Oracle® Application Server

Adapter for SAP R/3 User's Guide

10*g* Release 2 (10.1.2)

B14061-03

April 2006

ORACLE®

Oracle Application Server Adapter for SAP R/3 User's Guide, 10*g* Release 2 (10.1.2)

B14061-03

Primary Author: Stefan Kostial

Contributing Authors: Vishal Saxena, Sunil Gopal, Sheela Vasudevan

Contents

A Configuring SAP for Inbound and Outbound Processing

Preface

This Preface contains these topics:

- Audience
- Documentation Accessibility
- Related Documents
- Conventions
- Help Us to Serve You Better

Audience

Oracle Application Server Adapter for SAP R/3 User's Guide is intended for those who perform the following tasks:

- Install applications
- Maintain applications

To use this document, you need to know how to install and configure Oracle BPEL Process Manager.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

http://www.oracle.com/accessibility/

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Related Documents

For more information, refer to these Oracle resources:

- *Oracle Application Server Adapter Concepts*
- *Oracle Application Server Adapters Installation Guide*

Printed documentation is available for sale in the Oracle Store at

`http://oraclestore.oracle.com/`

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

`http://www.oracle.com/technology/membership/`

If you already have a user name and password for OTN, then you can go directly to the documentation section of the OTN Web site at

`http://www.oracle.com/technology/documentation/`

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- Conventions in Text
- Conventions in Code Examples
- Conventions for Windows Operating Systems

Conventions in Text

We use the following conventions in text to help you more quickly identify special terms. The table also provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an **index-organized table**.
Italics	Italic typeface indicates book titles or emphasis.	*Oracle Database Concepts* Ensure that the recovery catalog and target database do *not* reside on the same disk.

Convention	Meaning	Example
UPPERCASE monospace (fixed-width) font	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, Recovery Manager keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, user names, and roles.	You can specify this clause only for a NUMBER column. You can back up the database by using the BACKUP command. Query the TABLE_NAME column in the USER_TABLES data dictionary view. Use the DBMS_STATS.GENERATE_STATS procedure.
lowercase monospace (fixed-width) font	Lowercase monospace typeface indicates executable programs, filenames, directory names, and sample user-supplied elements. *Note:* Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter sqlplus to start SQL*Plus. The password is specified in the orapwd file. Back up the datafiles and control files in the /disk1/oracle/dbs directory. The department_id, department_name, and location_id columns are in the hr.departments table. Connect as oe user. The JRepUtil class implements these methods.
lowercase italic monospace (fixed-width) font	Lowercase italic monospace font represents placeholders or variables.	You can specify the *parallel_clause*. Run *old_release*.SQL where *old_release* refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[ ]	Anything enclosed in brackets is optional.	DECIMAL (*digits* [, *precision*])
{ }	Braces are used for grouping items.	{ENABLE | DISABLE}
|	A vertical bar represents a choice of two options.	{ENABLE | DISABLE} [COMPRESS | NOCOMPRESS]
...	Ellipsis points mean repetition in syntax descriptions.	CREATE TABLE ... AS *subquery*;
	In addition, ellipsis points can mean an omission in code examples or text.	SELECT *col1*, *col2*, ... , *coln* FROM employees;
Other symbols	You must use symbols other than brackets ([ ]), braces ({ }), vertical bars (|), and ellipsis points (...) exactly as shown.	acctbal NUMBER(11,2); acct CONSTANT NUMBER(4) := 3;
Italics	Italicized text indicates placeholders or variables for which you must provide particular values.	CONNECT SYSTEM/*system_password* DB_NAME = *database_name*

Convention	Meaning	Example

`UPPERCASE`	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. Because these terms are not case sensitive, you can use them in either UPPERCASE or lowercase.	`SELECT last_name, employee_id FROM employees;`
		`SELECT * FROM USER_TABLES;`
		`DROP TABLE hr.employees;`

`lowercase`	Lowercase typeface indicates user-defined programmatic elements, such as names of tables, columns, or files.
	Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.
		`SELECT last_name, employee_id FROM employees;`
		`sqlplus hr/hr`
		`CREATE USER mjones IDENTIFIED BY ty3MU9;`

Conventions for Windows Operating Systems

The following table describes conventions for Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Click **Start**, and then choose the ***menu item***	How to start a program.	To start the Database Configuration Assistant, click **Start**, and choose **Programs**. In the Programs menu, choose **Oracle** - ***HOME_NAME*** and then click **Configuration and Migration Tools**. Choose **Database Configuration Assistant**.
File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (|), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the filename begins with \\, then Windows assumes it uses the Universal Naming Convention.	c:\winnt"\"system32 is the same as C:\WINNT\SYSTEM32
`C:\>`	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the *command prompt* in this manual.	`C:\oracle\oradata>`
Special characters	The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	`C:\>exp HR/HR TABLES=employees QUERY=\"WHERE job_id='SA_REP' and salary<8000\"`

Convention	Meaning	Example
HOME_NAME	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	`C:\> net start Oracle`*`HOME_NAME`*`TNSListener`
ORACLE_HOME and *ORACLE_BASE*	In releases prior to Oracle8i release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level *ORACLE_HOME* directory. This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level *ORACLE_HOME* directory. There is a top level directory called *ORACLE_BASE* that by default is `C:\oracle\product\10.1.0`. If you install the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is `C:\oracle\product\10.1.0\db_`*n*, where *n* is the latest Oracle home number. The Oracle home directory is located directly under *ORACLE_BASE*. All directory path examples in this guide follow OFA conventions. Refer to *Oracle Database Installation Guide for Windows* for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.	Change to the *ORACLE_BASE*`\`*ORACLE_HOME*`\rdbms\admin` directory.

Help Us to Serve You Better

To help our consultants answer your questions effectively, please be prepared to provide specifications and sample files and to answer questions about errors and problems.

The following list includes the specifications our consultants require.

- **Platform**:
- **Operating System**:
- **Operating System Version**:
- **Product List**:
- **Adapters**:
- **Adapter Deployment**:

  For example, *J2CA* or *Business Services Engine (BSE)*
- **Container Version**:

The following table lists components. Specify the version in the column provided.

Component	Version
Adapter	
EIS (DBMS/APP)	
HOTFIX/Service Pack	

In the following table, specify the JVM version and vendor.

JVM Version	Vendor

The following table lists additional questions to help us serve you better.

Request/Question	Error/Problem Details or Information
Provide usage scenarios or summarize the application that produces the problem.	
Has this happened previously?	
Can you reproduce this problem consistently?	
Any **change in the application environment**: software configuration, EIS/database configuration, application, and so on?	
Under what circumstance does the problem *not* occur?	
Describe the **steps** to reproduce the problem.	
Describe the **problem**.	
Specify the **error** message(s).	

The following is a list of error or problem files that might be applicable.

- XML schema
- XML instances
- Other input documents (transformation)
- Error screen shots
- Error output files
- Trace and log files
- Log transaction

1 Introduction

Oracle Application Server connects to an SAP system through Oracle Application Server Adapter for SAP R/3 (OracleAS Adapter for SAP). OracleAS Adapter for SAP provides connectivity and carries out interactions on an SAP system. This chapter discusses the following topics:

Adapter Features

OracleAS Adapter for SAP is a remote function call adapter that provides a means to exchange real-time business data between SAP R/3 systems and other application, database, or external business partner systems. The **adapter** enables external applications for inbound and outbound processing with SAP. OracleAS Adapter for SAP can be deployed as a J2EE Connector Architecture (J2CA) version 1.0 resource adapter. This deployment is referred to as OracleAS Adapter J2CA. It can also be deployed as a Web services servlet and is referred to as OracleAS Adapter Business Services Engine (BSE).

OracleAS Adapter for SAP uses XML messages to enable non-SAP applications to communicate and exchange transactions with SAP using services and events.The role of services and events is outlined. Services and events are described as follows:

- Services: Enable applications to call an SAP business object or business operation.
- Events: Enable applications to access SAP data only when an SAP business event occurs.

To support event functionality, the following two features are implemented:

- Port: A **port** associates a particular business object exposed by an adapter with a particular disposition. A disposition defines the protocol and location of the event data. The port defines the end point of the event consumption.

  The port is the Oracle adapter component that pushes the event received from the enterprise information system (EIS) to the adapter client. The port supported in this release is Remote Method Invocation (RMI).

> **Note:** You are not required to create or configure ports for use with BPEL Process Manager. However, in this release you can associate an event schema to a port under a J2CA configuration. See "J2CA Filtering Port Option for Integration with BPEL Process Manager" on page 2-24 for details on using ports under a J2CA configuration.

- Channel: A **channel** represents configured connections to particular instances of back-end or other types of systems. A channel binds one or more event ports to a particular **listener** managed by an adapter.

  The channel is the adapter component that receives events in real time from the 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. This enables the user to choose the optimal channel component based on deployment requirements. In the case of this adapter, the channel is an RFC server.

OracleAS Adapter for SAP provides:

- Support for bidirectional message interactions.
- OracleAS Adapter Application Explorer (Application Explorer), a GUI tool which uses SAP object repository metadata to build XML schemas and Web services to handle adapter requests or event data.
- Support for Remote Function Calls (RFC), Business Application Programming Interfaces (BAPI), and Intermediate Documents (IDoc) interfaces to SAP.
- XML schemas for the J2CA 1.0 resource adapter.
- Web services for BSE.

Data Type Limitation: Data types `h` and `g` are not supported. Type `h` represents a deep structure. Type `g` represents a variable length string. RFCTYPE_XSTRING and RFCTYPE_XMLDATA, as defined in SAPRFC.H, are not supported due to a limitation in the RFC Protocol.

> **See Also:** *Oracle Application Server Adapter Concepts*

SAP Certification

OracleAS Adapter for SAP provides state-of-the-art middleware solutions for SAP Basis and SAP Web application server-based systems. This adapter has achieved the following three interface certifications that promote cost-effective and low-risk solutions.

- **CA-ALE (Certified Adapter - Application Link Enabling) certification.** Enhances electronic data interchange (EDI) subsystem interface with SAP Basis and SAP Web Application Server. Using direct program-to-program remote communication and transformation from non-SAP systems to SAP solution-based systems, OracleAS Adapter for SAP expedites the conversion, import, and export of critical IDocs.
- **CA-AMS (Andrew Message System) certification.** Rapidly bridges SAP Basis and SAP Web Application Server data exchange with other applications through pure message delivery. As an Application Link Enabling (ALE) Message Handler, the

adapter sends IDoc messages without a requirement for conversion from one or more SAP solution-based systems.

- **CA-XML (Extensible Markup Language) certification.** Eases the communication between external middleware with SAP Basis and SAP Web Application Server over the Internet using XML, HTTP, or HTTPS. The adapter immediately transfers SAP solution specifications into XML for straight transfer into application subsystem repositories. The CA-XML-certified adapter directly receives and converts messages to be transformed from or into XML formats to or from SAP solution-based systems over the Internet.

Supported Versions and Platforms

The following SAP platforms are supported by OracleAS Adapter for SAP:

- SAP Web Application Server version 6.20 or 6.40
- SAP R/3 4.6C and 4.6D
- SAP R/3 Enterprise 47x100 and 47x200
- SAP Java Connector (SAP JCo) 2.16.

  For the current release status of the SAP Java Connector, refer to SAP Note #549268 in the SAP Service Marketplace.

> **Note:** Release versions may vary by product component. In addition, SAP functions may vary by SAP product version and support package.

OracleAS Adapter for SAP is supported on the following operating systems:

- Windows 2000/2003/XP
- Solaris version 2.8, 2.9, and later
- Linux x86 Red Hat Advanced Server version 2.1 and higher, SuSE version 8.1 and later, United Linux version 1.0 and later
- IBM AIX version 5.1 and later
- HP-UX version 11i and later
- HP Tru64 version 5.1a and later

SAP R/3 Versions and APIs

Adapter Platform	SAP R/3 Release	API
UNIX (HP-UX, Solaris)	SAP R/3 4.6C, 4.6D; SAP R/3 Enterprise 47x100 and 47x200*	SAP Java Connector (SAPJCo) 2.16**
Windows	SAP R/3 4.6C, 4.6D; SAP R/3 Enterprise 47x100 and 47x200*	SAP Java Connector (SAPJCo) 2.16**

*SAP R/3 Enterprise Version 47x100 and 47x200 is supported on the SAP Web Application Server Versions 6.20 and 6.40.

**For the current release status of the SAP Java Connector, refer to SAP Note #549268 in the SAP Service Marketplace.

Integration with SAP

You can use OracleAS Adapter for SAP to initiate an SAP business process, such as add/update account, or you can use the adapter as part of an integration effort to connect SAP and non-SAP systems.

BAPI and RFC are called synchronously by the adapter and always return data (either technical error information or a well-formed response document). IDocs are processed asynchronously.

The adapter is bidirectional and can process an event in SAP by receiving RFC and IDocs directly from SAP. The output sent by SAP can be in any of the following forms:

- An RFC request, for example, `RFC_CUSTOMER_GET`
- A BAPI request, for example, `BAPI_COMPANYCODE_GETLIST`
- An IDoc

For request processing, OracleAS Adapter for SAP can send requests to SAP using the BAPI, RFC, or IDoc interfaces. For details, see "SAP Business Components" on page 1-4.

OracleAS Adapter for SAP integrates SAP IDocs, RFC, and BAPI with mission-critical SAP system applications and other enterprise applications. The benefits of the adapter include:

- Elimination of the requirement for custom coding.
- Consistent data representation—a standard XML representation of event data and request/response documents for SAP. The developer is freed from the specific details of the SAP interface (BAPI, RFC, IDoc) and the specific configuration details of the target SAP system.
- Adherence to SAP ABAP serialization rules and SAP Interface Repository standards published by SAP AG.

SAP Business Components

OracleAS Adapter for SAP is designed to provide standard access to SAP business objects such as Remote Function Call (RFC) modules, Business Application Programming Interfaces (BAPI), and Intermediate Documents (IDocs) that are used to support existing business processes. The business objects and methods are available to the adapter as requests of SAP and to the event adapter when SAP initiates its remote requests. These objects work in the following ways:

- **BAPIs** are interfaces within the business framework that link SAP components to one another or to third-party components. BAPIs are called synchronously and return information.
- **RFC Modules** are SAP application interfaces that enable clients to utilize SAP technologies and receive responses.

Note: Depending on the release or service pack installed, certain RFCs, for example, `RFC_CUSTOMER_GET`, may not exist in your particular SAP system. Therefore, the examples included in this document may not be relevant to your system. If this is the case, then you should use the examples as a general reference for adapter functionality and choose an RFC that exists within your SAP application environment.

As described in SAP Release Note 109533, SAP Function Modules (RFCs) can be delivered with different release statuses. SAP supports only RFCs that are awarded with the Released for Customer status. There is no claim to the release independencies of the interfaces and the continued existence/functionality of the modules. For more information on the status of a specific function module, consult the SAP Service Marketplace.

- **IDocs** enable different application systems to be linked by a message-based interface. The IDoc type indicates the SAP format to use to transfer the data for a business transaction. An IDoc is a real business process in the form of an IDoc type that can transfer several message types. An IDoc type is described by the following components:
  - **Control records.** A control record contains data that identifies the sender, the receiver, and the IDoc structure. An IDoc contains one control record.
  - **Data records.** A data record consists of a fixed administration part and a data part (segment). The number and format of the segments can be different for each IDoc type.
  - **Status records.** A status record describes the processing stages through which an IDoc passes.

  The following scenario is an example of IDoc functionality and its components:

  Purchase order number 4711 was sent to a vendor as IDoc number 0815. IDoc number 0815 is formatted in IDoc type ORDERS01 and has the status records *created* and *sent*. The purchase order corresponds to the logical message ORDERS.

Adapter Architecture

OracleAS Adapter for SAP works with Application Explorer in conjunction with one of the following components:

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

Application Explorer (used to configure SAP connections and create Web services and events) can be configured to work in a Web services environment in conjunction with BSE. When working in a J2CA environment, the connector uses the Common Client Interface (CCI) to provide integration services using adapters instead of Web services.

Oracle Application Server Adapter Business Services Engine (BSE) Architecture

Figure 1–1 shows the generic architecture for BSE for packaged applications. Application Explorer works in conjunction with BSE, as deployed to the Oracle Containers for J2EE (OC4J) container of Oracle Application Server.

