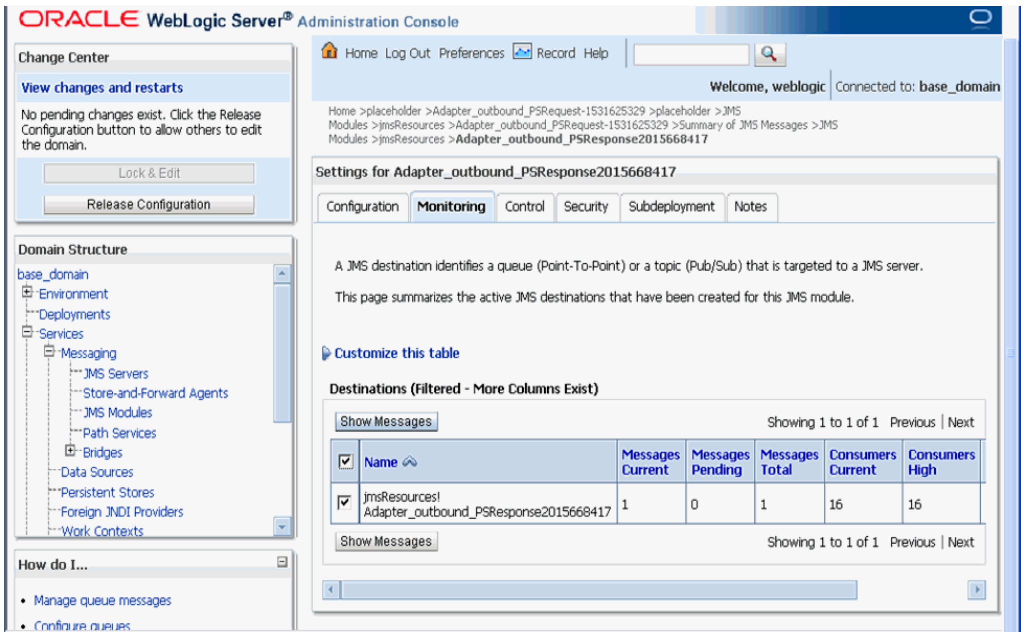
Save

Use this page to define the general configuration parameters for this queue, such as selecting a destination key for sorting messages as they arrive on the queue.

Name:	Adapter_outbound_PSResponse2015668417	The name of this JMS queue. More Info...
JNDI Name:	Adapter_outbound_PSResponse	The global JNDI name used to look up the destination within the JNDI namespace. More Info...
Template:	None	The JMS template from which the destination is derived. A template provides an efficient means of defining multiple destinations with similar configuration values. More

34. Enable the check box and click **Show Messages**, as shown in Figure 7–129.

Figure 7–129 Destination Messages



ORACLE WebLogic Server® Administration Console

Change Center

View changes and restarts

No pending changes exist. Click the Release Configuration button to allow others to edit the domain.

Lock & Edit

Release Configuration

Domain Structure

base_domain
- Environment
- Deployments
- Services
  - Messaging
    - JMS Servers
    - Store-and-Forward Agents
    - JMS Modules
    - Path Services
    - Bridges
  - Data Sources
  - Persistent Stores
  - Foreign JNDI Providers
  - Work Contexts

How do I...

- Manage queue messages
- Configure queues

Home Log Out Preferences Record Help

Welcome, weblogic | Connected to: base_domain

Home >placeholder >Adapter_outbound_PSRequest-1531625329 >placeholder >JMS Modules >jmsResources >Adapter_outbound_PSRequest-1531625329 >Summary of JMS Messages >JMS Modules >jmsResources >**Adapter_outbound_PSResponse2015668417**

Settings for Adapter_outbound_PSResponse2015668417

Configuration | Monitoring | Control | Security | Subdeployment | Notes

A JMS destination identifies a queue (Point-To-Point) or a topic (Pub/Sub) that is targeted to a JMS server.

This page summarizes the active JMS destinations that have been created for this JMS module.

Customize this table

Destinations (Filtered - More Columns Exist)

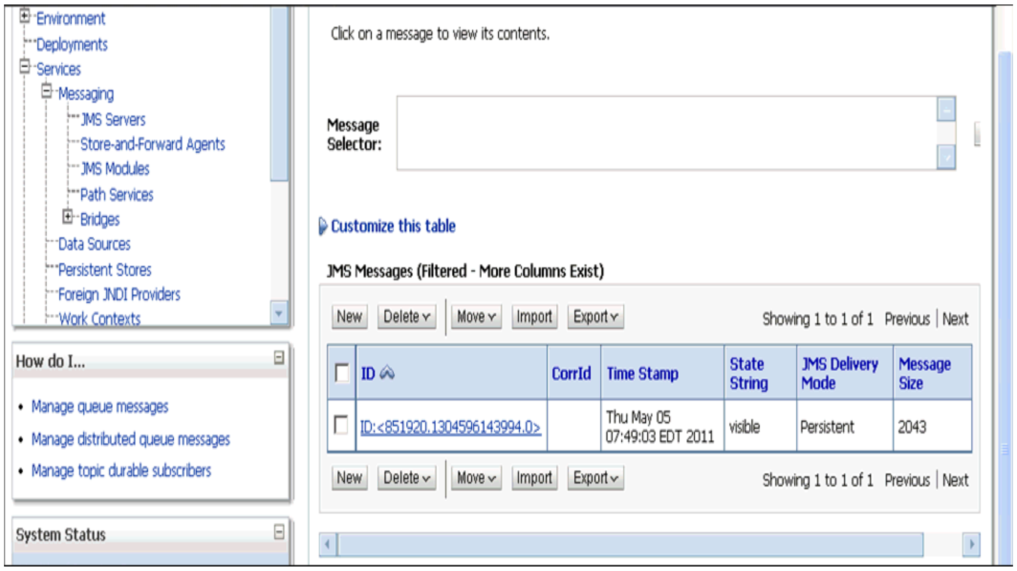
Show Messages — Showing 1 to 1 of 1 Previous | Next

☑	Name	Messages Current	Messages Pending	Messages Total	Consumers Current	Consumers High
☑	jmsResources!Adapter_outbound_PSResponse2015668417	1	0	1	16	16

Show Messages — Showing 1 to 1 of 1 Previous | Next

35. Click the **ID** link, as shown in Figure 7–130.

Figure 7–130 JMS Messages



Click on a message to view its contents.

Message Selector:

JMS Messages (Filtered - More Columns Exist)

	ID	CorrId	Time Stamp	State String	JMS Delivery Mode	Message Size
☐	ID:<851920.1304596143994.0>		Thu May 05 07:49:03 EDT 2011	visible	Persistent	2043

The response document is shown under the Text field.

7.6 Configuring HTTP Proxy Services Using Oracle Service Bus (J2CA Configuration)

This section describes how to configure HTTP Proxy Services using Oracle Service Bus for a J2CA configuration.

1. Start the Oracle Service Bus and create the required project folders.

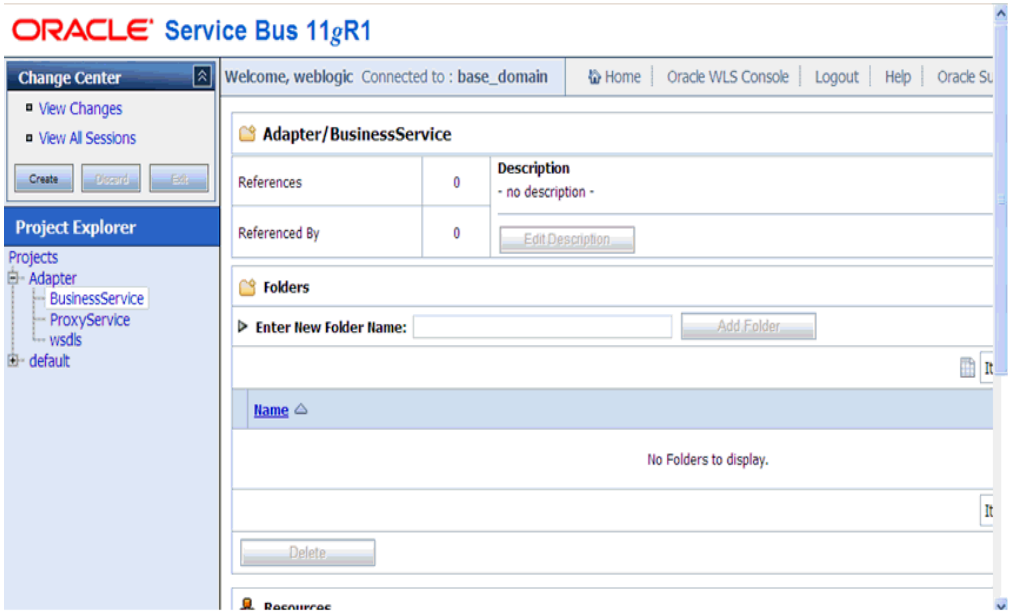
   For more information, see Section 7.2.1, "Starting Oracle Service Bus and Creating Project Folders" on page 7-2.

2. Generate and publish the WSDL from Application Explorer to the created project folder, and create a Business Service using the published WSDL.

   For more information, see Section 7.2.3, "Publishing a WSDL From Application Explorer to Oracle Service Bus" on page 7-6.

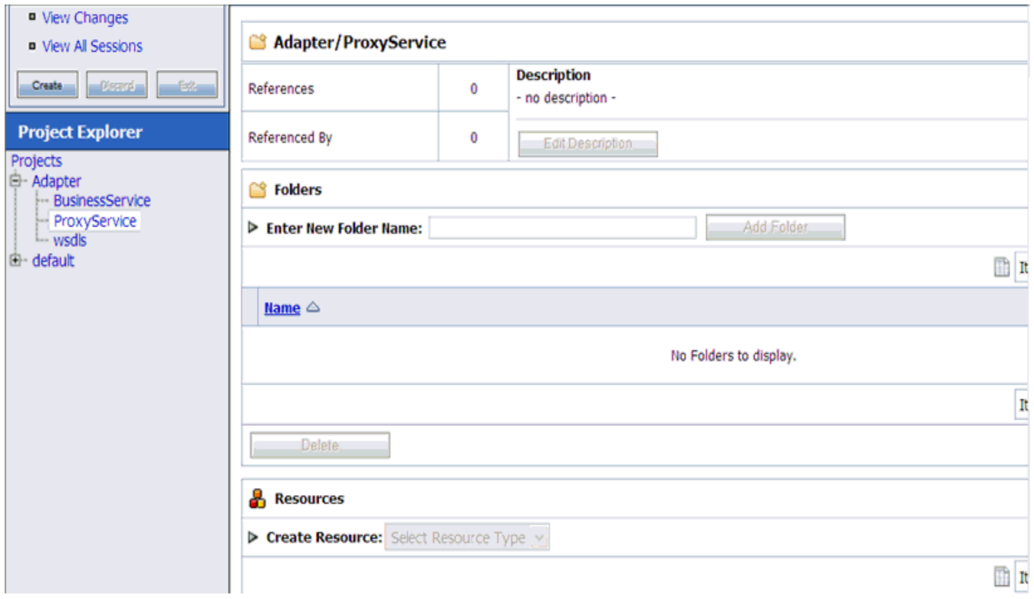
3. Open the Service Bus console page, as shown in Figure 7–131.

Figure 7–131 Service Bus Console Page



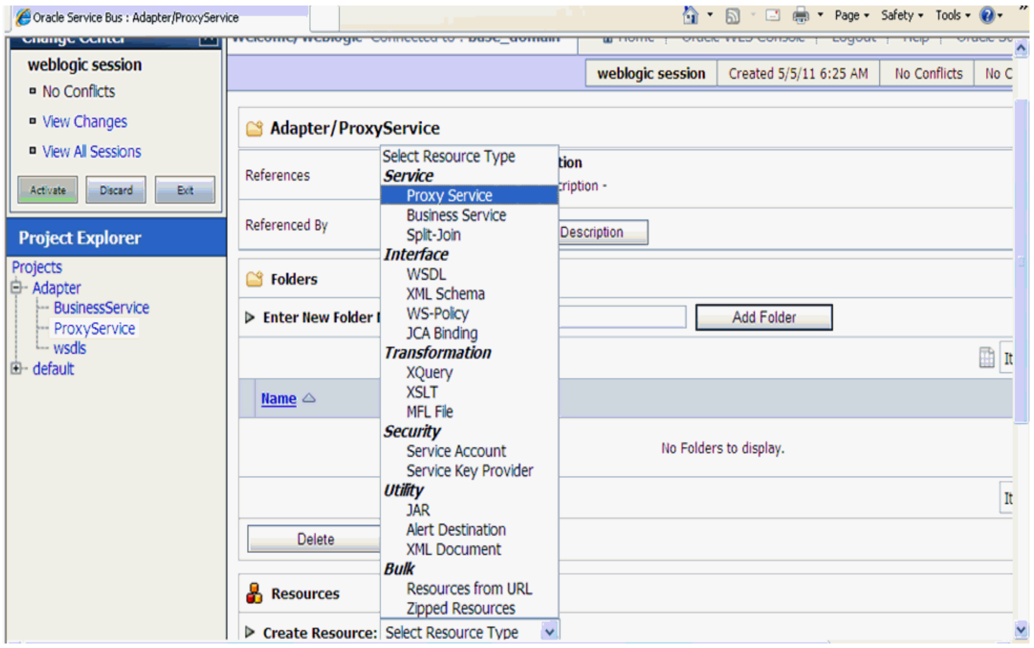
4. In the Project Explorer, select the **ProxyService** project folder, and click **Create**, as shown in Figure 7–132.

Figure 7–132 Project Explorer



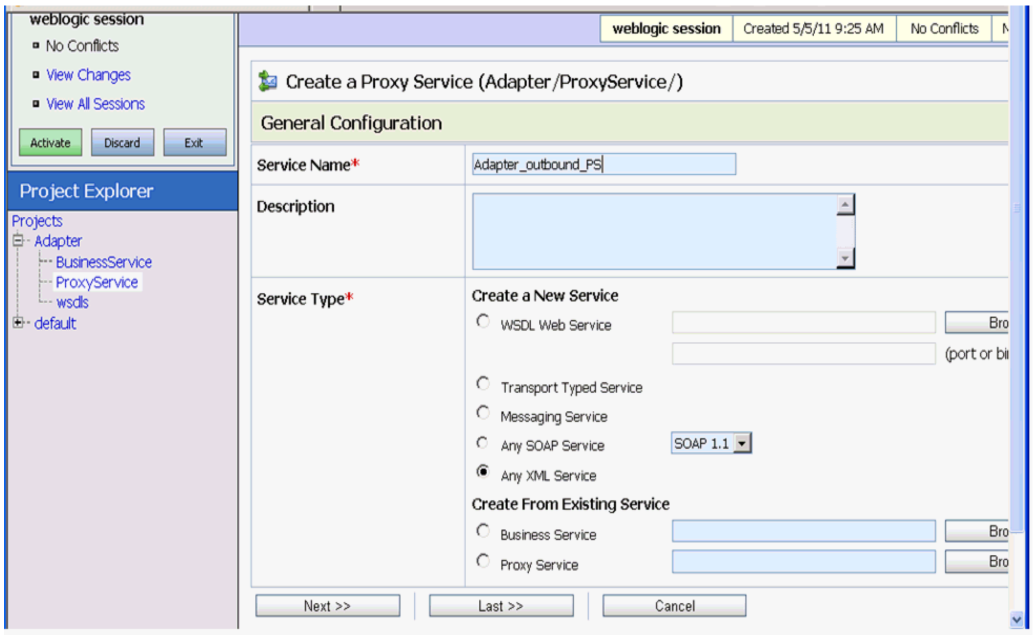
5. In the Create Resource list on the right pane, select **Proxy Service**, as shown in Figure 7–133.

Figure 7–133 Proxy Service



6. In the Service Name field, enter an appropriate name, as shown in Figure 7–134.

Figure 7–134 Service Name

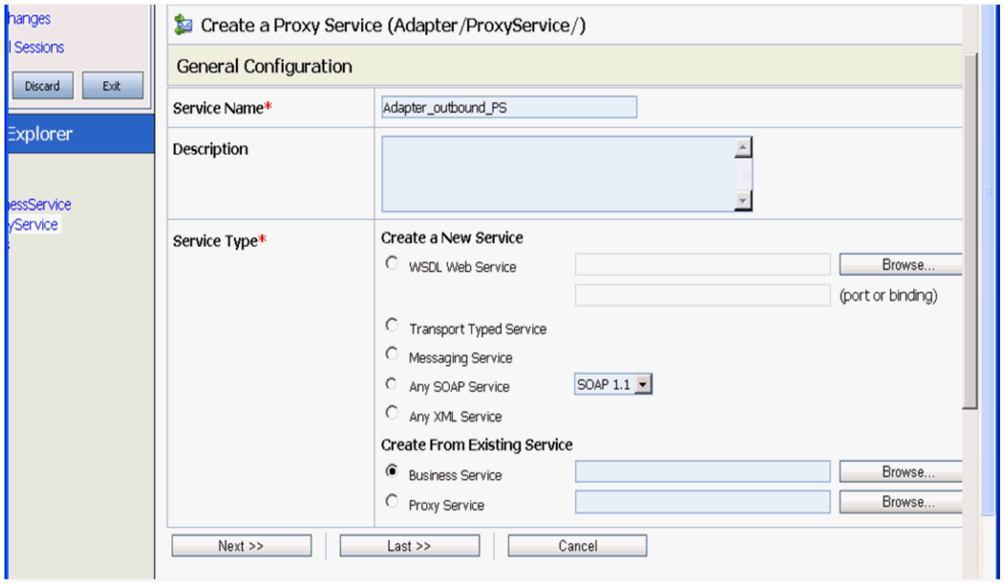


Service Name*: Adapter_outbound_PS

Service Type*: Any XML Service

7. In the Service Type section, under Create From Existing Service, select the **Business Service** radio button and click **Browse**, as shown in Figure 7–135.

Figure 7–135 General Configuration



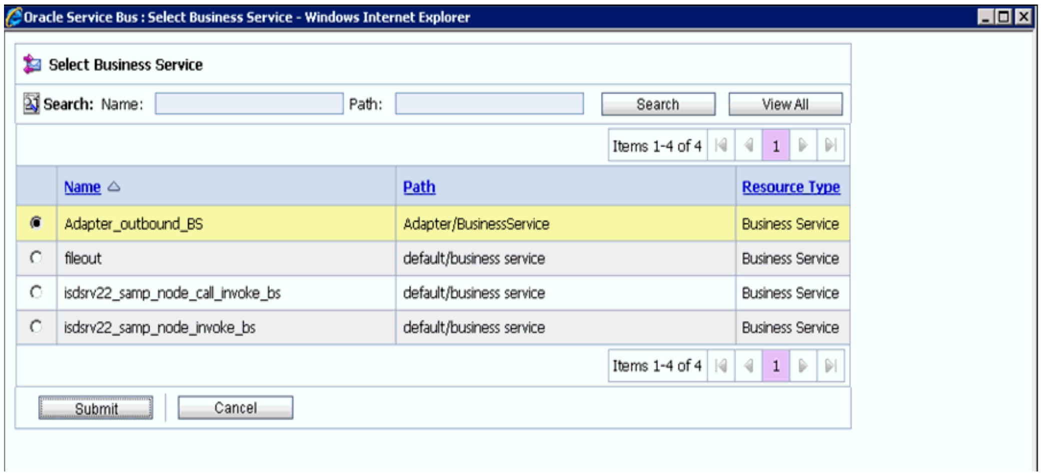
Create a Proxy Service (Adapter/ProxyService/)

General Configuration

Service Name*	Adapter_outbound_PS
Description	
Service Type*	Create a New Service
	○ WSDL Web Service (port or binding)
	○ Transport Typed Service
	○ Messaging Service
	○ Any SOAP Service SOAP 1.1
	○ Any XML Service
	Create From Existing Service
	● Business Service
	○ Proxy Service

8. Select the existing Business Service and click **Submit**, as shown in Figure 7–136.

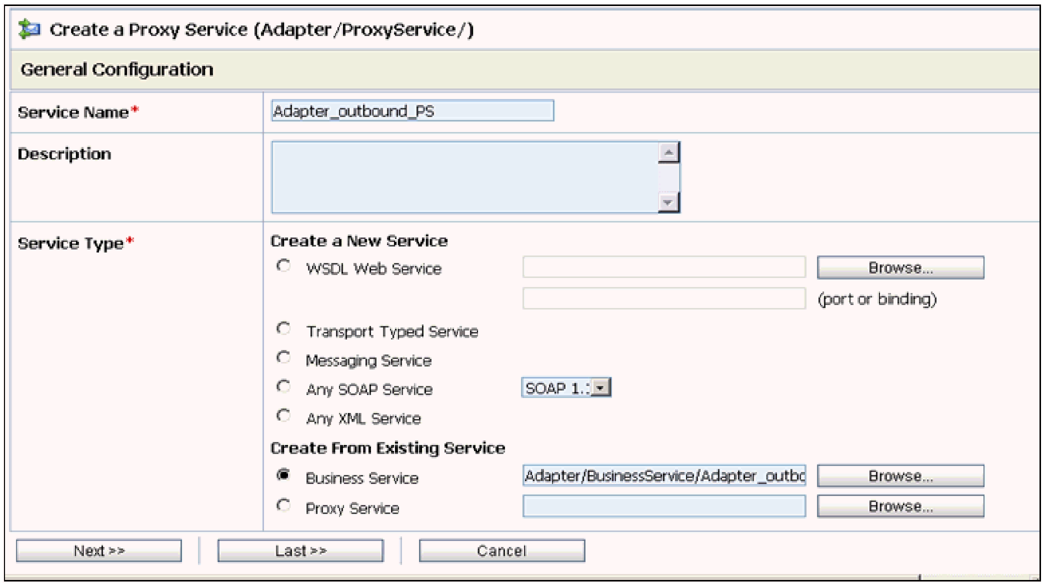
Figure 7–136 Business Service



	Name	Path	Resource Type
●	Adapter_outbound_BS	Adapter/BusinessService	Business Service
○	fileout	default/business service	Business Service
○	isdsrv22_samp_node_call_invoke_bs	default/business service	Business Service
○	isdsrv22_samp_node_invoke_bs	default/business service	Business Service

9. Click **Next**, as shown in Figure 7–137.

Figure 7–137 General Configuration



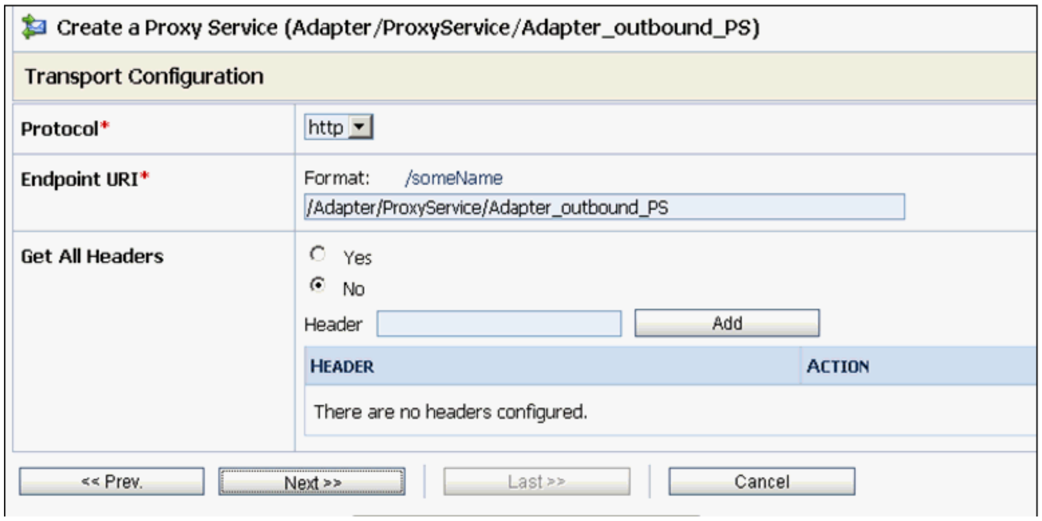
Create a Proxy Service (Adapter/ProxyService/)

General Configuration

Service Name*	Adapter_outbound_PS
Description	
Service Type*	Create a New Service: WSDL Web Service (port or binding); Transport Typed Service; Messaging Service; Any SOAP Service (SOAP 1.1); Any XML Service. Create From Existing Service: [✓ Business Service] Adapter/BusinessService/Adapter_outbc; Proxy Service

10. Select **http** in the Protocol list and click **Next**, as shown in Figure 7–138.

Figure 7–138 Transport Configuration



Create a Proxy Service (Adapter/ProxyService/Adapter_outbound_PS)

Transport Configuration

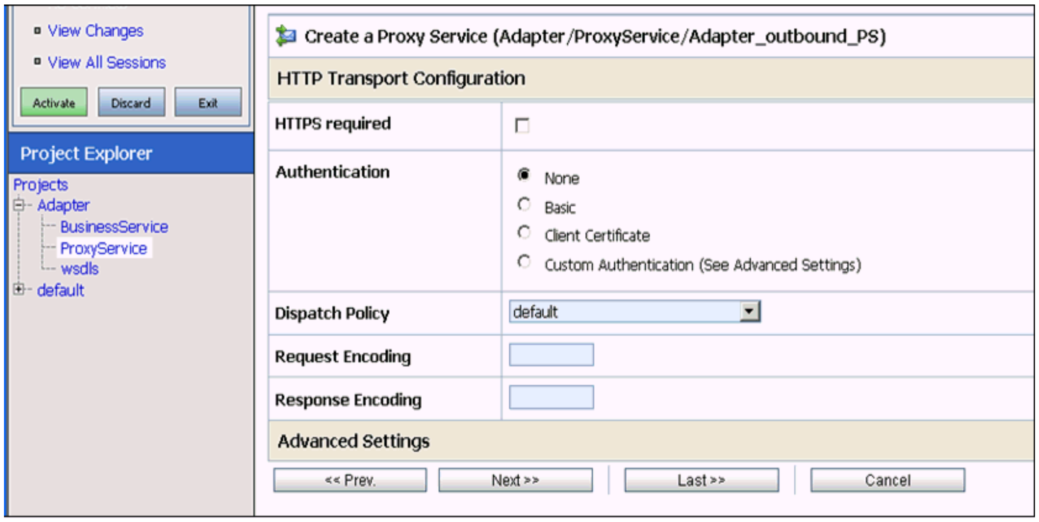
Protocol*	http
Endpoint URI*	Format: /someName /Adapter/ProxyService/Adapter_outbound_PS
Get All Headers	Yes [✓ No]

Header

HEADER	ACTION
There are no headers configured.	

11. Click **Next**, as shown in Figure 7–139.

Figure 7–139 HTTP Transport Configuration



Create a Proxy Service (Adapter/ProxyService/Adapter_outbound_PS)

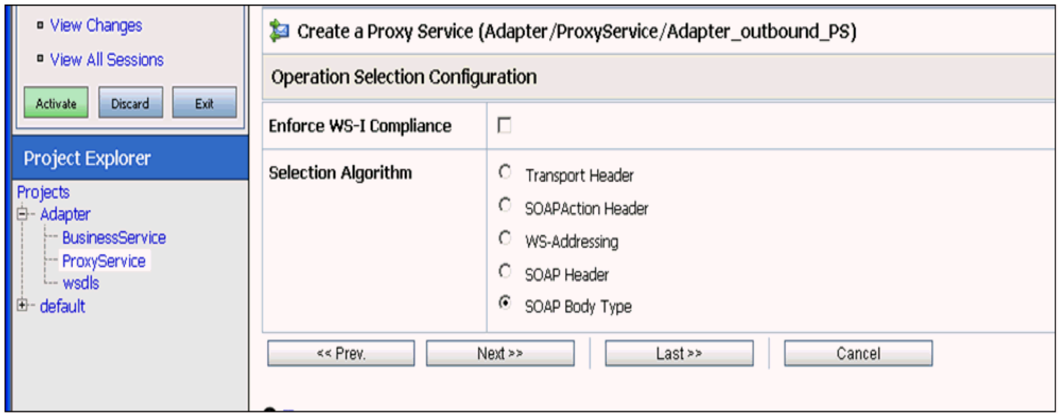
HTTP Transport Configuration

HTTPS required	☐
Authentication	[✓ None] Basic Client Certificate Custom Authentication (See Advanced Settings)
Dispatch Policy	default
Request Encoding	
Response Encoding	

Advanced Settings

12. Click **Next**, as shown in Figure 7–140.

Figure 7–140 Operation Selection Configuration



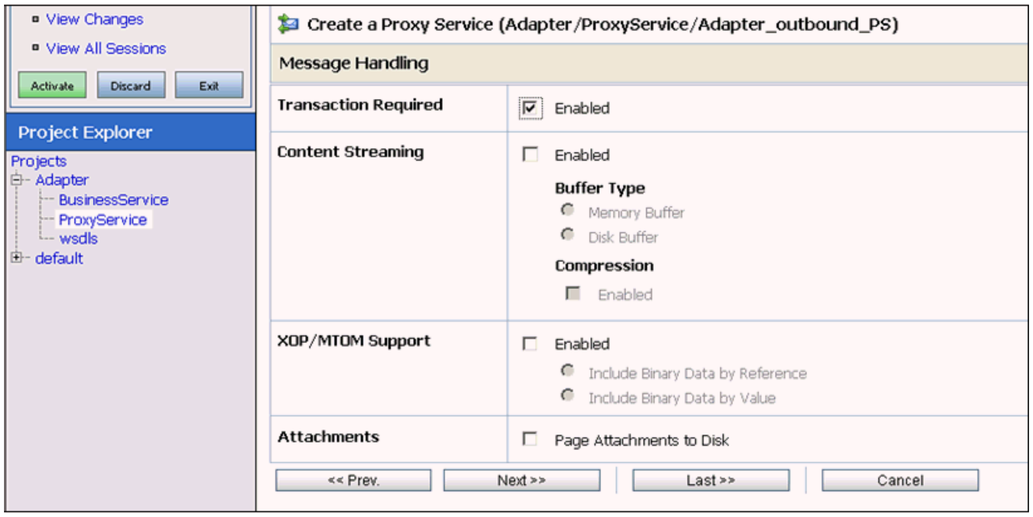
Create a Proxy Service (Adapter/ProxyService/Adapter_outbound_PS)

Operation Selection Configuration

Enforce WS-I Compliance	☐
Selection Algorithm	Transport Header SOAPAction Header WS-Addressing SOAP Header [✓ SOAP Body Type]

13. Enable the **Transaction Required** check box and click **Next**, as shown in Figure 7–141.

Figure 7–141 Message Handling



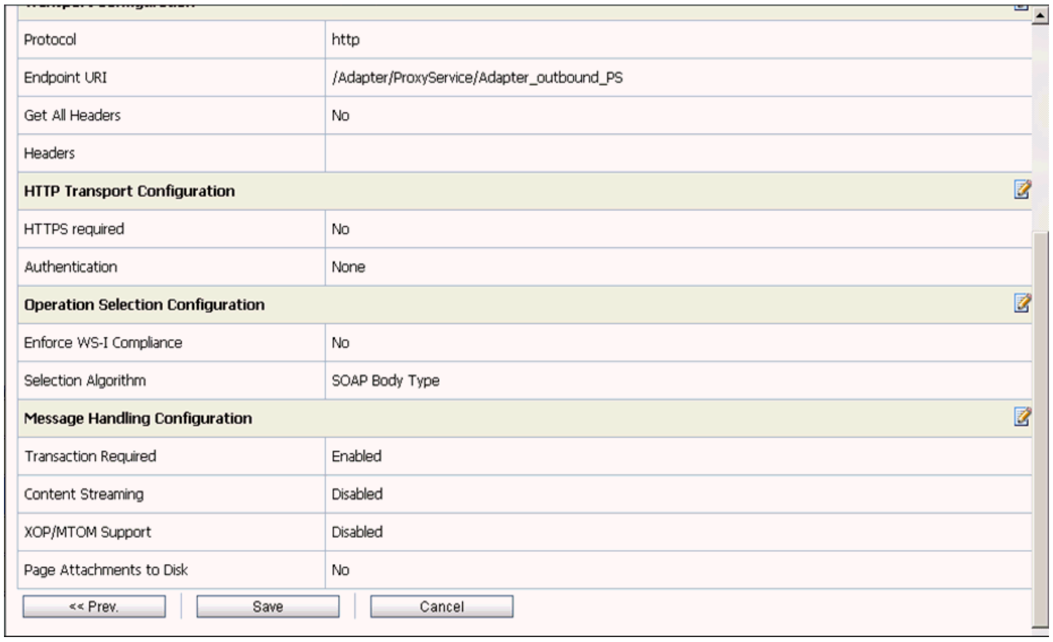
Create a Proxy Service (Adapter/ProxyService/Adapter_outbound_PS)

Message Handling

Transaction Required	☑ Enabled
Content Streaming	☐ Enabled **Buffer Type** ○ Memory Buffer ○ Disk Buffer **Compression** ☐ Enabled
XOP/MTOM Support	☐ Enabled ○ Include Binary Data by Reference ○ Include Binary Data by Value
Attachments	☐ Page Attachments to Disk

14. Click **Save**, as shown in Figure 7–142.

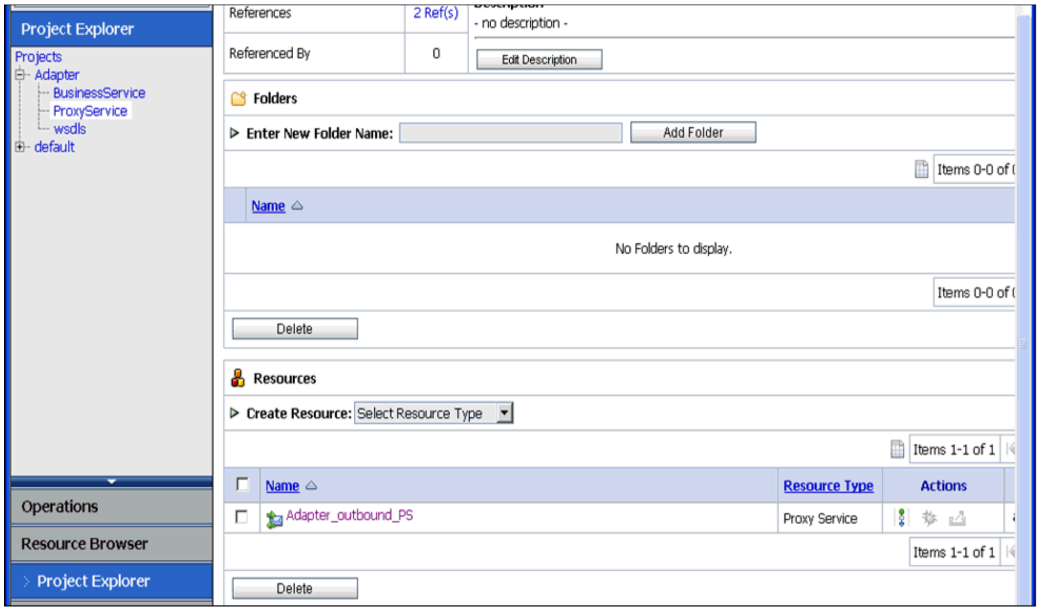
Figure 7–142 Save



Protocol	http
Endpoint URI	/Adapter/ProxyService/Adapter_outbound_PS
Get All Headers	No
Headers	
HTTP Transport Configuration	
HTTPS required	No
Authentication	None
Operation Selection Configuration	
Enforce WS-I Compliance	No
Selection Algorithm	SOAP Body Type
Message Handling Configuration	
Transaction Required	Enabled
Content Streaming	Disabled
XOP/MTOM Support	Disabled
Page Attachments to Disk	No

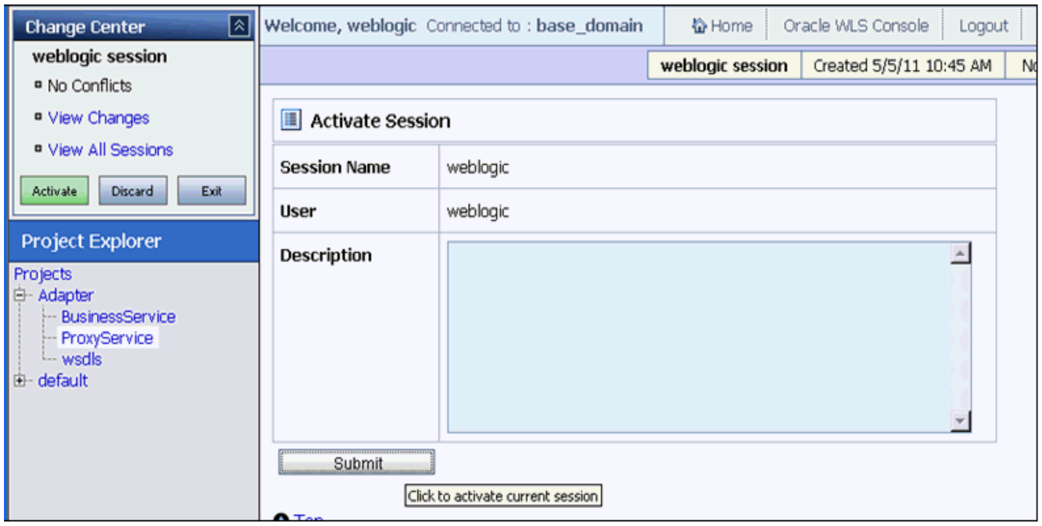
The created Proxy Service is saved, as shown in Figure 7–143.

Figure 7–143 Proxy Service



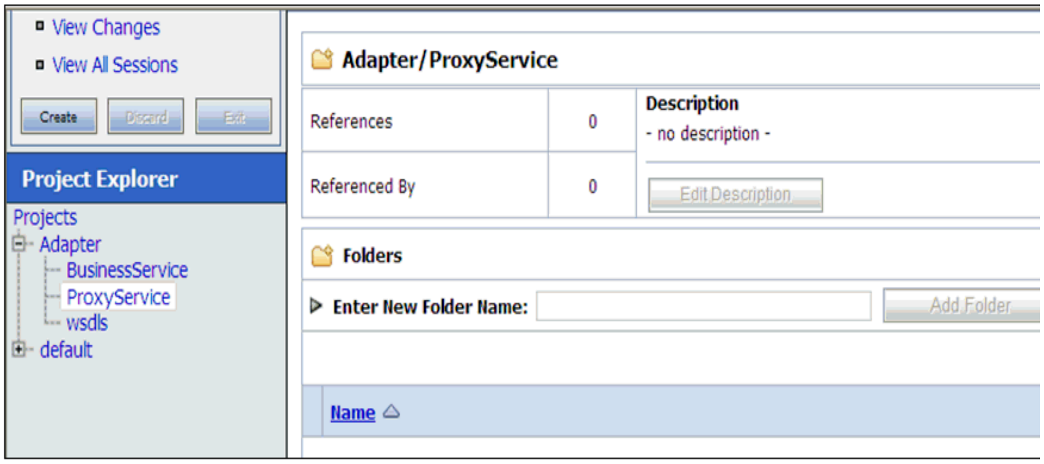
15. Click **Activate** in the left pane, and then **Submit** on the right pane, as shown in Figure 7–144.

Figure 7–144 Activate Session



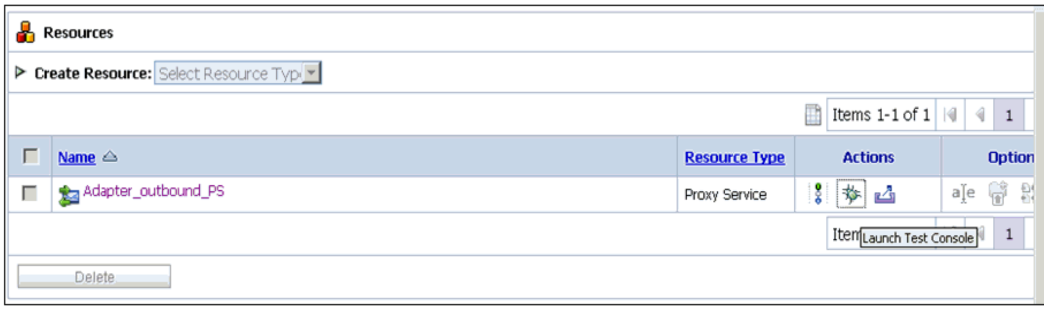
16. Click **ProxyService** in the Projects folder on the left pane, as shown in Figure 7–145.

Figure 7–145 ProxyService



17. Click the **Launch Test Console** icon for the created Proxy Service, as shown in Figure 7–146.

Figure 7–146 Launch Test Console



18. Uncheck the **Direct Call** check box, provide the input values for **Payload**, and click **Execute**.

19. Review the **Response Document**.

8 Configuring an Outbound and Inbound Process for Oracle Service Bus Using JDeveloper

Oracle Application Adapter for J.D. Edwards OneWorld integrates seamlessly with Oracle JDeveloper to facilitate Web service integration.

This chapter contains the following sections:

- Section 8.1, "Configuring an OSB Outbound Process Using JDeveloper (J2CA Configuration)"
- Section 8.2, "Configuring an OSB Inbound Process Using JDeveloper (J2CA Configuration)"
- Section 8.3, "Configuring an OSB Outbound Process Using JDeveloper (BSE Configuration)"
- Section 8.4, "Configuring a JMS Inbound Process Using JDeveloper (J2CA Configuration)"
- Section 8.5, "Configuring a JMS Outbound Process Using JDeveloper (J2CA Configuration)"
- Section 8.6, "Configuring an HTTP Outbound Process Using JDeveloper (J2CA Configuration)"

8.1 Configuring an OSB Outbound Process Using JDeveloper (J2CA Configuration)

This section describes how to configure an OSB outbound process to your J.D. Edwards OneWorld system, using Oracle JDeveloper for J2CA configurations.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\OSB_
Jdeveloper\J2CA\JDEdwards_Sample_J2CA_OSB_Outbound_Project
```

This section includes the following topics:

- Section 8.1.1, "Creating a Service Bus Application for OSB"
- Section 8.1.2, "Defining an OSB Outbound Process"
- Section 8.1.3, "Deploying the OSB Outbound Process"

Prerequisites

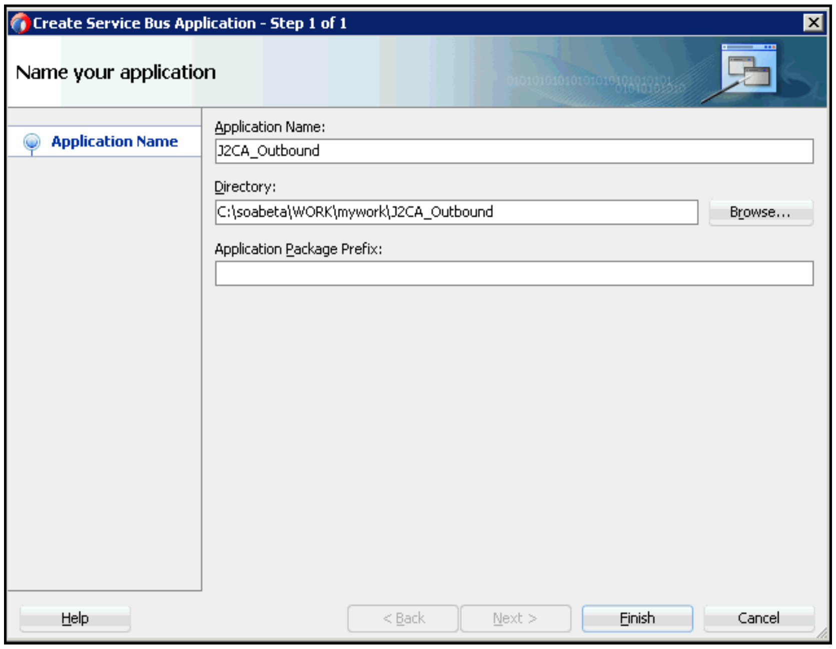
Before you design an OSB outbound process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.4.1, "Generating WSDL for Request/Response Service" on page 4-8.

8.1.1 Creating a Service Bus Application for OSB

Perform the following steps in JDeveloper to create a service bus application for OSB.

1. Create a new OSB application.
2. Enter a name for the OSB Application (for example, J2CA_Outbound) and click **Finish**, as shown in Figure 8–1.

Figure 8–1 Name Your Application Pane



Create Service Bus Application - Step 1 of 1

Name your application

Application Name

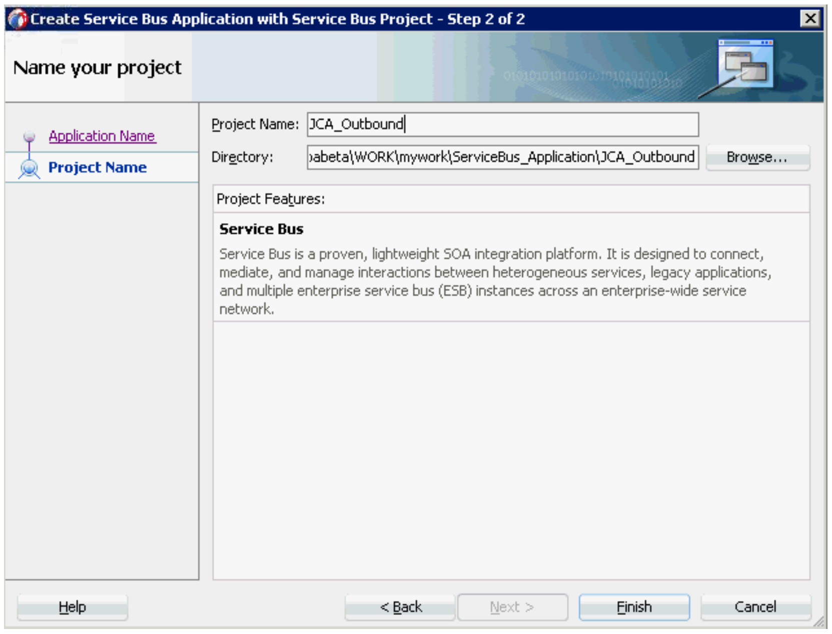
Application Name: J2CA_Outbound

Directory: C:\soabeta\WORK\mywork\J2CA_Outbound

Application Package Prefix:

3. Enter a project name (for example, JCA_Outbound), and click **Finish**, as shown in Figure 8–2.

Figure 8–2 Name Your Project Pane



Project Name: JCA_Outbound

Directory: ›abeta\WORK\mywork\ServiceBus_Application\JCA_Outbound

Project Features:

Service Bus

Service Bus is a proven, lightweight SOA integration platform. It is designed to connect, mediate, and manage interactions between heterogeneous services, legacy applications, and multiple enterprise service bus (ESB) instances across an enterprise-wide service network.

8.1.2 Defining an OSB Outbound Process

This section describes how to define an OSB outbound process. The following topics are included:

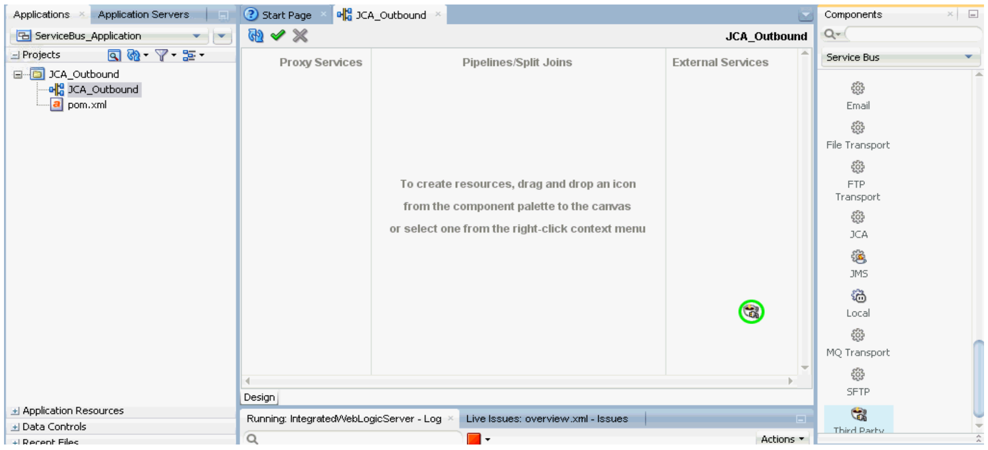
- Section 8.1.2.1, "Configuring a Third-Party Adapter Service Component"
- Section 8.1.2.2, "Configuring a File Transport Type Business Service"
- Section 8.1.2.3, "Creating a Proxy Service With Pipeline"
- Section 8.1.2.4, "Configuring the Routing Rules"

8.1.2.1 Configuring a Third-Party Adapter Service Component

Perform the following steps to create a third party adapter service component along with the Business Service:

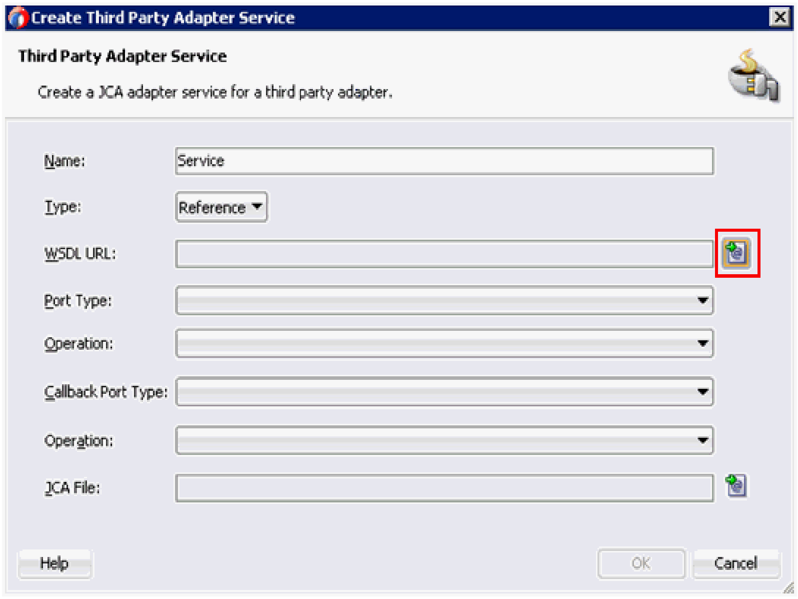
1. Drag and drop the **Third Party Adapter** component from the Service Bus Components pane to the External Services pane, as shown in Figure 8–3.

Figure 8–3 Third Party Adapter Component



The Create Third Party Adapter Service dialog is displayed, as shown in Figure 8–4.

Figure 8–4 Create Third Party Adapter Service Pane


Create Third Party Adapter Service

Third Party Adapter Service

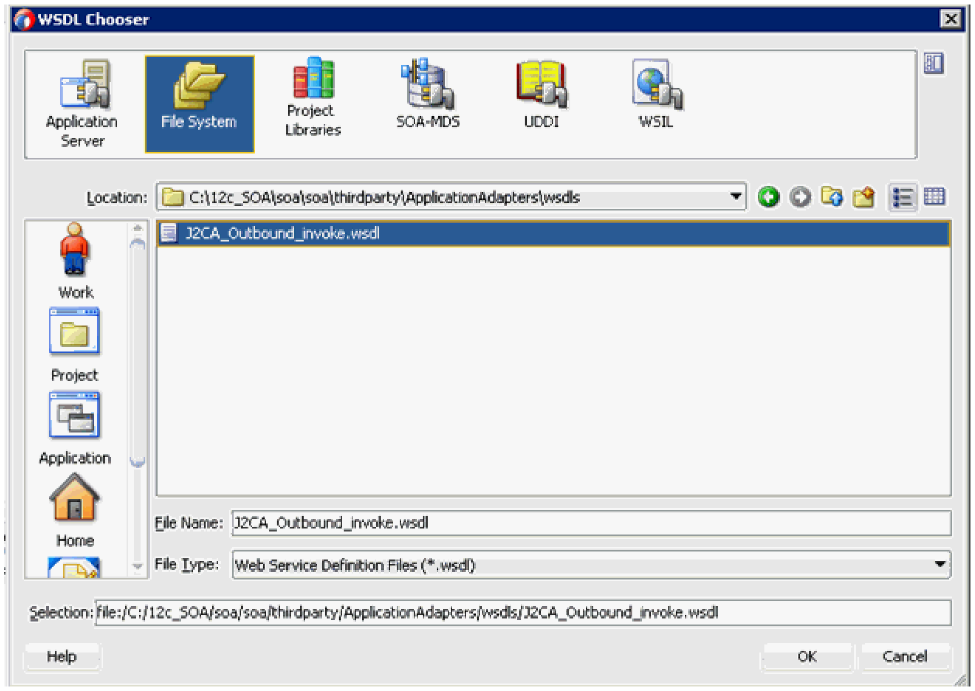
Create a JCA adapter service for a third party adapter.

Name:	Service
Type:	Reference
WSDL URL:	
Port Type:	
Operation:	
Callback Port Type:	
Operation:	
JCA File:	

2. Enter an appropriate name for the Third Party Adapter Service which will be used as the Business Service name.
3. Ensure that **Reference** is selected from the Type drop-down list (by default).
4. Click the Find existing WSDLs icon, which is located to the right of the WSDL URL field.

   The WSDL Chooser dialog is displayed, as shown in Figure 8–5.

Figure 8–5 WSDL Chooser Dialog



Location: C:\12c_SOA\soa\soa\thirdparty\ApplicationAdapters\wsdls

J2CA_Outbound_invoke.wsdl

File Name: J2CA_Outbound_invoke.wsdl

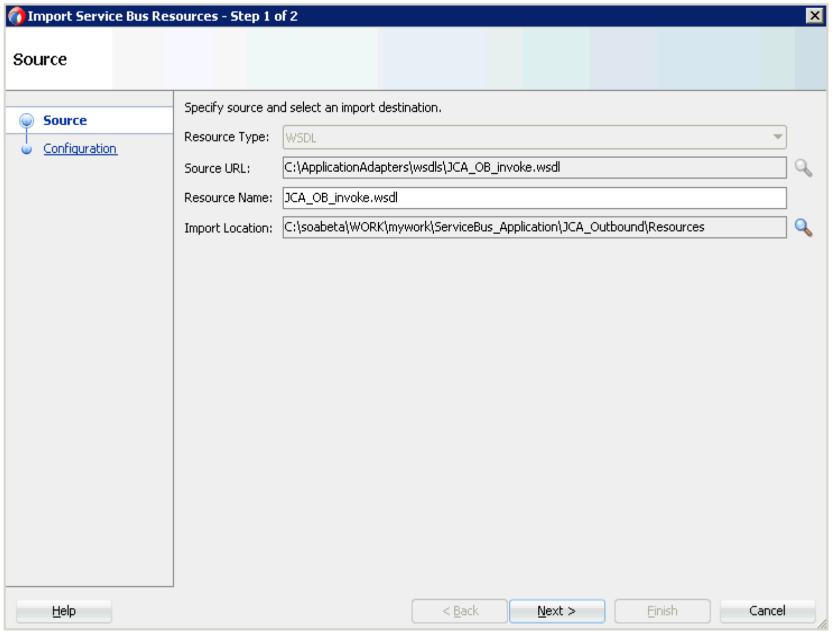
File Type: Web Service Definition Files (*.wsdl)

Selection: file:/C:/12c_SOA/soa/soa/thirdparty/ApplicationAdapters/wsdls/J2CA_Outbound_invoke.wsdl

5. Select the **File System** tab, then browse, and select an outbound WSDL file from the WSDL directory.
6. Click **OK**.

   The Import Service Bus Resources dialog is displayed.
7. Click **Next**, as shown in Figure 8–6.

Figure 8–6 Source Pane



Specify source and select an import destination.

Resource Type: WSDL

Source URL: C:\ApplicationAdapters\wsdls\JCA_OB_invoke.wsdl

Resource Name: JCA_OB_invoke.wsdl

Import Location: C:\soabeta\WORK\mywork\ServiceBus_Application\JCA_Outbound\Resources

8. In the Configuration pane, click **Finish**.

You are returned to the Create Third Party Adapter Service Dialog.

9. Click the Find JCA file icon which is located to the right of the JCA File field.

   The Transformation Chooser dialog is displayed.

10. Select the JCA properties file from the WSDL directory.

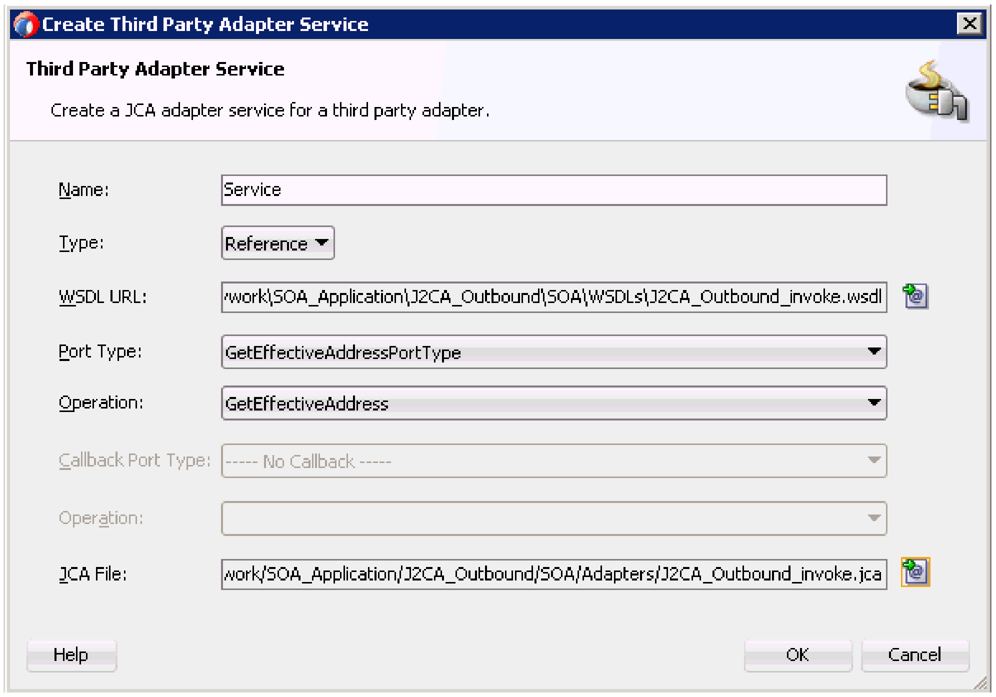
11. Click **OK**. The Copy File message is displayed.

12. Click **Yes**.

    A copy of the JCA properties file is made in the project folder.

    You are returned to the Create Third Party Adapter Service dialog, as shown in Figure 8–7.

Figure 8–7 Create Third Party Adapter Service Dialog



Create Third Party Adapter Service

Third Party Adapter Service

Create a JCA adapter service for a third party adapter.

Name:	Service
Type:	Reference
WSDL URL:	work\SOA_Application\J2CA_Outbound\SOA\WSDLs\J2CA_Outbound_invoke.wsdl
Port Type:	GetEffectiveAddressPortType
Operation:	GetEffectiveAddress
Callback Port Type:	----- No Callback -----
Operation:	
JCA File:	work/SOA_Application/J2CA_Outbound/SOA/Adapters/J2CA_Outbound_invoke.jca

13. Click **OK**.

    The Business service component is created in the External Services pane.

8.1.2.2 Configuring a File Transport Type Business Service

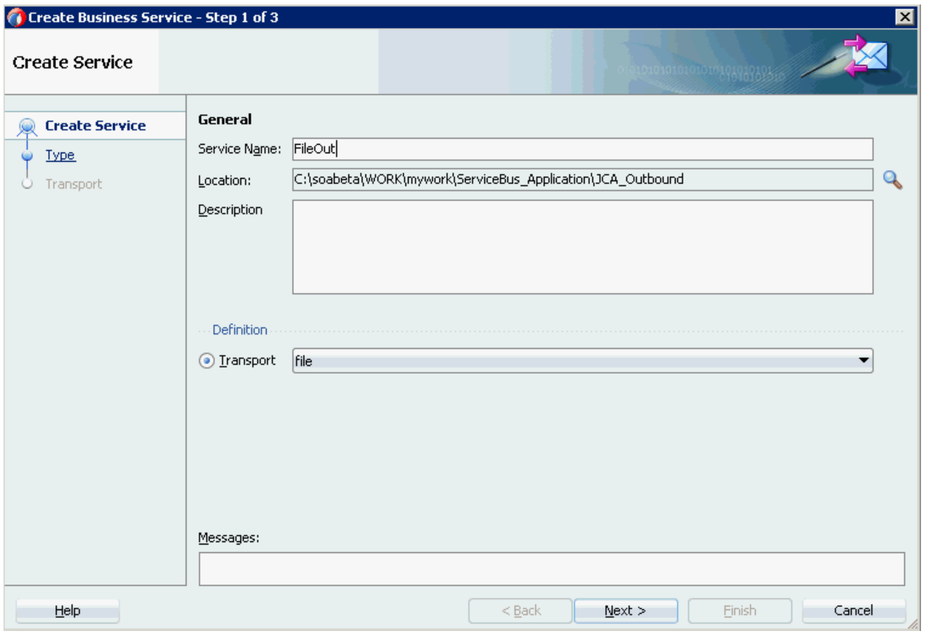
Perform the following steps to create a File Transport Business Service:

1. Drag and drop the **File Transport** component from the Advanced pane to the External Services pane.

   The Create Business Service dialog is displayed.

2. In the Service Name field, enter any name you wish for the Business Service (for example, FileOut), and click **Next**, as shown in Figure 8–8.

Figure 8–8 Create Service Pane



Create Business Service - Step 1 of 3

Create Service

General

Service Name: FileOut

Location: C:\soabeta\WORK\mywork\ServiceBus_Application\JCA_Outbound

Description

Definition

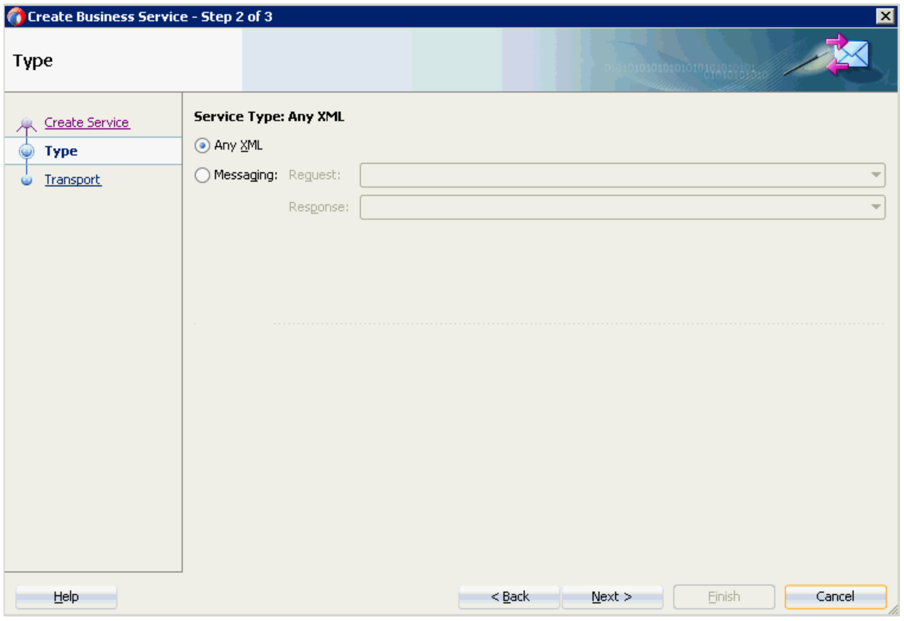
Transport: file

Messages:

The Type pane is displayed. The **Any XML** option is selected by default.

3. Click **Next**, as shown in Figure 8–9.

Figure 8–9 Type Pane



Create Business Service - Step 2 of 3

Type

Service Type: Any XML

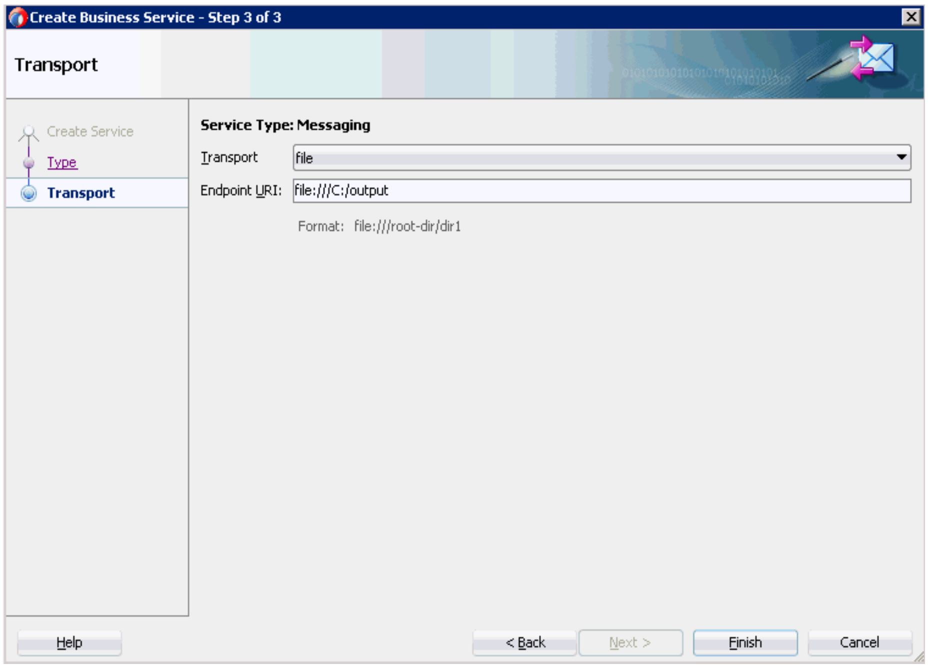
[✓ Any XML]

Messaging: Request: Response:

The Transport pane appears.

4. Provide the output location in the Endpoint URI field (for example, c:/output) and click **Finish**, as shown in Figure 8–10.

Figure 8–10 Transport Pane



Service Type: Messaging

Transport: file

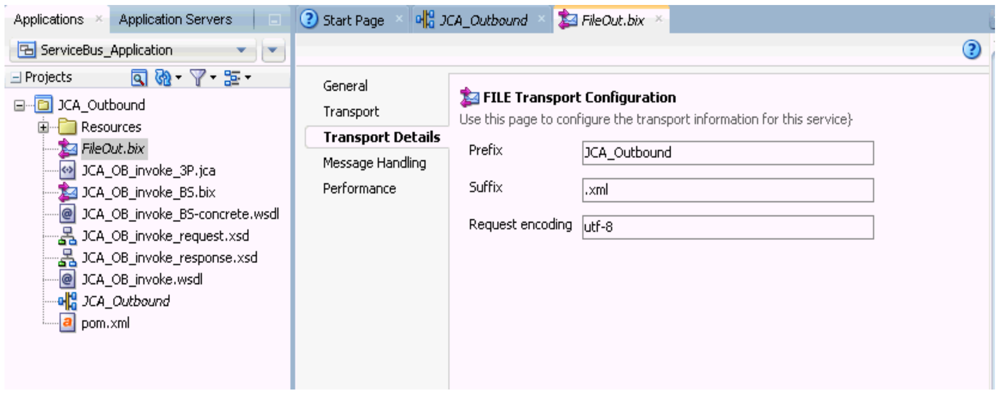
Endpoint URI: file:///C:/output

Format: file:///root-dir/dir1

The File Transport Business service Fileout is created and displayed.

5. Double-click the created Business service **Fileout** and provide the values for the Prefix and Suffix fields in the Transport Details Tab, as shown in Figure 8–11.

Figure 8–11 Transport Details



FILE Transport Configuration

Use this page to configure the transport information for this service}

Prefix: JCA_Outbound

Suffix: .xml

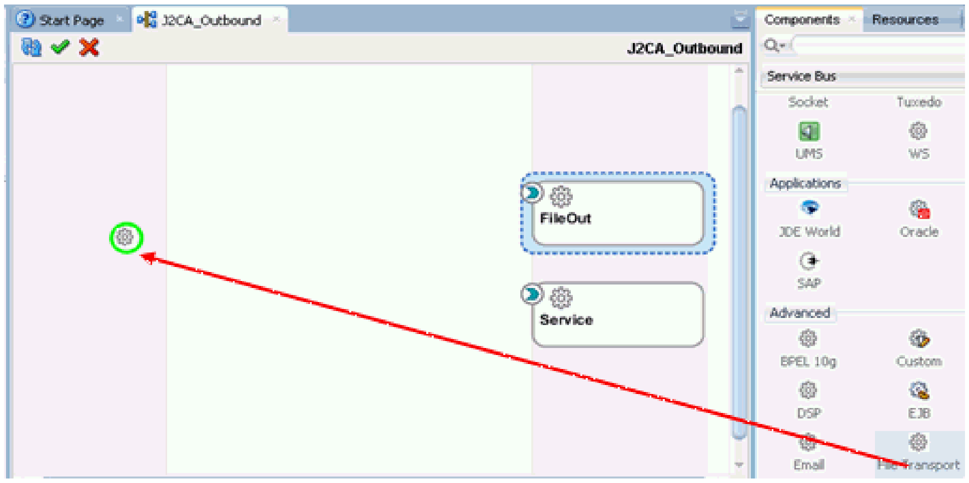
Request encoding: utf-8

8.1.2.3 Creating a Proxy Service With Pipeline

Perform the following steps to create a Proxy Service with Pipeline:

1. Drag and drop the **File Transport** component from the Advanced Components pane to the Proxy Services pane, as shown in Figure 8–12.

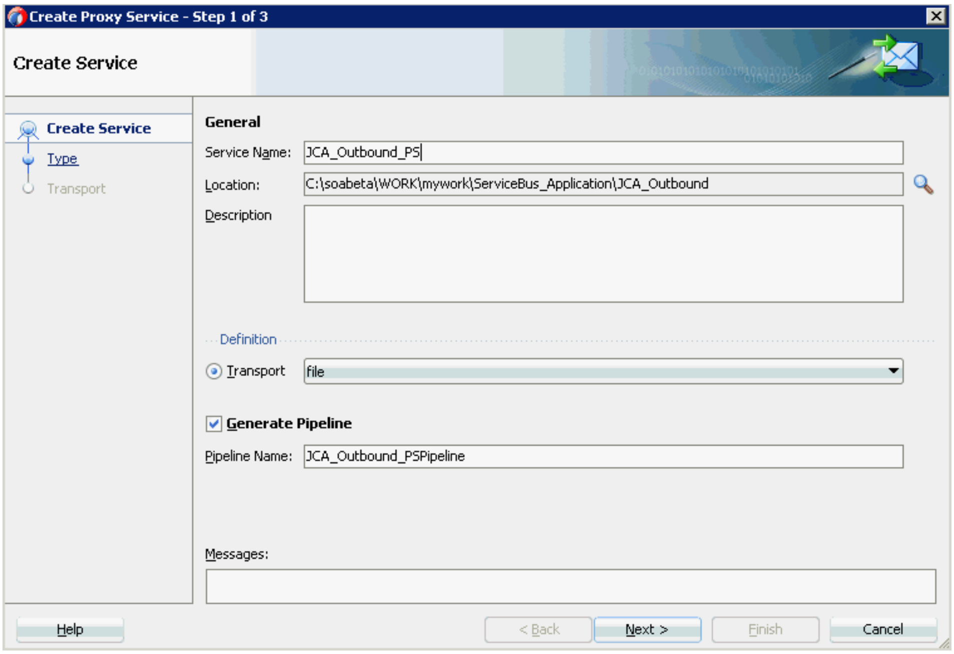
Figure 8–12 File Transport Component



The Create Proxy Service pane is displayed.

2. In the Service Name field, enter any name you wish for the Proxy service (for example, JCA_Outbound_PS). By default, **Generate Pipeline** is selected.
3. Click **Next**, as shown in Figure 8–13.

Figure 8–13 Create Service Pane



General

Service Name: JCA_Outbound_PS

Location: C:\soabeta\WORK\mywork\ServiceBus_Application\JCA_Outbound

Description

Definition

Transport: file

[✓] Generate Pipeline

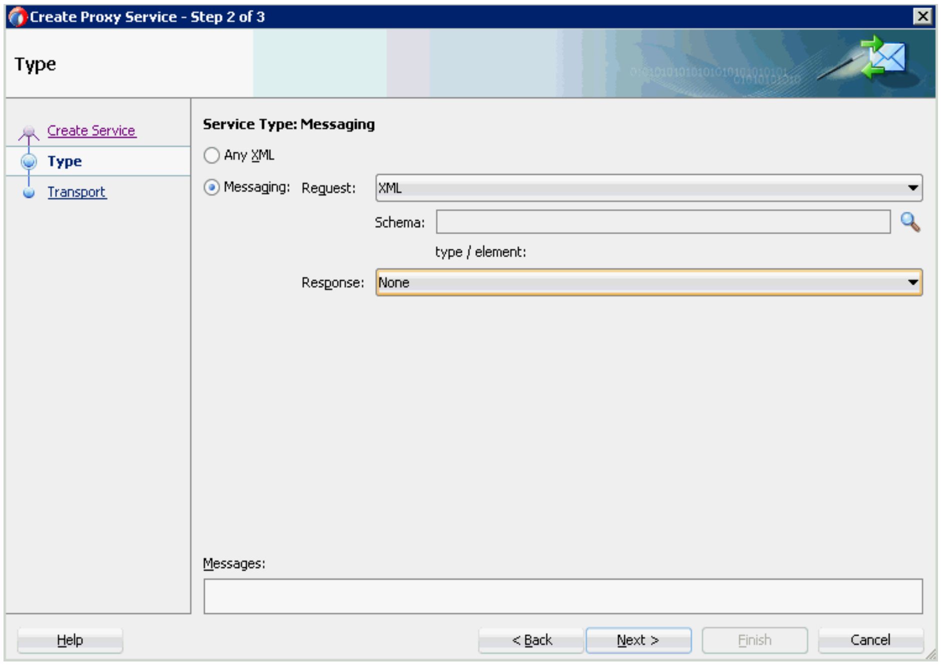
Pipeline Name: JCA_Outbound_PSPipeline

Messages:

The Type pane is displayed.

4. Select the **Messaging** option, set the Request to **XML** and Response as **None**, and then click **Next**, as shown in Figure 8–14.

Figure 8–14 Type Pane



Service Type: Messaging

Any XML

Messaging: Request: XML

Schema:

type / element:

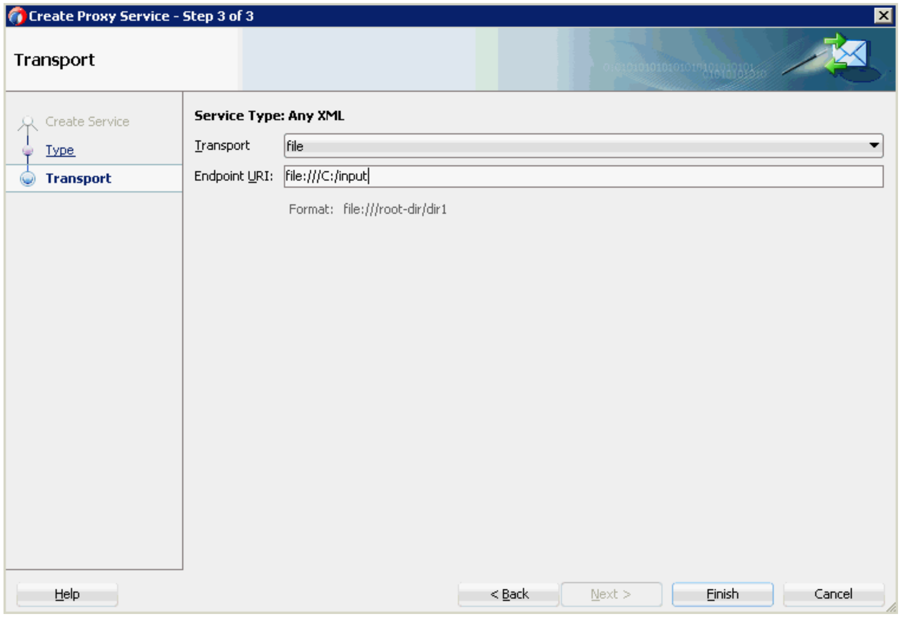
Response: None

Messages:

The Transport window is displayed.

5. Provide the input location in the Endpoint URI field (for example, c:/input) and click **Finish**, as shown in Figure 8–15.

Figure 8–15 Transport Window



Service Type: Any XML

Transport: file

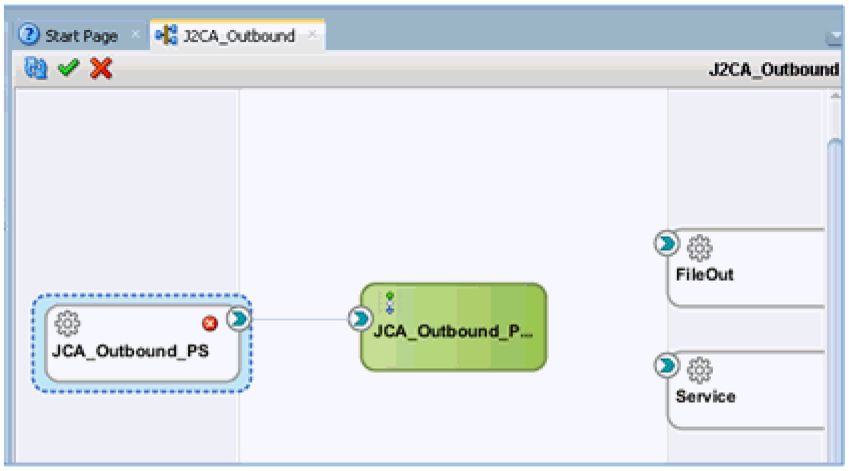
Endpoint URI: file:///C:/input

Format: file:///root-dir/dir1

The Proxy service along with the pipeline is created and displayed.

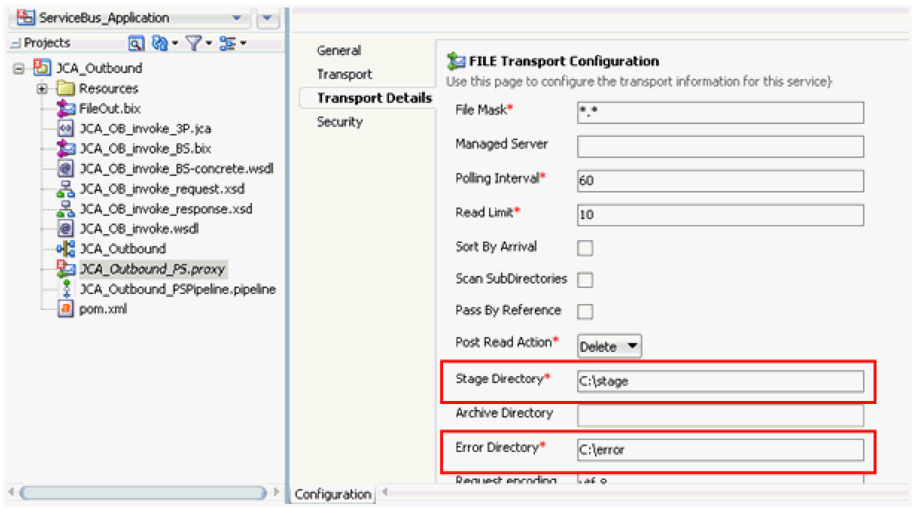
6. Double-click the created Proxy Service (for example: JCA_Outbound_PS), as shown inFigure 8–16.

Figure 8–16 Proxy Service Edit



7. In the displayed Proxy Service configuration page, select **Transport Details** and provide the values for Stage and Error Directory, as shown in Figure 8–17.

Figure 8–17 File Transport Configuration



JCA_Outbound
Resources
FileOut.bix
JCA_OB_invoke_3P.jca
JCA_OB_invoke_BS.bix
JCA_OB_invoke_BS-concrete.wsdl
JCA_OB_invoke_request.xsd
JCA_OB_invoke_response.xsd
JCA_OB_invoke.wsdl
JCA_Outbound
JCA_Outbound_PS.proxy
JCA_Outbound_PSPipeline.pipeline
pom.xml

FILE Transport Configuration

Use this page to configure the transport information for this service)

File Mask*	*.*
Managed Server	
Polling Interval*	60
Read Limit*	10
Sort By Arrival	☐
Scan SubDirectories	☐
Pass By Reference	☐
Post Read Action*	Delete
Stage Directory*	C:\stage
Archive Directory	
Error Directory*	C:\error
Request encoding	utf-8

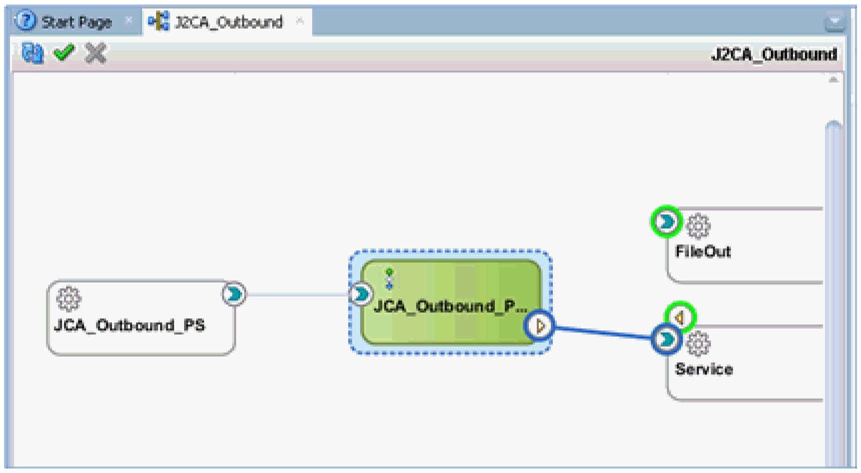
8. Save and close the Proxy Service configuration page.

8.1.2.4 Configuring the Routing Rules

Perform the following steps to configure the routing rules:

1. Connect the Pipeline to the Business Service (for example, Service) as shown in Figure 8–18.

Figure 8–18 Business Service Pipeline

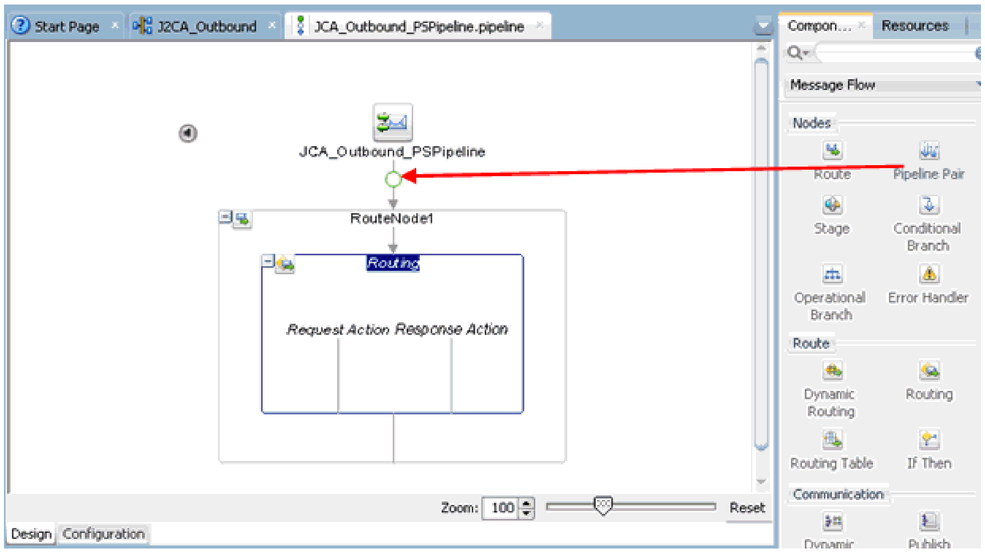


2. Double-click on the pipeline (for example, JCA_Outbound_PSPipeline) in the Pipelines/Split Joins pane.

   The Pipeline configuration page is displayed.

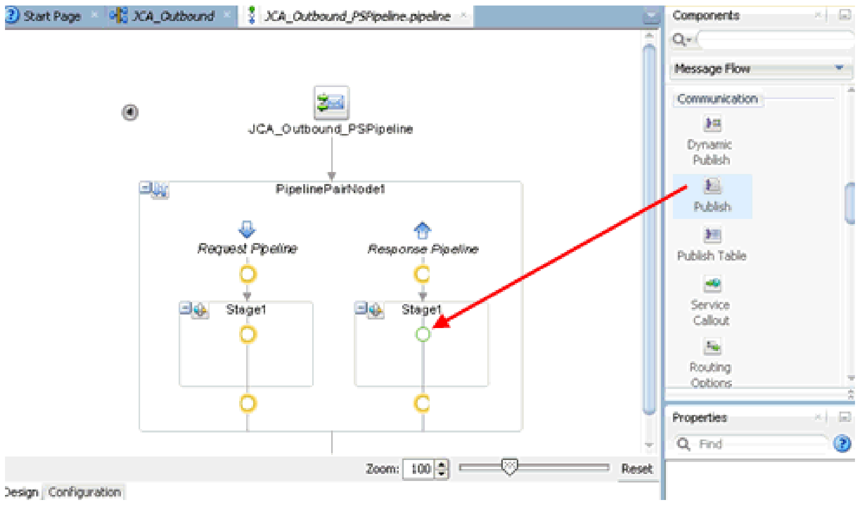
3. Drag and drop the **Pipeline Pair** node from Nodes pane to the area below the Pipeline (for example: JCA_Outbound_PSPipeline), as shown in Figure 8–19.

Figure 8–19 Pipeline Pair Node



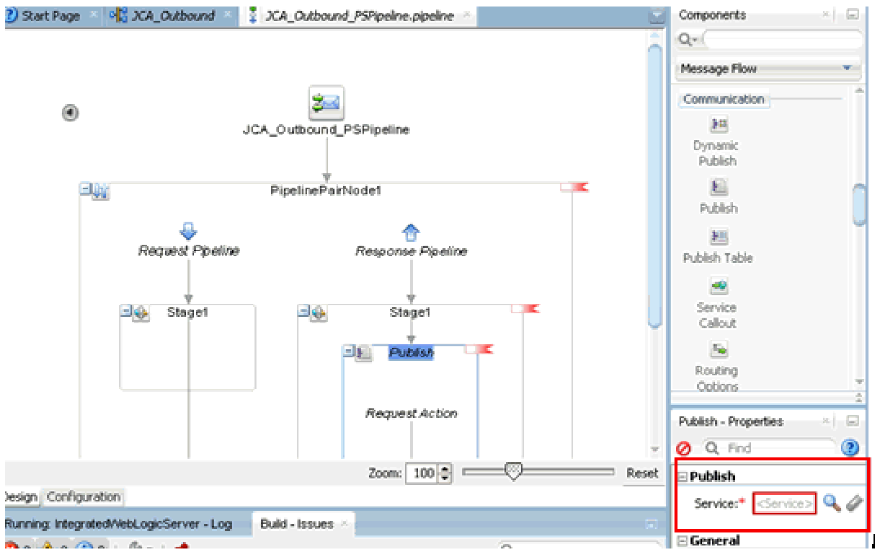
4. Drag and drop the **Publish** node from the Communication pane to the area beneath Stage1 of the Response Pipeline, as shown in Figure 8–20.

Figure 8–20 Publish Node



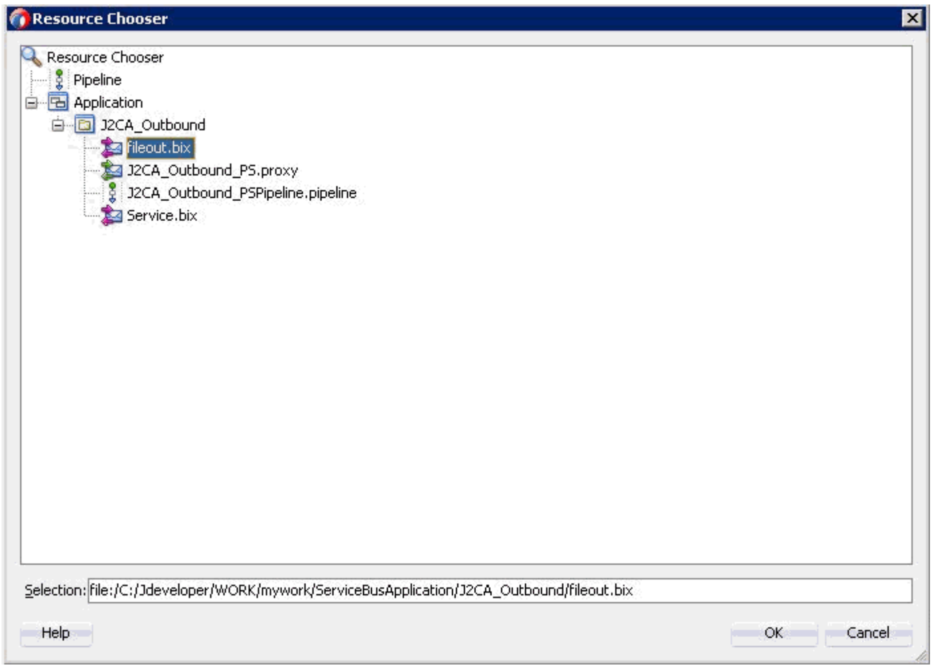
5. Click on the browse icon to the right of the Service field in the right pane of Publish Properties, as shown in Figure 8–21.

Figure 8–21 Browse Icon



6. In the displayed Resource Chooser window, select the **Fileout.bix** File Transport Business service and click **OK**, as shown in Figure 8–22.

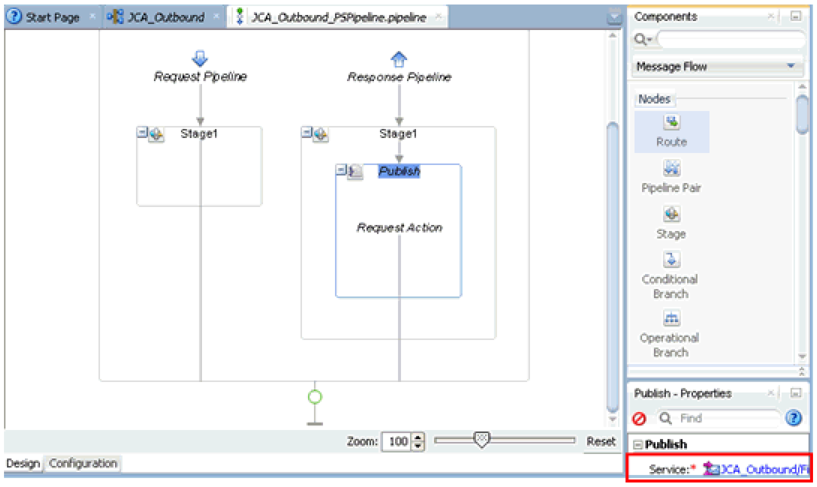
Figure 8–22 Resource Chooser



Selection: file:/C:/Jdeveloper/WORK/mywork/ServiceBusApplication/J2CA_Outbound/fileout.bix

In the right pane, the selected service is configured in the Publish pane, as shown in Figure 8–23.

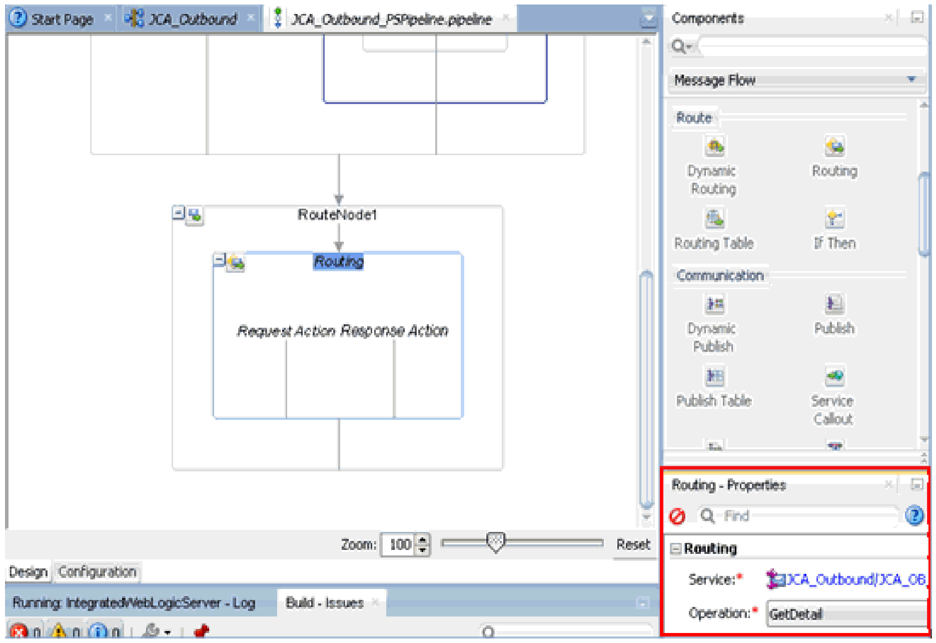
Figure 8–23 Publish Pane



Service:* JCA_Outbound/Fi

7. Click on the Routing to verify the Service is selected properly, as shown in Figure 8–24.

Figure 8–24 Pipeline Configuration



Routing

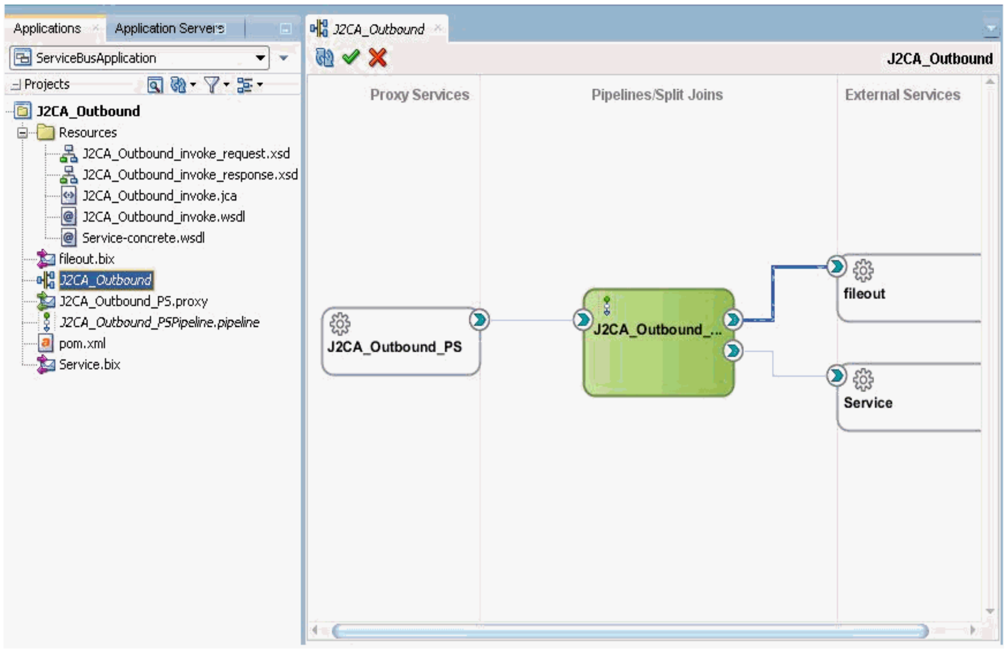
Service:* JCA_Outbound/JCA_OB

Operation:* GetDetail

8. Save and Close the Pipeline configuration page.

9. Double-click the overview.xml file (for example: JCA_Outbound), and click **Save All** in the menu bar to save the OSB process, as shown in Figure 8–25.

Figure 8–25 Save All Icon

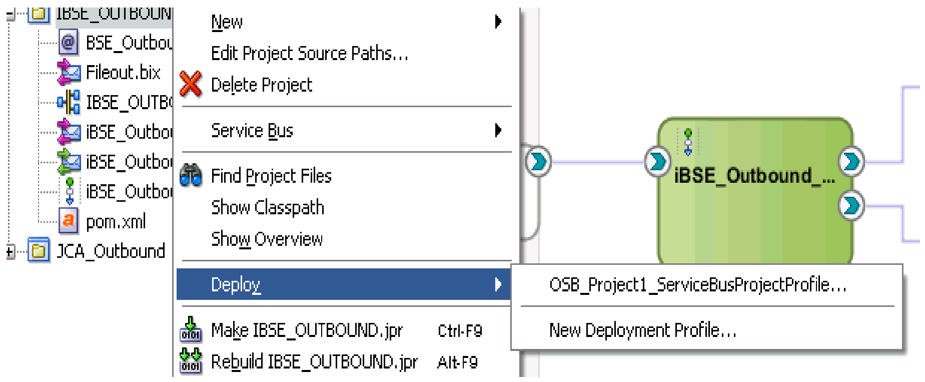


8.1.3 Deploying the OSB Outbound Process

Perform the following steps to deploy the OSB outbound process.

1. Right-click the OSB project, select **Deploy**, and then select **OSB_Project1_ServiceBusProjectProfile...**, as shown in Figure 8–26.

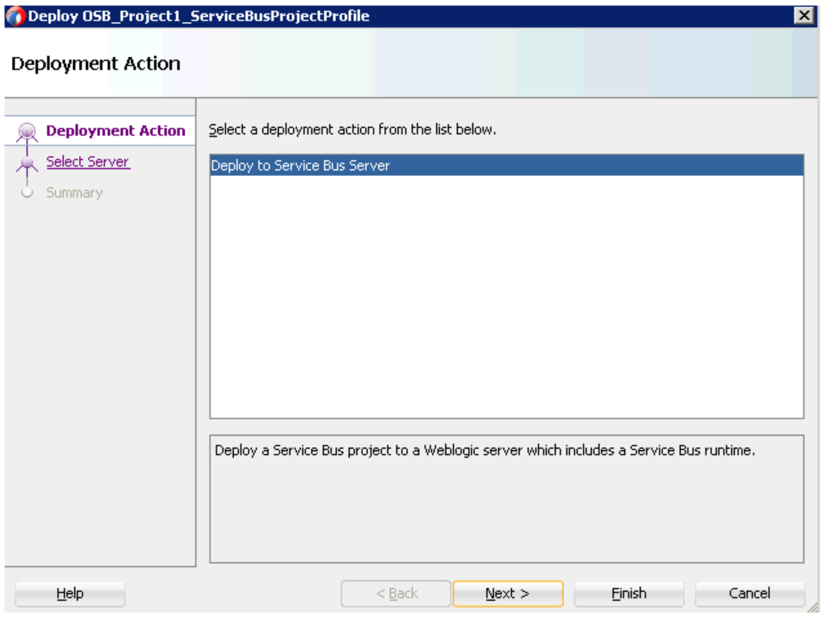
Figure 8–26 Deploy Option



The Deployment Action page is displayed.

2. Click **Next**, as shown in Figure 8–27.

Figure 8–27 Deployment Action Page



Select a deployment action from the list below.

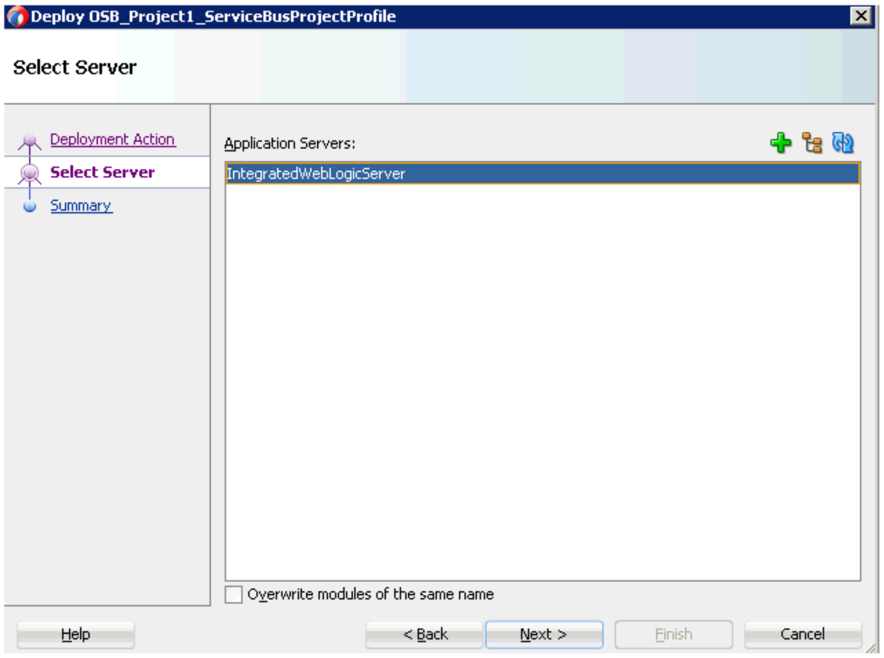
Deploy to Service Bus Server

Deploy a Service Bus project to a Weblogic server which includes a Service Bus runtime.

The Select Server page is displayed.

3. Select an available application server that was configured and click **Next**, as shown in Figure 8–28.

Figure 8–28 Select Server Page



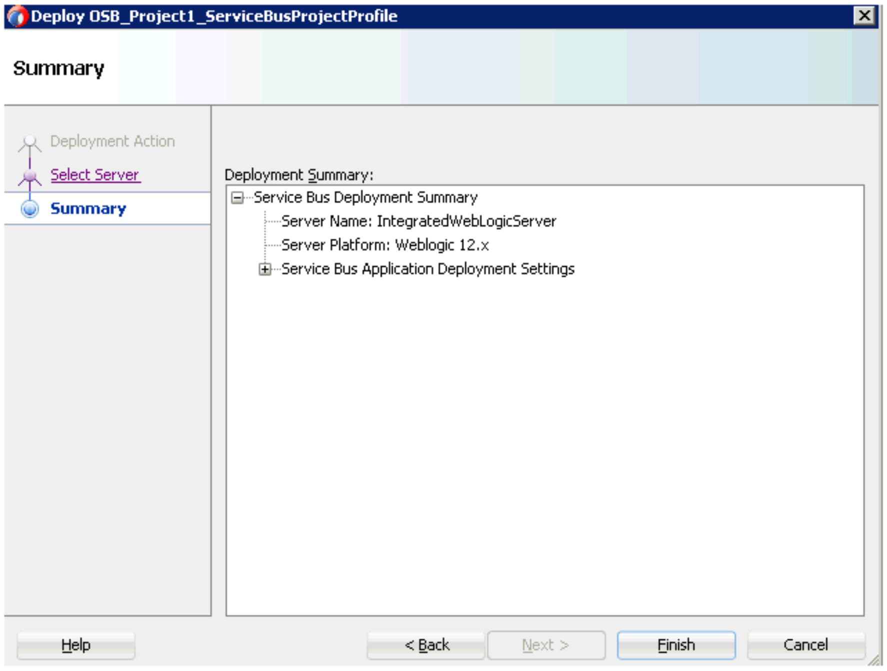
Application Servers:

IntegratedWebLogicServer

Overwrite modules of the same name

The Summary page is displayed, as shown in Figure 8–29.

Figure 8–29 Summary Page



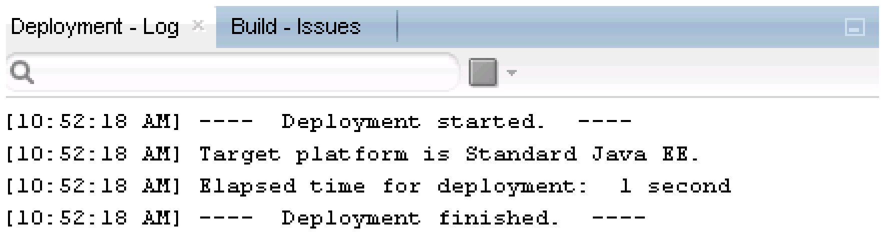
Deployment Summary:

Service Bus Deployment Summary
- Server Name: IntegratedWebLogicServer
- Server Platform: Weblogic 12.x
- Service Bus Application Deployment Settings

4. Review and verify all the available deployment information for your project and click **Finish**.

   The process is deployed successfully, as shown in Figure 8–30.

Figure 8–30 Successful Deployment Message



