JD Edwards EnterpriseOne Tools

Interoperability Reference Implementations Guide

9.2

JD Edwards EnterpriseOne Tools
Interoperability Reference Implementations Guide

9.2

Part Number: E53615-03

Contents

Preface

Welcome to the JD Edwards EnterpriseOne documentation.

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 that have purchased support 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 Information

For additional information about JD Edwards EnterpriseOne applications, features, content, and training, visit the JD Edwards EnterpriseOne pages on the JD Edwards Resource Library located at:

http://learnjde.com

Conventions

The following text conventions are used in this document:

Convention	Meaning
Bold	Boldface type indicates graphical user interface elements associated with an action or terms defined in text or the glossary.
Italics	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 examples, text that appears on a screen, or text that you enter.
> Oracle by Example	Indicates a link to an Oracle by Example (OBE). OBEs provide hands-on, step- by-step instructions, including screen captures that guide you through a process using your own environment. Access to OBEs requires a valid Oracle account.

1 Understanding Business Service Reference Implementations

Overview

Business service reference implementations are complete beginning-to-end examples of business services. These examples differ from the code samples in the Business Services Development Methodology Guide in that the business service reference implementations are fully functional. You can use the reference implementations delivered with JD Edwards EnterpriseOne Tools as a template for developing custom business services.

With JD Edwards EnterpriseOne Tools 9.1 Update 2, business services with media object operations reference implementations include a published business service project (JPR01MO1 - RI_AddressBookMediaObjectManager) and a standalone Java consumer (MOBSSVJavaClient) for JPR01MO1.

Note:

- *Published Business Service: JPR01MO1 - RI_AddressBookMediaObjectManager*
- *Java Client Reference Implementation for Media Object Operations*

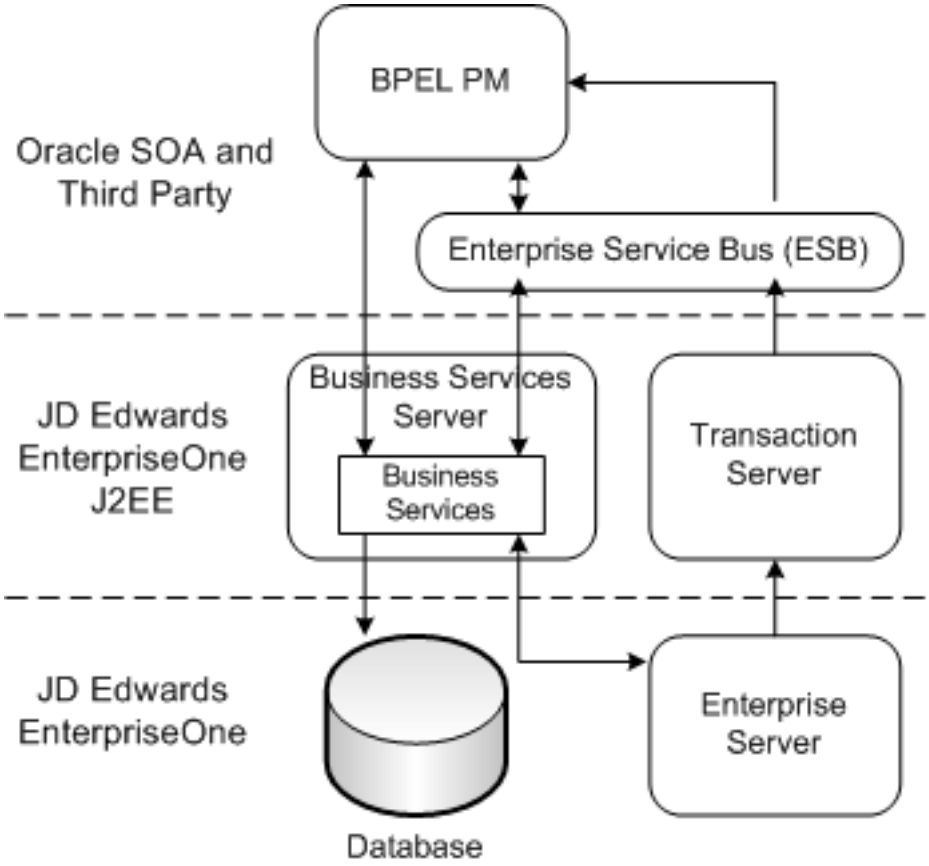
You can also use reference implementations to test the JD Edwards EnterpriseOne business services environment. The reference implementations provided enable you to test these configurations:

- JD Edwards EnterpriseOne as a service provider.
- JD Edwards EnterpriseOne as a service consumer.
- Event notification in a JD Edwards EnterpriseOne SOA environment.

The naming convention for reference implementations includes the prefix "RI" to differentiate them from actual objects that you can create for business services. Do not use this prefix when creating business services. Instead, refer to the *JD Edwards EnterpriseOne Tools Business Services Development Methodology Guide* for information about naming conventions for business service objects.

JD Edwards EnterpriseOne with Oracle's ESB and BPEL PM

If your JD Edwards EnterpriseOne integrations setup uses Oracle's ESB and BPEL PM for orchestration, you can use reference implementations shipped with JD Edwards EnterpriseOne to test this configuration. This illustration shows the JD Edwards EnterpriseOne integrations setup with Oracle ESB and BPEL PM:



Accessing Reference Implementation Javadoc

Additional detailed information about the reference implementations discussed in this guide can be found in the Javadoc. The Javadoc is stored in the installation directory of the JD Edwards EnterpriseOne Windows client that you use to develop business services (otherwise referred to as the development client).

Prerequisites

Before you can access the Javadoc on the development client, you must:

- Install the reference implementations.
- Use JD Edwards EnterpriseOne to build a package that contains the reference implementations and deploy it to the development client.

To access the reference implementation Javadoc:

1. On the development client, access the JD Edwards EnterpriseOne installation directory:

   `\\EnterpriseOneinstallpath\Pathcode\java\javadoc`
2. Double-click the index.html file to open view the Javadoc.

   The HTML file provides information about all the published business services and business services, including the reference implementations.

2 Service Provider Reference Implementations

Published Business Service Reference Implementations

Service provider reference implementations include published business service reference implementations, which are examples of JD Edwards EnterpriseOne web services that are exposed for consumption by external systems. These reference implementations contain published business services and business services. When invoked by external systems, these web services call business services, which are JD Edwards EnterpriseOne objects that perform a specific business process.

With JD Edwards EnterpriseOne Tools Release 9.1 Update 2, reference implementation JPR01MO1 - RI_AddressBookMediaObjectManager is available for testing your development environment for media object operations. The media object published business service reference implementation is a JD Edwards EnterpriseOne web service that is exposed for consumption by external systems. When invoked by external systems, this web service calls business services, which are JD Edwards EnterpriseOne objects that perform the actual media object operation.

See *Published Business Service: JPR01MO1 - RI_AddressBookMediaObjectManager* for information about this reference implementation.

You can use published business service reference implementations as a model for creating your own business services. In addition, you can run them to test the business services development environment.

Published Business Service Reference Implementation Components

The tables in this section describe the components of each published business service reference implementation. The last column in each table describes the implementation pattern along with a cross reference to the section in the *JD Edwards EnterpriseOne Tools Business Services Development Methodology Guide* that provides further details about the implementation pattern.

This section describes the components of these published business service reference implementations:

- JPR01000 - RI_AddressBookManager
- JPR01001 - RI_AddressBookTransactionManager
- JPR01002 - RI_AddressBookStagingManager
- JPR01020 - RI_CustomerManager
- JPRCUST0 - RI_CustomABManager
- JPRCUST1 - RI_CustomAddressBookManager
- JPRCUST2 - RI_CustomAddressBookStagingManager
- JPR01MO1 - RI_AddressBookMediaObjectManager

Published Business Service: JPR01000 - RI_AddressBookManager

This table describes the components of JPR01000 - RI_AddressBookManager:

> **Note:** When you develop a business service, you need to test it as you develop it. For JPR01000 - RI_AddressBookManager, JD Edwards EnterpriseOne provides a test object that makes use of the functionality of the RI_AddressBookManager service. It is included in the reference implementation as an example of how a developer codes test objects. The JTR87010 test object is automatically delivered with JD Edwards EnterpriseOne and requires no additional configuration.