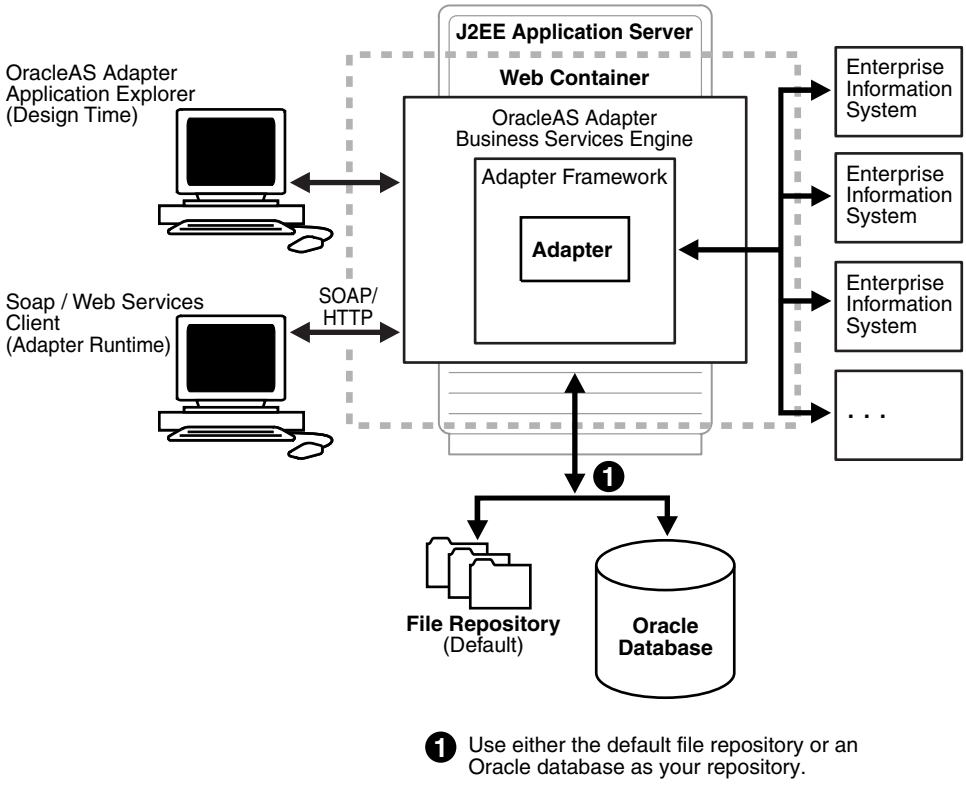
Application Explorer, a design-time tool deployed along with BSE, is used to configure adapter connections, browse EIS objects, configure services, and configure listeners to listen for EIS events. Metadata created while you perform these operations are stored in the repository by BSE.

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

BSE supports both a file-based and an Oracle database repository. The BSE repository stores the EIS connection information and the Web Service Definition Language (WSDL) for adapter services. A single instance of BSE can connect to multiple EIS applications.

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

Figure 1–1 Oracle Application Server Adapter Business Services (BSE) Architecture



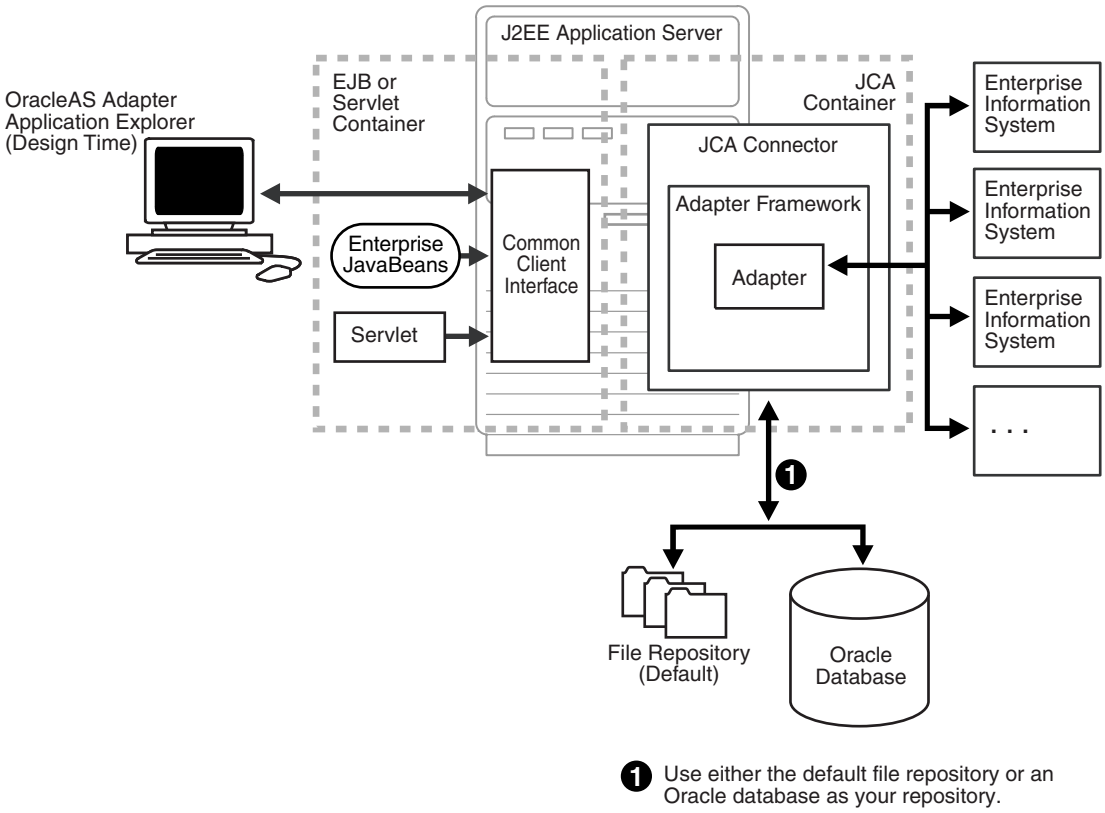
Oracle Application Server Adapter Generic J2CA Architecture

Figure 1–2 shows the generic architecture for OracleAS Adapter J2CA for packaged applications. This is a pure J2CA 1.0 resource adapter deployed in managed mode in the OC4J container of the Oracle Application Server. It is a universal adapter. One adapter can connect to many EIS applications.

The OracleAS Adapter J2CA repository contains the list of EIS connection names and the associated connection parameters. 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 OC4J

deployment descriptor `oc4j-ra.xml`. See Chapter 3, "OC4J Deployment and Integration" for more information on OC4J deployment.

Figure 1–2 Oracle Application Server Adapter Generic J2CA Architecture



See Also:

- *Oracle Application Server Adapter Concepts*
- *Oracle Application Server Adapters Installation Guide*

BSE Versus OracleAS Adapter J2CA Deployment

If you are using OracleAS Adapter for SAP with BPEL Process Manager, please note that:

- Only OracleAS Adapter J2CA deployment supports inbound integration (event notification) with BPEL Process Manager.
- Both OracleAS Adapter J2CA and BSE deployments support outbound integration (request-response service) with BPEL Process Manager.

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

1. BSE is the preferred deployment option because it:
   - Can be deployed in a separate instance of Oracle Application Server.
   - Provides better distribution of load.

- Provides better isolation from any errors from third party libraries.
- Provides better capability to isolate issues for debugging purposes.
- Conforms more closely to the Service Oriented Architecture (SOA) model for building applications.

2. OracleAS Adapter J2CA provides slightly better performance.

   OracleAS Adapter J2CA does provide slightly better performance than BSE. However, the difference decreases as the transaction rate increases.

3. OracleAS Adapter J2CA and the BSE option both provide identity propagation at runtime.

   The BSE option provides the capability to pass identity using the SOAP header. For OracleAS Adapter J2CA, user name and password can be passed using the connection specification of the CCI.

2

Configuring OracleAS Adapter for SAP

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

This chapter discusses the following topics:

Starting Application Explorer

To start Application Explorer:

1. Start the server where Application Explorer is deployed.
2. From the Windows **Start** menu, select **Programs**, ***OracleAS_home*** **Adapters**, and then **Application Explorer**.

   On Windows, `iaexplorer.bat` is located under *`OracleAS_home`*`\adapters\application\tools`, where `OracleAS_home` is the directory where Oracle Application Server is installed.

   On UNIX, load the script `iwae.sh`, located under *`OracleAS_home`*`/adapters/application/tools`, where `OracleAS_home` is the directory where Oracle Application Server is installed.

   Application Explorer starts. You can now define new targets to your SAP system.

Configuring Settings for BSE or J2CA

You need not configure BSE for a file-based repository because it is configured during the Oracle installation. You also need not configure the OracleAS Adapter J2CA because the `ra.xml` file is configured automatically during installation.

Configuring BSE

After BSE is deployed to Oracle Application Server, you can configure it through the BSE configuration page. This configuration is required only when using a database repository with BSE.

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

To configure BSE:

1. Display the following page in your browser:

   `http://hostname:port/ibse`

   Where `hostname` is the machine where BSE is installed and `port` is the HTTP port for Oracle Application Server.

   For example,

   `http://localhost:7777/ibse`

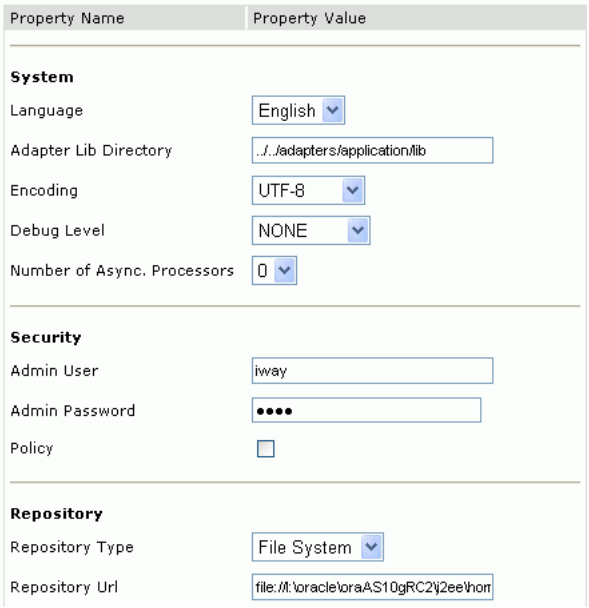
   > **Note:** If you are accessing this page for the first time, it may take longer to load.

2. Log on when prompted.

   When first installed, the user ID and password are:

   - User name: iway
   - Password: iway

The BSE configuration page is displayed.



3. Ensure that the Adapter Lib Directory parameter specifies the path to the lib directory, for example:

   OracleAS_home\adapters\application\lib

   Where OracleAS_home is the directory where Oracle Application Server is installed.

   After you specify the path, adapters in the lib directory are available to BSE.

4. For security purposes, enter a new password in the **Admin Password** field.

   Note: The Repository URL field specifies where the file system repository is located. To use a database repository, you must enter the repository connection information. For the initial verification, use a file system repository. See "Configuring an Oracle Repository" on page 2-6 for information on switching to a database repository.

5. Click **Save**.

Configuring BSE System Settings

To configure BSE system settings:

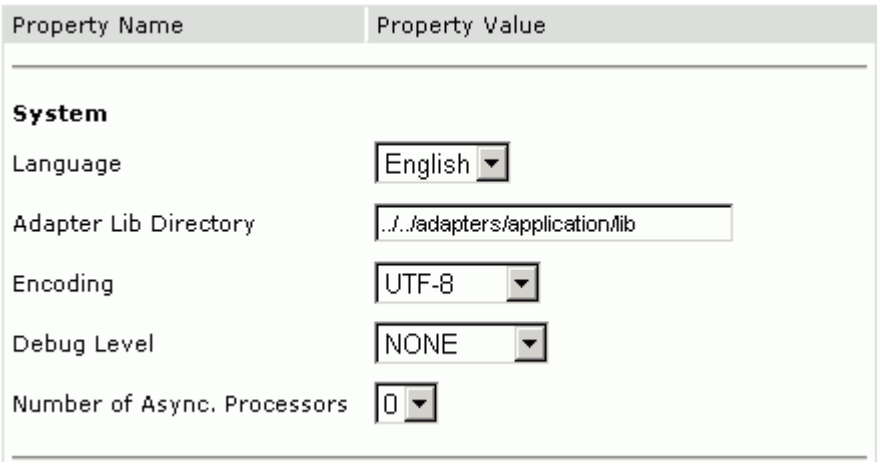
1. Display the **BSE configuration** page in a browser:

   http://*hostname:port*/ibse/IBSEConfig

   Where hostname is the machine where BSE is installed and port is the port number on which BSE is listening.

   Note: The server to which BSE is deployed must be running.

The BSE settings pane is displayed, as shown in the following figure.

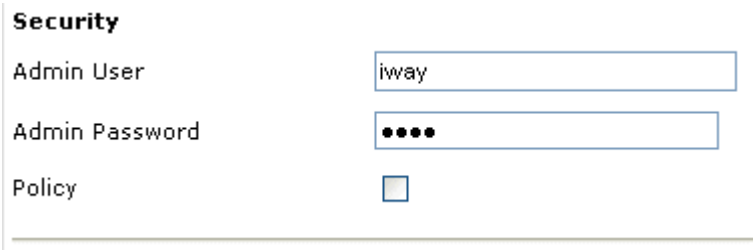


2. Configure the system settings.

   The following table lists the parameters with descriptions of the information to provide.

Parameter	Description
Language	Specify the required language.
Adapter Lib Directory	Enter the full path to the directory where the adapter jar files reside.
Encoding	Specify the default encoding from one of the following options: ■ UTF-8 ■ EBCDIC-CP-US ■ ISO-88859-1 ■ Shift JIS ■ UNICODE
Debug Level	Specify the debug level from one of the following options: ■ None ■ Fatal ■ Error ■ Warning ■ Info ■ Debug
Number of Async. Processors	Select the number of asynchronous processors.

The following image shows the Security pane.

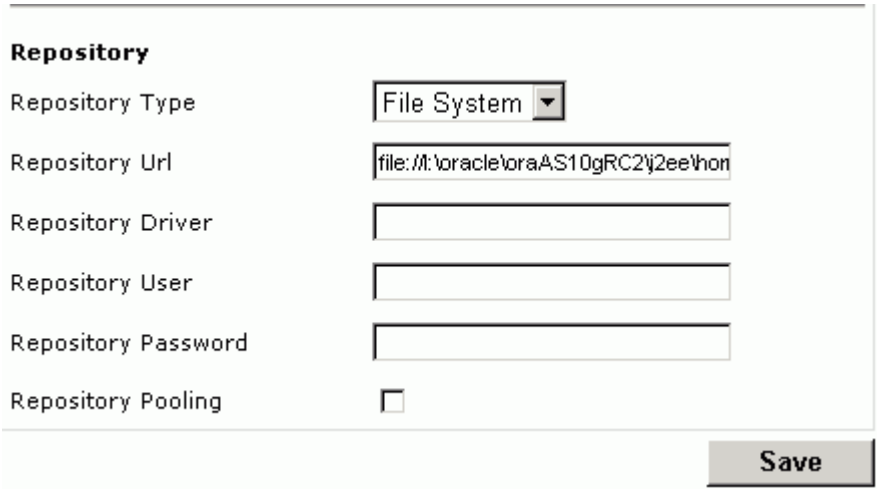


3. Configure the security settings.

   The following table lists the parameters with descriptions of the information to provide.

Parameter	Description
Admin User	Provide a BSE administrator ID.
Admin Password	Enter the password associated with the BSE administrator ID.
Policy	Select the check box to enable policy security.

The following image shows all fields and check boxes for the Repository pane.



4. Configure the repository settings.

   BSE requires a repository to store transactions and metadata required for the delivery of Web services.

   See "Configuring a File System Repository" on page 2-6 and "Configuring an Oracle Repository" on page 2-6 for more information.

   The following table lists the parameters with descriptions of the information to provide.

Parameter	Description
Repository Type	Select one of the following repositories from the list: ■ Oracle ■ File (Do not use for BSE in production environments.)

Parameter	Description
Repository URL	Enter the URL to use when opening a connection to the database.
Repository Driver	Provide the driver class to use when opening a connection to the database (optional).
Repository User	Enter the user ID to use when opening a connection to the database.
Repository Password	Enter the password associated with the user ID.
Repository Pooling	Select the check box to enable pooling.

5. Click **Save**.

Configuring a File System Repository

If you do not have access to a database for the repository, you can store repository information in an XML file on your local machine. However, a file system repository is less secure and efficient than a database repository. When BSE is first installed, it is automatically configured to use a file system repository.

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

The default location for the repository on Windows is:

```
OracleAS_home\j2ee\OC4J_CONTAINER\applications\ws-app-adapter
\ibse\ibserepo.xml
```

On other platforms, use the corresponding location.

If you are using a file system repository, you are not required to configure any additional BSE components.

Configuring an Oracle Repository

To configure an Oracle repository:

1. Contact your database administrator to obtain an Oracle user ID and password to create the BSE repository.

   This user ID should have rights to create and modify tables as well as the ability to create and run stored procedures.

2. Open a command prompt and navigate to the setup directory. The default directory location on Windows is:

   ```
   OracleAS_home\adapters\application\etc
   ```

   For other platforms, use the corresponding location.