```
[10:52:18 AM] ----  Deployment started.  ----
[10:52:18 AM] Target platform is Standard Java EE.
[10:52:18 AM] Elapsed time for deployment:  1 second
[10:52:18 AM] ----  Deployment finished.  ----
```

5. Copy and paste an input XML file in the input folder you have configured (for example, C:\input).

The output is received in the configured output location (for example, C:\output).

8.2 Configuring an OSB Inbound Process Using JDeveloper (J2CA Configuration)

This section describes how to configure an OSB inbound process to your J.D. Edwards OneWorld system, using Oracle JDeveloper for J2CA configurations.

A sample project has been provided for this inbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_Samples\OSB_
Jdeveloper\J2CA\JDEdwards_Sample_J2CA_OSB_Inbound_Project
```

This section includes the following topics:

- Section 8.2.1, "Creating a Service Bus Application for OSB"
- Section 8.2.2, "Defining an OSB Inbound Process"
- Section 8.2.3, "Deploying the OSB Inbound Process"

Prerequisites

Before you design an OSB inbound process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.

8.2.1 Creating a Service Bus Application for OSB

To configure an OSB inbound process, you must create service bus application for OSB. For more information, see Section 8.1.1, "Creating a Service Bus Application for OSB" on page 8-2.

8.2.2 Defining an OSB Inbound Process

This section describes how to define an OSB inbound process. The following topics are included:

- Section 8.2.2.1, "Configuring a Third-Party Adapter Service Component"
- Section 8.2.2.2, "Creating a Pipeline"
- Section 8.2.2.3, "Configuring a File Transport Type Business Service"

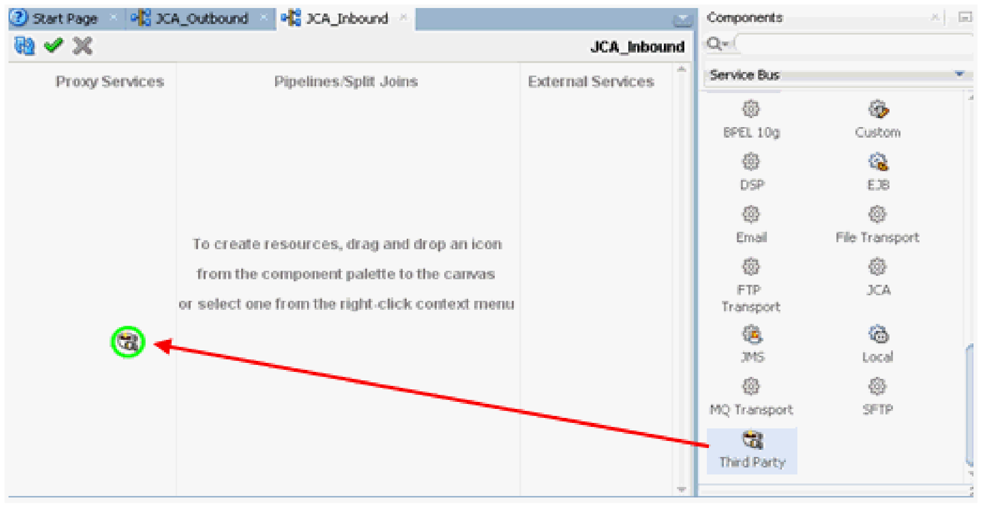
- Section 8.2.2.4, "Configuring the Routing Rules"

8.2.2.1 Configuring a Third-Party Adapter Service Component

Perform the following steps to create a third party adapter service component:

1. Drag and drop the **Third Party** adapter component from the Service Bus Components Pane to the Proxy Services, as shown in Figure 8–31.

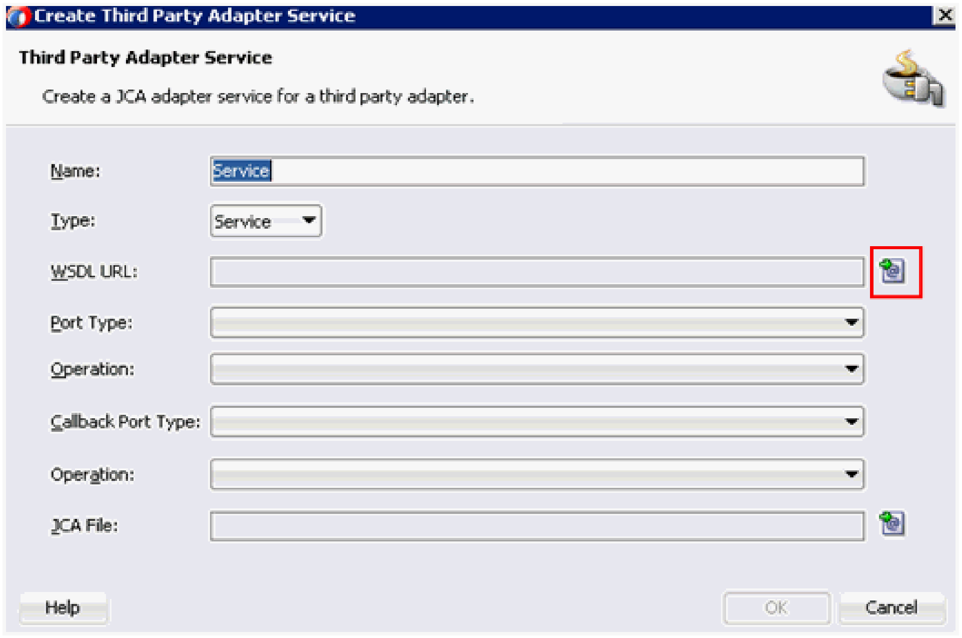
Figure 8–31 Third Party Adapter Service Component



   The Create Third Party Adapter Service dialog is displayed.

2. Enter any name you wish for the Third Party Adapter Service or leave it to the default value.
3. Ensure that **Service** is selected from the Type drop-down list (by default).
4. Click the Find existing WSDLs icon, which is located to the right of the WSDL URL field, as shown in Figure 8–32.

Figure 8–32 Third Party Adapter Service Dialog


Create Third Party Adapter Service
Third Party Adapter Service
Create a JCA adapter service for a third party adapter.
Name:
Service
Type:
Service
WSDL URL:
Port Type:
Operation:
Callback Port Type:
Operation:
JCA File:
Help
OK
Cancel

The WSDL Chooser dialog is displayed.

5. Select the File system folder, then browse and select an inbound WSDL file from the WSDL directory.
6. Click **OK**.

   The Import Service Bus Resources dialog is displayed.
7. Click **Next**.
8. In the Configuration window, click **Finish**.

   You are returned to the Create Third Party Adapter Service dialog.
9. Click the Find JCA file icon, which is located to the right of the JCA File field.

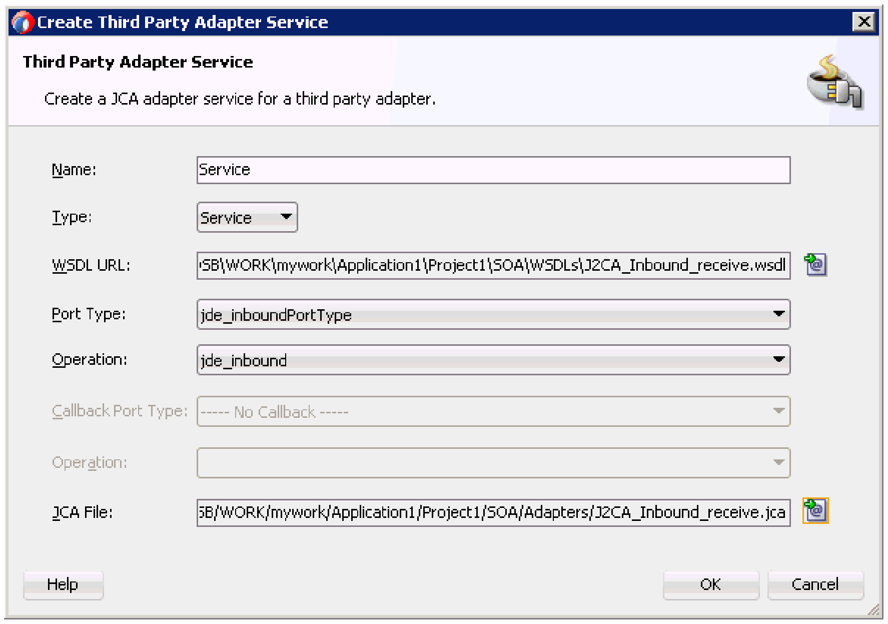
   The Transformation Chooser dialog is displayed.
10. Select the JCA properties file from the WSDL directory.
11. Click **OK**.

    The Copy File message is displayed.
12. Click **Yes**.

    A copy of the JCA properties file is created in the project folder.

    You are returned to the Create Third Party Adapter Service dialog, as shown in Figure 8–33.

Figure 8–33 Create Third Party Adapter Service Dialog


Create Third Party Adapter Service

Third Party Adapter Service

Create a JCA adapter service for a third party adapter.

Name:	Service
Type:	Service
WSDL URL:	SB\WORK\mywork\Application1\Project1\SOA\WSDLs\J2CA_Inbound_receive.wsdl
Port Type:	jde_inboundPortType
Operation:	jde_inbound
Callback Port Type:	----- No Callback -----
Operation:	
JCA File:	SB/WORK/mywork/Application1/Project1/SOA/Adapters/J2CA_Inbound_receive.jca

13. Click **OK**.

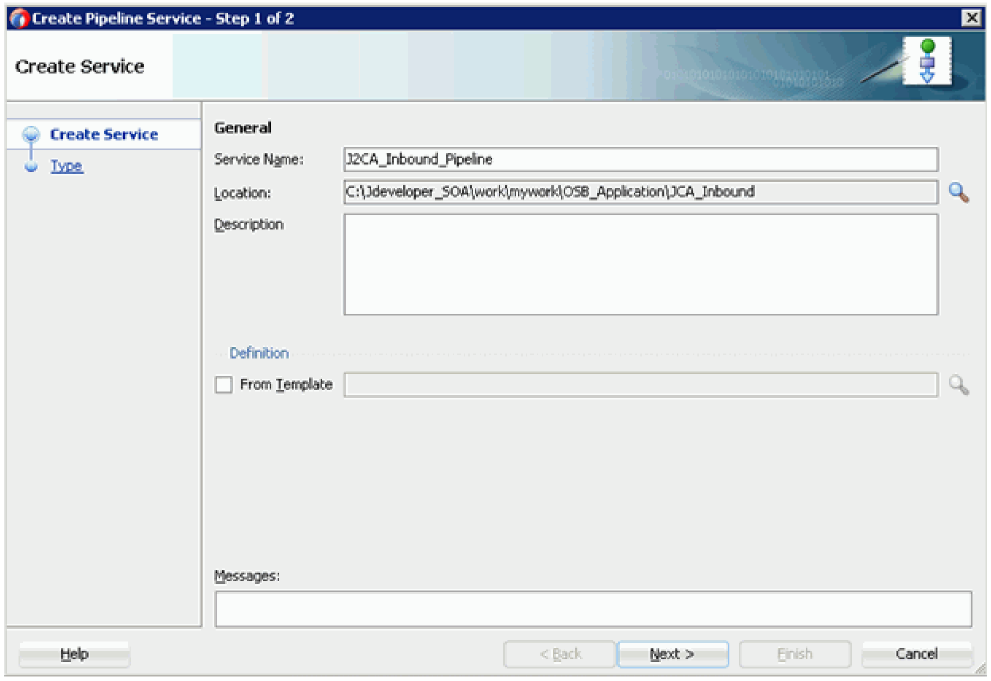
    The third party adapter service component is created in the Proxy Services pane.

8.2.2.2 Creating a Pipeline

Perform the following steps to generate inbound proxy service with Pipeline:

1. Under Service Bus, click **Resources**.
2. Drag and drop the Pipeline to the Pipelines/Split Joins pane.
3. Provide a name for the Pipeline and click next, as shown in Figure 8–34.

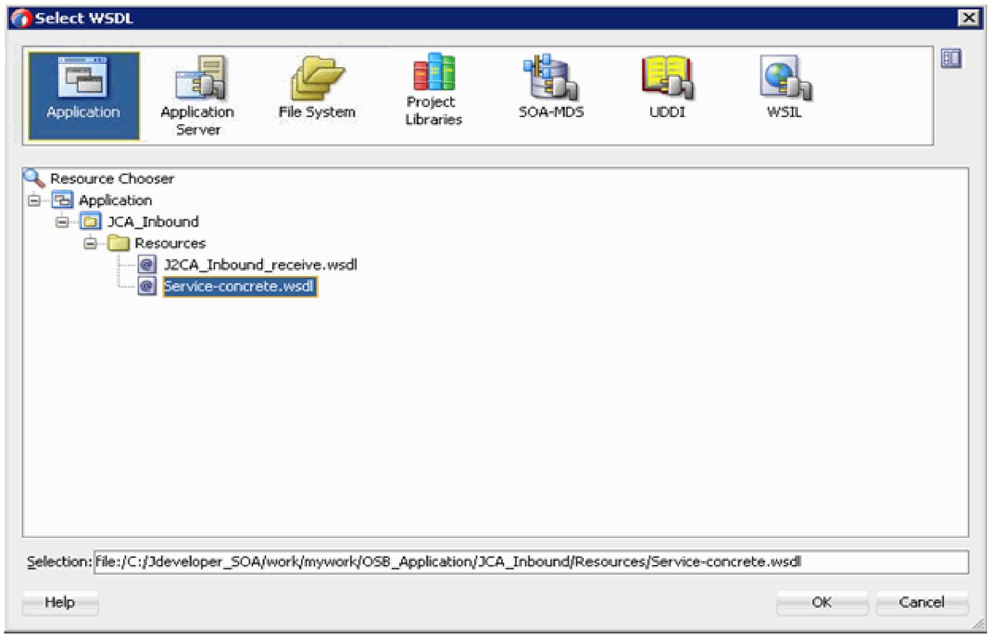
Figure 8–34 Create Service Page



Service Name: J2CA_Inbound_Pipeline
Location: C:\Jdeveloper_SOA\work\mywork\OSB_Application\JCA_Inbound
Description
From Template
Messages:

4. In the Create Pipeline Service window, select the **WSDL** option and click on the WSDL URL.
5. Select **Application** in the WSDL chooser window, then select **service-concrete.wsdl** in the appropriate OSB project, and then click **OK**, as shown in Figure 8–35.

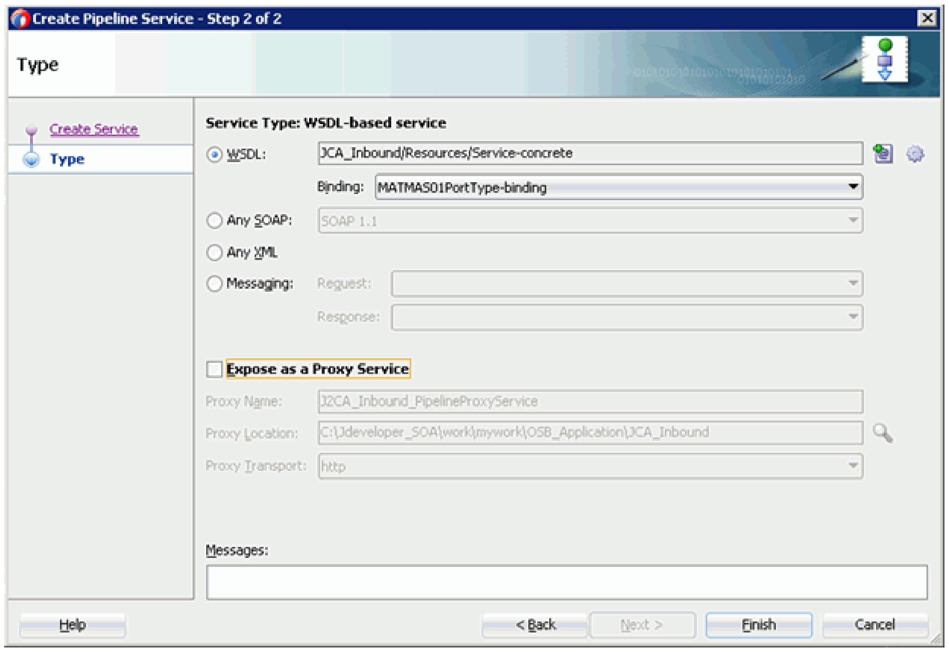
Figure 8–35 Select WSDL Page



Selection: file:/C:/Jdeveloper_SOA/work/mywork/OSB_Application/JCA_Inbound/Resources/Service-concrete.wsdl

6. Clear the Expose as a Proxy Service check box and click **Finish**, as shown in Figure 8–36.

Figure 8–36 Type Page


Create Pipeline Service - Step 2 of 2

Type

Service Type: WSDL-based service

WSDL: JCA_Inbound/Resources/Service-concrete

Binding: MATMAS01PortType-binding

Any SOAP: SOAP 1.1

Any XML

Messaging: Request: Response:

Expose as a Proxy Service

Proxy Name: J2CA_Inbound_PipelineProxyService

Proxy Location: C:\Jdeveloper_SOA\work\mywork\OSB_Application\JCA_Inbound

Proxy Transport: http

Messages:

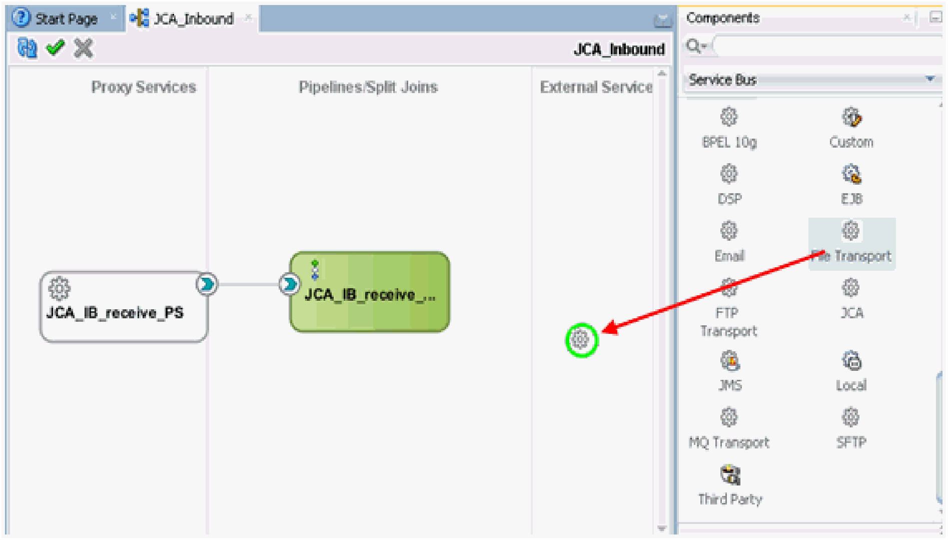
7. Drag and drop the Proxy Service to the Pipelines/Split Joins pane.

8.2.2.3 Configuring a File Transport Type Business Service

Perform the following steps to create the File Transport Type Business Service:

1. Drag and drop the **File Transport** component from the Advanced pane to the External Services pane, as shown in Figure 8–37.

Figure 8–37 File Transport Node



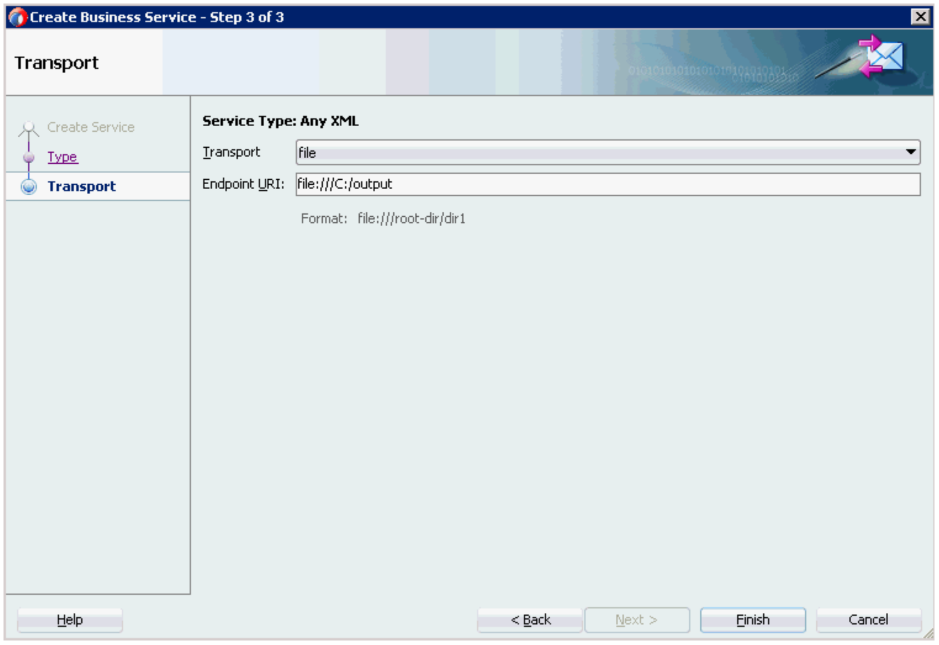
The Create Business Service dialog is displayed.

2. In the Service Name field, enter any name you wish for the Business Service (for example, FileOut), and click **Next**.

   In the displayed Type Window, the Any XML option is selected by default.

3. Click **Next**.
4. In the displayed Transport window, provide the output location in the Endpoint URI field (for example, c:\output), and click **Finish**, as shown in Figure 8–38.

Figure 8–38 Transport Pane


Create Business Service - Step 3 of 3

Transport

Service Type: Any XML

Transport: file

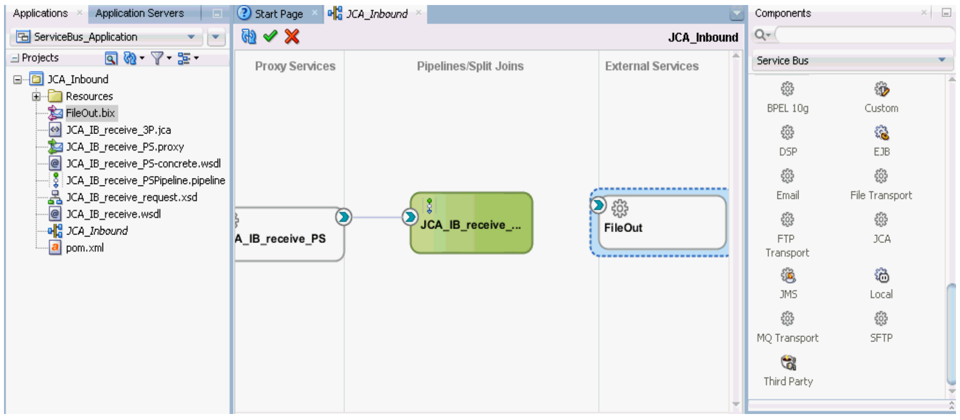
Endpoint URI: file:///C:/output

Format: file:///root-dir/dir1

The FileOut Business service is created.

5. Double-click the FileOut Business service, as shown in Figure 8–39.

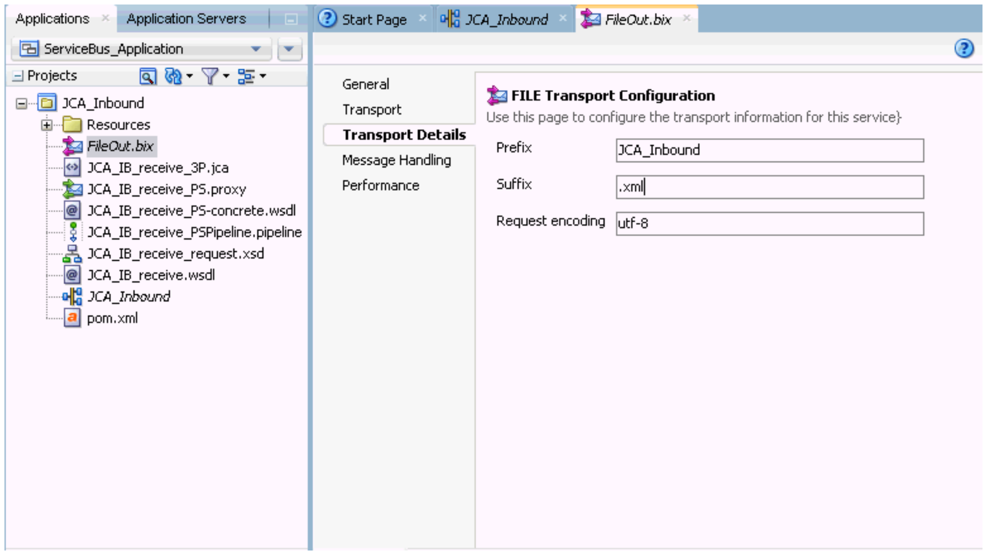
Figure 8–39 FileOut Business Service



The Configuration page is displayed.

6. Navigate to the Transport Details tab and provide the values for the Prefix and Suffix fields, as shown in Figure 8–40.

Figure 8–40 File Transport Configuration



FILE Transport Configuration

Use this page to configure the transport information for this service}

Prefix	JCA_Inbound
Suffix	.xml
Request encoding	utf-8

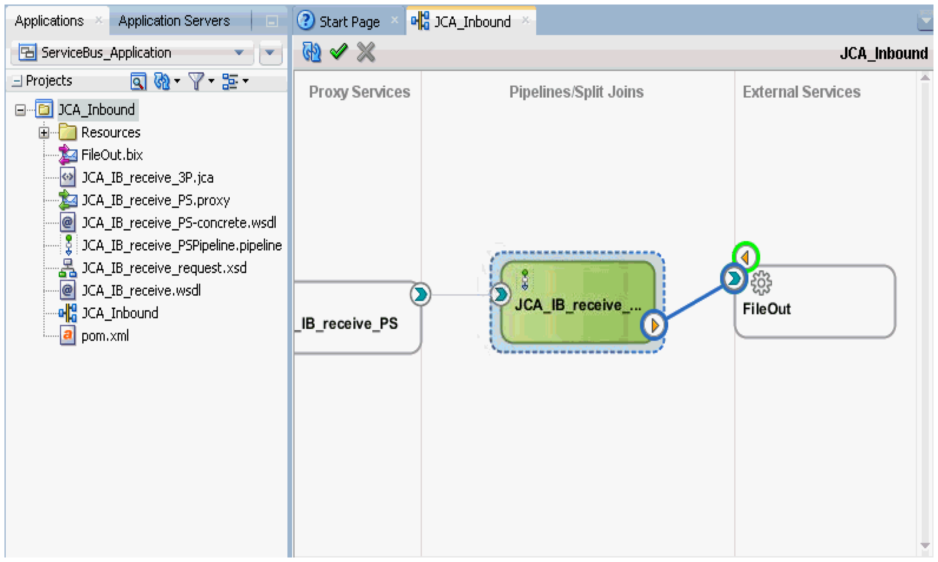
7. Save and close the Configuration page.

8.2.2.4 Configuring the Routing Rules

Perform the following steps to configure the routing rules.

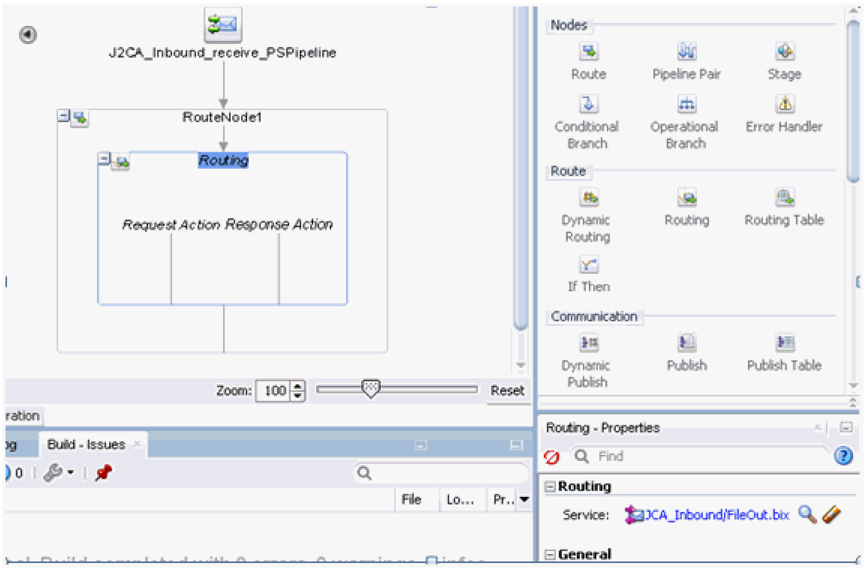
1. Create a connection between the Pipeline (for example, JCA_IB_receive_PSPipeline) and the File Type Business Service (for example, FileOut), as shown inFigure 8–41.

Figure 8–41 Mapping Proxy and FileOut



2. Double-click the Pipeline (for example, J2CA_Inbound_receive_PSPipeline).
3. Click the Routing pane and ensure that the File Type Business Service (for example, FileOut) is properly configured in the Service field, as shown in Figure 8–42.

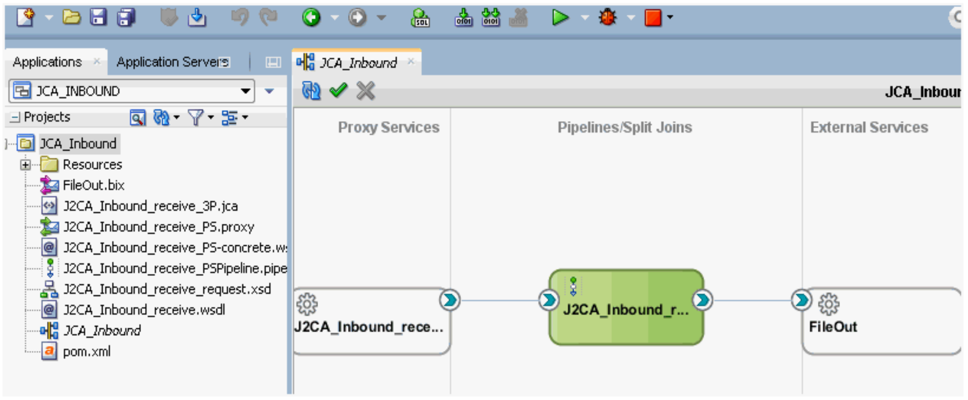
Figure 8–42 Routing Pane



4. Save and close the Pipeline configuration page.

5. Double-click on the overview.xml file (for example, JCA_Inbound) and click **Save All** in the menu bar to save the OSB process, as shown in Figure 8–43.

Figure 8–43 Save All



8.2.3 Deploying the OSB Inbound Process

To deploy the created OSB inbound process, see steps 1 - 4 in Section 8.1.3, "Deploying the OSB Outbound Process" on page 8-15.

Once the OSB inbound process is deployed successfully, trigger an event from the J.D. Edwards OneWorld system and check if the output is received in the configured output location (for example, C:\output).

For more information on triggering an event, see Section 4.5.5, "Triggering an Event in J.D. Edwards OneWorld" on page 4-47.

8.3 Configuring an OSB Outbound Process Using JDeveloper (BSE Configuration)

This section describes how to configure an OSB outbound process to your J.D. Edwards OneWorld system, using Oracle JDeveloper for BSE configurations.

A sample project has been provided for this outbound use case scenario in the following folder of the Application Adapters installation:

```
<ADAPTER_HOME>\etc\sample\JDEdwards_Samples.zip\JDEdwards_
Samples\OSB\BSE\JDEdwards_Sample_BSE_OSB_Outbound_Project
```

This section includes the following topics:

- Section 8.3.1, "Creating a Service Bus Application for OSB"
- Section 8.3.2, "Defining an OSB Outbound Process"
- Section 8.3.3, "Deploying the OSB Outbound Process"

Prerequisites

Before you design an OSB outbound process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.6.1, "Generating a WSDL File for Request and Response Services Using a Web Service" on page 4-52.

8.3.1 Creating a Service Bus Application for OSB

To configure an OSB outbound process, you must create a service bus application for OSB. For more information, see Section 8.1.1, "Creating a Service Bus Application for OSB" on page 8-2.

8.3.2 Defining an OSB Outbound Process

This section describes how to define an OSB outbound process. The following topics are included:

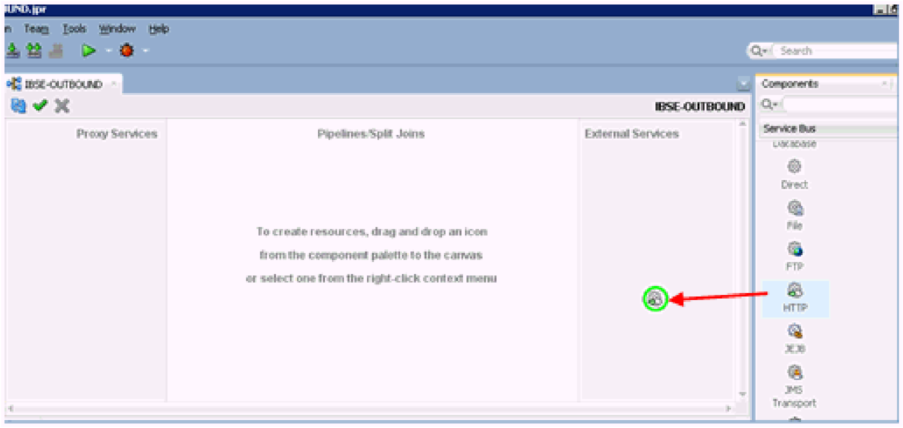
- Section 8.3.2.1, "Configuring a WSDL-based Business Service"
- Section 8.3.2.2, "Creating a Proxy Service With Pipeline"
- Section 8.3.2.3, "Configuring a File Transport Type Business Service"
- Section 8.3.2.4, "Configuring the Routing Rules"

8.3.2.1 Configuring a WSDL-based Business Service

Perform the following steps to configure a WSDL-based Business Service:

1. Drag and drop the **HTTP** component from the Technology Components pane to the External Services area, as shown in Figure 8–44.

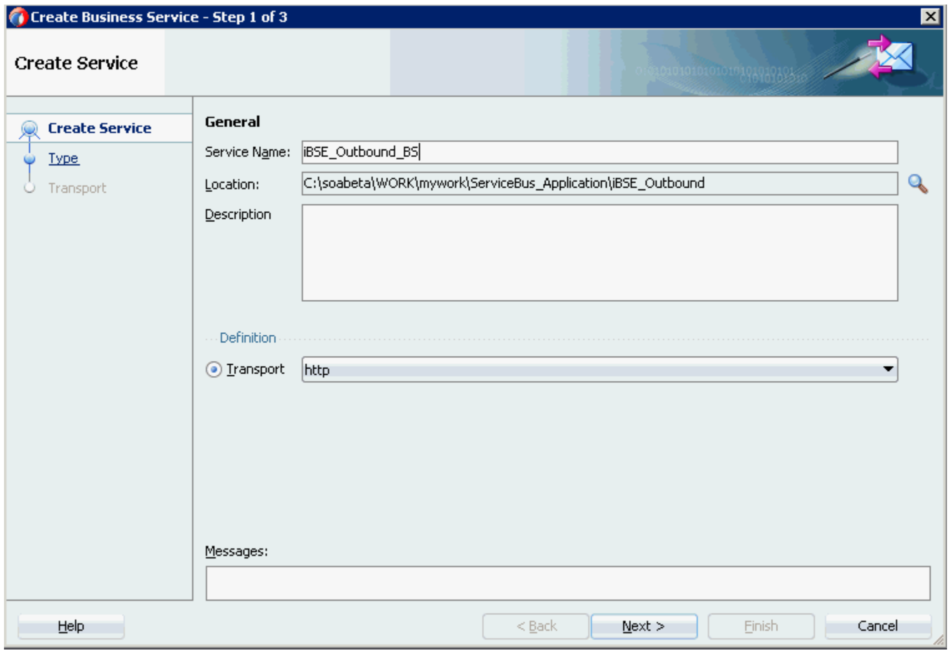
Figure 8–44 HTTP Component



The Create Business Service window is displayed.

2. In the Service Name field, enter any name you wish for the Business Service and click **Next**, as shown in Figure 8–45.

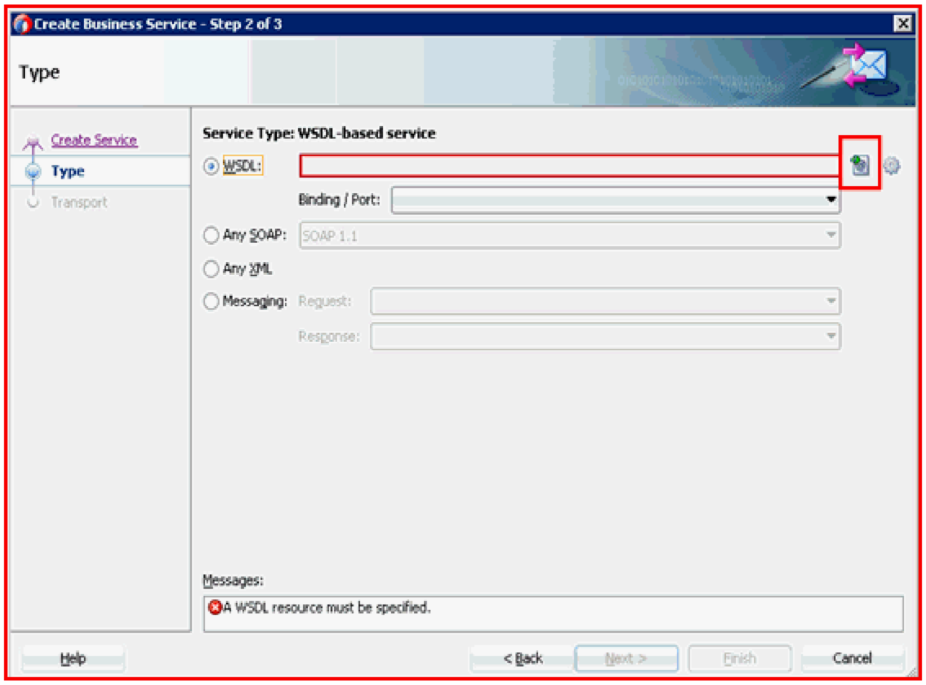
Figure 8–45 Create Business Service



Service Name: iBSE_Outbound_BS
Location: C:\soabeta\WORK\mywork\ServiceBus_Application\iBSE_Outbound
Description
Transport http
Messages:

3. In the displayed Service Type window, select the WSDL option and click the **Select WSDL** icon, as shown in Figure 8–46.

Figure 8–46 Type Pane

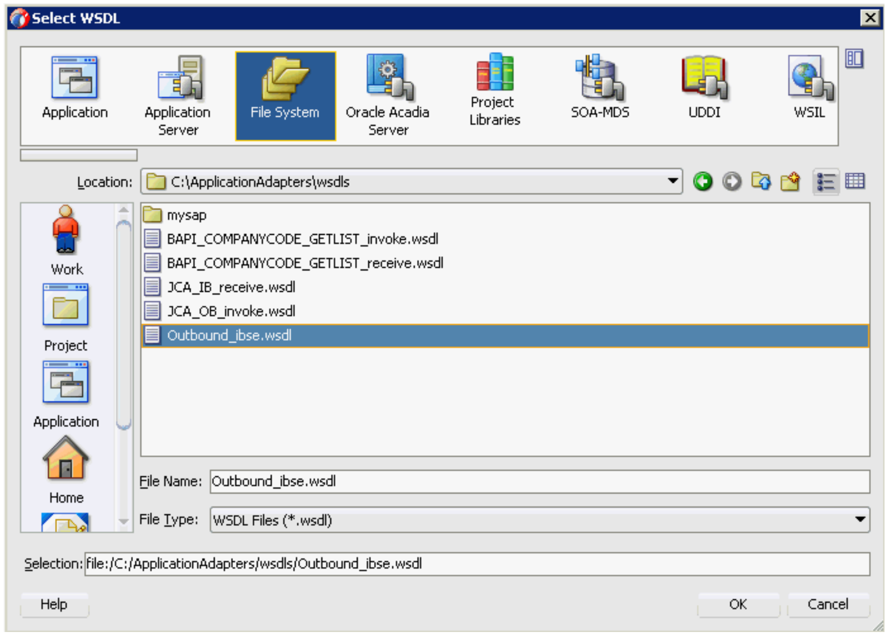


A WSDL resource must be specified.

The Select WSDL window is displayed.

4. Select the File System folder icon, browse to the iBSE WSDL file and select it from the WSDL location, and then click **OK**, as shown in Figure 8–47.

Figure 8–47 Select WSDL Window



Location: C:\ApplicationAdapters\wsdls

mysap
BAPI_COMPANYCODE_GETLIST_invoke.wsdl
BAPI_COMPANYCODE_GETLIST_receive.wsdl
JCA_IB_receive.wsdl
JCA_OB_invoke.wsdl
Outbound_ibse.wsdl

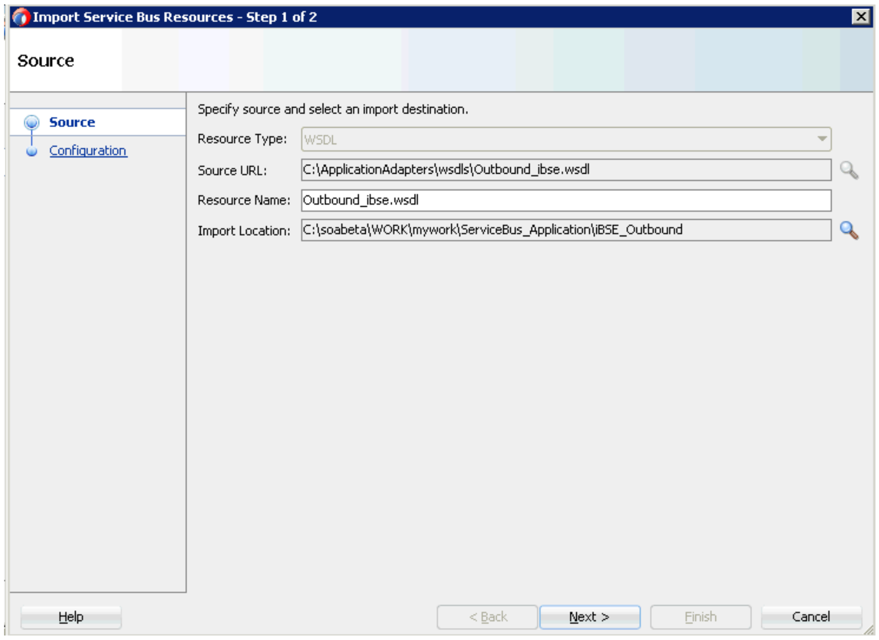
File Name: Outbound_ibse.wsdl

File Type: WSDL Files (*.wsdl)

Selection: file:/C:/ApplicationAdapters/wsdls/Outbound_ibse.wsdl

5. In the displayed Source pane, click **Next**, as shown in Figure 8–48.

Figure 8–48 Source Pane


Import Service Bus Resources - Step 1 of 2

Source

Specify source and select an import destination.

Resource Type: WSDL

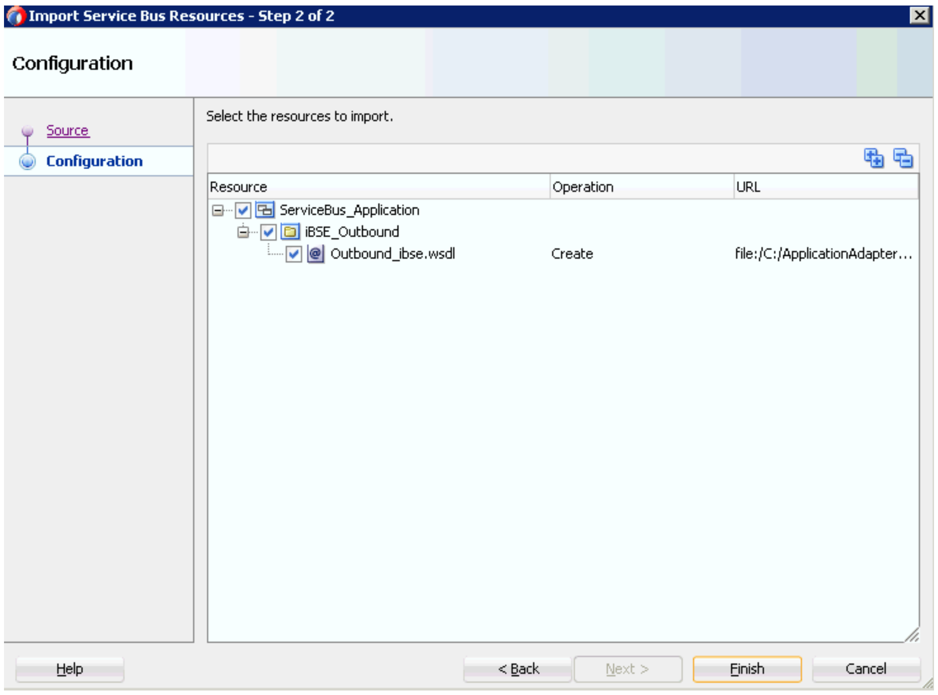
Source URL: C:\ApplicationAdapters\wsdls\Outbound_ibse.wsdl

Resource Name: Outbound_ibse.wsdl

Import Location: C:\soabeta\WORK\mywork\ServiceBus_Application\iBSE_Outbound

6. In the displayed Configuration pane, click **Finish**, as shown in Figure 8–49.

Figure 8–49 Configuration Pane


Import Service Bus Resources - Step 2 of 2

Configuration

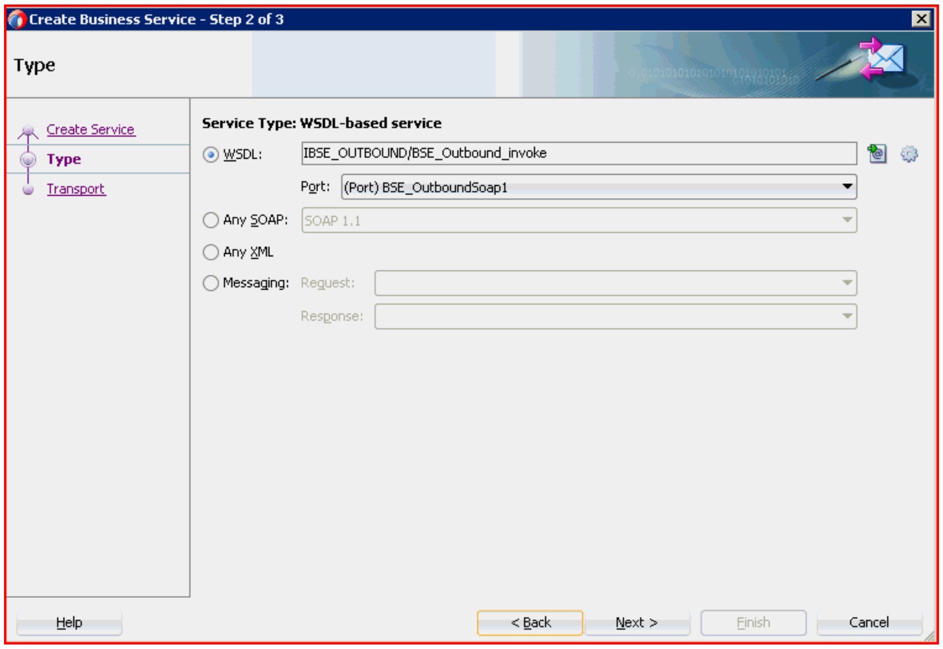
Select the resources to import.

Resource	Operation	URL
ServiceBus_Application		
iBSE_Outbound		
Outbound_ibse.wsdl	Create	file:/C:/ApplicationAdapter…

You are returned to the Create Business Service window.

7. In the displayed Type pane, click **Next**, as shown in Figure 8–50.

Figure 8–50 Type Pane



Create Business Service - Step 2 of 3

Type

Service Type: WSDL-based service

WSDL: IBSE_OUTBOUND/BSE_Outbound_invoke

Port: (Port) BSE_OutboundSoap1

Any SOAP: SOAP 1.1

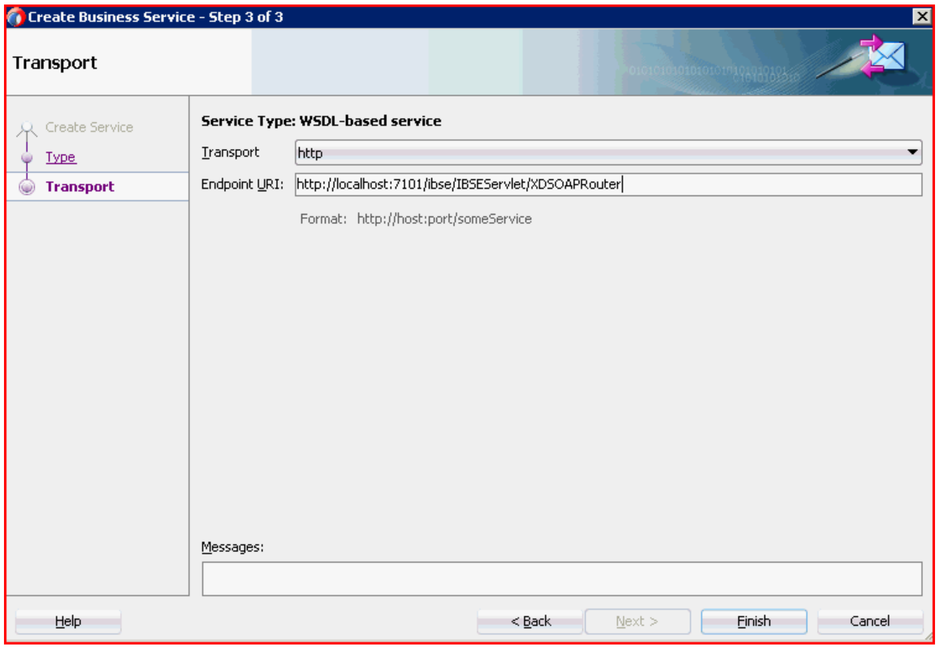
Any XML

Messaging: Request:

Response:

8. In the displayed Transport window, you can modify the Endpoint URI field if the hostname and port number varies, and then click **Finish**, as shown in Figure 8–51.