Business Service	Method	Description	Implementation Pattern and Reference to Business Services Development Methodology Guide
JR010010 - RI_AddressBookProcessor	addAddressBook	Adds an address book record with phone number information.	A business service calling a BSFN, utility, or another business service. See: Calling Business Functions. Calling Other Business Services.
JR010010 - RI_AddressBookProcessor	addAddressBookAndParent	Adds an address book record with phone information and creates a parent record to demonstrate an explicit transaction.	An explicit transaction with a manual connection. See Using an Explicit Transaction.
JR010030 - RI_PhonesProcessor	addPhones	Adds phones from a collection of phone information.	A business service that is not exposed publicly in a web service; it is used only by other business services. See Calling Other Business Services.
JR010040 - RI_AddressBookQueryProce	getAddressBook	Queries the V0101XPI view and returns address book information based on data passed in.	Query of "Select" database operation and auto commit transaction. See: Creating a Query Database Operation Business Service. Understanding Transaction Processing.

Published Business Service: JPR01001 - RI_AddressBookTransactionManager

This table describes the components of JPR01001 - RI_AddressBookTransactionManager:

Business Service	Method	Description	Implementation Pattern and Reference to Business Services Development Methodology Guide
JR010011 - RI_AddressBookTransactionPr	addAddressBook	Adds an address book record with phone number information.	A business service calling a business function, utility, or another business service. See: Calling Business Functions. Calling Other Business Services.
JR010011 - RI_AddressBookTransactionPr	addAddressBookandBusinessU	An explicit transaction business function and database operation. See Using an Explicit Transaction.	no value
JR010011 - RI_AddressBookTransactionPr	addAddressBookAndParent	Adds an address book record with phone information and creates a parent record to demonstrate an explicit transaction.	An explicit transaction business function. See Using an Explicit Transaction.
JR010040 - RI_AddressBookQueryProcess	getAddressBook	Queries the V0101XPI view and returns address book information based on data passed in.	Select database operation and auto commit transaction. See: Creating a Query Database Operation Business Service. Understanding Transaction Processing.

Published Business Service: JPR01002 - RI_AddressBookStagingManager

This table describes the components of JPR01002 - RI_AddressBookStagingManager:

Business Service	Method	Description	Implementation Pattern and Reference to Business Services Development Methodology Guide
JR010050 - RI_AddressBookStagingProc	insertAddressBookStaging	Inserts records into the Address Book Z table F0101Z2	Insert database operation. See Creating an Insert Database Operation Business Service.
JR010050 - RI_AddressBookStagingProc	updateAddressBookStaging	Updates records in the Address Book Z table F0101Z2	Update database operation. See Creating an Update Database Operation Business Service.
JR010050 - RI_AddressBookStagingProc	deleteAddressBookStaging	Deletes record from the Address Book Z table F0101Z2	Delete database operation. See Creating a Delete Database Operation Business Service.

Published Business Service: JPR01020 - RI_CustomerManager

This table describes the components of JPR01020 - RI_CustomerManager:

Business Service	Method	Description	Implementation Pattern and Reference to Business Services Development Methodology Guide
JR010023 - RI_CustomerCreditQueryPrc	getCustomerCreditInfo	Retrieves customer credit information based on input of customer number.	A query performed by a business function. See Value Object Overview.
JR010020 - RI_EntityProcessor (Utility)	processEntity	A re-useable business service. See Calling Other Business Services.	no value

Published Business Service: JPRCUST0 - RI_CustomABManager

This published business service exposes this method:

customAddAddressBook
An example of a published business service calling another published business service or a customer's extension of a delivered RI_AddressBookManager published business service calling the addAddressBook method.

See *"Customizing a Published Business Service" in the JD Edwards EnterpriseOne Tools Business Services Development Methodology Guide* .

Published Business Service: JPRCUST1 - RI_CustomAddressBookManager

This published business service exposes this method:

callAddABInsertBU
An example of a published business service calling another published business service or a customer's extension of a delivered RI_AddressBookTransactionManager published business service calling the addAddressBookAndBusinessUnit method.

See *"Customizing a Published Business Service" in the JD Edwards EnterpriseOne Tools Business Services Development Methodology Guide* .

Published Business Service: JPRCUST2 - RI_CustomAddressBookStagingManager

This published business service exposes this method:

callInsertABStaging
An example of a published business service calling another published business service or a customer's extension of a delivered RI_AddressBookStagingManager published business service calling the insertAddressBookStaging method.

See *"Customizing a Published Business Service" in the JD Edwards EnterpriseOne Tools Business Services Development Methodology Guide* .

Published Business Service: JPR01MO1 - RI_AddressBookMediaObjectManager

This table describes the components of JPR01MO1- RI_AddressBookMediaObjectManager:

Business Service	Method	Description	Implementation Pattern and Reference to Business Services Development Methodology Guide
JR010MO1	addAddressBookMO	Adds an address book record along with media objects.	A business service calling a media object operation. See Creating an Insert Media Object Operation Business Service.
JR010MO1	getAddressBookMO	Retrieves an address book record along with any media objects.	A business service calling a media object operation. See Creating a Select Media Object Operation Business Service.
JR010MO1	deleteAddressBookMO	Deletes media objects for an address book number.	A business service calling a media object operation. See Creating a Delete Media Object Call Business Service.

Prerequisites

These prerequisites enable you to download, install, and use reference implementation JPR01MO1:

- Download and install the ESU for published business service reference implementation JPR01MO1 and internal business service reference implementation JR010MO1 to the deployment server.

  The ESUs that correspond to the following bug numbers must be installed for testing media object operations for reference implementations JPR01MO1 and JR010MO1:

  - EnterpriseOne Applications Release 9.0: Bug 14125712
  - EnterpriseOne Applications Release 9.1: Bug 14125693

- On the EnterpriseOne development client machine, using OMW, first search for the JPR01MO1 object and then perform a Get operation in OMW to get the JPR01MO1 object locally to <E1_Install_Path>\DV900\java\source \oracle\e1\bssv\JPR010MO1 location.
- On the EnterpriseOne development client machine, using OMW, search for the JR010MO1 object and then perform a Get operation in OMW to get the JR010MO1 object locally to <E1_Install_Path>\DV900\java\source \oracle\e1\bssv \JR010MO1 location.

- If you want to test the reference implementation using a standalone business services server, you can build a JAX-WS business service package and deploy the package to your business services server.

  Note:

  - *"Assembling JAX-WS Based Business Services for Package Build" in the JD Edwards EnterpriseOne Tools Package Management Guide .*
  - *"Understanding the Build Process for a JAX-WS Based business Service Package" in the JD Edwards EnterpriseOne Tools Package Management Guide .*
  - *"Deploying the Package to the Business Services Server" in the JD Edwards EnterpriseOne Tools Package Management Guide .*

Creating, Deploying, and Testing Reference Implementation JPR01MO1

You can use reference implementation JPR01MO1 as a model to create a custom published business service that uses media object operations. After you create your custom published business service, you deploy it to the JDeveloper 11g Integrated WebLogic Server for testing. You can verify that your business service server is working properly by testing the JPR01MO1 reference implementation in the JDeveloper 11g Integrated WebLogic Server.

See *"Understanding MTOM-Based Media Object Web Services" in the JD Edwards EnterpriseOne Tools Business Services Development Guide for more information about creating, deploying, and testing media object web services.*

Accessing Business Service Reference Implementations

You can access business service reference implementations through Object Management Workbench (OMW) in the JD Edwards EnterpriseOne Windows client.

Prere quisites

Install JDeveloper.

To access a business service reference implementation:

Access OMW.

1. In OMW, create a project.
2. Search for a reference implementation and then add it to your project.
3. Select the reference implementation and then click the Design button.
4. On the Business Function Design form, click the Design Tools tab.
5. Click Invoke JDeveloper.