   This directory contains SQL to create the repository tables in the following file:

   ```
   iwse.ora
   ```

   > **Note:** If Oracle is not on the same machine as the Oracle Application Server, copy the iwse.ora file to the Oracle machine. Then, from a command prompt on the Oracle machine, navigate to the directory containing the iwse.ora file.

3. Enter the following command:

```
sqlplus userid/password @database @ iwse.ora
```

Configuring J2CA

During the J2CA deployment of OracleAS Adapter for SAP, OC4J generates a deployment descriptor called `oc4j-ra.xml`. This descriptor provides OC4J-specific deployment information for resource adapters. See Chapter 3, "OC4J Deployment and Integration" for more information on J2CA deployment and configuration.

No configuration changes are necessary if you are using the default file based repository with J2CA deployment.

Configuring a Database Repository for J2CA

To configure a database repository for J2CA:

1. Execute the `iwse.ora` SQL statement on the machine where the database is installed.
2. Copy the `jcatransport.properties` file to the following directory:

   OracleAS_home\adapters\application\config\jca_sample

3. Uncomment the following fields and enter details for them in the `jcatransport.properties` file. For example:

```
iwafjca.repo.url=jdbc:oracle:thin:@90.0.0.51:1521:orcl
iwafjca.repo.user=scott
iwafjca.repo.password=scott1
```

4. Alter the JDBC driver path in Application Explorer's lcp. For example:

```
lcp=..\lib\orabpel-adapters.jar;C:\jdev\jdbc\lib\classes12.jar;C:\jdev\jdbc\lib
\nls_charset12.jar;%lcp%
to
lcp=..\lib\orabpel-adapters.jar;..\..\..\jdbc\lib\classes12.jar;..\..\..\jdbc\l
ib\nls_charset12.jar;%lcp%
```

Creating a Repository Configuration

Before you use Application Explorer with OracleAS Adapter for SAP, 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 runtime.

A default J2CA repository is created for the default ManagedConnectionFactory. The name of this configuration is `jca_sample`.

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

Creating a Configuration for BSE

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

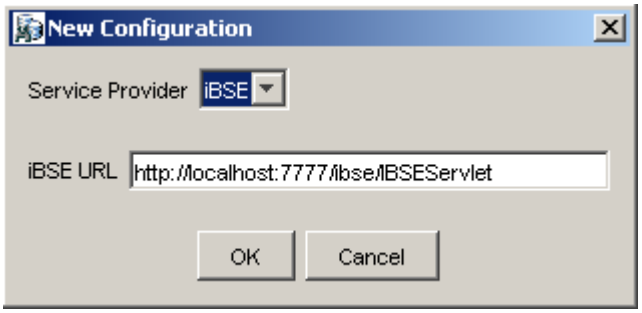
Defining a New Configuration for BSE

To define a new configuration for BSE:

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

   The New Configuration dialog box is displayed.

2. Enter a name for the new configuration, for example, SampleConfig, and click **OK**.



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 with the following format:

   `http://`*`hostname:port`*`/ibse/IBSEServlet`

   Where `hostname` is the machine on which your application server resides and `port` is the port number where the application server is listening.

5. Click **OK**.

   A node representing the new configuration appears beneath the root Configurations node.

   The Web service repository configuration file is stored in *`OracleAS_home`*`\j2ee\home\applications\ws-app-adapter\ibse`.

Creating a Configuration for J2CA

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

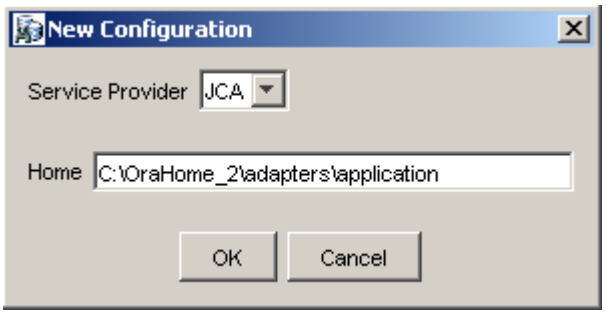
Defining a New Configuration for J2CA

To define a new configuration for J2CA:

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

   The New Configuration dialog box is displayed.

2. Enter a name for the new configuration, for example, SampleConfig, and click **OK**.



3. From the **Service Provider** list, select **JCA**.
4. In the **Home** field, enter a path to your J2CA configuration directory where the repository, schemas, and other information is stored, for example:

   OracleAS_home\adapters\application

5. Click **OK**.

   A node representing the new configuration appears beneath the root Configurations node.

   The OracleAS Adapter J2CA configuration file is stored in *OracleAS_home*\adapters\application\config*configuration_name*

   Where OracleAS_home is the directory where Oracle Application Server is installed and configuration_name is the name of the configuration you created; for example, SampleConfig.

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, you will not see the Business Services node. The following is an example of a BSE configuration named SampleConfig:

- Use the **Adapters** folder to create inbound interaction with SAP. For example, you use the SAP node in the Adapters folder to configure a service that updates SAP.
- Use the **Events** folder to configure listeners that listen for events in SAP.

- Use the **Business Services** folder (available for BSE configurations only) to test Web services created in the Adapters folder. You can also control security settings for the Web services by using the security features of the Business Services folder.

You can now define new targets to SAP.

Establishing a Connection (Target) for SAP

Defining the application includes adding a target for OracleAS Adapter for SAP. Setting up the target in Application Explorer requires information that is specific to the target.

To browse the available business functions, you must first define a target to SAP. After you define the target, it is automatically saved. You must connect to the SAP 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.

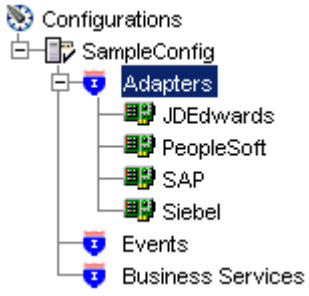
Defining a Target to SAP

To connect to SAP for the first time, you must define a new target. OracleAS Adapter for SAP supports SAP standard security and the additional protocol of SNC. Once connected to the SAP application server, application security is managed by user ID, roles and profiles. For more information on SAP application security, see the appropriate SAP documentation.

To define a target:

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

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



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

   The Add Target dialog box is displayed. Provide the following information:

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

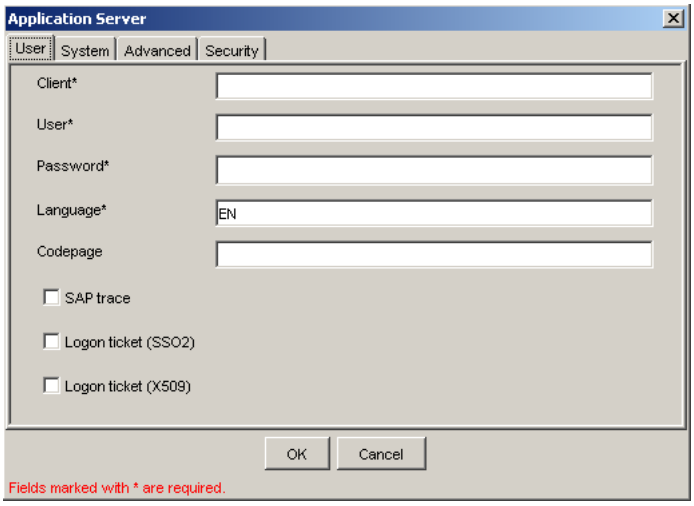
   b. In the **Description** field, enter a description for the target (optional).

   c. From the **Target Type** list, select the type of target you are connecting to. The supported target types include **Message Server** or **Application Server** (default).

Note: For load balancing purposes, application servers from one SAP system are usually configured in logon groups, where each group serves a particular kind of user. The application servers in each group are assigned to users by a least-heavily-loaded strategy. This load balancing is done by message servers. Each SAP system has exactly one message server, which can be reached through TCP on a specific message server port.

3. Click **OK**.

   The Application Server dialog box is displayed.



   The following tabs are available:

   - User (Required)
   - System (Required)
   - Advanced
   - Security

4. For the **User** tab (required), enter the appropriate information for your SAP target based on the information in the following table.

Table 2–1 User Tab Parameters

Target Parameter	Description
Client	The client number defined for the SAP application for client communications.
User	A valid user ID for the SAP application.
Password	A valid password for the SAP application.
Language	A language key. EN (English) is the default.
Code page	A character code page value.
SAP Trace	Select this option to enable traces.
Logon ticket (SSO2)	If you are using a Secure Network Communications (SNC) adapter with SAP, select the Logon ticket (SSO2) check box.

Table 2–1 (Cont.) User Tab Parameters

Target Parameter	Description
Logon ticket (X509)	If you are using an SNC adapter with SAP, select the Logon ticket (X509) check box.

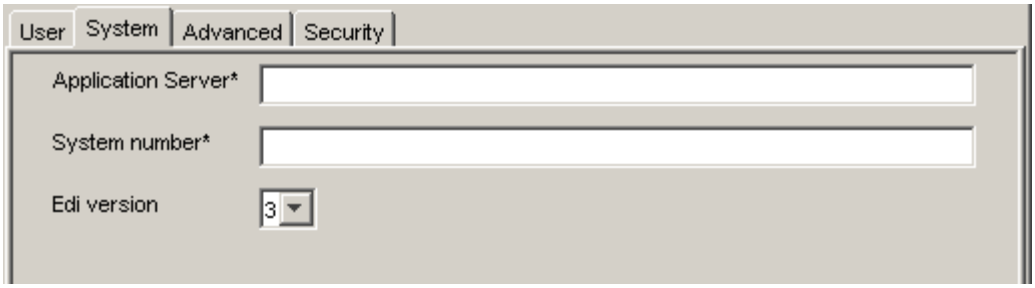
Secure Network Communications (SNC) provides protection for the communication links between the distributed components of an R/3 System. Using SNC, SAP R/3 can support products that adhere to the GSS-API Version 2 standard. SNC supports application level (end-to-end security), Smartcard authentication, and Single Sign-On (SSO).

Depending on the SAP system release, logging on using SSO or X.509 certificates is supported.

- For **SSO**, specify the user to be $MYSAPSSO2$ and pass the base64 encoded ticket as the passwd parameter.
- For **X509**, specify the user to be $X509CERT$ and pass the base64 encoded certificate as the passwd parameter.

For more information, see your SAP system documentation.

5. For the **System** tab (required), enter the appropriate information for your SAP target based on the information in this section.



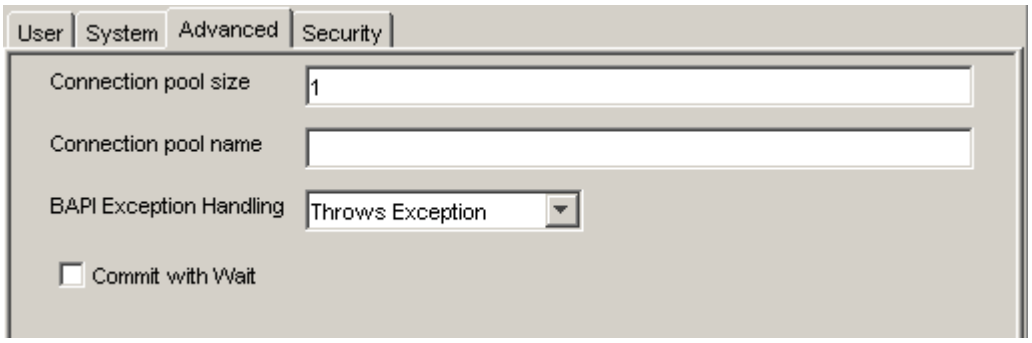
The System tab enables you to provide the application server name, system number, and EDI version for the SAP system to which you are connecting.

Table 2–2 System Tab Parameters

Target Parameter	Description
Application Server	The host name or IP address for the computer that is hosting the SAP application.
System Number	The system number defined to SAP for client communications.
EDI Version	The Electronic Data Interchange (EDI) document version that you are using with the adapter. Version 3 is the default value.

6. For the **Advanced** tab (optional), enter the appropriate information for your SAP target based on the information in this section.

The Advanced tab enables you to specify SAP connection pooling information for the SAP system to which you are connecting.



Connections to an R/3 server take up valuable resources on both the client and the remote server. You can create a pool of connections to minimize the resource and time constraints. In estimating the size of the pool, you may calculate pool size by the amount of server resources to be consumed, the number and size of the documents to be received, and the size of your Java Virtual Machine. The section of SAP documentation "Memory Management (BC-CST-MM)" explains in detail the resources required on the SAP system.

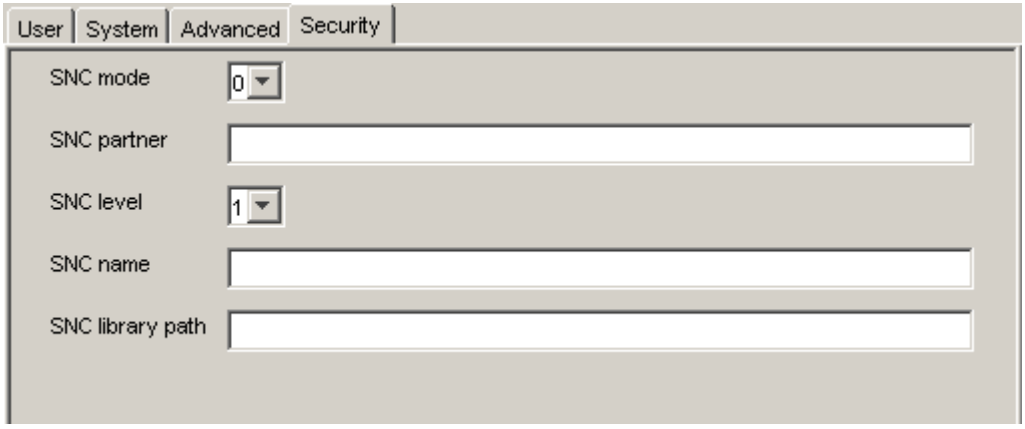
Table 2–3 Advanced Tab Parameters

Target Parameter	Description
Connection pool size	Connection pool size is used to specify the number of client connections in a pool you want to make available to SAP for Web service calls. Enter the number of connections you want to make available to SAP. To use a connection pool, enter a value that is greater than 1. **Important:** The default value of 1 does not create a connection pool. Instead, a single SAP connection with sequential processing is shared. A pooled connection invokes multiple connections to SAP with parallel processing. If you are using Application Explorer to create Web services, the connection pool size value is used by your Web service during runtime. As a result, ensure that the connection pool size is sufficient for your purposes.
Connection pool name	Enter the name of your SAP connection pool only if you specified a connection pool size greater than 1.
BAPI Exception Handling	From the list in the event of a BAPI exception, you can select **Creates Error Document** or **Throws Exception**. To receive more detailed error messages, select **Creates Error Document**. As a rule: ■ If your application is **Java centric**, select **Throws Exception** so that code components can catch the exception and react accordingly. ■ If your application is **document based**, select **Creates Document** to create an XML document that contains the Java exception. It is up to your application to read the XML document and obtain the error.

Table 2–3 (Cont.) Advanced Tab Parameters

Target Parameter	Description
Commit with wait	If a high degree of accuracy is required in your application, select the **Commit with Wait** check box. The adapter waits until all records are physically written to the database before returning from the function call. The "Commit With Wait" has a performance impact on adapter performance, so consider carefully before selecting it. The commit behavior of BAPIs is described in the SAP documentation under "BAPI Programming Guide and Reference (CA-BFA)." All SAP Business Objects that change data must commit work to the database. Some BAPIs developed in version 3.1 of the R/3 system use an internal commit behavior, and their commit behavior cannot be changed by the adapter. As soon as they are called, they commit the work they did. BAPIs developed since release 3.1 use the external commit method. The adapter issues a commit command, and the commit is put in the database queue. If there is an application error in the first part of the commit, the error message "Posting could not be carried out" is returned, and the adapter rolls back the transaction. If in writing to the database, a database error occurs, a short dump is issued in the database records of SAP, but no message is returned to the adapter about the failure. This option is disabled by default.

7. For the **Security** tab (optional), enter the appropriate information for your SAP target based on the information in this section.



The Security tab enables you to specify Secure Network Communication (SNC) information for the SAP system to which you are connecting.

Table 2–4 Security Tab Parameters

Target Parameter	Description
SNC mode	By default, SNC is disabled. To enable SNC, select 1 from the list.
SNC partner	Enter the name of the RFC server or message server (load balancing) that provides the SNC services.
SNC level	From the list select the version of the SNC library.
SNC name	Enter the name of the SNC library you are using.

Table 2–4 (Cont.) Security Tab Parameters

Target Parameter	Description
SNC library path	Enter the path to the SNC library.

SNC provides protection for the communication links between the distributed components of an R/3 System. Using SNC, SAP R/3 can support products that adhere to the GSS-API Version 2 standard. SNC supports application level (end-to-end security), Smartcard authentication, and single sign-on (SSO).