Figure 8–51 Transport Pane



Create Business Service - Step 3 of 3

Transport

Service Type: WSDL-based service

Transport: http

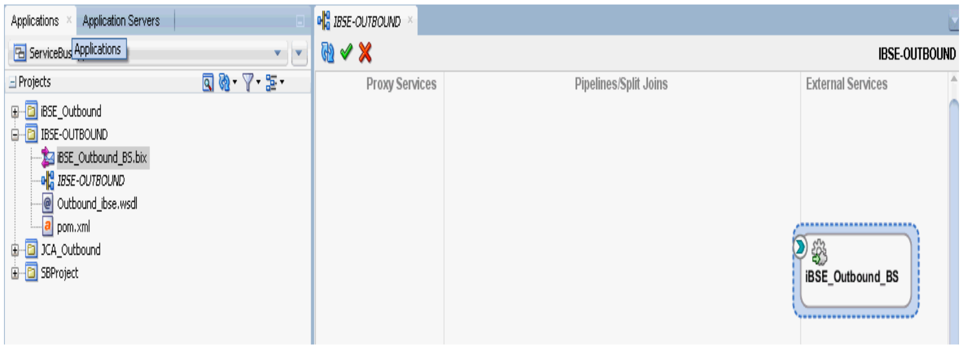
Endpoint URI: http://localhost:7101/ibse/IBSEServlet/XDSOAPRouter

Format: http://host:port/someService

Messages:

The Business Service is created and displayed in the External Services pane, as shown in Figure 8–52.

Figure 8–52 External Services Pane

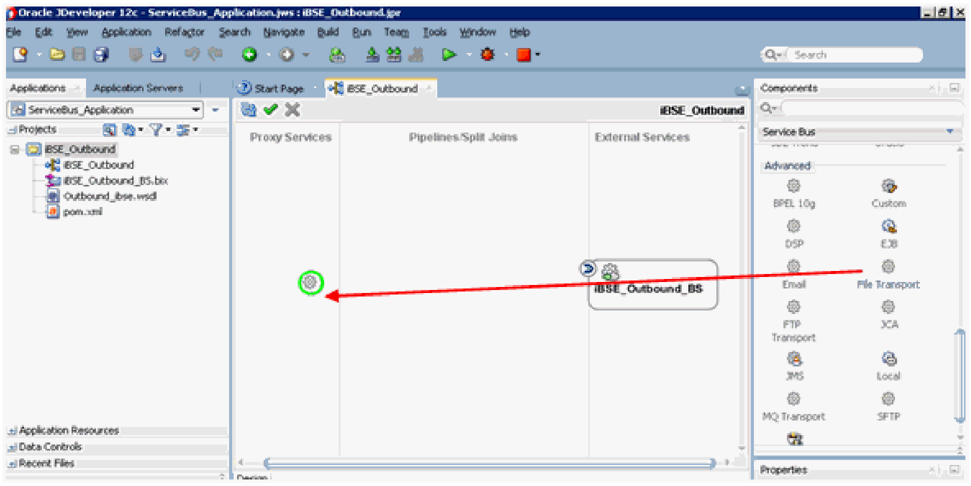


8.3.2.2 Creating a Proxy Service With Pipeline

Perform the following steps to create a Proxy Service with Pipeline:

1. Drag and drop the **File Transport** component from the Advanced Components pane to the Proxy Services pane, as shown in Figure 8–53.

Figure 8–53 File Transport Component

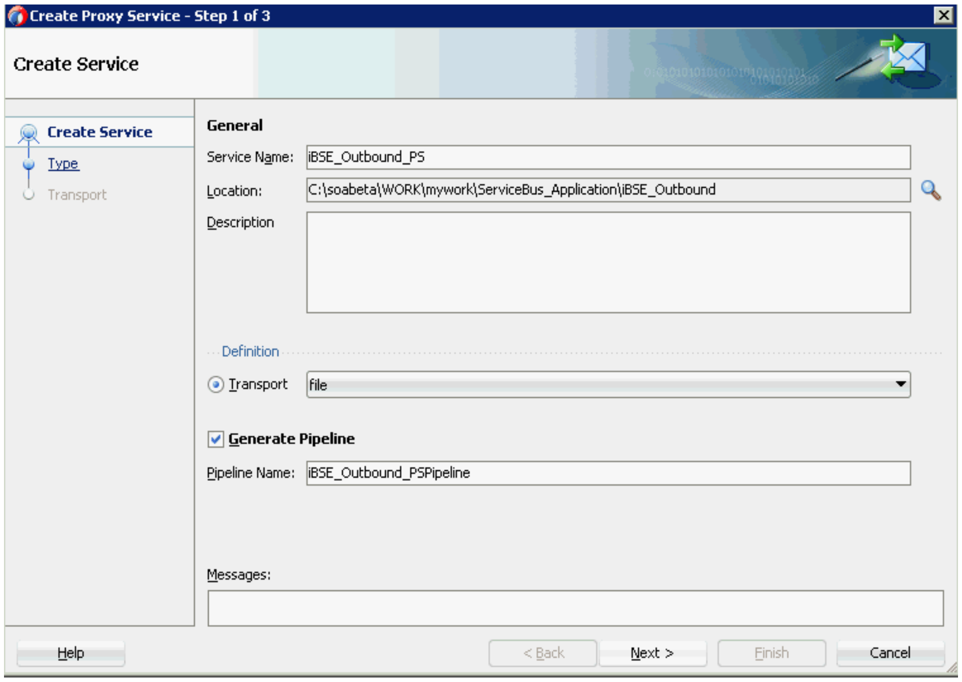


The Create Proxy Service pane is displayed.

2. In the Service Name field, enter any name you wish for the Proxy service (for example, JCA_Outbound_PS). By default, **Generate Pipeline** is selected.

3. Click **Next**, as shown in Figure 8–54.

Figure 8–54 Create Service Pane

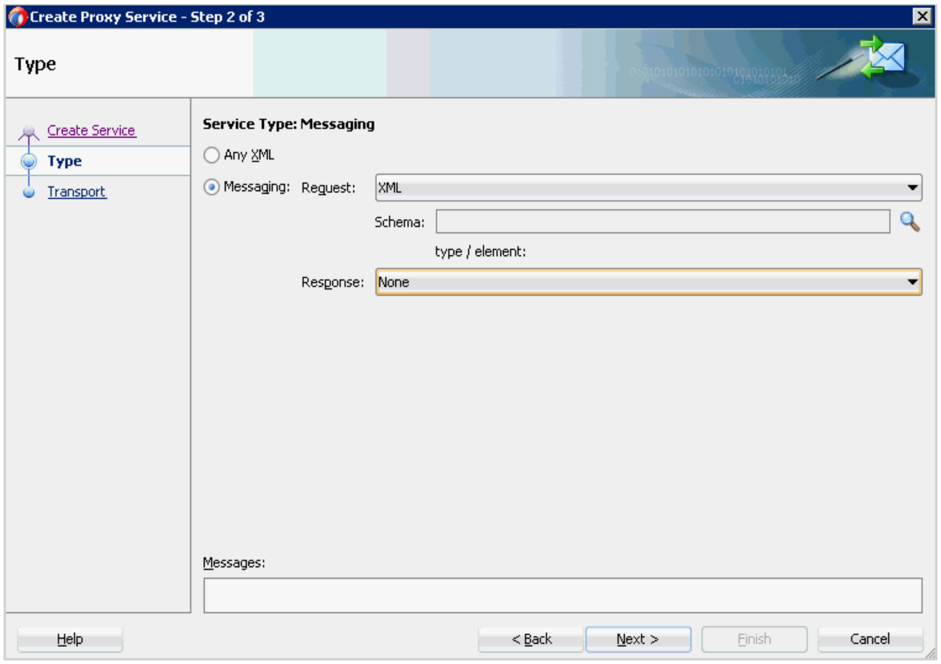


General
Service Name: iBSE_Outbound_PS
Location: C:\soabeta\WORK\mywork\ServiceBus_Application\iBSE_Outbound
Description
Definition
Transport: file
Generate Pipeline: ✓
Pipeline Name: iBSE_Outbound_PSPipeline
Messages:

The Type pane is displayed.

4. Select the **Messaging** option, set the Request to **XML** and Response as **None**, and then click **Next**, as shown in Figure 8–55.

Figure 8–55 Type Pane

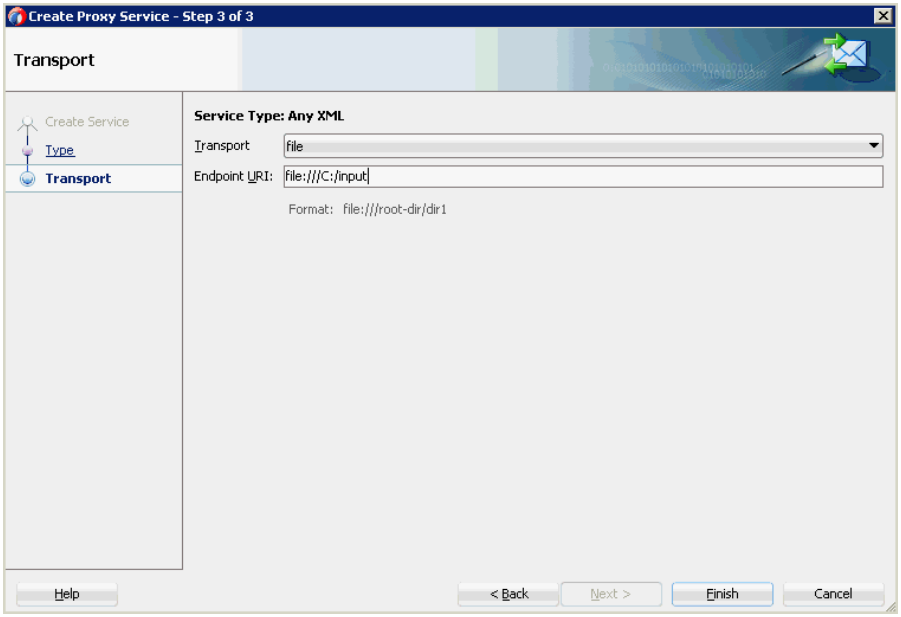


Service Type: Messaging
Any XML [✓ Messaging]
Request: XML
Schema:
type / element:
Response: None
Messages:

The Transport window is displayed.

5. Provide the input location in the Endpoint URI field (for example, c:/input) and click **Finish**, as shown in Figure 8–56.

Figure 8–56 Transport Window


Create Proxy Service - Step 3 of 3

Transport

Create Service
Type
Transport

Service Type: Any XML

Transport: file

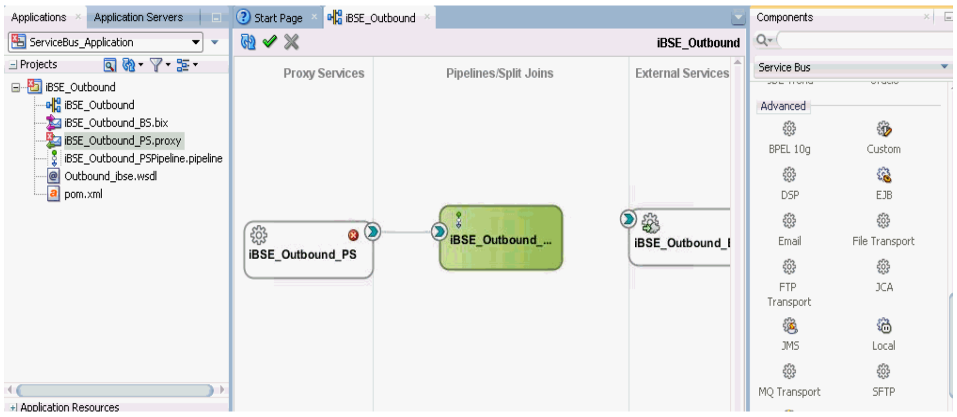
Endpoint URI: file:///C:/input

Format: file:///root-dir/dir1

The Proxy service along with the pipeline is created and displayed.

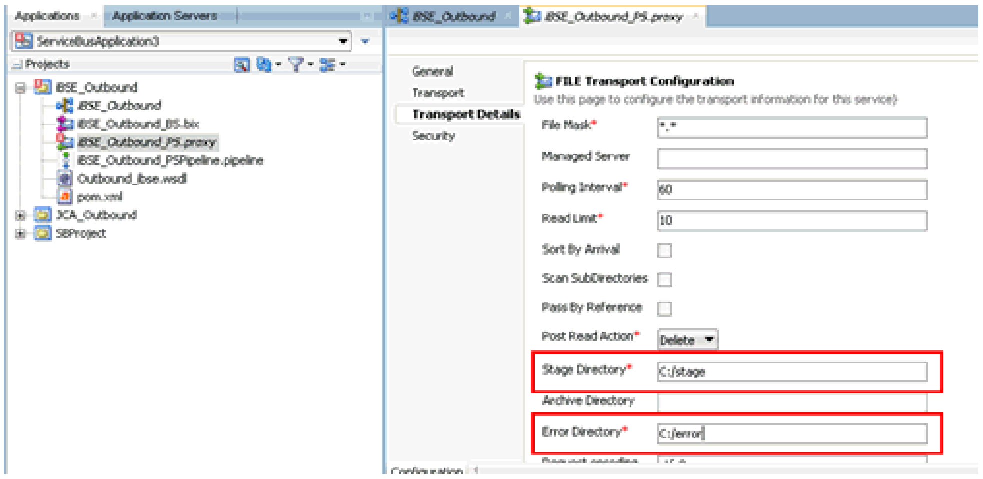
6. Double-click the created Proxy Service (for example: iBSE_Outbound_PS), as shown inFigure 8–57.

Figure 8–57 Proxy Service Edit



7. In the displayed Proxy Service configuration page, select **Transport Details** and provide the values for Stage and Error Directory, as shown in Figure 8–58.

Figure 8–58 File Transport Configuration



FILE Transport Configuration

Use this page to configure the transport information for this service)

File Mask* *.*

Managed Server

Polling Interval* 60

Read Limit* 10

Sort By Arrival

Scan SubDirectories

Pass By Reference

Post Read Action* Delete

Stage Directory* C:/stage

Archive Directory

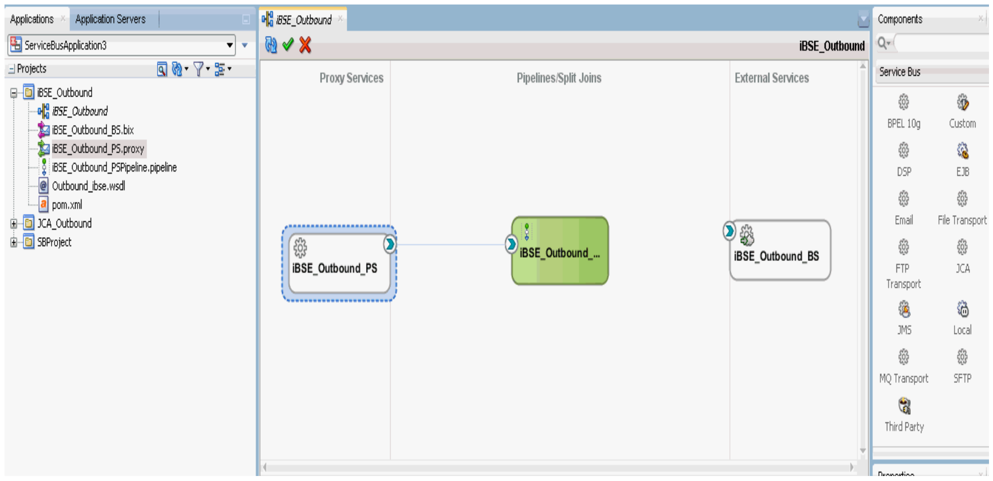
Error Directory* C:/error

8. Save and close the Proxy Service configuration page.

9. Double-click the overview.xml file (for example, iBSE_Outbound).

   The Proxy service is updated and displayed, as shown in Figure 8–59.

Figure 8–59 Proxy Service

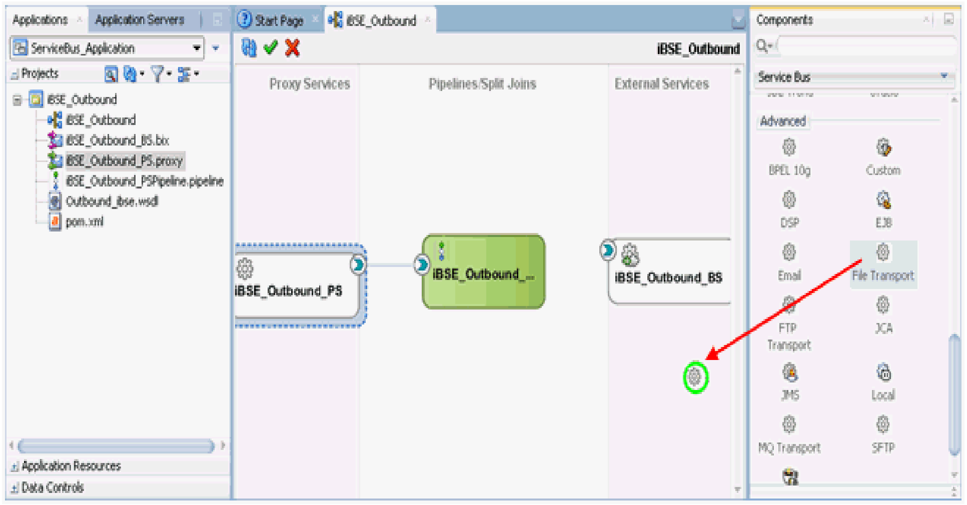


8.3.2.3 Configuring a File Transport Type Business Service

Perform the following steps to create a File Transport Type Business Service:

1. Drag and drop the **File Transport** component from the Advanced pane to the External Services pane, as shown in Figure 8–60.

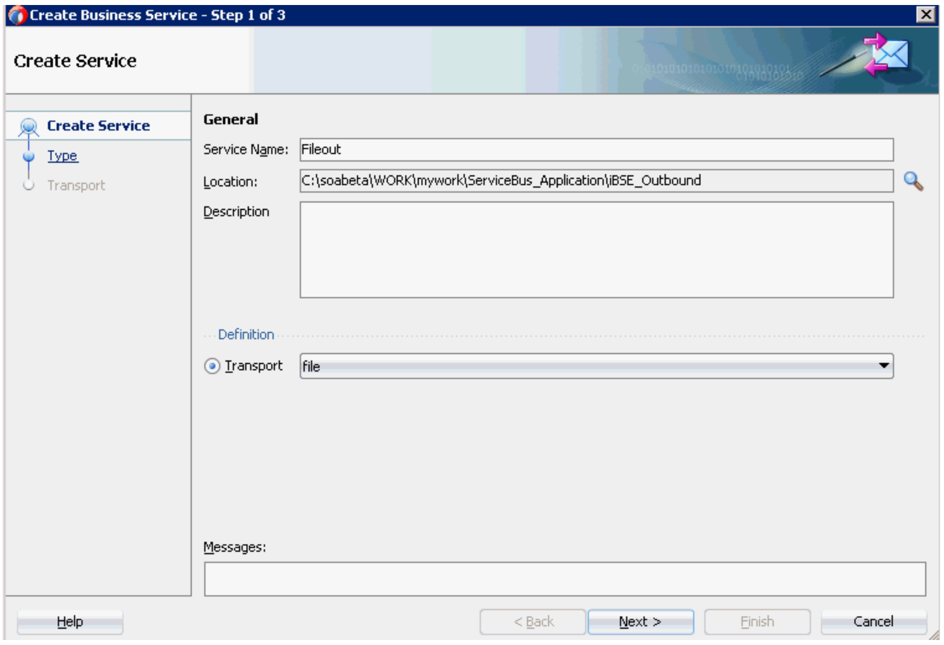
Figure 8–60 File Transport Component



The Create Business Service dialog is displayed.

2. In the Service Name field, enter any name you wish for the Business Service (for example, FileOut), and click **Next**, as shown in Figure 8–61.

Figure 8–61 Create Service Pane



Service Name: Fileout

Location: C:\soabeta\WORK\mywork\ServiceBus_Application\iBSE_Outbound

Transport: file

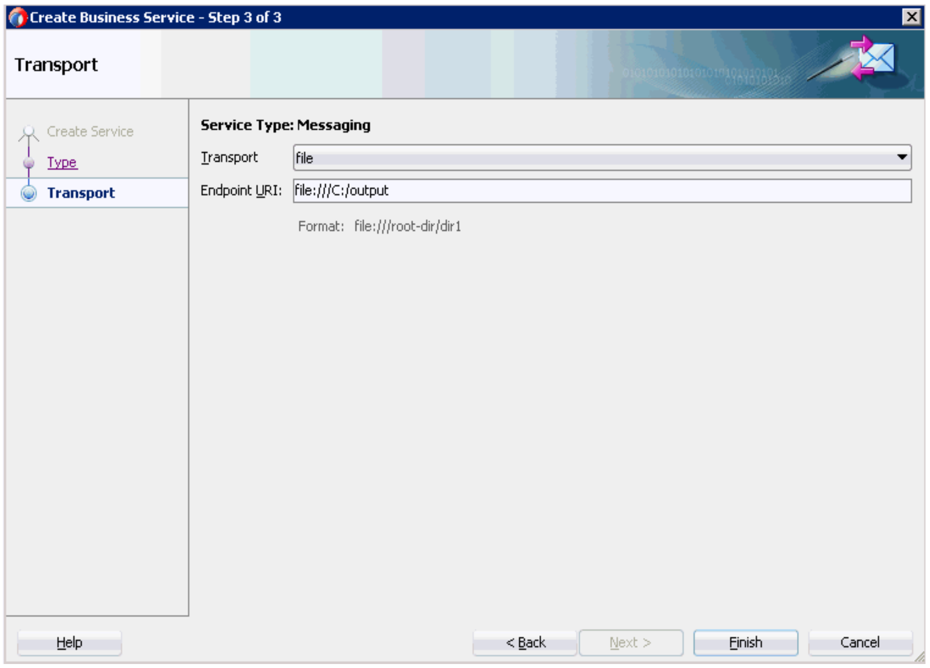
The Type pane is displayed. The **Any XML** option is selected by default.

3. Click **Next**.

The Transport pane appears.

4. Provide the output location in the Endpoint URI field (for example, c:/output) and click **Finish**, as shown in Figure 8–62.

Figure 8–62 Transport Pane



Service Type: Messaging

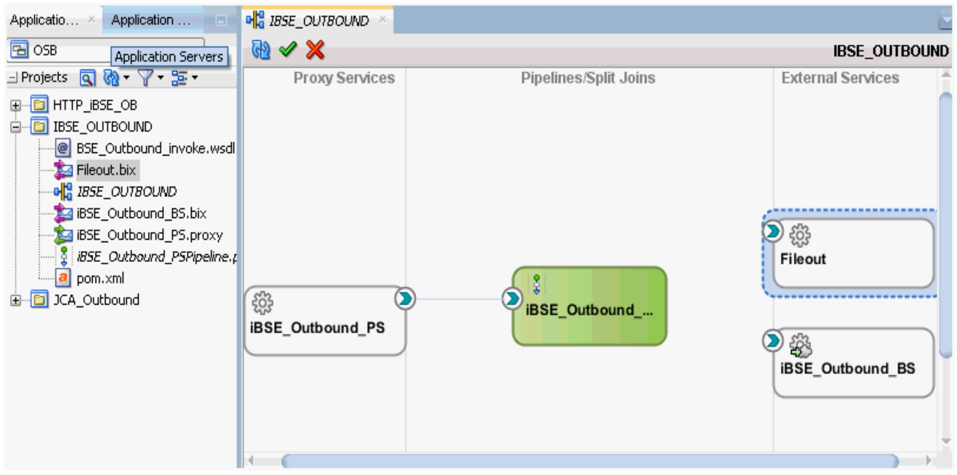
Transport: file

Endpoint URI: file:///C:/output

Format: file:///root-dir/dir1

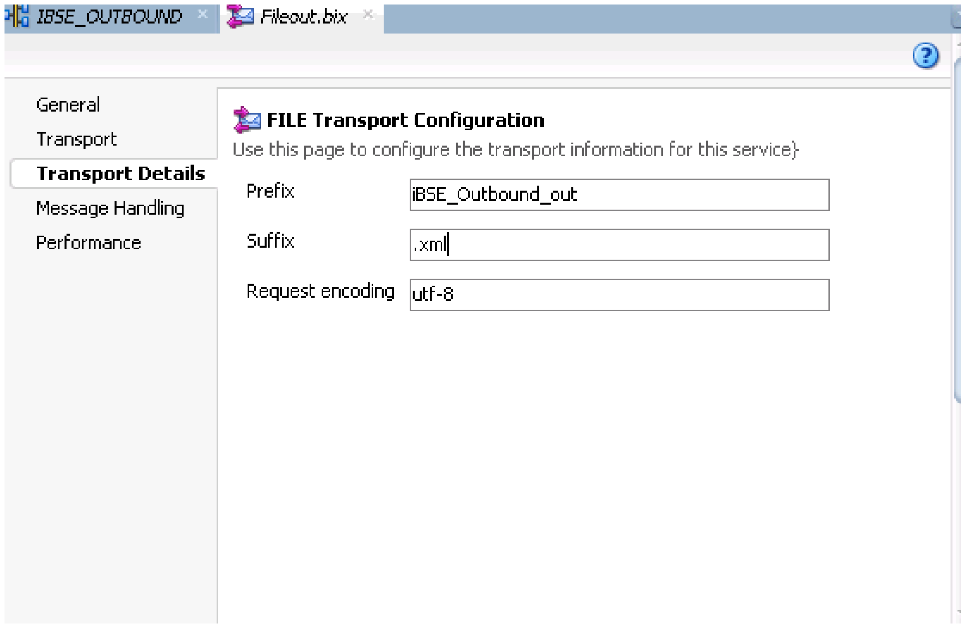
The File Transport Business service Fileout is created and displayed, as shown in Figure 8–63.

Figure 8–63 Fileout Business Service



5. Double-click the created Business service **Fileout** and provide the values for the Prefix and Suffix fields in the Transport Details Tab, as shown in Figure 8–64.

Figure 8–64 Transport Details



FILE Transport Configuration

Use this page to configure the transport information for this service}

Prefix	iBSE_Outbound_out
Suffix	.xml
Request encoding	utf-8

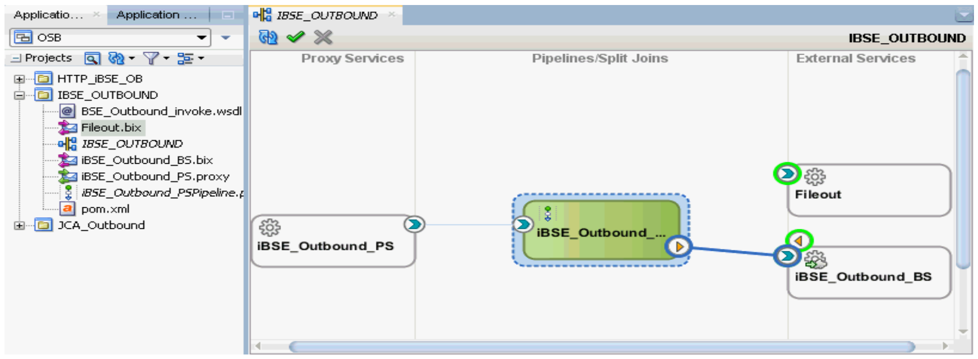
6. Save and close the configuration page, and double-click on overview.xml (for example, iBSE_Outbound).

8.3.2.4 Configuring the Routing Rules

Perform the following steps to configure the routing rules:

1. Create a connection between the Pipeline Component (for example, iBSE_Outbound_PSPipeline) and the WSDL based Business Service (for example, iBSE_Outbound_BS), as shown in Figure 8–65.

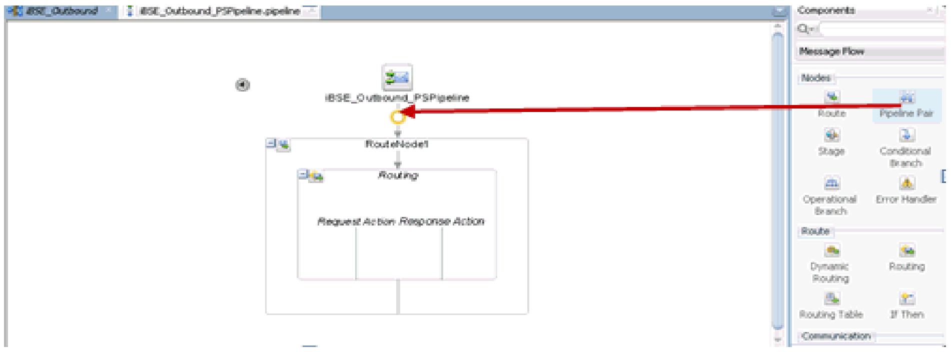
Figure 8–65 Pipeline Component



2. Double-click on the **Pipeline** component (for example, iBSE_Outbound_PSPipeline) in the Pipelines/Split Joins pane.

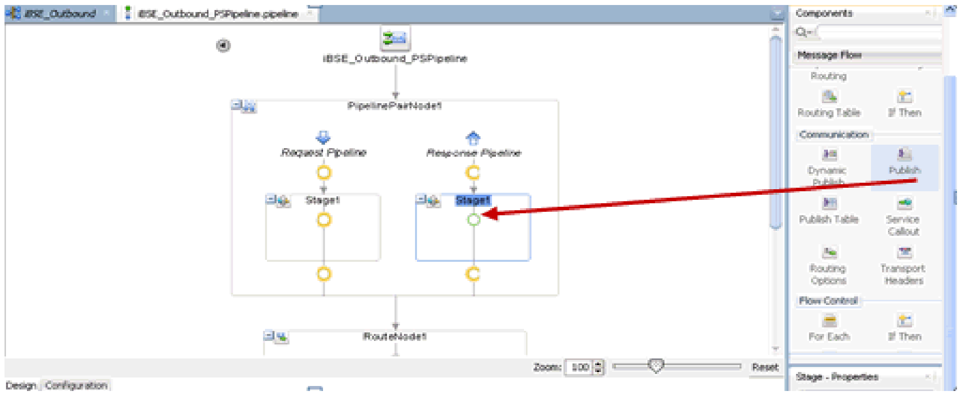
3. Drag and drop the **Pipeline Pair** node from Nodes pane to the area between the Pipeline (for example: iBSE_Outbound_PSPipeline) and RouteNode1, as shown in Figure 8–66.

Figure 8–66 Pipeline Pair Node



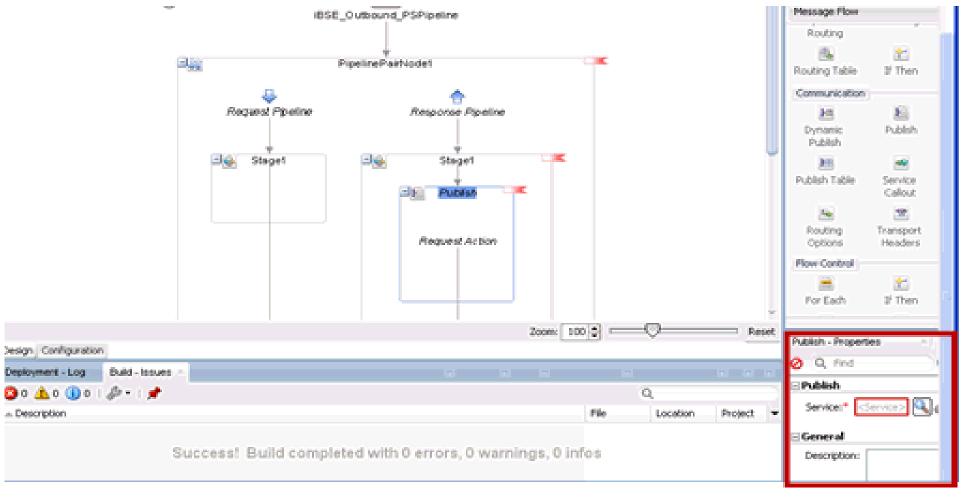
4. Drag and drop the **Publish** node from the Communication pane to the area beneath Stage1 of the Response Pipeline, as shown in Figure 8–67.

Figure 8–67 Publish Node



5. Click on the browse icon to the right of the Service field in the right pane of Publish Properties, as shown in Figure 8–68.

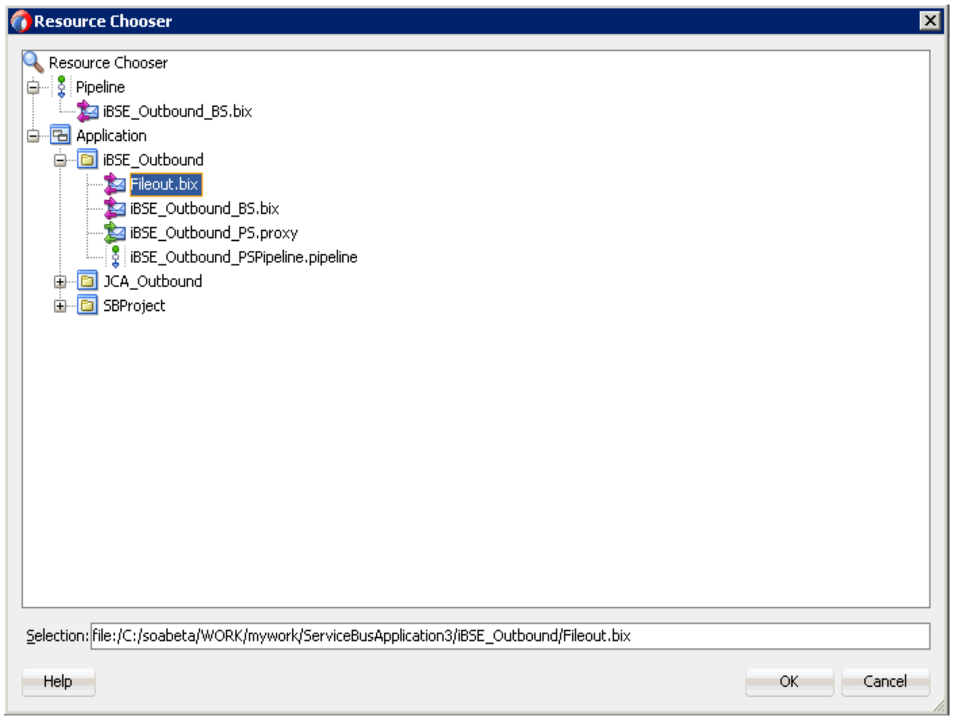
Figure 8–68 Browse Icon



Success! Build completed with 0 errors, 0 warnings, 0 infos

6. In the displayed Resource Chooser window, select the **Fileout.bix** File Transport Business service and click **OK**, as shown in Figure 8–69.

Figure 8–69 Resource Chooser

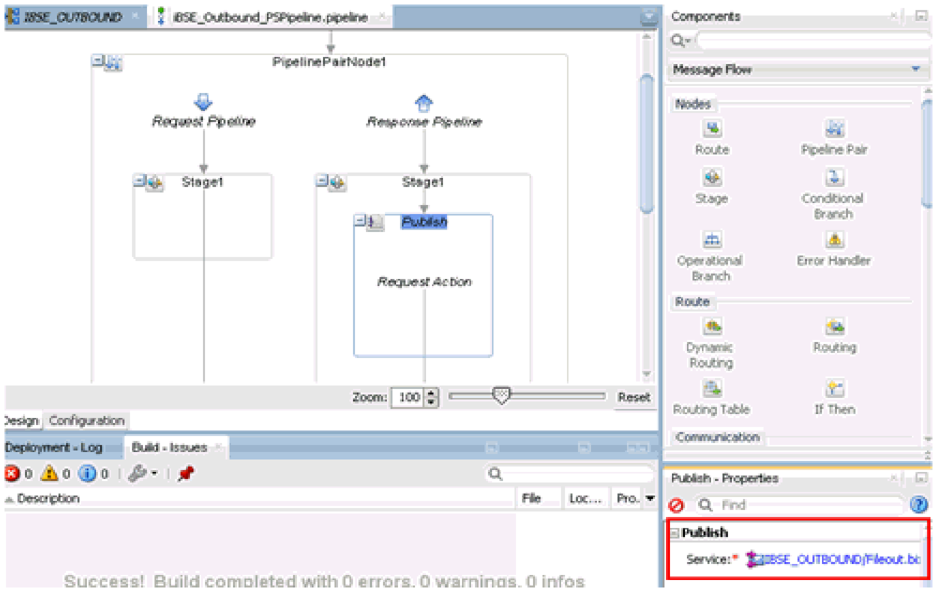


Selection: file:/C:/soabeta/WORK/mywork/ServiceBusApplication3/iBSE_Outbound/Fileout.bix

You are returned to the Pipeline configuration page.

In the right pane, the selected service is configured in the Publish pane, as shown in Figure 8–70.

Figure 8–70 Publish Pane

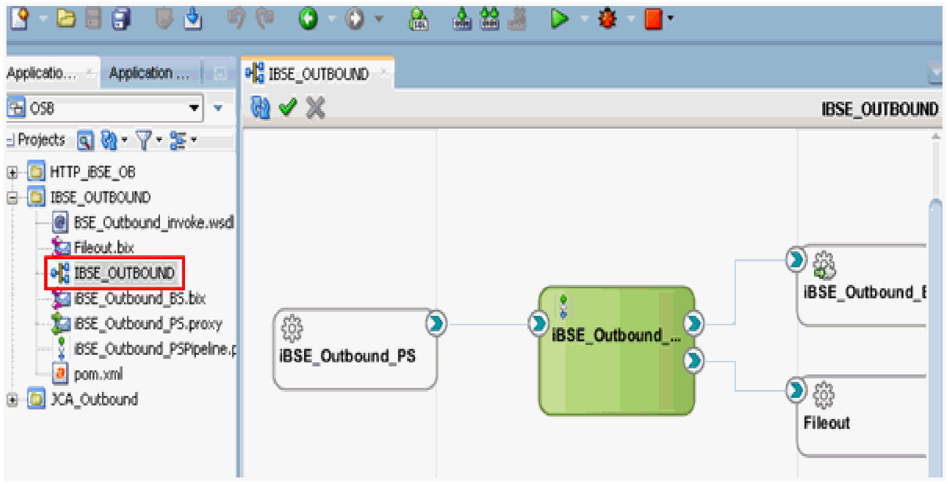


Service:* iBSE_OUTBOUND/Fileout.bix

Success! Build completed with 0 errors, 0 warnings, 0 infos

7. Save and close the Pipeline configuration page.
8. Double-click the overview.xml file (for example: iBSE_Outbound), and click **Save All** in the menu bar to save the OSB process, as shown in Figure 8–71.

Figure 8–71 Save All Icon



8.3.3 Deploying the OSB Outbound Process

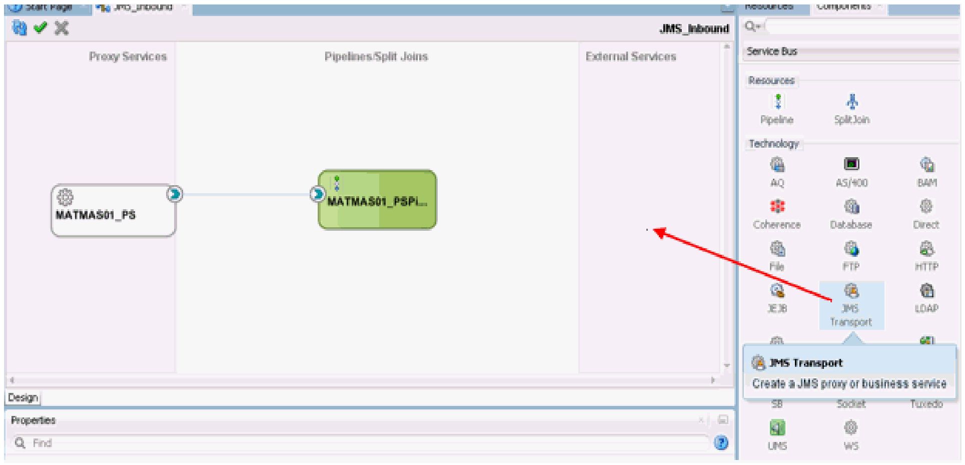
To deploy the created OSB outbound process and invoke the input XML document, see Section 8.1.3, "Deploying the OSB Outbound Process".

8.4 Configuring a JMS Inbound Process Using JDeveloper (J2CA Configuration)

This section describes how to configure a JMS inbound process to your J.D. Edwards OneWorld system, using Oracle JDeveloper for J2CA configurations.

1. Before you design a JMS process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.5.1, "Generating WSDL for Event Integration" on page 4-34.
2. Start the Oracle JDeveloper and create a Service Bus Application for OSB. For more information, see Section 8.1.1, "Creating a Service Bus Application for OSB" on page 8-2.
3. Create a Third Party Adapter Service Component. For more information, see Section 8.2.2.1, "Configuring a Third-Party Adapter Service Component" on page 8-19.
4. Create a Proxy Service along with the pipeline from the JCA Binding File. For more information, see Section 8.2.2.2, "Creating a Pipeline" on page 8-21.
5. Create a JMS Transport Business Service and perform the following steps:
   a. Drag and drop the **JMS Transport** component from the Technology Components pane to the External Services pane, as shown in Figure 8–72.

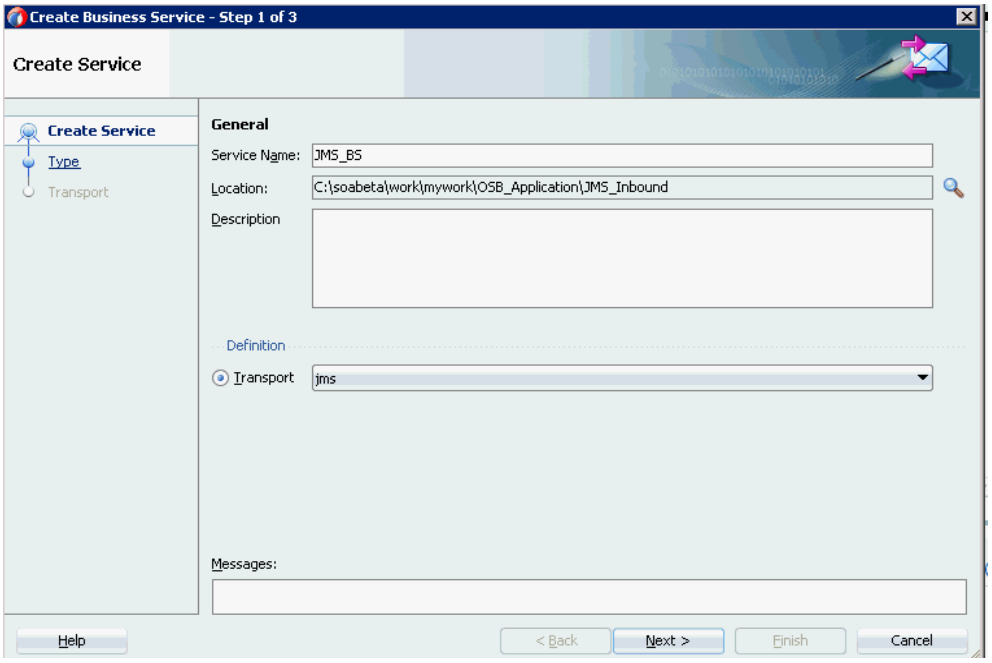
Figure 8–72 JMS Transport Component



The Create Business Service dialog is displayed.

b. In the Service Name field, enter any name you wish for the Business service (for example, JMS_BS) and click **Next**, as shown in Figure 8–73.

Figure 8–73 Create Service Pane



Service Name: JMS_BS

Location: C:\soabeta\work\mywork\OSB_Application\JMS_Inbound

Description

Transport: jms

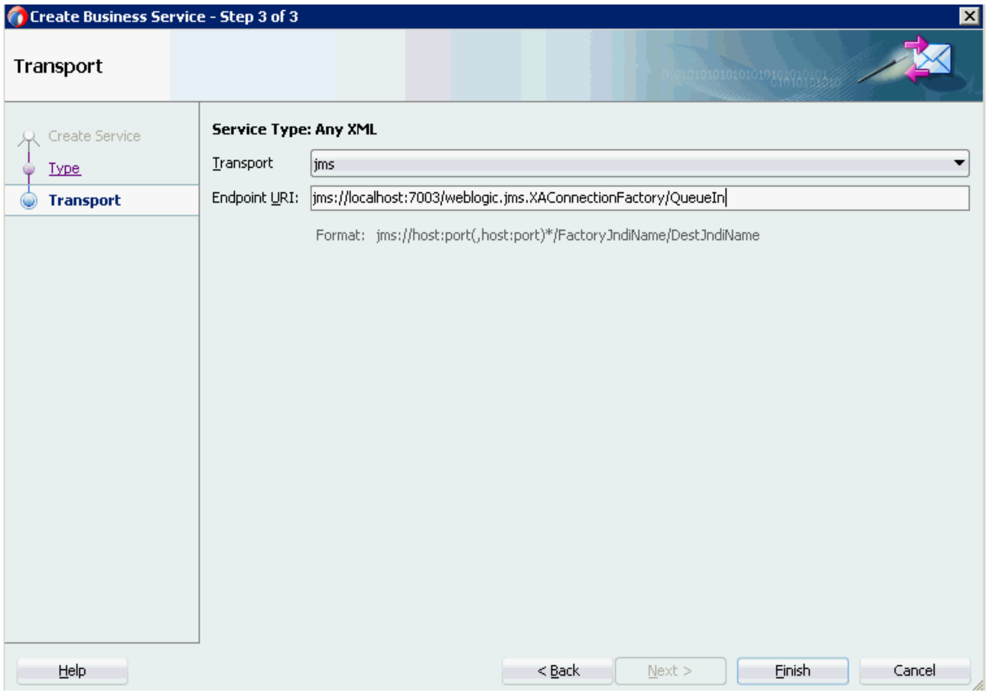
Messages:

c. In the displayed Type window, select **Any XML** and then click **Next**.

The Transport window is displayed.

d. Modify the appropriate hostname and port number by replacing `DestJndiName` with `QueueIn` in the Endpoint URI field (for example, `jms://localhost:7003/weblogic.jms.XAConnectionFactory/QueueIn`), and then click **Finish**, as shown in Figure 8–74.

Figure 8–74 Transport Window



Create Business Service - Step 3 of 3

Transport

Service Type: Any XML

Transport: jms

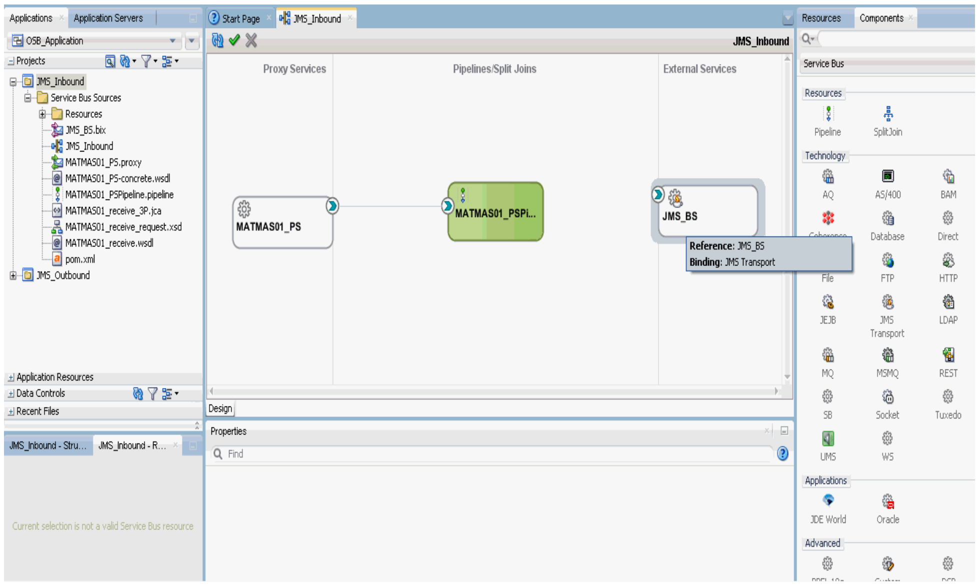
Endpoint URI: jms://localhost:7003/weblogic.jms.XAConnectionFactory/QueueIn

Format: jms://host:port(,host:port)*/FactoryJndiName/DestJndiName

The JMS Business service is created and displayed.

e. Double-click **JMS_BS** as shown in Figure 8–75.

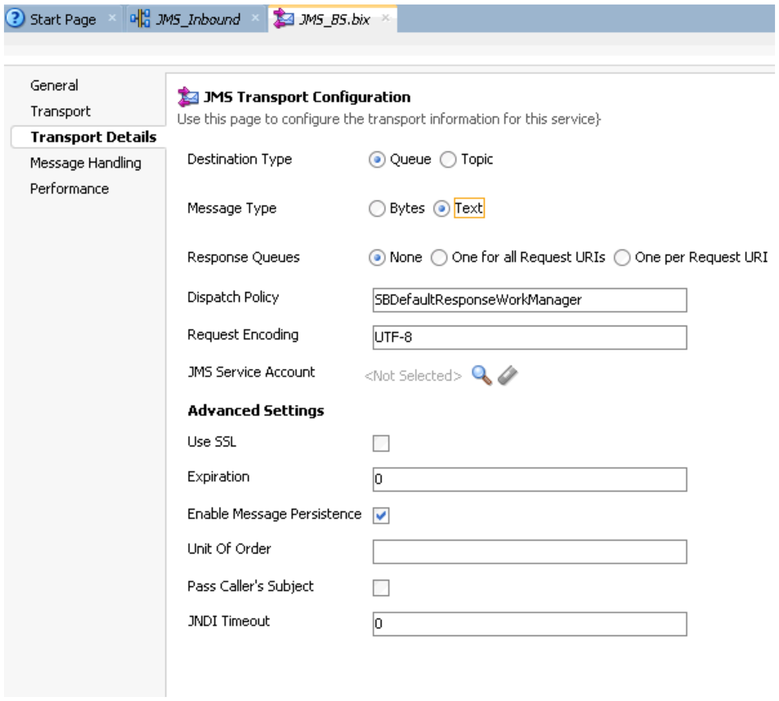
Figure 8–75 JMS Business Service



Reference: JMS_BS
Binding: JMS Transport

f. In the displayed Business Service configuration page, provide the following parameters in the Transport Details tab, as shown in Figure 8–76.

Figure 8–76 JMS Transport Configuration



JMS Transport Configuration

Use this page to configure the transport information for this service}

Destination Type	[✓ Queue] Topic
Message Type	Bytes [✓ Text]
Response Queues	[✓ None] One for all Request URIs One per Request URI
Dispatch Policy	SBDefaultResponseWorkManager
Request Encoding	UTF-8
JMS Service Account	<Not Selected>

Advanced Settings

Use SSL	☐
Expiration	0
Enable Message Persistence	☑
Unit Of Order	
Pass Caller's Subject	☐
JNDI Timeout	0

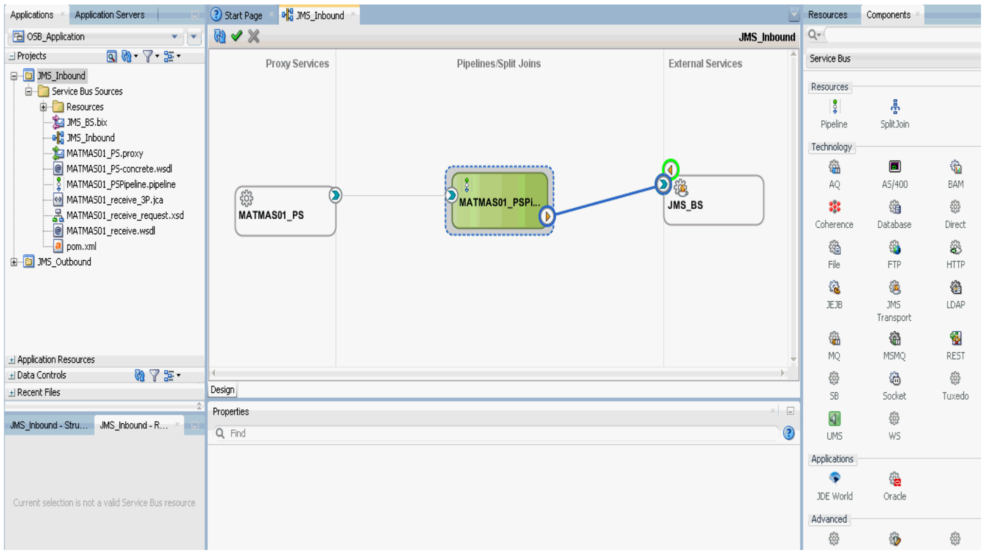
g. In the Destination Type section, select **Queue**.

h. In the Message Type section, select **Text**.

6. Save and close the Configuration page of the business service.

7. Create a connection between **Pipeline** (for example, xxxx_PSPipeline) and **JMS Business Service** (for example, JMS_BS) as shown in figure Figure 8–77.

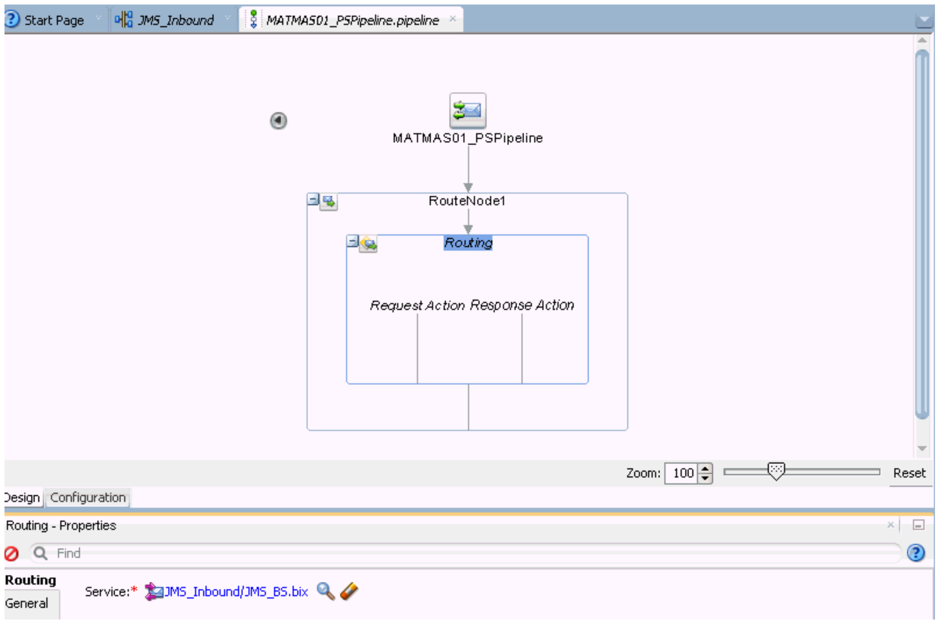
Figure 8–77 Configuration Page



8. Double-click **Pipeline**.

The Pipeline Configuration page is displayed as shown in Figure 8–78.

Figure 8–78 Pipeline Configuration

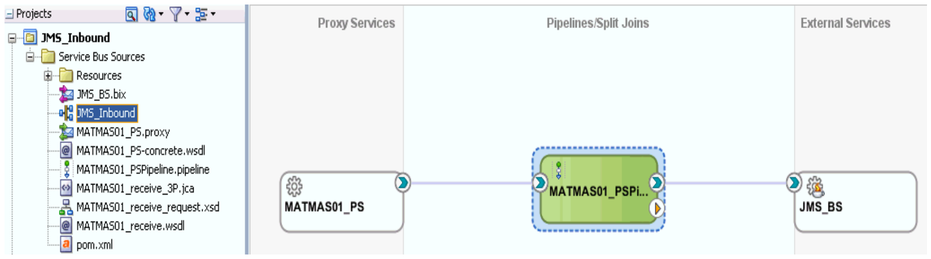


9. Check that the details are configured properly, and then save and close the Pipeline configuration page.

   You are returned to the composite editor window.

10. Click **Save All** in the menu bar to save the OSB JMS process, as shown in Figure 8–79.

Figure 8–79 Save All Icon



11. Deploy the OSB JMS inbound process. For more information, see Section 8.2.3, "Deploying the OSB Inbound Process" on page 8-26.

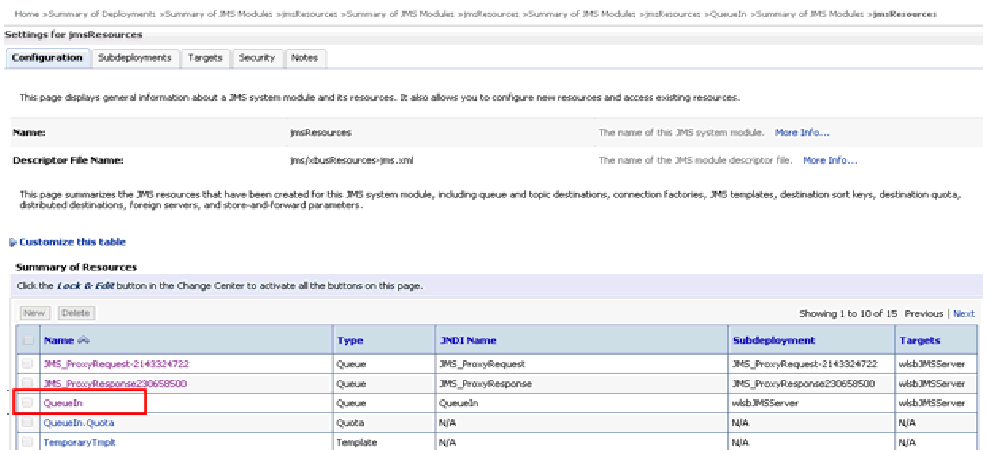
12. Once the process is deployed successfully, trigger the event messages. For more information, see Section 4.5.5, "Triggering an Event in J.D. Edwards OneWorld" on page 4-47.

13. Log on to the Oracle WLS console.

14. In the Oracle WLS console, expand **Services**, click **Messaging**, select **JMS Modules**, and then click **jmsResources**.

15. Click the appropriate response link (for example, QueueIn) as shown in Figure 8–80.

Figure 8–80 QueueIn Response Link


Home >Summary of Deployments >Summary of JMS Modules >jmsResources >Summary of JMS Modules >jmsResources >Summary of JMS Modules >jmsResources >QueueIn >Summary of JMS Modules >**jmsResources**

Settings for jmsResources

This page displays general information about a JMS system module and its resources. It also allows you to configure new resources and access existing resources.

Name:	jmsResources	The name of this JMS system module. More Info...
Descriptor File Name:	jms/xbusResources-jms.xml	The name of the JMS module descriptor file. More Info...

This page summarizes the JMS resources that have been created for this JMS system module, including queue and topic destinations, connection factories, JMS templates, destination sort keys, destination quota, distributed destinations, foreign servers, and store-and-forward parameters.

Summary of Resources

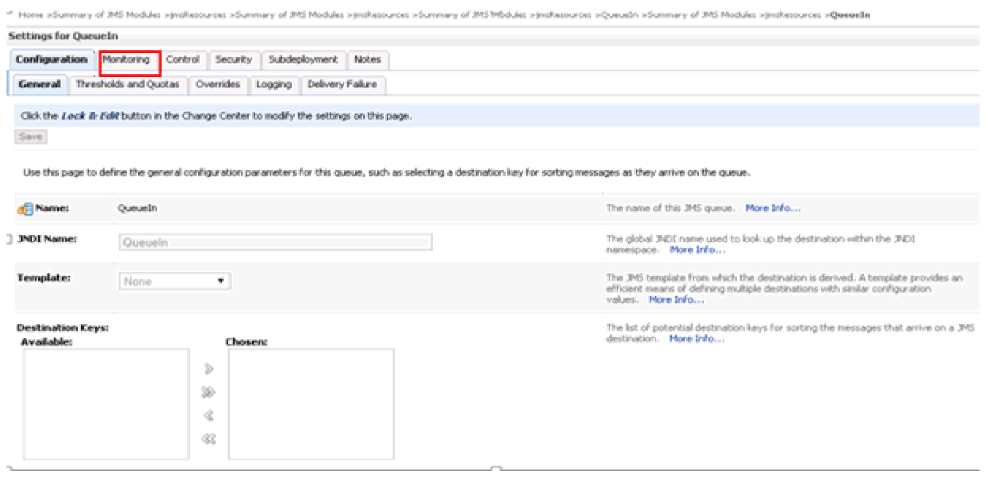
Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

Showing 1 to 10 of 15 Previous | Next

Name	Type	JNDI Name	Subdeployment	Targets
JMS_ProxyRequest-2143324722	Queue	JMS_ProxyRequest	JMS_ProxyRequest-2143324722	wlsbJMSServer
JMS_ProxyResponse230658500	Queue	JMS_ProxyResponse	JMS_ProxyResponse230658500	wlsbJMSServer
QueueIn	Queue	QueueIn	wlsbJMSServer	wlsbJMSServer
QueueIn.Quota	Quota	N/A	N/A	N/A
TemporaryTmplt	Template	N/A	N/A	N/A

16. Click the Monitoring tab, as shown in Figure 8–81.

Figure 8–81 Monitoring Tab


Home >Summary of JMS Modules >jmsResources >Summary of JMS Modules >jmsResources >Summary of JMS Modules >jmsResources >QueueIn >Summary of JMS Modules >jmsResources >**QueueIn**

Settings for QueueIn

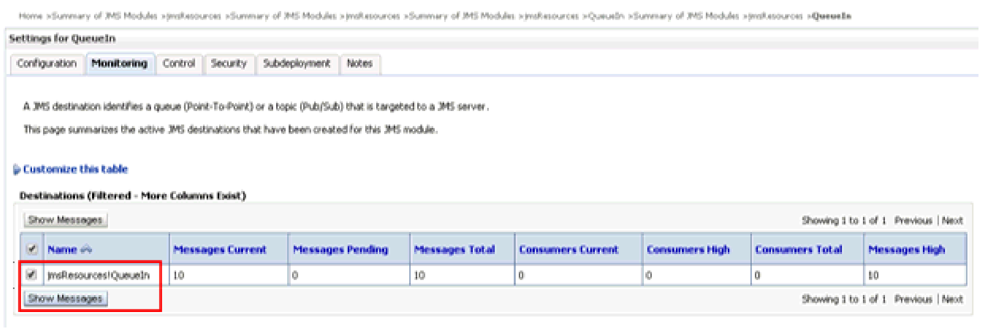
Click the *Lock & Edit* button in the Change Center to modify the settings on this page.

Use this page to define the general configuration parameters for this queue, such as selecting a destination key for sorting messages as they arrive on the queue.

Name:	QueueIn	The name of this JMS queue. More Info...
JNDI Name:	QueueIn	The global JNDI name used to look up the destination within the JNDI namespace. More Info...
Template:	None	The JMS template from which the destination is derived. A template provides an efficient means of defining multiple destinations with similar configuration values. More Info...
Destination Keys:		The list of potential destination keys for sorting the messages that arrive on a JMS destination. More Info...

17. Select the check box and click the **Show Messages** button, as shown in Figure 8–82.

Figure 8–82 Show Messages Button


Home >Summary of JMS Modules >jmsResources >Summary of JMS Modules >jmsResources >Summary of JMS Modules >jmsResources >QueueIn >Summary of JMS Modules >jmsResources >**QueueIn**

Settings for QueueIn

A JMS destination identifies a queue (Point-To-Point) or a topic (Pub/Sub) that is targeted to a JMS server.

This page summarizes the active JMS destinations that have been created for this JMS module.

Destinations (Filtered - More Columns Exist)

Showing 1 to 1 of 1 Previous | Next

✓	Name	Messages Current	Messages Pending	Messages Total	Consumers Current	Consumers High	Consumers Total	Messages High
✓	jmsResources!QueueIn	10	0	10	0	0	0	10

Showing 1 to 1 of 1 Previous | Next

18. Click the ID link with the appropriate time and date.

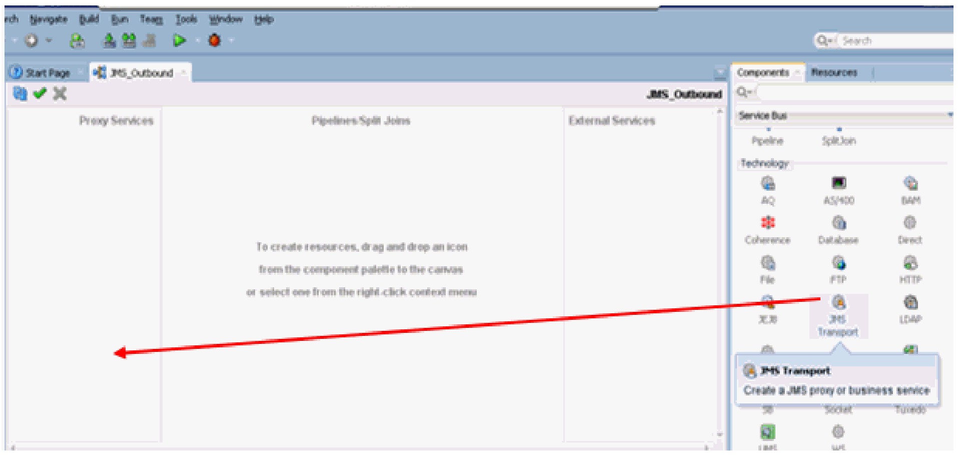
    The response document is shown under the Text field.

8.5 Configuring a JMS Outbound Process Using JDeveloper (J2CA Configuration)

This section describes how to configure a JMS outbound process to your J.D. Edwards OneWorld system, using Oracle JDeveloper for J2CA configurations.

1. Before you design a JMS process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.4.1, "Generating WSDL for Request/Response Service" on page 4-8.
2. Start the Oracle JDeveloper and create a Service Bus Application for OSB. For more information, see Section 8.1.1, "Creating a Service Bus Application for OSB" on page 8-2.
3. Create a Third Party Adapter Service Component. For more information, see Section 7.3.2.1, "Configuring a Third Party Adapter Service Component" on page 7-14.
4. Create a WSDL-based Business Service from the JCA Binding File. For more information, see Section 8.1.2.1, "Configuring a Third-Party Adapter Service Component" on page 8-3.
5. Create a JMS Proxy Service with a Pipeline and perform the following steps:
   a. Drag and drop the **JMS Transport** component from the Technology Components pane to the Proxy Services pane, as shown in Figure 8–83.

Figure 8–83 JMS Transport Component

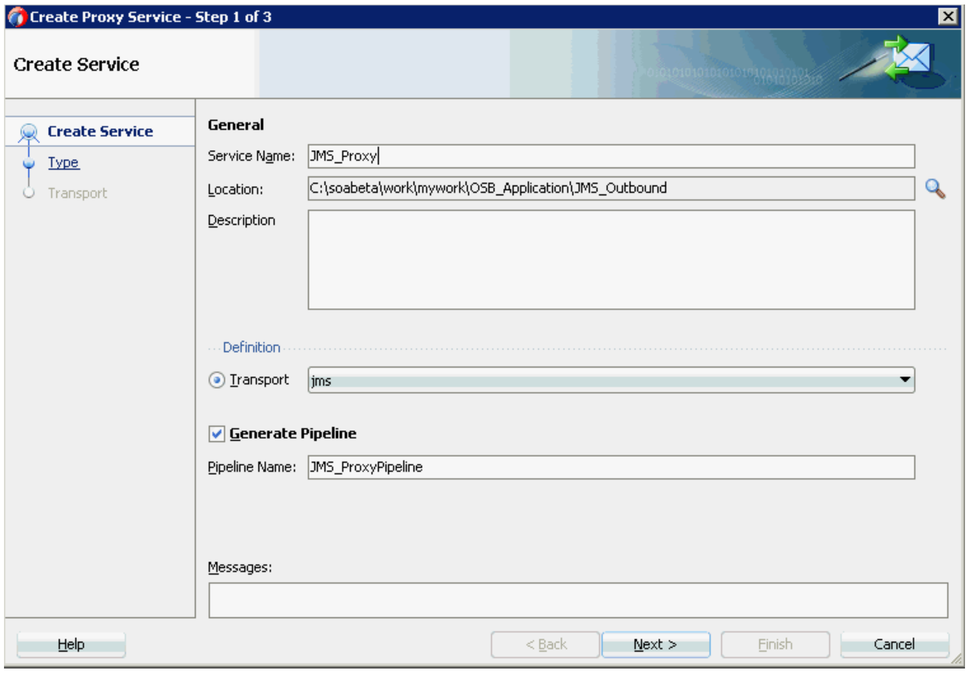


To create resources, drag and drop an icon from the component palette to the canvas or select one from the right-click context menu

   The Create Business Service dialog is displayed.

   b. In the Service Name field, enter any name you wish for the Proxy service (for example, JMS_Proxy). By default, Generate Pipeline is selected.
   c. Click **Next**, as shown in Figure 8–84.

Figure 8–84 Create Proxy Service Pane



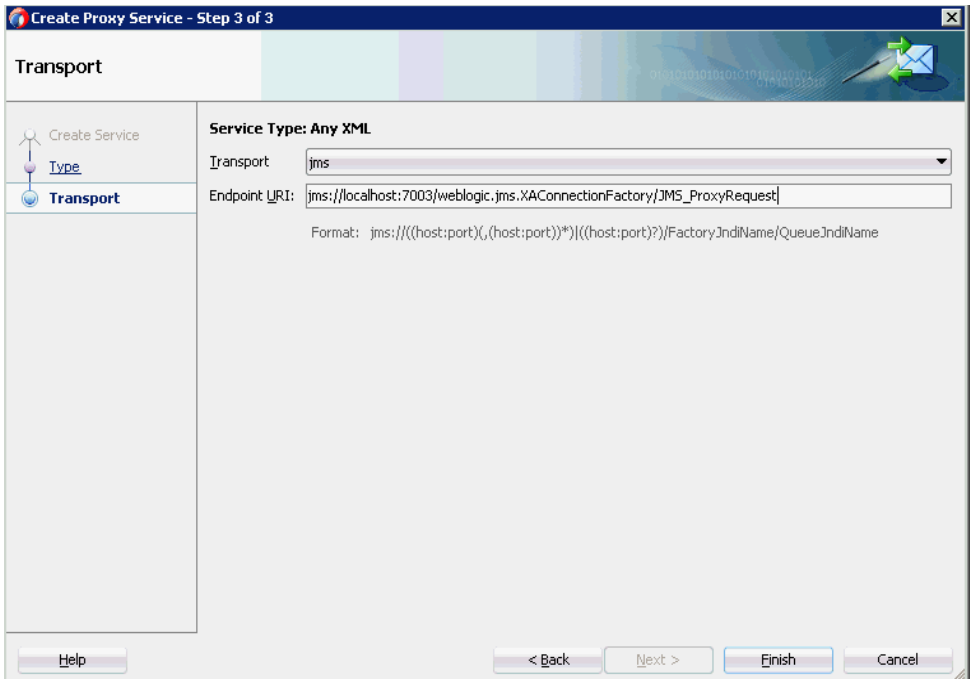
General
Service Name: JMS_Proxy
Location: C:\soabeta\work\mywork\OSB_Application\JMS_Outbound
Description
Definition
Transport: jms
Generate Pipeline: ✓
Pipeline Name: JMS_ProxyPipeline
Messages:

d. In the displayed Type window, select **Any XML** and then click **Next**.

The Transport window is displayed.

e. Modify the appropriate hostname and port number by replacing the Endpoint URI field (for example, `jms://localhost:7003/weblogic.jms.XAConnectionFactory/JMS_ProxyRequest`), and then click **Finish**, as shown in Figure 8–85.

Figure 8–85 Transport Window

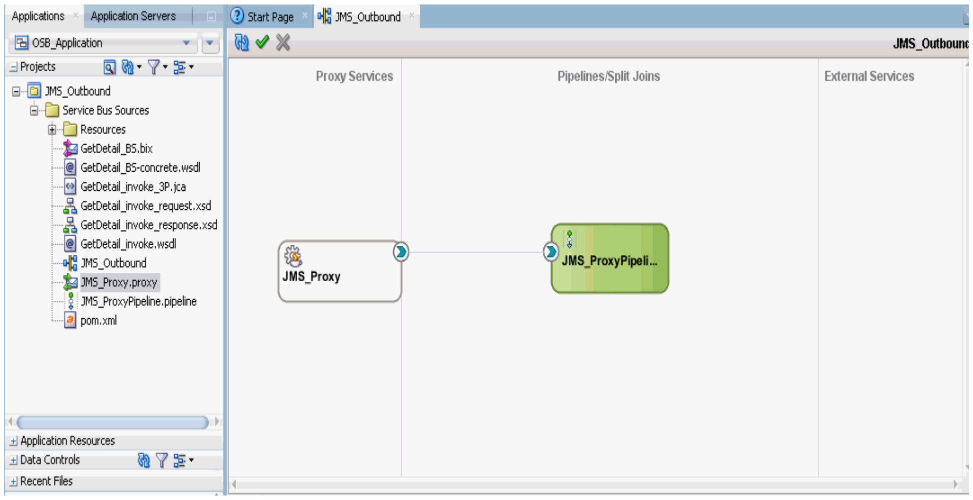


Service Type: Any XML
Transport: jms
Endpoint URI: jms://localhost:7003/weblogic.jms.XAConnectionFactory/JMS_ProxyRequest
Format: jms://((host:port)(,(host:port))*)|((host:port)?)/FactoryJndiName/QueueJndiName

The JMS Proxy service along with the pipeline is created and displayed.

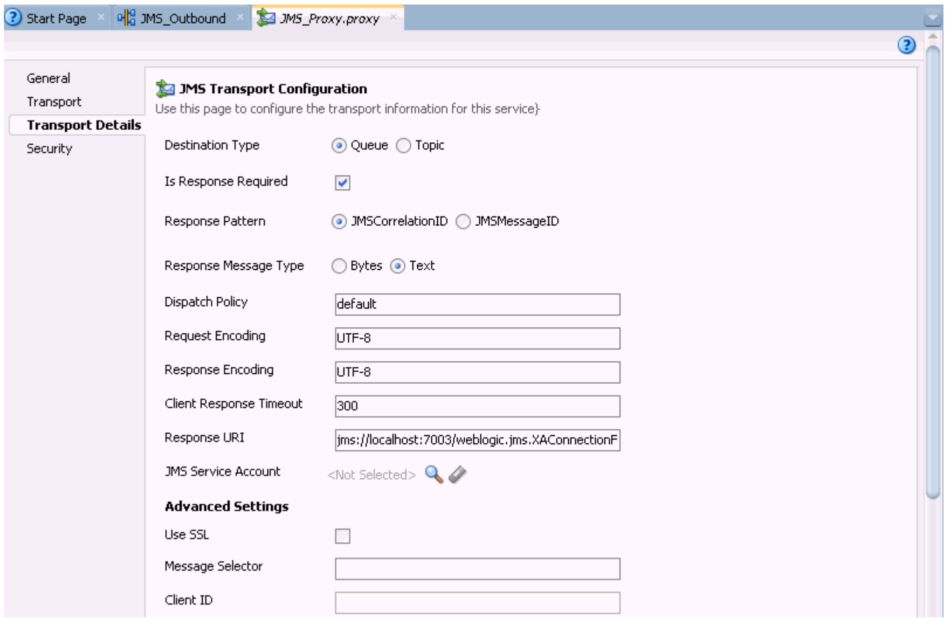
f. Double-click the created Proxy Service (for example, JMS_Proxy), as shown in Figure 8–86.

Figure 8–86 JMS Proxy Service



g. In the displayed configuration page of the Proxy Service, provide the following parameters in the Transport Details tab, as shown in Figure 8–87.

Figure 8–87 JMS Transport Configuration



JMS Transport Configuration

Use this page to configure the transport information for this service}

Destination Type	[✓ Queue] Topic
Is Response Required	✓
Response Pattern	[✓ JMSCorrelationID] JMSMessageID
Response Message Type	Bytes [✓ Text]
Dispatch Policy	default
Request Encoding	UTF-8
Response Encoding	UTF-8
Client Response Timeout	300
Response URI	jms://localhost:7003/weblogic.jms.XAConnectionF
JMS Service Account	<Not Selected>

Advanced Settings

Use SSL	
Message Selector	
Client ID	

h. In the Destination Type section, select **Queue**.

i. Select the **Is Response Required** check box.

j. In the Response Message Type section, select **Text**.