See *"Working with Business Services" in the JD Edwards EnterpriseOne Tools Object Management Workbench Guide* for more information.

BPEL Reference Implementations for Business Services Running on WebSphere Application Server

Oracle provides the following BPEL reference implementations to test that a JD Edwards EnterpriseOne business service running on the IBM WebSphere Application Server can be consumed by Oracle's BPEL PM:

BPELProcessForWebSphereAddABAndGetCustomerCredit

In addition, you can use the BPEL reference implementations as a template for creating a BPEL flow that calls a business service.

The BPEL reference implementations use these JD Edwards EnterpriseOne published business service reference implementations:

- JPR01020: RI_CustomerManager

This reference implementation contains the getCustomreCredit method, which retrieves customer credit information based on input of customer number.

- JPR01000: RI_AddressBookManager

This reference implementation contains the addAddressBook method, which adds an address book record with phone number information.

These reference implementations must be installed before you run a BPEL reference implementation. See *Published Business Service Reference Implementations* in this guide for more information.

Accessing the BPELProcessForAddABAndGetCustomerCredit Reference Implementation

The files that you need to install the BPEL reference implementation are in a Zip file located in this JD Edwards EnterpriseOne installation directory:

\\EnterpriseOneInstallDirectory\System\Classes\samples\BPEL

The Zip file includes a readme file that contains a list of prerequisites and provides instructions on how to install and run the reference implementation from the BPEL Admin Console.

> **Note:** You cannot run this reference implementation from Object Management Workbench in JD Edwards EnterpriseOne because it is not a JD Edwards EnterpriseOne object.

BPEL Reference Implementations for Business Services Running on a WebLogic Server

Oracle provides the following BPEL reference implementations to test that a JD Edwards EnterpriseOne business service running on the Oracle WebLogic Server can be consumed by Oracle's BPEL PM:

BPELForWeblogicGetCustomerCreditInfo

In addition, you can use the BPEL reference implementations as a template for creating a BPEL flow that calls a business service.

The BPEL reference implementation use this JD Edwards EnterpriseOne published business service reference implementation:

JPR01020: RI_CustomerManager

This reference implementation contains the getCustomerCredit method, which retrieves customer credit information based on input of customer number.

This reference implementations must be installed before you run a BPEL reference implementation. See *Published Business Service Reference Implementations* in this guide for more information.

Accessing the BPELForWeblogicGetCustomerCreditInfo Reference Implementation

The files that you need to install the BPEL reference implementation are in a Zip file located in this JD Edwards EnterpriseOne installation directory:

\\EnterpriseOneInstallDirectory\System\Classes\samples\BPEL11g

The Zip file includes a readme file that contains a list of prerequisites and provides instructions on how to install and run the reference implementation from the BPEL Admin Console.

AddressBookStagingESBReference Implementation

The AddressBookStagingESB reference implementation is an example of an ESB flow that calls a business service in JD Edwards EnterpriseOne. You can use this ESB reference implementation to test that a JD Edwards EnterpriseOne business service can be consumed by Oracle's ESB. In addition, you can use this reference implementation as a template for creating an ESB flow that consumes a JD Edwards EnterpriseOne business service.

This reference implementation uses the methods in the JPR01002 business service that:

- Insert address book records.
- Update address book records.
- Delete address book records.

Accessing the AddressBookStagingESB Reference Implementation

The files that you need to install and run this reference implementation are in a Zip file located in this JD Edwards EnterpriseOne installation directory:

\\EnterpriseOneInstallDirectory\System\Classes\samples\ESB

The Zip file includes a readme file that provides instructions on how to install and run the AddressBookStagingESB reference implementation.

> **Note:** You cannot run this reference implementation from Object Management Workbench in JD Edwards EnterpriseOne because it is not a JD Edwards EnterpriseOne object.

MediatorForWeblogicGetCustomerCreditInfo Reference Implementation

The MediatorForWeblogicGetCustomerCreditInfo reference implementation is an example of a Mediator flow that calls a business service in JD Edwards EnterpriseOne. You can use this Mediator reference implementation to test that a JD Edwards EnterpriseOne business service can be consumed by Oracle's Mediator component. In addition, you can use this reference implementation as a template for creating a Mediator flow that consumes a JD Edwards EnterpriseOne business service.

The Mediator reference implementation uses this JD Edwards EnterpriseOne published business service reference implementation:

JPR01020: RI_CustomerManager

This reference implementation contains the getCustomerCredit method, which retrieves customer credit information based on input of customer number.

Accessing the MediatorForWeblogicGetCustomerCreditInfo Reference Implementation

The files that you need to install and run this reference implementation are in a Zip file located in this JD Edwards EnterpriseOne installation directory:

\\EnterpriseOneInstallDirectory\System\Classes\samples\ESB11g

The Zip file includes a readme file that provides instructions on how to install and run the MediatorForWeblogicGetCustomerCreditInfo reference implementation.

3 Service Consumer Reference Implementations

Testing the Business Services Server as a Web Service Consumer

This section provides an overview, lists prerequisites, and discusses how to:

- Configure the reference implementations used for testing the Business Services Server.
- Run the RI AddressBook test program.

Overview

JD Edwards EnterpriseOne provides the RI AddressBook program (P954001), a test program that enables you to verify that the Business Services Server is set up correctly as a web service consumer. To test the configuration, you use P954001 to request address book information for a particular user in JD Edwards EnterpriseOne.

P954001 is not programmed to retrieve address book information directly from the Address Book Master table (F0101); in other words, it is not built on F0101. Instead, to retrieve the address book information, P954001 is programmed to make a call to a web service that is deployed on the Business Services Server. The web service is responsible for retrieving the associated address book data, based on the address book number that was entered in P954001. The web service sends the information back to P954001. If the information for the address book record appears in P954001, then the Business Services Server is set up correctly for consuming web services.

P954001 uses one of the following web service reference implementations to test JD Edwards EnterpriseOne as a web service consumer:

- JRH90I30 RI_ABWebServiceProcessor
- JRH90I31 RI_WAS_ABWebServiceProcessor

The reference implementation that is used depends on whether the Business Services Server is deployed on Oracle Application Server or WebSphere Application Server.

Both reference implementations expose the getAddressBook method, which is used by the reference implementations to call a web service. This method calls the third-party web service to look up an address book number. For the purpose of this reference implementation, the third-party web service being called is the JD Edwards EnterpriseOne Addressbook web service.

Prerequisites

Before you perform the tasks in this section, you must:

- Install these reference implementations:
    - JRH90I30 RI_ABWebServiceProcessor

- JRH90I31 RI_WAS_ABWebServiceProcessor

- Build and deploy the Development Business Services Server.

  See *"Deploying a Development Business Services Server" in Appendix B of the JD Edwards EnterpriseOne Tools Business Services Development Guide* .

- Set up an OCM record for enterprise server to point to the Business Services Server.

  See *"Setting Up OCM for Business Functions Calling Business Services" in the JD Edwards EnterpriseOne Tools Business Services Development Guide* .

Configuring the Reference Implementations Used for Testing the Business Services Server

In addition to the prerequisites, you must perform these tasks before you can use the P954001 test program:

- Set the business service property value.

  The system uses this property value to determine which reference implementation is used, depending on whether the Business Services Server is deployed on Oracle Application Server or WebSphere Application Server.

- Create a softcoding record for the reference implementation.

  A softcoding record contains the actual softcoding value that a business service uses to call an external web service.

To set the business service property value:

In the JD Edwards EnterpriseOne web client, enter P951000 in the Fast Path to access the Launch Service Property program.

1. On Work with Business Service Property, select the All option for Level.
2. In the Key search field in the grid, enter JRH90I30_APP_SERVER_OAS_OR_WAS and click Find.
3. Select the Service Property record and then click the Select button.
4. On Modify Business Service Property, enter OAS or WAS in the Value field, depending on the type of operating system the Development Business Services Server is deployed on.
5. Select the System option for Level.
6. Click OK to save the record.

To create a soft coding record for the reference implementation:

Use the soft coding template delivered in JD Edwards EnterpriseOne to create a soft coding record.