If you are using SAP Enterprise Portal, the J2EE engine generates the SAP logon ticket automatically. A possible SNC scenario would be from SAP Enterprise Portal to OracleAS Adapter for SAP.

If you want to use SAP logon tickets to enable SSO to non-SAP components, consult the SAP documentation regarding Pluggable Authentication Services. A possible SNC scenario in this case would be from a non-SAP Enterprise Portal to OracleAS Adapter for SAP.

8. When you have provided all the required information for your target, click **OK**.

   After the extraction finishes, the new target, SAPTarget, appears under the SAP adapter node.

You can now connect to your SAP target.

See "Creating XML Schemas" on page 2-17 for information on how to create schemas for the adapter.

Connecting to a Defined SAP Target

To connect to an existing target:

1. In the left pane, expand the **Adapters** node.
2. Expand the **SAP** node.
3. Click the target name under the SAP node (for example, SAPTarget).

   The Connection dialog box displays the values you entered for connection parameters.
4. Verify your connection parameters.
5. Provide the correct password.
6. Right-click the target name and select **Connect**.

   The x icon disappears, indicating that the node is connected.

Managing a Connection to SAP

To manage SAP connections, you can:

- Disconnect from a connection that is not currently in use.

  Although you can maintain multiple open connections to different transaction processing systems, it is recommended to disconnect from connections not in use.

- Edit a target.

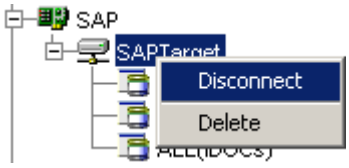
  You can modify the connection parameters when your system properties change. After you disconnect, you can modify an existing target.

- Delete a connection that is no longer needed.

Disconnecting from a Connection to SAP

To disconnect a target:

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



Disconnecting from the SAP target drops the connection with SAP, but the node remains.

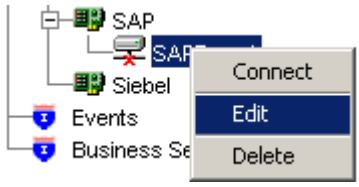
The x icon appears, indicating that the node is disconnected.

Modifying Connection Parameters

After you create a target for SAP using Application Explorer, you can edit any of the information that you provided previously.

To edit a target:

1. Verify that the target you want to edit is disconnected.
2. Right-click the target and select **Edit**.



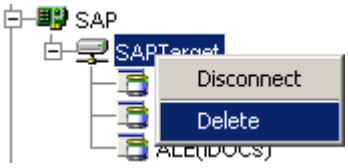
The Application Server dialog box displays the target connection information.

3. Change the properties in the dialog box as required and click **OK**.

Deleting a Connection to SAP

To delete a target:

1. Locate the target you want to delete.
2. Right-click the target (for example, SAPTarget), and select **Delete**.



The node disappears from the list of available connections.

Viewing Application System Objects

As you connect to SAP, Application Explorer enables you to explore and browse SAP business objects that are used to support existing business processes.

Note: Depending on the release or service pack installed, certain RFCs, for example, RFC_CUSTOMER_GET, may not exist in your particular SAP system. Therefore, the examples included in this documentation may not be relevant to your system. If this is the case, you should use the examples as a general reference for adapter functionality and choose an RFC that exists within your SAP application environment.

As described in SAP Release Note 109533, SAP Function Modules (RFCs) can be delivered with different release statuses. SAP supports only RFCs that are awarded the Released for Customer status. There is no claim to the release independencies of the interfaces and the continued existence/functionality of the modules. For more information on the status of a specific function module, consult your SAP Service Marketplace.

See the *SAP User's Guide* for more information.

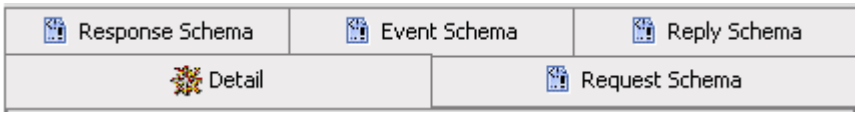
Creating XML Schemas

After you explore the SAP business function library and select an object, you can use Application Explorer to create the XML request schema and the XML response schema for that function.

To create request and response schemas for an SAP business function.

1. Connect to an SAP target as described in "Connecting to a Defined SAP Target" on page 2-15.
2. Expand the **Business Object Repository** node.
3. Click the icon to the left of the **Financial Accounting** node.
4. Scroll down and click the icon to the left of the **Company** business object.
5. Scroll down and select the BAPI named `BAPI_COMPANY_GETLIST`.

The following screen appears on the right.



6. To view the XML for each schema type, click the appropriate tab.

Generating WSDL (J2CA Configurations Only)

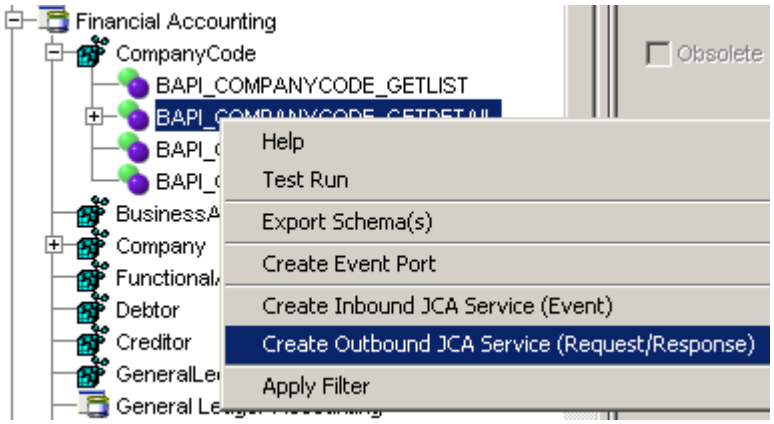
The Web Service Definition Language (WSDL) description of a service enables you to make the service available to other services within a host server. You use Application Explorer to create both request-response (outbound) and event notification (inbound) J2CA services of the adapter.

> **Note:** The **Create Inbound JCA Service (Event)** option is only available when the selected node supports events.

To generate a WSDL file for request-response service:

1. After you create a schema, right-click the respective object.

   The following menu is displayed:



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

   The Export WSDL dialog box is displayed.

3. Accept the default name and location 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`.

> **Note:** You can organize your WSDL files in subfolders, creating your own WSDL hierarchy structure. Create the folders under *OracleAS_home*\adapters\application\wsdls\. The WSIL browser in JDeveloper will display the full tree structure of your WSDL hierarchy.

4. Click **OK**.

   The WSDL file is saved in the specified location.

The procedure for generating WSDL for event notification is similar to request-response. To generate WSDL for event notification, you must first create a channel for every event. See "Generating WSDL for Event Notification" on page 5-14 for a detailed example.

Creating and Testing a Web Service (BSE Configurations Only)

Using Application Explorer, you can explore the business function repository and generate Web services (also known as a **business service**) for the SAP functions you want to use with the adapter. The following procedure uses the SAP BAPI method called BAPI_MATERIAL_GETLIST as an example and returns a list of materials from SAP.

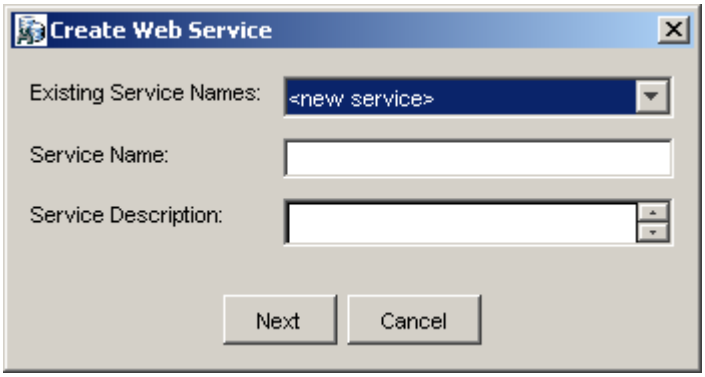
> **Note:** In a J2EE Connector Architecture (J2CA) implementation of the adapter, Web services are not available. When the adapter is deployed to use the OracleAS Adapter J2CA, the Common Client Interface provides integration services using the adapter.

Creating a Web Service

To create a Web service for an SAP business function:

1. Connect to your **SAP** target and expand the **Business Object Repository** node.
2. Select the BAPI_MATERIAL_GETLIST method from the **Business Object Repository**.
3. Right-click the node from which you want to create a business service and select **Create Web Service**.

   The Create Web Service dialog box is displayed. You can add the business function as a method for a new Web service or as a method for an existing one.



Perform the following steps:

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

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

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

4. Click **Next**.

   The License and Method dialog box is displayed.

   Provide the following information:

   a. In the **License Name** field, select one or more license codes to assign to the Web service. To select more than one, hold down the **Ctrl** key and click the licenses.

   b. In the **Method Name** field, enter a descriptive name for the method.

   c. In the **Method Description** field, enter a brief description of the method.

5. Click **OK**.

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

Testing a Web Service

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, an input field appears.

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

   Application Explorer displays the results.

Identity Propagation

If you test or run a Web service using a third party XML editor, the Username and Password values that you specify in the SOAP header must be valid and are used to connect to SAP. The user name and password values that you provided for SAP 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.

Configuring an Event Adapter

Events are generated as a result of 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, your application is a consumer of this event.

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

> **Note:** If you are using a J2CA configuration, you must create a new channel for every event and select this channel when you generate WSDL. Additionally, you are not required to create or configure ports for use with BPEL Process Manager. See "J2CA Filtering Port Option for Integration with BPEL Process Manager" on page 2-24 for details on using ports under a J2CA configuration.

- A **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. See "Creating and Editing an Event Port" on page 2-21 for more information.
- A **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 the adapter. See "Creating a Channel" on page 2-24 for more information.

> **Note:** OC4J currently conforms to J2CA 1.0, which does not call for event capabilities. When conforming to J2CA 1.0, only service interactions are supported.

Creating and Editing an Event Port

Application Explorer enables you to create event ports from the Adapters node or from the Events node.

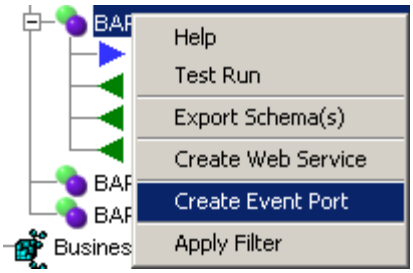
> **Note:** You are not required to create event ports for J2CA configurations. You must create event ports for BSE configurations only.

Creating an Event Port from the Adapters Node

You *cannot* create an event port from the Services node; you must create it from the Adapters node.

To create an event port directly from the Adapters node:

1. Right-click a node under BAPI, RFC, or IDOC.
2. Select **Create Event Port**.



The Create Event Port dialog box is displayed. Perform the following steps:

a. Enter a name for the event port and provide a brief description.

b. From the **Protocol** drop-down list, select the required disposition, for example, RMI.

c. Enter the disposition URL.

d. Specify the location of your Web service.

3. Click **OK**.

See "Creating an Event Port from the Events Node" on page 2-22 for information on configuring port dispositions.

Creating an Event Port from the Events Node

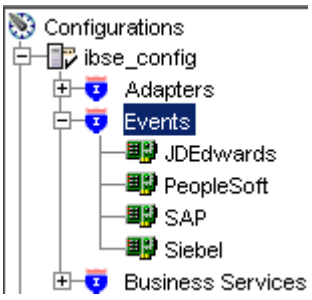
The following procedures describe how to create an event port from the Events node for various dispositions using Application Explorer. You can switch between a BSE and a J2CA deployment by choosing one or the other from the menu in the upper right of Application Explorer.

See "Creating an Event Port from the Adapters Node" on page 2-22 for information on creating an event port directly from the Adapters node.

Creating an Event Port for RMI

To create a specific event port for RMI:

1. Click the **Events** node.



2. Expand the **SAP** node.

3. Right-click the **Ports** node and select **Add Port**.

   The Add Port dialog box is displayed. Provide the following information:

   a. Enter a name for the event port and provide a brief description.

   b. From the **Protocol** list, select **RMI**.

   c. In the **URL** field, specify a destination file to which the event data is written using the following format:

   ```
   rmi://host:port;RemoteObject=APPNAME;errorTo=pre-defined port name or
   another disposition url
   ```

   The following table defines the parameters for the disposition.

Parameter	Description
host	The host name or IP address from which the RMI server accepts RMI requests. If you omit this attribute, the RMI server will accept RMI requests from any host.
port	The port number on which the RMI server listens for RMI requests.
RemoteObject	A home or Enterprise Java Bean (EJB) object.
errorTo	Predefined port name or another disposition URL to which error logs are sent.

4. Click **OK**.

   The port appears under the ports node in the left pane. In the right pane, a table appears that summarizes the information associated with the event port you created.

   You can now associate the event port with a channel. See "Creating and Editing a Channel" on page 2-24 for more information.

Editing an Event Port

To edit an event port:

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

   The Edit Port pane is displayed.

3. Make the required changes and click **OK**.

Deleting an Event Port

To delete an event port:

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

   A confirmation dialog box is displayed.

3. To delete the event port you selected, click **OK**.

   The event port disappears from the list in the left pane.

J2CA Filtering Port Option for Integration with BPEL Process Manager

This feature provides the option to use event ports when using OracleAS Adapter for SAP with BPEL PM under a J2CA configuration. Without this feature, all messages from a particular channel are forwarded to the endpoint and then to the BPEL PM Server. In this case, no schema validation takes place. With the filtering feature, you have the option to associate an event schema to a port that you create and configure in Application Explorer. In this case, at runtime the message is validated against the event schema and only the validated event message is forwarded to the endpoint. If the event message does not match the event schema that is associated to the port, then an error message is written to the log and the event message is ignored.

Creating and Editing a Channel

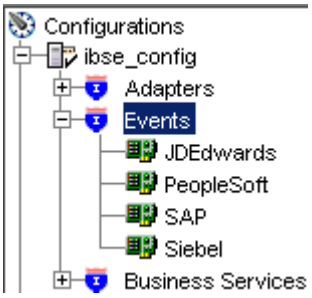
The following procedure describes how to create a channel for your event. All defined event ports must be associated with a channel.

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

Creating a Channel

To create a channel:

1. Click the **Events** node.

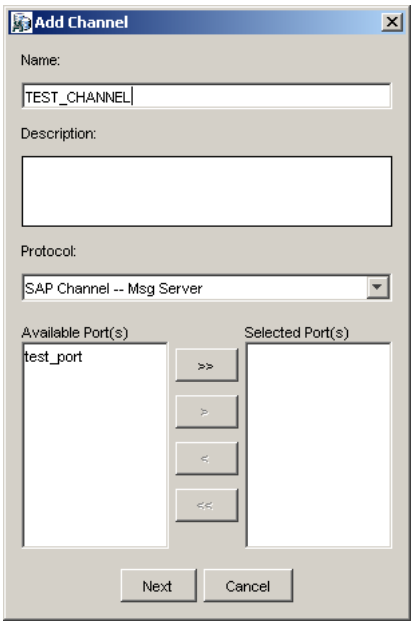


2. Expand the **SAP** node.

   The ports and channels nodes appear in the left pane.

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

The Add Channel dialog box is displayed.



Provide the following information:

a. Enter a name for the channel, for example, TEST_CHANNEL.

b. Enter a brief description.

c. From the **Protocol** list, select **SAP Channel -- Msg Server** or **SAP Channel -- App Server**.

Important: If you are using BPEL Process Manager with a J2CA configuration without the optional port feature, skip the following two steps. See "J2CA Filtering Port Option for Integration with BPEL Process Manager" on page 2-24.

d. Select an event port from the list of available ports. To select more than one, hold down the **Ctrl** key and click the ports.

e. Click **>>** to transfer the port(s) to the list of selected ports.

4. Click **Next**.

The Message Server dialog box is displayed. The following tabs are available:

- System (Required)
- User (Required)
- Advanced
- Preemitter

5. For the **System** tab, enter the appropriate information for your SAP channel based on the information in the following table.

Table 2–5 System Tab Parameters

Target Parameter	Description
Gateway host	A host name for the SAP Gateway.
Gateway service	A service for the SAP Gateway.

Table 2–5 (Cont.) System Tab Parameters

Target Parameter	Description
Program ID of the server	An SAP program ID you want to use for this channel.
Message Server	A host name for the message server.
R/3 name	An SAP R/3 name.
Server group	An SAP server group.

6. For the **User** tab, enter the appropriate information for your SAP channel based on the information in the following table.

Table 2–6 User Tab Parameters

Target Parameter	Description
Client	The client number defined for the SAP application for client communications.
User	A valid user ID for the SAP application.
Password	A valid password for the SAP application.
Language	A language key. EN (English) is the default.
Code page	A character code page value.