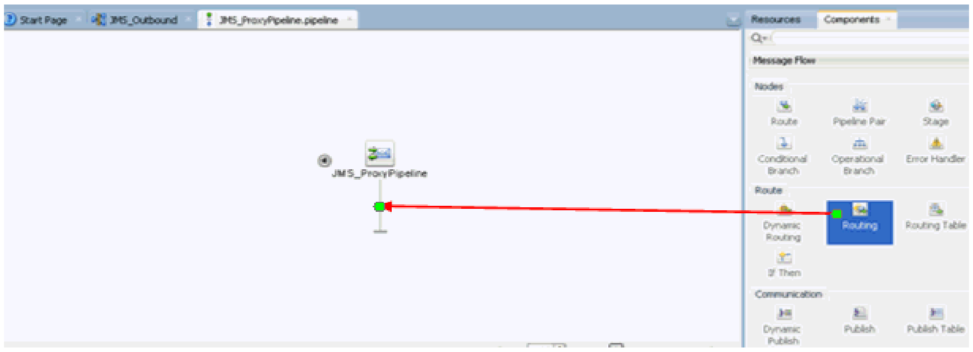
k. In the Response URI field, provide the Endpoint URI used in the JMS Transport Configuration and change `Request` to `Response`. For example,

```
jms://localhost:7003/weblogic.jms.XAConnectionFactory/JMS_
ProxyResponse
```

6. Save and close the Configuration page of the Proxy service.
7. Configure the Routing Rules and proceed with the following steps:
   a. Double-click on the pipeline (for example, JMS_ProxyPipeline) in the Pipelines/Split Joins pane.

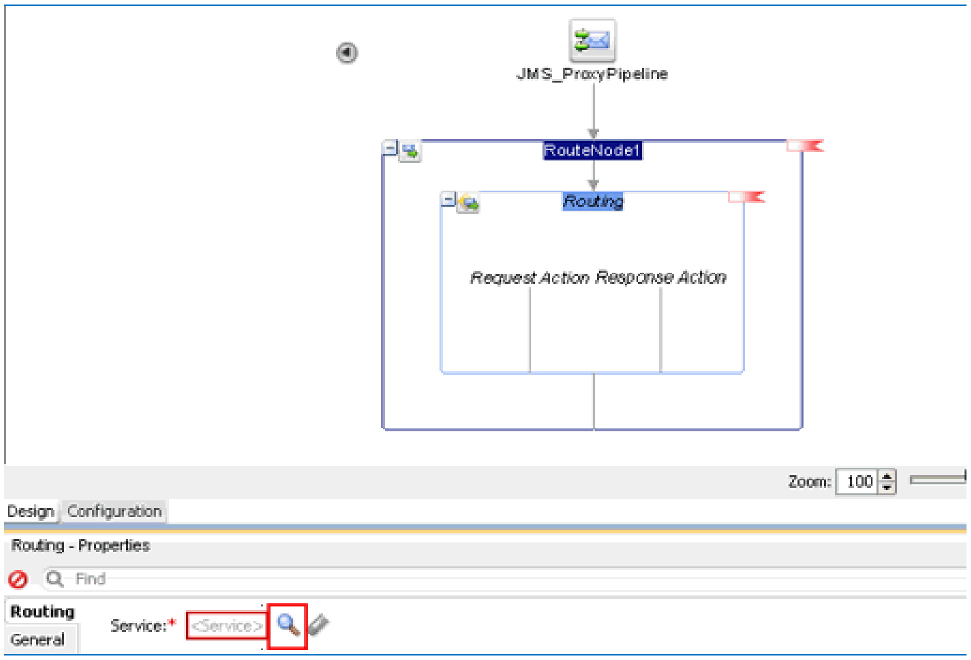
      The Pipeline configuration page is displayed.
   b. Drag and drop the **Routing** component from the Route section to the area below the Pipeline (for example, JMS_ProxyPipeline), as shown in Figure 8–88.

Figure 8–88 Routing Component



   c. In the Pipeline Configuration page, select **Routing** and click the browse icon to the right of the Service field in the Routing Properties pane, as shown in Figure 8–89.

Figure 8–89 Browse Service



d. In the displayed Resource Chooser window, select the WSDL-based Business service (for example, xxxxx_BS.bix) and click **OK**.

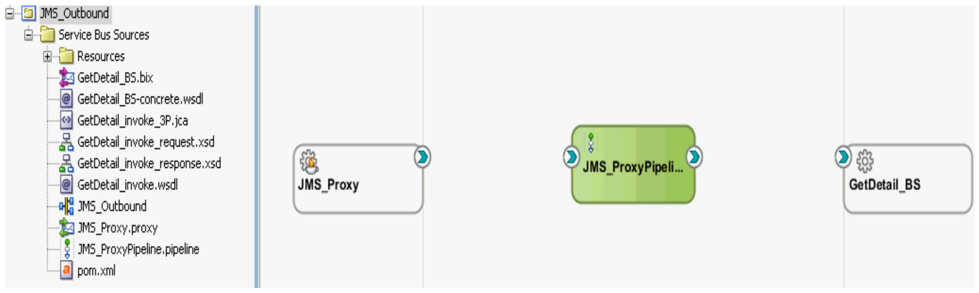
You are returned to the Pipeline configuration page.

e. Save and Close the Pipeline configuration page.

You are returned to the composite editor window.

f. Click **Save All** in the menu bar to save the OSB JMS process, as shown in Figure 8–90.

Figure 8–90 Transport Window

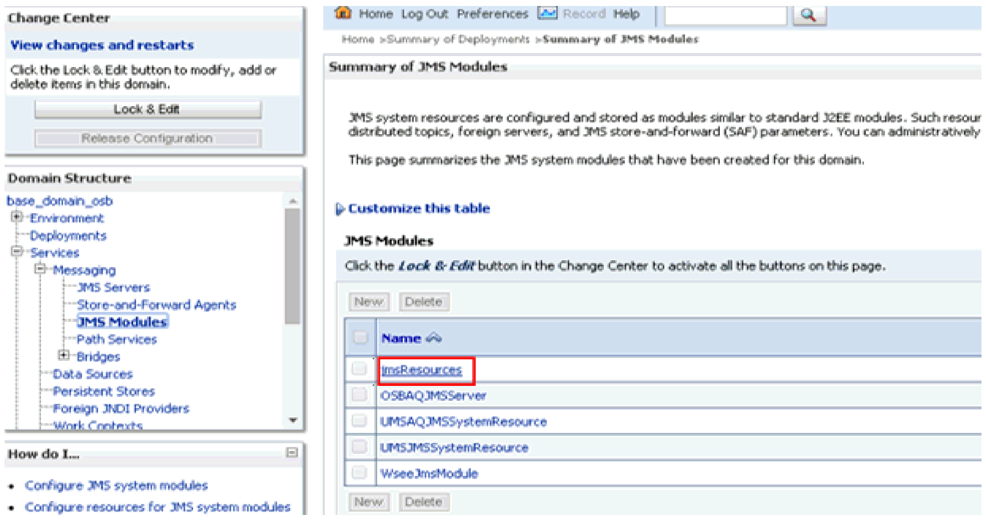


8. Deploy the OSB JMS outbound process. For more information, see Section 8.1.3, "Deploying the OSB Outbound Process" on page 8-15.

9. Once the process is deployed successfully, log on to the Oracle WLS Console.

10. In the Oracle WLS console, expand **Services**, click **Messaging**, select **JMS Modules**, and then click **jmsResources**, as shown in Figure 8–91.

Figure 8–91 JMS Resources



Click the Lock & Edit button to modify, add or delete items in this domain.

JMS system resources are configured and stored as modules similar to standard J2EE modules. Such resour distributed topics, foreign servers, and JMS store-and-forward (SAF) parameters. You can administratively

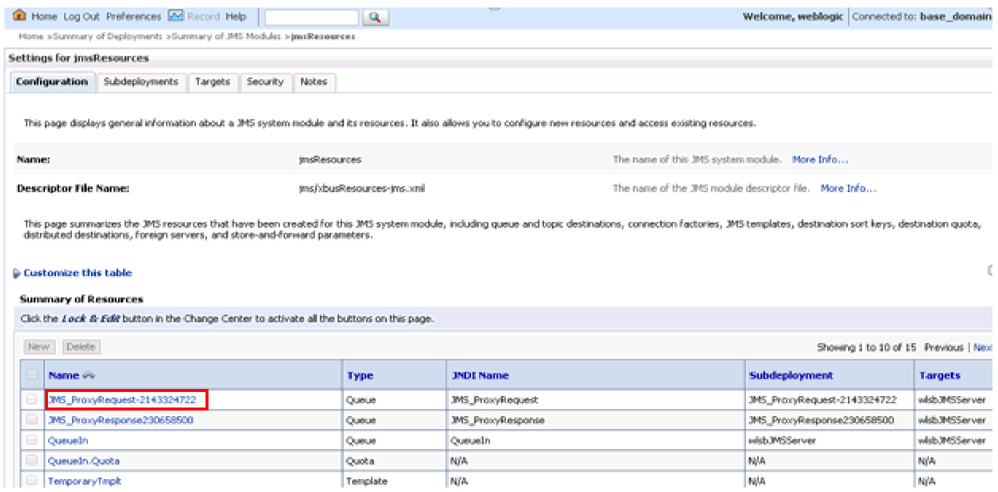
This page summarizes the JMS system modules that have been created for this domain.

Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

- Configure JMS system modules
- Configure resources for JMS system modules

11. Click the appropriate request link (for example, JMS_ProxyRequest) as shown in Figure 8–92.

Figure 8–92 JMS_ProxyRequest Link


Home Log Out Preferences Record Help
Welcome, weblogic | Connected to: base_domain
Home >Summary of Deployments >Summary of JMS Modules >jmsResources
Configuration
Subdeployments
Targets
Security
Notes
Customize this table
New
Delete
Showing 1 to 10 of 15 Previous | Nex

Settings for jmsResources

This page displays general information about a JMS system module and its resources. It also allows you to configure new resources and access existing resources.

Name:	jmsResources	The name of this JMS system module. More Info...
Descriptor File Name:	jms/xbusResources-jms.xml	The name of the JMS module descriptor file. More Info...

This page summarizes the JMS resources that have been created for this JMS system module, including queue and topic destinations, connection factories, JMS templates, destination sort keys, destination quota, distributed destinations, foreign servers, and store-and-forward parameters.

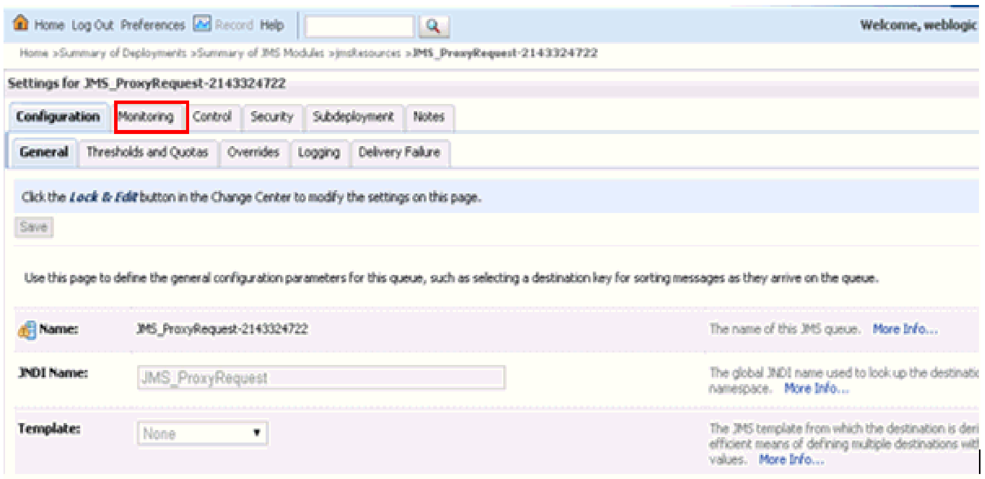
Summary of Resources

Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

Name	Type	JNDI Name	Subdeployment	Targets
JMS_ProxyRequest-2143324722	Queue	JMS_ProxyRequest	JMS_ProxyRequest-2143324722	wlsbJMSServer
JMS_ProxyResponse230658500	Queue	JMS_ProxyResponse	JMS_ProxyResponse230658500	wlsbJMSServer
QueueIn	Queue	QueueIn	wlsbJMSServer	wlsbJMSServer
QueueIn.Quota	Quota	N/A	N/A	N/A
TemporaryTmplt	Template	N/A	N/A	N/A

12. Click the Monitoring tab, as shown in Figure 8–93.

Figure 8–93 Monitoring Tab



Settings for JMS_ProxyRequest-2143324722

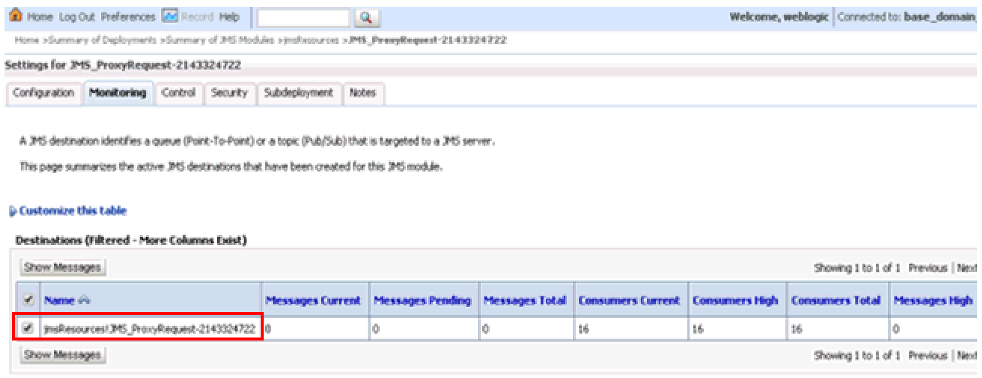
Click the *Lock & Edit* button in the Change Center to modify the settings on this page.

Use this page to define the general configuration parameters for this queue, such as selecting a destination key for sorting messages as they arrive on the queue.

Name:	JMS_ProxyRequest-2143324722	The name of this JMS queue. More Info...
JNDI Name:	JMS_ProxyRequest	The global JNDI name used to look up the destinati namespace. More Info...
Template:	None	The JMS template from which the destination is deri efficient means of defining multiple destinations wit values. More Info...

13. Select the check box and click the **Show Messages** button, as shown in Figure 8–94.

Figure 8–94 Show Messages Button


Home Log Out Preferences Record Help
Welcome, weblogic | Connected to: base_domain
Home >Summary of Deployments >Summary of JMS Modules >jmsResources >JMS_ProxyRequest-2143324722
Configuration
Monitoring
Control
Security
Subdeployment
Notes
Customize this table
Show Messages
Showing 1 to 1 of 1 Previous | Next

Settings for JMS_ProxyRequest-2143324722

A JMS destination identifies a queue (Point-To-Point) or a topic (Pub/Sub) that is targeted to a JMS server.

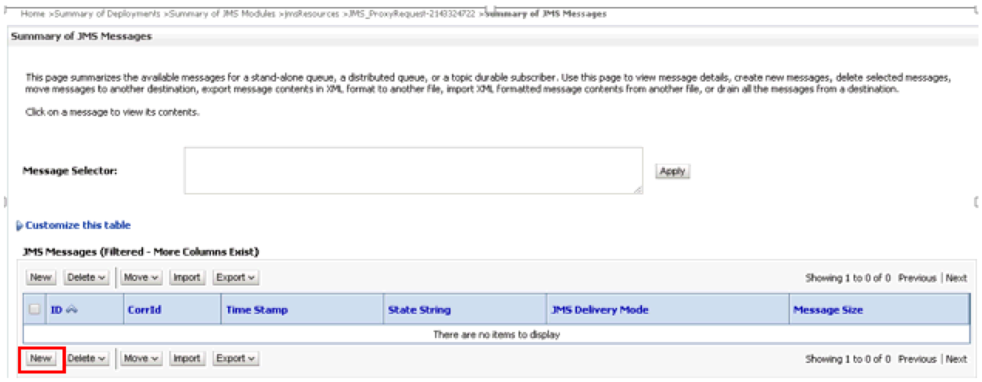
This page summarizes the active JMS destinations that have been created for this JMS module.

Destinations (Filtered - More Columns Exist)

✓	Name	Messages Current	Messages Pending	Messages Total	Consumers Current	Consumers High	Consumers Total	Messages High
✓	jmsResources!JMS_ProxyRequest-2143324722	0	0	0	16	16	16	0

14. Click **New**, as shown in Figure 8–95.

Figure 8–95 JMS Messages



Home >Summary of Deployments >Summary of JMS Modules >jmsResources >JMS_ProxyRequest-2143324722 >**Summary of JMS Messages**

Summary of JMS Messages

This page summarizes the available messages for a stand-alone queue, a distributed queue, or a topic durable subscriber. Use this page to view message details, create new messages, delete selected messages, move messages to another destination, export message contents in XML format to another file, import XML formatted message contents from another file, or drain all the messages from a destination.

Click on a message to view its contents.

Message Selector:

JMS Messages (Filtered - More Columns Exist)

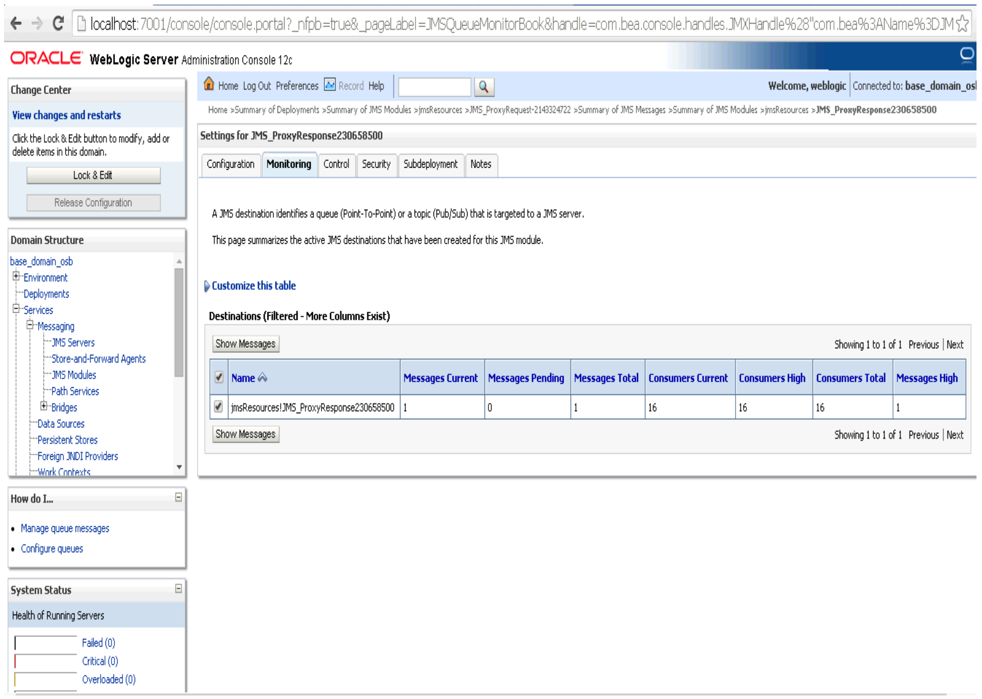
Showing 1 to 0 of 0 Previous | Next

	ID	CorrId	Time Stamp	State String	JMS Delivery Mode	Message Size
There are no items to display						

Showing 1 to 0 of 0 Previous | Next

15. Provide the input payload in the Body field and click **OK**.
16. In the Oracle WLS console, expand **Services**, click **Messaging**, select **JMS Modules**, and then click **jmsResources**.
17. Click the appropriate response link (for example, JMS_ProxyResponse).
18. Click the Monitoring tab.
19. Select the check box and click **Show Messages**, as shown in Figure 8–96.

Figure 8–96 Destination Messages



localhost:7001/console/console.portal?_nfpb=true&_pageLabel=JMSQueueMonitorBook&handle=com.bea.console.handles.JMXHandle%28"com.bea%3AName%3DJM

Welcome, weblogic | Connected to: base_domain_osb

Home >Summary of Deployments >Summary of JMS Modules >jmsResources >JMS_ProxyRequest-2143324722 >Summary of JMS Messages >Summary of JMS Modules >jmsResources >**JMS_ProxyResponse230658500**

Settings for JMS_ProxyResponse230658500

A JMS destination identifies a queue (Point-To-Point) or a topic (Pub/Sub) that is targeted to a JMS server.

This page summarizes the active JMS destinations that have been created for this JMS module.

Destinations (Filtered - More Columns Exist)

Showing 1 to 1 of 1 Previous | Next

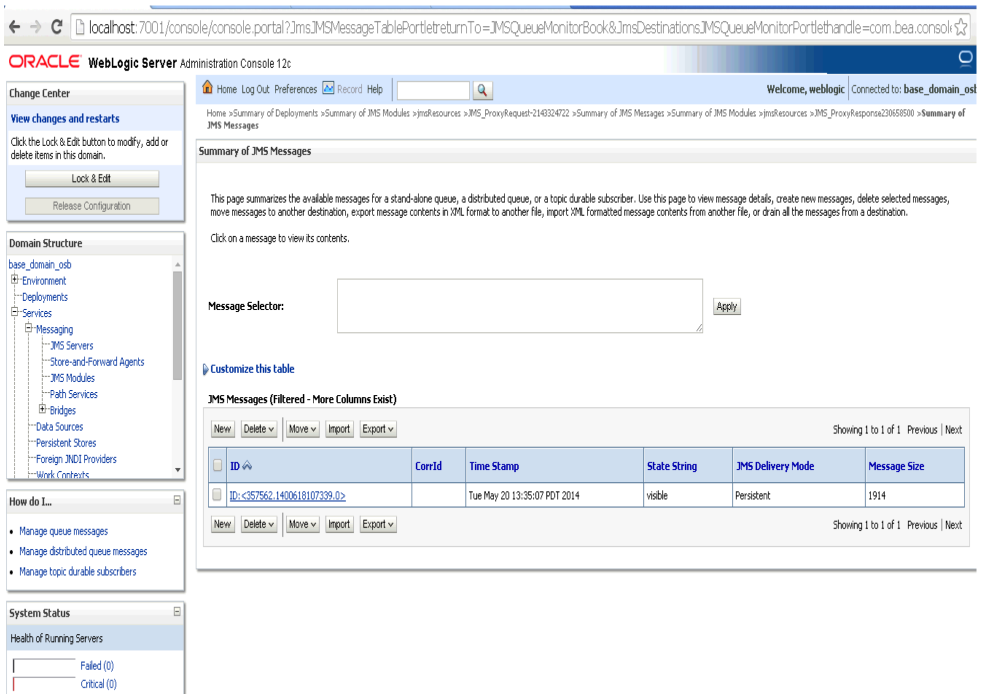
✓	Name	Messages Current	Messages Pending	Messages Total	Consumers Current	Consumers High	Consumers Total	Messages High
✓	jmsResources!JMS_ProxyResponse230658500	1	0	1	16	16	16	1

Showing 1 to 1 of 1 Previous | Next

Click the Lock & Edit button to modify, add or delete items in this domain.

20. Click the ID link with the appropriate time and date, as shown in Figure 8–97.

Figure 8–97 Summary of JMS Messages Window



localhost:7001/console/console.portal?JmsJMSMessageTablePortletreturnTo=JMSQueueMonitorBook&JmsDestinationsJMSQueueMonitorPortlethandle=com.bea.consol

Home >Summary of Deployments >Summary of JMS Modules >jmsResources >JMS_ProxyRequest-2143324722 >Summary of JMS Messages >Summary of JMS Modules >jmsResources >JMS_ProxyResponse230658500 >**Summary of JMS Messages**

Summary of JMS Messages

This page summarizes the available messages for a stand-alone queue, a distributed queue, or a topic durable subscriber. Use this page to view message details, create new messages, delete selected messages, move messages to another destination, export message contents in XML format to another file, import XML formatted message contents from another file, or drain all the messages from a destination.

Click on a message to view its contents.

Message Selector:

JMS Messages (Filtered - More Columns Exist)

Showing 1 to 1 of 1 Previous | Next

	ID	CorrId	Time Stamp	State String	JMS Delivery Mode	Message Size
☐	ID:<357562.1400618107339.0>		Tue May 20 13:35:07 PDT 2014	visible	Persistent	1914

Showing 1 to 1 of 1 Previous | Next

ORACLE WebLogic Server Administration Console 12c
Home Log Out Preferences Record Help
Welcome, weblogic Connected to: base_domain_osb
Change Center
View changes and restarts
Click the Lock & Edit button to modify, add or delete items in this domain.
Lock & Edit
Release Configuration
Domain Structure
base_domain_osb
Environment
Deployments
Services
Messaging
JMS Servers
Store-and-Forward Agents
JMS Modules
Path Services
Bridges
Data Sources
Persistent Stores
Foreign JNDI Providers
Work Contexts
How do I...
Manage queue messages
Manage distributed queue messages
Manage topic durable subscribers
System Status
Health of Running Servers
Failed (0)
Critical (0)
Apply
Customize this table
New Delete Move Import Export

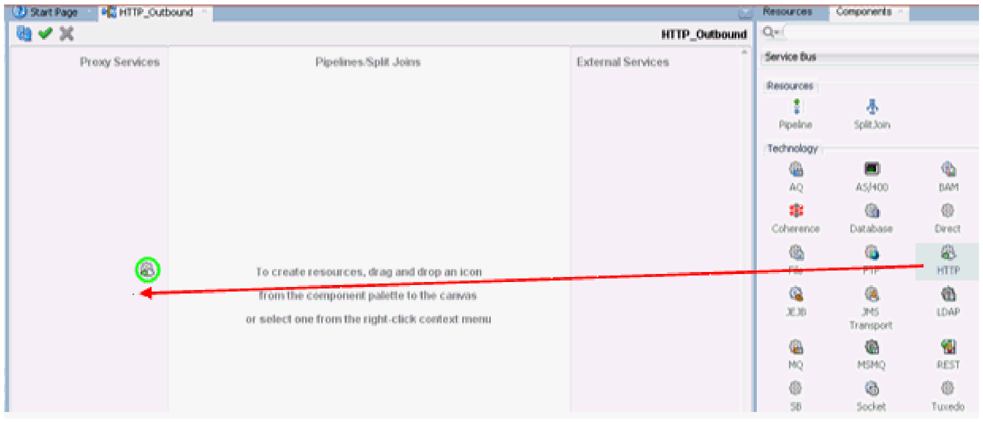
The response document is shown under the Text field.

8.6 Configuring an HTTP Outbound Process Using JDeveloper (J2CA Configuration)

This section describes how to configure HTTP Outbound process to your J.D. Edwards OneWorld system, using Oracle JDeveloper for J2CA configurations.

1. Before you design an HTTP Outbound process, you must generate the respective WSDL file using Application Explorer. For more information, see Section 4.4.1, "Generating WSDL for Request/Response Service" on page 4-8.
2. Start the Oracle JDeveloper and create a Service Bus Application for OSB. For more information, see Section 8.1.1, "Creating a Service Bus Application for OSB" on page 8-2.
3. Create a Third Party Adapter Service Component. For more information, see Section 8.1.2.1, "Configuring a Third-Party Adapter Service Component" on page 8-3.
4. Create an HTTP Proxy Service with a Pipeline and perform the following steps:
   a. Drag and drop the **HTTP** component from the Technology Components pane to the Proxy Services pane, as shown in Figure 8–98.

Figure 8–98 HTTP Component



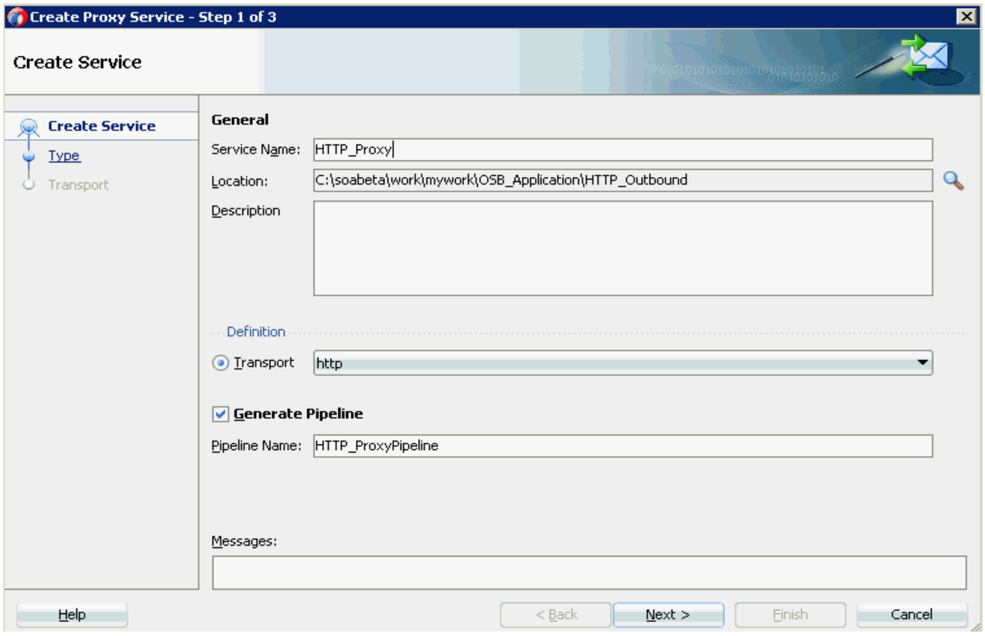
To create resources, drag and drop an icon from the component palette to the canvas or select one from the right-click context menu

The Create Proxy Service dialog is displayed.

b. In the Service Name field, enter any name you wish for the Proxy service (for example, HTTP_Proxy). By default, Generate Pipeline is selected.

c. Click **Next**, as shown in Figure 8–99.

Figure 8–99 Create Proxy Service Pane



General

Service Name: HTTP_Proxy

Location: C:\soabeta\work\mywork\OSB_Application\HTTP_Outbound

Description

Definition

(●) Transport http

[✓] Generate Pipeline

Pipeline Name: HTTP_ProxyPipeline

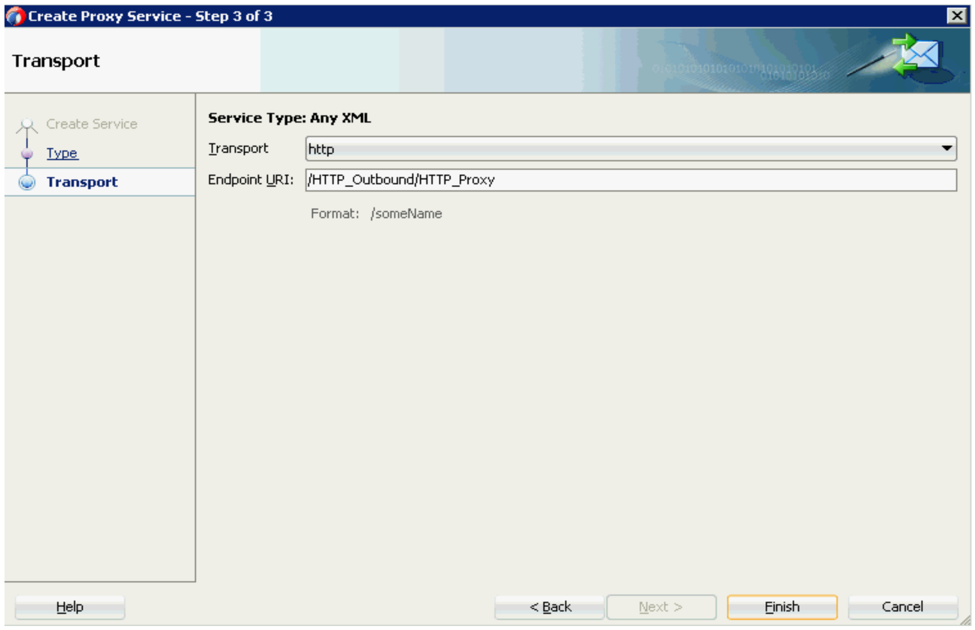
Messages:

d. In the displayed Type window, select **Any XML** and then click **Next**.

The Transport window is displayed.

e. Leave the default values and then click **Finish**, as shown in Figure 8–100.

Figure 8–100 Transport Window



Service Type: Any XML

Transport: http

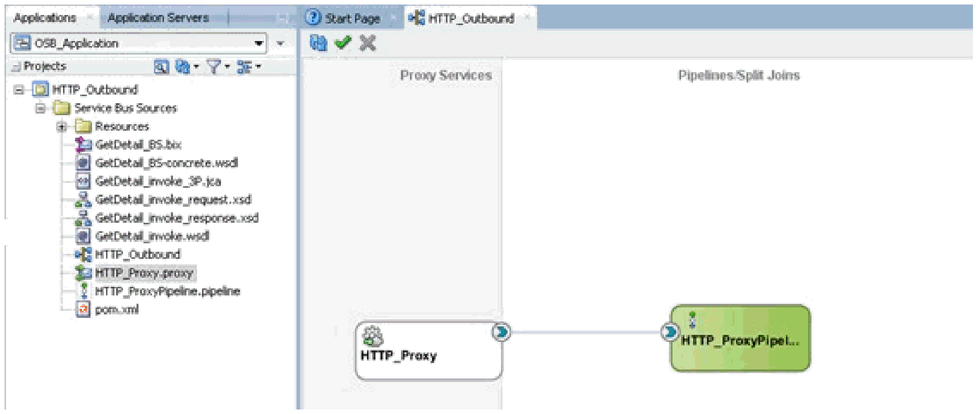
Endpoint URI: /HTTP_Outbound/HTTP_Proxy

Format: /someName

The HTTP Proxy service along with the pipeline is created and displayed.

f. Double-click the created pipeline (for example, HTTP_ProxyPipeline) in the Pipelines/Split Joins pane, as shown in Figure 8–101.

Figure 8–101 Proxy Service

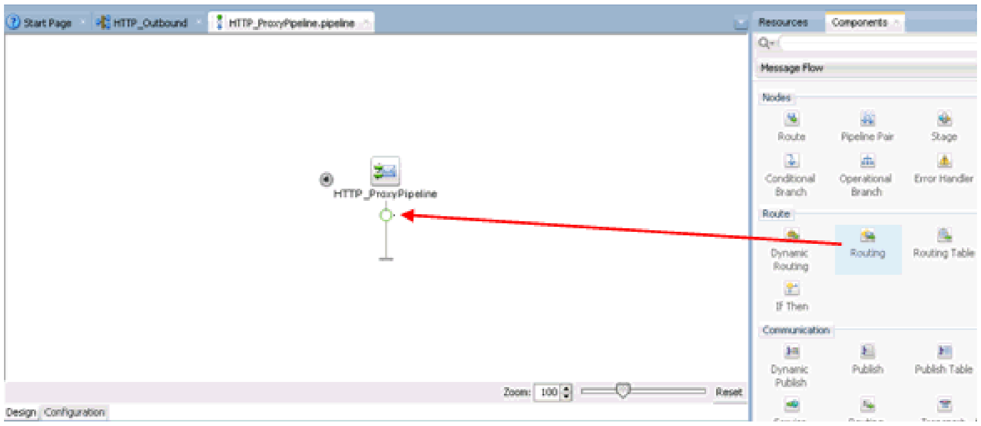


The Pipeline Configuration page is displayed.

5. Configure the Routing Rules and proceed with the following steps:

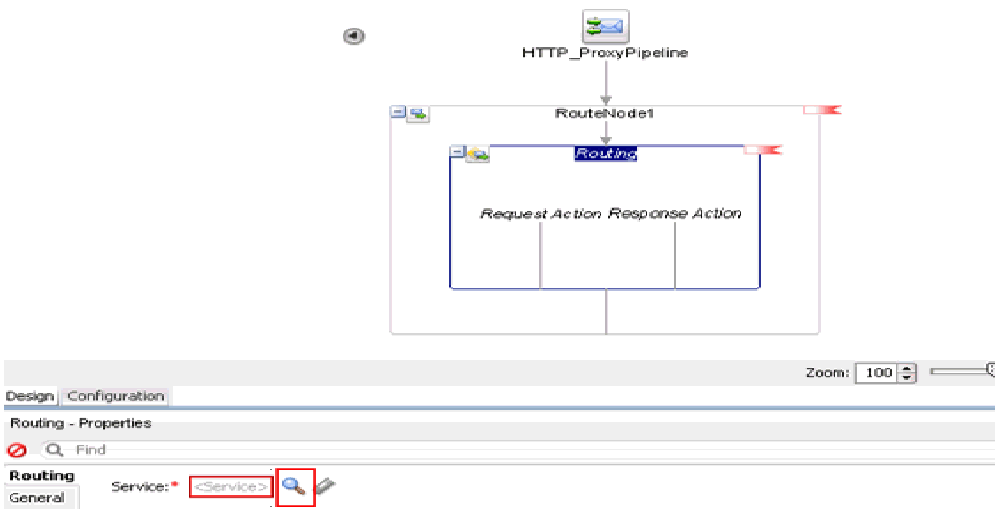
a. Drag and drop the **Routing** component from the Route section to the area below the Pipeline (for example, HTTP_ProxyPipeline), as shown in Figure 8–102.

Figure 8–102 Routing Component



b. In the Pipeline Configuration page, select **Routing** and click the browse icon to the right of the Service field in the Routing Properties pane, as shown in Figure 8–103.

Figure 8–103 Browse Service



c. In the displayed Resource Chooser window, select the WSDL-based Business service (for example, xxxxx_BS.bix) and click **OK**.

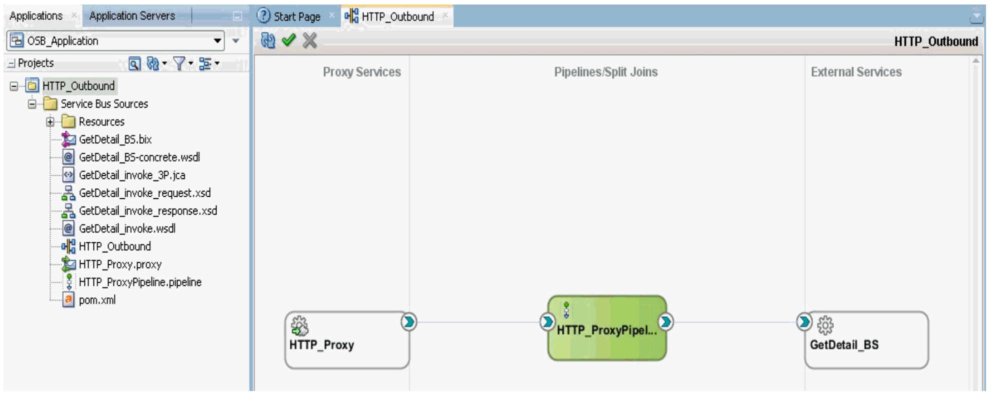
You are returned to the Pipeline configuration page.

d. Save and Close the Pipeline configuration page.

You are returned to the composite editor window.

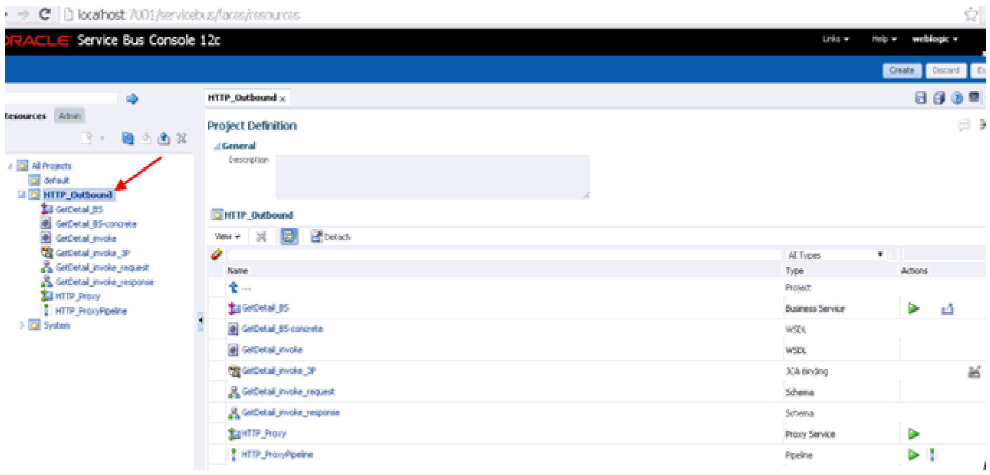
e. Click **Save All** in the menu bar to save the OSB HTTP process, as shown in Figure 8–104.

Figure 8–104 Transport Window



6. Deploy the OSB HTTP outbound process. For more information, see Section 8.1.3, "Deploying the OSB Outbound Process" on page 8-15.
7. Once the process is deployed successfully, log on to the Service Bus Console.
8. In the Service Bus console, click on the deployed HTTP Outbound project (for example, HTTP_Outbound), as shown in Figure 8–105.

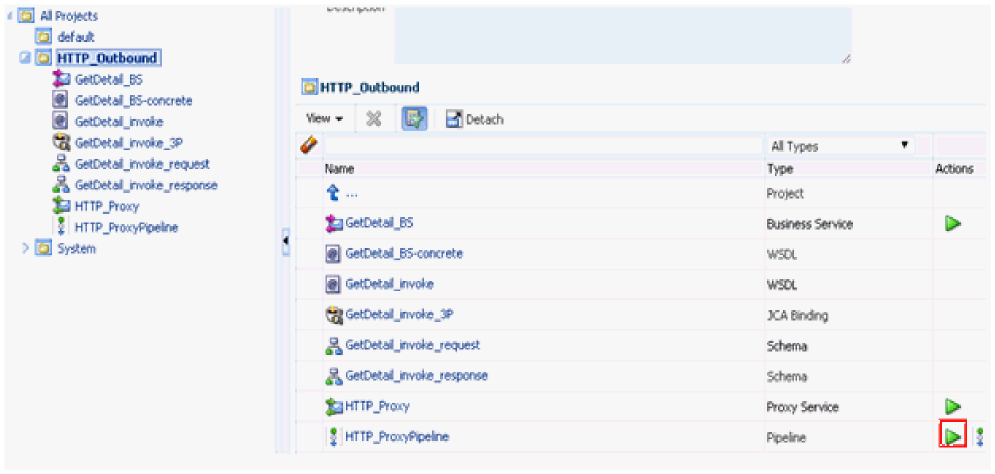
Figure 8–105 Service Bus Console



Name	Type	Actions
...	Project	
GetDetail_BS	Business Service	
GetDetail_BS-concrete	WSDL	
GetDetail_invoke	WSDL	
GetDetail_invoke_3P	JCA Binding	
GetDetail_invoke_request	Schema	
GetDetail_invoke_response	Schema	
HTTP_Proxy	Proxy Service	
HTTP_ProxyPipeline	Pipeline	

9. Click on the Test OSB Console icon for the created pipeline, as shown in Figure 8–106.

Figure 8–106 Test OSB Console Icon



Name	Type	Actions
...	Project	
GetDetail_BS	Business Service	▶
GetDetail_BS-concrete	WSDL	
GetDetail_invoke	WSDL	
GetDetail_invoke_3P	JCA Binding	
GetDetail_invoke_request	Schema	
GetDetail_invoke_response	Schema	
HTTP_Proxy	Proxy Service	▶
HTTP_ProxyPipeline	Pipeline	▶

10. In the displayed Test OSB Console page, provide the input XML and click the **Execute** button.

    In the displayed Test OSB Console page, the response is received.

9 Key Features

This chapter describes key features for the Oracle Application Adapter for J.D. Edwards OneWorld. This chapter contains the following topics:

- Section 9.1, "Configuring the Logging Feature"
- Section 9.2, "Configuring the Diagnosibility Feature"
- Section 9.3, "Configuring the SOA Debugging Feature"
- Section 9.4, "Exception Filter"
- Section 9.5, "Credential Mapping for Oracle SOA Suite (BPEL, Mediator, or BPM)"
- Section 9.6, "Credential Mapping for Oracle Service Bus (OSB)"

9.1 Configuring the Logging Feature

In Oracle 12*c* (12.2.1.0.0), J2CA and BSE adapter logs will be updated in Oracle logs in the *{server-name}*-diagnostic.log file available in the following location:

```
<ORACLE_HOME>\user_projects\domains\base_domain\servers\<server_Name>\logs.
```

Note: The Application Explorer log files for J2CA would be created under the <ADAPTER_HOME>\config\xxxxxxx\log folder where *xxxxxxx* is the name of the J2CA configuration that was created in Application Explorer. Each J2CA configuration in Application Explorer has a corresponding log folder under the named J2CA configuration folder.

This section describes how to configure the Logging feature. It contains the following topics:

- Section 9.1.1, "Configuring Log File Management for the J2CA Connector Application"
- Section 9.1.2, "Configuring Log File Management for Business Services Engine (BSE)"

9.1.1 Configuring Log File Management for the J2CA Connector Application

Log file management for the J2CA Connector Application is governed by the Loggers defined in:

```
<ORACLE_HOME>\user_projects\domains\base_
```

```
domain\config\fmwconfig\servers\${server-name}\logging.xml
```

Any new loggers will have to be added to this file if they are to be managed from the em console.

For example:

```
<logger name='oracle.soa.adapter.iwaf' level='NOTIFICATION:1'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.classloader' level='NOTIFICATION:1'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.connection' useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.connection.IAEAdapter'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.connection.Sample'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.connection.JDEdwards'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.inbound' useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.inbound.IAEAdapter'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.inbound.Sample' useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.inbound.JDEdwards'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.outbound' useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.outbound.IAEAdapter'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.outbound.Sample' useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.outbound.JDEdwards'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.transaction' useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.transaction.IAEAdapter'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.transaction.Sample'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.transaction.JDEdwards'
useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.IAEAdapter' useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.Sample' useParentHandlers='true'/>
<logger name='oracle.soa.adapter.iwaf.JDEdwards' useParentHandlers='true'/>
```

This sets the logging level of all the loggers under oracle.soa.adapter.iwaf to NOTIFICATION:1 (INFO), which is the default setting level by Oracle.

The logging level of all the loggers can also be configured from the em console with the following steps:

1. Start the Oracle WebLogic Server for the Oracle WebLogic Server domain that you configured.
2. Open the Oracle WebLogic Server Enterprise Manager Console in a web browser by entering the following URL:

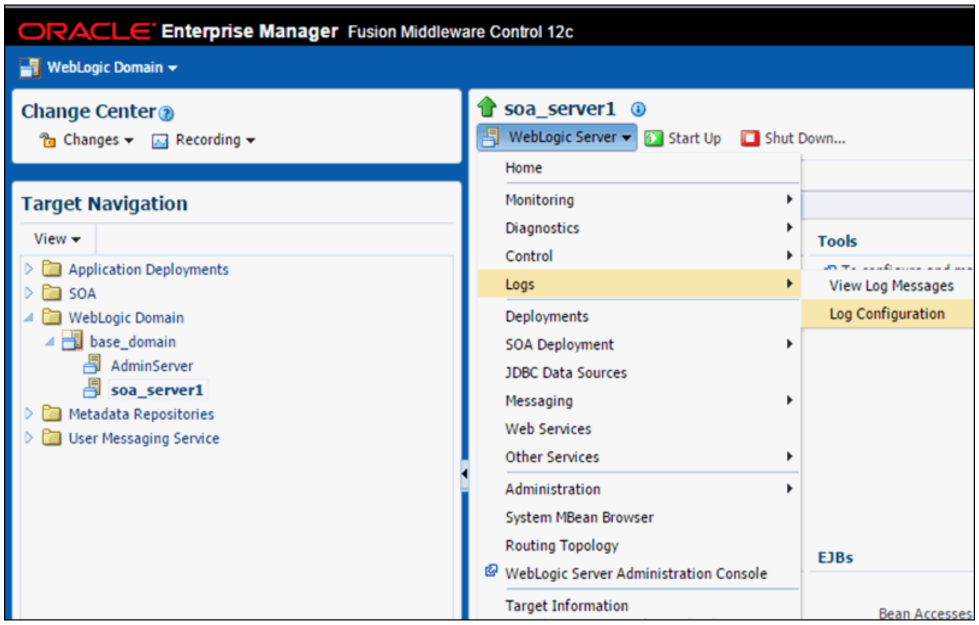
   http://*host name*:*port*/em

   where `host name` is the name of the system where Oracle WebLogic Server is running and `port` is the port for the Oracle WebLogic Server that is running. The default port for the Oracle WebLogic Server is 7001. However, this value can vary between installations.
3. Log in to the Oracle WebLogic Server Administrative Console using an account that has administrator privileges.

4. Under the Target Navigation pane, click **Weblogic Domain**, select **Domain Created**, and click the appropriate server (Managed Server or Integrated Server).

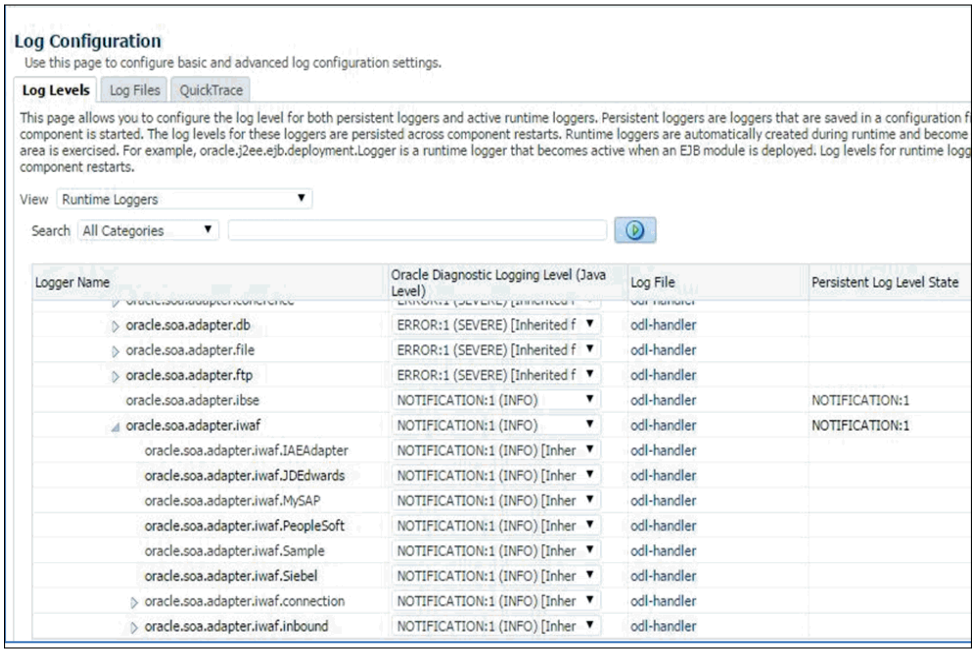
5. In the Server pane, expand WebLogic Server, select **Logs**, and then click **Log Configuration**, as shown in Figure 9–1.

Figure 9–1 Log Configuration Option



6. Under the **Log Levels** tab, expand the Oracle root logger until oracle.soa.adapter.iwaf is visible, as shown in Figure 9–2.

Figure 9–2 Log Levels Tab



Log Configuration

Use this page to configure basic and advanced log configuration settings.

Log Levels | Log Files | QuickTrace

This page allows you to configure the log level for both persistent loggers and active runtime loggers. Persistent loggers are loggers that are saved in a configuration f
component is started. The log levels for these loggers are persisted across component restarts. Runtime loggers are automatically created during runtime and become
area is exercised. For example, oracle.j2ee.ejb.deployment.Logger is a runtime logger that becomes active when an EJB module is deployed. Log levels for runtime logg
component restarts.

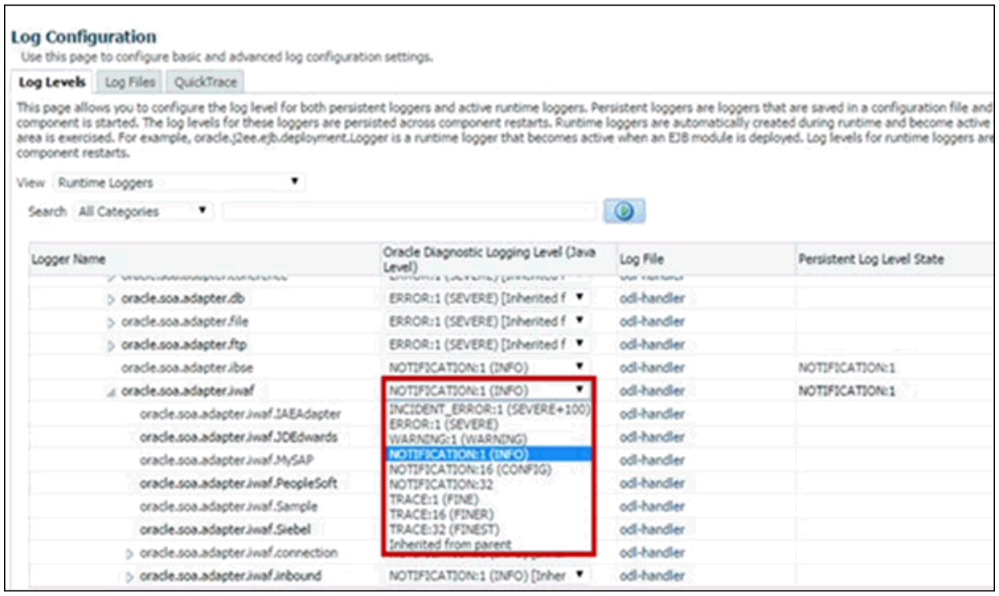
View: Runtime Loggers

Search: All Categories

Logger Name	Oracle Diagnostic Logging Level (Java Level)	Log File	Persistent Log Level State
oracle.soa.adapter.db	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.file	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ftp	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ibse	NOTIFICATION:1 (INFO)	odl-handler	NOTIFICATION:1
oracle.soa.adapter.iwaf	NOTIFICATION:1 (INFO)	odl-handler	NOTIFICATION:1
oracle.soa.adapter.iwaf.IAEAdapter	NOTIFICATION:1 (INFO) [Inher	odl-handler	
oracle.soa.adapter.iwaf.JDEdwards	NOTIFICATION:1 (INFO) [Inher	odl-handler	
oracle.soa.adapter.iwaf.MySAP	NOTIFICATION:1 (INFO) [Inher	odl-handler	
oracle.soa.adapter.iwaf.PeopleSoft	NOTIFICATION:1 (INFO) [Inher	odl-handler	
oracle.soa.adapter.iwaf.Sample	NOTIFICATION:1 (INFO) [Inher	odl-handler	
oracle.soa.adapter.iwaf.Siebel	NOTIFICATION:1 (INFO) [Inher	odl-handler	
oracle.soa.adapter.iwaf.connection	NOTIFICATION:1 (INFO) [Inher	odl-handler	
oracle.soa.adapter.iwaf.inbound	NOTIFICATION:1 (INFO) [Inher	odl-handler	

7. In the Oracle Diagnostic Logging Level (Java Level) column, select the required log level from the oracle.soa.adapter.iwaf drop-down list, as shown in Figure 9–3.

Figure 9–3 Oracle Diagnostic Logging Level (Java Level) Column



Log Configuration

Use this page to configure basic and advanced log configuration settings.

This page allows you to configure the log level for both persistent loggers and active runtime loggers. Persistent loggers are loggers that are saved in a configuration file and component is started. The log levels for these loggers are persisted across component restarts. Runtime loggers are automatically created during runtime and become active area is exercised. For example, oracle.j2ee.ejb.deployment.Logger is a runtime logger that becomes active when an EJB module is deployed. Log levels for runtime loggers are component restarts.

View: Runtime Loggers

Search: All Categories

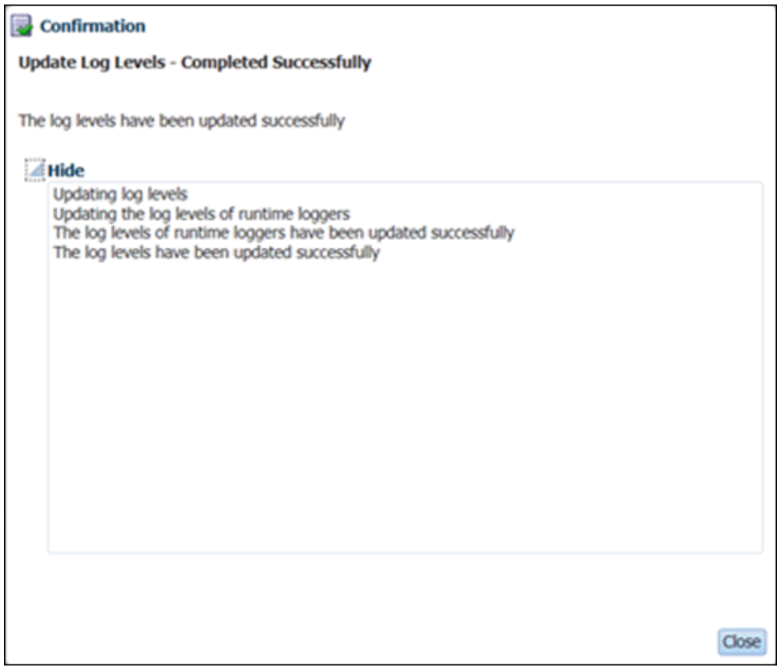
Logger Name	Oracle Diagnostic Logging Level (Java Level)	Log File	Persistent Log Level State
oracle.soa.adapter.db	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.file	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ftp	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ibse	NOTIFICATION:1 (INFO)	odl-handler	NOTIFICATION:1
oracle.soa.adapter.iwaf	NOTIFICATION:1 (INFO)	odl-handler	NOTIFICATION:1
oracle.soa.adapter.iwaf.IAEAdapter		odl-handler	
oracle.soa.adapter.iwaf.JDEdwards		odl-handler	
oracle.soa.adapter.iwaf.MySAP		odl-handler	
oracle.soa.adapter.iwaf.PeopleSoft		odl-handler	
oracle.soa.adapter.iwaf.Sample		odl-handler	
oracle.soa.adapter.iwaf.Siebel		odl-handler	
oracle.soa.adapter.iwaf.connection		odl-handler	
oracle.soa.adapter.iwaf.inbound	NOTIFICATION:1 (INFO) [Inher	odl-handler	

```
INCIDENT_ERROR:1 (SEVERE+100)
ERROR:1 (SEVERE)
WARNING:1 (WARNING)
NOTIFICATION:1 (INFO)
NOTIFICATION:16 (CONFIG)
NOTIFICATION:32
TRACE:1 (FINE)
TRACE:16 (FINER)
TRACE:32 (FINEST)
Inherited from parent
```

8. Click **Apply**.

   A confirmation message appears, indicating that the update was completed successfully, as shown in Figure 9–4.

Figure 9–4 Confirmation Message



Confirmation

Update Log Levels - Completed Successfully

The log levels have been updated successfully

```
Updating log levels
Updating the log levels of runtime loggers
The log levels of runtime loggers have been updated successfully
The log levels have been updated successfully
```

9. Click **Close**.

The following table shows how the Log Level property is updated in the *{server-name}*-diagnostic.log based on the corresponding Log Level property settings in the em console.

Table 9–1 Log Level Property Settings

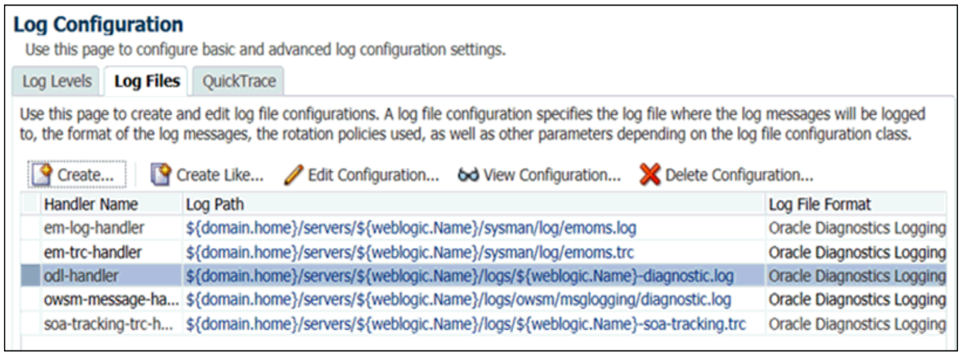
Log Level Set in Em Console	Log Level Updated in {server-name}-diagnostic.log
ERROR:1	ERROR
WARNING:1	WARNING
NOTIFICATION:1	NOTIFICATION
NOTIFICATION:16	NOTIFICATION
NOTIFICATION:32	NOTIFICATION
TRACE:1	NOTIFICATION
TRACE:16	NOTIFICATION
TRACE:32	TRACE:32

Note: Setting the LogLevel as TRACE:32 in the em console, displays the FINEST details in the log (displaying the input passed to the adapter, response received from the adapter and other additional details) with the log level displayed as TRACE:32 in the *{server-name}*-diagnostic.log.

For development and test environments, TRACE:32 is the preferred log level, which displays all of the log details. For production environments, ERROR is the preferred log level.

The log messages are written to a disk file, and the file path can be found in the **Log Files** tab. The Handler Name in the Log Files table corresponds to the Log File name in the Log Levels table. All loggers in the hierarchy below oracle.soa.adapter are currently handled by the odl-handler, as shown in Figure 9–5.

Figure 9–5 Log Configuration Pane



Log Configuration

Use this page to configure basic and advanced log configuration settings.

Use this page to create and edit log file configurations. A log file configuration specifies the log file where the log messages will be logged to, the format of the log messages, the rotation policies used, as well as other parameters depending on the log file configuration class.

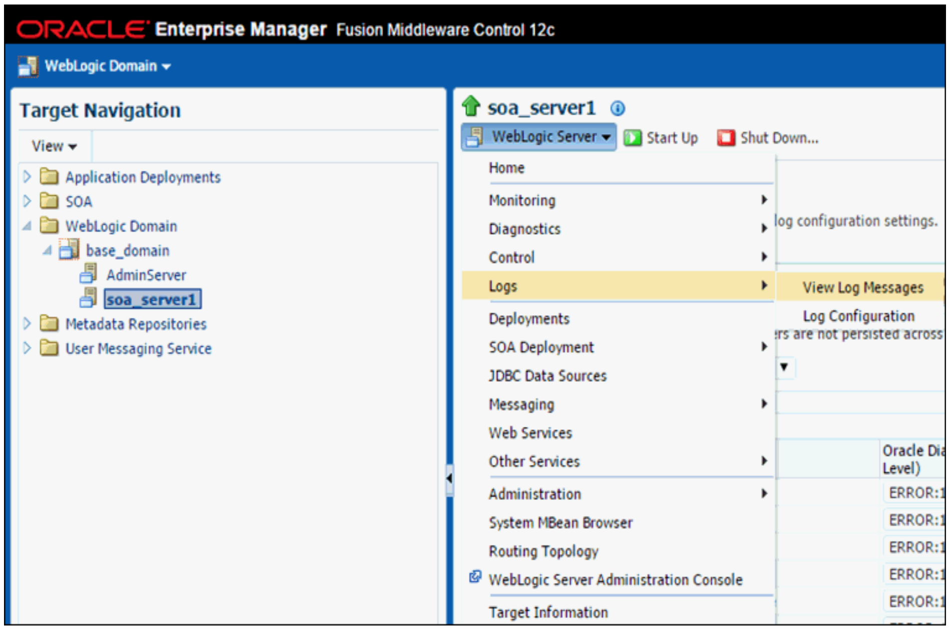
Handler Name	Log Path	Log File Format
em-log-handler	${domain.home}/servers/${weblogic.Name}/sysman/log/emoms.log	Oracle Diagnostics Logging
em-trc-handler	${domain.home}/servers/${weblogic.Name}/sysman/log/emoms.trc	Oracle Diagnostics Logging
odl-handler	${domain.home}/servers/${weblogic.Name}/logs/${weblogic.Name}-diagnostic.log	Oracle Diagnostics Logging
owsm-message-ha...	${domain.home}/servers/${weblogic.Name}/logs/owsm/msglogging/diagnostic.log	Oracle Diagnostics Logging
soa-tracking-trc-h...	${domain.home}/servers/${weblogic.Name}/logs/${weblogic.Name}-soa-tracking.trc	Oracle Diagnostics Logging

The logs are updated in *{server-name}*-diagnostic.log available in the following location:

```
<ORACLE_HOME>\user_projects\domains\base_domain\servers\<server_Name>\logs
```

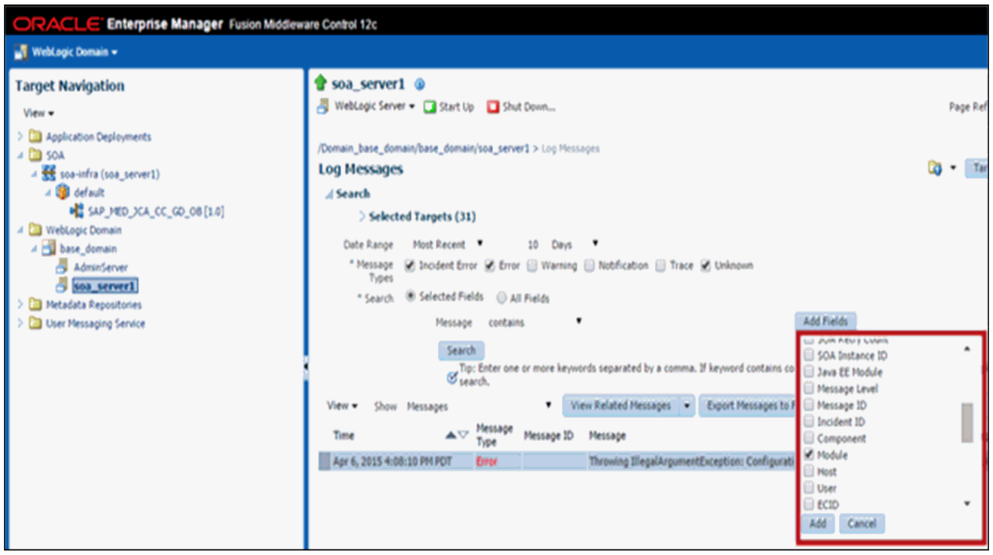
10. In the Server pane, display the log messages in the em console by clicking the **Weblogic Server** drop-down list, selecting **Logs**, and then clicking **View Log Messages**, as shown in Figure 9–6.

Figure 9–6 View Log Messages Option



11. On the Log Messages pane, complete the required search criteria. You can also add the **Module** field to the search criteria, which contains the name of the logger of interest, as shown in Figure 9–7.

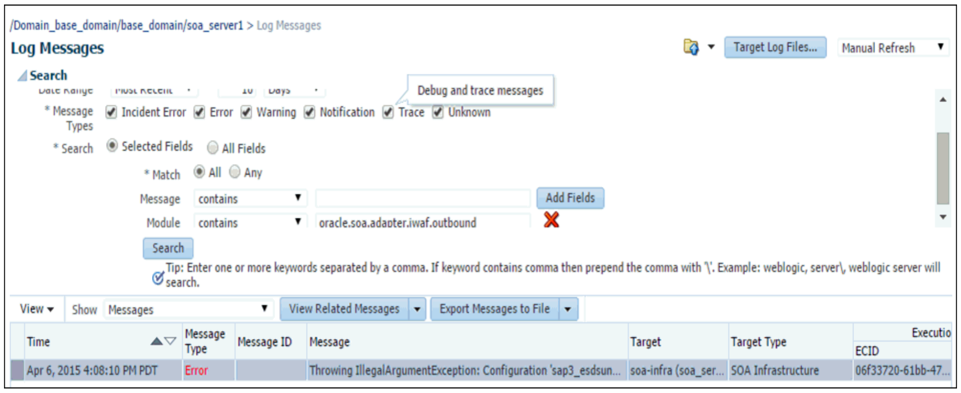
Figure 9–7 Module Field



12. Click **Add**.

13. In the Module field, enter the name of the logger of interest, and if required, select the additional Message Types (Warning, Notification, Trace, and so on) and then click **Search**, as shown in Figure 9–8.

Figure 9–8 Log Messages Pane



/Domain_base_domain/base_domain/soa_server1 > Log Messages

Log Messages

Target Log Files... Manual Refresh

Search

Date Range: Most Recent 10 Days

Debug and trace messages

* Message Types: ✓ Incident Error ✓ Error ✓ Warning ✓ Notification ✓ Trace ✓ Unknown

* Search: [✓ Selected Fields] All Fields

* Match: [✓ All] Any

Message: contains

Module: contains oracle.soa.adapter.iwaf.outbound

Add Fields

Search

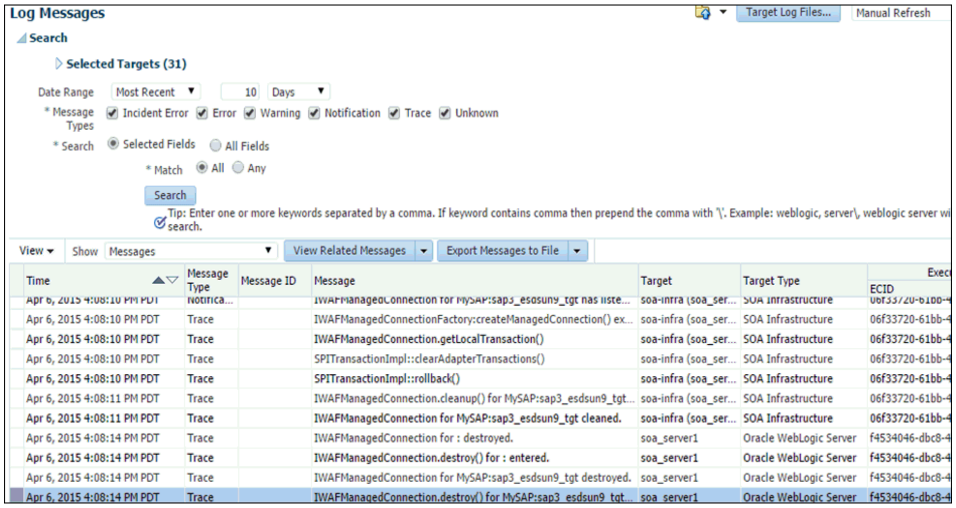
Tip: Enter one or more keywords separated by a comma. If keyword contains comma then prepend the comma with '\'. Example: weblogic, server\, weblogic server will search.

View Show Messages View Related Messages Export Messages to File

Time	Message Type	Message ID	Message	Target	Target Type	Executio ECID
Apr 6, 2015 4:08:10 PM PDT	Error		Throwing IllegalArgumentException: Configuration 'sap3_esdsun...	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-47...

The messages from the specified logger are displayed in the table below the search criteria, as shown in Figure 9–9.

Figure 9–9 Logger Messages



Log Messages

Target Log Files... Manual Refresh

Search

Selected Targets (31)

Date Range: Most Recent 10 Days

* Message Types: ✓ Incident Error ✓ Error ✓ Warning ✓ Notification ✓ Trace ✓ Unknown

* Search: [✓ Selected Fields] All Fields

* Match: [✓ All] Any

Search

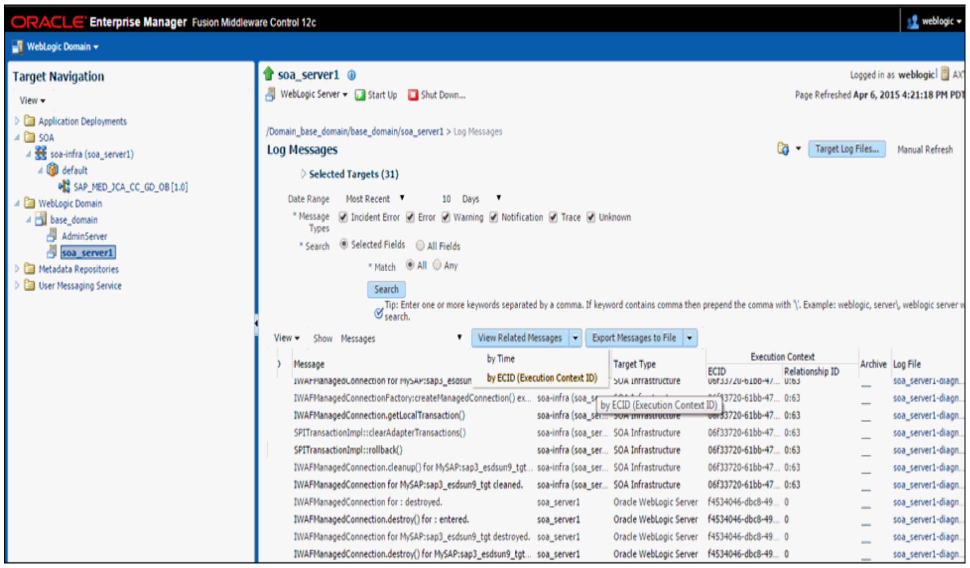
Tip: Enter one or more keywords separated by a comma. If keyword contains comma then prepend the comma with '\'. Example: weblogic, server\, weblogic server wi search.

View Show Messages View Related Messages Export Messages to File

Time	Message Type	Message ID	Message	Target	Target Type	Exec ECID
Apr 6, 2015 4:08:10 PM PDT	Notifica...		IWAFManagedConnection for MySAP:sap3_esdsun9_tgt has liste...	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-4
Apr 6, 2015 4:08:10 PM PDT	Trace		IWAFManagedConnectionFactory:createManagedConnection() ex...	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-4
Apr 6, 2015 4:08:10 PM PDT	Trace		IWAFManagedConnection.getLocalTransaction()	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-4
Apr 6, 2015 4:08:10 PM PDT	Trace		SPITransactionImpl::clearAdapterTransactions()	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-4
Apr 6, 2015 4:08:10 PM PDT	Trace		SPITransactionImpl::rollback()	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-4
Apr 6, 2015 4:08:11 PM PDT	Trace		IWAFManagedConnection.cleanup() for MySAP:sap3_esdsun9_tgt...	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-4
Apr 6, 2015 4:08:11 PM PDT	Trace		IWAFManagedConnection for MySAP:sap3_esdsun9_tgt cleaned.	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-4
Apr 6, 2015 4:08:14 PM PDT	Trace		IWAFManagedConnection for : destroyed.	soa_server1	Oracle WebLogic Server	f4534046-dbc8-4
Apr 6, 2015 4:08:14 PM PDT	Trace		IWAFManagedConnection.destroy() for : entered.	soa_server1	Oracle WebLogic Server	f4534046-dbc8-4
Apr 6, 2015 4:08:14 PM PDT	Trace		IWAFManagedConnection for MySAP:sap3_esdsun9_tgt destroyed.	soa_server1	Oracle WebLogic Server	f4534046-dbc8-4
Apr 6, 2015 4:08:14 PM PDT	Trace		IWAFManagedConnection.destroy() for MySAP:sap3_esdsun9_tgt...	soa_server1	Oracle WebLogic Server	f4534046-dbc8-4

14. Select any row in the table. To get identical details, click the **View Related Messages** drop-down list, and select *ECID* (execution Context ID) as shown in Figure 9–10.

Figure 9–10 ECID Option



Page Refreshed Apr 6, 2015 4:21:18 PM PDT

/Domain_base_domain/base_domain/soa_server1 > Log Messages

Log Messages

Selected Targets (31)

Date Range: Most Recent 10 Days

* Message Types: ✓ Incident Error ✓ Error ✓ Warning ✓ Notification ✓ Trace ✓ Unknown

* Search: ● Selected Fields ○ All Fields

* Match: ● All ○ Any

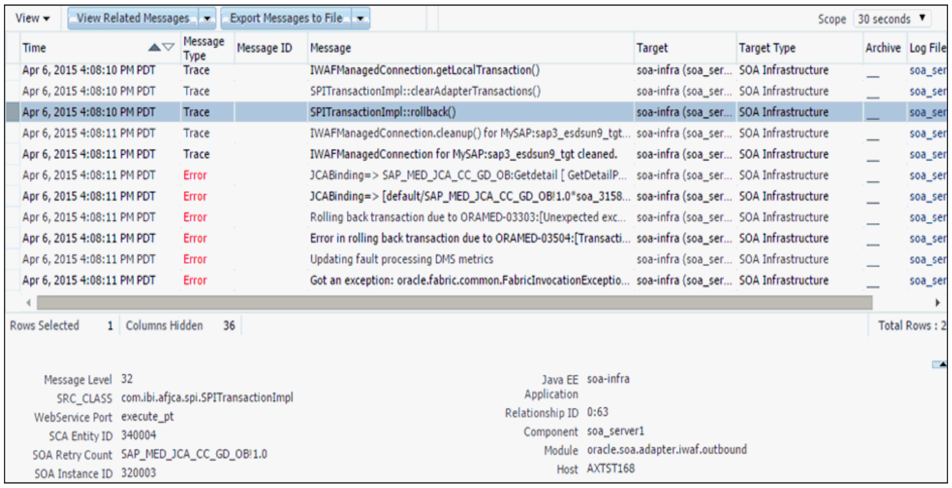
Tip: Enter one or more keywords separated by a comma. If keyword contains comma then prepend the comma with '\'. Example: weblogic, server\, weblogic server search.