In the JD Edwards EnterpriseOne web client, enter P954000 in the Fast Path.

1. On Work with Web Service Soft Coding Records, click Add.
2. On Add Web Service Soft Coding Record, complete these fields:

Field	Description
User / Role	Enter a valid user or role.

Environment	Enter an environment.
Template Name	Enter E1_JRH90I30.

3. Click the Populate Soft Coding Value button.
4. Locate the <stub-property><value> node in the XML and replace the value from the template with:

   http://servermachinename.domain:port/environment/RI_AddressBookManager
5. Locate username-token and change the username to a user that has credentials to consume the web service.
6. In the Mask Fields grid, enter a password for this user in the Mask Value text field.
7. Click OK to save the record.

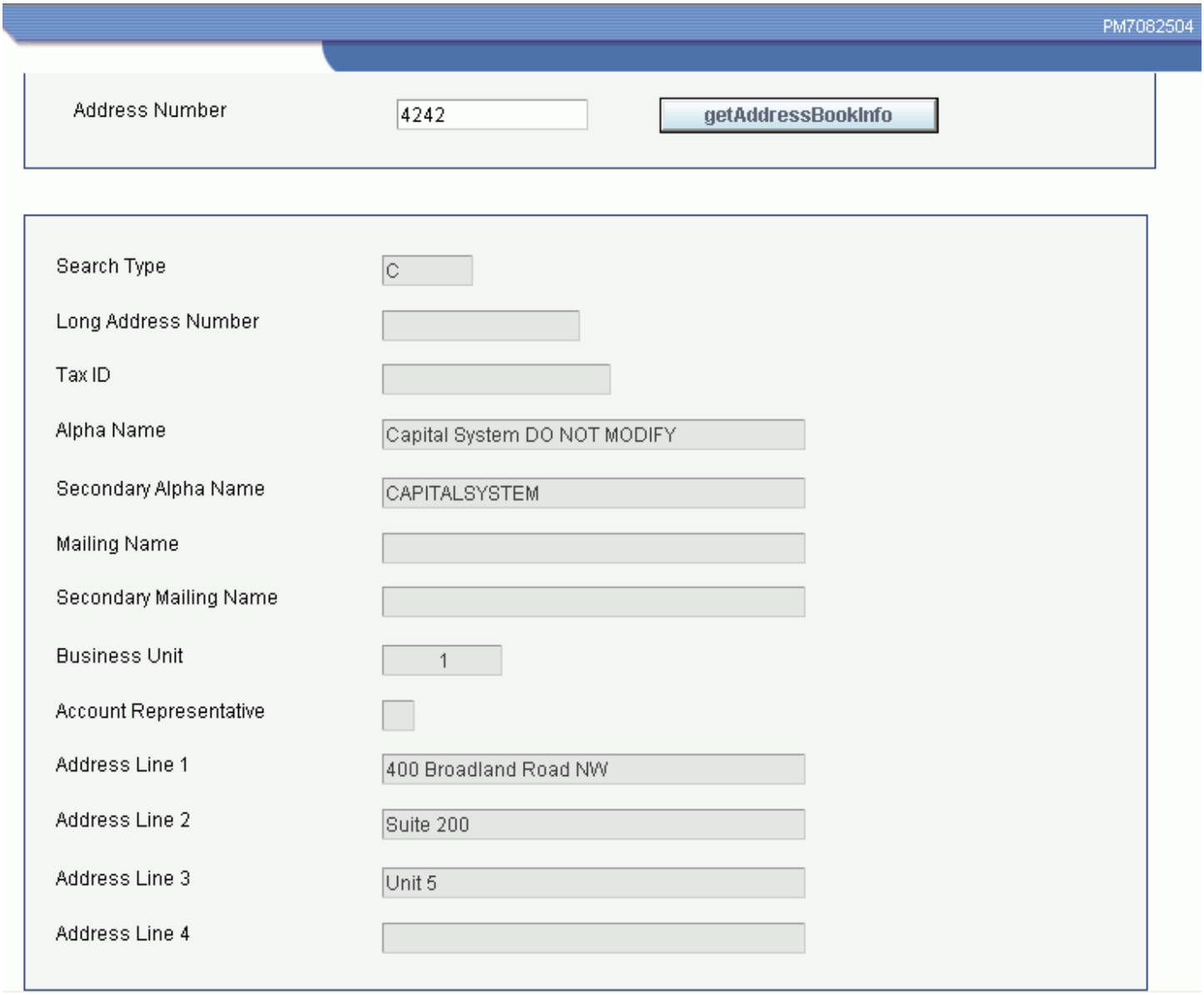
Running the RI AddressBook Test Program

In the JD Edwards EnterpriseOne web client, enter P954001 in the Fast Path to access the RI AddressBook program.

Enter 4242 in the Address Number field and then click the getAddressBookInfo button.

The RI AddressBook program calls a business function that calls a business service. The business service consumes the AddressBook web service. The AddressBook web service retrieves the address book record for the address book number that was entered. If the information for the address book record appears in the application form, then the business services server has been configured correctly.

The following example shows a returned address book record in the RI Address Book program:



Java Test Client Reference Implementations

You can install, configure, and run reference implementations for Java test clients that call business services through a web service interface. Oracle provides the WASCustomerManagerJavaClient reference implementation to test this functionality on the WebSphere Application Server.

The Java test client reference implementation invokes the JPR01020: RI_CustomerManager business service reference implementation. This reference implementation contains the getCustomerCreditInfo method, which retrieves customer credit information based on input of customer number.

Accessing the Java Test Client Reference Implementations

The files that you need to install the Java test client reference implementations are in a Zip file located in this JD Edwards EnterpriseOne installation directory:

```
\\EnterpriseOneInstallDirectory\System\Classes\samples\BSSVJavaClient
```

The Zip file includes a readme file that contains a list of prerequisites and provides instructions on how to install and run the reference implementation.

HTTP Request Reference Implementations

The HTTP request reference implementations are examples of how business services can enable JD Edwards EnterpriseOne to communicate with a third-party system using HTTP POST. The HTTP request reference implementations include:

- JRH90I32

  This reference implementation is an example a business service that is used to post XML data to a third-party site.

- JPRH90I0

  This reference implementation is an example of a published business service that takes the HTTP request from the third party and checks for authentication and authorization of the credentials in the message. JPRH90I0 sends a message to the JD Edwards EnterpriseOne web client instance that is waiting for a response from the third-party site.

The following scenario depicts what would happen if these reference implementations were deployed on the Business Services Server in a JD Edwards EnterpriseOne web service consumer setup:

1. A JD Edwards EnterpriseOne web client user runs an application that requests information from a third-party site.
2. The web client invokes a business function that calls the JRH90I32 business service.
3. JRH90I32 runs and posts XML data to the third-party site.

   The XML data contains callback information that the third-party system can use to send an asynchronous reply to the request.
4. The Business Service Server waits for the third-party site to send an HTTP message to the Business Service Server.
5. The third-party site processes the information that was provided in the JRH90I32 RI and sends the Business Service Server an HTTP request.
6. JPRH90I0 receives the HTTP request from the third party and checks for authentication and authorization of the credentials in the message.
7. Upon successful authentication and authorization, JPRH90I0 sends a message that contains the requested information from the third-party site to the JD Edwards EnterpriseOne Web client instance.

Note: You can only use HTTP request reference implementations as examples for creating your own custom business services.

Downloading and Deploying the JDeveloper 11g Reference Implementation Business Service for WebLogic Server

This section provides an overview of the JDeveloper 11g Business Service reference implementation for the WebLogic Server and discusses how to:

- Download the reference implementation.
- Deploy the reference implementation to the WebLogic Server.

Understanding the JDeveloper 11g Reference Implementation for WebLogic Server Configuration

This reference implementation (JRH90I33) is a JDeveloper 11g business service for the WebLogic Server, specifically for the consumer scenario. If you are planning to install JDeveloper 11g and build web services for the WebLogic Server, you can download and install the business service reference implementation from the Update Center on My Oracle Support.