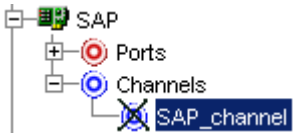
7. For the **Advanced** tab (optional), enter the appropriate information for your SAP channel based on the information in the following table.

Table 2–7 Advanced Tab Parameters

Target Parameter	Description
IDOC Format	Select an IDoc type from the list.
User Defined Function Modules	Enter the path to the user-defined function module you created.
SAP trace	Select this check box if you want to enable SAP traces for troubleshooting purposes.
Unicode	Select this check box if you are expecting your response in Unicode format.
Processing Mode	Select the type of synchronous processing from the list.

8. Click **OK**.

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

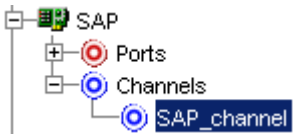


   An X over the icon indicates that the channel is currently disconnected. You must start the channel to activate your event configuration.

Note: If you are using OracleAS Adapter for SAP with BPEL Process Manager, do not start the channel, as it is managed by the BPEL PM Server. If you start the channel for testing and debugging purposes, stop it before runtime.

9. Right-click the **channels** node and select **Start**.

   The channel you created becomes active.



   The X over the icon disappears.

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

    The channel becomes inactive and an X appears over the icon.

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 Channel pane is displayed.

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

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 box is displayed.

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

   The channel disappears from the list in the left pane.

3

OC4J Deployment and Integration

This chapter describes Oracle Containers for J2EE (OC4J) deployment and integration with OracleAS Adapter for SAP.

This chapter discusses the following topics:

- Adapter Integration with OC4J
- Deployment of Adapter
- Updating Adapter Configuration
- How to Write a Java Application Client Using the CCI API

See Also:

- *Oracle Application Server Adapter Concepts*

Adapter Integration with OC4J

OracleAS Adapter for SAP is deployed within an OC4J container during installation. All client applications run within the OC4J environment. In J2CA deployment, the Common Client Interface (CCI) integrates an OC4J client application with a resource adapter.

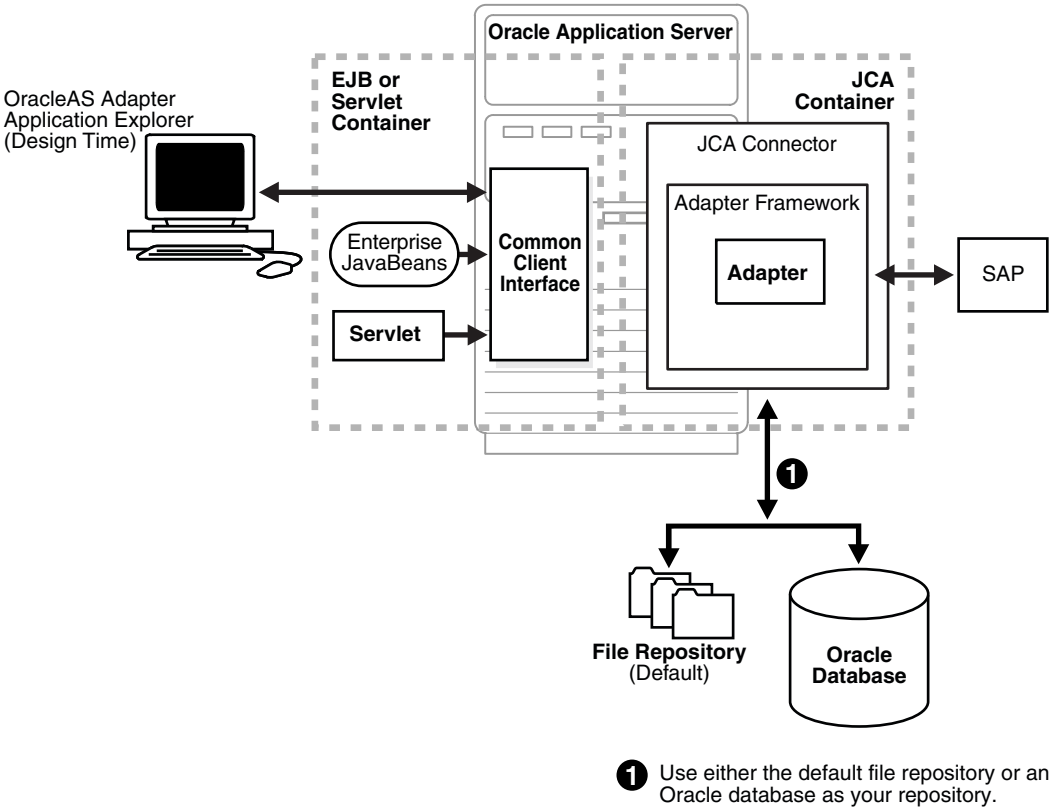
See Also:

- "Oracle Application Server Adapters Integration with OC4J" in *Oracle Application Server Adapter Concepts*

Deployment of Adapter

Figure 3–1 shows deployment of the J2CA Connector to the Oracle Application Server. In a runtime service scenario, an Enterprise Java Bean (EJB), 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



See Also:

- *Oracle Application Server Adapter Concepts*

Updating Adapter Configuration

During the J2CA deployment of OracleAS Adapter for SAP, OC4J generates a deployment descriptor called `oc4j-ra.xml`, located in *`OC4J_home`*`\integration\orabpel\system\appserver\oc4j\j2ee\home\application-deployments\default\iwafjca`.

> **Note:** Your installation contains more than one file named `oc4j-ra.xml`. The OC4J deployment descriptor described in this section is located in the directory specified.

Creating a Managed Connector Factory Object

The `oc4j-ra.xml` descriptor provides OC4J-specific deployment information for resource adapters. For example, the default `jca_sample` configuration in Application Explorer is represented in the `oc4j-ra.xml` file as follows:

```
<?xml version="1.0"?>
<!DOCTYPE oc4j-connector-factories PUBLIC "-//Oracle//DTD Oracle Connector >
9.04//EN" "http://xmlns.oracle.com/ias/dtds/oc4j-connector-factories-9_04.dtd"
<oc4j-connector-factories>
   <connector-factory location="eis/OracleJCAAdapter/DefaultConnection"
connector-name="IWAFJCA10">
```

```
   <config-property name="IWayHome" value="../../adapters/application"/>
   <config-property name="IWayConfig" value="jca_sample"/>
   <config-property name="IWayRepoURL" value=""/>
   <config-property name="IWayRepoUser" value=""/>
   <config-property name="IWayRepoPassword" value=""/>
   <config-property name="logLevel" value="debug"/>
   </connector-factory>
</oc4j-connector-factories>
```

The parameters defined in the oc4j-ra.xml file are described in the following table:

Parameter Name	Description
IWayHome	The base installation directory for the OracleAS packaged application adapter.
IWayConfig	The adapter configuration name as defined in Application Explorer. For example, OracleAS Adapter for SAP 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 BSE repository. See"Configuring an Oracle Repository" in Chapter 2, "Configuring OracleAS Adapter for SAP" for more information.
IWayRepoUser	User name to use when connecting to the database. This is necessary only when using an Oracle database as the BSE repository. See"Configuring an Oracle Repository" in Chapter 2, "Configuring OracleAS Adapter for SAP" for more information.
IWayRepoPassword	Password. If provided, it overwrites configuration. This is necessary only when using an Oracle database as the BSE repository. See"Configuring an Oracle Repository" in Chapter 2, "Configuring OracleAS Adapter for SAP" for more information.
loglevel	It overwrites the level set by the ManagedConnectorFactory property.

Creating Multiple Managed Connector Factory Objects

To establish multiple managed connector factory objects, you must edit the `oc4j-ra.xml` file and add more `<connector-factory>` nodes. For example, the default `jca_sample` configuration in Application Explorer is represented in the `oc4j-ra.xml` file as follows:

```
<?xml version="1.0"?>
<!DOCTYPE oc4j-connector-factories PUBLIC "-//Oracle//DTD Oracle Connector
9.04//EN" "http://xmlns.oracle.com/ias/dtds/oc4j-connector-factories-9_04.dtd">
<oc4j-connector-factories>
   <connector-factory location="eis/OracleJCAAdapter/DefaultConnection"
connector-name="IWAFJCA10">
   <config-property name="IWayHome" value="../../adapters/application"/>
   <config-property name="IWayConfig" value="jca_sample"/>
   <config-property name="IWayRepoURL" value=""/>
```

```
   <config-property name="IWayRepoUser" value=""/>
   <config-property name="IWayRepoPassword" value=""/>
   <config-property name="logLevel" value="debug"/>
   </connector-factory>
</oc4j-connector-factories>
```

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

```
<?xml version="1.0"?>
<!DOCTYPE oc4j-connector-factories PUBLIC "-//Oracle//DTD Oracle Connector
9.04//EN" "http://xmlns.oracle.com/ias/dtds/oc4j-connector-factories-9_04.dtd">
<oc4j-connector-factories>
   <connector-factory location="eis/OracleJCAAdapter/DefaultConnection1"
connector-name="IWAFJCA10">
   <config-property name="IWayHome" value="../../adapters/application"/>
   <config-property name="IWayConfig" value="jca_sample"/>
   <config-property name="IWayRepoURL" value=""/>
   <config-property name="IWayRepoUser" value=""/>
   <config-property name="IWayRepoPassword" value=""/>
   <config-property name="logLevel" value="debug"/>
   </connector-factory>
<connector-factory location="eis/OracleJCAAdapter/DefaultConnection2"
connector-name="IWAFJCA10">
   <config-property name="IWayHome" value="../../adapters/application"/>
   <config-property name="IWayConfig" value="jca_sample2"/>
   <config-property name="IWayRepoURL" value=""/>
   <config-property name="IWayRepoUser" value=""/>
   <config-property name="IWayRepoPassword" value=""/>
   <config-property name="logLevel" value="debug"/>
   </connector-factory>
</oc4j-connector-factories>
```

How to Write a Java Application Client Using the CCI API

The following example shows the code structure for using CCI with packaged application adapters. The code sample is shown in four steps.

Step 1. Obtain the Connection Factory

The connection factory is obtained by JNDI lookup.

```
InitialContext context = new InitialContext();
ConnectionFactory cf = (ConnectionFactory)context.lookup(iwayJndi)
```

Step 2. Obtaining a Connection for the Adapter

IWAFConnectionSpec is an implementation of ConnectionSpec used for creating a design time or runtime service adapter connection. The ConnectionSpec has seven parameters. Connection Pooling is fully supported and established based on these parameters, except log level.

Parameter Name	Description
adapterName	Name of the packaged application adapter.
config -	Adapter configuration name. NOT REQUIRED FOR IWAEAdapter.
language	Default is `en`.
country	Default is `us`.

Parameter Name	Description
userName	User name. If provided, it overwrites configuration.
password	Password. If provided, it overwrites configuration.
logLevel	It overwrites the level set by the ManagedConnectionFactory property.

Note: Currently the OracleAS Adapter J2CA supports only basic security mapping. The DEBUG log level provides detaild information on the mapping behavior. It functions as follows:

- If the user name and password are not set, and no security is provided by the application server, the OracleAS Adapter J2CA will still let it pass and rely on the adapter configuration security information.
- If the user name and password are set, these values will overwrite the adapter configuration. The OracleAS Adapter J2CA compares this information with the security information provided by the application server and log in case the values do not match. However, it still allows the information through.

The iWAFConnectionSpec can be made to initiate an interaction with SAP by specifying the adapter name and configuration parameters in the ConnectionSpec. For example,

```
iWAFConnectionSpec cs = new IWAFConnectionSpec();
  cs.setAdapterName(ADAPTER);
  cs.setConfig(TARGET);
  cs.setLogLevel(LOG_LEVEL);  // Adapter layer log level
  Connection c = cf.getConnection(cs);// where cf is the connection factory
```

In this snippet, `ADAPTER` and `TARGET` refer to the adapter being deployed and the name of a target defined in Application Explorer, respectively. See "Complete Code Sample" on page 3-6 for more information.

Step 3. Create Interaction with InteractionSpec for Runtime

```
Interaction i = c.createInteraction();
  IWAFInteractionSpec is = new IWAFInteractionSpec();
  is.setFunctionName(IWAFInteractionSpec.PROCESS);
```

Two functions can be set: PROCESS and IWAE. PROCESS is used at runtime. IWAE is used when you are using the IAEAdapter at design time.

Step 4. Create Input Record and Run Interaction

In this case, to complete the EIS invocation, an SAP RFC message is referenced. The schema is provided by Application Explorer.

A standard J2CA Indexed Record is used in this example:

```
// Use JCA IndexRecord, named "input" for runtime processing.
IndexedRecord rIn = cf.getRecordFactory().createIndexedRecord("input");
rIn.add(msg_run);
  IndexedRecord rOut = (IndexedRecord)i.execute(is, rIn);
```

```
System.out.println((String)rOut.get(0));
```

A special record is supported in this example:

```
//IWAFRecord rIn = new IWAFRecord("input");
//rIn.setRootXML(msg_run);
//IWAFRecord response = executeRunInteraction(c, rIn);
  //IWAFRecord rOut = (IWAFRecord)i.execute(is, rIn);
//System.out.println(rOut.getRootXML());
```

Where `msg_run` is an instance XML document generated from the schema created by Application Explorer. For example, the following is a sample SAP request XML document.

```
<?xml version="1.0" encoding="UTF-8"?>
<BAPI_CUSTOMER_GETDETAIL2>
<COMPANYCODE></COMPANYCODE>
<CUSTOMERNO>0000401026</CUSTOMERNO>
</BAPI_CUSTOMER_GETDETAIL2>
```

Complete Code Sample

The following is a sample of the complete code:

```
import javax.resource.cci.*;
import com.ibi.afjca.cci.*;
import com.ibi.afjca.spi.*;

/**
 * The purpose of this sample is to illustrate how to use the IWAF Universal
 * JCA connector.
 *
 */
public class IWAFJCASimple {

private static String  HOME    = "c:/iway/xfoc/components/iwafcont/dist";
private static String  CONFIG    = "base";
private static String  LOG_LEVEL = "FATAL";

private static String ADAPTER = "SAP";
private static String TARGET  = "SAP_connection";

// Input Message
private static String msg_run = "<SAP/>";

  public static void main(String[] args) throws Exception {

// 1. Getting the Connection factory through JNDI lookup
// -----------------------------------------------------
  InitialContext context = new InitialContext();
  ConnectionFactory cf = (ConnectionFactory)context.lookup(iwayJndi)
  // 2. Getting a connection for a particular adapter target, in this case SAP
// -----------------------------------------------------
  IWAFConnectionSpec cs = new IWAFConnectionSpec();
  cs.setAdapterName(ADAPTER);
  cs.setConfig(TARGET);
  cs.setLogLevel(LOG_LEVEL);  // Adapter layer log level
  Connection c = cf.getConnection(cs);// where cf is the connection factory

  // 3. Create interaction with interactionSpec for RUNTIME
// -----------------------------------------------------
  Interaction i = c.createInteraction();
```

```
  IWAFInteractionSpec is = new IWAFInteractionSpec();
  is.setFunctionName("PROCESS");

  // 4. Create input Record and execute interaction
// -----------------------------------------------------

  // 4.1 Using JCA standard Indexed Record
// Use JCA IndexRecord, named "input" for runtime processing.
IndexedRecord rIn = cf.getRecordFactory().createIndexedRecord("input");
rIn.add(msg_run);
  IndexedRecord rOut = (IndexedRecord)i.execute(is, rIn);
System.out.println((String)rOut.get(0));

  // 4.2 Our own Record is supported here
//IWAFRecord rIn = new IWAFRecord("input");
//rIn.setRootXML(msg_run);
//IWAFRecord response = executeRunInteraction(c, rIn);
  //IWAFRecord rOut = (IWAFRecord)i.execute(is, rIn);
//System.out.println(rOut.getRootXML());

  } // main()

}
```

4

Integration with BPEL Process Manager

OracleAS Adapter for SAP integrates seamlessly with Business Process Execution Language (BPEL) Process Manager to facilitate Web service integration. 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 includes the following topics:

Overview of Adapter Integration with BPEL Process Manager

To integrate with BPEL Process Manager, OracleAS Adapter for SAP must be deployed in the same OC4J container as BPEL Process Manager. The underlying adapter services must be exposed as WSDL files, which are generated during design time in Oracle Application Server Adapter Application Explorer (Application Explorer) for both request-response (outbound) and event notification (inbound) services of the adapter. See "Generating WSDL (J2CA Configurations Only)" on page 2-18 for more information.

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 a BPEL designer and deployed to a BPEL server. Upon deployment to the BPEL server, every newly built process is automatically deployed to the Oracle BPEL Console, where you run, monitor, and administer BPEL processes, as well as listen to adapter events.