Show Messages

Message		Target Type	Execution Context ECID	Execution Context Relationship ID	Archive	Log File
IWAFManagedConnection for MySAP:sap3_esdsun		SOA Infrastructure	06f33720-61bb-47...	0:63	__	soa_server1-diagn...
IWAFManagedConnectionFactory:createManagedConnection() ex...	soa-infra (soa_se		06f33720-61bb-47...	0:63	__	soa_server1-diagn...
IWAFManagedConnection.getLocalTransaction()	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-47...	0:63	__	soa_server1-diagn...
SPITransactionImpl::clearAdapterTransactions()	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-47...	0:63	__	soa_server1-diagn...
SPITransactionImpl::rollback()	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-47...	0:63	__	soa_server1-diagn...
IWAFManagedConnection.cleanup() for MySAP:sap3_esdsun9_tgt...	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-47...	0:63	__	soa_server1-diagn...
IWAFManagedConnection for MySAP:sap3_esdsun9_tgt cleaned.	soa-infra (soa_ser...	SOA Infrastructure	06f33720-61bb-47...	0:63	__	soa_server1-diagn...
IWAFManagedConnection for : destroyed.	soa_server1	Oracle WebLogic Server	f4534046-dbc8-49...	0	__	soa_server1-diagn...
IWAFManagedConnection.destroy() for : entered.	soa_server1	Oracle WebLogic Server	f4534046-dbc8-49...	0	__	soa_server1-diagn...
IWAFManagedConnection for MySAP:sap3_esdsun9_tgt destroyed.	soa_server1	Oracle WebLogic Server	f4534046-dbc8-49...	0	__	soa_server1-diagn...
IWAFManagedConnection.destroy() for MySAP:sap3_esdsun9_tgt...	soa_server1	Oracle WebLogic Server	f4534046-dbc8-49...	0	__	soa_server1-diagn...

Details are displayed, as shown in Figure 9–11.

Figure 9–11 Message Details



Scope 30 seconds

Time	Message Type	Message ID	Message	Target	Target Type	Archive	Log File
Apr 6, 2015 4:08:10 PM PDT	Trace		IWAFManagedConnection.getLocalTransaction()	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:10 PM PDT	Trace		SPITransactionImpl::clearAdapterTransactions()	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:10 PM PDT	Trace		SPITransactionImpl::rollback()	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:11 PM PDT	Trace		IWAFManagedConnection.cleanup() for MySAP:sap3_esdsun9_tgt...	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:11 PM PDT	Trace		IWAFManagedConnection for MySAP:sap3_esdsun9_tgt cleaned.	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:11 PM PDT	Error		JCABinding=> SAP_MED_JCA_CC_GD_OB:Getdetail [GetDetailP...	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:11 PM PDT	Error		JCABinding=> [default/SAP_MED_JCA_CC_GD_OB!1.0*soa_3158...	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:11 PM PDT	Error		Rolling back transaction due to ORAMED-03303:[Unexpected exc...	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:11 PM PDT	Error		Error in rolling back transaction due to ORAMED-03504:[Transacti...	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:11 PM PDT	Error		Updating fault processing DMS metrics	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser
Apr 6, 2015 4:08:11 PM PDT	Error		Got an exception: oracle.fabric.common.FabricInvocationExceptio...	soa-infra (soa_ser...	SOA Infrastructure	__	soa_ser

Rows Selected 1 Columns Hidden 36 Total Rows : 2

Message Level	32	Java EE Application	soa-infra
SRC_CLASS	com.ibi.afjca.spi.SPITransactionImpl	Relationship ID	0:63
WebService Port	execute_pt	Component	soa_server1
SCA Entity ID	340004	Module	oracle.soa.adapter.iwaf.outbound
SOA Retry Count	SAP_MED_JCA_CC_GD_OB!1.0	Host	AXTST168
SOA Instance ID	320003		

9.1.2 Configuring Log File Management for Business Services Engine (BSE)

Similar to J2CA for BSE, the Log file management is governed by the Loggers defined in:

```
<ORACLE_HOME>\user_projects\domains\base_
domain\config\fmwconfig\servers\${server-name}\logging.xml
```

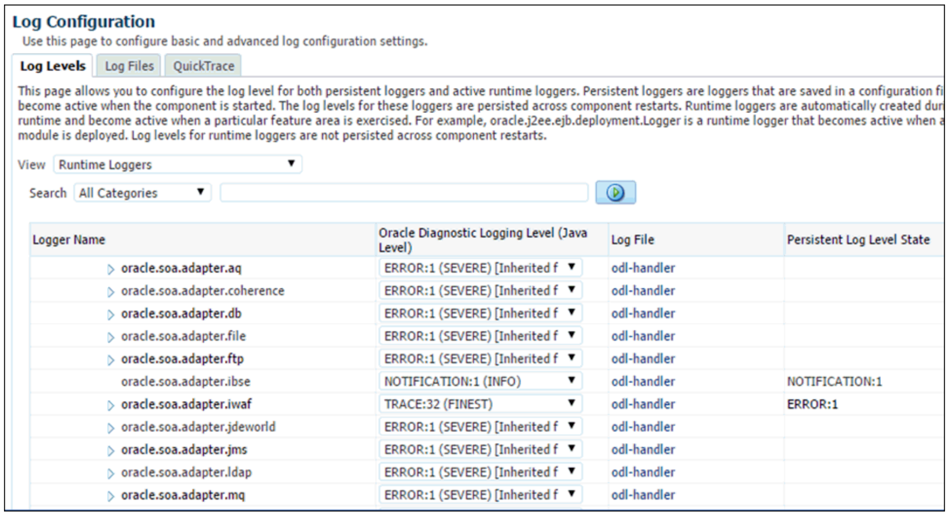
The following syntax sets the logging level of all the loggers under oracle.soa.adapter.ibse to NOTIFICATION:1 (INFO) which is the default setting level by oracle.

```
<logger name='oracle.soa.adapter.ibse' level='NOTIFICATION:1'
useParentHandlers='true'/>
```

The logging level of all the loggers can also be configured from the em console with the following steps:

1. Repeat steps 1 through 5, as described in Section 9.1.1, "Configuring Log File Management for the J2CA Connector Application".
2. Under the **Log Levels** tab, expand the Oracle root logger until the oracle.soa.adapter.ibse Logger name is visible, as shown in Figure 9–12.

Figure 9–12 Log Levels Tab



Log Configuration

Use this page to configure basic and advanced log configuration settings.

This page allows you to configure the log level for both persistent loggers and active runtime loggers. Persistent loggers are loggers that are saved in a configuration fi
become active when the component is started. The log levels for these loggers are persisted across component restarts. Runtime loggers are automatically created dur
runtime and become active when a particular feature area is exercised. For example, oracle.j2ee.ejb.deployment.Logger is a runtime logger that becomes active when a
module is deployed. Log levels for runtime loggers are not persisted across component restarts.

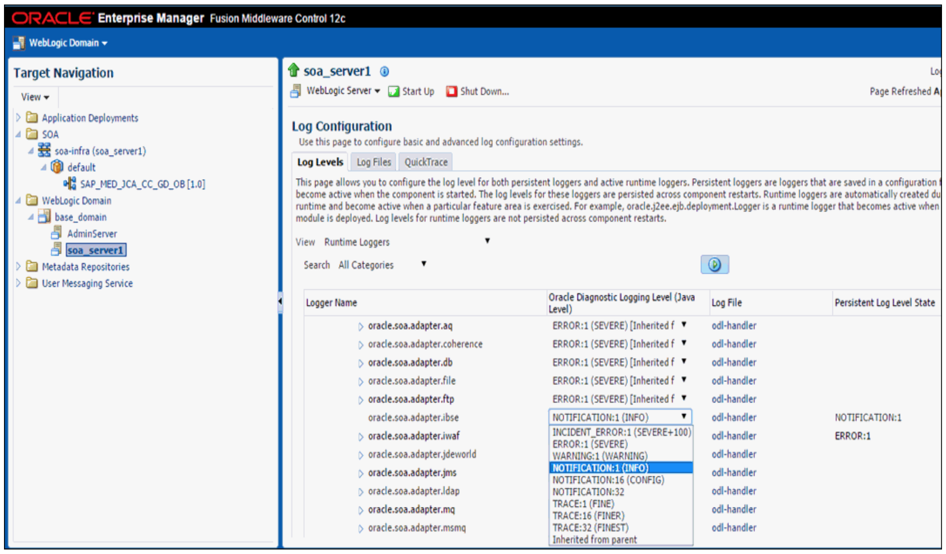
View: Runtime Loggers

Search: All Categories

Logger Name	Oracle Diagnostic Logging Level (Java Level)	Log File	Persistent Log Level State
oracle.soa.adapter.aq	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.coherence	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.db	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.file	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ftp	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ibse	NOTIFICATION:1 (INFO)	odl-handler	NOTIFICATION:1
oracle.soa.adapter.iwaf	TRACE:32 (FINEST)	odl-handler	ERROR:1
oracle.soa.adapter.jdeworld	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.jms	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ldap	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.mq	ERROR:1 (SEVERE) [Inherited f	odl-handler	

3. In the Oracle Diagnostic Logging Level (Java Level) column, select the required log level from the oracle.soa.adapter.ibse drop-down list, as shown in Figure 9–13.

Figure 9–13 Oracle Diagnostic Logging Level Column



Log Configuration

Use this page to configure basic and advanced log configuration settings.

This page allows you to configure the log level for both persistent loggers and active runtime loggers. Persistent loggers are loggers that are saved in a configuration f
become active when the component is started. The log levels for these loggers are persisted across component restarts. Runtime loggers are automatically created du
runtime and become active when a particular feature area is exercised. For example, oracle.j2ee.ejb.deployment.Logger is a runtime logger that becomes active when
module is deployed. Log levels for runtime loggers are not persisted across component restarts.

View: Runtime Loggers

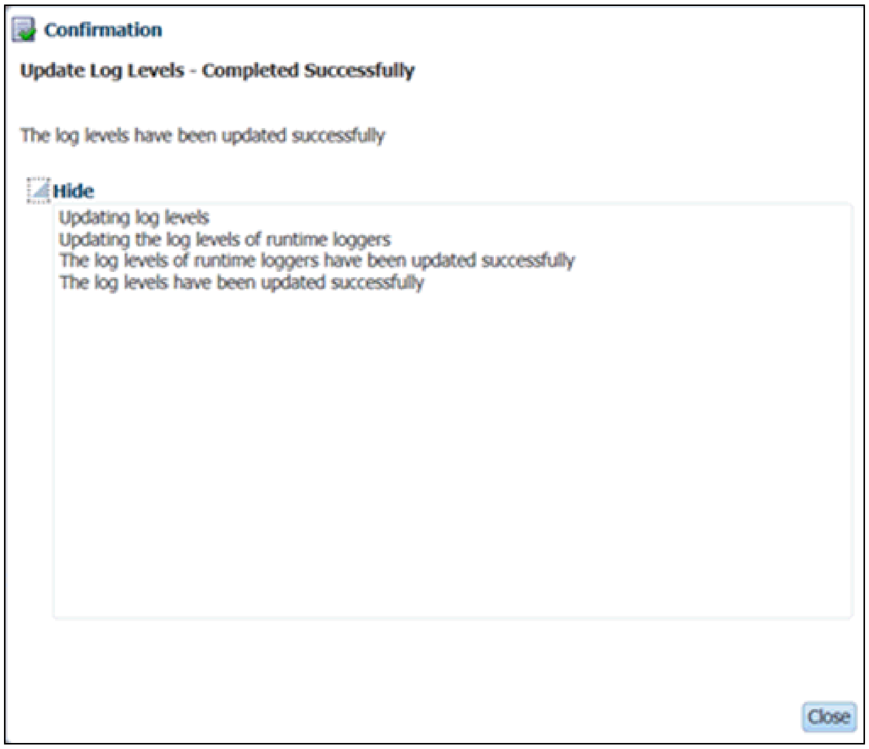
Search: All Categories

Logger Name	Oracle Diagnostic Logging Level (Java Level)	Log File	Persistent Log Level State
oracle.soa.adapter.aq	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.coherence	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.db	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.file	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ftp	ERROR:1 (SEVERE) [Inherited f	odl-handler	
oracle.soa.adapter.ibse	NOTIFICATION:1 (INFO)	odl-handler	NOTIFICATION:1
oracle.soa.adapter.iwaf		odl-handler	ERROR:1
oracle.soa.adapter.jdeworld		odl-handler	
oracle.soa.adapter.jms		odl-handler	
oracle.soa.adapter.ldap		odl-handler	
oracle.soa.adapter.mq		odl-handler	
oracle.soa.adapter.msmq		odl-handler	

4. Click **Apply**.

   A confirmation message appears, indicating that the update was completed successfully, as shown in Figure 9–14.

Figure 9–14 Confirmation Message



Update Log Levels - Completed Successfully

The log levels have been updated successfully

Updating log levels
Updating the log levels of runtime loggers
The log levels of runtime loggers have been updated successfully
The log levels have been updated successfully

5. Click **Close**.

   The following table shows how the Log Level property is updated in the *{server-name}*-diagnostic.log based on the corresponding Log Level property settings in the em console.

Table 9–2 Log Level Property Settings

Log Level Set in Em Console	Log Level Updated in {server-name}-diagnostic.log
ERROR:1	ERROR
WARNING:1	WARNING
NOTIFICATION:1	NOTIFICATION
NOTIFICATION:16	NOTIFICATION
NOTIFICATION:32	NOTIFICATION
TRACE:1	NOTIFICATION
TRACE:16	NOTIFICATION
TRACE:32	TRACE:32

For verification of logs, see steps 11 to 18 of the J2CA Logging feature, found in Section 9.1.1, "Configuring Log File Management for the J2CA Connector Application".

9.2 Configuring the Diagnosibility Feature

This section describes how to configure and use the Diagnosibility feature for the Oracle Fusion Middleware Application Adapters for Oracle WebLogic Server. It contains the following topic:

- Section 9.2.1, "Supporting Protocols"

The Diagnosibility feature captures the endpoint health status (where available) of the adapters, and provides a corresponding alert to the Oracle Adapter Framework, so it may be displayed in the EM console.

Note: The Diagnosibility feature supports only inbound adapter processes

Make sure that there is an inbound process deployed before moving to the next section

This information can be viewed for the adapters as mentioned in the following steps:

1. Start the Oracle WebLogic Servers and open the Oracle WebLogic Server Enterprise Manager Console in a web browser by entering the following URL:

   ```
   http://host name:port/em
   ```

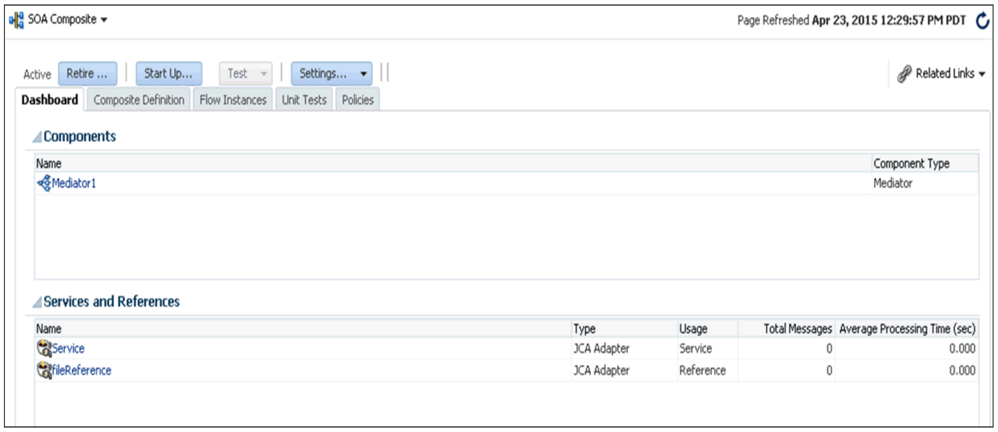
   Where *host name* is the name of the system where Oracle WebLogic Server is running, and *port* is the port for the Oracle WebLogic Server that is running.

2. Log in to the Oracle WebLogic Server Administrative Console using an account that has administrator privileges.

3. On the right pane, expand **SOA**, **soa-Infra (server_name)**, **Default**, and then **Deployed inbound process**.

4. Click the process and then select the service listed in the Services and References Section in the right pane, as shown in Figure 9–15.

Figure 9–15 Services and References Section



Page Refreshed Apr 23, 2015 12:29:57 PM PDT

Components

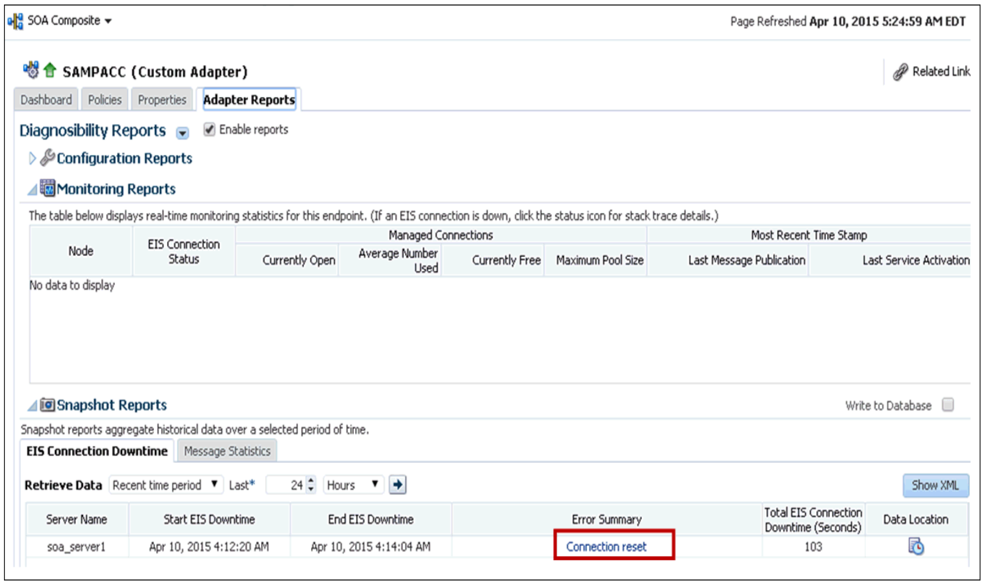
Name	Component Type
Mediator1	Mediator

Services and References

Name	Type	Usage	Total Messages	Average Processing Time (sec)
Service	JCA Adapter	Service	0	0.000
fileReference	JCA Adapter	Reference	0	0.000

5. In the Adapter Reports tab, select the **Enable reports** check box, and then expand **Snapshot Reports** to view the details of the EIS downtime. Clicking on the Error Summary will show the stack trace, as shown in Figure 9–16.

Figure 9–16 Stack Trace



SOA Composite ▾ Page Refreshed **Apr 10, 2015 5:24:59 AM EDT**

SAMPACC (Custom Adapter) Related Link

Dashboard | Policies | Properties | **Adapter Reports**

Diagnosibility Reports ✓ Enable reports

Configuration Reports

Monitoring Reports

The table below displays real-time monitoring statistics for this endpoint. (If an EIS connection is down, click the status icon for stack trace details.)

Node	EIS Connection Status	Managed Connections				Most Recent Time Stamp	
		Currently Open	Average Number Used	Currently Free	Maximum Pool Size	Last Message Publication	Last Service Activation

No data to display

Snapshot Reports Write to Database

Snapshot reports aggregate historical data over a selected period of time.

EIS Connection Downtime | Message Statistics

Retrieve Data Recent time period ▾ Last* 24 Hours ▾ Show XML

Server Name	Start EIS Downtime	End EIS Downtime	Error Summary	Total EIS Connection Downtime (Seconds)	Data Location
soa_server1	Apr 10, 2015 4:12:20 AM	Apr 10, 2015 4:14:04 AM	Connection reset	103	

9.2.1 Supporting Protocols

This section describes the supporting protocols for the Oracle Fusion Middleware Application Adapter for J.D. Edwards OneWorld, and consists of the following topic:

- Section 9.2.1.1, "Oracle Fusion Middleware Application Adapter for J.D. Edwards OneWorld Endpoint Status"

9.2.1.1 Oracle Fusion Middleware Application Adapter for J.D. Edwards OneWorld Endpoint Status

The Oracle Fusion Middleware Application Adapter for J.D. Edwards OneWorld (inbound) can use the following protocols:

- HTTP
- TCP
- File (Not Supported)

The HTTP and TCP listening protocol adapters listen on a socket. As a result, by their nature, they cannot determine whether there is anything live on the other side until they receive something. Even when receiving a request, it is impossible to determine with certainty where the request originated.

For these protocols, the EIS is regarded and used when a request is being received. EIS determines if the communication error happens while the request is being received. However, this is a very unrefined and rudimentary determination.

However, for the Oracle Fusion Middleware Application Adapter for J.D. Edwards OneWorld, after a TCP or HTTP request is received, the J.D. Edwards OneWorld server is contacted for more information. This allows the adapter to get a more accurate status of the possibility of communication with the actual server.

9.3 Configuring the SOA Debugging Feature

This section describes how to configure and use the SOA Debugging feature for the Oracle Application Adapter for J.D. Edwards OneWorld. It contains the following topics:

- Section 9.3.1, "Guidelines for Using the SOA Debugger"
- Section 9.3.2, "Prerequisite"
- Section 9.3.3, "Debugging a BPEL Process in Oracle JDeveloper"
- Section 9.3.4, "Debugging an OSB Process in Oracle JDeveloper"

Note: The SOA Debugging feature is currently supported only for J2CA configurations and it is not applicable for BSE configurations.

For SOA, this feature is explained using a BPEL process. The same is applicable for Mediator and BPM processes.

9.3.1 Guidelines for Using the SOA Debugger

This section describes guidelines for using the SOA Debugger.

1. Only one client at a time can connect to the SOA Debugger.
2. Adapter endpoint errors are not displayed in the SOA Debugger in Oracle JDeveloper. These errors are logged in the log file.

Note: The SOA Debugger is currently available for BPEL, Mediator, BPM, and OSB processes with Development mode only.

9.3.2 Prerequisite

Ensure that the *IntegratedWebLogicServer* domain and a BPEL process are already created in Oracle JDeveloper.

9.3.3 Debugging a BPEL Process in Oracle JDeveloper

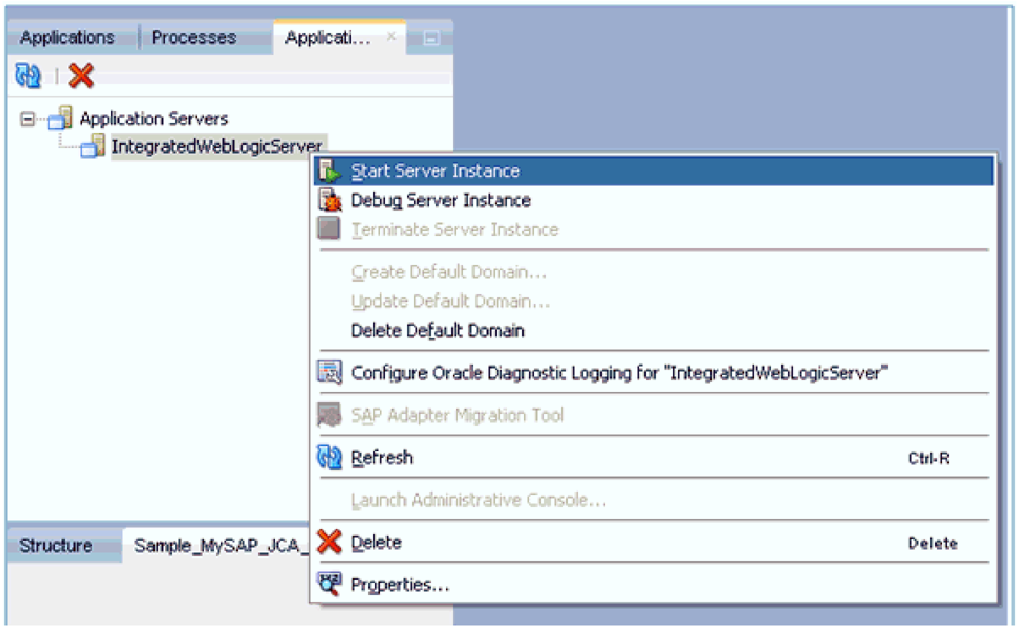
This section describes how to debug a BPEL process in Oracle JDeveloper. It contains the following topics:

- Section 9.3.3.1, "Debugging an Outbound BPEL Process in Oracle JDeveloper"
- Section 9.3.3.2, "Debugging an Inbound BPEL Process in Oracle JDeveloper"

9.3.3.1 Debugging an Outbound BPEL Process in Oracle JDeveloper

1. Open Oracle JDeveloper.
2. Start the *IntegratedWeblogicServer* domain.
   a. Click the **Application Servers** tab in the left pane.
   b. Under the Application Servers node, right-click **IntegratedWeblogicServer** and select **Start Server Instance** from the context menu, as shown in Figure 9–17.

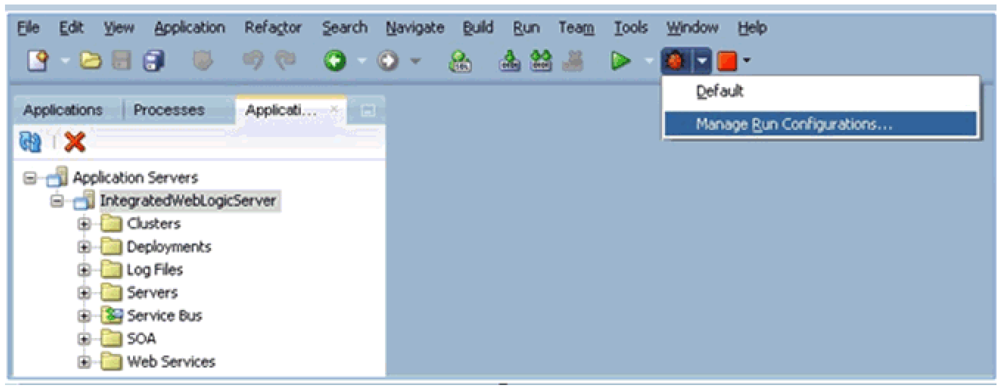
Figure 9–17 Start Server Instance



3. Set the Debugging environment.

   a. Click the down arrow next to the Debug icon and select **Manage Run Configurations** from the context menu, as shown in Figure 9–18.

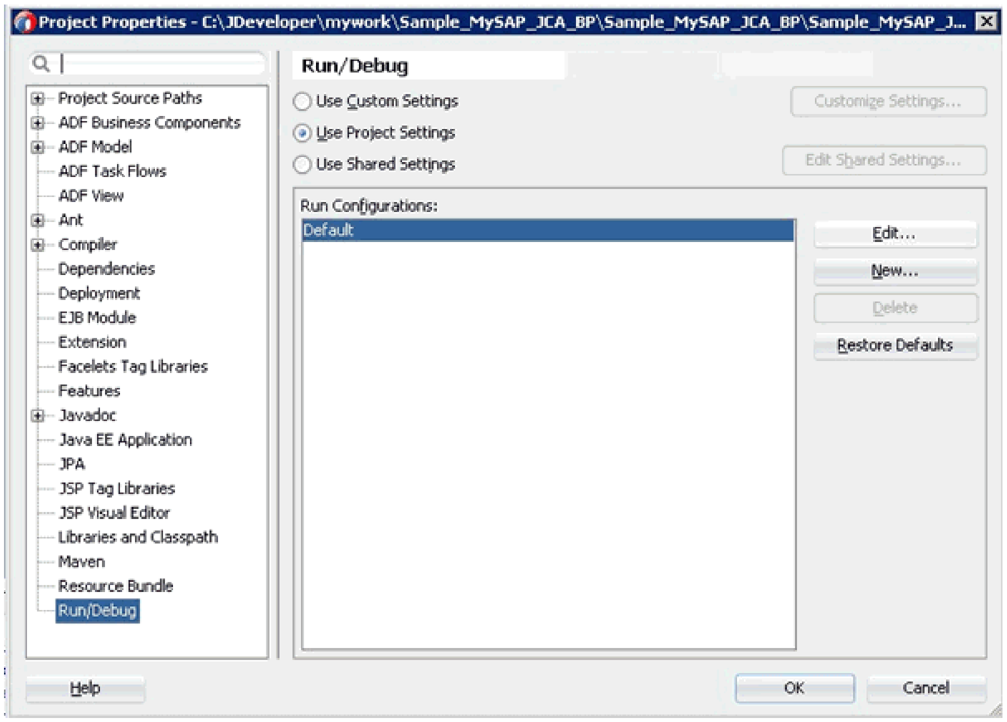
Figure 9–18 Manage Run Configurations



   b. Or, right-click the project and select **Project Properties**.

      The Project Properties dialog is displayed, as shown in Figure 9–19.

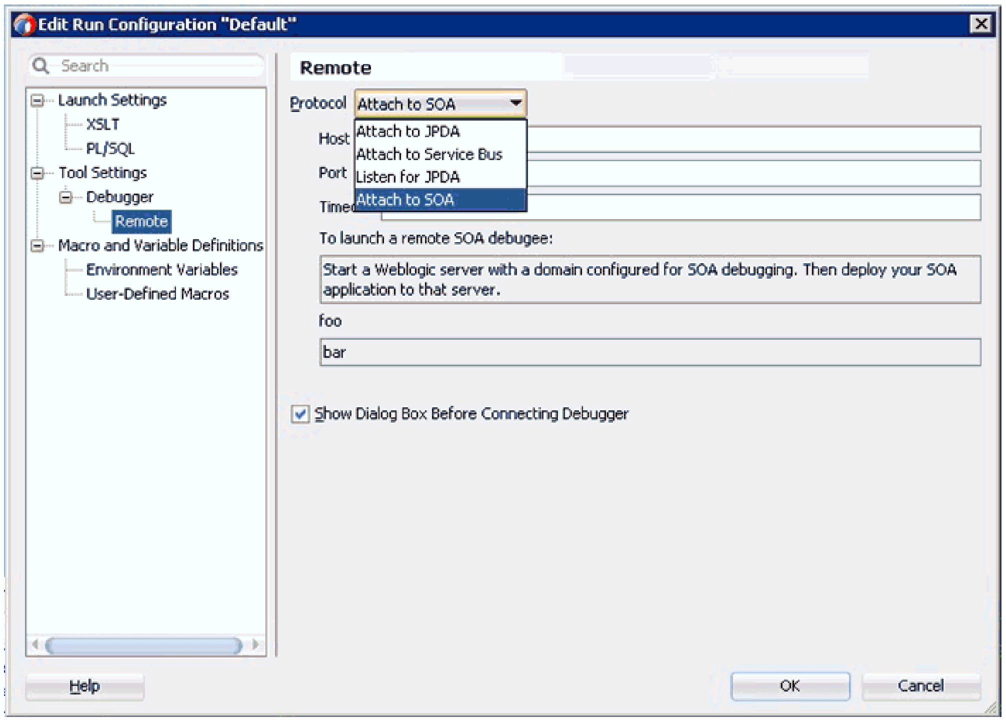
Figure 9–19 Project Properties Dialog



c. Select **Run/Debug** in the left pane and then click **Edit** in the Run/Debug pane (Run Configurations area).

The Edit Run Configuration dialog is displayed, as shown in Figure 9–20.

Figure 9–20 Edit Run Configuration Dialog



To launch a remote SOA debugee:

Start a Weblogic server with a domain configured for SOA debugging. Then deploy your SOA application to that server.

foo

bar

☑ Show Dialog Box Before Connecting Debugger

   d. Expand **Tool Settings**, **Debugger** in the left pane, and then click **Remote**.
   e. From the Protocol list, select **Attach to SOA**.
   f. Leave the default values for the Host and Port.
   g. Click **OK**.

4. Deploy the project.

   a. Right-click the project and select **Deploy**.

      The Deployment Action dialog is displayed.

      Select the application to deploy and click **Next**.

      The Deploy Configuration dialog is displayed.

   b. Click **Next**.

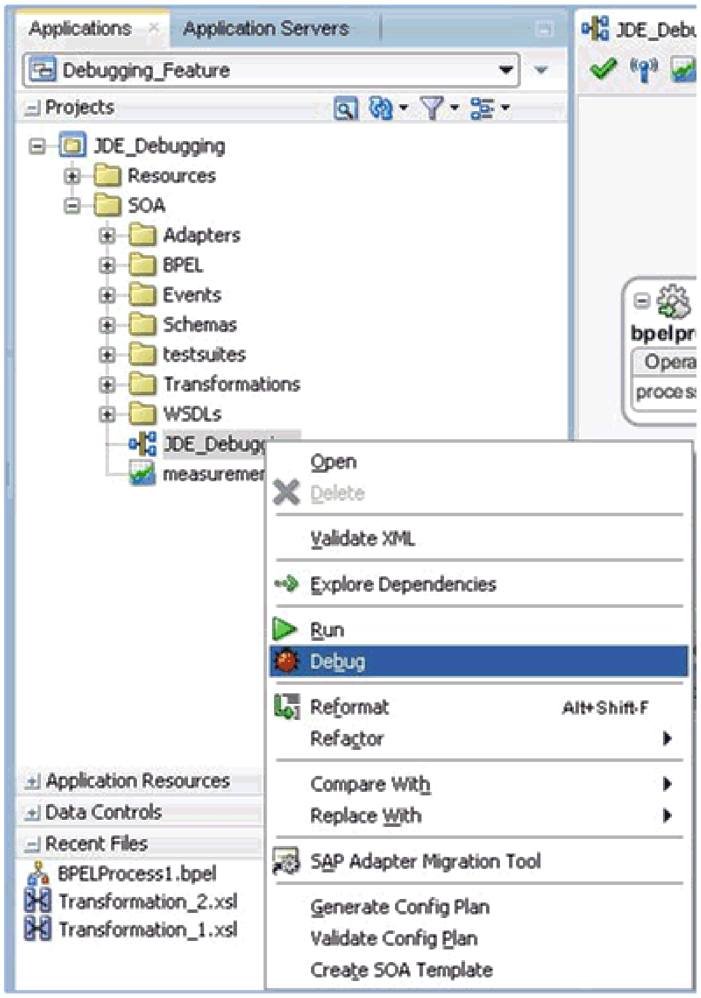
      The Application Servers dialog is displayed.

   c. Select **IntegratedWebLogicServer** and then click **Finish** to complete the deployment.
   d. Ensure that the project deployment has completed without any errors or issues before proceeding to the next step.

5. Connect a BPEL process to the SOA Debugger.

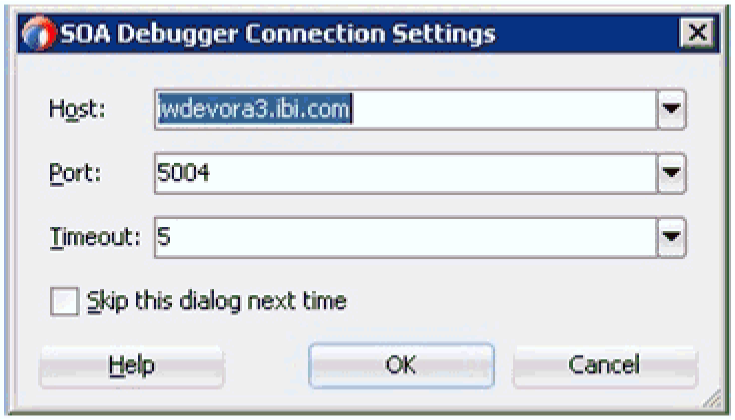
   a. In the Applications tab on the left pane, right-click a composite XML or project for an existing BPEL process and then select **Debug** from the context menu, as shown in Figure 9–21.

Figure 9–21 Select BPEL Process to Debug



The SOA Debugger Connection Settings dialog is displayed, as shown in Figure 9–22.

Figure 9–22 SOA Debugger Connection Settings Dialog


SOA Debugger Connection Settings

Host: iwdevora3.ibi.com

Port: 5004

Timeout: 5

☐ Skip this dialog next time

b. Click **OK**.

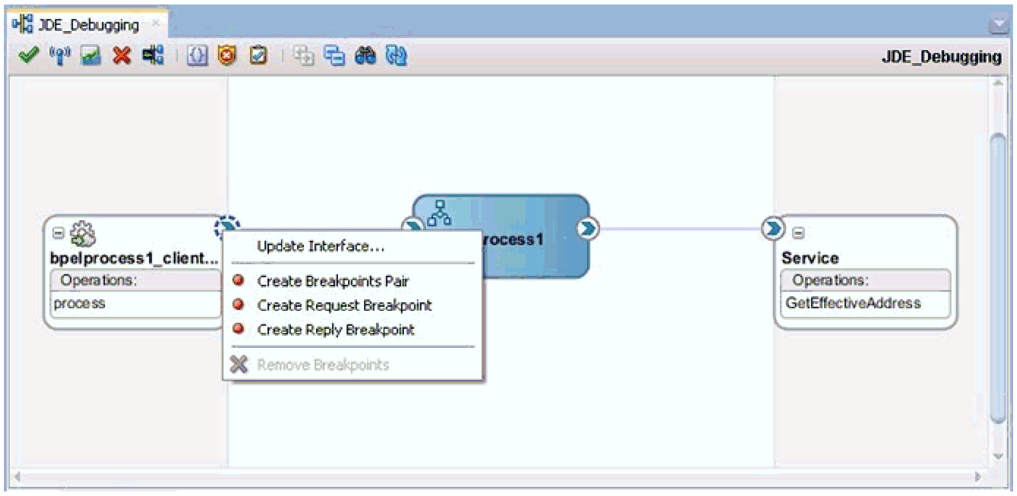
Once the BPEL process is connected to the SOA Debugger, the following messages will be displayed in the Debugging log:

```
Debugger attempting to connect to remote process at iwdevora3.ibi.com 5004.
Debugger connected to remote process at iwdevora3.ibi.com 5004.
Debugger process virtual machine is SOA Debugger
```

6. Set the Breakpoints and initiate debugging.

   a. Right-click on the components and select the Breakpoint type to set, as shown in Figure 9–23.

Figure 9–23 Selecting Breakpoints



Create Breakpoints Pair - Set this Breakpoint type for a request-reply (outbound-inbound) interaction. This is useful for scenarios in which both the request and reply are important.

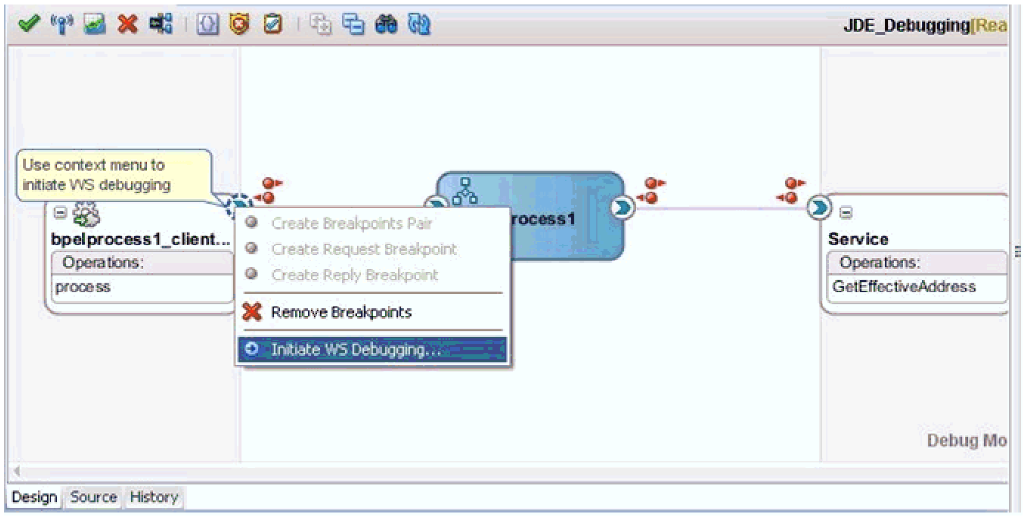
Create Request Breakpoint - Set this Breakpoint type for a request (outbound) interaction. This is useful for scenarios in which only the request is important.

Create Reply Breakpoint - Set this Breakpoint type for a reply (inbound) interaction. This is useful for scenarios in which only the reply is important.

Initiate WS Debugging - Set this Breakpoint type to initiate a debugging session. For example, the debugging session encompasses an initiating SOAP request from a web service to a BPEL process to an adapter reference binding component.

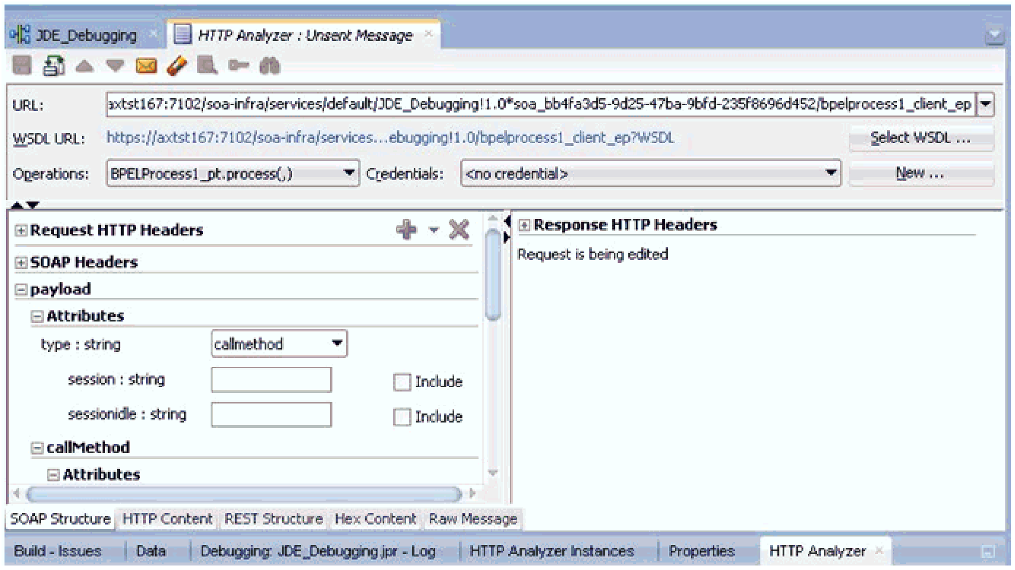
   b. Once the Breakpoints are set, right-click the right handle and select **Initiate WS Debugging**, as shown in Figure 9–24.

Figure 9–24 Initiate WS Debugging



The HTTP Analyzer dialog is displayed, as shown in Figure 9–25.

Figure 9–25 HTTP Analyzer Dialog



URL: axtst167:7102/soa-infra/services/default/JDE_Debugging!1.0*soa_bb4fa3d5-9d25-47ba-9bfd-235f8696d452/bpelprocess1_client_ep

WSDL URL: https://axtst167:7102/soa-infra/services...ebugging!1.0/bpelprocess1_client_ep?WSDL

Operations: BPELProcess1_pt.process(,)

Credentials: <no credential>

type : string callmethod

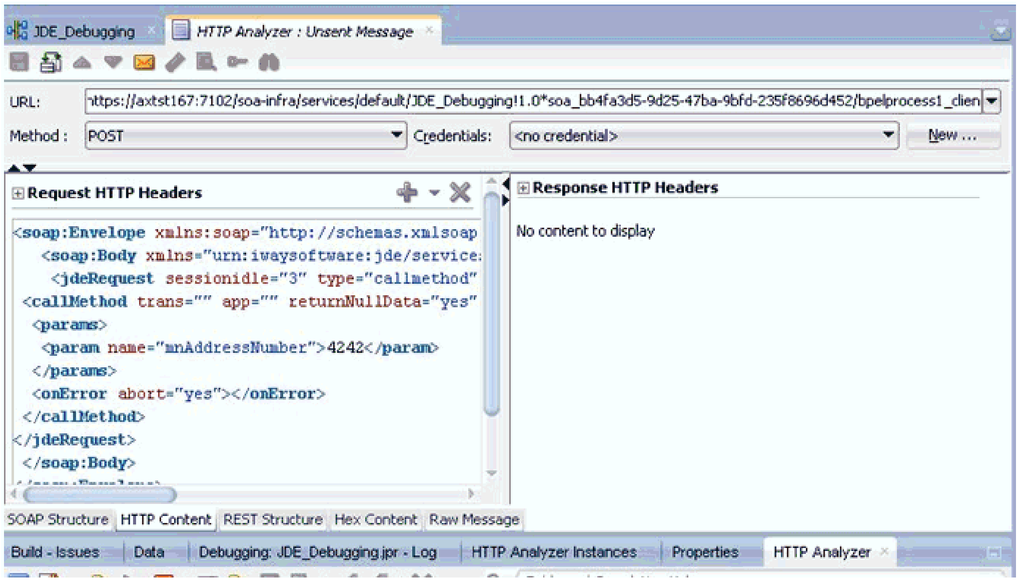
session : string

sessionidle : string

Request is being edited

c. Select HTTP content from the below tab. Now, copy and paste the payload into the body, as shown in Figure 9–26.

Figure 9–26 Select HTTP Content



URL: `ttps://axtst167:7102/soa-infra/services/default/JDE_Debugging!1.0*soa_bb4fa3d5-9d25-47ba-9bfd-235f8696d452/bpelprocess1_clien`

Method : POST

Credentials: <no credential>

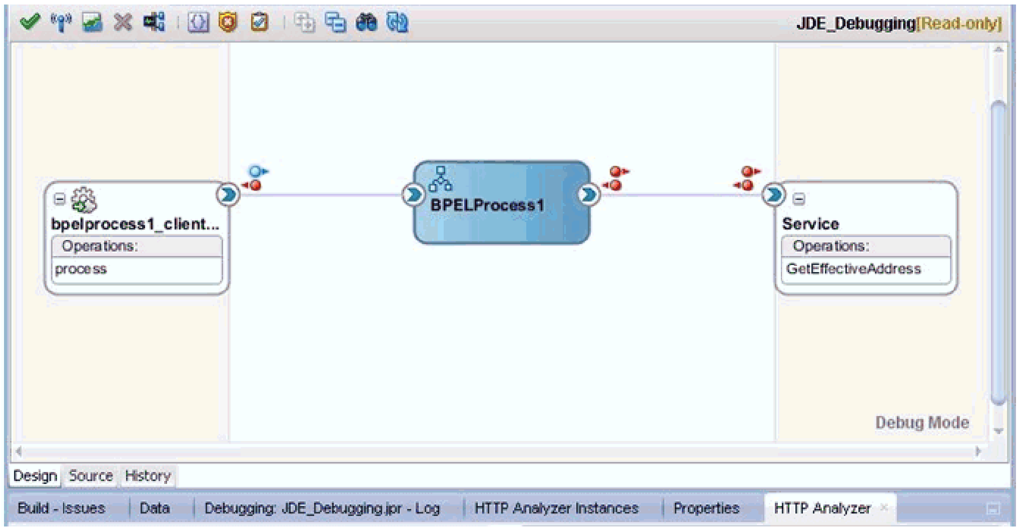
```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap
   <soap:Body xmlns="urn:iwaysoftware:jde/service:
    <jdeRequest sessionidle="3" type="callmethod"
 <callMethod trans="" app="" returnNullData="yes"
  <params>
   <param name="mnAddressNumber">4242</param>
  </params>
  <onError abort="yes"></onError>
 </callMethod>
</jdeRequest>
 </soap:Body>
```

No content to display

d. Click **Send Request**.

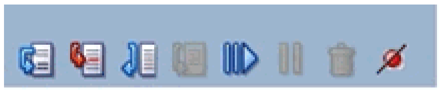
The BPEL Process stops at the designated Breakpoint and blinks in a blue color, as shown in Figure 9–27.

Figure 9–27 BPEL Process Stopped at Breakpoint



e. Use the available Step options to step through the Debugging process, as shown in Figure 9–28.

Figure 9–28 Step Options



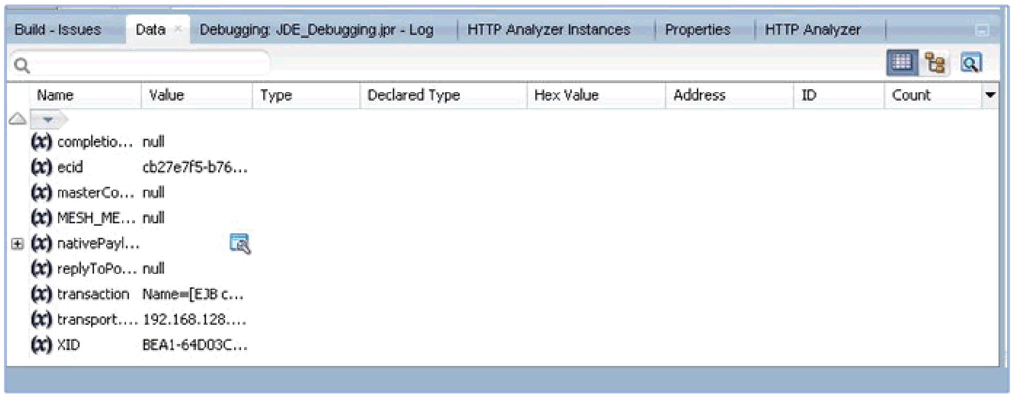
Icon	Description
	Ends or detaches from a debugging session.
	Steps over a frame. This places you at the next Breakpoint (for example, the receive activity in the BPEL process on which a Breakpoint was set. If there are no Breakpoints, it steps over all the frames and returns to the first frame. You can also press **F8** to step over a frame.
	Steps into the next valid location. This can be a new frame or the same frame, but in a different location. You can also press **F7** to step into a frame.
	Steps out of a frame. This option is only used to process a BPEL scope or sequence activity. After completion of scope processing, it pauses at the next scope or activity in the process. You can also press **Shift-F7**.
	Resumes a step operation. You can also press **F9** to resume.

7. View the Request payload and header information.

a. Click **Windows**, select **Debugger**, and then **Data**.

To view sample header information when a Breakpoint stops at the Request Breakpoint for Oracle Application Adapter for J.D. Edwards OneWorld, see Figure 9–29.

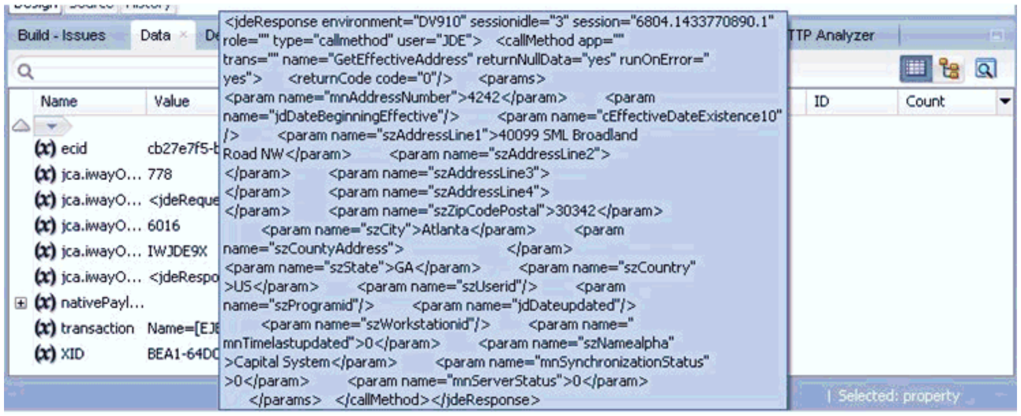
Figure 9–29 Header Information



Name	Value	Type	Declared Type	Hex Value	Address	ID	Count
(x) completio...	null						
(x) ecid	cb27e7f5-b76...						
(x) masterCo...	null						
(x) MESH_ME...	null						
(x) nativePayl...							
(x) replyToPo...	null						
(x) transaction	Name=[EJB c...						
(x) transport....	192.168.128....						
(x) XID	BEA1-64D03C...						

A sample response payload for Oracle Application Adapter for J.D. Edwards OneWorld is shown in Figure 9–30.

Figure 9–30 Sample Response Payload



```
Name                Value
(x) ecid            cb27e7f5-t
(x) jca.iwayO...    778
(x) jca.iwayO...    <jdeReque
(x) jca.iwayO...    6016
(x) jca.iwayO...    IWJDE9X
(x) jca.iwayO...    <jdeRespo
(x) nativePayl...
(x) transaction     Name=[EJ
(x) XID             BEA1-64DC
```

```
<jdeResponse environment="DV910" sessionidle="3" session="6804.1433770890.1"
role="" type="callmethod" user="JDE">   <callMethod app=""
trans="" name="GetEffectiveAddress" returnNullData="yes" runOnError="
yes">     <returnCode code="0"/>     <params>
<param name="mnAddressNumber">4242</param>       <param
name="jdDateBeginningEffective"/>       <param name="cEffectiveDateExistence10"
/>       <param name="szAddressLine1">40099 SML Broadland
Road NW</param>       <param name="szAddressLine2">
</param>       <param name="szAddressLine3">
</param>       <param name="szAddressLine4">
</param>       <param name="szZipCodePostal">30342</param>
      <param name="szCity">Atlanta</param>       <param
name="szCountyAddress">                   </param>
<param name="szState">GA</param>       <param name="szCountry"
>US</param>       <param name="szUserid"/>       <param
name="szProgramid"/>       <param name="jdDateupdated"/>
      <param name="szWorkstationid"/>       <param name="
mnTimelastupdated">0</param>       <param name="szNamealpha"
>Capital System</param>       <param name="mnSynchronizationStatus"
>0</param>       <param name="mnServerStatus">0</param>
    </params>   </callMethod></jdeResponse>
```

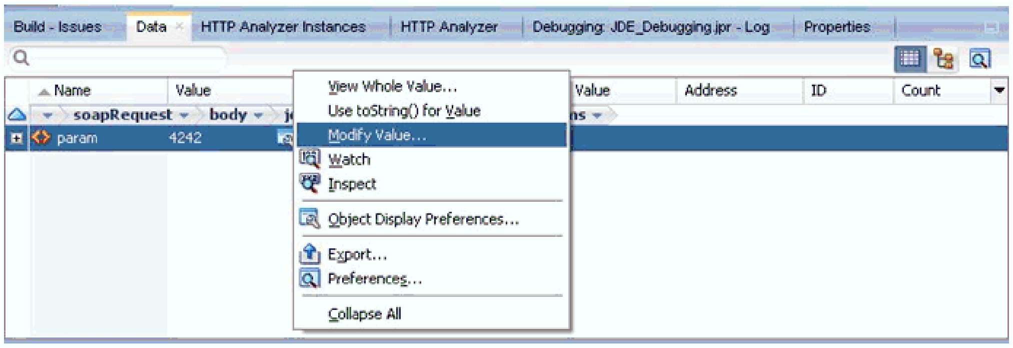
Note: The payload display is limited to the screen size as shown in Figure 9–30. However, the Breakpoint at the BPEL process displays the complete payload, allows scrolling and viewing all elements of the payload.

---

8. Modify the Request payload content.

   a. Expand the SOAP request, select the field to modify, right-click and select **Modify Value** from the context menu, as shown in Figure 9–31.

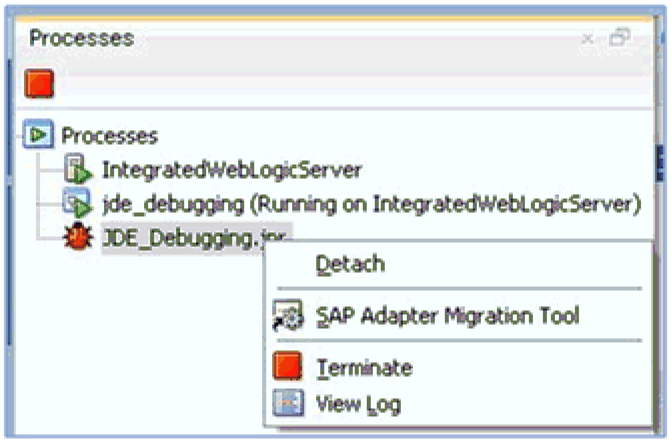
Figure 9–31 Modify Value



9. End or detach the Debugging session.

   a. Click **Window** and then **Processes**. Right click on the process in the Processes tab and select **Detach** or **Terminate**, as shown in Figure 9–32.

Figure 9–32 Detach Debugging Session



b. Select one of the following options:

   Detach - Removes the SOA Debugger without ending the debugging process.

   Terminate - Ends the debugging process.

9.3.3.2 Debugging an Inbound BPEL Process in Oracle JDeveloper

1. Deploy the Inbound BPEL process.

   a. Right-click the project and select **Deploy**.

      The Deployment Action dialog is displayed.

   b. Select **Deploy to Application Server** and click **Next**.

      The Deploy Configuration dialog is displayed.

   c. Click **Next**.

      The Select Server dialog is displayed.

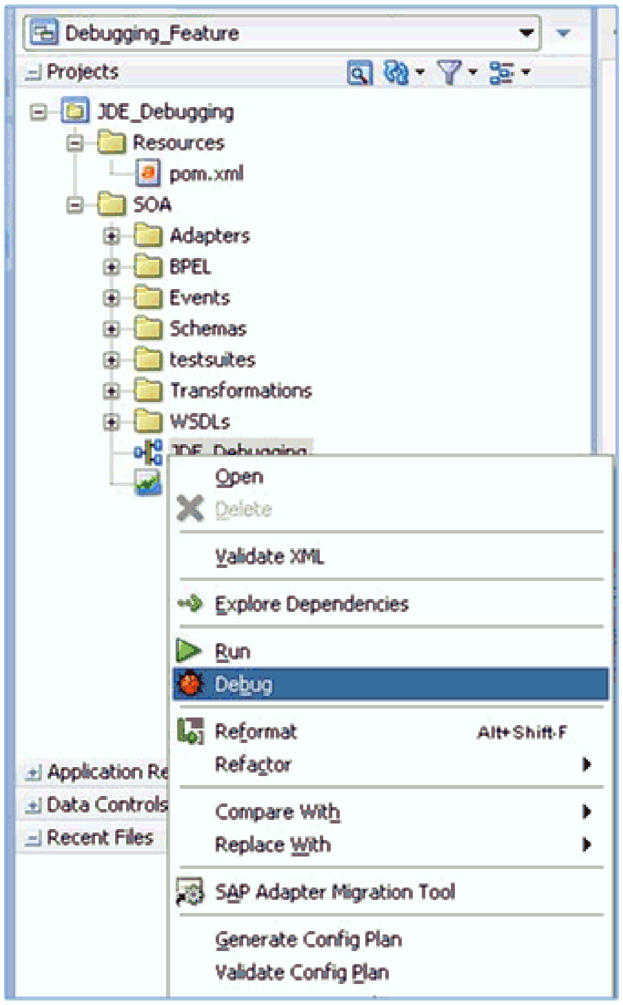
   d. Select the server to deploy and click **Finish**.

   e. Ensure that the project deployment has completed without any errors or issues before proceeding to the next step.

2. Connect a BPEL process to the SOA Debugger.

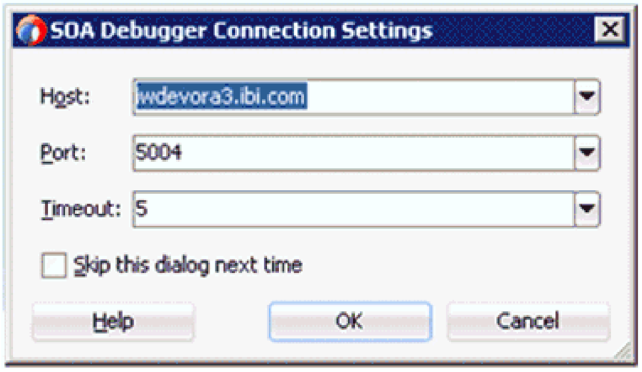
   a. Right-click an inbound composite and select **Debug**, as shown in Figure 9–33.

Figure 9–33 Select BPEL Process to Debug



The SOA Debugger Connection Settings dialog is displayed, as shown in Figure 9–34.

Figure 9–34 SOA Debugger Connection Settings Dialog


SOA Debugger Connection Settings

Host: wdevora3.ibi.com

Port: 5004

Timeout: 5

☐ Skip this dialog next time

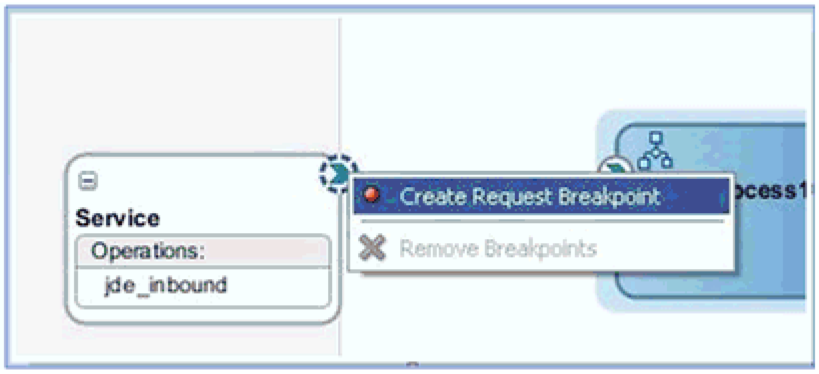
b. Click **OK**.

Once the BPEL process is connected to the SOA Debugger, the following messages will be displayed in the Debugging log:

```
Debugger attempting to connect to remote process at iwdevora3.ibi.com 5004.
Debugger connected to remote process at iwdevora3.ibi.com 5004.
Debugger process virtual machine is SOA Debugger
```

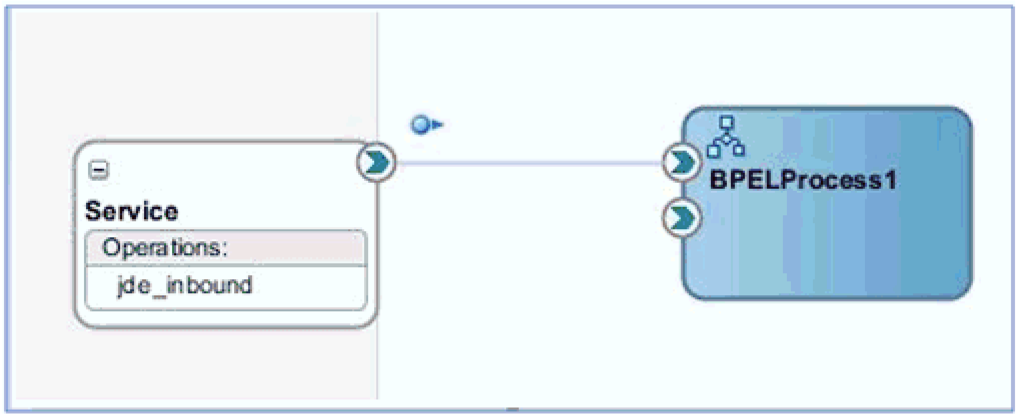
3. Set the Breakpoints, as shown in Figure 9–35.

Figure 9–35 Setting Breakpoints



4. Once the Breakpoints are set and a message is received through the inbound process (for example, by triggering from J.D. Edwards OneWorld), the process stops at the designated Breakpoints, as shown in Figure 9–36.

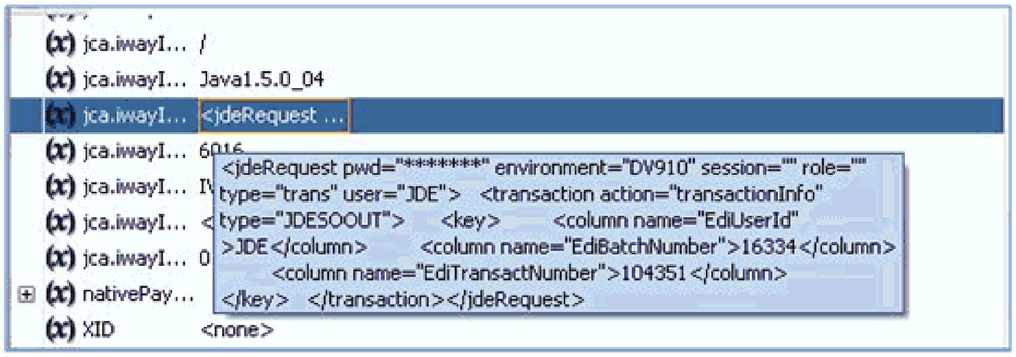
Figure 9–36 Stopping the Process at the Breakpoint



5. View the Request payload and header information.

   a. Click **Windows**, select **Debugger**, and then **Data**, as shown in Figure 9–37.

Figure 9–37 Debugger Data

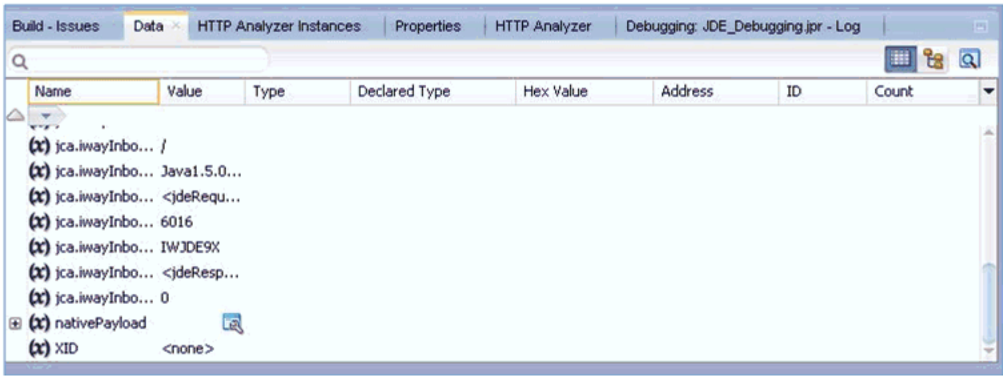


```
(x) jca.iwayI... /
(x) jca.iwayI... Java1.5.0_04
(x) jca.iwayI... <jdeRequest ...
(x) jca.iwayI... 6016
(x) jca.iwayI... I\
(x) jca.iwayI... <
(x) jca.iwayI... 0
(x) nativePay...
(x) XID          <none>
```

```
<jdeRequest pwd="*******" environment="DV910" session="" role=""
type="trans" user="JDE">  <transaction action="transactionInfo"
type="JDESOOUT">     <key>        <column name="EdiUserId"
>JDE</column>       <column name="EdiBatchNumber">16334</column>
        <column name="EdiTransactNumber">104351</column>
</key>  </transaction></jdeRequest>
```

b. Header information displayed for Inbound Request Breakpoint (SalesOrder) is shown in Figure 9–38.

Figure 9–38 Inbound Request Breakpoint (SalesOrder)

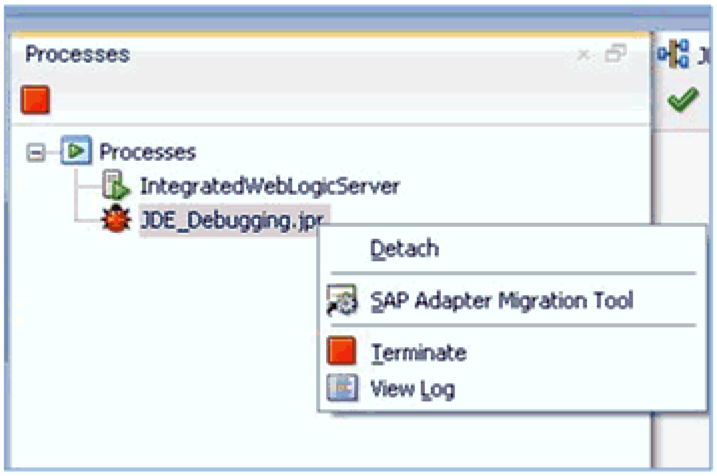


Name	Value	Type	Declared Type	Hex Value	Address	ID	Count
(x) jca.iwayInbo...	/						
(x) jca.iwayInbo...	Java1.5.0...						
(x) jca.iwayInbo...	<jdeRequ...						
(x) jca.iwayInbo...	6016						
(x) jca.iwayInbo...	IWJDE9X						
(x) jca.iwayInbo...	<jdeResp...						
(x) jca.iwayInbo...	0						
(x) nativePayload							
(x) XID	<none>						

6. End or detach the Debugging session.

   a. Click **Window** and then **Processes**. Right click on the process in the Processes tab and select **Detach** or **Terminate**, as shown in Figure 9–39.

Figure 9–39 Detach Debugging Session



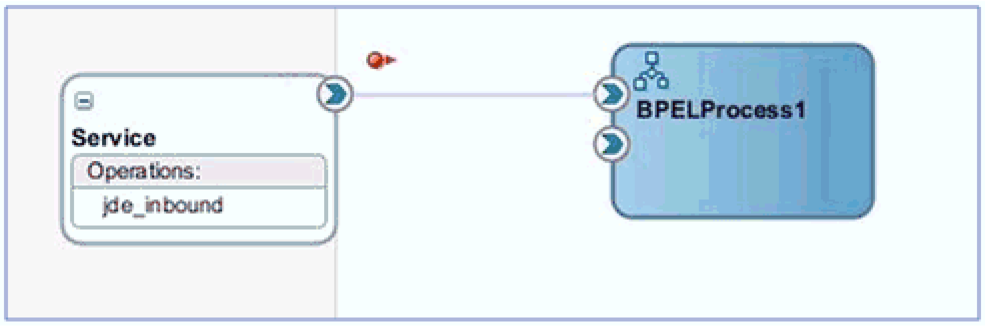
   b. Select one of the following options:

Detach - Removes the SOA Debugger without ending the debugging process.

Terminate - Ends the debugging process.

The process will be detached and is displayed, as shown in Figure 9–40.

Figure 9–40 BPEL Process Detached



9.3.4 Debugging an OSB Process in Oracle JDeveloper

This section describes how to debug an OSB process in Oracle JDeveloper. It contains the following topics:

- Section 9.3.4.1, "Prerequisites"
- Section 9.3.4.2, "Debugging an Outbound OSB Process in Oracle JDeveloper"
- Section 9.3.4.3, "Debugging an Inbound OSB Process in Oracle JDeveloper"

9.3.4.1 Prerequisites

Ensure that the *IntegratedWebLogicServer* domain and an OSB process with file output are already created in Oracle JDeveloper.

Note: Ensure that the application name and the project name are the same.

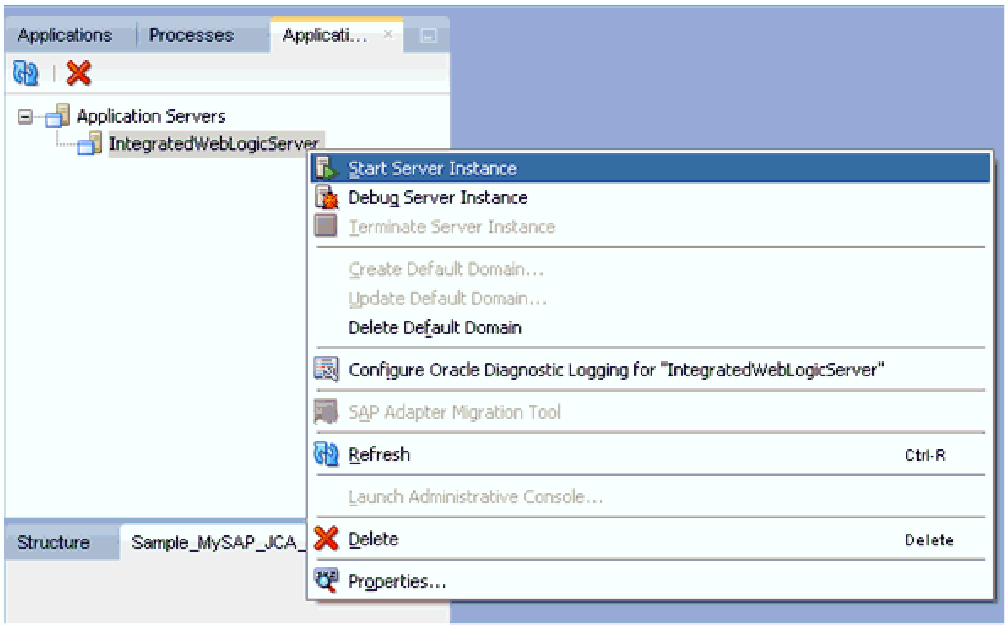
---

1. Open Oracle JDeveloper and start *IntegratedWeblogicServer* or `startWebLogic.cmd`.

   a. Click the **Application Servers** tab in the left pane.

   b. Under the Application Servers node, right-click **IntegratedWeblogicServer** and select **Start Server Instance** from the context menu, as shown in Figure 9–41.

Figure 9–41 Start Server Instance

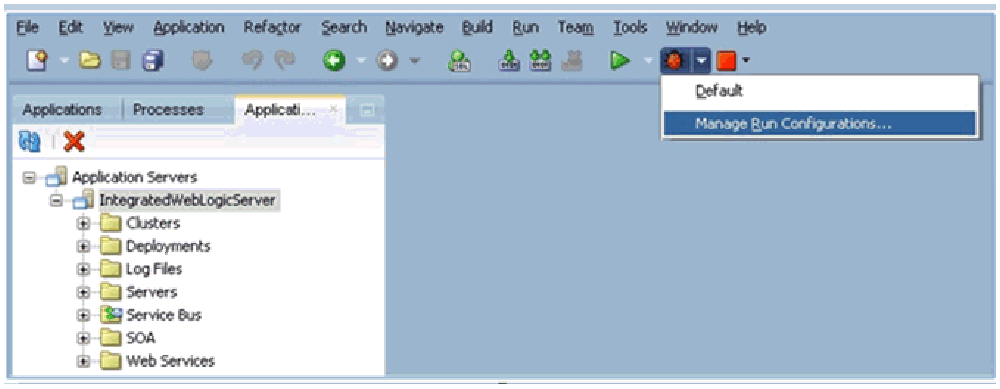


c. Or, start Oracle WebLogic server from the command prompt using `startWebLogic.cmd`.

2. Set the Debugging environment.

   a. Click the down arrow next to the Debug icon and select **Manage Run Configurations** from the context menu, as shown in Figure 9–42.

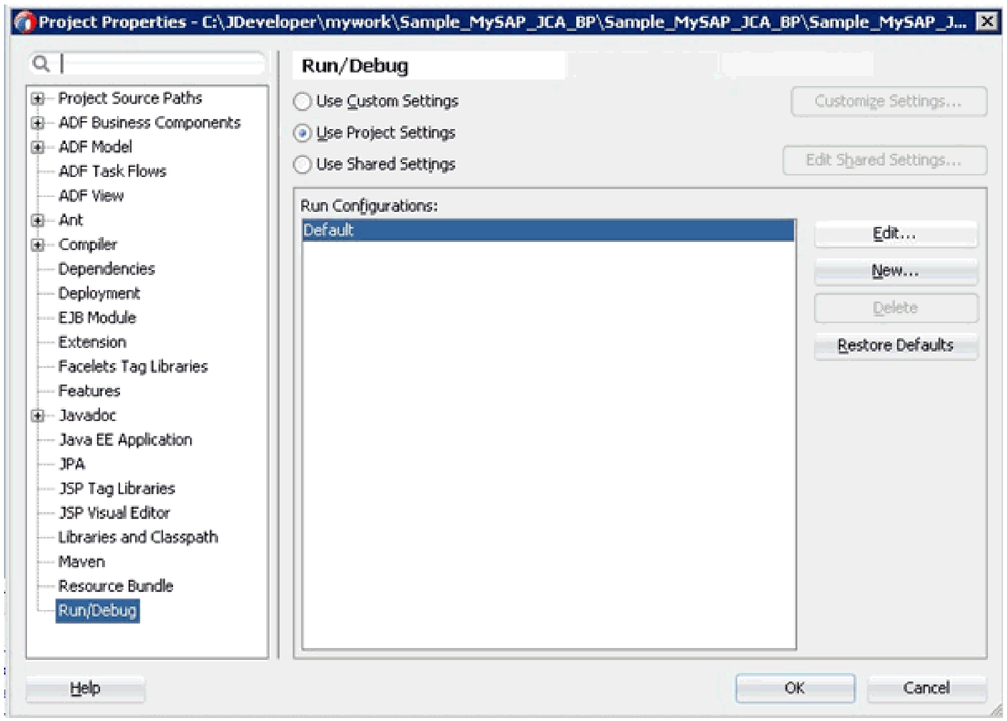
Figure 9–42 Manage Run Configurations



   b. Or, right-click the project and select **Project Properties**.

   The Project Properties dialog is displayed, as shown in Figure 9–43.

Figure 9–43 Project Properties Dialog


Project Properties - C:\JDeveloper\mywork\Sample_MySAP_JCA_BP\Sample_MySAP_JCA_BP\Sample_MySAP_J...