The business service code is built from JDeveloper 11g and should be used only with Tools 8.98 Update 3 release and later. Applying this reference implementation without the corresponding tools release will disable the business service package build on the WebSphere Application Server.

Downloading the Reference Implementation

You download the JDeveloper 11g reference implementation business service for a WebLogic Server from the Update Center on My Oracle Support. From the Update Center, search for one of these configuration files to download to your system:

- TLRI900001 for the E900 release
- TLRI812001 for the 812 release
- TLRI81A001 for the 811 SP1 release

Use Change Assistant to deploy the configuration to your target environment.

Searching for and Downloading the Reference Implementation

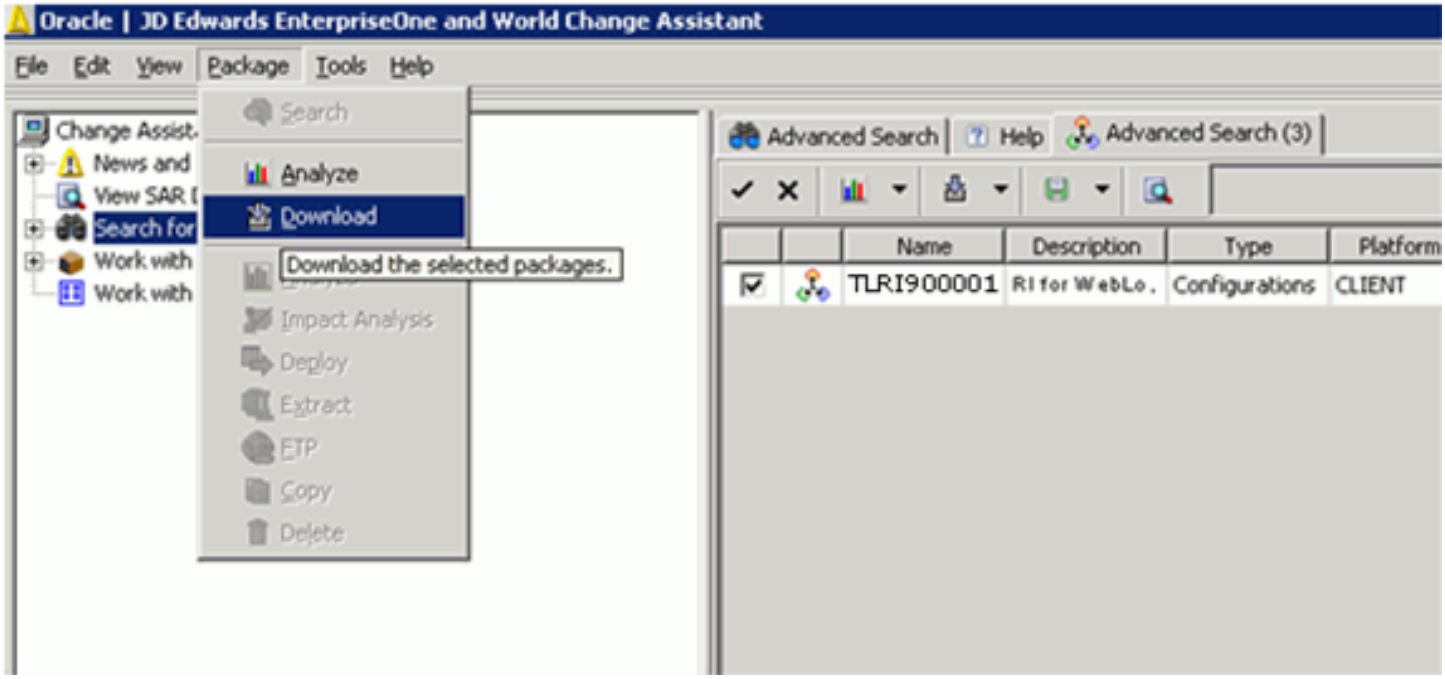
You use the JD Edwards EnterpriseOne Change Assistant application, which is available on the Update Center on My Oracle Support, to download the configuration file. The following tasks explain how to search for, download and deploy the configuration file for E900. You use the same steps to download and deploy 812 and 811SP1 configuration files.

To search for and download the reference implementation:

Access JD Edwards EnterpriseOne Change Assistant.

1. In the navigation area, select the Search for Packages node, and click the Advanced Search tab.
2. In the Search by Types area, select Configurations from the Types drop-down list.
3. In the Search by Names area, enter the Configuration File Name (for example, TLRI900001) or part of the filename with a wildcard (*), and then click Search.

   Change Assistant searches for matches across available updates and displays the top 20 matches in a search results grid in the Search area.
4. Select the check box next to the configuration file (TLRI900001) from the search results.



5. From the Package menu, click Download.

   Change Assistant downloads the selected configuration file to your downloads folder. You also download ESUs from the Update Center to this folder.

To deploy the configuration file:

On the JD Edwards EnterpriseOne Change Assistant.

1. Select the check box for the configuration file name that you want to deploy.
2. From the Package menu, click Deploy.
3. Change Assistant prompts you with the Environment Selection form.
4. In the Deploying To area of the Environment Selection form, select one or more environments to which you want to deploy the configuration file.
5. Click OK.
6. Click Yes or No on the Warning message.

   When you click Yes, the selected configuration is deployed to the selected environments.

Change Assistant displays the Change Assistant Deployment Summary form that indicates that the deployment was a success or failure.

To test that the deployment was successful, check for the presence of the JRH90I33.jar file under <Deployment Server>\E900\DV900\Java\Sbfjars.

Also look in the Object Librarian master table (F9860) where Object Name JRH90I33 should exist.

Deploying the Reference Implementation to the WebLogic Server

When the reference implementation (JRH90I33) is available on your deployment server, you use OMW to add it to your JDeveloper project. You then use a JD Edwards EnterpriseOne soft coding record to deploy the reference implementation to the WebLogic Server.

To deploy the reference implementation to the WebLogic Server:

Access OMW in JD Edwards EnterpriseOne.

1. In OMW, search for and select object JRH90I33, and add the object to your project.
2. Click Get (or Check-Out subsequently) to get the sources of the object to your local client.
3. From OMW, invoke JDeveloper with project JRH90I33 selected.
4. Open the Soft Coding Records application (P954000).
   If you are using a web service that is secured, you must pass values for all of the fields.
5. If you are using an anonymous login, delete the username and password values.
   For example:
   <username></username>
   <password></password>
6. Add the policy file, for example <Wssp1.2-2007-Https-UsernameToken-Plain.xml>, to the server domain folder location where the proxy project is deployed.
   This project RI uses the newly added API and fetches the softcoding record values as follows:

```
softCodingRecord =            SoftCodingRecordAccess.getSoftCodingRecord(context,
 softCodingKey);

use the API < getSoftcodingRecordFieldvalue(Context context,String fieldname,
 Element softCodingRecord )> to get the values for individual fields.

 // String url ="http://pc-asurendr-ptp:7001/DV812/RI_AddressBookManager";
    String url = SoftCodingRecordAccess.getSoftcoding RecordFieldvalue(context,
 "endpoint",softCodingRecord);


          //2.String username = "weblogic";
   String username = SoftCodingRecordAccess.getSoftcoding
 RecordFieldvalue(context,"username", softCodingRecord);


          //3.String password = "password";
   String password = SoftCodingRecordAccess.getSoftcoding
 RecordFieldvalue(context,"password", softCodingRecord);


         //4. Policy Name policyname = SoftCodingRecordAccess.getSoftcoding Record
 "Wssp1.2-2007-Https-UsernameToken-//Plain.xml";
```

```
String policyname = Fieldvalue(context, "policy", softCodingRecord);


        //5.Trustkey name
          // String trustKeyName = "DemoTrust.jks";
  String trustKeyName = SoftCodingRecordAccess.getSoftcoding
RecordFieldvalue(context,"trustkey",softCodingRecord);
```

7. Verify that the above code is added.

Note:

If you do not want to manually place the current security policy file (which is **<Wssp1.2-2007-Https-UsernameToken-Plain.xml>** to the WebLogic Server domain folder, where the business services server instance is installed and the proxy project is deployed, you must make the following code change in the JRH90I33 BSSV proxy project:

At line No 136 in the JRH90I33 code on Oracle Update Center, the replacement code is:

```
FileInputStream inbound = new FileInputsStream(policyname);
```

The policy name is taken from the Soft Coding Record. A sample value of policyname is:

```
Wssp1.2-2007-Https-UsernameToken-Plain.xml
```

With the above code, for the business service proxy object JRH90I33 to successfully consume a secure published business running on WebLogic Server, you must manually place the policy file in the WebLogic Server domain folder.