When using the adapter with BPEL Process Manager installed on OracleAS Middle Tier, your middle-tier BPEL PM home directory is OC4J_BPEL, located as follows:

```
OracleAS_home\j2ee\OC4J_BPEL
```

See Also:

- *Oracle Application Server Adapter Concepts*
- *Oracle BPEL Process Manager Developer's Guide*

Deployment of Adapter

During installation, OracleAS Adapter for SAP is deployed as a J2CA 1.0 resource adapter within the OC4J J2CA container. The adapter must be deployed in the same OC4J container as BPEL Process Manager.

See Also: *Oracle Application Server Adapter Concepts*

Design Time

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

- OracleAS Adapter Application Explorer (Application Explorer)
- Oracle JDeveloper BPEL Designer (JDeveloper) or Eclipse

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

Before you design a BPEL process, you must create a schema and generate the respective WSDL file using Application Explorer. See "Generating WSDL (J2CA Configurations Only)" on page 2-18 for more information.

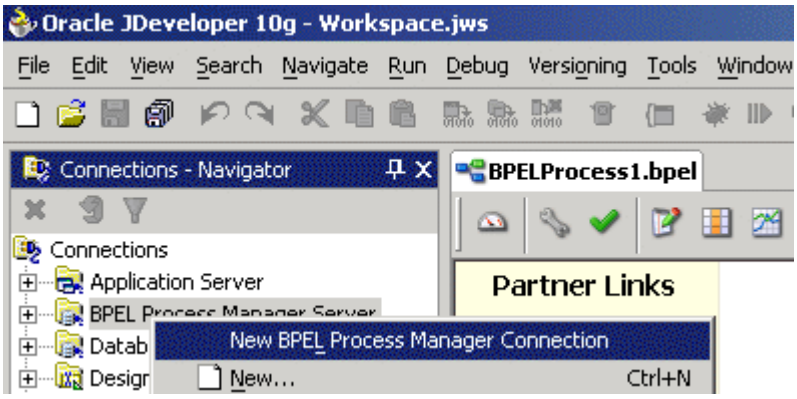
Design a BPEL Process for Request-Response Service (Outbound)

An outbound BPEL process consists of PartnerLink, Invoke, and Assign process activities. You must first create a new BPEL Process Manager connection and a synchronous BPEL process template.

Create a New Connection to BPEL PM Server

To create a new BPEL Process Manager connection:

1. Display the connections by clicking the **Connections** tab at the bottom of the upper left pane in JDeveloper.

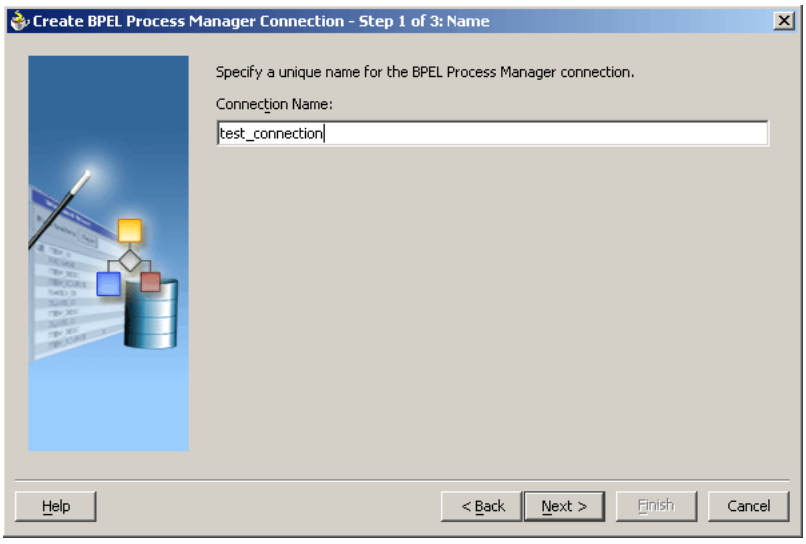


2. Right-click **BPEL Process Manager Server** and select **New BPEL Process Manager Connection**.

   The Create BPEL Process Manager Connection - Welcome dialog box is displayed.

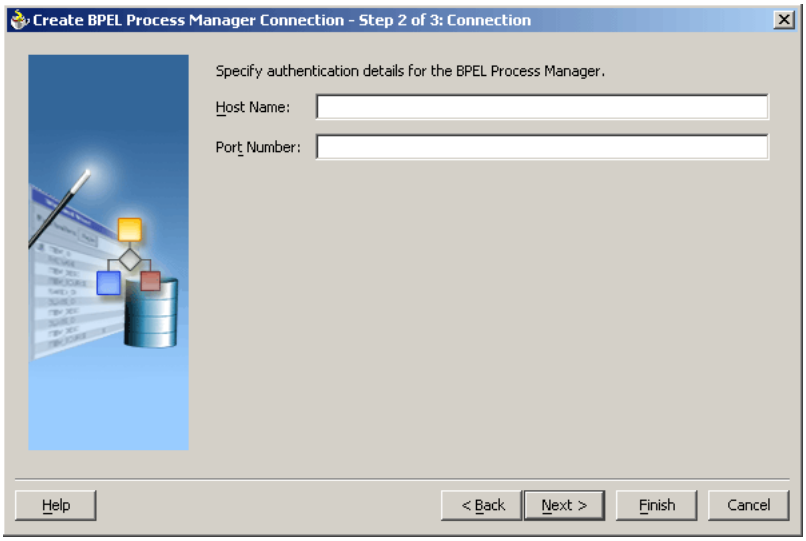
3. Click **Next**.

The Create BPEL Process Manager Connection dialog box is displayed.



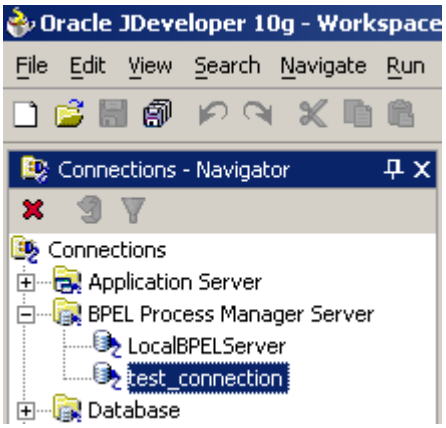
4. Specify a unique name for your BPEL Server connection and click **Next**.

   The Create BPEL Process Manager Connection dialog box is displayed.



5. Specify a valid host name and port number for the BPEL PM Server you wish to connect to.
6. Click **Finish**.

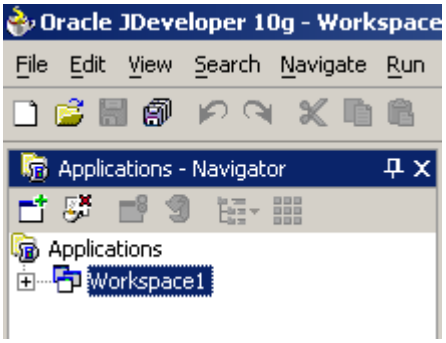
Your newly created server connection is displayed in the Connections tab under the BPEL Process Manager Server node.



Create a New BPEL Project for Outbound Interaction (Synchronous Process)

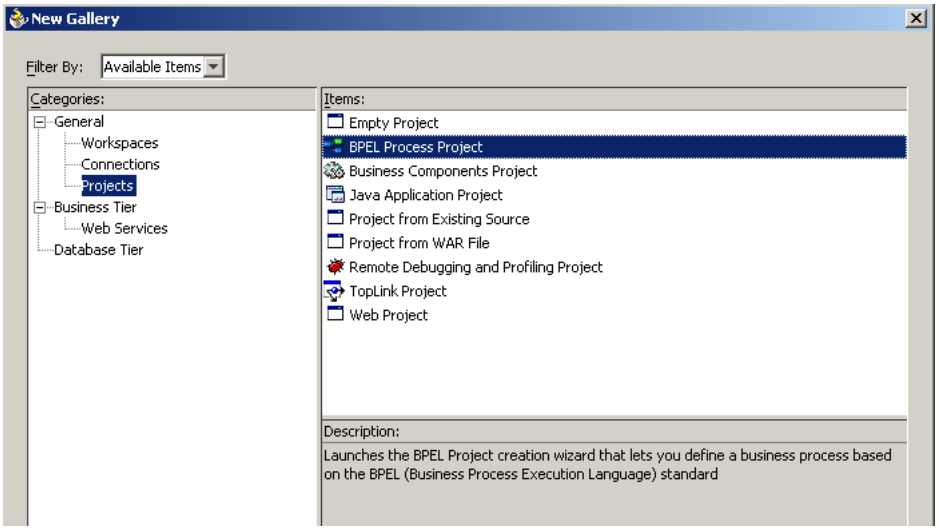
To create a new BPEL project for a synchronous process:

1. At the bottom of the upper left pane, click the **Applications** tab and select a workspace for your project.



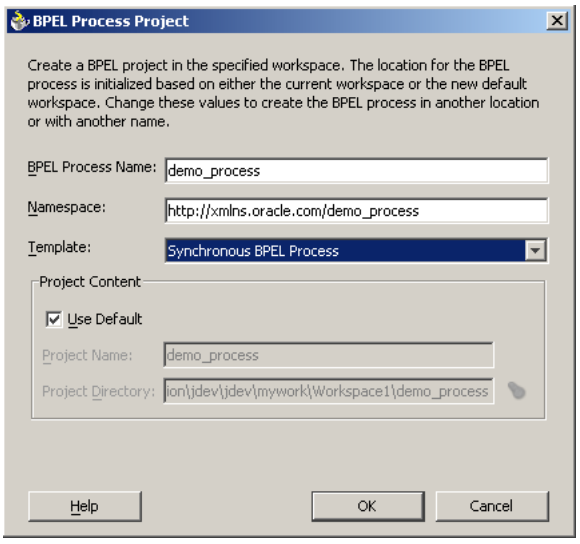
2. Right-click the workspace and select **New Project**.

   The New Gallery window is displayed.



3. From the Items list, select **BPEL Process Project** and click **OK**.

The BPEL Process Project dialog box is displayed.



4. Perform the following steps:

   a. Specify a name for the BPEL process.

      The Namespace field is updated automatically.

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

   c. Click **OK**.

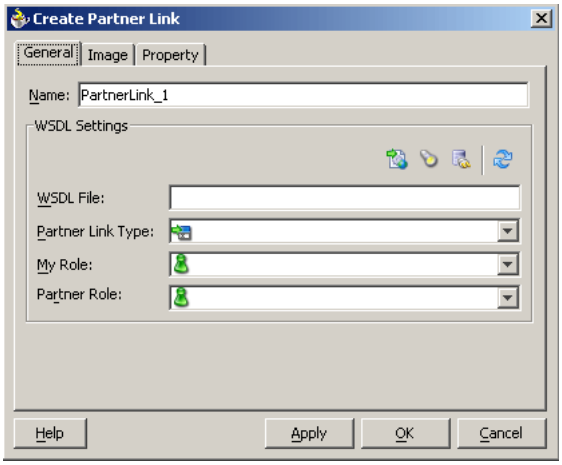
Create an Outbound PartnerLink Activity

When designing a BPEL process, a PartnerLink activity must be created to invoke the SAP service. A PartnerLink describes a set of operations within a Web service. The WSDL document is the external contract to which the Web service conforms. Given a WSDL, any BPEL process can initiate a Web service through a PartnerLink.

To create an outbound PartnerLink using the WSDL file you generated in Application Explorer:

1. From the Process Activities pane on the right, drag and drop a PartnerLink to the visual editor.

   The Create Partner Link dialog box is displayed.

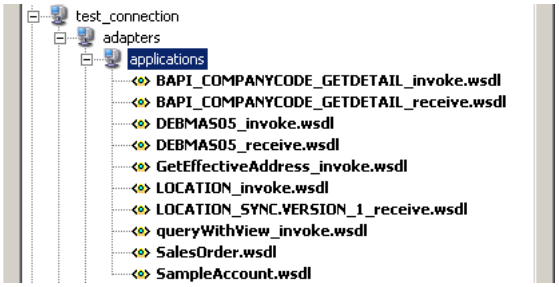


2. Click the **WSIL browser** icon (second icon from the left preceding the **WSDL File** field).

   The WSDL Chooser dialog box is displayed.

3. Expand your new connection, then expand **adapters**, and then **applications**.

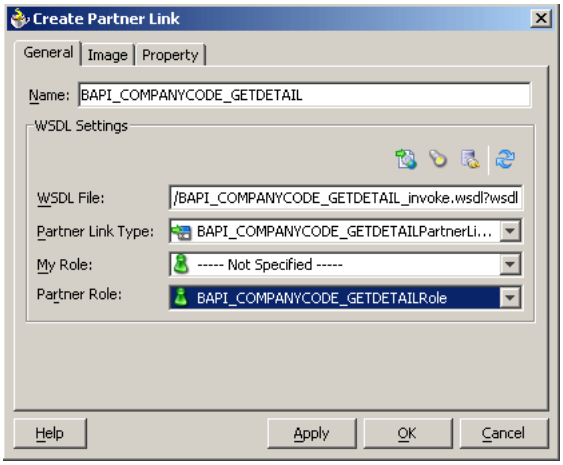
   The WSDL tree displayed in the WSDL Chooser dialog box lists any WSDL files you have created using Application Explorer. The WSDL tree is generated by a WSDL servlet, which is automatically deployed as part of the BPEL Server installation.



> **Note:** If you have organized your WSDL files in subfolders, the WSIL browser will display the full tree structure of your WSDL hierarchy. By default, the names of all WSDL files generated for outbound adapter services end with `_invoke`.

4. Select **BAPI_COMPANYCODE_GETDETAIL_invoke.wsdl** and click **OK**.

   The **WSDL File** field in the Create Partner Link dialog box displays the name and location of the selected WSDL file. The **Partner Link Type** field specifies the PartnerLink defined in the WSDL file.



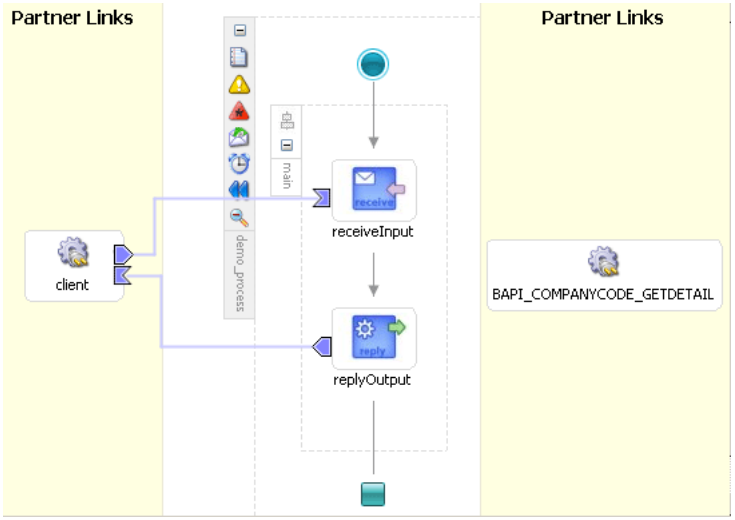
Perform the following steps:

a. Leave the **My Role** field unspecified. The role of the PartnerLink is null, as it will be synchronously invoked from the BPEL process.

b. From the **Partner Role** list, select the default value **BAPI_COMPANYCODE_GETDETAILRole**. This is the role of the BPEL process.

5. Click **OK**.

The new PartnerLink appears in the visual editor.



6. Select **Save** from the **File** menu.

Create an Outbound Invoke Activity

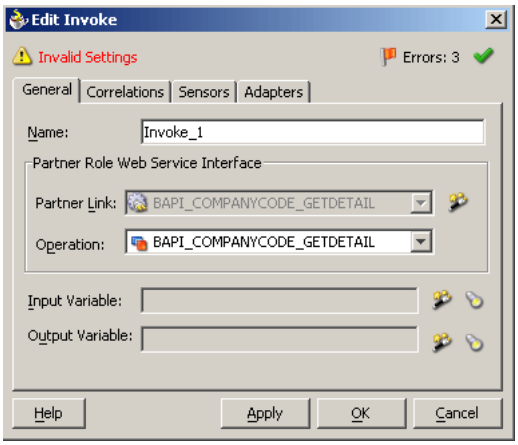
This activity enables you to specify an operation you want to invoke for the service identified by its PartnerLink. The Invoke activity opens a port in the process that is used to send and receive data. It uses this port to submit required data and receive a response. For synchronous callbacks, only one port is needed for both the send and the receive functions.

To create an outbound Invoke activity:

1. From the **Process Activities** pane on the right, drag an **Invoke** activity to the visual editor and place it between the Receive activity (`receiveInput`) and the Reply activity (`replyOutput`).

2. Extend a connection between the Invoke activity and your newly-created PartnerLink.

The Edit Invoke dialog box is displayed. Note that the Partner Link and Operation fields are automatically populated with your WSDL files.