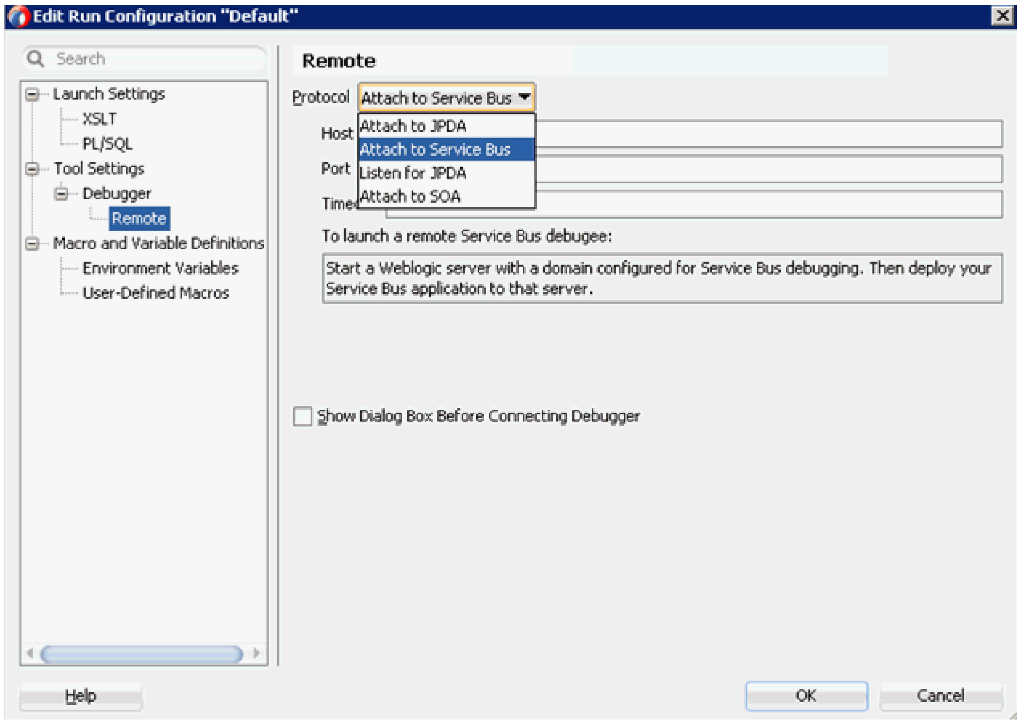
Project Source Paths
ADF Business Components
ADF Model
ADF Task Flows
ADF View
Ant
Compiler
Dependencies
Deployment
EJB Module
Extension
Facelets Tag Libraries
Features
Javadoc
Java EE Application
JPA
JSP Tag Libraries
JSP Visual Editor
Libraries and Classpath
Maven
Resource Bundle
Run/Debug

Run/Debug
Use Custom Settings
[✓ Use Project Settings]
Use Shared Settings
Run Configurations:
Default

c. Select **Run/Debug** in the left pane and then click **Edit** in the Run/Debug pane (Run Configurations area).

The Edit Run Configuration dialog is displayed, as shown in Figure 9–44.

Figure 9–44 Edit Run Configuration Dialog


Edit Run Configuration "Default"

Launch Settings
XSLT
PL/SQL
Tool Settings
Debugger
Remote
Macro and Variable Definitions
Environment Variables
User-Defined Macros

Remote
Protocol: Attach to Service Bus
Attach to JPDA
Attach to Service Bus
Listen for JPDA
Attach to SOA
Host
Port
Timeout
To launch a remote Service Bus debugee:
Start a Weblogic server with a domain configured for Service Bus debugging. Then deploy your Service Bus application to that server.
Show Dialog Box Before Connecting Debugger

d. Expand **Tool Settings**, **Debugger** in the left pane, and then click **Remote**.

e. From the Protocol list, select **Attach to Service Bus**.

f. Leave the default values for the Host and Port.

g. Click **OK**.

9.3.4.2 Debugging an Outbound OSB Process in Oracle JDeveloper

1. Deploy the project.

   a. Right-click the project and select **Deploy**.

      The Deployment Action dialog is displayed.

   b. Select the application to deploy and click **Next**.

      The Deploy Configuration dialog is displayed.

   c. Click **Next**.

      The Application Servers dialog is displayed.

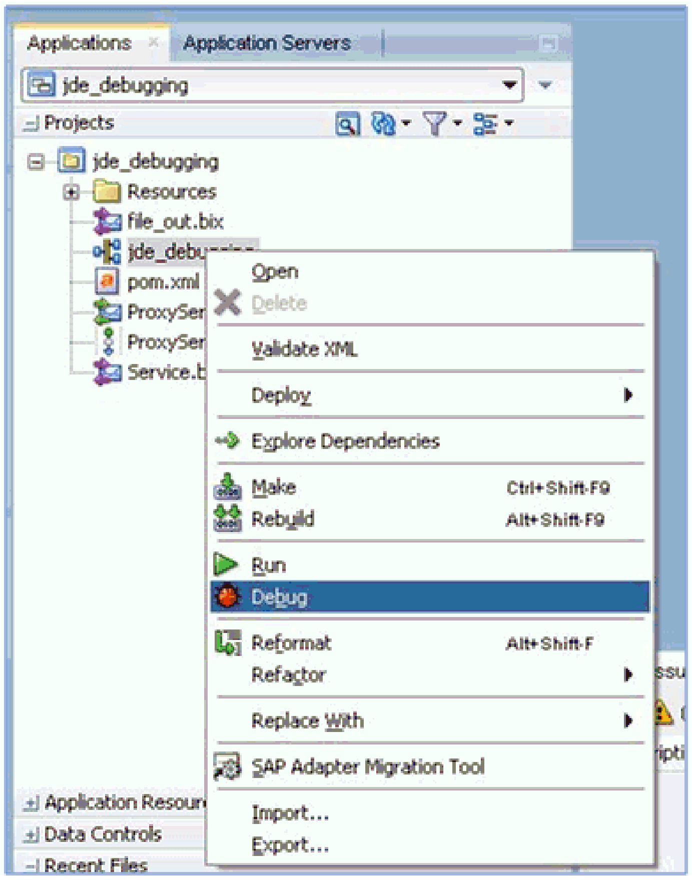
   d. Select **IntegratedWebLogicServer** and then click **Finish** to complete the deployment.

   e. Ensure that the project deployment has completed without any errors or issues before proceeding to the next step.

2. Connect an OSB process to the SOA Debugger.

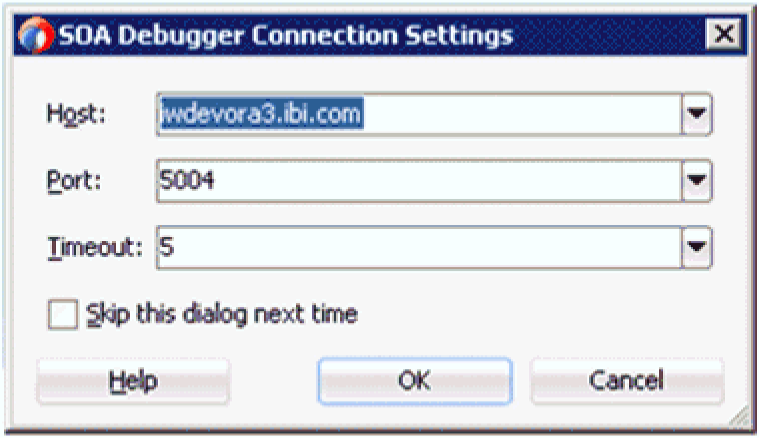
   a. In the Applications tab on the left pane, right-click a composite XML or project for an existing OSB process and then select **Debug** from the context menu, as shown in Figure 9–45.

Figure 9–45 Select OSB Process to Debug



The SOA Debugger Connection Settings dialog is displayed, as shown in Figure 9–46.

Figure 9–46 SOA Debugger Connection Settings Dialog



Host: iwdevora3.ibi.com

Port: 5004

Timeout: 5

☐ Skip this dialog next time

b. Click **OK**.

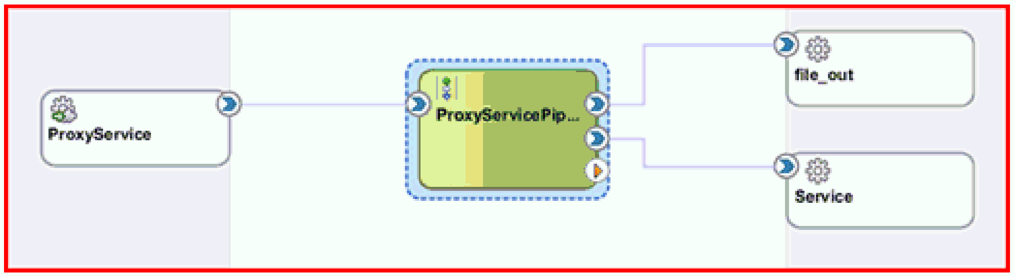
Once the OSB process is connected to the SOA Debugger, the following messages will be displayed in the Debugging log:

```
Debugger attempting to connect to remote process at iwdevora3.ibi.com 5004.
Debugger connected to remote process at iwdevora3.ibi.com 5004.
Debugger process virtual machine is SOA Debugger
```

3. Set the Breakpoints.

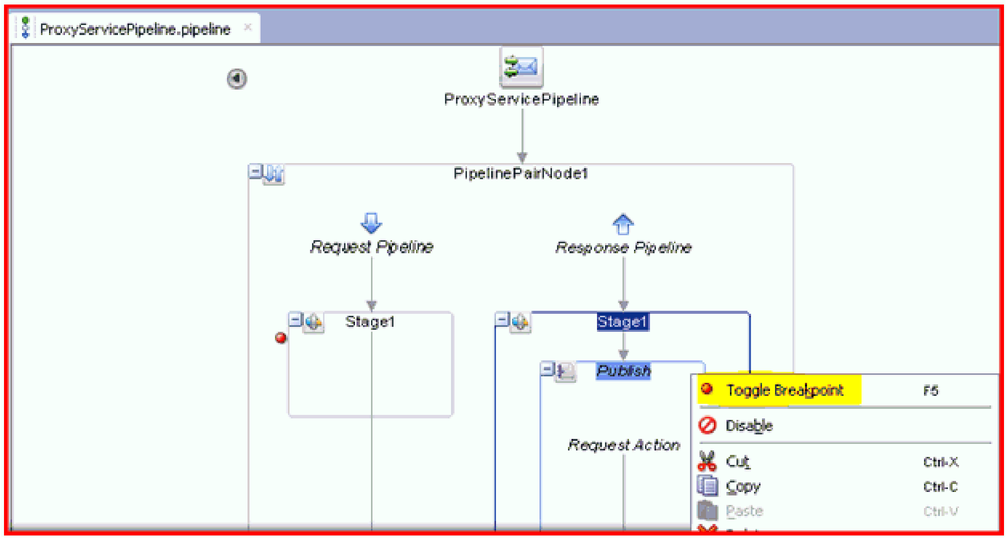
   a. Open the pipeline in its editor by double-clicking on the pipeline, as shown in Figure 9–47.

Figure 9–47 ProxyServicePipeline



   b. Expand the actions until you reach the node where the Breakpoint must be added. In this case, expand **PipelinePairNode1**, right-click the **Stage1** node (under **Request Pipeline**), and select **Toggle Breakpoint** from the context menu, as shown in Figure 9–48.

Figure 9–48 Toggle Breakpoint



Repeat this step for the Publish node (under **Response Pipeline**).

A red icon appears next to the node to indicate that a Breakpoint has been set.

Note: To disable a Breakpoint, right-click the node and select **Disable Breakpoint**.

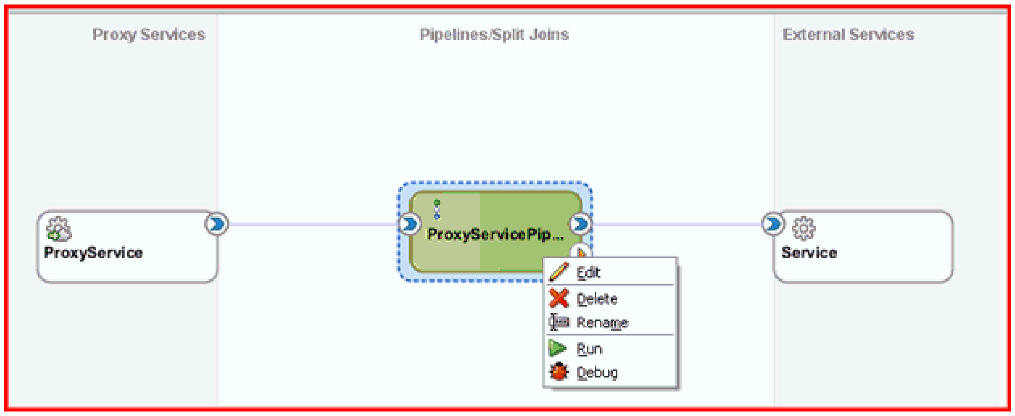
To remove a Breakpoint, right-click the node and select **Toggle Breakpoint** again.

---

4. Initiate Debugging.

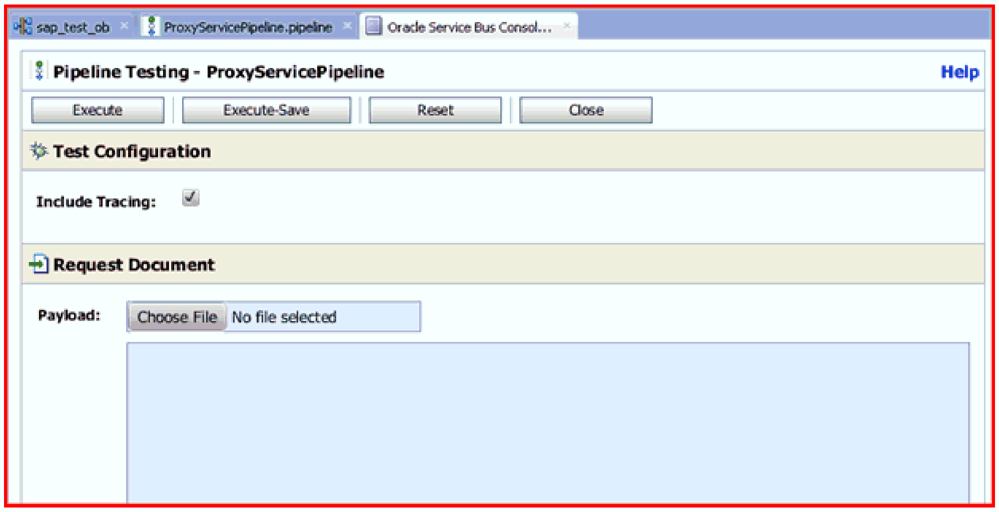
a. Right-click the pipeline in the Application Navigator, and select **Debug**, as shown in Figure 9–49.

Figure 9–49 Select Debug



The process is deployed to the integrated server and the Test Configuration pane will be displayed, as shown in Figure 9–50.

Figure 9–50 Test Configuration Pane



Pipeline Testing - ProxyServicePipeline

Test Configuration

Include Tracing: ✓

Request Document

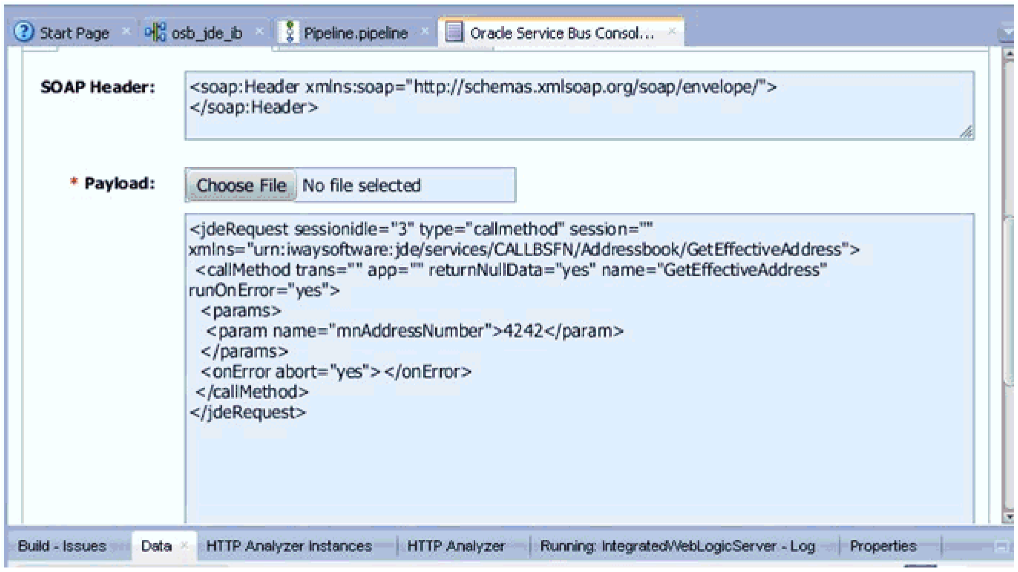
Payload:

Note: If there is no domain currently running, then the Create Default Domain dialog will be displayed. Enter the connection information for the integrated server and then click **OK**. This process may take several minutes.

---

b. In the Test Configuration pane, enter the test data in the Request Document area, and configure any additional input as required, as shown in Figure 9–51.

Figure 9–51 Test Request Document



SOAP Header:

```
<soap:Header xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
</soap:Header>
```

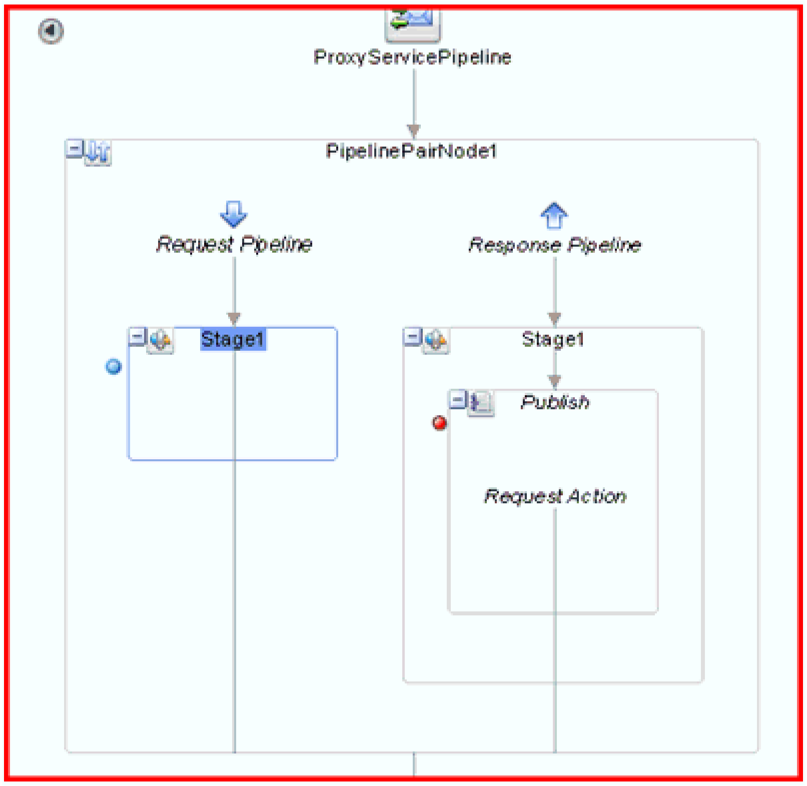
* Payload: No file selected

```
<jdeRequest sessionidle="3" type="callmethod" session=""
xmlns="urn:iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress">
 <callMethod trans="" app="" returnNullData="yes" name="GetEffectiveAddress"
runOnError="yes">
  <params>
   <param name="mnAddressNumber">4242</param>
  </params>
  <onError abort="yes"></onError>
 </callMethod>
</jdeRequest>
```

c. Click **Execute**.

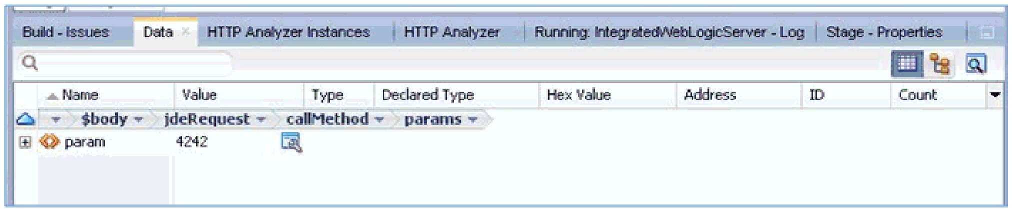
The Test Configuration pane executes the command, but the OSB process stops at the designated Breakpoint and blinks in a blue color, as shown in Figure 9–52.

Figure 9–52 OSB Process Stopped at Breakpoint



d. Check the Data tab at the bottom to verify that the input has passed, as shown in Figure 9–53.

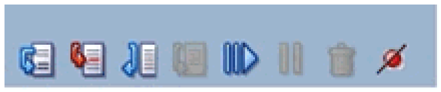
Figure 9–53 Data Tab



Name	Value	Type	Declared Type	Hex Value	Address	ID	Count
param	4242						

e. Use the available Step options to step through the Debugging process, as shown in Figure 9–54.

Figure 9–54 Step Options

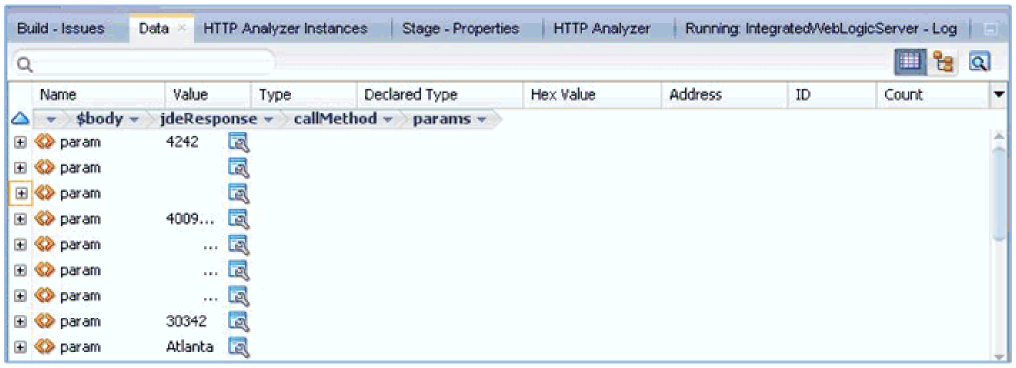


Icon	**Description**
■	Ends or detaches from a debugging session.
	Steps over a frame. This places you at the next Breakpoint (for example, the receive activity in the OSB process on which a Breakpoint was set. If there are no Breakpoints, it steps over all the frames and returns to the first frame. You can also press **F8** to step over a frame.
	Steps into the next valid location. This can be a new frame or the same frame, but in a different location. You can also press **F7** to step into a frame.
	Steps out of a frame. This option is only used to process an OSB scope or sequence activity. After completion of scope processing, it pauses at the next scope or activity in the process. You can also press **Shift-F7**.
	Resumes a step operation. You can also press **F9** to resume.

f. Use **Step Over** to go to the next Breakpoint (Response Pipeline in this example).

You will be able to see the output in the Data tab, as shown in Figure 9–55.

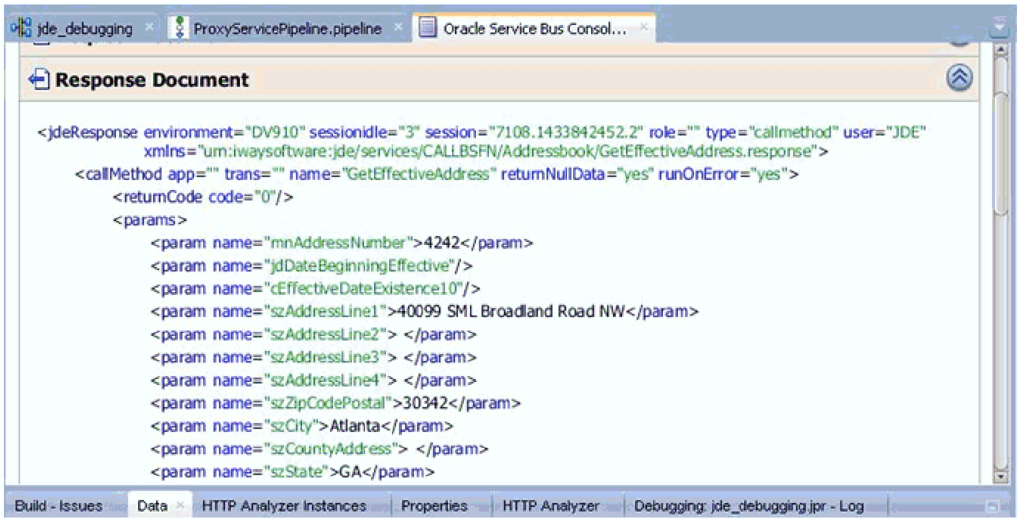
Figure 9–55 Data Tab Output



Name	Value	Type	Declared Type	Hex Value	Address	ID	Count
param	4242						
param							
param							
param	4009...						
param	...						
param	...						
param	...						
param	30342						
param	Atlanta						

You will also be able to see the response in the Oracle Service Bus Console, as shown in Figure 9–56.

Figure 9–56 Oracle Service Bus Console

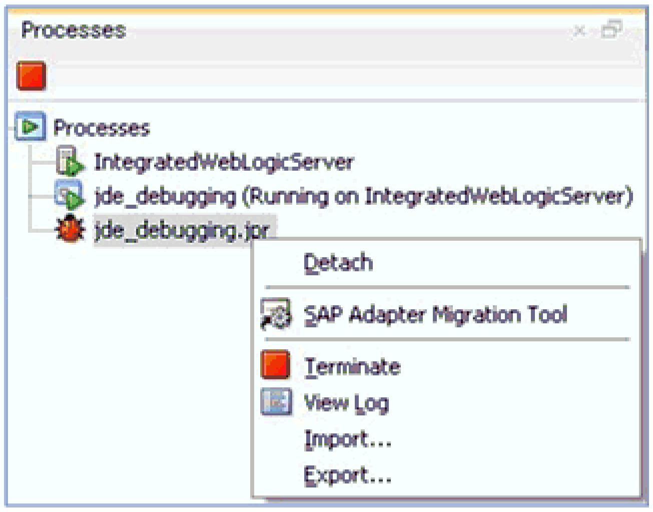


```
<jdeResponse environment="DV910" sessionidle="3" session="7108.1433842452.2" role="" type="callmethod" user="JDE"
            xmlns="urn:iwaysoftware:jde/services/CALLBSFN/Addressbook/GetEffectiveAddress.response">
    <callMethod app="" trans="" name="GetEffectiveAddress" returnNullData="yes" runOnError="yes">
        <returnCode code="0"/>
        <params>
            <param name="mnAddressNumber">4242</param>
            <param name="jdDateBeginningEffective"/>
            <param name="cEffectiveDateExistence10"/>
            <param name="szAddressLine1">40099 SML Broadland Road NW</param>
            <param name="szAddressLine2"> </param>
            <param name="szAddressLine3"> </param>
            <param name="szAddressLine4"> </param>
            <param name="szZipCodePostal">30342</param>
            <param name="szCity">Atlanta</param>
            <param name="szCountyAddress"> </param>
            <param name="szState">GA</param>
```

5. End or detach the Debugging session.

   a. Click **Window** and then **Processes**. Right click on the process in the Processes tab and select **Detach** or **Terminate**, as shown in Figure 9–57.

Figure 9–57 Detach Debugging Session



b. Select one of the following options:

   Detach - Removes the SOA debugger without ending the debugging process.

   Terminate - Ends the debugging process.

9.3.4.3 Debugging an Inbound OSB Process in Oracle JDeveloper

1. Create an inbound OSB process.

   Ensure that the application name and the project name are the same.

2. Deploy the project.

   a. Right-click the project and select **Deploy**.

      The Deployment Action dialog is displayed.

   b. Select the application to deploy and click **Next**.

      The Deploy Configuration dialog is displayed.

   c. Click **Next**.

      The Application Servers dialog is displayed.

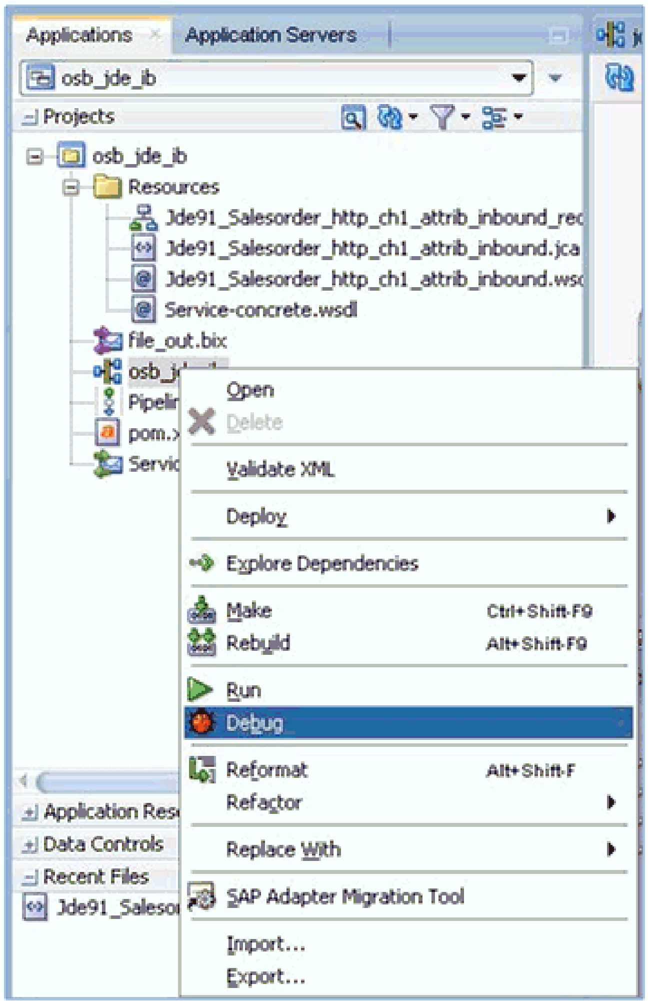
   d. Select **IntegratedWebLogicServer** and then click **Finish** to complete the deployment.

   e. Ensure that the project deployment has completed without any errors or issues before proceeding to the next step.

3. Connect an OSB process to the SOA Debugger.

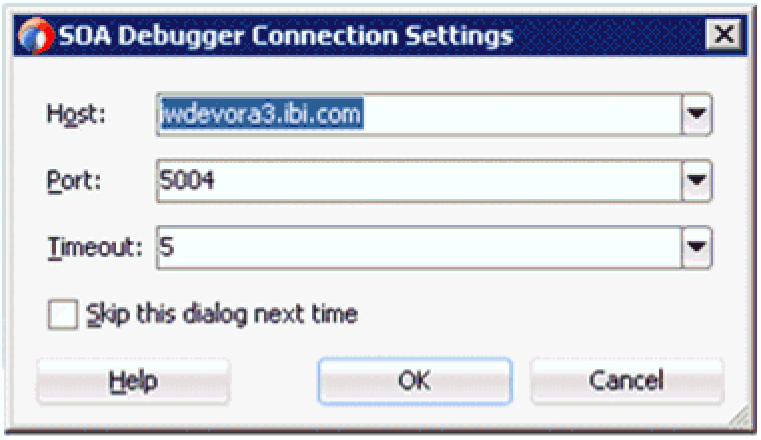
   a. In the Applications tab on the left pane, right-click a composite XML or project for an existing OSB process and then select **Debug** from the context menu, as shown in Figure 9–58.

Figure 9–58 Select OSB Process to Debug



The SOA Debugger Connection Settings dialog is displayed, as shown in Figure 9–59.

Figure 9–59 SOA Debugger Connection Settings Dialog


SOA Debugger Connection Settings

Host: wdevora3.ibi.com

Port: 5004

Timeout: 5

☐ Skip this dialog next time

b. Click **OK**.

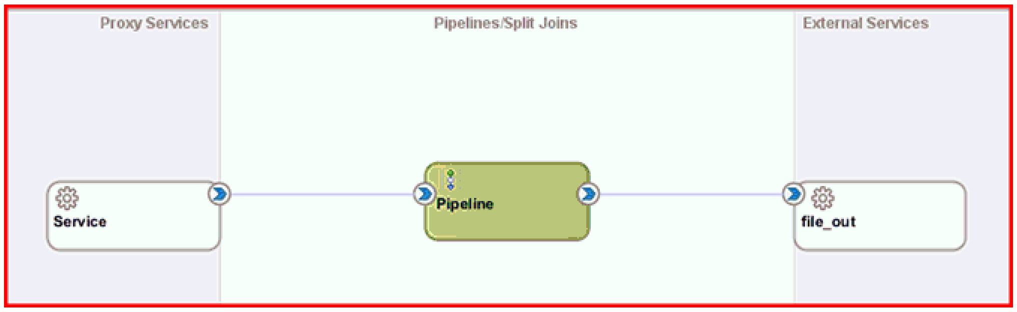
Once the OSB process is connected to the SOA Debugger, the following messages will be displayed in the Debugging log:

```
Debugger attempting to connect to remote process at iwdevora3.ibi.com 5004.
Debugger connected to remote process at iwdevora3.ibi.com 5004.
Debugger process virtual machine is SOA Debugger
```

4. Set the Breakpoints.

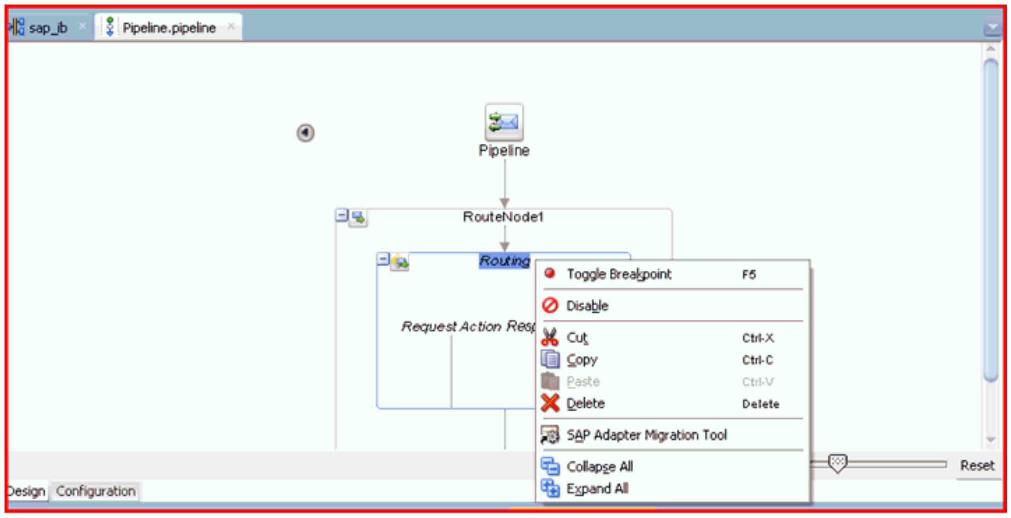
   a. Open the pipeline in its editor by double-clicking on the pipeline, as shown in Figure 9–60.

Figure 9–60 ProxyServicePipeline



   b. Expand the actions until you reach the node where the Breakpoint must be added. In this case, expand until you reach **Routing**. Right-click the **Routing** node and select **Toggle Breakpoint** from the context menu, as shown in Figure 9–61.

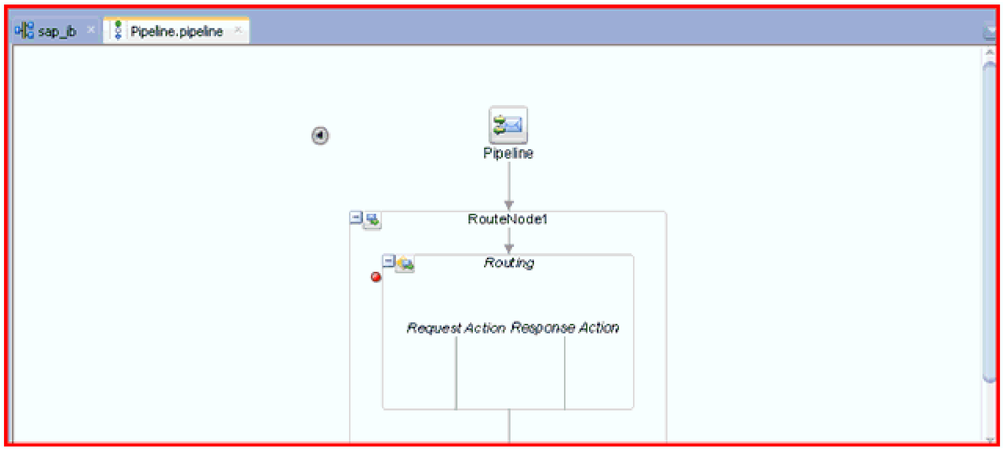
Figure 9–61 Toggle Breakpoint



Repeat the above step for each node to which you want to add a Breakpoint. In this example, a Breakpoint is set only for the **Routing** node.

A red icon appears next to the node to indicate that a Breakpoint has been set, as shown in Figure 9–62.

Figure 9–62 Red Icon for Set Breakpoint



Note: To disable a Breakpoint, right-click the node and select **Disable Breakpoint**.

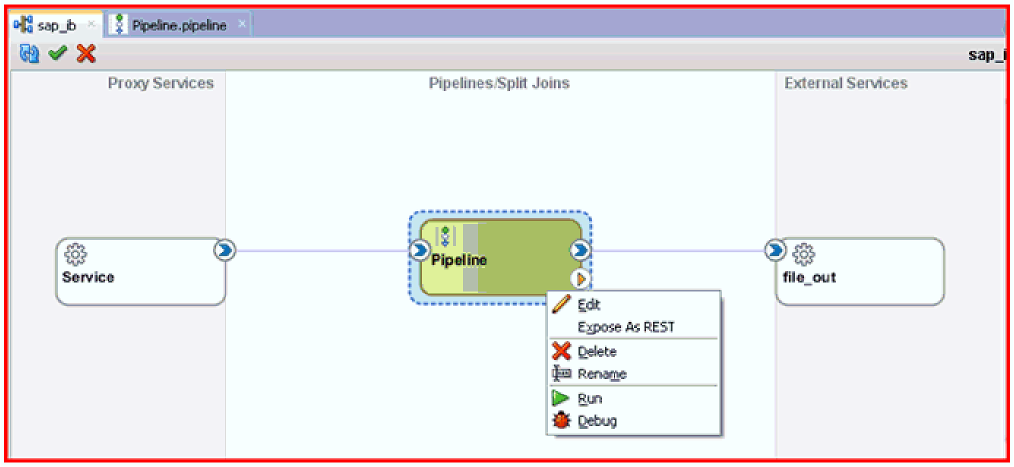
To remove a Breakpoint, right-click the node and select **Toggle Breakpoint** again.

---

5. Initiate Debugging.

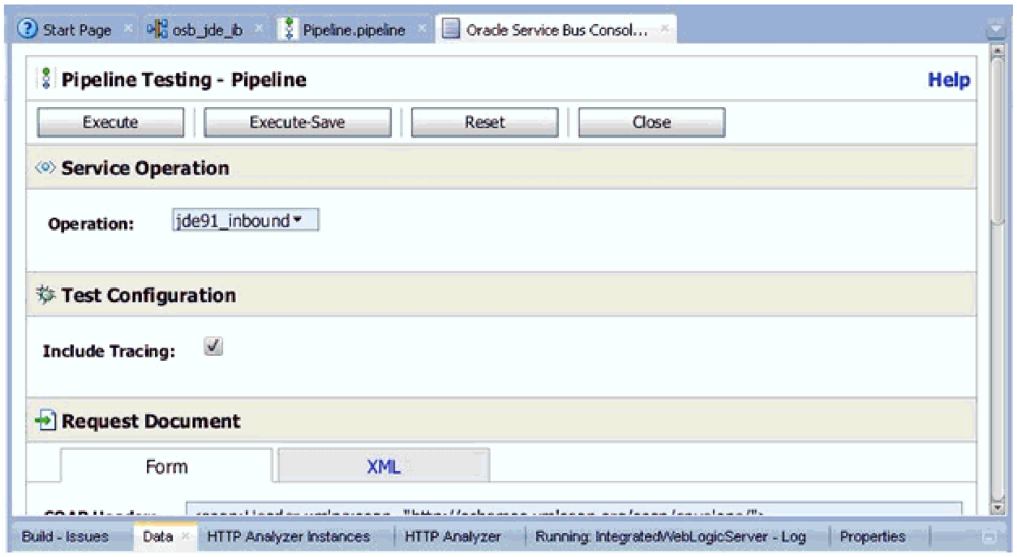
a. Right-click the pipeline in the Application Navigator, and select **Debug**, as shown in Figure 9–63.

Figure 9–63 Select Debug



The process is deployed to the integrated server and the Test Configuration pane will be displayed, as shown in Figure 9–64.

Figure 9–64 Test Configuration Pane

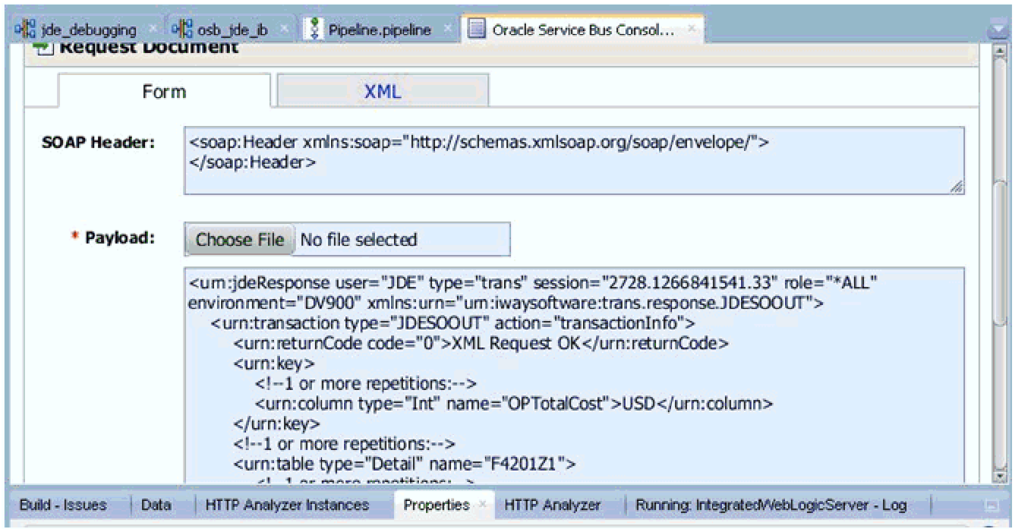


Note: If there is no domain currently running, then the Create Default Domain dialog will be displayed. Enter the connection information for the integrated server and then click **OK**. This process may take several minutes.

---

b. In the Test Configuration pane, enter the test data in the Request Document area, and configure any additional input as required, as shown in Figure 9–65.

Figure 9–65 Test Request Document



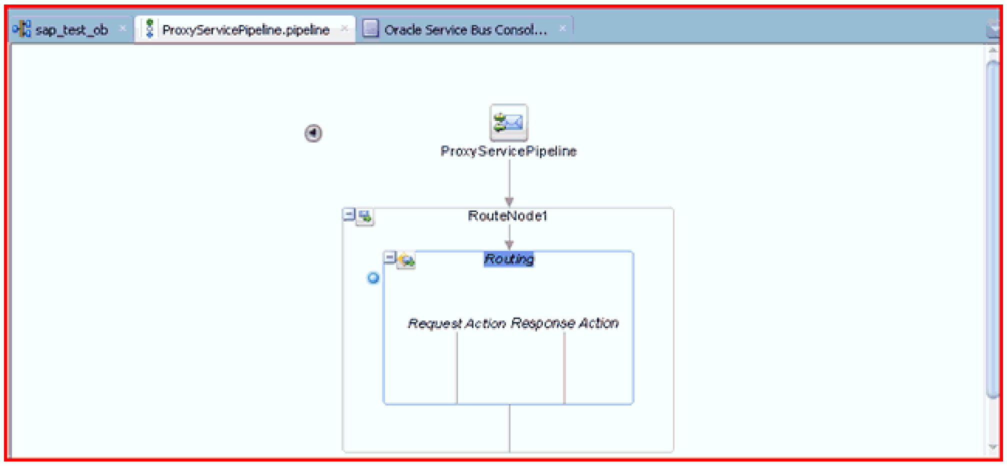
```
<soap:Header xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
</soap:Header>
```

```
<urn:jdeResponse user="JDE" type="trans" session="2728.1266841541.33" role="*ALL"
environment="DV900" xmlns:urn="urn:iwaysoftware:trans.response.JDESOOUT">
    <urn:transaction type="JDESOOUT" action="transactionInfo">
        <urn:returnCode code="0">XML Request OK</urn:returnCode>
        <urn:key>
            <!--1 or more repetitions:-->
            <urn:column type="Int" name="OPTotalCost">USD</urn:column>
        </urn:key>
        <!--1 or more repetitions:-->
        <urn:table type="Detail" name="F4201Z1">
```

c. Click **Execute**.

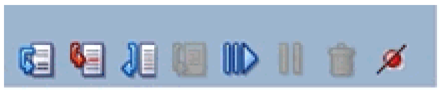
The Test Configuration pane executes the command, but the OSB process stops at the designated Breakpoint and blinks in a blue color, as shown in Figure 9–66.

Figure 9–66 OSB Process Stopped at Breakpoint


sap_test_ob
ProxyServicePipeline.pipeline
Oracle Service Bus Consol...
ProxyServicePipeline
RouteNode1
Routing
Request Action Response Action

d. Use the available Step options to step through the Debugging process, as shown in Figure 9–67.

Figure 9–67 Step Options



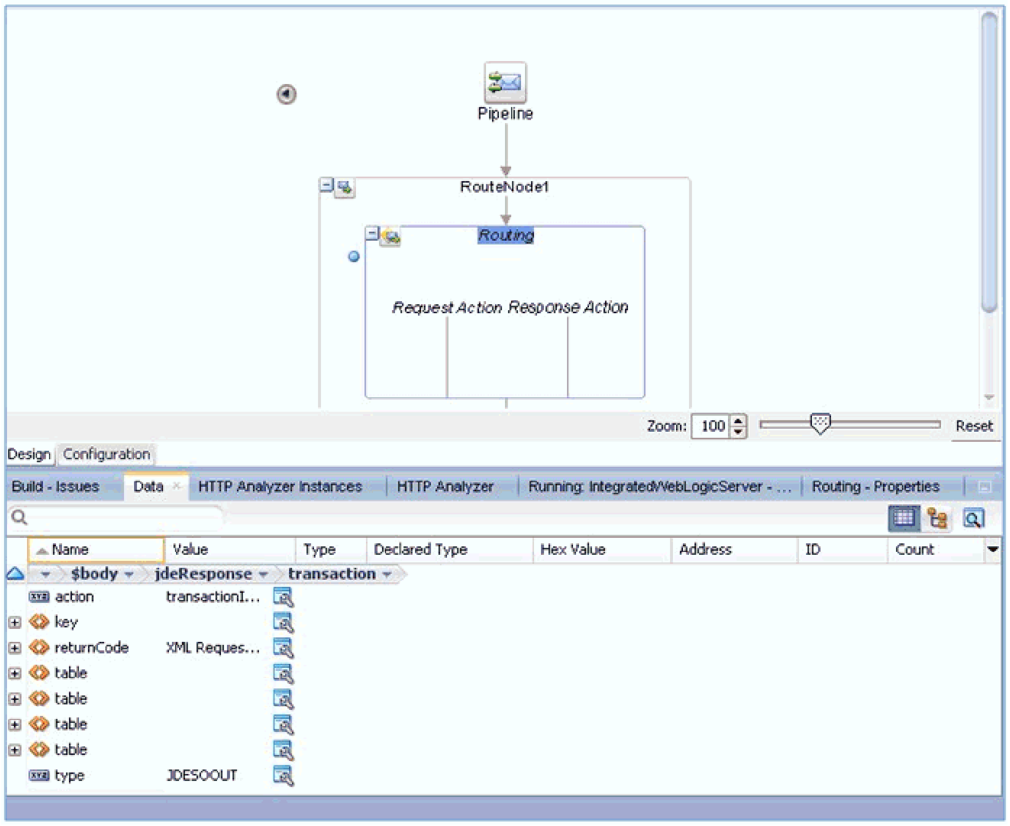
Icon	Description
🟥	Ends or detaches from a debugging session.
	Steps over a frame. This places you at the next Breakpoint (for example, the receive activity in the OSB process on which a Breakpoint was set. If there are no Breakpoints, it steps over all the frames and returns to the first frame. You can also press **F8** to step over a frame.
	Steps into the next valid location. This can be a new frame or the same frame, but in a different location. You can also press **F7** to step into a frame.
	Steps out of a frame. This option is only used to process an OSB scope or sequence activity. After completion of scope processing, it pauses at the next scope or activity in the process. You can also press **Shift-F7**.
	Resumes a step operation. You can also press **F9** to resume.

e. Use **Step Over** to complete the process execution

Note: Since there is only one Breakpoint in this example, using **Step Over** completes the process execution.

You will be able to see the response document displayed in the Data tab, as shown in Figure 9–68.

Figure 9–68 Data Tab Output

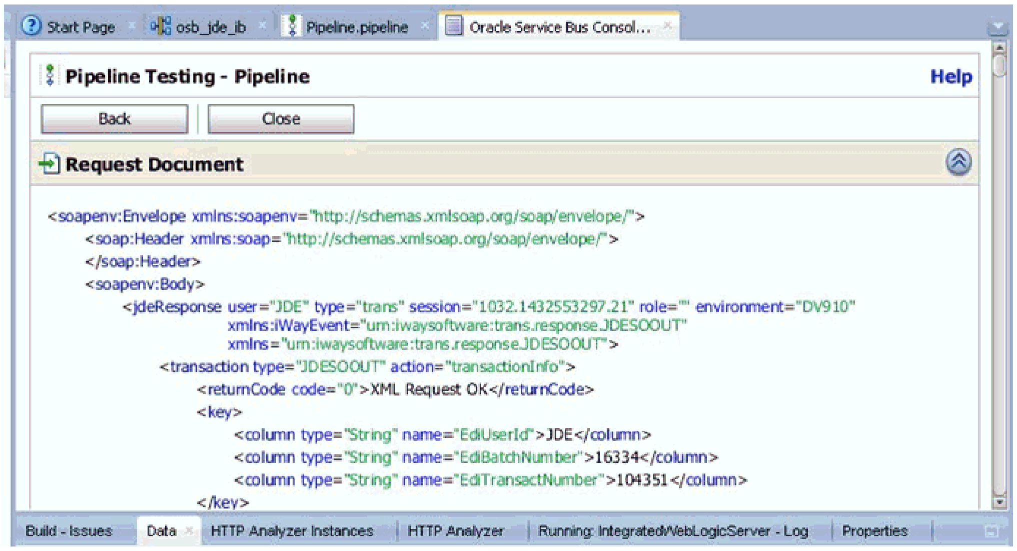


$body › jdeResponse › transaction

Name	Value	Type	Declared Type	Hex Value	Address	ID	Count
action	transactionI...						
key							
returnCode	XML Reques...						
table							
table							
table							
table							
type	JDESOOUT						

You will also be able to see the response in the Oracle Service Bus Console, as shown in Figure 9–69.

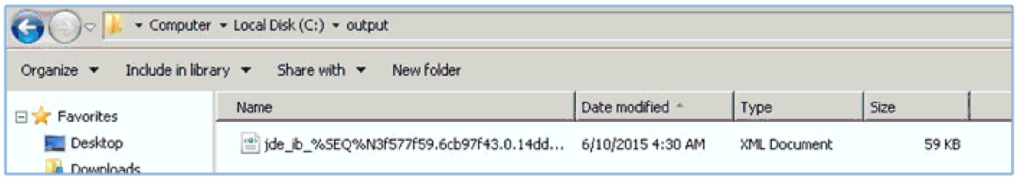
Figure 9–69 Oracle Service Bus Console



```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Header xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    </soap:Header>
    <soapenv:Body>
        <jdeResponse user="JDE" type="trans" session="1032.1432553297.21" role="" environment="DV910"
                    xmlns:iWayEvent="urn:iwaysoftware:trans.response.JDESOOUT"
                    xmlns="urn:iwaysoftware:trans.response.JDESOOUT">
            <transaction type="JDESOOUT" action="transactionInfo">
                <returnCode code="0">XML Request OK</returnCode>
                <key>
                    <column type="String" name="EdiUserId">JDE</column>
                    <column type="String" name="EdiBatchNumber">16334</column>
                    <column type="String" name="EdiTransactNumber">104351</column>
                </key>
```

The output will also be available in the configured output location, as show in Figure 9–70.

Figure 9–70 Configured Output Location

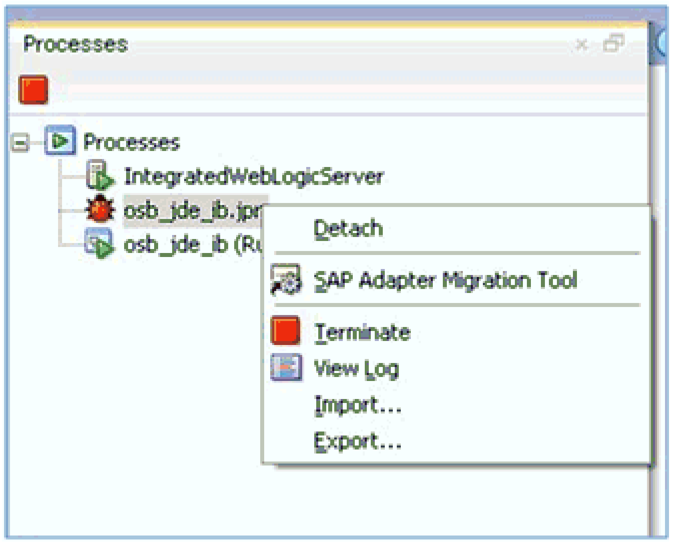


Name	Date modified	Type	Size
jde_ib_%SEQ%N3f577f59.6cb97f43.0.14dd...	6/10/2015 4:30 AM	XML Document	59 KB

6. End or detach the Debugging session.

a. Click **Window** and then **Processes**. Right click on the process in the Processes tab and select **Detach** or **Terminate**, as shown in Figure 9–71.

Figure 9–71 Detach Debugging Session



b. Select one of the following options:

   Detach - Removes the SOA debugger without ending the debugging process.

   Terminate - Ends the debugging process.

9.4 Exception Filter

This section describes how to configure exception filter functionality for the Oracle Application Adapter for J.D. Edwards OneWorld and includes a sample testing scenario.

This section contains the following topic:

- Section 9.4.1, "Configuring the Exception Filter"

The exception filter is supported only for outbound processes that use J2CA configurations. This feature is not supported for BSE configurations and inbound processes that use J2CA configurations.

The exception filter uses the `com.ibi.afjca.oracle.AdapterExceptionFilter` class to filter the generated exceptions. This class filters the exceptions and categorizes them into the following categories:

- PCRetriableResourceException
- PCResourceException

The following exceptions are represented in the fault policies file:

- PCRetriableResourceException - A remote fault.
- PCResourceException - A binding fault.

9.4.1 Configuring the Exception Filter

Exception filter configuration consists of the following steps and topics:

1. Section 9.4.1.1, "Generating a WSDL File"
2. Section 9.4.1.2, "Creating a BPEL process With Exception Filter Functionality"
3. Section 9.4.1.3, "Creating Fault Policies and Fault Binding Files"
4. Section 9.4.1.4, "Adjusting for Known Deployment Issues With 12c"
5. Section 9.4.1.5, "Deploying and Testing the BPEL Process With Exception Filter Functionality"

9.4.1.1 Generating a WSDL File

To generate the WSDL file:

1. Open Application Explorer and create a J2CA configuration.

   For more information, see "Creating a Configuration for J2CA" on page 2-3.

2. Create a target for the PeopleSoft adapter and then connect to the target.

   For more information, see "Establishing a Connection (Target) for J.D. Edwards OneWorld" on page 2-5.

3. Generate a WSDL for the appropriate object.

   For more information, see "Generating WSDL (J2CA Configurations Only)" on page 2-11.

9.4.1.2 Creating a BPEL process With Exception Filter Functionality

To create a BPEL process with exception filter functionality:

1. Open JDeveloper and create a new SOA application.

   For more information, see Section 4.4.2, "Creating an Empty Composite for SOA" on page 4-9.

2. Create a new SOA project (for example, Exception_Filter).

3. Create a third party adapter service component.

   For more information, see Section 4.4.3.1, "Configuring a Third Party Adapter Service Component" on page 4-11.

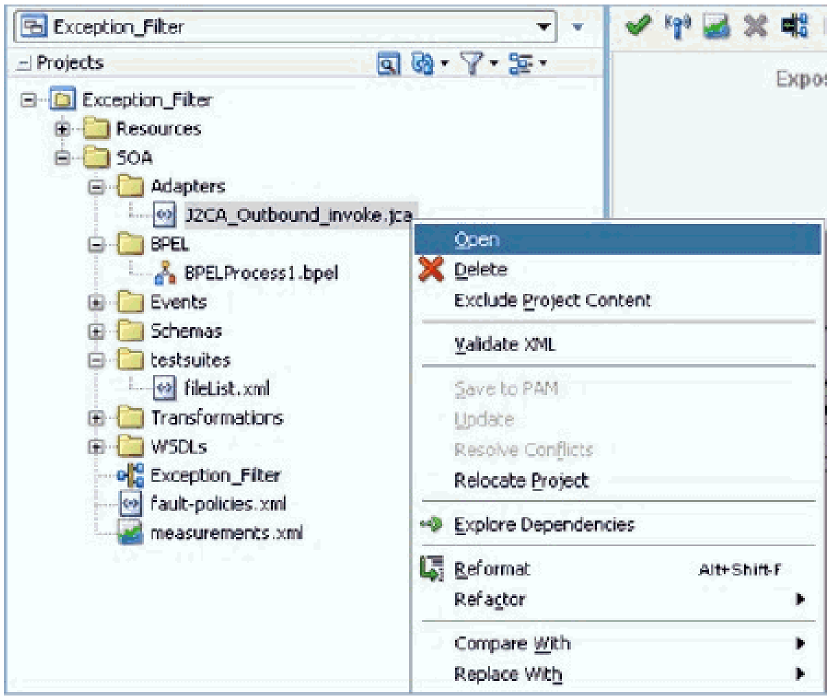
   Once the third party adapter service component is created, the WSDL file (with corresponding schemas and JCA file) is imported to the JDeveloper project.

   For more information, see Section 4.4.3, "Defining a BPEL Outbound Process" on page 4-11.

4. Modify the imported JCA file.

   a. Right-click the imported JCA file and select **Open** from the menu, as shown in Figure 9–72.

Figure 9–72 Application Navigator Tab



   b. In the `<interaction-spec>` element, add the `ExceptionFilter` property. For example:

```
<interaction-spec className="com.ibi.afjca.cci.IWAFInteractionSpec">
<property name="FunctionName" value="PROCESS"/><property
name="ExceptionFilter"
value="com.ibi.afjca.oracle.AdapterExceptionFilter"/></interaction-spec>
```

   c. Save the modified JCA file.

5. Once the third party adapter service component is created and the JCA file is modified, continue with the remainder of the BPEL process creation.

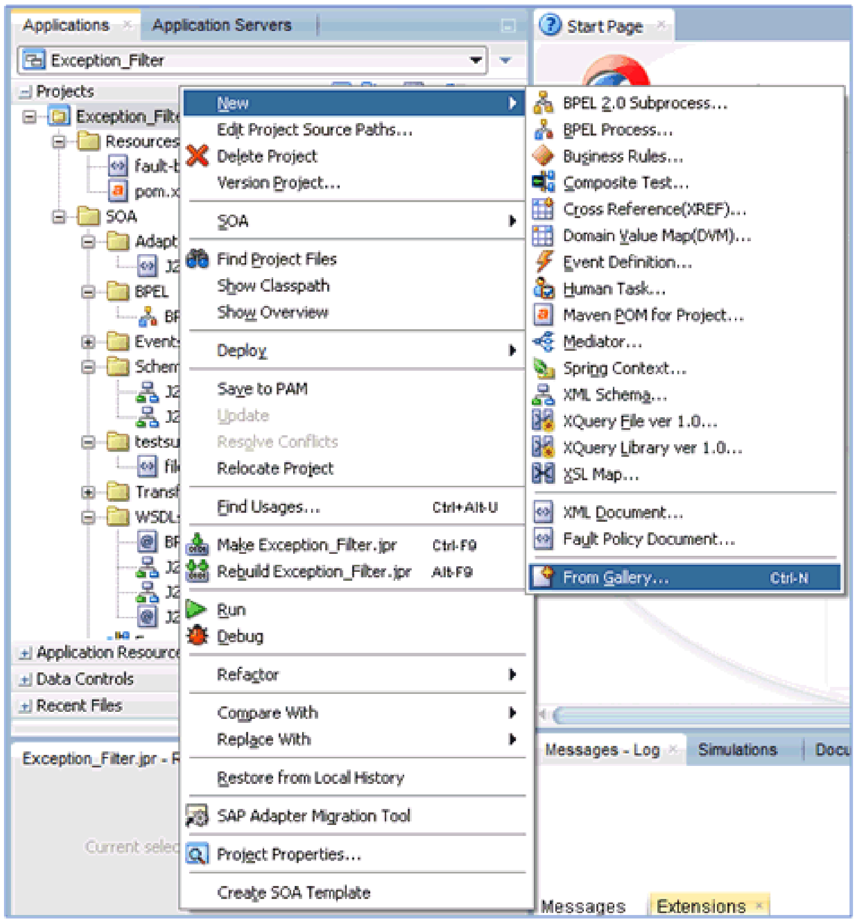
   For more information, see Section 4.4.3, "Defining a BPEL Outbound Process" on page 4-11.

9.4.1.3 Creating Fault Policies and Fault Binding Files

To create fault binding files:

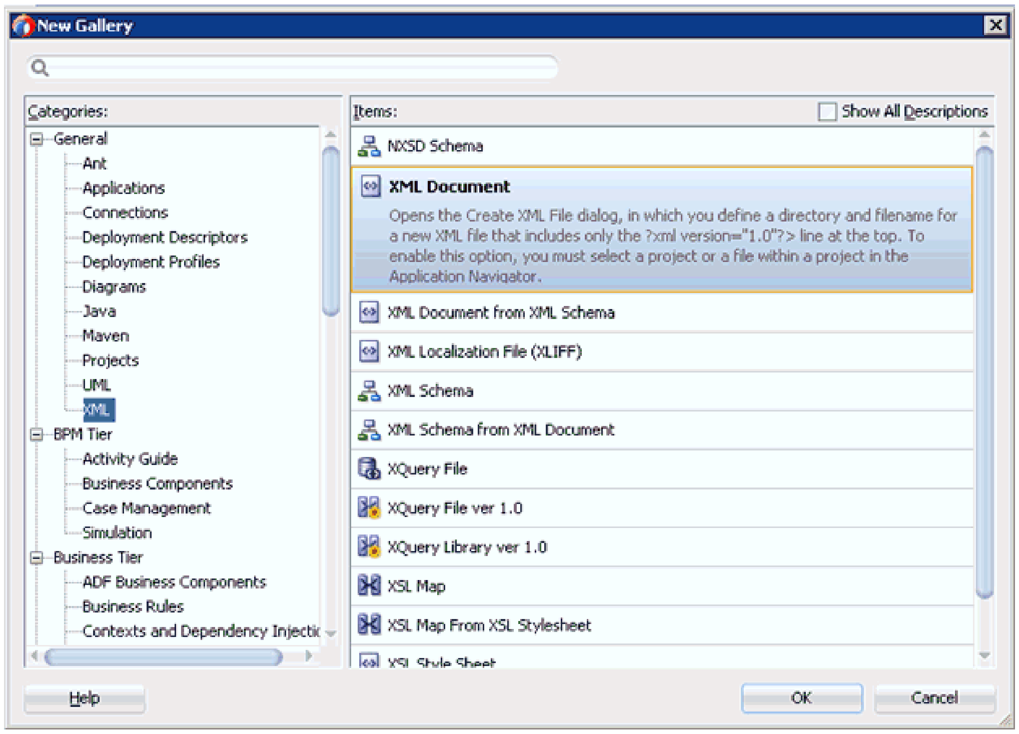
1. Right-click the created SOA project (for example, Exception_Filter), select **New**, and then click **From Gallery**, as shown in Figure 9–73.

Figure 9–73 Applications Tab



The New Gallery dialog is displayed. Under the General category, click **XML**, as shown in Figure 9–74.

Figure 9–74 New Gallery Dialog

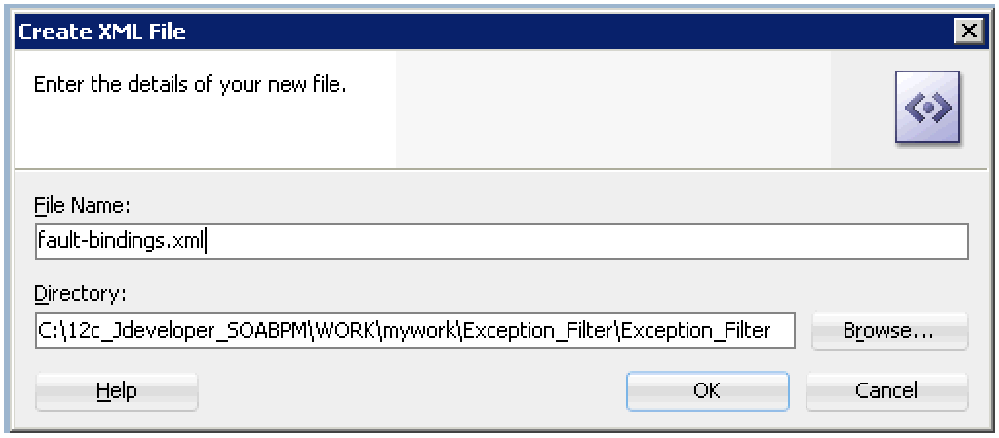


Opens the Create XML File dialog, in which you define a directory and filename for a new XML file that includes only the ?xml version="1.0"?> line at the top. To enable this option, you must select a project or a file within a project in the Application Navigator.

2. Select **XML Document** under Items and then click **OK**.

   The Create XML File dialog is displayed, as shown in Figure 9–75.

Figure 9–75 Create XML File Dialog



Enter the details of your new file.

File Name: fault-bindings.xml

Directory: C:\12c_Jdeveloper_SOABPM\WORK\mywork\Exception_Filter\Exception_Filter

3. In the File Name field, type **fault-bindings.xml** and click **OK**.
4. Add the appropriate fault binding functions in the **fault-bindings.xml** file.

   To view a sample **fault-bindings.xml** file, see "Sample Fault-Bindings.xml File" on page 9-49.

Note: The parameter in the `<name>` element is the name of the created BPEL process.

---

5. Save the **fault-bindings.xml** file.

Sample Fault-Bindings.xml File

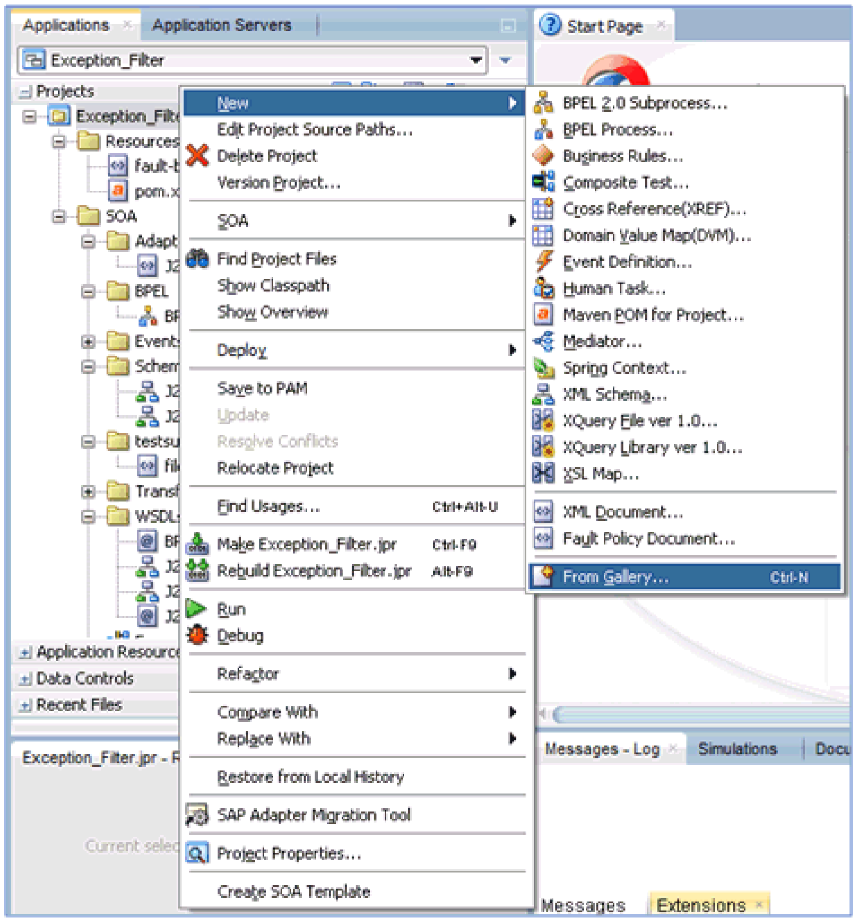
```
<?xml version="1.0" encoding="UTF-8" ?>
<faultPolicyBindings version="2.0.1"
xmlns="http://schemas.oracle.com/bpel/faultpolicy"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<component faultPolicy="bpelFaultHandling">
<name>BPELProcess1</name>
</component>

</faultPolicyBindings>
```

Creating Fault Policies Files

To create fault policies files:

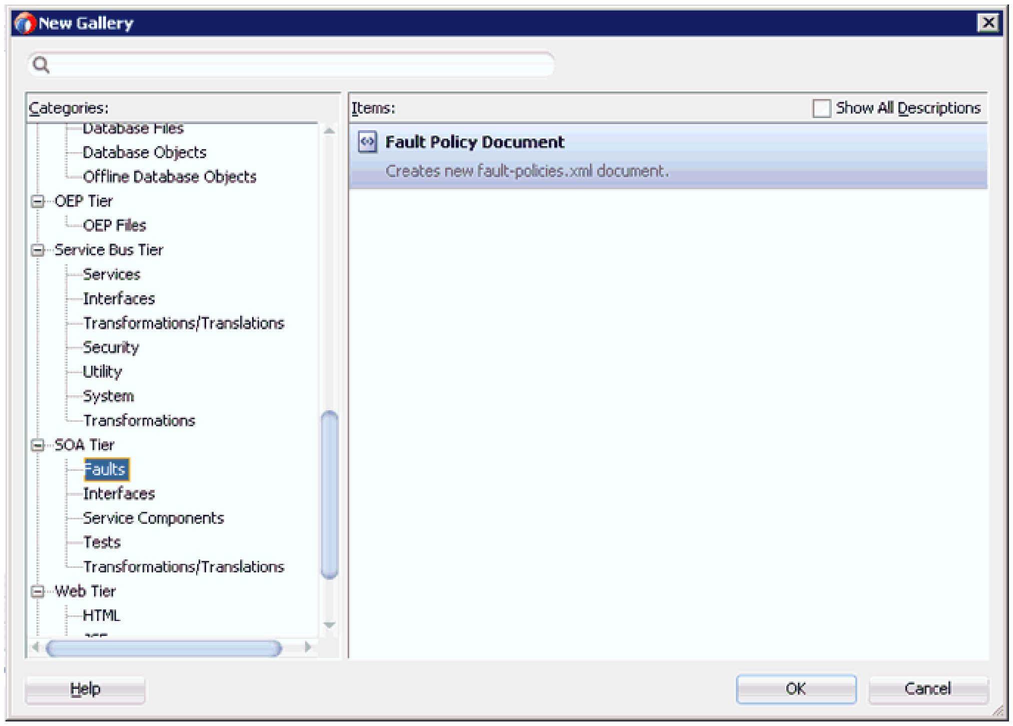
1. Right-click the created SOA project (for example, Exception_Filter), select **New**, and then click **From Gallery**, as shown in Figure 9–76.

Figure 9–76 Applications Tab



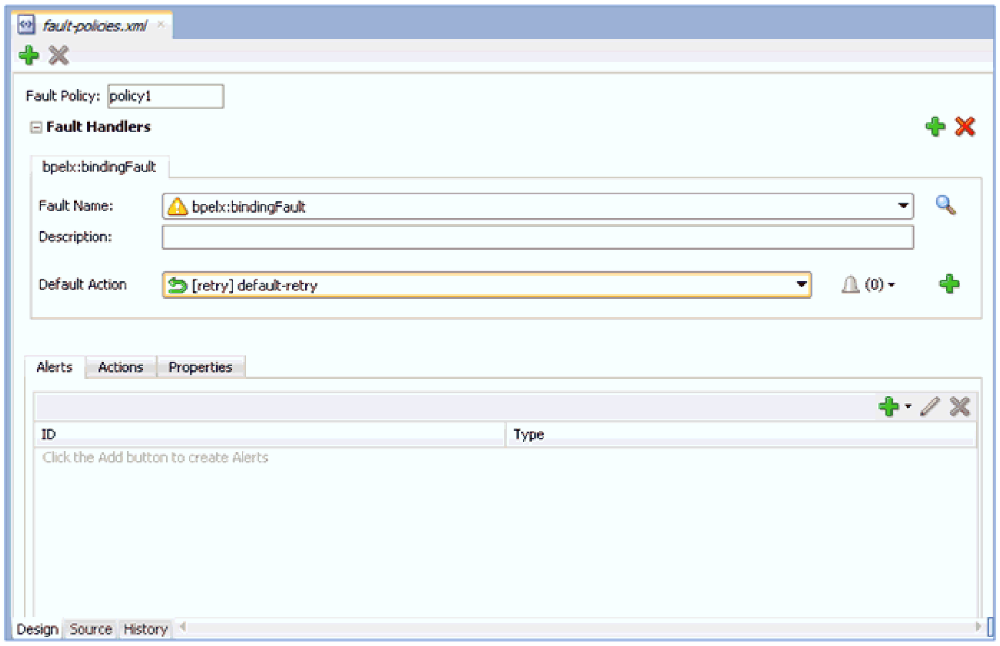
The New Gallery dialog is displayed. Under the SOA Tier category, select **Faults**, as shown in Figure 9–77.

Figure 9–77 New Gallery Dialog



2. Select **Fault Policy Document** under Items and then click **OK**.
3. In the fault-policies.xml tab, select **bpelx:bindingFault** from the Fault Name drop-down list and **[retry] default-retry** from the Default Action drop-down list, as shown in Figure 9–78.

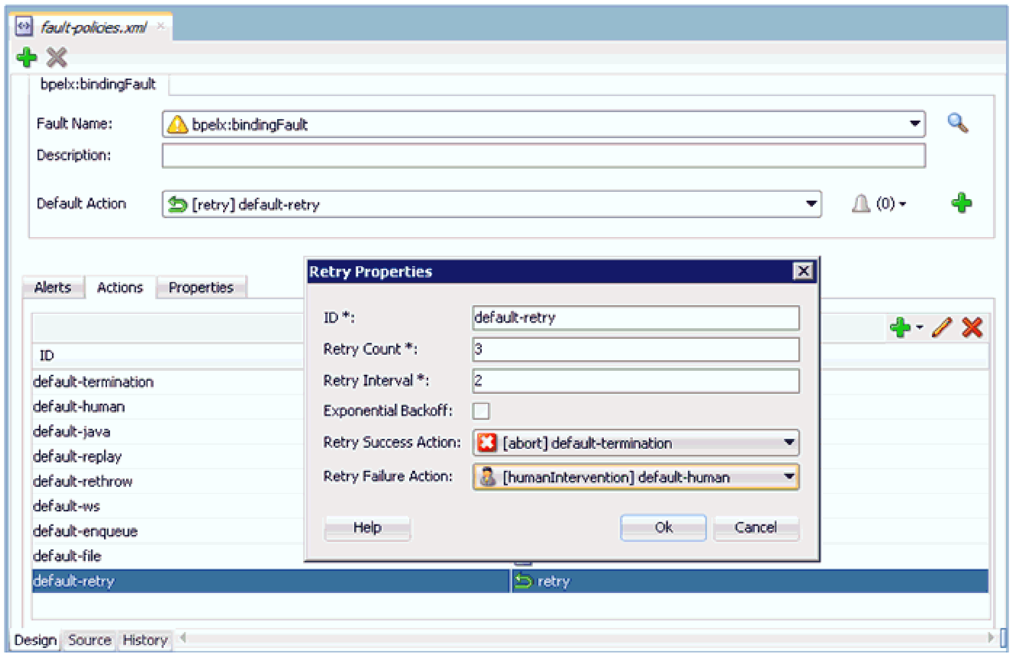
Figure 9–78 Fault-policies.xml Tab



4. Click the **Actions** tab and then double-click **default-retry**.

The Retry Properties dialog box is displayed, as shown in Figure 9–79.