Note:

If you use the following code, you do not need to manually place the policy file in the WebLogic Server domain folder,

At line No 136 in the JRH90I33 code on Oracle Update Center, replace this code:

```
FileInputStreamInbound = new FileInputStream(policyname);
```

With this code:

```
Java.io.InputStream inbound = (new Object(().getClass().getResourceAsStream("/webLogic/
wsee/policy/runtime/" + policyname);
```

This replacement code ensures that the policy file specified in "policyname" will be dynamically loaded when the business services proxy object running on WebLogic Server is trying to consume a secure web service, and it will use the inbound object created by the above line of code while calling the getRI_Addressbook ManagerHttpPort() API.

After you make a change, check in the modified JRH90I33 object, and build and deploy the business services package.

In summary, whenever you create a business service proxy object for a secure web service running on WebLogic

Server, use this code to dynamically load the policy file during runtime:

```
getClass().getResosurceAsStream("/weblogic/wsee/policy/runtime/" + policyname);
```

Reference Implementation for Testing the Business Services Server as a JAX-WS Web Service Consumer

The JAX-WS consumer business service reference implementation JRH90I34 can be used to test the business service consumer scenario against the target JAX-WS web service RI_AddressBookManager of the JPR01000 published business service project. The JAX-WS consumer reference implementation was developed using the WSDL of the JAX-WS web service RI_AddressBookManager running on the JDeveloper 11g integrated WebLogic server, and it will also work against the JAX-WS RI_AddressBookManager running on a standalone WLS/WAS.

The reference implementation AddressBook test application (P954001) calls the getAddressBook operation of the JAX-WS web service RI_AddressBookManager using the JAX-WS consumer business service JRH90I34. The web service retrieves the associated address book data based on the address book number that you enter in P954001. The web service sends the information back to P954001 using JAX-WS consumer business service JRH90I34. If the information for the address book record appears in P95400I, then the Business Services server is set up correctly for consuming JAX-WS web services.

The methodology in this section, which uses JRH90I34 to test the consumer scenario using a JAX-WS based proxy, can be used as a reference to test the consumption of other third-party JAX-WS web services.

Prerequisites

Before you perform the tasks in this section, you must:

- Apply the ESU for business function B953002. B953002 has been changed so that it can call the JAX-WS Consumer RI JRH90I34.
- Download and install the ESU for the JAX-WS business service consumer RI, JRH90I34, to the Deployment server.
  The ESUs that correspond to the following bug numbers must be installed for testing the Web Services Consumer scenario with the JAX-WS Business Service Consumer reference implementation JRH90I34.
  - Business Function B953002
    - EnterpriseOne Applications Release 9.0: Bug 14156492
    - EnterpriseOne Applications Release 9.1: Bug 14156483
  - Business Service JRH90I34
    - EnterpriseOne Applications Release 9.0: Bug 14124057
    - EnterpriseOne Applications Release 9.1: Bug 14124040
- On the EnterpriseOne development client machine, using OMW, first Search for the JRH90I34 object and then do a Get operation in OMW to get the JRH90I34 object locally to <E1_Install_Path>\ DV900\java\source\oracle \e1\bssv\JRH90I34 location.
- Ensure the JAX-WS published business service RI_AddressBookManager (of JPR01000 business service project) is up and running with the web service returning proper values for the getAddressBook operation. RI_AddressBookManager can be running on the JDeveloper 11g Integrated WLS or on a standalone WLS/WAS.

Configuring Reference Implementation JRH90I34 for Testing the JAX-WS Web Services Consumer Scenario

After you set up the prerequisites, you must perform these tasks before you can use the P954001 test program to run the getAddressBook operation of the JAX-WS web service RI_AddressBookManager using the JAX-WS consumer business service JRH90I34.

1. Open the JRH90I34 business service project in JDeveloper 11g from OMW.
2. Create and save a deployment profile for the JRH90I34 project.
   See *"Creating a Deployment Profile for the Business Service Consumer Project" in the JD Edwards EnterpriseOne Tools Business Services Development Guide*
3. Start the integrated WebLogic server.
4. Deploy the updated JRH90I34 project to the integrated WebLogic server.
   See *"Deploying the Business Service to the Integrated WebLogic Server" in the JD Edwards EnterpriseOne Tools Business Services Development Guide*
5. Setup and activate an OCM record for the enterprise server to point to the business services server.
   The EnterpriseOne Web client user must have authority to work with the OCM application.

- On the EnterpriseOne Web client, type GH9011 in the Fast Path.
- Expand System Administration Tools, right click on Object Configuration Manager and click Versions.
- Select version ZJDE0003 for the Object Configuration Manager version.
- Select the appropriate data source.

  If you are using the local H4A, business functions will run locally in activeconsole, so use the local data source.

  If you are using a production environment, business functions will run on the enterprise server, so use the server data source.
- Click Add to add a new OCM record, and enter the following information:

  - Environment Name: Your Test environment name.
  - Service Name: BSSV
  - User/Role: EnterpriseOne user ID, Role ID, or *PUBLIC.
  - Server: BSSV server name.

    If the consumer business service is running locally on the JDeveloper 11g integrated WebLogic server, use the machine name where JDeveloper 11g is installed.
  - Port: JDENET listening port on the business service server.

    The port name should be the same as the serviceNameListen in the jdeinterop.ini file.
- Save the OCM record.
- Activate the OCM record.
- Test the connectivity with the Business Services server by selecting the OCM record, and then selecting Ping BSSV Server from the Row menu.

  Ensure the connection is successful.

If you are using a production environment, restart the Enterprise server to load the new OCM entry. If you are using a local EnterpriseOne development client, log out and then log in again.

6. Add a system-level service property record for the JRH90134 project.

   Open the Business Service Property Program (P951000).

   On Work with Service Property, click Add and enter the following information on the Add BSSV Property form:

   - Key: JRH90I30_APP_SERVER_OAS_OR_WAS_OR_WLS
   - Value: JAXWS
   - Level: System

   See *"Adding a Business Service Property Record" in the JD Edwards EnterpriseOne Tools Business Services Development Guide*

7. Add a softcoding record for the JRH90I34 project.

   Open the Web Service Soft Coding Records program (P954000) from the EnterpriseOne Web client.

   On Work with Soft Coding, click Add, and enter the following information on the Add Web Service Soft Coding Record form:

   - User / Role: Enter the name of the user ID for testing.
   - Environment Name: Enter the name of the test environment.
   - Soft Coding Key: JRH90I34.
   - Soft Coding Value: Enter the following xml.

```
<scwls>
<endpoint>http://dngorajesvm4:7101/context-root-JPR01000/
RI_AddressBookManagerPort?WSDL</endpoint>
</scwls>
```

   > **Note:** The endpoint points to the URL where the target JAX-WS web service is running. In the sample soft coding xml specified above, the endpoint is pointing to the target RI_AddressBookManager web service running locally on the JDeveloper 11g integrated WebLogic server. The web service running locally is not secured, so it is not required to specify user name and password fields in softcoding xml.

   If the target JAX-WS web service is secure and requires the WS-Security Header with user name and password credentials to be passed in the SOAP header, enter the masked user name and password values in the Enter Mask Fields section on the Update Web Service Soft Coding Record form and use the following xml in the Soft Coding Value field:

```
<scwls>
<endpoint>https://dnvmw8r201:9484/DV900/RI_AddressBookManager?WSDL</endpoint>
<username>JDE</username>
<password>_||_password_||_</password>
```

```
</scwls>
```

Note:

The endpoint points to the RI_AddressBookManager JAX-WS web service running on a standalone WebLogic server.