Note: Ignore any invalid settings and error warnings.

Perform the following steps:

a. In the **Name** field, provide a meaningful name for the Invoke activity.

b. Click the first icon to the right of the **Input Variable** field, then click **OK** in the Create Variable window that is displayed.

   A global variable is automatically created in the Input Variable field.

c. Click the first icon to the right of the **Output Variable** field, then click **OK** in the Create Variable window that is displayed.

   A global variable is automatically created in the Output Variable field.

d. Click **Apply**.

   The Edit Invoke window should no longer display any warnings or errors.

3. Click **OK**.
4. Select **Save** from the **File** menu.

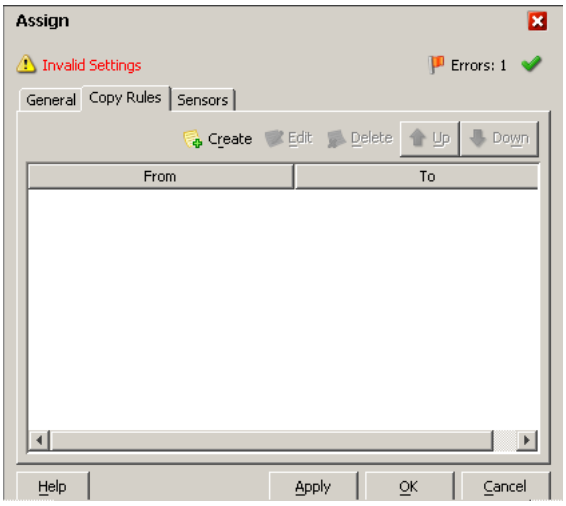
Create an Assign Activity

An Assign activity provides a method for simple data manipulation, such as copying the contents of one variable to another. This Assign activity maps the input variable of the SAP process to the SAP PartnerLink input.

To create an Assign activity:

1. From the **Process Activities** pane on the right, drag an **Assign** activity to the visual editor and place it between the Receive activity (`receiveInput`) and the new Invoke activity (`GetCompanyDetail`).
2. Double-click the **Assign** activity icon.

The Assign dialog box is displayed.



Note: Ignore any invalid settings and error warnings.

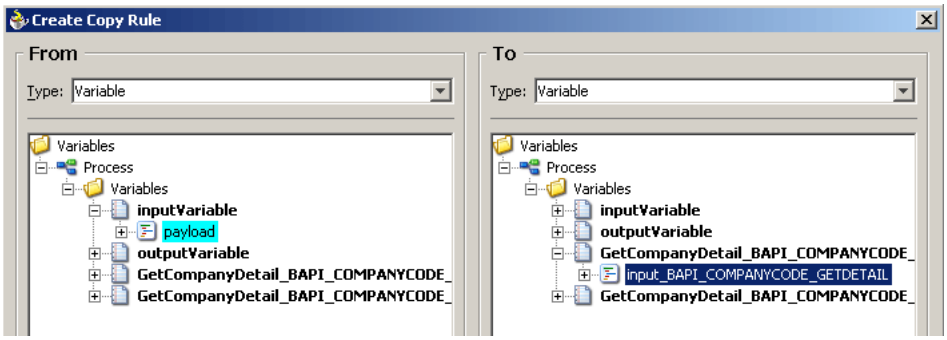
3. In the Copy Rules tab, click **Create**.

   The Create Copy Rule dialog box is displayed.

   a. In the **From** pane, expand **Variables**, then **inputVariable**, and then highlight **payload**.

   b. In the **To** pane, expand **Variables**, then **GetCompanyDetail_BAPI_COMPANYCODE_GETDETAIL_InputVariable**, and then highlight **input_BAPI_COMPANYCODE_GETDETAIL**.

   Your Create Copy Rule dialog box should look as follows:



4. To close the Create Copy Rule dialog box and the Assign dialog box, click **OK**.

Create a Second Assign Activity

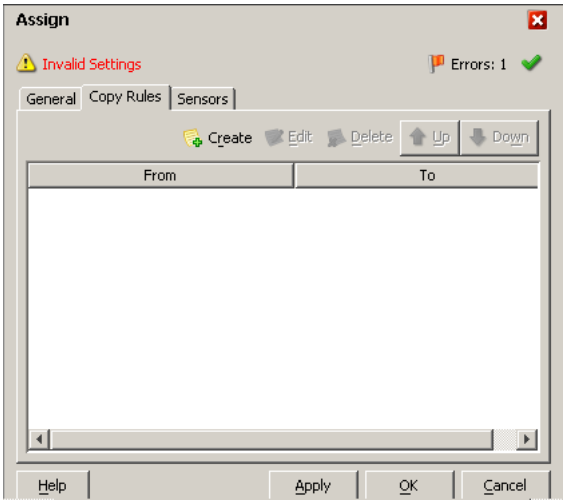
This Assign activity maps the output variable of the SAP process to the SAP PartnerLink output.

To create a second Assign activity:

1. From the **Process Activities** pane on the right, drag another **Assign** activity to the visual editor and place it between the Invoke activity (`GetCompanyDetail`) and the Reply activity (`replyOutput`).

2. Double-click the **Assign** activity icon.

   The Assign settings dialog box is displayed.



> **Note:** Ignore any invalid settings and error warnings.

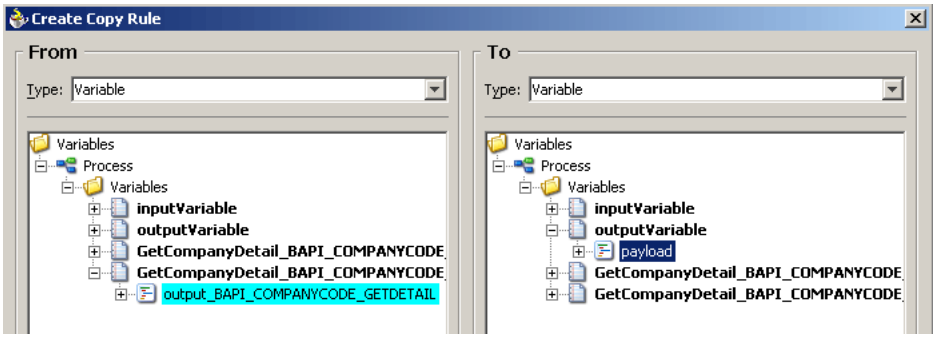
3. In the Copy Rules tab, click **Create**.

   The Create Copy Rule dialog box is displayed. Perform the following steps:

   a. In the **From** pane, expand **Variables**, then **GetCompanyDetail_BAPI_COMPANYCODE_GETDETAIL_OutputVariable**, and then highlight **output_BAPI_COMPANYCODE_GETDETAIL**.

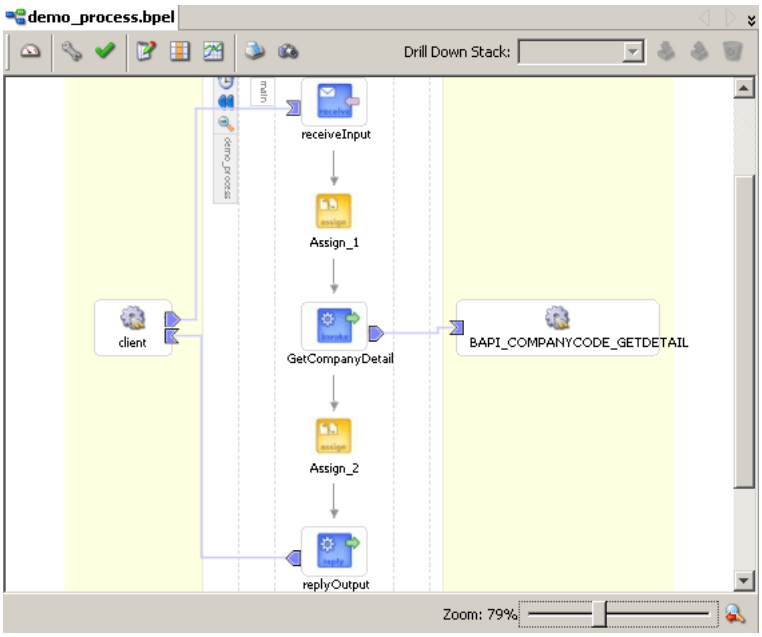
   b. In the **To** pane, expand **Variables**, then **outputVariable**, and then highlight **payload**.

   Your Create Copy Rule dialog box should look as follows:



4. To close the Create Copy Rule dialog box and the Assign dialog box, click **OK**.
5. Select **Save** from the **File** menu.

The following image shows the diagram view of your completed BPEL process.



See "Invoking Adapter Request-Response Service from BPEL Process Manager" on page 4-16 for information on how to deploy and manage your outbound process.

See Also:

- *Oracle BPEL Process Manager Developer's Guide*
- *Oracle Application Server Adapter Concepts*

Design a BPEL Process for Event Handling (Inbound)

An inbound BPEL process consists of a PartnerLink and a Receive process activity. You must first create a channel and a new BPEL Process Manager Server connection. See Chapter 5, "BPEL Process Manager Integration Examples" for instructions on how to perform these procedures.

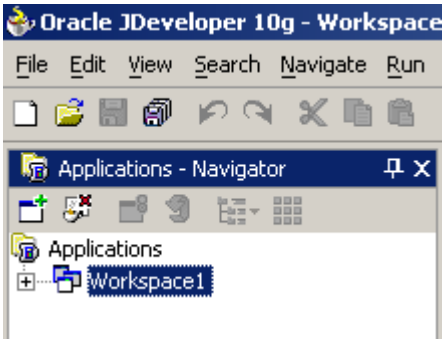
Note: You must create a separate channel for every event and select that channel when you generate WSDL for inbound interaction using Application Explorer. Do not start the channel in Application Explorer, as BPEL Process Manager manages endpoint activation independently. See "SAP Event Integration" on page 5-11 for more information.

Create a New BPEL Project for Inbound Interaction (Empty Process)

Before you create a BPEL project, verify that your BPEL Server is running. After you have created a new server connection, you are ready to design an empty process template for your BPEL project.

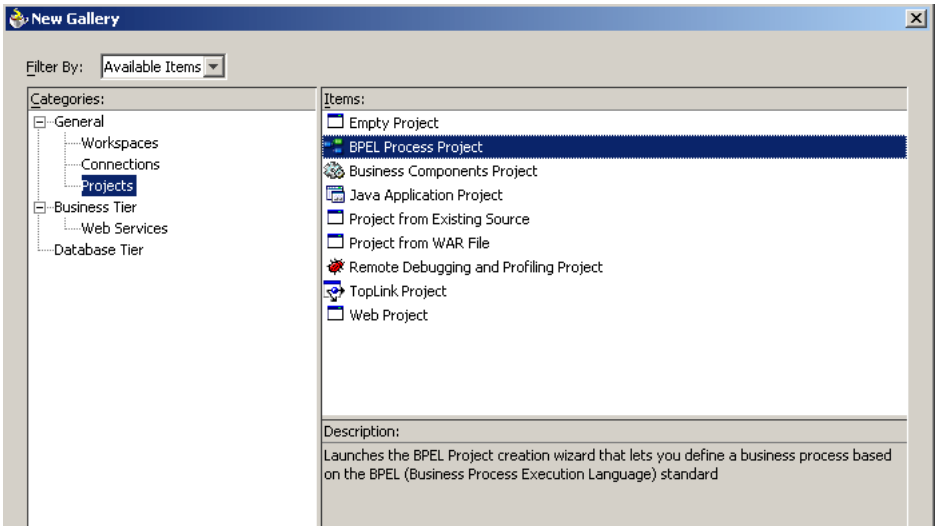
To create a new BPEL project for inbound interaction:

1. Click the **Applications** tab and select a workspace for your project.



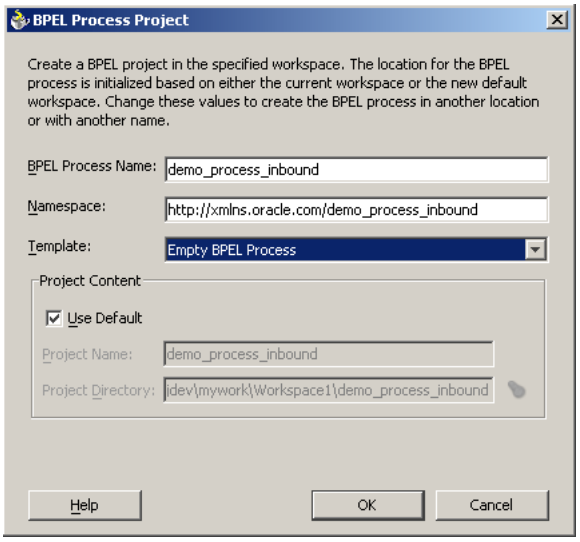
2. Right-click the workspace and select **New Project**.

   The New Gallery window is displayed.



3. From the Items list, select **BPEL Process Project** and click **OK**.

   The BPEL Process Project dialog box is displayed.



4. Perform the following steps:
   a. Specify a name for the process.

      The Namespace field is updated automatically.

   b. From the Template list, select **Empty BPEL Process**.

   c. Click **OK**.

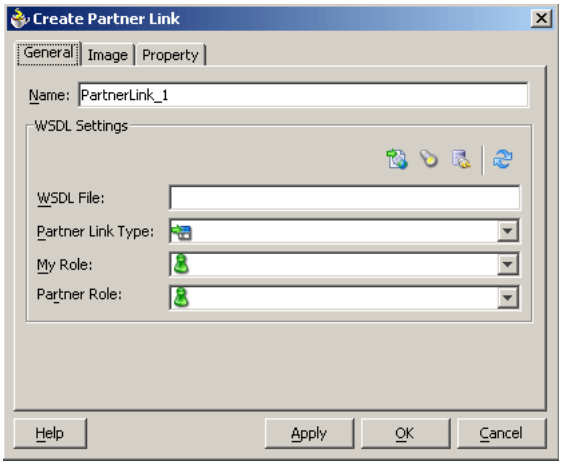
Create an Inbound PartnerLink Activity

When designing a BPEL process, a PartnerLink activity must be created to invoke the SAP service. A PartnerLink describes a set of operations within a Web service. The WSDL document is the external contract to which the Web service conforms. Given a WSDL, any BPEL process can initiate a Web service through a PartnerLink.

To create an inbound PartnerLink using the WSDL file you generated in Application Explorer:

1. From the Process Activities pane on the right, drag and drop a PartnerLink to the visual editor.

   The Create Partner Link dialog box is displayed.

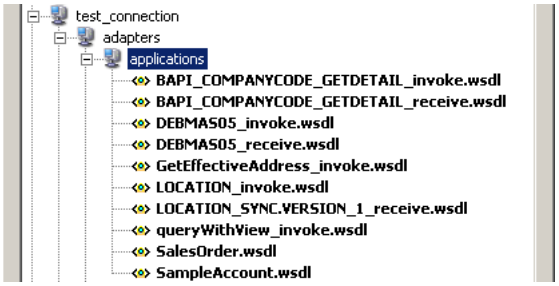


2. Click the **WSIL browser** icon (second icon from the left preceding the **WSDL File** field).

   The WSDL Chooser dialog box is displayed.

3. Expand your new connection, then expand **adapters**, and then **applications**.

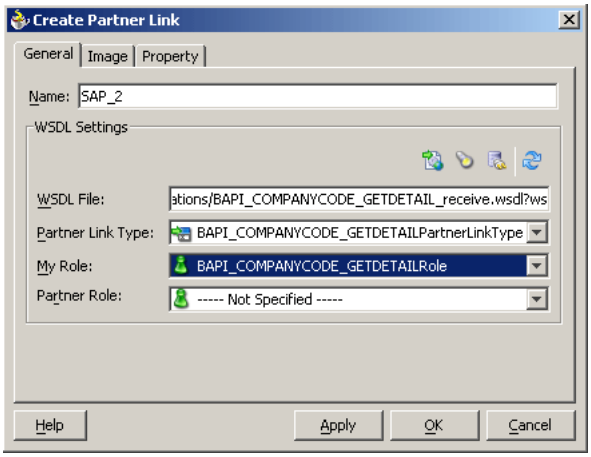
The WSDL tree displays the WSDL files you created using Application Explorer. The WSDL tree is generated by a WSDL servlet, which is automatically deployed as part of the BPEL Server installation.



Note: If you have organized your WSDL files in subfolders, the WSIL browser will display the full tree structure of your WSDL hierarchy. By default, the names of all WSDL files generated for inbound adapter services end with `_receive`.

4. Select **BAPI_COMPANYCODE_GETDETAIL_receive.wsdl** and click **OK**.

   The Create Partner Link dialog box is displayed.



   The **WSDL File** field displays the name and location of the selected WSDL file. The **Partner Link Type** field specifies the PartnerLink defined in the WSDL file.