Figure 9–79 Retry Properties Dialog Box



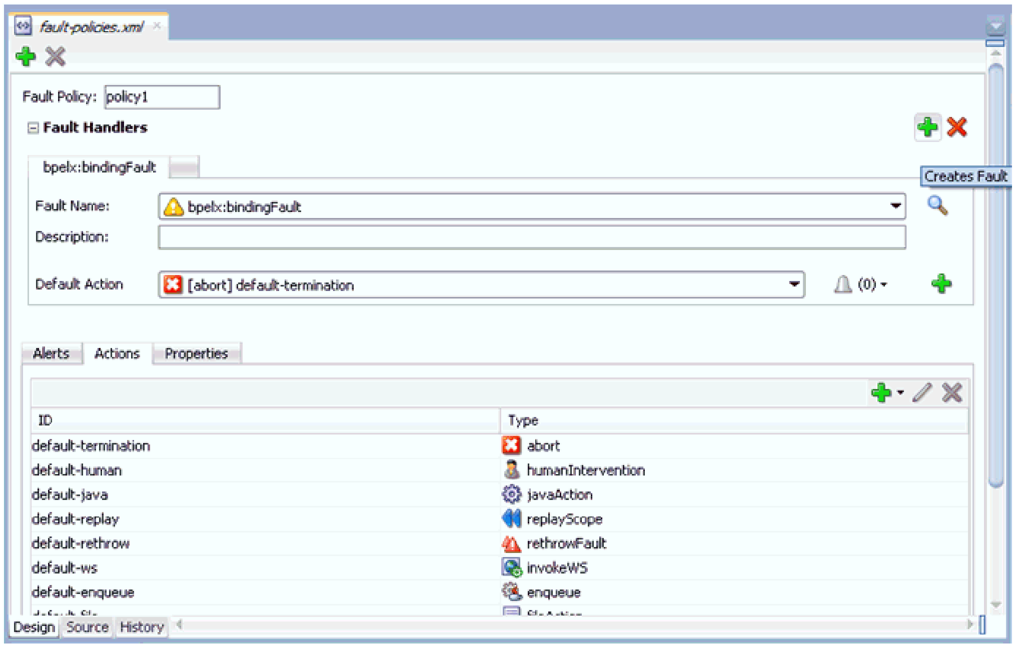
Fault Name: bpelx:bindingFault
Description:
Default Action [retry] default-retry

ID	
default-termination	
default-human	
default-java	
default-replay	
default-rethrow	
default-ws	
default-enqueue	
default-file	
default-retry	retry

ID *: default-retry
Retry Count *: 3
Retry Interval *: 2
Exponential Backoff:
Retry Success Action: [abort] default-termination
Retry Failure Action: [humanIntervention] default-human

5. Select **[abort] default-termination** from the Retry Success Action drop-down list and **[humanIntervention] default-human** from the Retry Failure Action drop-down list.
6. Click **OK**.
7. Click **Add** to create another fault handler, as shown in Figure 9–80.

Figure 9–80 Fault-policies.xml Tab

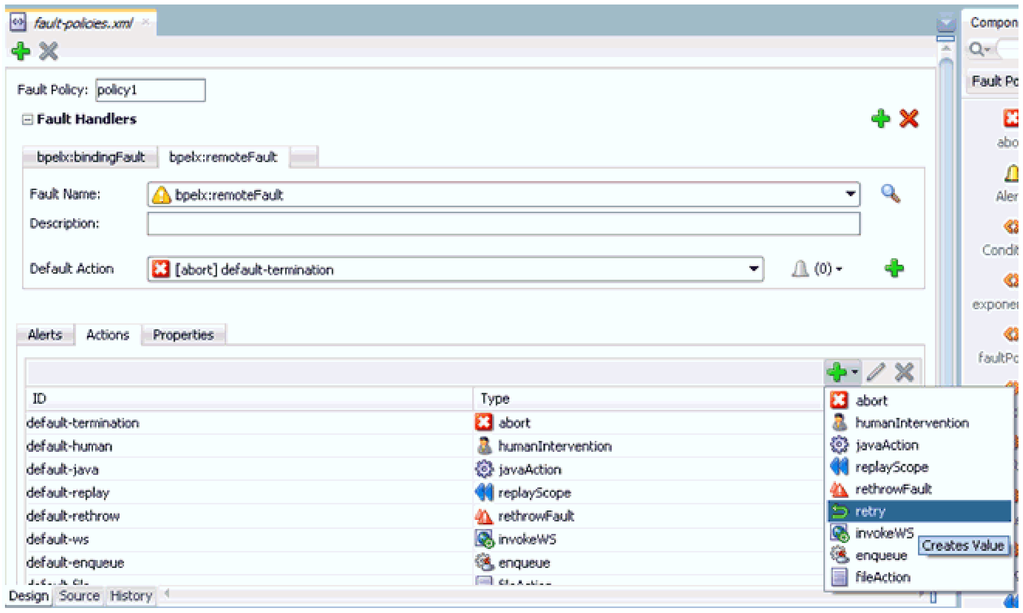


Fault Policy: policy1
Fault Name: bpelx:bindingFault
Description:
Default Action [abort] default-termination

ID	Type
default-termination	abort
default-human	humanIntervention
default-java	javaAction
default-replay	replayScope
default-rethrow	rethrowFault
default-ws	invokeWS
default-enqueue	enqueue

8. In the fault-policies.xml tab, select **bpelx:remoteFault** from the Fault Name drop-down list and **[abort] default-termination** from the Default Action drop-down list.
9. In the Actions tab, click **Add** and then select **retry**, as shown in Figure 9–81.

Figure 9–81 Actions Tab



Fault Policy: policy1

Fault Handlers

bpelx:bindingFault | bpelx:remoteFault

Fault Name: bpelx:remoteFault

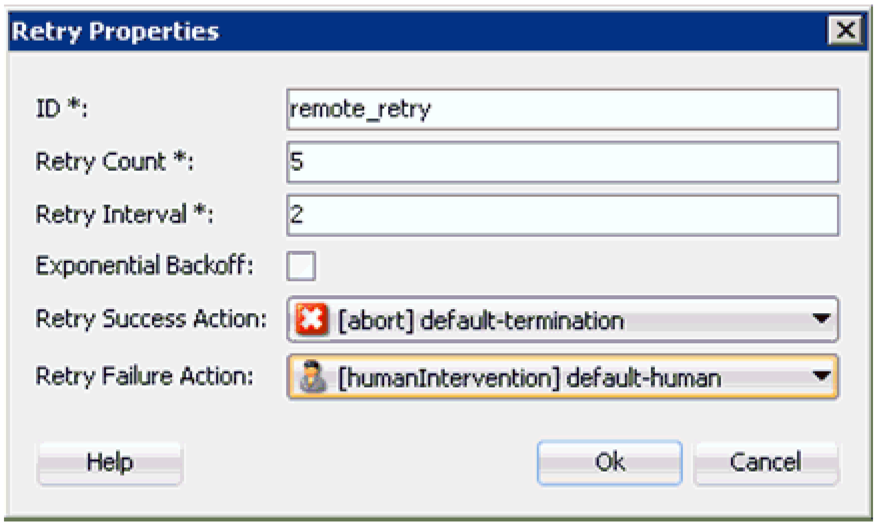
Description:

Default Action [abort] default-termination

ID	Type
default-termination	abort
default-human	humanIntervention
default-java	javaAction
default-replay	replayScope
default-rethrow	rethrowFault
default-ws	invokeWS
default-enqueue	enqueue

The Retry Properties dialog is displayed, as shown in Figure 9–82.

Figure 9–82 Retry Properties Dialog Box



ID *: remote_retry

Retry Count *: 5

Retry Interval *: 2

Exponential Backoff: ☐

Retry Success Action: [abort] default-termination

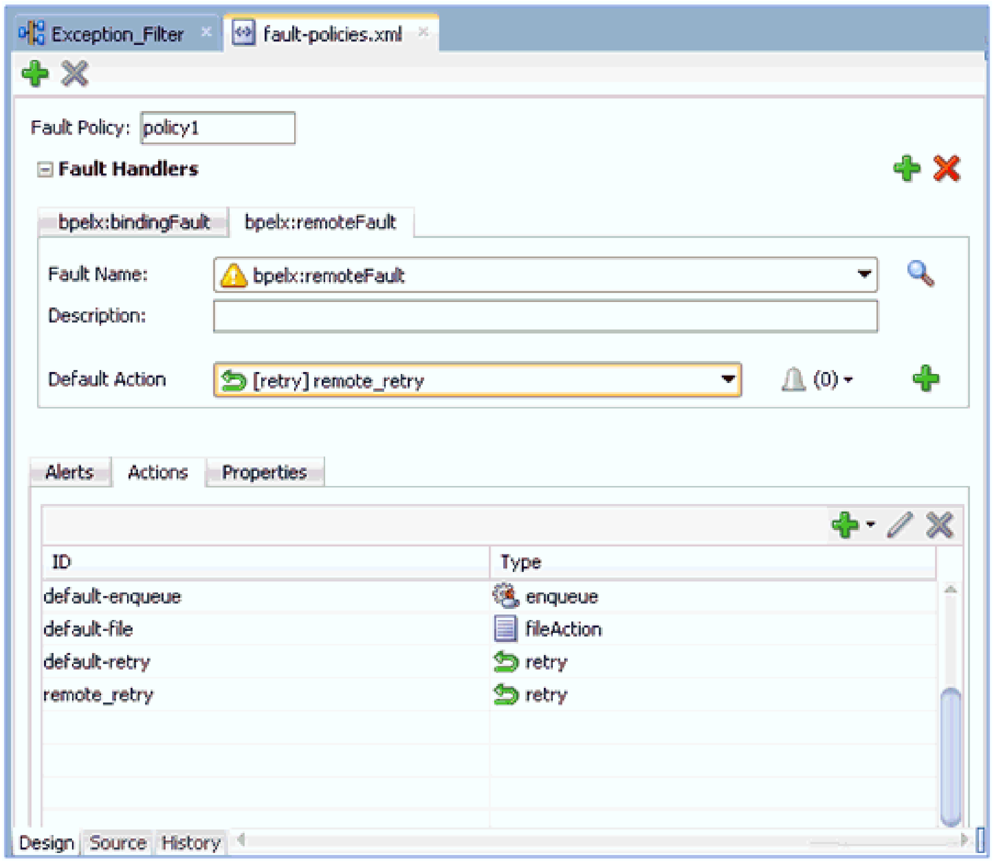
Retry Failure Action: [humanIntervention] default-human

10. Provide values for the ID, Retry Count, and Retry Interval fields.
11. Select **[abort] default-termination** from the Retry Success Action drop-down list and **[humanIntervention] default-human** from the Retry Failure Action drop-down list.
12. Click **OK**.

The created Retry ID will be listed under the Actions tab.

From the Default Action drop-down list, select the newly created Retry ID (for example, remote_retry) as shown in Figure 9–83.

Figure 9–83 Fault-policies.xml Tab



Fault Policy: policy1

Fault Name: bpelx:remoteFault

Description:

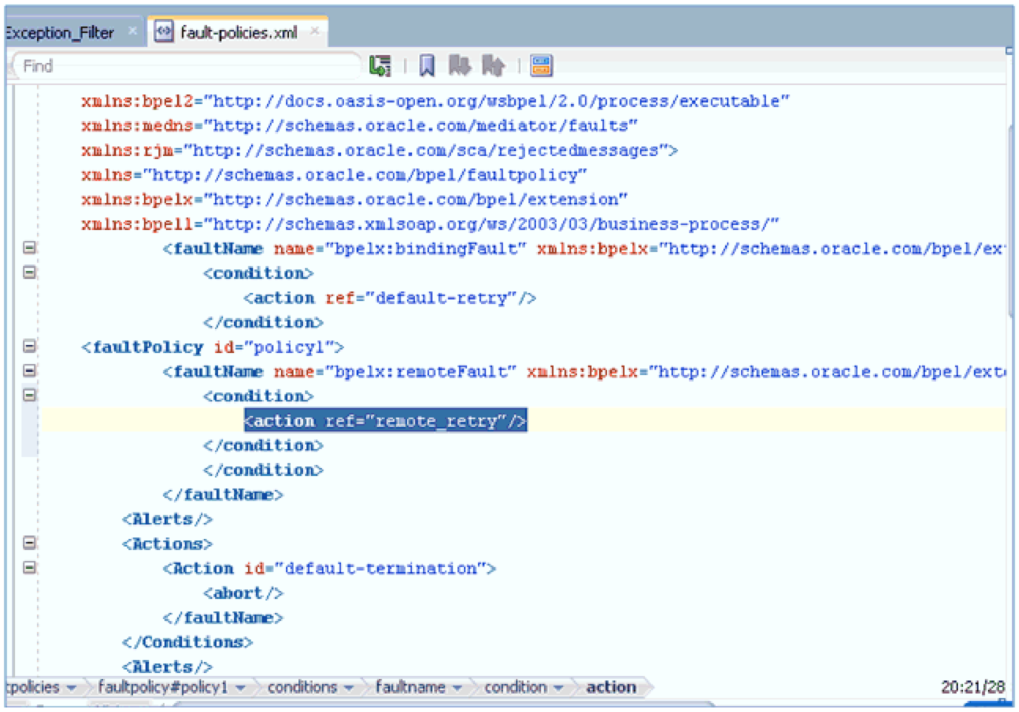
Default Action [retry] remote_retry (0)

ID	Type
default-enqueue	enqueue
default-file	fileAction
default-retry	retry
remote_retry	retry

13. Click **Save All**.

14. Click the **Source** tab to verify that the fault polices are added properly, as shown in Figure 9–84.

Figure 9–84 Source Tab

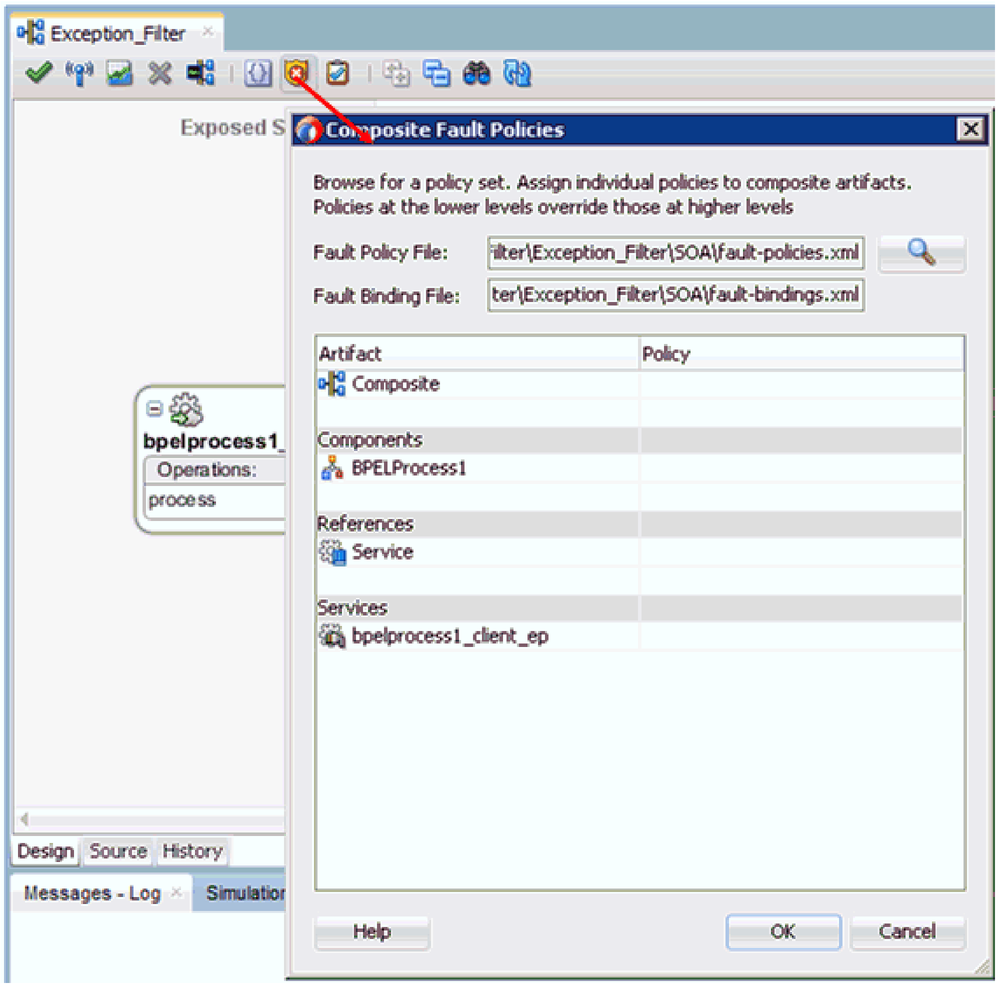


```
xmlns:bpel2="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
xmlns:medns="http://schemas.oracle.com/mediator/faults"
xmlns:rjm="http://schemas.oracle.com/sca/rejectedmessages">
xmlns="http://schemas.oracle.com/bpel/faultpolicy"
xmlns:bpelx="http://schemas.oracle.com/bpel/extension"
xmlns:bpel1="http://schemas.xmlsoap.org/ws/2003/03/business-process/"
        <faultName name="bpelx:bindingFault" xmlns:bpelx="http://schemas.oracle.com/bpel/ex
            <condition>
                <action ref="default-retry"/>
            </condition>
<faultPolicy id="policy1">
        <faultName name="bpelx:remoteFault" xmlns:bpelx="http://schemas.oracle.com/bpel/ext
            <condition>
                <action ref="remote_retry"/>
            </condition>
            </condition>
        </faultName>
    <Alerts/>
    <Actions>
        <Action id="default-termination">
            <abort/>
        </faultName>
    </Conditions>
    <Alerts/>
```

15. Double-click the **Exception_Filter** project and then click **Edit Composite Fault Policies**.

The Composite Fault Policies window is displayed. Ensure that the Fault Policy and the fault-bindings are selected properly, as shown in Figure 9–85.

Figure 9–85 Composite Fault Policies Window



Browse for a policy set. Assign individual policies to composite artifacts.
Policies at the lower levels override those at higher levels

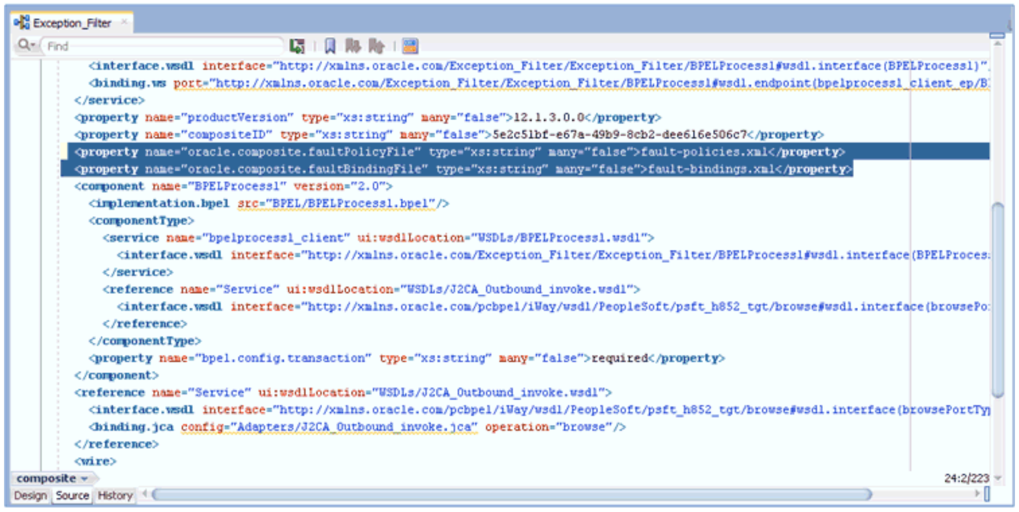
Fault Policy File: ilter\Exception_Filter\SOA\fault-policies.xml

Fault Binding File: ter\Exception_Filter\SOA\fault-bindings.xml

Artifact	Policy
Composite	
Components	
BPELProcess1	
References	
Service	
Services	
bpelprocess1_client_ep	

16. Click **Save All**.
17. Click the **Source** tab to verify that the *`fault-bindings.xml`* and *`fault-policies.xml`* files are added properly, as shown in Figure 9–86.

Figure 9–86 Source Tab



```
      <interface.wsdl interface="http://xmlns.oracle.com/Exception_Filter/Exception_Filter/BPELProcess1#wsdl.interface(BPELProcess1)".
      <binding.ws port="http://xmlns.oracle.com/Exception_Filter/Exception_Filter/BPELProcess1#wsdl.endpoint(bpelprocess1_client_ep/B
    </service>
    <property name="productVersion" type="xs:string" many="false">12.1.3.0.0</property>
    <property name="compositeID" type="xs:string" many="false">5e2c51bf-e67a-49b9-8cb2-dee616e506c7</property>
    <property name="oracle.composite.faultPolicyFile" type="xs:string" many="false">fault-policies.xml</property>
    <property name="oracle.composite.faultBindingFile" type="xs:string" many="false">fault-bindings.xml</property>
    <component name="BPELProcess1" version="2.0">
      <implementation.bpel src="BPEL/BPELProcess1.bpel"/>
      <componentType>
        <service name="bpelprocess1_client" ui:wsdlLocation="WSDLs/BPELProcess1.wsdl">
          <interface.wsdl interface="http://xmlns.oracle.com/Exception_Filter/Exception_Filter/BPELProcess1#wsdl.interface(BPELProces
        </service>
        <reference name="Service" ui:wsdlLocation="WSDLs/J2CA_Outbound_invoke.wsdl">
          <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/iWay/wsdl/PeopleSoft/psft_h852_tgt/browse#wsdl.interface(browsePo
        </reference>
      </componentType>
      <property name="bpel.config.transaction" type="xs:string" many="false">required</property>
    </component>
    <reference name="Service" ui:wsdlLocation="WSDLs/J2CA_Outbound_invoke.wsdl">
      <interface.wsdl interface="http://xmlns.oracle.com/pcbpel/iWay/wsdl/PeopleSoft/psft_h852_tgt/browse#wsdl.interface(browsePortTy
      <binding.jca config="Adapters/J2CA_Outbound_invoke.jca" operation="browse"/>
    </reference>
    <wire>
```

9.4.1.4 Adjusting for Known Deployment Issues With 12*c*

For more information on how to adjust for known deployment issues with 12c, see Section 4.4.3.3, "Adjusting for Known Deployment Issues With 12c" on page 4-26.

9.4.1.5 Deploying and Testing the BPEL Process With Exception Filter Functionality

To deploy and test the BPEL process with exception filter functionality:

1. Deploy the created BPEL process.

   For more information, see Section 4.4.4, "Deploying the BPEL Outbound Process" on page 4-28.

2. Simulate a communication error by disconnecting the system (where the servers are running) from the network.

3. Invoke the deployed BPEL process with a valid input.

   For more information, see Section 4.4.5, "Invoking the Input XML Document in the Oracle Enterprise Manager Console" on page 4-31.

4. Select the process ID.

   You can observe the BPEL process being retried or aborted based on the configuration of the **fault-policies.xml** file.

9.5 Credential Mapping for Oracle SOA Suite (BPEL, Mediator, or BPM)

This section describes how to configure credential mapping functionality for the Oracle Application Adapter for J.D. Edwards OneWorld in a configuration that uses Oracle SOA Suite (BPEL, Mediator, or BPM). A sample testing scenario is also included. This section contains the following topic:

- Section 9.5.1, "Configuring Credential Mapping"

Credential mapping is supported only for outbound processes that use J2CA configurations. This feature is not supported for BSE configurations and inbound processes that use J2CA configurations.

Note: The J2CA connector is common to all four application adapters (SAP R/3, PeopleSoft, Siebel, and J.D. Edwards OneWorld). If credential mapping is required, then ensure that only one application adapter is used in a particular instance. For example, in one adapter instance only the J.D. Edwards OneWorld application adapter can be used. Credential mapping cannot be configured at the individual adapter level. If you require the use of credential mapping for two adapters, then both adapters must be running in two independent adapter instances.

To pass user credentials to the J2CA resource adapter, create a credential map from the Oracle WebLogic Server user credentials to the EIS user credentials (J.D. Edwards OneWorld adapter). Then associate a credential policy with a BPEL, Mediator, or BPM Web service and invoke the Web service using Oracle WebLogic Server user credentials. These credentials are mapped to the EIS user credentials and then passed to the J2CA container, which uses them to connect with the EIS adapter (J.D. Edwards OneWorld).

9.5.1 Configuring Credential Mapping

This section discusses configuring credential mapping, and consists of the following steps and topics:

1. Deploy the adapter.

   For more information, see Chapter 3, "Oracle WebLogic Server Deployment and Integration".

2. Associate Oracle WebLogic Server credentials with EIS credentials.

   For more information, see Section 9.5.1.1, "Associating Oracle WebLogic Server Credentials With EIS Credentials" on page 9-57.

3. Generate a WSDL file.

   For more information, see Section 9.5.1.2, "Generating a WSDL File" on page 9-60.

4. Create and deploy an outbound process.

   For more information, see Section 9.5.1.3, "Creating and Deploying an Outbound Process" on page 9-60.

5. Invoke and verify that the EIS credentials have passed.

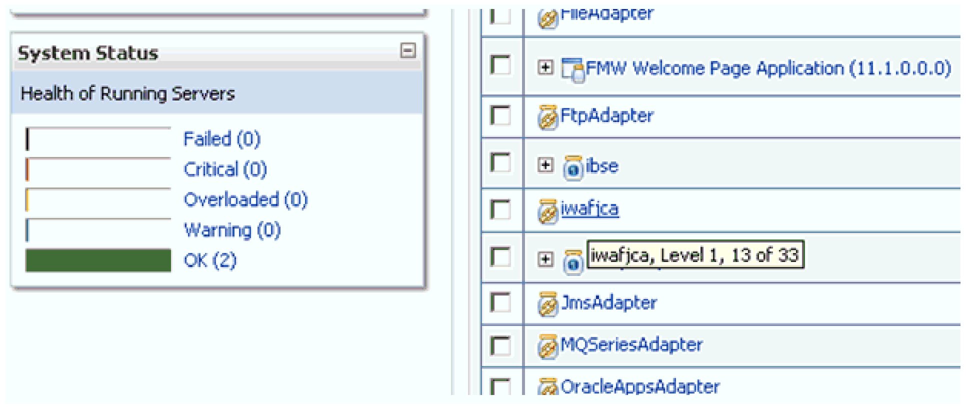
   For more information, see Section 9.5.1.4, "Verifying the EIS Credentials" on page 9-61.

9.5.1.1 Associating Oracle WebLogic Server Credentials With EIS Credentials

To associate Oracle WebLogic Server credentials with EIS credentials:

1. Log in to the Oracle WebLogic Server Administration Console.
2. In the Domain Structure section in the left pane, click **Deployments**, as shown in Figure 9–87.

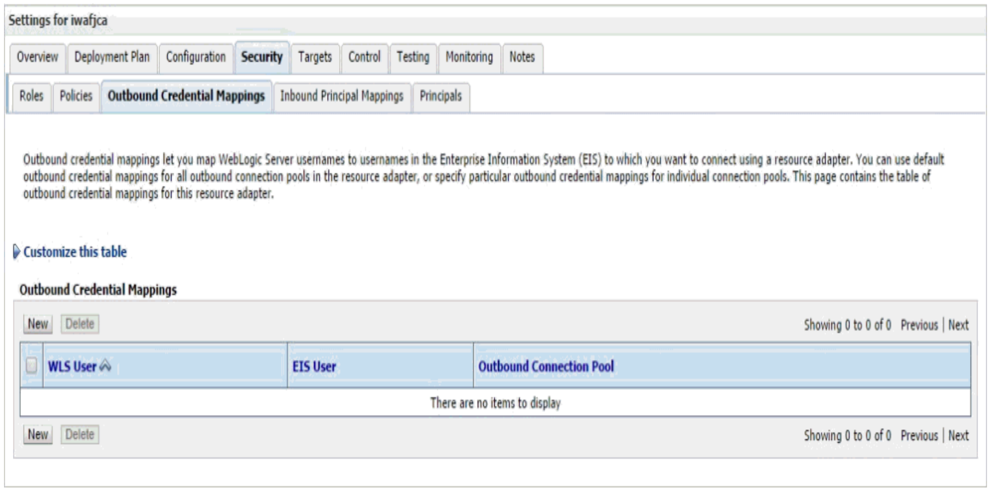
Figure 9–87 Deployments Page



3. Click the **iwafjca** resource adapter.

   The Settings for iwafjca page is displayed, as shown in Figure 9–88.

Figure 9–88 Settings for iwafjca Page



Outbound credential mappings let you map WebLogic Server usernames to usernames in the Enterprise Information System (EIS) to which you want to connect using a resource adapter. You can use default outbound credential mappings for all outbound connection pools in the resource adapter, or specify particular outbound credential mappings for individual connection pools. This page contains the table of outbound credential mappings for this resource adapter.

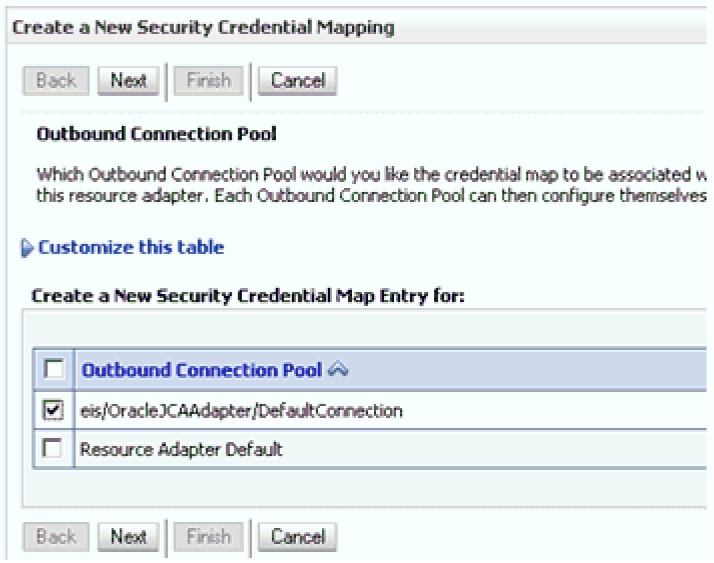
Outbound Credential Mappings

WLS User	EIS User	Outbound Connection Pool
There are no items to display		

4. Click the **Outbound Credential Mappings** tab under the Security tab, and then click **New**.

   The Create a New Security Credential Mapping page is displayed, as shown in Figure 9–89.

Figure 9–89 Create a New Security Credential Mapping Page


Create a New Security Credential Mapping

Outbound Connection Pool

Which Outbound Connection Pool would you like the credential map to be associated w
this resource adapter. Each Outbound Connection Pool can then configure themselves

Customize this table

Create a New Security Credential Map Entry for:

	Outbound Connection Pool
✓	eis/OracleJCAAdapter/DefaultConnection
	Resource Adapter Default

5. Select the outbound connection pool.

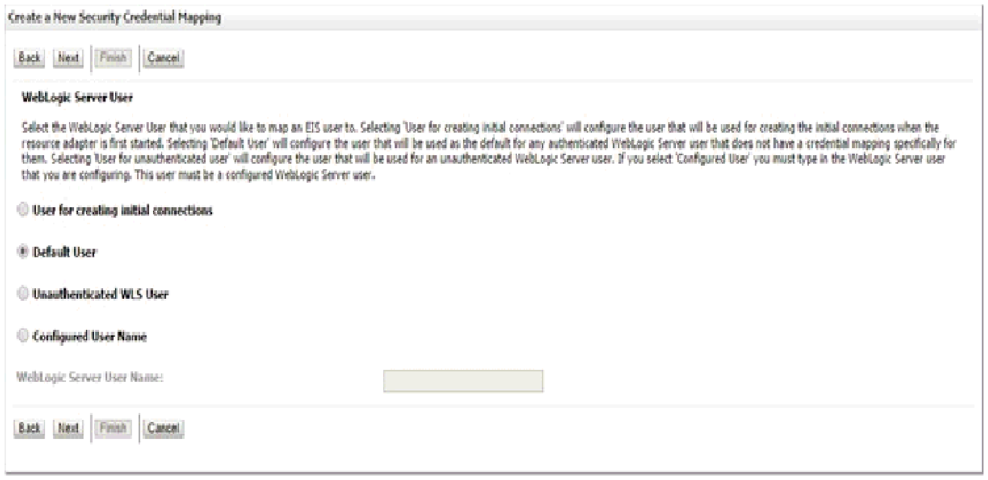
   For example:

   ```
   eis/OracleJCAAdapter/DefaultConnection
   ```

6. Click **Next**.

   The WebLogic Server User page is displayed, as shown in Figure 9–90.

Figure 9–90 WebLogic Server User Page


Create a New Security Credential Mapping

WebLogic Server User

Select the WebLogic Server User that you would like to map an EIS user to. Selecting 'User for creating initial connections' will configure the user that will be used for creating the initial connections when the resource adapter is first started. Selecting 'Default User' will configure the user that will be used as the default for any authenticated WebLogic Server user that does not have a credential mapping specifically for them. Selecting 'User for unauthenticated user' will configure the user that will be used for an unauthenticated WebLogic Server user. If you select 'Configured User' you must type in the WebLogic Server user that you are configuring. This user must be a configured WebLogic Server user.

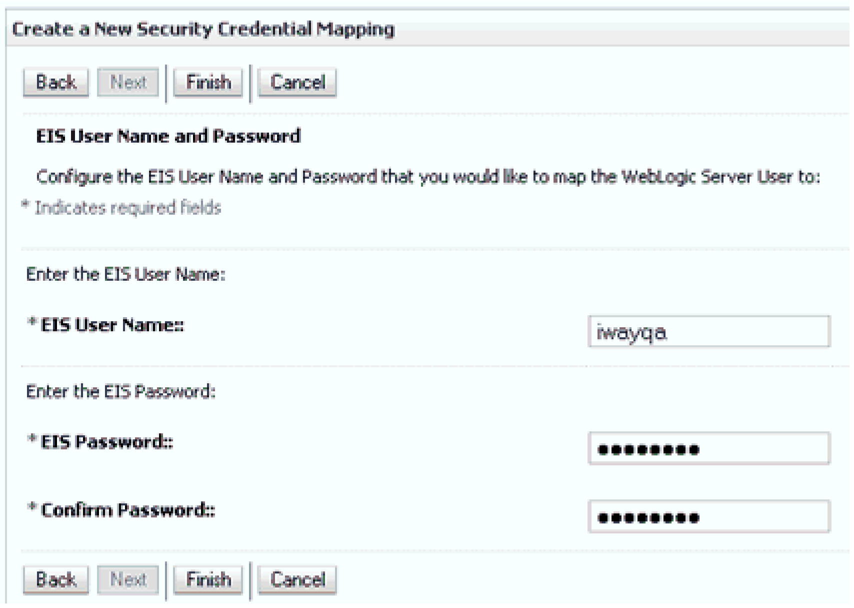
User for creating initial connections [✓ Default User] Unauthenticated WLS User Configured User Name

WebLogic Server User Name:

7. Select **Default User**, enter a valid Oracle WebLogic Server user name, and then click **Next**.

   The EIS User Name and Password page is displayed, as shown in Figure 9–91.

Figure 9–91 EIS User Name and Password Page


Create a New Security Credential Mapping

EIS User Name and Password

Configure the EIS User Name and Password that you would like to map the WebLogic Server User to:

* Indicates required fields

Enter the EIS User Name:

* EIS User Name:: iwayqa

Enter the EIS Password:

* EIS Password:: ●●●●●●●●

* Confirm Password:: ●●●●●●●●

8. Enter the user name and password for the EIS and click **Finish**.

   The credentials for an Oracle WebLogic Server user are now mapped with an EIS user (J.D. Edwards OneWorld). The mapping is invoked automatically before invoking the J2CA service.

9.5.1.2 Generating a WSDL File

To generate a WSDL file:

1. Open Application Explorer and create a J2CA configuration.

   For more information, see Section 2.3.2, "Creating a Configuration for J2CA" on page 2-3.

2. Create a target for the J.D. Edwards OneWorld adapter and then connect to the target.

   For more information, see Section 2.4, "Establishing a Connection (Target) for J.D. Edwards OneWorld" on page 2-5.

3. Generate a WSDL for the appropriate object.

   For more information, see Section 2.6, "Generating WSDL (J2CA Configurations Only)" on page 2-11.

9.5.1.3 Creating and Deploying an Outbound Process

This section describes how to configure an outbound process. For demonstration purposes, specific references to the BPEL outbound process are made. However, the same steps apply to Mediator and BPM outbound processes.

For more information about creating a Mediator outbound process, see Chapter 5, "Integration With Mediator Service Components in the Oracle SOA Suite".

For more information about creating a BPM outbound process, see Chapter 6, "Integration With BPM Service Components in the Oracle SOA Suite".

To create a BPEL outbound process, see the following sections:

- Section 4.4.2, "Creating an Empty Composite for SOA"
- Section 4.4.3, "Defining a BPEL Outbound Process"
- Section 4.4.4, "Deploying the BPEL Outbound Process"

9.5.1.4 Verifying the EIS Credentials

Invoke the input XML and ensure that the EIS target credentials are overridden with the credentials configured in the WebLogic Administration Console for the Default User as described in this section.

1. Invoke the deployed BPEL outbound process with a valid input.

   For more information, see Section 4.4.5, "Invoking the Input XML Document in the Oracle Enterprise Manager Console" on page 4-31.

2. Check the J2CA log files and locate the encrypted password, which shows that the user credentials have been passed to the EIS through Oracle WebLogic Server.

   For example:

```
FINEST IWAFManagedConnectionFactory com.ibi.afjca.Util
getPasswordCredential(78) InLoop:
User-iwayqa:Password-ENCR(3109311731831131382333215315332323192322731773172)
FINEST IWAFManagedConnectionFactory com.ibi.afjca.Util
getPasswordCredential(90) Use the system PasswordCredential:
User-iwayqa:Password-ENCR(3109311731831131382333215315332323192322731773172)
```

9.6 Credential Mapping for Oracle Service Bus (OSB)

This section describes how to configure credential mapping functionality for the Oracle Application Adapter for J.D. Edwards OneWorld in a configuration that uses Oracle Service Bus (OSB). A sample testing scenario is also included. This section contains the following topic:

- Section 9.6.1, "Configuring Credential Mapping"

Credential mapping is supported only for outbound processes that use J2CA configurations. This feature is not supported for BSE configurations and inbound processes that use J2CA configurations.

Note: The J2CA connector is common to all four application adapters (SAP R/3, PeopleSoft, Siebel, and J.D. Edwards OneWorld). If credential mapping is required, then ensure that only one application adapter is used in a particular instance. For example, in one adapter instance only the J.D. Edwards OneWorld application adapter can be used. Credential mapping cannot be configured at the individual adapter level. If you require the use of credential mapping for two adapters, then both adapters must be running in two independent adapter instances.

To pass user credentials to the iWay J2CA resource adapter, create a credential map from the Oracle WebLogic Server user credentials to the EIS user credentials (J.D. Edwards OneWorld adapter). Then associate a credential policy with a Web service and invoke the Web service using Oracle WebLogic Server user credentials. These credentials are mapped to the EIS user credentials and then passed to the iWay J2CA container, which uses them to connect with the EIS adapter (J.D. Edwards OneWorld).

9.6.1 Configuring Credential Mapping

Configuring credential mapping consists of the following steps and topics:

1. Deploy the adapter.

   For more information, see Chapter 3, "Oracle WebLogic Server Deployment and Integration".

2. Associate Oracle WebLogic Server credentials with EIS credentials.

   For more information, see Section 9.6.1.1, "Associating Oracle WebLogic Server Credentials With EIS Credentials".

3. Generate a WSDL file.

   For more information, see Section 9.6.1.2, "Generating a WSDL File".

4. Create an Oracle Service Bus (OSB) outbound process.

   For more information, see Section 9.6.1.3, "Creating an Oracle Service Bus (OSB) Outbound Process" on page 9-65.

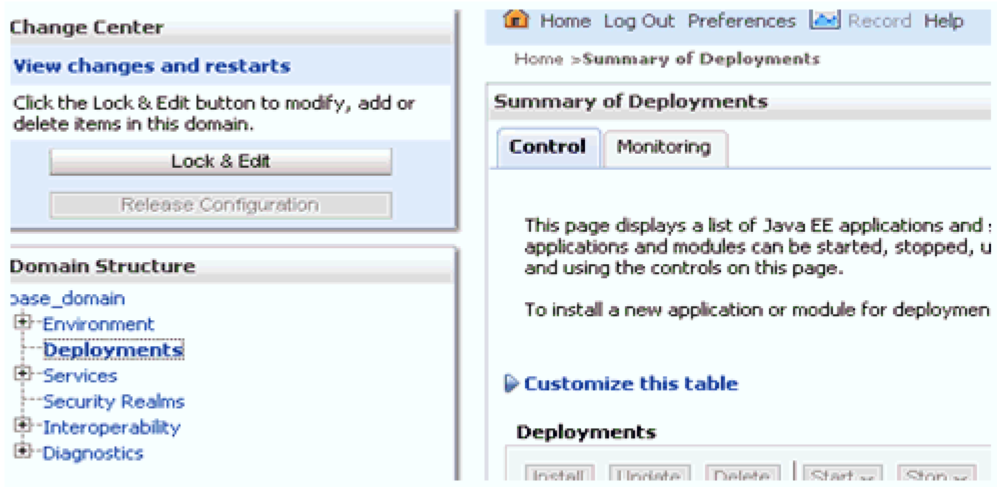
9.6.1.1 Associating Oracle WebLogic Server Credentials With EIS Credentials

To associate Oracle WebLogic Server credentials with EIS credentials:

1. Log in to the Oracle WebLogic Server Administration Console.

2. In the Domain Structure section in the left pane, click **Deployments**, as shown in Figure 9–92.

Figure 9–92 Domain Structure Section



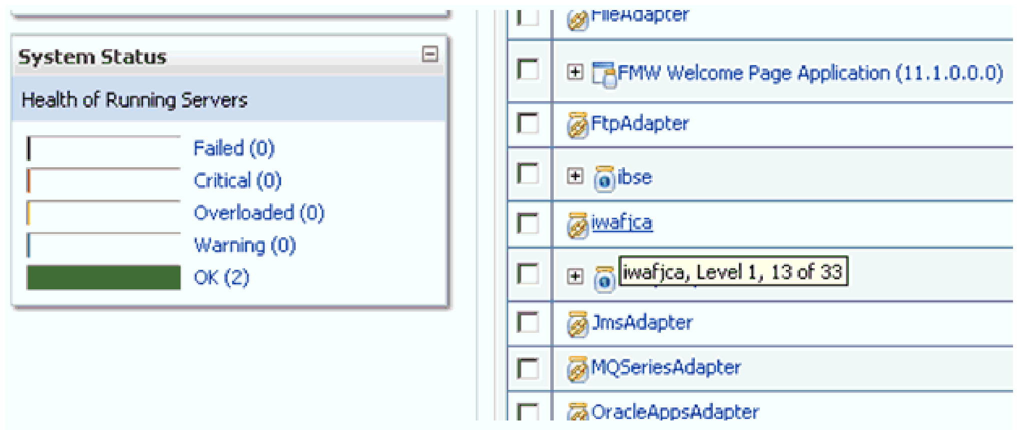
Click the Lock & Edit button to modify, add or delete items in this domain.

This page displays a list of Java EE applications and
applications and modules can be started, stopped, u
and using the controls on this page.

To install a new application or module for deploymen

The Deployments page is displayed, as shown in Figure 9–93.

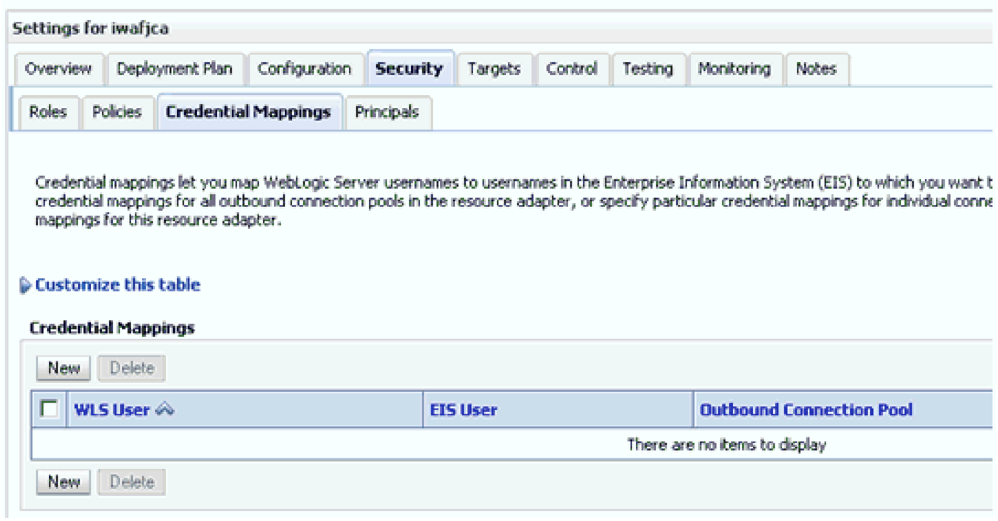
Figure 9–93 Deployments Page



3. Click the **iwafjca** resource adapter.

   The Settings for iwafjca page is displayed, as shown in Figure 9–94.

Figure 9–94 Settings for iwafjca Page



Credential mappings let you map WebLogic Server usernames to usernames in the Enterprise Information System (EIS) to which you want t
credential mappings for all outbound connection pools in the resource adapter, or specify particular credential mappings for individual conne
mappings for this resource adapter.

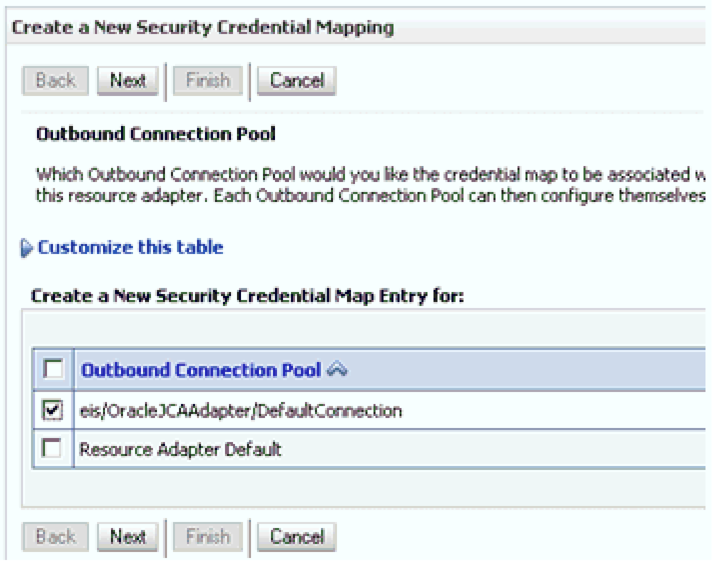
Credential Mappings

WLS User	EIS User	Outbound Connection Pool
There are no items to display		

4. Click the **Credential Mappings** tab under the Security tab, and then click **New**.

   The Create a New Security Credential Mapping page is displayed, as shown in Figure 9–95.

Figure 9–95 Create a New Security Credential Mapping Page


Create a New Security Credential Mapping

Outbound Connection Pool

Which Outbound Connection Pool would you like the credential map to be associated w
this resource adapter. Each Outbound Connection Pool can then configure themselves

Create a New Security Credential Map Entry for:

	Outbound Connection Pool
☐	
☑	eis/OracleJCAAdapter/DefaultConnection
☐	Resource Adapter Default

5. Select the outbound connection pool.

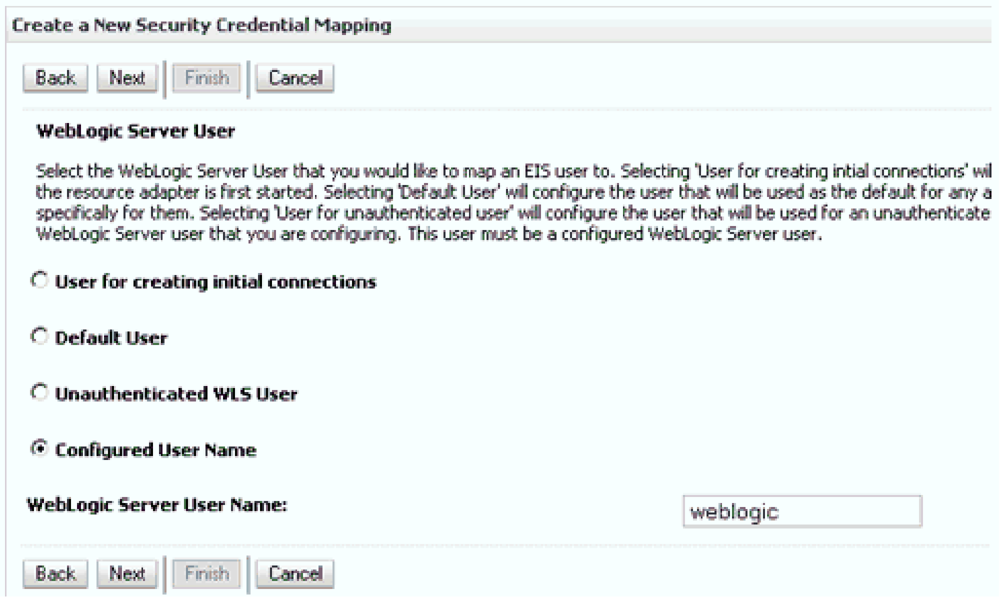
   For example:

   ```
   eis/OracleJCAAdapter/DefaultConnection
   ```

6. Click **Next**.

   The WebLogic Server User page is displayed, as shown in Figure 9–96.

Figure 9–96 WebLogic Server User Page


Create a New Security Credential Mapping

WebLogic Server User

Select the WebLogic Server User that you would like to map an EIS user to. Selecting 'User for creating intial connections' wil
the resource adapter is first started. Selecting 'Default User' will configure the user that will be used as the default for any a
specifically for them. Selecting 'User for unauthenticated user' will configure the user that will be used for an unauthenticate
WebLogic Server user that you are configuring. This user must be a configured WebLogic Server user.

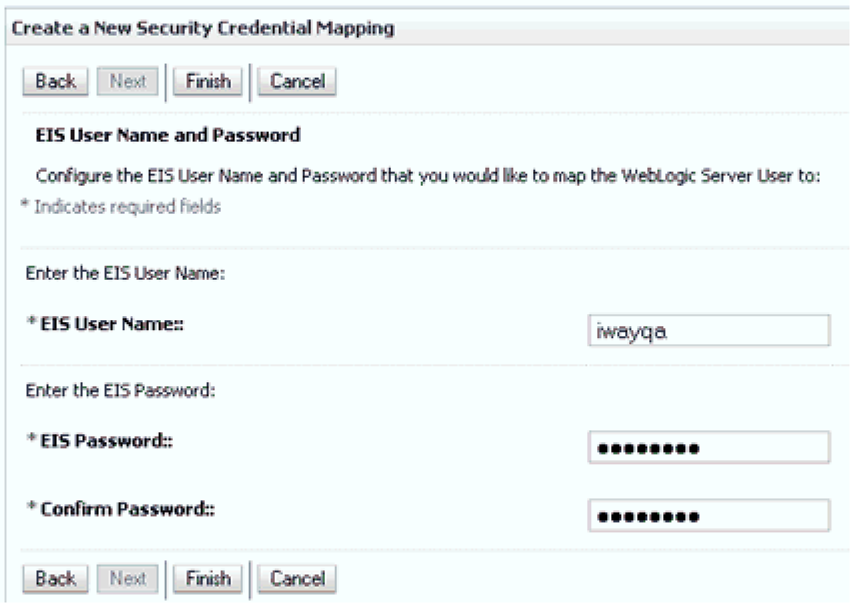
User for creating initial connections
Default User
Unauthenticated WLS User
[✓ Configured User Name]

WebLogic Server User Name: weblogic

7. Select Configured User Name, enter a valid Oracle WebLogic Server user name, and then click **Next**.

   The EIS User Name and Password page is displayed, as shown in Figure 9–97.

Figure 9–97 EIS User Name and Password Page


Create a New Security Credential Mapping

EIS User Name and Password

Configure the EIS User Name and Password that you would like to map the WebLogic Server User to:

* Indicates required fields

Enter the EIS User Name:

* EIS User Name::	iwayqa

Enter the EIS Password:

* EIS Password::	●●●●●●●●
* Confirm Password::	●●●●●●●●

8. Enter the user name and password for the EIS and click **Finish**.

   The credentials for an Oracle WebLogic Server user are now mapped with an EIS user (J.D. Edwards OneWorld). The mapping is invoked automatically before invoking the J2CA service.

9.6.1.2 Generating a WSDL File

To generate a WSDL file:

1. Set the class path for Application Explorer to integrate with Oracle Service Bus (OSB).

   For more information, see Section 7.2.2, "Setting the Class Path for Application Explorer to Integrate With Oracle Service Bus" on page 7-6.

2. Open Application Explorer and create a J2CA configuration.

   For more information, see Section 2.3.2, "Creating a Configuration for J2CA" on page 2-3.

3. Create a target for the J.D. Edwards OneWorld adapter and then connect to the target.

   For more information, see Section 2.4, "Establishing a Connection (Target) for J.D. Edwards OneWorld" on page 2-5.

4. Generate a WSDL for the appropriate object.

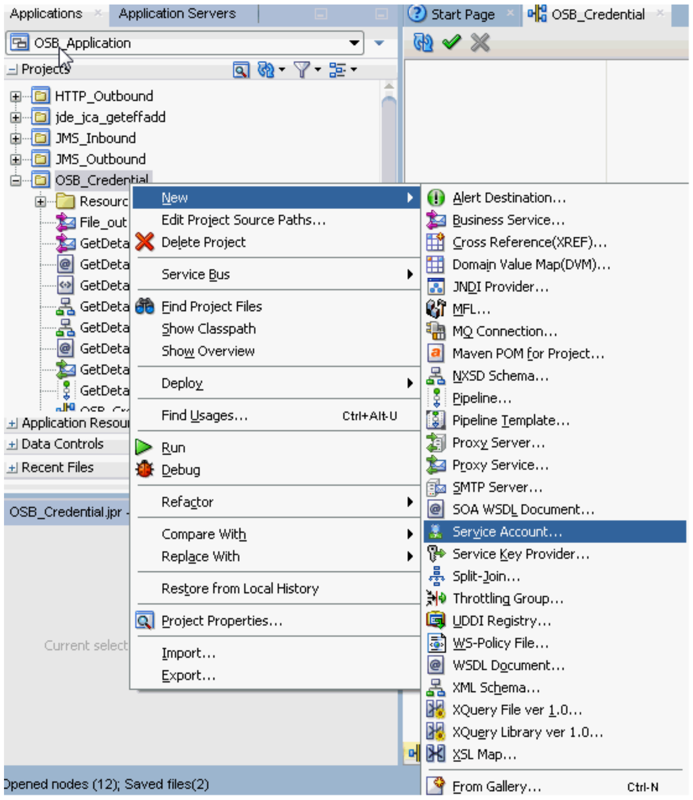
   For more information, see Section 4.4.1, "Generating WSDL for Request/Response Service" on page 4-8.

9.6.1.3 Creating an Oracle Service Bus (OSB) Outbound Process

For more information on creating an Oracle Service Bus (OSB) outbound process, see Section 8.1.2, "Defining an OSB Outbound Process" on page 8-3.

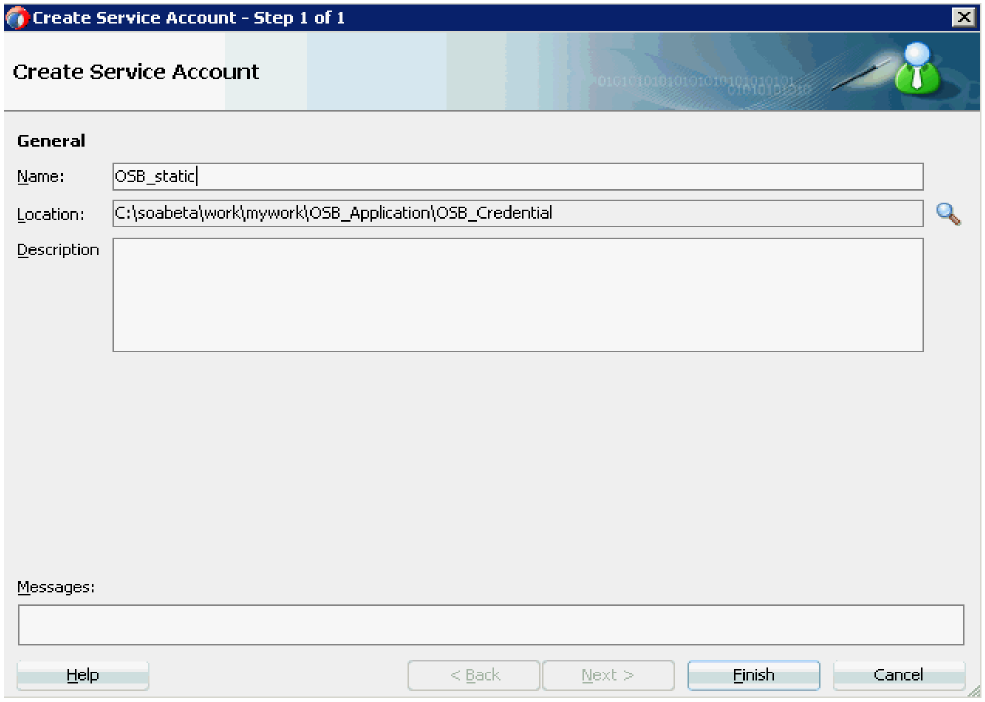
1. Configure a Service account by right-clicking the OSB Project, selecting **New**, and then clicking **Service Account**, as shown in Figure 9–98.

Figure 9–98 Select Service Account Option



The Create Service Account pane is displayed, as shown in Figure 9–99.

Figure 9–99 Create Service Account Pane



Name: OSB_static

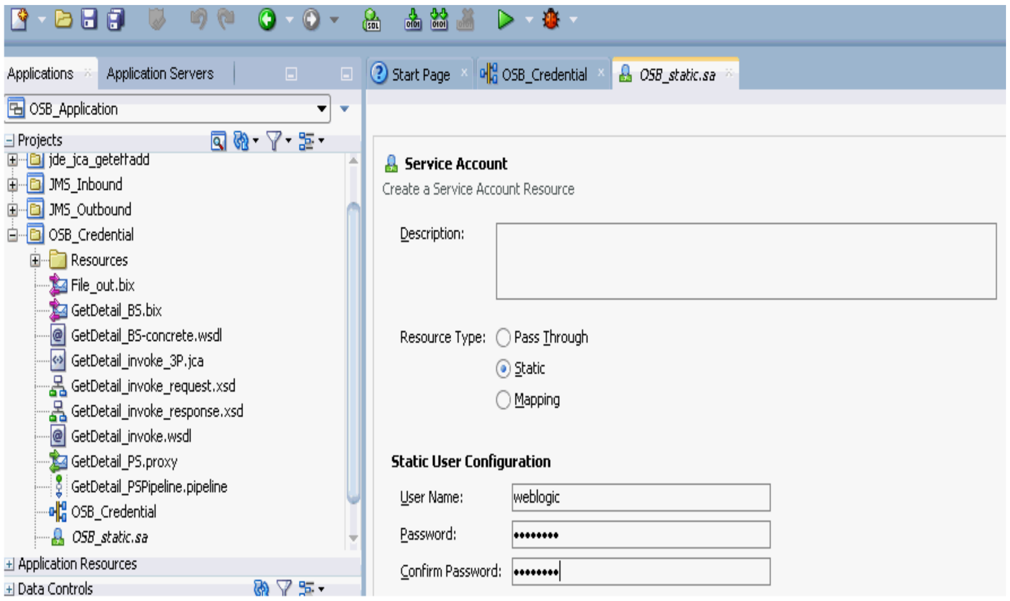
Location: C:\soabeta\work\mywork\OSB_Application\OSB_Credential

2. Provide a name for the Service Account and click **Finish**.

The configuration page of Service Account is displayed.

3. In the Resource Type section, select **Static**.
4. Provide a valid user name and password for the Oracle WebLogic Server, as shown in Figure 9–100.

Figure 9–100 Service Account Configuration Page



Service Account

Create a Service Account Resource

Description:

Resource Type: ( ) Pass Through [✓ Static] ( ) Mapping

Static User Configuration

User Name: weblogic

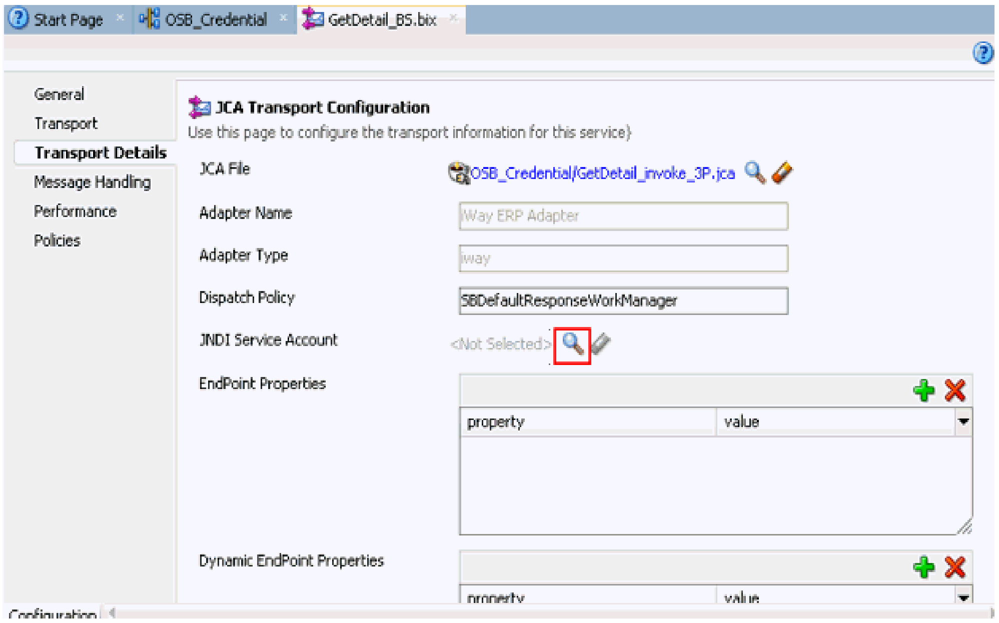
Password: ••••••••

Confirm Password: ••••••••

5. Save and close the configuration page.
6. In the composite Editor window, double-click the created WSDL-based Business Service from step 3.

   The configuration page of the WSDL-based Business Service is displayed.
7. Select the Transport Details tab, as shown in Figure 9–101.

Figure 9–101 Transport Details Tab



JCA Transport Configuration

Use this page to configure the transport information for this service)

JCA File: OSB_Credential/GetDetail_invoke_3P.jca

Adapter Name: JWay ERP Adapter

Adapter Type: jway

Dispatch Policy: SBDefaultResponseWorkManager

JNDI Service Account: <Not Selected>

EndPoint Properties

property	value

Dynamic EndPoint Properties

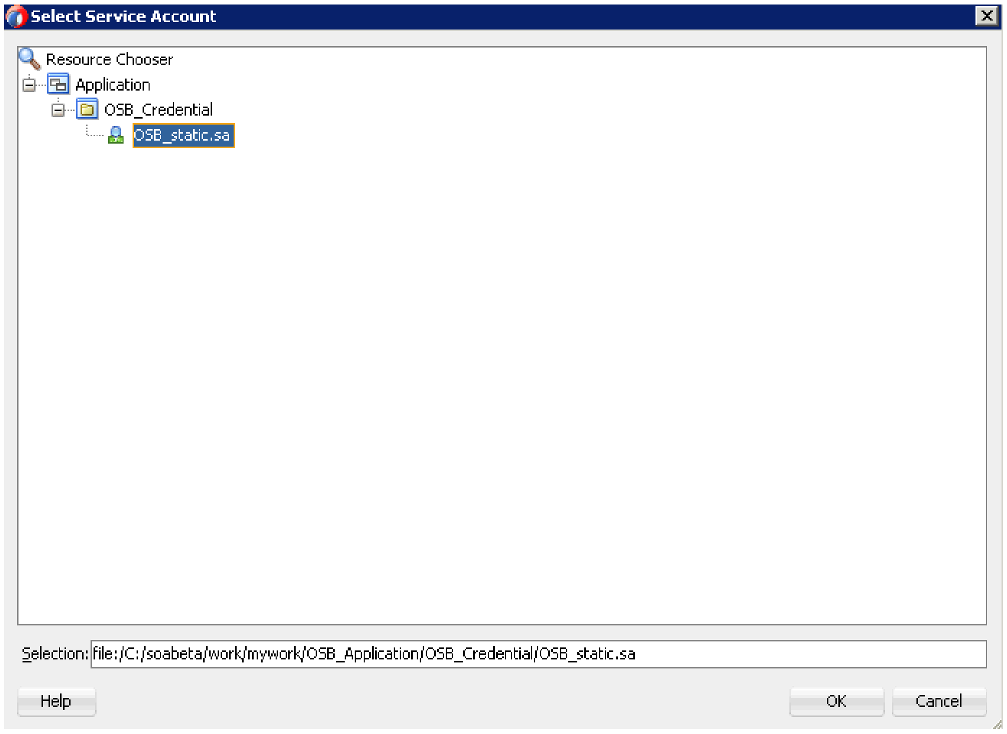
property	value

8. In the JNDI Service Account section, click the Browse icon.

   The Select Service Account window is displayed.

9. Select the created service account and click **OK**, as shown in Figure 9–102.

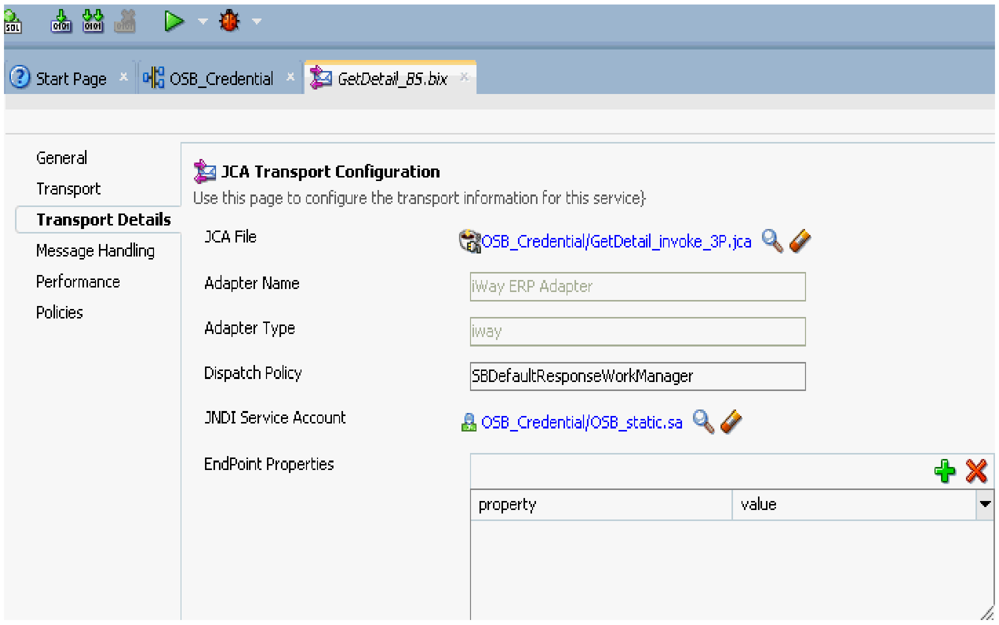
Figure 9–102 Select Service Account



Selection: file:/C:/soabeta/work/mywork/OSB_Application/OSB_Credential/OSB_static.sa

10. Save and close the configuration page, as shown in Figure 9–103

Figure 9–103 Business Service Configuration Page


Start Page
OSB_Credential
GetDetail_BS.bix

General
Transport
Transport Details
Message Handling
Performance
Policies

JCA Transport Configuration
Use this page to configure the transport information for this service}

JCA File	OSB_Credential/GetDetail_invoke_3P.jca
Adapter Name	iWay ERP Adapter
Adapter Type	iway
Dispatch Policy	SBDefaultResponseWorkManager
JNDI Service Account	OSB_Credential/OSB_static.sa
EndPoint Properties	

property	value

11. Deploy the OSB process.

    For more information, see Section 8.1.3, "Deploying the OSB Outbound Process" on page 8-16.

12. Once the process is deployed successfully, copy and paste a valid input XML file in the input folder you configured, and check to see that the output is received in the configured output location.

13. Check the J2CA log files and locate the encrypted password, which shows that the user credentials have been passed to the EIS through Oracle WebLogic Server.

    For example:

```
FINEST IWAFManagedConnectionFactory com.ibi.afjca.Util
getPasswordCredential(78) InLoop:
User-iwayqa:Password-ENCR(3189319731831132182333215323332323192322731773252)
FINEST IWAFManagedConnectionFactory com.ibi.afjca.Util
getPasswordCredential(90) Use the system PasswordCredential:
User-iwayqa:Password-ENCR(3109313331831131702333215320132323192322731773236)
```

10 Troubleshooting and Error Messages

This chapter explains the limitations and workarounds when connecting to J.D. Edwards OneWorld. It contains the following topics:

- Section 10.1, "Troubleshooting"
- Section 10.2, "BSE Error Messages"

The adapter-specific errors listed in this chapter can arise whether using the adapter with an Oracle Adapter J2CA or with a Oracle Adapter Business Services Engine (BSE) configuration.

10.1 Troubleshooting

This topic provides troubleshooting information for J.D. Edwards OneWorld, and contains the following topics:

- Section 10.1.1, "Application Explorer"
- Section 10.1.2, "J.D. Edwards One World"
- Section 10.1.3, "Oracle Adapter J2CA"

Log file information that can be relevant in troubleshooting can be found in the following locations based on your adapter installation:

- The Oracle Adapter J2CA trace information can be found under the following directory:

  ```
  <ADAPTER_HOME>\config\configuration_name\log
  ```

- BSE trace information can be found under the following directory:

  ```
  <ORACLE_HOME>\user_projects\domains\base_domain\servers\soa_
  server1\stage\ibse\ibse.war\ibselogs
  ```

- The log file for Application Explorer can be found under the following directory:

  ```
  <ADAPTER_HOME>\tools\iwae\bin
  ```

10.1.1 Application Explorer

This topic discusses the different types of errors that can occur when using Application Explorer.

Error	Solution
Cannot connect to Oracle Application Adapter for J.D. Edwards OneWorld from Application Explorer: `Problem activating adapter. (Failed to connect to J.D.Edwards OneWorld, check system availability and configuration parameters:...) Check logs for more information.`	Ensure that: ■ J.D. Edwards OneWorld is running. ■ The J.D. Edwards OneWorld user ID and password is correct. ■ The port number is correct.
The following error message appears: java.lang.IllegalStateException: java.lang.Exception: Error Logon to J.D. Edwards OneWorld System	You have provided invalid connection information for J.D. Edwards OneWorld or the wrong JAR file is in the lib directory.
J.D. Edwards OneWorld does not appear in the Application Explorer Adapter node list.	Ensure that the J.D. Edwards OneWorld JAR files, are added to the lib directory.
Logon failure error at run-time.	If the password for connecting to your J.D. Edwards OneWorld system is not specified when creating a target or with the Edit option in Application Explorer, then you are unable to connect to J.D. Edwards OneWorld. The connection password is not saved in `repository.xml`. Update the password using the Edit option in Application Explorer, then restart the application server.
The following exception occurs when you start Application Explorer by activating `ae.bat` (not `iaexplorer.exe`): `java.lang.ClassNotFoundException: org.bouncycastle.jce.provider.BouncyCastleProvider`	This is a benign exception. It does not affect adapter functionality. Download BouncyCastle files from: `ftp://ftp.bouncycastle.org/pub`

Error	Solution

Unable to start Application Explorer in a Solaris environment. The following exception is thrown in the console:
 	
`javax.resource.ResourceException: IWAFManagedConnectionFactory: License violation.at com.ibi.afjca.spi.IWAFManagedConnectionFactory.createConnectionFactory(IWAFManagedConnectionFactory.java:98)at com.iwaysoftware.iwae.common.JCATransport.getConnectionFactory(JCATransport.java:133) at com.iwaysoftware.iwae.common.JCATransport.initJCA(JCATransport.java:69)at com.iwaysoftware.iwae.common.JCATransport.<init>(JCATransport.java:62)at com.iwaysoftware.iwae.common.AdapterClient.<init>(AdapterClient.java:85)at com.ibi.bse.ConfigWorker.run(ConfigWorker.java:41)at java.lang.Thread.run(Thread.java:534)`
 	
`Could not create the connection factory.`
	`JAVACMD` is not set on the user system. Before starting Application Explorer, export `JAVACMD` as follows:
	
	`JAVACMD=/<jdk_home>/bin/java`, where `<jdk_home>` is the directory where JDK is installed on your system.

10.1.2 J.D. Edwards One World

Error	Cause	Solution
Action code invalid	In the Sales Order request, the Action code appears as "H," an invalid action code.	Use: ■ "I" for inquiry. ■ "C" for change. ■ "D" for delete. ■ "A" to add a new record.
Invalid address number.	The address number does not exist in the Address Book Master file (F0101).	Enter an address number using the Address Book Revisions program (PO1051). Ensure that the number entered is correct.
Record invalid	The record being processed either already exists for an ADD function or does not exist for an INQUIRY, CHANGE, or DELETE function.	If you are attempting to inquire, change, or delete a record you added previously, then there could be database problems in your production library. Contact your data processing department.
Item Branch record does not exist.	An Item Branch record (F4102) does not exist for this item in the Branch/Plant specified.	Correct the Branch or enter an Item Branch record for this item in Branch Plant Item Information (P41026).
&1 does not match any of the valid values.	The &1 does not match any of the valid values specified in the Data Dictionary for this field.	Enter a valid value.

Error	Cause	Solution
Date out of range.	The Last Service Date and the Inspection Date must be within the range of the effective dates of the Service Contract.	Change the date to be greater than or equal to the beginning effective date and less than or equal to the ending effective date of the Service Contract.
Jde.net timeout exception	Net timeout is set to a wrong value	Verify that net timeout is set to 180 at `jde.ini` of [NETWORK QUEUE SETTINGS], for example `JDENETTimeout=180`
Cannot connect to EnterpriseOne Version 8.10	Missing required library files	`Kernel.jar` and `Connector.jar` are required for version B7333. `jdeutil.jar` and `log4j.jar` are required for EnterpriseOne Version 8.10, in addition to `Kernel.jar` and `Connector.jar`.

10.1.3 Oracle Adapter J2CA

Error	Solution
In Application Explorer, the following error message appears when you attempt to connect to an Oracle Adapter J2CA configuration: `Could not initialize JCA`	In the Details tab in the right pane, ensure that the directory specified in the Home field points to the correct directory, for example: *<ADAPTER_HOME>*`\tools\iwae\bin\..\..\..\`

10.2 BSE Error Messages

This section discusses the different types of errors that can occur when processing Web services through BSE.

This section contains the following topics:

- Section 10.2.1, "General Error Handling in BSE"
- Section 10.2.2, "Adapter-Specific Error Handling"

10.2.1 General Error Handling in BSE

BSE serves as both a SOAP gateway into the adapter framework and as the engine for some of the adapters. In both design time and run-time, various conditions can cause errors in BSE when Web services that use adapters run. Some of these conditions and resulting errors are exposed the same way, regardless of the specific adapter; others are exposed differently, based on the adapter being used. This topic explains what you can expect when you encounter some of the more common error conditions on an adapter-specific basis.Usually the SOAP gateway (agent) inside BSE passes a SOAP request message to the adapter required for the Web service. If an error occurs, then how it is exposed depends on the adapter and the API or interfaces that the adapter uses. A few scenarios cause the SOAP gateway to generate a SOAP fault. In general, anytime the SOAP agent inside BSE receives an invalid SOAP request, a SOAP fault element is generated in the SOAP response. The SOAP fault element contains fault string and fault code elements. The fault code contains a description of the SOAP agent error. The following SOAP response document results when BSE receives an invalid SOAP request:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">

 <SOAP-ENV:Body>
      <SOAP-ENV:Fault>
         <faultcode>SOAP-ENV:Client</faultcode>
         <faultstring>Parameter node is missing</faultstring>
      </SOAP-ENV:Fault>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

In this example, BSE did not receive an element in the SOAP request message that is mandatory for the WSDL for this Web service.