The password field in the softcoding XML is the Mask field in the Mask Fields grid and the actual value of the password is specified in the Mask For example the value for the user could be JDE, and the password could be JDE.

Note:

If the target web-service (JAX-WS consumer JRH90I34 is deployed to the integrated WLS of

JDeveloper 12c) is secured with SSL, then in the Integrated WLS start script (**startWebLogic.cmd**), add the following argument to the JAVA_OPTION jvm arguments to ignore host name verification.

```
-Dweblogic.security.SSL.ignoreHostnameVerification=true
```

The following is an example:

```
set JAVA_OPTIONS=%JAVA_OPTIONS% -Dweblogic.security.SSL.
 ignoreHostnameVerification=true
```

Typically the start script startWebLogic.cmd is at this location:

C:\Users\<user_name\AppData\Roaming\JDeveloper\system12.1.2.0.40.66.68\DefaultDomain\bin\startWebLogic.cmd

Running Reference Implementation JRH90I34

After you configure reference implementation JRH90I34, do the following to run the JAX-WS based business service consumer scenario.

- Restart the JDeveloper 11g integrated WebLogic server where the JAX-WS consumer reference implementation JRH90I34 is running.
- Run the Reference Implementation AddressBook application (P954001) from the EnterpriseOne Web client by entering a valid address book number (such as 4242 or 1001), and then click the getAddressBookInfo button.

The application calls business function B953002, which makes a call to the JAX-WS business service consumer reference implementation JRH90I34. JRH90I34 invokes the getAddressBook operation of the target JAX-WS provider web service, RI_AddressBookManager, to fetch the address book record for address book number that you entered.

Troubleshooting Tips

If you have any issues while testing the consumer business service scenario, configure the following logs:

- BSSV log.

  Use the jdelog.properties file in <E1_Install_Path>\ini\sbf folder if the JAX-WS consumer reference implementation is running locally on the JDeveloper 11g integrated WebLogic server.
- Jde and JdeDebug logs in local or for the COK on the Enterprise server, depending on where business function B953002 is running.
- IntegratedWeblogicServer log, which is located at:

  C:\Documents and Settings\USER_PROFILE\Application Data\JDeveloper \system11.1.1.3.37.56.60\DefaultDomain\servers\DefaultServer\logs

Java Client Reference Implementation for Media Object Operations

The MOBSSVJavaClient reference implementation is a standalone media object client for the JPR01MO1 published business service. This reference implementation is delivered for JD EDwards EnterpriseOne Tools Release 9.1 Update 2 and later releases. The reference implementation is located on the development client at System/classes/samples.

The MOBSSVJavaClient reference implementation is a standalone program that runs from the command prompt. The program takes the end point URL of the JPR01MO1 program that is deployed on the Business Services Server as input and performs media object operations on that Business Services Server.

Before you can run the MOBSSVJavaClient reference implementation, the published business services project JRP01MO1 must be deployed to the Business Services Server.

The steps for building and running the MOBSSVJavaClient reference implementation are provided in the MOBSSVJavaClient_Readme.html file packaged along with the reference implementation.

4 Event Notification Reference Implementation

RIForOutbound Reference Implementation Overview

The RIForOutbound reference implementation is an example of how you can use JD Edwards EnterpriseOne to send event notifications to third-party systems.

JD Edwards EnterpriseOne sends event notifications as JMS messages through JMS Queue and JMS Topic. The JD Edwards EnterpriseOne Transaction Server is the primary business event system for publishing guaranteed event notifications. When a transaction occurs in JD Edwards EnterpriseOne, the Transaction Server retrieves the data based on event configuration, converts the data to a properly formatted XML document, and routes the event to the JMS Queue or JMS Topic subscriber that are configured on a server accessible to orchestration systems. Any orchestration system that can read from JMS Queues and JMS Topics can consume these event notifications from JD Edwards EnterpriseOne.

To use this reference implementation, you must use the BPEL or ESB orchestration systems. JMS Queue and JMS Topic must be configured on an application server that is accessible to both the JD Edwards EnterpriseOne Transaction Server and the BPEL or ESB server. The application server can be on the same machine as the BPEL or ESB server.

Accessing the RIForOutbound Reference Implementation

The files for the RIForOutbound reference implementation are in a Zip file located in the JD Edwards EnterpriseOne installation directory:

```
\\EnterpriseOneInstallDirectory\System\Classes\samples\ESB
```

The Zip file includes a readme file that contains instructions on how to configure and test this reference implementation.

You must perform these tasks to use the RIForOutbound reference implementation to test your setup:

1. Create a JMS Queue on the BPEL/ESB application server.
2. Use the Interoperability Event Subscriber program (P90702A) in JD Edwards EnterpriseOne to create a JMS Queue subscriber.
3. Set up an administrator user (for example oc4jadmin) in the JDE.INI file on the Transaction Server. The username and password are used to make a connection to JMS queue. You must restart the Transaction Server after configuring this JDE.INI setting.
4. Follow the instructions in the readme, including the instructions on how to configure the ESB project to run the RIForOutbound reference implementation.

5 Reference Implementations for Auditing

Reference Implementations for Auditing

Note: Each section in this chapter contains the location of the reference implementations.

Reference Implementations for Testing a Business Services Auditing Configuration

Oracle provides two different reference implementations that you can use to test a business services auditing configuration:

- RI_AuditAddressBookManager
- RI_AuditInsertABStagingManager

RI_Audit AddressBookManager Reference Implementation

Oracle provides the JPR95001 - RI_AuditAddressBookManager reference implementation, which you can use to extend the RI_AddressBookManager published business service to pass audit information. This table describes the components of JPR95001 -RI_AuditAddressBookManager and the location of these components in the JD Edwards EnterpriseOne installation directory:

Reference Implementation Component	Description	Location
RI_AuditAddressBookManager	A published business service class that extends the AddressBookManager reference implementation so that it inserts an address book record with audit information.	B9\STAGINGA\java\source \oracle\e1\bssv\JPR95001
auditAddAddressBook	A method in the RI_ AuditAddressBookManager that is used to insert an address book record with audit information. This method uses the addAddressBook method of the RI_AddressBookManager class to insert the address book record.	Same as previous.
RI_AuditAddAddressBook	The value object class that contains getter and setter methods for the GUID, application ID, workstation name, and IP address.	B9\STAGINGA\java \source\oracle\e1\bssv \JPR95001\valueobject

See *"Configuring a Published Business Service to Pass Audit Information" in the JD Edwards EnterpriseOne Tools Auditing Administration Including 21 CFR Part 11 Administration Guide* for instructions on how to use this reference implementation to test this auditing configuration.

RI_AuditInsertABStagingManager Reference Implementation

The JPR95002 - RI_AddressBookStagingManager reference implementation is an example of a published business service that adds an address book record directly to the interoperability table. You can configure the RI_AuditInsertABStagingManager class to extend RI_AddressBookStagingManager, so that you can test the passing of audit information with an address book record when the address book record is added to the interoperability table.

This table describes the components of JPR95002 -RI_AuditInsertABStagingManager and the location of these components in the JD Edwards EnterpriseOne installation directory:

Reference Implementation Component	Description	Location
RI_ AuditInsertABStagingManager	A published business service class that extends the AddressBookStagingManager reference implementation so that it inserts a record with audit information into the interoperability table.	B9\STAGINGA\java\source \oracle\e1\bssv\JPR95002
auditInsertAddressBookStaging	A method in the RI_ AuditInsertABStagingManager that is used to insert an address book record with audit information into the interoperability table. This method uses the insertAddressBookStaging method of the RI_AddressBookStagingManager class to insert the address book record.	Same as previous.
RI_ AuditInsertAddressBookStaging	The value object class that contains getter and setter methods for the GUID, application ID, workstation name, and IP address.	B9\STAGINGA\java \source\oracle\e1\bssv \JPR95002\valueobject

See *"Configuring a Published Business Service to Pass Audit Information" in the JD Edwards EnterpriseOne Tools Auditing Administration Including 21 CFR Part 11 Administration Guide* for instructions on how to use this reference implementation to test this auditing configuration.

Reference Implementation for Testing a Java Connector Auditing Configuration

The files that you need to configure and test the Java Connector reference implementation for auditing are available in a zip file (AuditPurchaseOrderSample.zip) that you can download from the Update Center on My Oracle Support. The zip file also contains a ReadMe file that provides additional instructions on how to configure and run this reference implementation.

> **Note:** *"Configuring a Java Connector to Pass Audit Information" in the JD Edwards EnterpriseOne Tools Auditing Administration Including 21 CFR Part 11 Administration Guide .*

Reference Implementation for Testing a COM Connector Auditing Configuration

The files and resources for configuring and testing a COM connector reference implementation are located in the Update Center on My Oracle Support.

> **Note:** *"Configuring a COM Connector to Pass Audit Information" in the JD Edwards EnterpriseOne Tools Auditing Administration Including 21 CFR Part 11 Administration Guide .*

6 Testing Published Business Services from JDeveloper 11g

Editing the SBFProjects.library

The SBFProjects.library file contains the list of JD Edwards EnterpriseOne foundation JAR files that are added to the classpath of any business service project opened in JDeveloper. The SBFProjects.library file must be edited to add three additional JAR files.

The SBFProjects.libraryfolder is at this location:

\\EnterpriseOne InstallDirectory\system\classes

Open the SBFProjects.library file in notepad and add the ApplicationAPIs_JAR.jar, ApplicationLogic_JAR.jar and BizLogicContainerClient_JAR.jar files.

Add ApplicationAPIs_JAR.jar and ApplicationLogic_JAR.jar as the first entries under the <entries> element and the BizLogicContainerClient_JAR.jar after the BizLogicContainer_JAR.jar element, as illustrated in this example:

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<JLibraryNode nselem="JLibraryNode" class="oracle.jdeveloper.library.JLibraryNode"
 deployedByDefault="false" xmlns="http://xmlns.oracle.com/jdeveloper/101303/jlibrarynode">
   <classPath>
      <entries>
                <Item path="ApplicationAPIs_JAR.jar " jar-entry=""/>"
                <Item path="ApplicationLogic_JAR.jar " jar-entry=""/>"
                <Item path="Base_JAR.jar " jar-entry=""/>"
                <Item path="BizLogicContainer_JAR.jar " jar-entry=""/>"
                <Item path="BizLogicContainerClient_JAR.jar " jar-entry=""/>"
```

Save the changes.

Starting the Integrated WebLogic Server

Next you start the Integrated WebLogic Server.

To start the Integrated WebLogic Server:

1. In JDeveloper 11g, go to View, and then click Application Server Navigator:
2. Go to View, and then click Application Server Navigator.

   IntegratedWebLogicServer appears under Application Servers.
3. Right-click on IntegratedWebLogicServer, and then select Start Server Instance.

Creating Web Services

You create web services using JDeveloper 11g.

To create web services:

In JDeveloper 11g:

1. Right-click on your java code (for example, CustomerManager.java), and then select Create Web Service.
2. Accept the default value for Web Service Name (for example, CustomerManagerService.)
3. Accept the default value for Port Name (for example, CustomerManagerPort.)
4. Next, take the default values.
5. In Message Format, change the SOAP Message Format to Document/Literal.
6. In Available Methods, select the Available Method that you want to test.
7. Accept the defaults for the remaining values.
8. Click Finish.

   This message appears:

```
**
Generating WSDL and mapping file
WARNING: OWS-00077 The Value Type class: oracle.e1.bssvfoundation.base.ValueObject
 does not have a valid Java Bean pattern.
Adding service files to deployment profile
Service generation finished
Generation complete. **
```

Editing Project Properties

Edit the project properties to ensure that the SBFProjects library is added to the deployment profile of the business service project in JDeveloper 11g.

To edit project properties:

1. In JDeveloper 11g, right-click the business service project, and then select Project Properties.
2. Click Deployment and highlight your deployment profiles; for example, JP010020.
3. Click Edit on the right-hand side.
4. In Web-INF/lib, click Contributors, and then select SBFProjects library.
5. In the Filter section, select the Presence of the E1 Specific Jar files.

   This should already be selected, but make sure that BizLogicContainerClient_JAR.jar is selected.
6. In the Platform, click Target connection, and then select the Integrated WebLogic Server.

   The default platform should be WebLogic 10.3.
7. Click OK, and then click OK again to save your changes.

Deploying the Project

You deploy your business services project by right clicking on your project and selecting Deploy. Then select the Integrated WLS.

You should receive messages similar to the following during the deployment process:

```
**
[01:00:32 PM] ----  Deployment started.  ----
[01:00:32 PM] Target platform is  (Weblogic 10.3).
[01:00:38 PM] Retrieving existing application information
[01:00:39 PM] Running dependency analysis...
[01:00:39 PM] Building...
[01:00:48 PM] Deploying profile...
[01:01:02 PM] Wrote Web Application Module to C:\E900\PS900\Java\source\oracle\e1\bssv
\JP010020\deploy\JP010020.war
[01:01:04 PM] Redeploying Application...
[01:01:44 PM] [Deployer:149192]Operation 'deploy' on application 'JP010020' is in progress
 on 'DefaultServer'
[01:01:45 PM] [Deployer:149194]Operation 'deploy' on application 'JP010020' has succeeded
 on 'DefaultServer'
[01:01:45 PM] Application Redeployed Successfully.
[01:01:45 PM] The following URL context root(s) were defined and can be used as a starting
 point to test your application:
[01:01:45 PM] http://10.139.119.189:7101/context-root-JP010020
[01:01:45 PM] Elapsed time for deployment:  1 minute, 13 seconds
[01:01:45 PM] ----  Deployment finished.  ----
**
```

Testing the Business Service

After you deploy your business service project, you should test it. You test your business service on your Integrated WebLogic Server. Use one of these links to access your WebLogic Server:

- http://localhost:7101/console/login/LoginForm.jsp
- http://IPorMachineNameofJdev11g:7101/console/login/LoginForm.jsp

Enter the default user ID and password to log in to the Integrated WebLogic Server.

To test the business service:

1. Open your Integrated WebLogic Server.
2. In the Integrated WebLogic Server Admin Console, navigate to:

   Environment::Servers::Default Server::Deployments
3. Expand the BSSV for example, JP010020.
4. Click on your web service; for example, CustomerManagerService.
5. Select the Testing tab.
6. Expand the name and click on the test client.
7. Modify your request document and test your business service.

The following example shows the test request and results for getCustomerCreditInformation:

Request:

```
<getCustomerCreditInformation xmlns="http://oracle.e1.bssv.JP010020/">
<entity>
<entityId>1001</entityId>
</entity>
</getCustomerCreditInformation>
```

Result:

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
  <env:Header />
    <env:Body>
    <orac:getCustomerCreditInformationResponse xmlns:orac="http://
oracle.e1.bssv.JP010020/">
      <orac:e1MessageList />
      <orac:creditHoldExempt>false</orac:creditHoldExempt>
      <orac:amountTotalExposure>191976.8</orac:amountTotalExposure>
      <orac:entity>
        <orac:entityLongId />
        <orac:entityTaxId>66595263000170      </orac:entityTaxId>
        <orac:entityId>1001</orac:entityId>
      </orac:entity>
      <orac:amountCreditLimit>30000.00</orac:amountCreditLimit>
    </orac:getCustomerCreditInformationResponse>
  </env:Body>
  </env:Envelope>
```

Note: If you are using SQL Server 2008, add sqljdbc4.jar to the SBFProjects.library and 000\MISC folder.

Troubleshooting Tips

If you have any issues while testing the business service, configure the following logs for further information.

- BSSV log-configured using jdelog.properties.
- IntegratedWeblogicServer log-This log is located at:

  C:\Documents and Settings\USER_PROFILE\Application Data\JDeveloper\system11.1.1.3.37.56.60\DefaultDomain\servers\DefaultServer\logs