10.2.2 Adapter-Specific Error Handling

This section contains the following topics:

- Section 10.2.2.1, "Invalid SOAP Request"
- Section 10.2.2.2, "Empty Result From Oracle WebLogic Server Adapter Request"
- Section 10.2.2.3, "Error Logging In"
- Section 10.2.2.4, "Empty Result From Oracle WebLogic Server Adapter Request"
- Section 10.2.2.4, "Empty Result From Oracle WebLogic Server Adapter Request"

When an adapter raises an exception during run-time, the SOAP agent in BSE produces a SOAP fault element in the generated SOAP response. The SOAP fault element contains fault code and fault string elements. The fault string contains the native error description from the adapter target system. Since adapters use the target system interfaces and APIs, whether an exception is raised depends on how the target systems interface or API treats the error condition. If a SOAP request message is passed to an adapter by the SOAP agent in BSE, and that request is invalid based on the WSDL for that service, then the adapter may raise an exception yielding a SOAP fault.

While it is almost impossible to anticipate every error condition that an adapter may encounter, the following is a description of how adapters handle common error conditions and how they are then exposed to the Web services consumer application.

10.2.2.1 Invalid SOAP Request

If Oracle WebLogic Server Adapter receives a SOAP request message that does not conform to the WSDL for the Web services being executed, then the following SOAP response is generated.

```
<?xml version="1.0" encoding="ISO-8859-1"
 ?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<SOAP-ENV:Body>
 <SOAP-ENV:Fault>
  <faultcode>SOAP-ENV:Server</faultcode>
  <faultstring>RPC server connection failed: Connection refused:
connect</faultstring>
 </SOAP-ENV:Fault>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

10.2.2.2 Empty Result From Oracle WebLogic Server Adapter Request

If Oracle WebLogic Server Adapter executes a SOAP request using input parameters passed that do not match records in the target system, then the following SOAP response is generated.

Note: The condition for this adapter does not yield a SOAP fault.

```
<SOAP-ENV:Envelope xmlns:xsi="http://www.w3.org/1999/XMLSchema-instance"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/1999/XMLSchema">
   <SOAP-ENV:Body>
      <m:RunDBQueryResponse xmlns:m="urn:schemas-iwaysoftware-com:iwse"
        xmlns="urn:schemas-iwaysoftware-com:iwse"
        cid="2A3CB42703EB20203F91951B89F3C5AF">
        <RunDBQueryResult run="1" />
     </m:RunDBQueryResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

10.2.2.3 Error Logging In

If Oracle WebLogic Server Adapter executes an invalid SOAP log in request, then the following SOAP response is generated.

```
[2004-07-19T16:28:56:718Z] DEBUG (SOAP1) W.SOAP1.2: POST received
[2004-07-19T16:28:56:718Z] DEBUG (SOAP1) W.SOAP1.2: in XDSOAPHTTPWorker agentName
is [XDSOAPRouter]
[2004-07-19T16:28:56:718Z] DEBUG (SOAP1) W.SOAP1.2: before parse:
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<SOAP-ENV:Header>
<m:ibsinfo xmlns:m="urn:schemas-iwaysoftware-...[861]
[2004-07-19T16:28:56:718Z] ERROR (SOAP1) W.SOAP1.2: Attempting string, no encoding
recognized in document
[2004-07-19T16:28:56:734Z] DEEP (SOAP1) W.SOAP1.2: parse complete in 16 msecs
[2004-07-19T16:28:56:859Z] DEEP (SOAP1) W.SOAP1.2: ST_NODICT
[2004-07-19T16:28:56:859Z] DEEP (SOAP1) W.SOAP1.2: ST_FINISH
[2004-07-19T16:28:56:859Z] DEBUG (SOAP1) extractControl - edaDoc: false
[2004-07-19T16:28:56:859Z] DEBUG (SOAP1) now: 2004-07-19T16:28:56Z expires:
2004-07-20T16:28:56Z
[2004-07-19T16:28:56:859Z] DEBUG (SOAP1) W.SOAP1.2: checking for cached agent
[2004-07-19T16:28:56:859Z] DEBUG (SOAP1) W.SOAP1.2: pushagent: adding agent
com.ibi.iwse.XDSOAPRouter
[2004-07-19T16:28:56:875Z] DEBUG (SOAP1) W.SOAP1.2: inside worker the soap Action
is [B0100033.GetEffectiveAddressRequest#test##]
[2004-07-19T16:28:56:875Z] DEBUG (SOAP1) W.SOAP1.2: precedence: 1
[2004-07-19T16:28:56:875Z] DEBUG (SOAP1) W.SOAP1.2: precedence: 1
[2004-07-19T16:28:56:875Z] DEBUG (SOAP1) W.SOAP1.2: numagents: 1
[2004-07-19T16:28:56:890Z] DEBUG (SOAP1) W.SOAP1.2: running agent 1 name
com.ibi.iwse.XDSOAPRouter document 1
[2004-07-19T16:28:56:890Z] INFO  (manager) MGR00X01: Adding active worker:
W.SOAP1.2
[2004-07-19T16:28:56:890Z] DEBUG (SOAP1) W.SOAP1.2: <?xml version="1.0"
encoding="UTF-8" ?>
<SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
```

```
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <SOAP-ENV:Header>
      <m:ibsinfo xmlns:m="urn:schemas-iwaysoftware-com:iwse">
         <m:service>B0100033</m:service>
         <m:method>GetEffectiveAddress</m:method>
         <m:license>test</m:license>
         <m:Username>user</m:Username>
         <m:Password>password</m:Password>
      </m:ibsinfo>
   </SOAP-ENV:Header>
   <SOAP-ENV:Body>
      <m:GetEffectiveAddress
xmlns:m="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress">
         <m:jdeRequest type="callmethod">
            <m:callMethod name="GetEffectiveAddress">
               <m:params>
                  <m:param name="mnAddressNumber">12345</m:param>
               </m:params>
               <m:onError/>
            </m:callMethod>
         </m:jdeRequest>
      </m:GetEffectiveAddress>
   </SOAP-ENV:Body>
   <SOAPAction agentName="XDSOAPRouter"
cid="1FF3D44E0B0AFB2A4E9538ED42B71437">B0100033.GetEffectiveAddressRequest#test##<
/SOAPAction>
</SOAP-ENV:Envelope>
[2004-07-19T16:28:56:890Z] DEBUG (SOAP1) W.SOAP1.2: business method:
m:GetEffectiveAddress
[2004-07-19T16:28:56:906Z] DEBUG (SOAP1) W.SOAP1.2: input:
[2004-07-19T16:28:56:906Z] DEBUG (SOAP1) W.SOAP1.2: <?xml version="1.0"
encoding="UTF-8" ?><jdeRequest xmlns:xsd="http://www.w3.org/2001/XMLSchema"
type="callmethod" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><callMethod
name="GetEffectiveAddress"><params><param name="mnAddressNumber">12345</param>
      </params><onError/></callMethod></jdeRequest>
[2004-07-19T16:28:58:234Z] DEBUG (SOAP1) W.SOAP1.2: Agent returned success
[2004-07-19T16:28:58:234Z] INFO  (manager) MGR00X02: Removing active worker:
W.SOAP1.2
[2004-07-19T16:28:58:234Z] DEBUG (SOAP1) W.SOAP1.2: doing docTran, docVal,
listTran for agent(1)
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: sendToAll reply to XDReply:
[protocol=http */null]
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: preemitters from doc: null
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: no preemitters, emitting
contents of doc, usestream=false encoding=UTF-8
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: writeEntity, len: 670 data:
<?xml version="1.0" encoding="UTF-8" ?><SOAP-ENV:Envelope
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><SOAP-ENV:Body><GetEffective
AddressResponse xmlns="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress:response"
cid="1FF3D44E0B0AFB2A4E9538ED42B71437"><jdeResponse user="USER" type="callmethod"
session="" environment="DV7333"><returnCode code="12">Environment
&apos;DV7333&apos; could not be initialized for user, check user, pwd and
environment attribute
values</returnCode></jdeResponse></GetEffectiveAddressResponse></SOAP-ENV:Body></S
OAP-ENV:Envelope>
```

```
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: writeString: HTTP/1.0
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: writeString: 200
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: writeString: OK
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: writeString: Content-Type:
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: writeString: text/xml
[2004-07-19T16:28:58:250Z] DEBUG (SOAP1) W.SOAP1.2: writeString: Content-Length:
[2004-07-19T16:28:58:265Z] DEBUG (SOAP1) W.SOAP1.2: writeString: 670
[2004-07-19T16:28:58:265Z] INFO  (SOAP1) W.SOAP1.2: W0000X13: Ended message
processing, rc=0
[2004-07-19T16:28:58:265Z] DEEP (SOAP1) W.SOAP1.2: storing used socket
[2004-07-19T16:28:58:265Z] DEBUG (SOAP1) W.SOAP1.2: entering waitforDocument
[2004-07-19T16:29:03:875Z] DEEP (SOAP1) W.SOAP1.2: cleanup: closing sockets(0)
```

10.2.2.4 Empty Result From Oracle WebLogic Server Adapter Request

If Oracle WebLogic Server Adapter executes a SOAP request using input parameters passed that do not match records in the target system, then the following SOAP response is generated.

Note: The condition for this adapter does not yield a SOAP fault.

```
[2004-07-19T16:27:05:640Z] DEBUG (SOAP1) W.SOAP1.2: POST received
[2004-07-19T16:27:05:640Z] DEBUG (SOAP1) W.SOAP1.2: in XDSOAPHTTPWorker agentName
is [XDSOAPRouter]
[2004-07-19T16:27:05:640Z] DEBUG (SOAP1) W.SOAP1.2: before parse:
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<SOAP-ENV:Body>
<m:GetEffectiveAddress xmlns:m="urn:iwaysoftwar...[590]
[2004-07-19T16:27:05:640Z] ERROR (SOAP1) W.SOAP1.2: Attempting string, no encoding
recognized in document
[2004-07-19T16:27:05:640Z] DEEP (SOAP1) W.SOAP1.2: parse complete in 0 msecs
[2004-07-19T16:27:05:781Z] DEEP (SOAP1) W.SOAP1.2: ST_NODICT
[2004-07-19T16:27:05:781Z] DEEP (SOAP1) W.SOAP1.2: ST_FINISH
[2004-07-19T16:27:05:781Z] DEBUG (SOAP1) extractControl - edaDoc: false
[2004-07-19T16:27:05:781Z] DEBUG (SOAP1) now: 2004-07-19T16:27:05Z expires:
2004-07-20T16:27:05Z
[2004-07-19T16:27:05:781Z] DEBUG (SOAP1) W.SOAP1.2: inside isAsync() the soap
Action is ["B0100033.GetEffectiveAddressRequest#test##"]
[2004-07-19T16:27:05:781Z] DEBUG (SOAP1) W.SOAP1.2: inside isAsync() the soap
Action is [B0100033.GetEffectiveAddressRequest#test##]
[2004-07-19T16:27:05:781Z] DEBUG (SOAP1) W.SOAP1.2: checking for cached agent
[2004-07-19T16:27:05:796Z] DEBUG (SOAP1) W.SOAP1.2: pushagent: adding agent
com.ibi.iwse.XDSOAPRouter
[2004-07-19T16:27:05:796Z] DEBUG (SOAP1) W.SOAP1.2: inside worker the soap Action
is [B0100033.GetEffectiveAddressRequest#test##]
[2004-07-19T16:27:05:796Z] DEBUG (SOAP1) W.SOAP1.2: precedence: 1
[2004-07-19T16:27:05:796Z] DEBUG (SOAP1) W.SOAP1.2: precedence: 1
[2004-07-19T16:27:05:796Z] DEBUG (SOAP1) W.SOAP1.2: numagents: 1
[2004-07-19T16:27:05:812Z] DEBUG (SOAP1) W.SOAP1.2: running agent 1 name
com.ibi.iwse.XDSOAPRouter document 1
[2004-07-19T16:27:05:812Z] INFO  (manager) MGR00X01: Adding active worker:
W.SOAP1.2
[2004-07-19T16:27:05:812Z] DEBUG (SOAP1) W.SOAP1.2: <?xml version="1.0"
encoding="UTF-8" ?>
<SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
```

```
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <SOAP-ENV:Body>
      <m:GetEffectiveAddress
xmlns:m="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress">
         <m:jdeRequest type="callmethod">
            <m:callMethod name="GetEffectiveAddress">
               <m:params>
                  <m:param name="mnAddressNumber">12345</m:param>
               </m:params>
               <m:onError/>
            </m:callMethod>
         </m:jdeRequest>
      </m:GetEffectiveAddress>
   </SOAP-ENV:Body>
   <SOAPAction agentName="XDSOAPRouter"
cid="9F71FEA4C932CD8786F7388D7EF293A1">B0100033.GetEffectiveAddressRequest#test##<
/SOAPAction>
</SOAP-ENV:Envelope>
[2004-07-19T16:27:05:812Z] DEBUG (SOAP1) W.SOAP1.2: business method:
m:GetEffectiveAddress
[2004-07-19T16:27:05:828Z] DEBUG (SOAP1) W.SOAP1.2: input:
[2004-07-19T16:27:05:828Z] DEBUG (SOAP1) W.SOAP1.2: <?xml version="1.0"
encoding="UTF-8" ?><jdeRequest xmlns:xsd="http://www.w3.org/2001/XMLSchema"
type="callmethod" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><callMethod
name="GetEffectiveAddress"><params><param name="mnAddressNumber">12345</param>
</params><onError/></callMethod></jdeRequest>
[2004-07-19T16:27:07:843Z] DEBUG (SOAP1) W.SOAP1.2: Agent returned success
[2004-07-19T16:27:07:843Z] INFO  (manager) MGR00X02: Removing active worker:
W.SOAP1.2
[2004-07-19T16:27:07:843Z] DEBUG (SOAP1) W.SOAP1.2: doing docTran, docVal,
listTran for agent(1)
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: sendToAll reply to XDReply:
[protocol=http */null]
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: preemitters from doc: null
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: no preemitters, emitting
contents of doc, usestream=false encoding=UTF-8
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: writeEntity, len: 643 data:
<?xml version="1.0" encoding="UTF-8" ?><SOAP-ENV:Envelope
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><SOAP-ENV:Body><GetEffective
AddressResponse xmlns="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress:response"
cid="9F71FEA4C932CD8786F7388D7EF293A1"><jdeResponse user="JDE" type="callmethod"
environment="DV7333"><callMethod name="GetEffectiveAddress"><returnCode code="2"/>
<params><param
name="mnAddressNumber">12345</param></params></callMethod></jdeResponse></GetEffec
tiveAddressResponse></SOAP-ENV:Body></SOAP-ENV:Envelope>
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: writeString: HTTP/1.0
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: writeString: 200
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: writeString: OK
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: writeString: Content-Type:
[2004-07-19T16:27:07:859Z] DEBUG (SOAP1) W.SOAP1.2: writeString: text/xml
[2004-07-19T16:27:07:875Z] DEBUG (SOAP1) W.SOAP1.2: writeString: Content-Length:
[2004-07-19T16:27:07:875Z] DEBUG (SOAP1) W.SOAP1.2: writeString: 643
[2004-07-19T16:27:07:875Z] INFO  (SOAP1) W.SOAP1.2: W0000X13: Ended message
processing, rc=0
[2004-07-19T16:27:07:875Z] DEEP (SOAP1) W.SOAP1.2: storing used socket
```

```
[2004-07-19T16:27:07:875Z] DEBUG (SOAP1) W.SOAP1.2: entering waitforDocument
[2004-07-19T16:27:12:781Z] DEEP (SOAP1) W.SOAP1.2: cleanup: closing sockets(0)
```

10.2.2.5 Invalid Call Method

If an invalid call is made to Oracle WebLogic Server Adapter, then the following SOAP response is generated.

```
[2004-07-19T16:24:34:859Z] DEBUG (SOAP1) W.SOAP1.2: POST received
[2004-07-19T16:24:34:859Z] DEBUG (SOAP1) W.SOAP1.2: in XDSOAPHTTPWorker agentName
is [XDSOAPRouter]
[2004-07-19T16:24:34:859Z] DEBUG (SOAP1) W.SOAP1.2: before parse:
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<SOAP-ENV:Body>
<m:GetEffectiveAddress xmlns:m="urn:iwaysoftwar...[581]
[2004-07-19T16:24:34:859Z] ERROR (SOAP1) W.SOAP1.2: Attempting string, no encoding
recognized in document
[2004-07-19T16:24:34:859Z] DEEP (SOAP1) W.SOAP1.2: parse complete in 0 msecs
[2004-07-19T16:24:34:875Z] DEEP (SOAP1) W.SOAP1.2: ST_NODICT
[2004-07-19T16:24:34:875Z] DEEP (SOAP1) W.SOAP1.2: ST_FINISH
[2004-07-19T16:24:34:875Z] DEBUG (SOAP1) extractControl - edaDoc: false
[2004-07-19T16:24:34:875Z] DEBUG (SOAP1) now: 2004-07-19T16:24:34Z expires:
2004-07-20T16:24:34Z
[2004-07-19T16:24:34:875Z] DEBUG (SOAP1) W.SOAP1.2: inside isAsync() the soap
Action is ["B0100033.GetEffectiveAddressRequest#test##"]
[2004-07-19T16:24:34:875Z] DEBUG (SOAP1) W.SOAP1.2: inside isAsync() the soap
Action is [B0100033.GetEffectiveAddressRequest#test##]
[2004-07-19T16:24:34:875Z] DEBUG (SOAP1) W.SOAP1.2: checking for cached agent
[2004-07-19T16:24:34:875Z] DEBUG (SOAP1) W.SOAP1.2: pushagent: adding agent
com.ibi.iwse.XDSOAPRouter
[2004-07-19T16:24:34:875Z] DEBUG (SOAP1) W.SOAP1.2: inside worker the soap Action
is [B0100033.GetEffectiveAddressRequest#test##]
[2004-07-19T16:24:34:890Z] DEBUG (SOAP1) W.SOAP1.2: precedence: 1
[2004-07-19T16:24:34:890Z] DEBUG (SOAP1) W.SOAP1.2: precedence: 1
[2004-07-19T16:24:34:890Z] DEBUG (SOAP1) W.SOAP1.2: numagents: 1
[2004-07-19T16:24:34:890Z] DEBUG (SOAP1) W.SOAP1.2: running agent 1 name
com.ibi.iwse.XDSOAPRouter document 1
[2004-07-19T16:24:35:031Z] INFO  (manager) MGR00X01: Adding active worker:
W.SOAP1.2
[2004-07-19T16:24:35:031Z] DEBUG (SOAP1) W.SOAP1.2: <?xml version="1.0"
encoding="UTF-8" ?>
<SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <SOAP-ENV:Body>
      <m:GetEffectiveAddress
xmlns:m="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress">
         <m:jdeRequest type="callmethod">
            <m:callMethod name="GetAddress">
               <m:params>
                  <m:param name="mnAddressNumber">34518</m:param>
               </m:params>
               <m:onError/>
            </m:callMethod>
         </m:jdeRequest>
      </m:GetEffectiveAddress>
   </SOAP-ENV:Body>
   <SOAPAction agentName="XDSOAPRouter"
cid="4C0AD8398CB7A5B4DED18057D963AA44">B0100033.GetEffectiveAddressRequest#test##<
/SOAPAction>
</SOAP-ENV:Envelope>
[2004-07-19T16:24:35:031Z] DEBUG (SOAP1) W.SOAP1.2: business method:
m:GetEffectiveAddress
[2004-07-19T16:24:35:031Z] DEBUG (SOAP1) W.SOAP1.2: input:
[2004-07-19T16:24:35:031Z] DEBUG (SOAP1) W.SOAP1.2: <?xml version="1.0"
encoding="UTF-8" ?><jdeRequest xmlns:xsd="http://www.w3.org/2001/XMLSchema"
type="callmethod" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><callMethod
name="GetAddress"><params><param name="mnAddressNumber">34518</param>
      </params><onError/></callMethod></jdeRequest>
[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: Agent returned success
[2004-07-19T16:24:36:781Z] INFO  (manager) MGR00X02: Removing active worker:
W.SOAP1.2
[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: doing docTran, docVal,
listTran for agent(1)
[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: sendToAll reply to XDReply:
[protocol=http */null]
[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: preemitters from doc: null
[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: no preemitters, emitting
contents of doc, usestream=false encoding=UTF-8
[2004-07-19T16:24:36:781Z] DEBUG (SOAP1) W.SOAP1.2: writeEntity, len: 595 data:
<?xml version="1.0" encoding="UTF-8" ?><SOAP-ENV:Envelope
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><SOAP-ENV:Body><GetEffective
AddressResponse xmlns="urn:iwaysoftware:ibse:jul2003:GetEffectiveAddress:response"
cid="4C0AD8398CB7A5B4DED18057D963AA44"><jdeResponse user="JDE" type="callmethod"
environment="DV7333"><callMethod name="GetAddress"><returnCode code="99"/><params>
</params></callMethod></jdeResponse></GetEffectiveAddressResponse></SOAP-ENV:Body>
</SOAP-ENV:Envelope>
[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: HTTP/1.0
[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: 200
[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: OK
[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: Content-Type:
[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: text/xml
[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: Content-Length:
[2004-07-19T16:24:36:796Z] DEBUG (SOAP1) W.SOAP1.2: writeString: 595
[2004-07-19T16:24:36:796Z] INFO  (SOAP1) W.SOAP1.2: W0000X13: Ended message
processing, rc=0
[2004-07-19T16:24:36:796Z] DEEP (SOAP1) W.SOAP1.2: storing used socket
[2004-07-19T16:24:36:812Z] DEBUG (SOAP1) W.SOAP1.2: entering waitforDocument
[2004-07-19T16:24:42:671Z] DEEP (SOAP1) W.SOAP1.2: cleanup: closing sockets(0)
```

A Configuring J.D. Edwards OneWorld for Outbound and Inbound Processing

J.D. Edwards OneWorld enables you to specify inbound functionality for Master Business Functions (MBF).

This chapter describe how to enable outbound and inbound transaction processing in J.D. Edwards OneWorld and how to modify the `jde.ini` file for XML support. It contains the following topics:

- Section A.1, "Modifying the JDE.INI File for Outbound and Inbound Processing"
- Section A.2, "Using the GenJava Development Tool (Outbound Processing)"
- Section A.3, "Triggering J.D. Edwards OneWorld Events"

A.1 Modifying the JDE.INI File for Outbound and Inbound Processing

This section describes the settings that are required in the JDE.INI file for the XML call object kernel (outbound and inbound processing).

Open the JDE.INI file and modify the **[JDENET_KERNEL_DEF6]** and **[JDENET_KERNEL_DEF15]** sections as follows:

```
[JDENET_KERNEL_DEF6]
krnlName=CALL OBJECT KERNEL
dispatchDLLName=XMLCallObj.dll
dispatchDLLFunction=_XMLTransactionDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1

[JDENET_KERNEL_DEF15]
krnlName=XML TRANSACTION KERNEL
dispatchDLLName=XMLTransactions.dll
dispatchDLLFunction=_XMLTransactionDispatch@28
maxNumberOfProcesses=1
numberOfAutoStartProcesses=1
```

The parameters containing an underscore (_) and @28 are for Windows NT operating systems only. For other operating systems, replace the parameters with the values in the following table:

Operating System	Call Object dispatch DLLName	XML Trans dispatch DLLName
AS400	XMLCALLOBJ	XMLTRANS

Operating System	Call Object dispatch DLLName	XML Trans dispatch DLLName
HP9000B	libxmlcallojb.sl	libxmltransactions.lo
Sun or RS6000	libxmlcallojb.so	Libxmltransactions.so

Note: The J.D. Edwards installation for version B7333(XE) does not include **[JDENET_KERNEL_DEF15]**. As a result, if you are using version B7333(XE), you must manually add it to the jde.ini file. For all other J.D. Edwards versions, **[JDENET_KERNEL_DEF15]** is included with the installation.

A.2 Using the GenJava Development Tool (Outbound Processing)

This section describes how to use the GenJava development tool, which is used to create Java wrappers for accessing the J.D. Edwards business functions. The Oracle Application Adapter for J.D. Edwards OneWorld uses these wrappers to call the J.D. Edwards business functions.

This section contains the following topic:

- Section A.2.1, "Running GenJava"

J.D. Edwards provides a Java Generation tool called GenJava that you can use to expose J.D. Edwards business functions externally as Java class files. A J.D. Edwards system administrator usually runs the GenJava tool.

During GenJava operation, you must specify a library of business functions, for example CALLBSFN. GenJava creates the associated Java class files for the business functions and related data structures. GenJava also compiles the business functions, generates Java documents, and packages them into two .JAR files. One .JAR file contains Java classes and the second .JAR file contains Java documents.

For example, if the business function library you specified in GenJava is CALLBSFN, the following files are found in the <install>\system\classes directory or any user-specified directory redirected by GenJava:

- JDEJAVA_CALLBSFN.xml
- JDEJAVA_CALLBSFNInterop.jar
- JDEJAVA_CALLBSFNInteropDoc.jar

Once they are generated, these library files must be added to the CLASSPATH.

GenJava also provides access to J.D. Edwards business functions by generating pure Java interfaces for these business functions. GenJava can be generated from a thick client or a deployment server.

A.2.1 Running GenJava

GenJava is located in the <install>\system\bin32 directory. You run GenJava from the command line. There are two GenJava command options that can be used to generate the wrappers.

GenJava Command Option 1

The following command generates Java wrappers for Category 1 (Master Business Functions), Category 2 (Major Business Functions), Category 3 (Minor Business

Functions), and Category - (Uncategorized Business Functions) in the CALLBSFN library:

```
GenJava /Cat 1 /Cat 2 /Cat 3 /Cat - CALLBSFN
```

GenJava Command Option 2

The GenJava command can also be run with a JDEScript file and prompts a J.D. Edwards log on window, where you must enter a valid user ID, password, and environment.

1. Using an editor, create a new file called `AddressBook.cmd` and enter the following commands:

   ```
   define library CALLBSFN
   login
   library CALLBSFN
   interface AddressBook
   import B0100031
   import B0100019
   import B0100032
   import B0100002
   import B0100033
   build
   logout
   ```

2. Run the following GenJava command:

   ```
   GenJava /cmd .\AddressBook.cmd
   ```

3. GenJava generates the following wrapper files in Java for all of the business functions that are mentioned in the script file:

   - `CALLBSFNInterop.jar`
   - `CALLBSFNInteropDoc.jar`
   - `CALLBSFN.xml`

   Note: If there is an error while these wrapper files are generated, then ensure that the CLASSPATH is set correctly.

4. Copy the wrapper files to the repository directory.

Ensure that the following files are added to the CLASSPATH of the system where you are running GenJava:

- *`base_JAR.jar`*
- *`jdeNet_JAR.jar`*
- *`system_JAR.jar`*
- *`connector.jar`*
- *`xalan.jar`*
- *`xerces.jar`*

These files are located in the following directory:

<JDE_EnterpriseOne_Home>`\System\classes\`

In addition, ensure that the `\bin` directory of your JDK installation is included in the Java Path. For example:

```
PATH
x:\E900\system\jdk\bin
```

For more information on using GenJava, see the *J.D. Edwards EnterpriseOne Tools 8.98 Connectors Guide*.

A.3 Triggering J.D. Edwards OneWorld Events

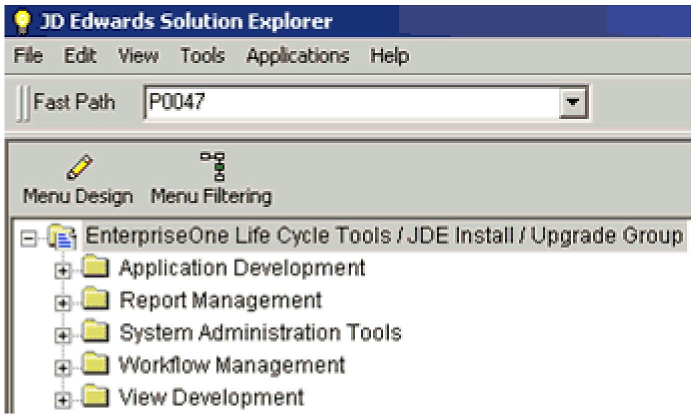
This section contains the following topics:

- Section A.3.1, "Starting the Outbound Scheduler Subsystem Process (R00460)"
- Section A.3.2, "Verifying the Subsystem Process"
- Section A.3.3, "Configuring P4210 (Sales Order) to Trigger an Event"
- Section A.3.4, "Verifying the Configuration Steps"

The flow of inbound data to third parties is controlled through the Data Export Controls application. For each transaction type and order type, one or more records can be defined with different function names and libraries.

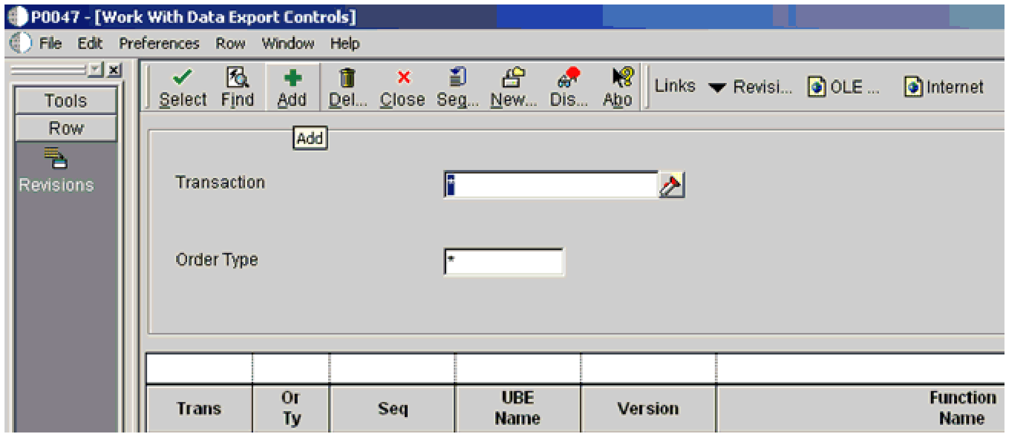
1. Type **P0047** in the Fast Path field and press **Enter** as shown in, Figure A–1 .

Figure A–1 JD Edwards Soultion Explorer


JD Edwards Solution Explorer
File Edit View Tools Applications Help
Fast Path
P0047
Menu Design
Menu Filtering
EnterpriseOne Life Cycle Tools / JDE Install / Upgrade Group
Application Development
Report Management
System Administration Tools
Workflow Management
View Development

The Work With Data Export Controls window is displayed, as shown in Figure A–2 .

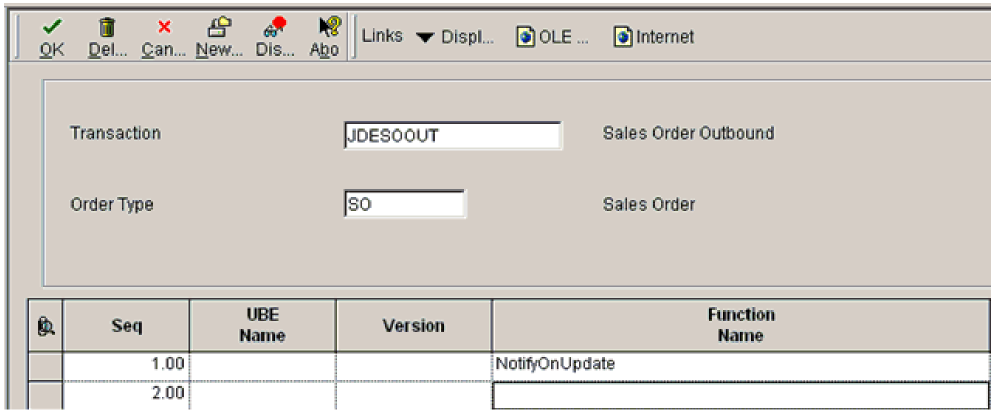
Figure A–2 Work With Data Export Controls Window


P0047 - [Work With Data Export Controls]
File Edit Preferences Row Window Help
Tools
Row
Revisions
Select Find Add Del... Close Seq... New... Dis... Abo
Links Revisi... OLE ... Internet
Add
Transaction
Order Type
Trans
Or Ty
Seq
UBE Name
Version
Function Name

2. Click **Add**.

   The Data Export Control Revisions window is displayed. Notice that the sequence (Seq) number automatically increments for each new line, as shown in Figure A–3.

Figure A–3 Data Export Control Revisions Window


OK Del... Can... New... Dis... Abo
Links Displ... OLE ... Internet

Transaction	JDESOOUT	Sales Order Outbound
Order Type	SO	Sales Order

Seq	UBE Name	Version	Function Name
1.00			NotifyOnUpdate
2.00			

3. Perform the following steps:

   a. Type **JDESOOUT** in the Transaction field.

   b. Type **SO** in the Order Type field.

   c. Type **NotifyOnUpdate** in the first row of the Function Name column.

   d. Type the absolute path to the location of the iwoevent.dll file in the first row of the Function Library column, for example:

   ```
   D:\JDEdwards\E812\DDP\Outbound\iwoevent.dll
   ```

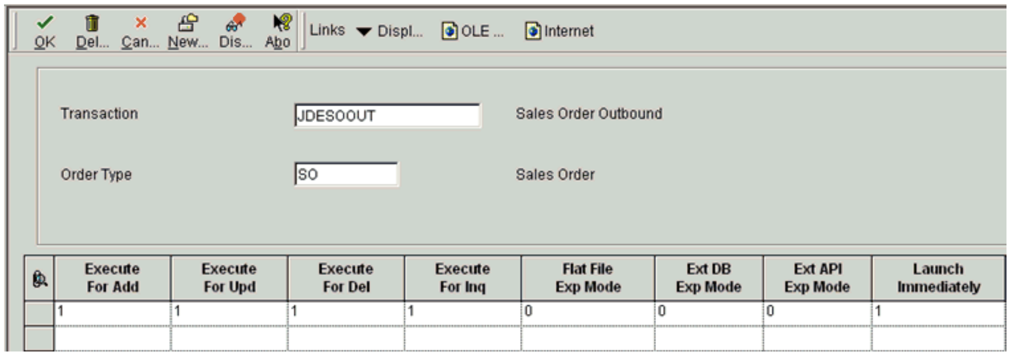
   e. Type **1** in the first row of the Execute for Add column if you want the notifications for add/insert.

   f. Make the same decision for update, delete, and inquiry and type **1** in the appropriate column.

   g. Type **1** in the Launch Immediately column to launch the Outbound Subsystem batch process (R00460).

4. Click **OK**, as shown in Figure A–4.

Figure A–4 Data Export Control Revisions Window



Transaction	JDESOOUT	Sales Order Outbound
Order Type	SO	Sales Order

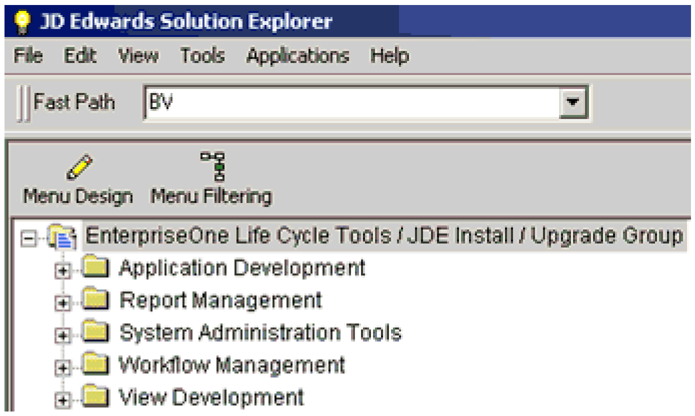
Execute For Add	Execute For Upd	Execute For Del	Execute For Inq	Flat File Exp Mode	Ext DB Exp Mode	Ext API Exp Mode	Launch Immediately
1	1	1	1	0	0	0	1

A.3.1 Starting the Outbound Scheduler Subsystem Process (R00460)

Once you have finished defining one or more records for each transaction type and order type, you must manually start the outbound scheduler subsystem process.

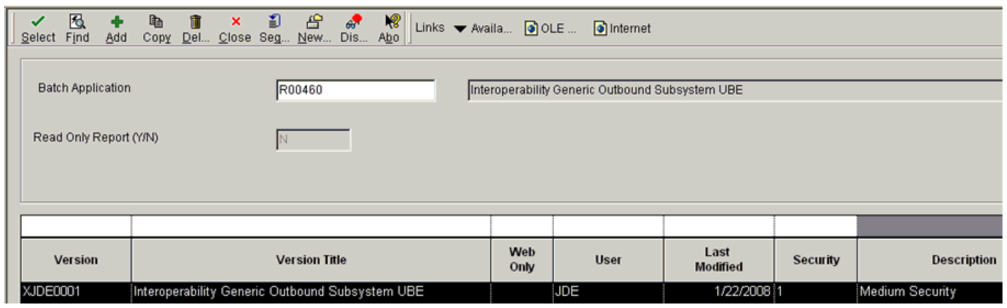
1. Type **BV** in the Fast Path field and press **Enter**, as shown in Figure A–5.

Figure A–5 JD Edwards Solution Explorer



The Work With Batch Versions - Available Versions window is displayed, as shown in Figure A–6.

Figure A–6 Work With Batch Versions - Available Versions Window

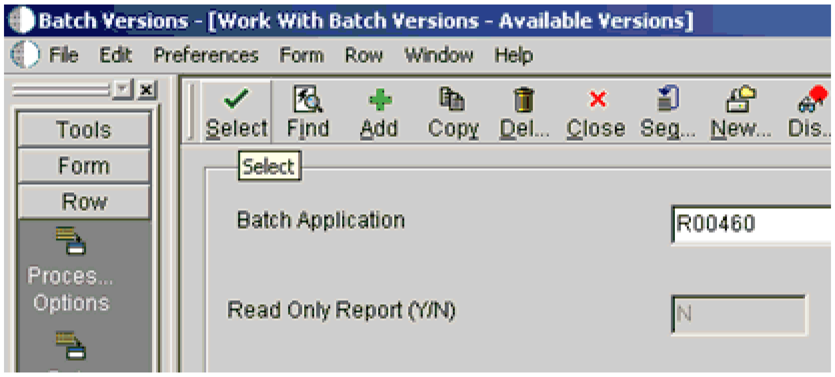


Batch Application	R00460	Interoperability Generic Outbound Subsystem UBE
Read Only Report (Y/N)	N	

Version	Version Title	Web Only	User	Last Modified	Security	Description
XJDE0001	Interoperability Generic Outbound Subsystem UBE		JDE	1/22/2008	1	Medium Security

2. Type **R00460** in the Batch Application field and click **Find**, as shown in Figure A–7.

Figure A–7 Batch Application Field


Batch Versions - [Work With Batch Versions - Available Versions]
File Edit Preferences Form Row Window Help
Tools
Form
Row
Proces... Options
Select Find Add Copy Del... Close Seq... New... Dis...
Select

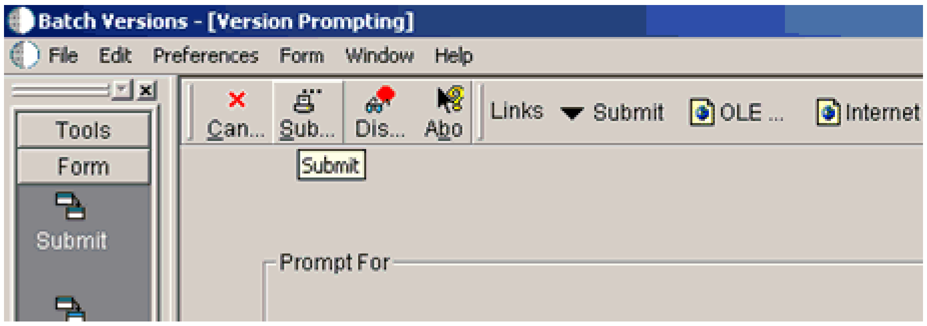
Batch Application R00460

Read Only Report (Y/N) N

3. Select **Interoperability Generic Outbound Subsystem UBE (XJDE0001)** and click **Select**.

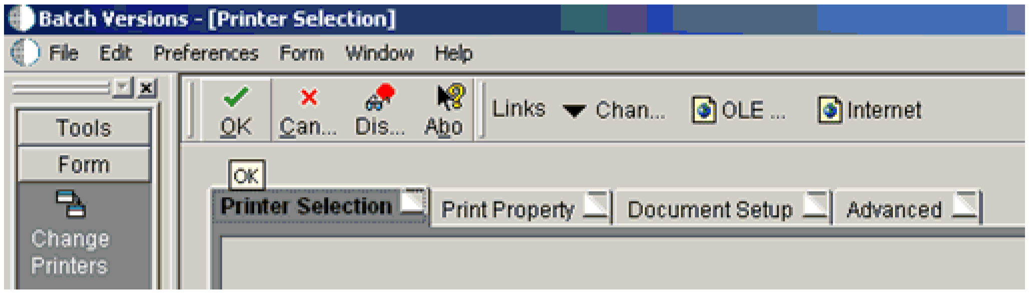
   The Version Prompting window is displayed, as shown in Figure A–8.

Figure A–8 Version Prompting Window


Batch Versions - [Version Prompting]
File Edit Preferences Form Window Help
Tools
Form
Submit
Can... Sub... Dis... Abo
Links Submit OLE ... Internet
Submit
Prompt For

4. Click **Submit**, as shown in Figure A–9.

Figure A–9 OK Option Selection


Batch Versions - [Printer Selection]
File Edit Preferences Form Window Help
Tools
Form
Change Printers
OK Can... Dis... Abo
Links Chan... OLE ... Internet
OK
Printer Selection
Print Property
Document Setup
Advanced

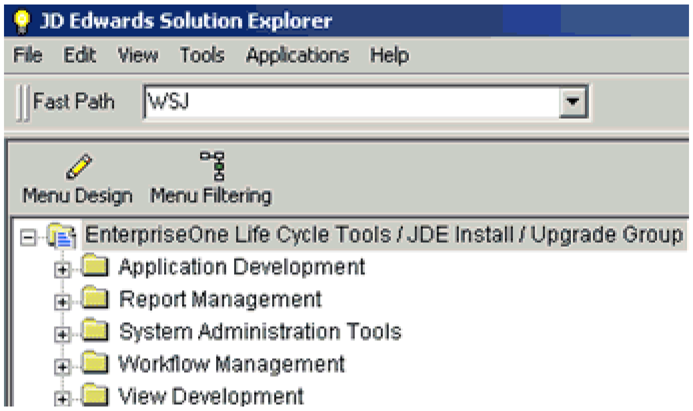
5. Navigate to the last screen and click **OK**.

A.3.2 Verifying the Subsystem Process

This section describes how to verify the outbound scheduler subsystem processs.

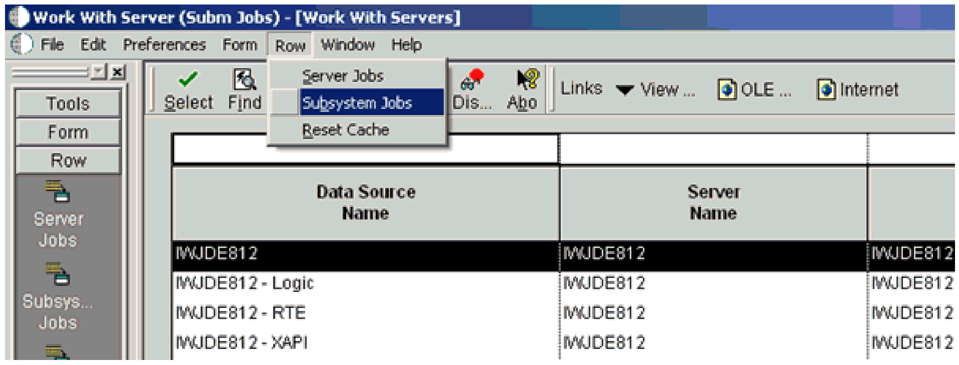
1. Type **WSJ** in the Fast Path field and press **Enter**, as shown in Figure A–10.

Figure A–10 JD Edwards Solution Explorer


JD Edwards Solution Explorer
File Edit View Tools Applications Help
Fast Path WSJ
Menu Design Menu Filtering
EnterpriseOne Life Cycle Tools / JDE Install / Upgrade Group
Application Development
Report Management
System Administration Tools
Workflow Management
View Development

The Work With Server (Subm Jobs) window is displayed, as shown in Figure A–11.

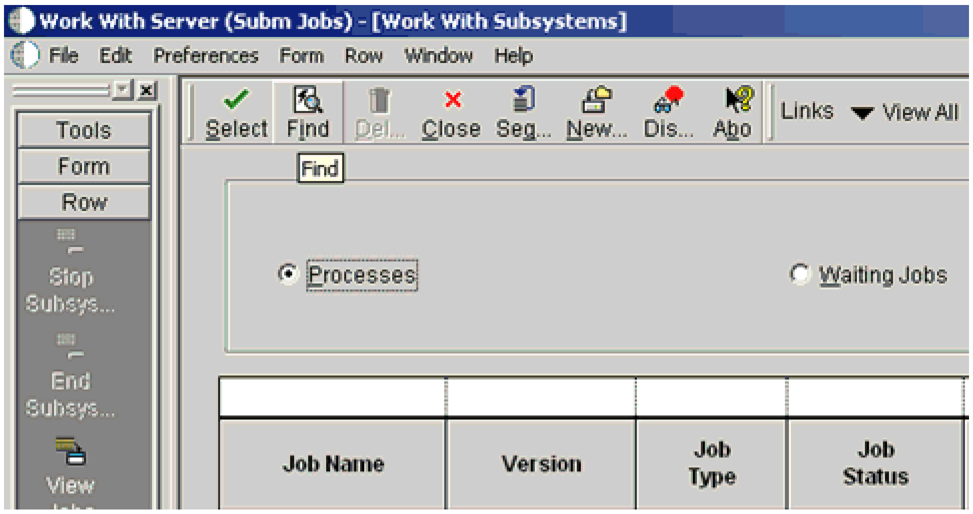
Figure A–11 Work With Server (Subm Jobs) window


Work With Server (Subm Jobs) - [Work With Servers]
File Edit Preferences Form Row Window Help
Tools
Form
Row
Server Jobs
Subsys... Jobs
Select Find
Server Jobs
Subsystem Jobs
Reset Cache
Dis... Abo
Links View ... OLE ... Internet

Data Source Name	Server Name	
IWJDE812	IWJDE812	IWJDE812
IWJDE812 - Logic	IWJDE812	IWJDE812
IWJDE812 - RTE	IWJDE812	IWJDE812
IWJDE812 - XAPI	IWJDE812	IWJDE812

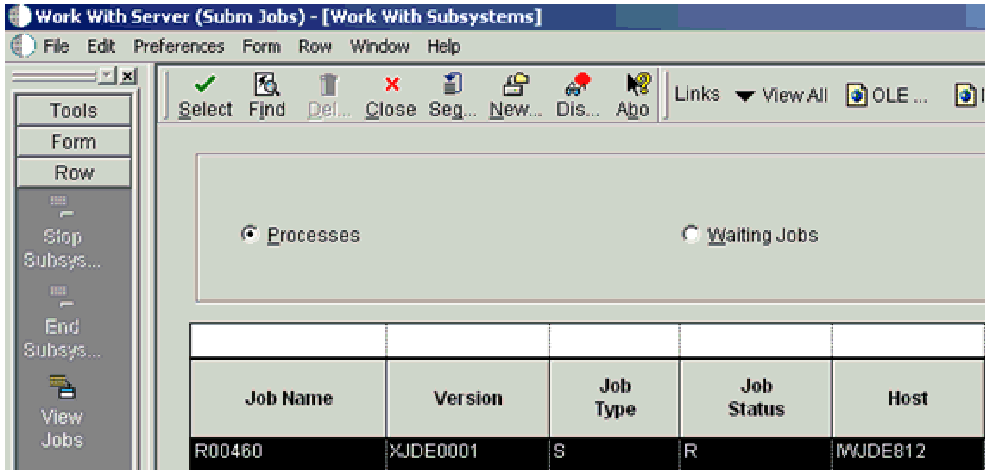
2. Select a corresponding server from the table.
3. Click **Row** from the menu bar and select **Subsystem Jobs**, as shown in Figure A–12.

Figure A–12 Find Option Selection


Work With Server (Subm Jobs) - [Work With Subsystems]
File Edit Preferences Form Row Window Help
Tools
Form
Row
Stop Subsys...
End Subsys...
View
Select Find Del... Close Seq... New... Dis... Abo
Links View All
Find
Processes
Waiting Jobs
Job Name
Version
Job Type
Job Status

4. Click **Find**, as shown in Figure A–13.

Figure A–13 Job Status Column


Work With Server (Subm Jobs) - [Work With Subsystems]
File Edit Preferences Form Row Window Help
Tools
Form
Row
Stop Subsys...
End Subsys...
View Jobs
Select Find Del... Close Seg... New... Dis... Abo
Links View All OLE ...
Processes
Waiting Jobs

Job Name	Version	Job Type	Job Status	Host
R00460	XJDE0001	S	R	IWJDE812

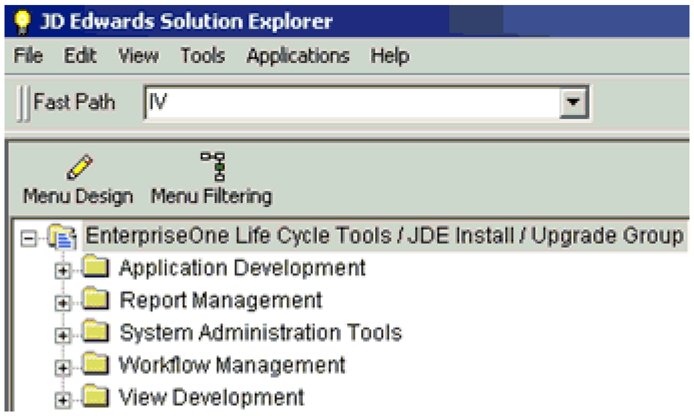
5. Verify that **R** is listed in the Job Status column.

A.3.3 Configuring P4210 (Sales Order) to Trigger an Event

This section describes how to configure a P4210 (Sales Order) to trigger an event.

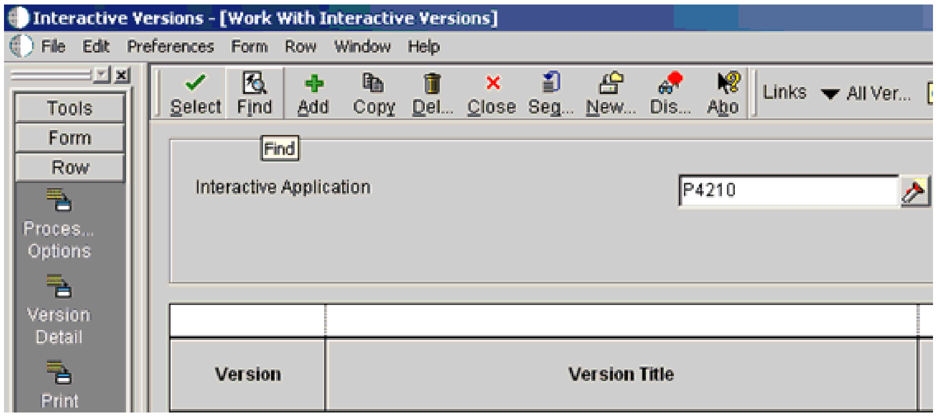
1. Type **IV** in the Fast Path field and press **Enter**, as shown in Figure A–14.

Figure A–14 JD Edwards Solution Explorer


JD Edwards Solution Explorer
File Edit View Tools Applications Help
Fast Path
IV
Menu Design
Menu Filtering
EnterpriseOne Life Cycle Tools / JDE Install / Upgrade Group
Application Development
Report Management
System Administration Tools
Workflow Management
View Development

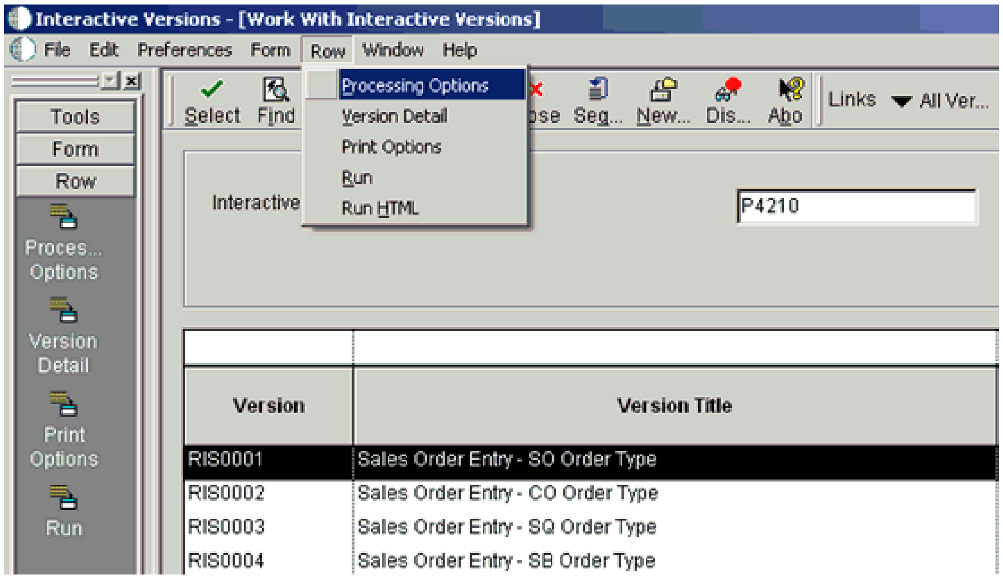
The Interactive Versions window is displayed, as shown in Figure A–15.

Figure A–15 Interactive Versions Window



2. Type **P4210** in the Interactive Application field and click **Find**, as shown in Figure A–16.

Figure A–16 Interactive Application Field



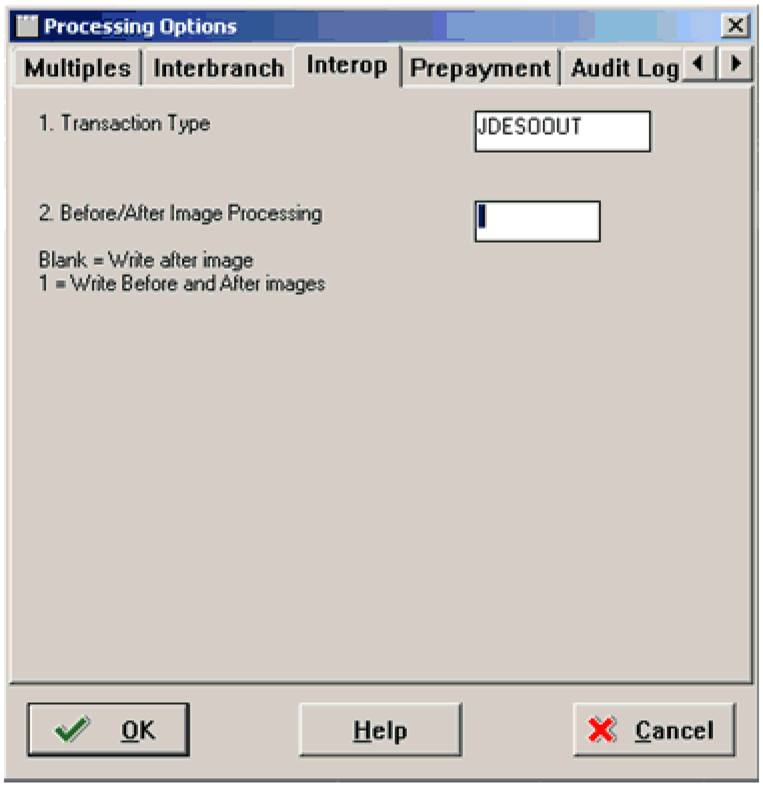
Version	Version Title
RIS0001	Sales Order Entry - SO Order Type
RIS0002	Sales Order Entry - CO Order Type
RIS0003	Sales Order Entry - SQ Order Type
RIS0004	Sales Order Entry - SB Order Type

3. Select a document version from the table, for example, **RIS0001 - Sales Order Entry - SO Order Type**.

4. Click **Row** from the menu bar and select **Processing Options**.

   The Processing Options dialog is displayed, as shown in Figure A–17.

Figure A–17 Processing Options Dialog



1. Transaction Type: JDESOOUT

2. Before/After Image Processing:

Blank = Write after image
1 = Write Before and After images

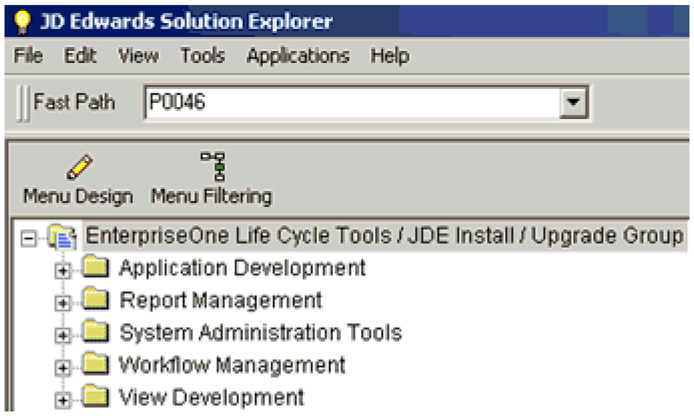
5. Click the **Interop** tab.
6. Type **JDESOOUT** in the Transaction Type field.
7. Add **Sales Order**.

A.3.4 Verifying the Configuration Steps

This section describes how to verify the configuration steps by updating F0046.

1. Type **P0046** in the Fast Path field and press **Enter**, as shown in Figure A–18.

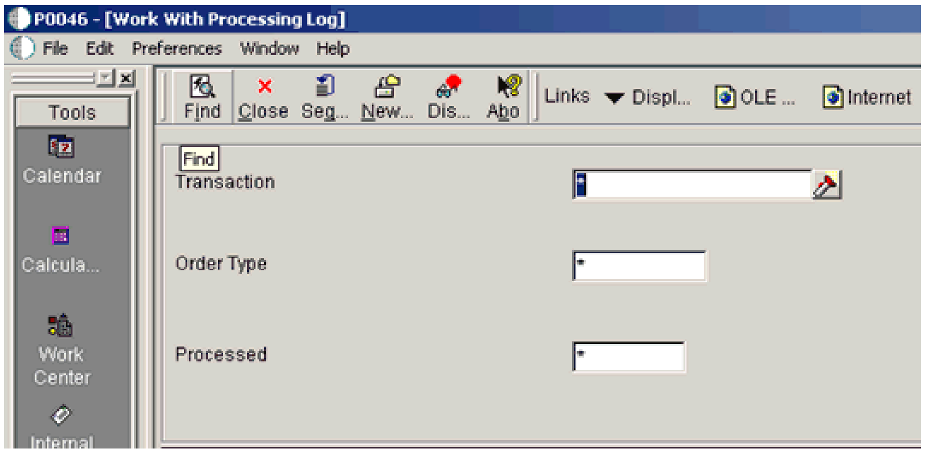
Figure A–18 JD Edwards Solution Explorer



Fast Path: P0046

The P0046 - Work With Processing Log window is displayed, as shown in Figure A–19.

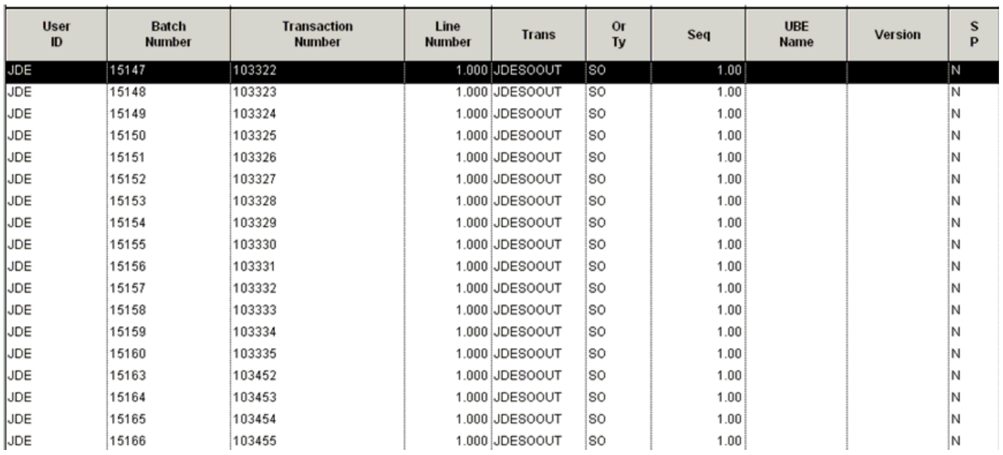
Figure A–19 Work With Processing Log Window


P0046 - [Work With Processing Log]
File Edit Preferences Window Help
Tools
Calendar
Calcula...
Work Center
Internal
Find Close Seg... New... Dis... Abo
Links Displ... OLE ... Internet
Find
Transaction
Order Type
Processed

2. Click **Find**.

   The following data is displayed, as shown in Figure A–20.

Figure A–20 Data Display Table



User ID	Batch Number	Transaction Number	Line Number	Trans	Or Ty	Seq	UBE Name	Version	S P
JDE	15147	103322	1.000	JDESOOUT	SO	1.00			N
JDE	15148	103323	1.000	JDESOOUT	SO	1.00			N
JDE	15149	103324	1.000	JDESOOUT	SO	1.00			N
JDE	15150	103325	1.000	JDESOOUT	SO	1.00			N
JDE	15151	103326	1.000	JDESOOUT	SO	1.00			N
JDE	15152	103327	1.000	JDESOOUT	SO	1.00			N
JDE	15153	103328	1.000	JDESOOUT	SO	1.00			N
JDE	15154	103329	1.000	JDESOOUT	SO	1.00			N
JDE	15155	103330	1.000	JDESOOUT	SO	1.00			N
JDE	15156	103331	1.000	JDESOOUT	SO	1.00			N
JDE	15157	103332	1.000	JDESOOUT	SO	1.00			N
JDE	15158	103333	1.000	JDESOOUT	SO	1.00			N
JDE	15159	103334	1.000	JDESOOUT	SO	1.00			N
JDE	15160	103335	1.000	JDESOOUT	SO	1.00			N
JDE	15163	103452	1.000	JDESOOUT	SO	1.00			N
JDE	15164	103453	1.000	JDESOOUT	SO	1.00			N
JDE	15165	103454	1.000	JDESOOUT	SO	1.00			N
JDE	15166	103455	1.000	JDESOOUT	SO	1.00			N

3. Search for the corresponding transaction.

   The iwoevnt.log file is created in the following directory:

   ```
   \\iwJDE812\JDEdwards\E812\DDP\system\bin32
   ```
   The iwoevent.log file is created in the outbound folder where the iwoevent.dll and iwoevent.cfg files are located. The following is an example of the event log file:

   ```
   Event call begin...
   Server time      : Tue May 27 07:23:55 2008
   userId           : JDE
   batchNumber      : 15205
   transactionNumber: 103494
   lineNumber       : 1.000000
   transactionType  : JDESOOUT
   sequenceNumber   : 1.000000
   Request xml:
   ========================
   <? xml version="1.0" encoding="UTF-8"?><jde><request><connection><dsn /><user
   /><password /><sp><proc>JDESOOUT</proc><data><ediUserId>JDE
   </ediUserId><ediBatchNumber>15205
   ```

```
</ediBatchNumber><ediTransactionNumber>103494
</ediTransactionNumber></data></sp></connection></request></jde>
========================
```

Glossary

adapter

Provides universal connectivity by enabling an electronic interface to be accommodated (without loss of function) to another electronic interface.

agent

Supports service protocols in listeners and documents.

business service

Also known as a Web service. A Web service is a self-contained, modularized function that can be published and accessed across a network using open standards. It is the implementation of an interface by a component and is an executable entity.

channel

Represents configured connections to particular instances of back-end systems. A channel binds one or more event ports to a particular listener managed by an adapter.

listener

A component that accepts requests from client applications.

port

Associates a particular business object exposed by the adapter with a particular disposition. A disposition is a URL that defines the protocol and location of the event data. The port defines the end point of the event consumption.

Index

A

access methods, 2-28
action codes, 10-3
adapter configuration
  requirements, 5-1
  updating, 3-3
adapter exceptions, 10-2 to 10-5
adapter integration
  BPEL Process Manager and, 4-1
  InterConnect and, 5-1
adapter types, 1-1, 1-6
  resource, 3-1 to 3-3
adapters, 1-1 to 1-8
  configuring, 3-3
  deploying, 1-5, 3-1 to 3-3, 4-2
  integrating, 5-1
  integrating with BPEL Process Manager, 6-1
  troubleshooting, 10-1 to 10-11
Add Channel dialog box, 4-35
Add Target dialog box, 2-6
Address Book Master file, 10-3
Address Book Revisions program, 10-3
address numbers, 10-3
agents, 1-3
alias section of iwoevent.cfg file, 2-29
aliases, 2-30
application adapter integration
  BPEL Process Manager and, 4-1
  InterConnect and, 5-1
application adapters, 1-1 to 1-8
  configuring, 3-3
  deploying, 1-5, 3-1 to 3-3, 4-2
  integrating, 5-1
  integrating with BPEL Process Manager, 6-1
  troubleshooting, 10-1 to 10-11
Application Explorer, 1-5, 2-1, 2-15, 4-7, 4-34
  application systems and, 2-5
  channels and, 2-15, 2-28
  debugging and, 10-2
  event ports and, 2-15
  J2CA configuration and, 3-3
  OracleAS Adapter J2CA and, 10-4
  schemas and, 2-10
  testing and, 10-2
  troubleshooting, 10-2
  WSDL files and, 4-34
Application parameter, 4-36
application systems, 2-15
  Application Explorer and, 2-5
  supported, 2-5

B

batch processes, 1-3
batch.log file, 2-28
BPEL Designer, 4-1, 6-1
BPEL Process Manager
  adapter integration and, 4-1
  integrating with adapters, 6-1
  OracleAS Adapter for J.D. Edwards OneWorld and, 4-1
Branch Plant Item Information, 10-3
BSE (OracleAS Adapter Business Services Engine), 1-5
  configuring, 2-2
  troubleshooting, 10-4 to 10-11
BSE deployment, 1-5
BSE URL field, 2-3
business events, 1-2
business functions, 1-8
  creating schemas for, 2-10
business services
  creating, 2-13 to 2-14
  testing, 2-14

C

CCI calls, 3-1
channel configuration parameters, 4-36
  Application, 4-36
  Host, 4-36
  Is Keep Alive, 4-36
  Is Length Prefix, 4-36
  Is XML, 4-36
  JDE Environment, 4-36
  Port Number, 4-36
  Schema style, 4-37
  Server IP address, 4-36
  Server Port, 4-36
  User id, 4-36
  User password, 4-36

channels, 2-15
  creating, 2-15, 2-16, 4-34 to 4-37
  debugging and, 4-37
  deleting, 2-28
  editing, 2-28
  ports and, 2-15
  starting, 4-37
  stopping, 4-37
  testing and, 4-37
Channels node, 4-35
channels. *See also* listeners
common section of iwoevent.cfg file, 2-29
configuration parameters, 3-4
  IWayConfig, 3-4
  IWayHome, 3-4
  IWayRepoPassword, 3-4
  IWayRepoURL, 3-4
  IWayRepoUser, 3-4
  Loglevel, 3-4
configuration requirements
  integration and, 5-1
configurations
  connecting to, 2-4
  OracleAS Adapter Business Services Engine (BSE), 2-2 to 2-4
  overwriting, 3-4
Configurations node, 2-2 to 2-3
configuring adapters, 3-3
connecting to J.D. Edwards OneWorld, 2-5 to 2-8, 10-2
connecting to OracleAS Adapter J2CA, 10-4
Connection dialog box, 2-8
connection information, 2-29, 10-2
connection parameters, 10-2
  Application, 2-8
  Hostname, 2-3
  JDE environment, 2-8
  Server IP address, 2-8
  Server Port, 2-8
  User id, 2-8
  User password, 2-8, 4-36
connections
  creating, 2-5 to 2-8
  troubleshooting, 10-2 to 10-4
Connector
  deploying to Oracle Application Server, 3-1
connector factories, 3-3
connector factory objects, 3-3
  multiple, 3-5
creating channels, 4-34 to 4-37
creating repository projects, 2-2 to 2-4
creating schemas, 2-10

D

data dictionaries, 10-3
Data Export Control table, 1-3
data queues, 1-3
Data Source Name (DSN), 2-30
database connections
  opening, 3-4
database tables, 1-3
databases
  connecting to, 3-4
  Oracle, 3-4
date ranges, 10-4
deploying adapters, 1-5, 3-1 to 3-3, 4-2
deployments
  BSE, 1-5
Description field, 2-6, 2-13
design time, 4-34
  configuring, 4-7, 4-34, 5-2
document types
  request, 1-3
  response, 1-3
  XML, 1-3
DSN (Data Source Name), 2-30

E

Eclipse. *See* JDeveloper
editing targets, 2-9
EIS (enterprise information systems), 1-3
EJB (Enterprise Java Beans), 3-1
Enterprise Connector for J2EE Connector Architecture (J2CA), 1-5, 2-2 to ??
enterprise information systems (EIS), 1-3
Enterprise Java Beans (EJB), 3-1
error messages, 10-2 to 10-11
  target systems and, 10-5
event adapters, 2-15, 2-28
event data
  receiving, 4-34
event integration, 4-34
event listeners, 2-28 to 2-29
event messages, 4-34
events, 1-2, 2-15
  configuring, 2-15
Existing Service Names list, 2-13
Export WSDL dialog box, 4-8
external listeners, 1-3

F

fault code elements, 10-5
fault string elements, 10-5
flat files, 1-3
functional modeling, 5-1

G

generating schemas, 2-10
GenJava program, 2-7
GenJava repository, 2-7

H

Host parameter, 4-36
Hostname parameter, 2-3

I

inbound J2CA services, 4-34
inbound processing, 1-4, A-1
Inspection date, 10-4
installation directories, 3-4
installation requirements
  integration and, 5-1
integration requirements, 5-1
integration scenarios, 5-1
InterConnect
  adapter integration and, 5-1
internal listeners, 1-3
interoperability framework, 1-3
  inbound processing, 1-4
  outbound processing, 1-4
invalid address numbers, 10-3
invalid records, 10-3
IP addresses, 2-8, 2-30
Is Keep Alive parameter, 4-36
Is Length Prefix parameter, 4-36
Is XML parameter, 4-36
Item Branch records, 10-3
IWayConfig parameter, 3-4
IWayHome parameter, 3-4
IWayRepoPassword parameter, 3-4
IWayRepoURL parameter, 3-4
IWayRepoUser parameter, 3-4
IWOEvent listener exit, 2-28
iwoevent.cfg file, 2-29 to 2-30
iwoevent.log file, 2-28

J

J2CA (Enterprise Connector for J2EE Connector Architecture), 1-5, 2-2 to ??
  OracleAS Adapter and, 3-1
J2CA configuration
  Application Explorer and, 3-3
J2CA resource adapters, 1-1, 3-1
J2CA services, 4-8 to 4-34
JAR files, 4-37, 10-2
Java files, 2-7
Java program clients, 3-1
J.D. Edwards OneWorld
  connecting to, 2-5 to 2-8, 10-2
J.D. Edwards OneWorld Event Listener, 2-28 to 2-30
J.D. Edwards OneWorld ThinNet API, 1-3 to 1-4, 1-8, 2-10
JDE Environment parameter, 2-8, 4-36
JDE OneWorld dialog box, 2-6
jde TransactionName, 2-30
JDEdwards node, 4-35
JDeveloper, 4-7, 6-1

L

Last Service date, 10-4
License and Method dialog box, 2-13
License field, 2-13
licenses, 2-13
list of nodes, 10-2
listener configuration files, 2-29, 2-30
listener exits, 2-28
listener types, 1-3
listeners, 1-3, 2-28 to 2-30
listeners. *See also* channels
log files, 10-1
log levels
  overwriting, 3-4
Loglevel parameter, 3-4
logon parameters, 10-2
Logon tab, 2-7

M

managed connector factories, 3-3
managed connector factory objects, 3-3
  multiple, 3-5
ManagedConnectionFactory parameter, 3-4
Master Business Functions (MBF), 1-2 to 1-5, 2-5, A-1
  executing, 2-10
MBF (Master Business Functions), 1-2 to 1-5, 2-5, A-1
  executing, 2-10
Mediator Inbound Process, 5-11
Mediator Outbound Process, 5-2
messages, 1-2 to 1-3
  event, 4-34
Method Name field, 2-13

N

Name field, 2-6
New Configuration dialog box, 2-3 to 2-4
Node list, 10-2
nodes, 2-2 to 2-3, 2-8
  Channels, 4-35
  Configurations, 2-2 to 2-3
  JDEdwards, 4-35

O

OneWorld Event Listener, 2-28 to 2-30
Oracle Application Server
  deployment of Connector to, 3-1
Oracle databases, 3-4
Oracle JDeveloper, 4-7, 6-1
Oracle JDeveloper BPEL Designer. *See* BPEL Designer, JDeveloper, or Oracle JDeveloper
Oracle's Unified Method (OUM), ix
OracleAS Adapter
  installation directory and, 3-4
  J2CA and, 3-1
OracleAS Adapter Application Explorer. *See* Application Explorer
OracleAS Adapter Business Services Engine (BSE), 1-5
  configuring, 2-2
  troubleshooting, 10-4 to 10-11
OracleAS Adapter for J.D. Edwards OneWorld
  BPEL Process Manager and, 4-1
  configuring, 2-1

deploying, 1-1, 4-2
InterConnect and, 5-1
troubleshooting, 10-1 to 10-11
OracleAS Adapter J2CA
Application Explorer and, 10-4
connecting to, 10-4
OracleAS InterConnect
adapter integration and, 5-1
outbound agents, 2-28
outbound processing, 1-4, 2-28
outbound transactions, 2-28

P

parameter types
channel configuration, 4-36
configuration, 3-4
connection, 2-8, 10-2
Parameters tab, 2-10
Password parameter, 10-2
passwords, 3-4
Port Number parameter, 4-36
port numbers, 2-8
ports, 2-15, 2-30
channels and, 2-15
Preparser tab, 4-36
Process Manager. *See* BPEL Process Manager
processes
outbound, 4-8
properties, 3-4
Protocol list, 4-35

Q

queues, 1-3

R

record identifiers, 2-28
records
processing, 10-3
repository projects
creating, 2-2 to 2-4
Web services and, 2-2
Repository tab, 2-7
request documents, 1-3
Request Schema tab, 2-10
request schemas, 2-10
resource adapters, 1-1, 1-6, 3-1 to 3-3
response documents, 1-3
Response Schema tab, 2-11
response schemas, 2-10
runtime
configuring, 5-2

S

scenarios
integration, 5-1
Schema style parameter, 4-37
schemas, 1-3
creating, 2-10
Server IP address parameter, 2-8, 4-36
server name, 2-8
Server Port parameter, 2-8, 4-36
Service Contract date, 10-4
service names, 2-13
Service Provider list, 2-3 to 2-4
Service-Oriented Architecture (SOA), 4-1
services, 1-2
creating, 2-13 to 2-14
testing, 2-14
servlets, 3-1
SOA (Service-Oriented Architecture), 4-1
SOAP agents, 10-4 to 10-5
SOAP faults, 10-4 to 10-5
SOAP requests, 10-5 to 10-8
errors and, 10-4 to 10-6, 10-8
SOAP responses, 10-4 to 10-8
software reuse, 5-1
Synchronization Type list, 4-36
synchronous access, 1-3

T

target systems, 10-5
Target Type list, 2-6
targets
connecting to, 2-6 to 2-8
defining, 2-6 to 2-8
deleting, 2-9
disconnecting from, 2-9
editing, 2-9
ThinNet API, 1-3 to 1-4, 1-8, 2-10
trace information, 10-1
trace settings, 2-30
trans section of iwoevent.cfg file, 2-29
transaction processing, 1-2, A-1
transaction tables, 1-4
troubleshooting, 10-1 to 10-11
Application Explorer, 10-2
BSE, 10-4 to 10-11
Web services, 10-4 to 10-11

U

updating adapter configuration, 3-3
User id parameter, 2-8, 4-36
User password parameter, 2-8, 4-36
UserName parameter, 10-2

W

Web service names, 2-13
Web services, 1-1
creating, 2-13 to 2-14
deploying, 10-4
integrating, 4-1
repository projects and, 2-2
testing, 2-14
troubleshooting, 10-4 to 10-11
Web Services Description Language (WSDL), 4-34

generating, 4-8, 4-37
WSDL (Web Services Description Language), 4-34
generating, 4-8, 4-37
WSDL documents, 4-1
WSDL files, 4-1, 6-1
Application Explorer and, 4-34
creating, 4-34

X

XDJdeOutboundAgent, 2-28
XML documents, 1-3
XML format, 1-3
XML messages, 1-2 to 1-3
XML schemas, 1-3
creating, 2-10
XMLInterop parameter, 4-36

Z

Z files, 2-28