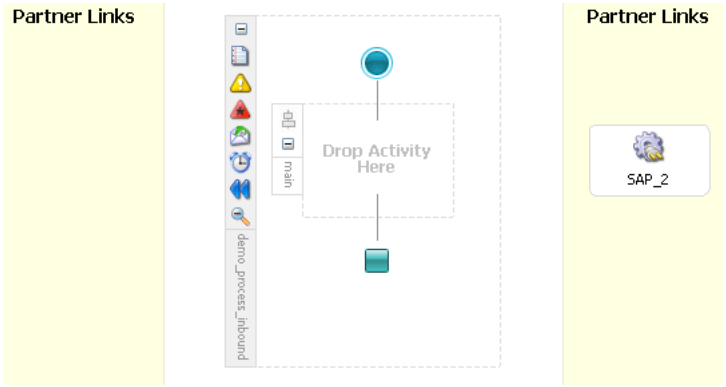
   Perform the following steps:

   a. From the **My Role** list, select the default value **BAPI_COMPANYCODE_GETDETAILRole**.

   b. Leave the **Partner Role** field unspecified.

5. Click **Apply**, and then **OK**.

The new SAP_2 PartnerLink appears in the visual editor.



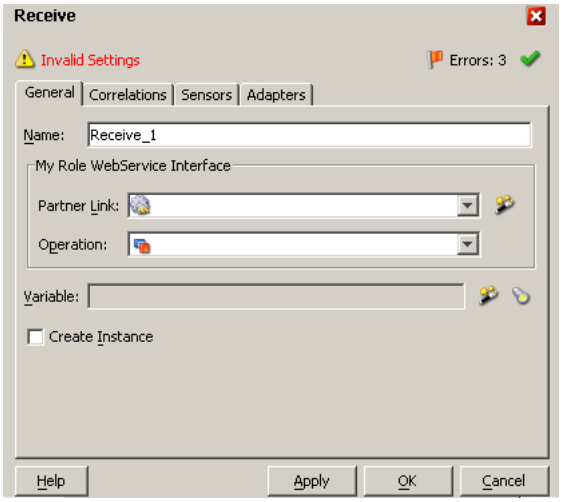
6. Select **Save** from the **File** menu.

Create an Inbound Receive Activity

To create an inbound Receive Activity:

1. From the **Process Activities** pane on the right, drag a **Receive** activity to the visual editor and place it in the designated placeholder labeled **Drop Activity Here**.
2. Double-click the **Receive** activity.

   The Receive dialog box is displayed.



   Perform the following steps:

   a. From the **Partner Link** menu, select the PartnerLink you created in the previous step.

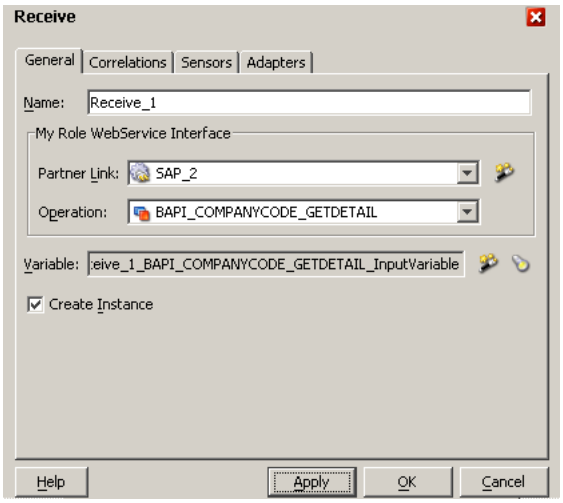
      The **Operation** field is automatically populated.

   b. Click the first icon to the right of the **Variable** field, then click **OK** in the Create Variable dialog box that is displayed.

   c. Verify that the **Create Instance** check box is selected.

3. Click **Apply**.

The Receive dialog box should no longer display any warnings or errors.



4. Click **OK**.

   A connection is created between the PartnerLink and the Receive activity. You have completed the design of your inbound BPEL process.

See "Listening to Adapter Events Inside BPEL Process Manager" on page 4-19 for information on how to deploy and manage your inbound process.

See Also:

- *Oracle BPEL Process Manager Developer's Guide*
- *Oracle Application Server Adapter Concepts*

Invoking Adapter Request-Response Service from BPEL Process Manager

The OracleAS Adapter for SAP request-response service is used to create, delete, update, and query back-end data as well as to call back-end workflows and transactions. The following section describes how to invoke the adapter synchronous request-response service, also referred to as Outbound Interaction, as well as how to manage the process in Oracle BPEL Console.

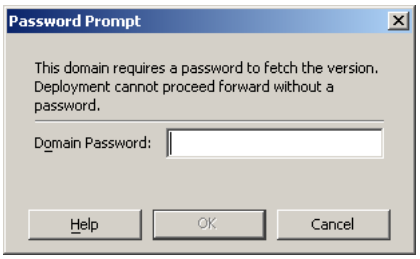
Deploy the Outbound BPEL Process

The procedures for deploying an inbound and an outbound BPEL process using the JDeveloper interface are identical.

To deploy your BPEL process in JDeveloper:

1. Right-click your process flow in the Applications - Navigator pane.
2. Select **Deploy**, then ***Your BPEL PM Server connection***, and then **Deploy to default domain**.

The Password Prompt dialog box is displayed.

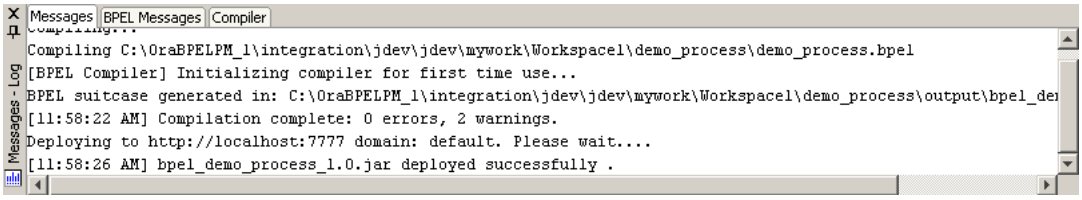


3. In the **Domain Password** field, enter your BPEL Process Manager password.

   The deployment process starts automatically after you enter the correct password.

4. Observe the **Messages** log at the bottom of the window.

   The Messages log displays the deployment status. In this example, it shows a successful deployment message for the process.



   If deployment was not successful, click the **Compiler** tab to view all error and warning messages generated during the deployment process.

Manage the Deployed Outbound Process in Oracle BPEL Console

JDeveloper deploys the developed process directly to the Oracle BPEL Console, which enables youto run, monitor, and administer BPEL processes.

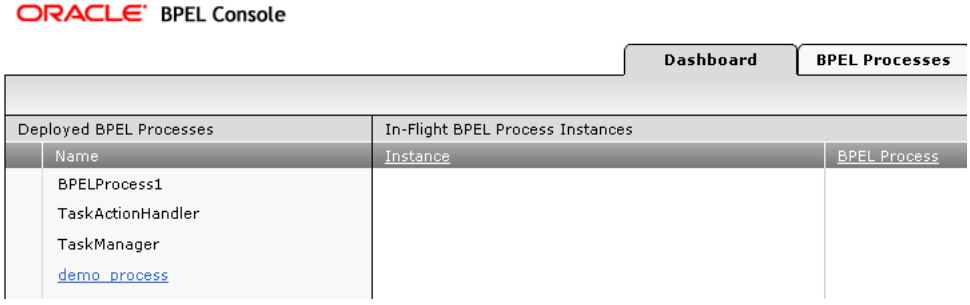
To invoke adapter request-response service:

1. Start the Oracle BPEL Console by entering the following URL in a browser:

   `http://host:port/BPELConsole`

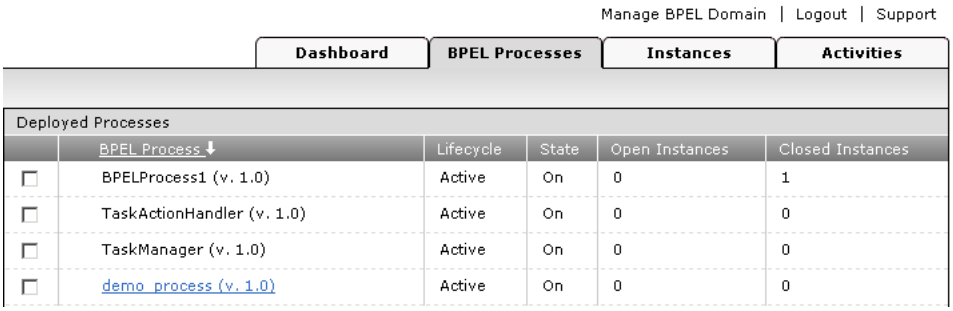
2. Select a domain and provide a valid password.

   The Oracle BPEL Console main page is displayed. All deployed BPEL processes are listed in the Dashboard tab.



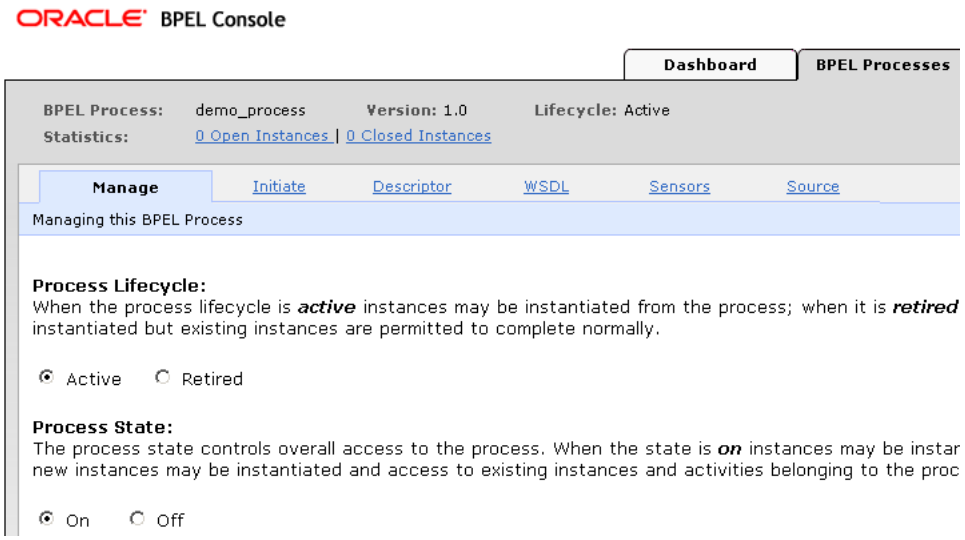
3. Click the **BPEL Processes** tab.

   This tab provides a more detailed view of each deployed process.



4. Click the SAP process link, `demo_process (v. 1.0)`.

   The Manage window provides options for managing this BPEL process. Do not change any of the following default settings.



5. Click the **Initiate** tab.

   The Initiate tab enables you to test your BPEL process.

Perform the following steps:

a. From the **Initiating a test instance** list, select **XML Source**.

b. Enter the following code in the text area provided for XML input:

```
<?xml version="1.0" encoding="UTF-8"?>
<CompanyCode.GetDetail SERVICENAME="CompanyCodeGetDetail" METHODNAME="BAPI_
COMPANYCODE_GETDETAIL" LICENSE="test" CompanyCodeId="1000"/>
```

6. Click **Post XML Message**.

   The response received from the SAP system is displayed in the Initiate window.

   See Also: *Oracle Application Server Adapter Concepts*

Listening to Adapter Events Inside BPEL Process Manager

The OracleAS Adapter for SAP event notification service, also referred to as Inbound Interaction, is used to listen to events that occur in an EIS. The following section describes how to deploy your inbound BPEL process and listen to adapter events at runtime using Oracle BPEL Console.

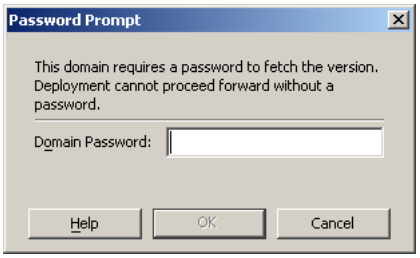
Deploy the Inbound BPEL Process

The procedures for deploying an inbound and an outbound BPEL process using the JDeveloper interface are identical.

To deploy your BPEL process in JDeveloper:

1. Right-click your process flow in the Applications pane.
2. Select **Deploy**, then ***Your BPEL PM Server connection***, and then **Deploy to default domain**.

   The Password Prompt dialog box is displayed.

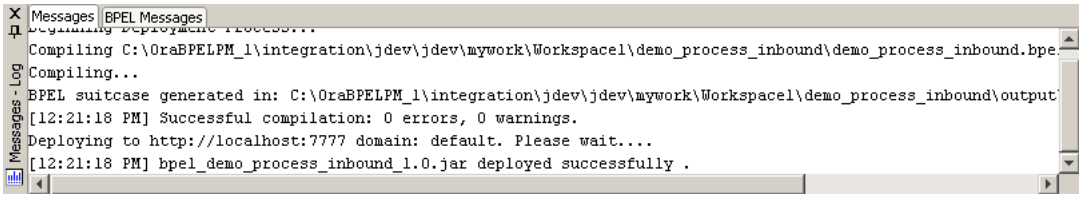


3. In the **Domain Password** field, enter your BPEL Process Manager password.

   The deployment process starts automatically after you enter the correct password.

4. Observe the **Messages** log at the bottom of the window.

   The Messages log displays the deployment status. In this example, it shows a successful deployment message for the process.



If deployment was not successful, click the **Compiler** tab to view all error and warning messages generated during the deployment process.

Listen to Adapter Events in Oracle BPEL Console

JDeveloper deploys the developed process directly to Oracle BPEL Console, which enables youto run, monitor, and administer BPEL processes, as well as to listen to adapter events at runtime using Oracle BPEL Console.

To listen to adapter events:

1. Start the Oracle BPEL Console by entering the following URL in a browser:

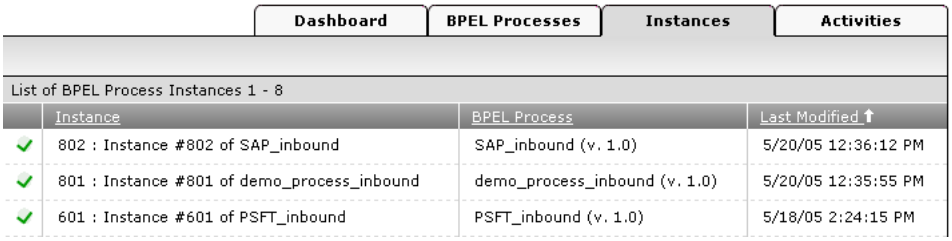
   `http://host:port/BPELConsole`

2. Select a domain and provide a valid password.

   The Oracle BPEL Console Dashboard tab is displayed.

3. Click the **Instances** tab.

   Upon receiving a runtime event, an instance of the event is displayed under the Instances tab.



4. To see the event message, click the instance, and then click **Audit**.

   The event message is displayed.

5. Click **More...** to view the entire message, or **View Raw XML** to view the XML source.

See Chapter 5, "BPEL Process Manager Integration Examples" on page 5-1 for more information.

See Also: *Oracle Application Server Adapter Concepts*

5

BPEL Process Manager Integration Examples

This chapter contains the following examples:

- SAP Service Integration
- SAP Event Integration

The scenarios shown in this chapter require the following prerequisites.

Prerequisites

The following are installation and configuration requirements:

- OracleAS Adapter for SAP must be deployed to Oracle Application Server.
- SAP must be configured for outbound processing. See Appendix A, "Configuring SAP for Inbound and Outbound Processing" for more information.
- Oracle BPEL PM Server must be properly configured and running.
- Oracle JDeveloper must be properly installed.

> **See Also:** *Oracle Application Server Adapters Installation Guide*

The examples in this chapter present the configuration steps necessary for demonstrating service and event integration with SAP. Prior to using this material, you must be familiar with the following:

- How to create a J2CA configuration, as BPEL PM is only compatible with the J2CA Connector. See "Creating a Configuration for J2CA" on page 2-8 for more information.
- How to configure OracleAS Adapter for SAP for services and events using Application Explorer. See Chapter 2, "Configuring OracleAS Adapter for SAP" for more information.

> **See Also:** *Oracle BPEL Process Manager Developer's Guide*

Adapter integration with Oracle BPEL Process Manager is a two-step process:

1. **Design Time:** OracleAS Adapter for SAP is configured in Application Explorer for services and events, as described in Chapter 2, "Configuring OracleAS Adapter for SAP". Integration logic is modeled using JDeveloper.
2. **Runtime:** After you deploy the BPEL process you designed in JDeveloper, you can test your service configuration or see newly received events in the BPEL Console.

SAP Service Integration

This example demonstrates SAP service integration. It describes design-time, followed by runtime configuration.

Design-Time Configuration

Before you design a process for SAP service integration, you must generate its respective WSDL file using Application Explorer.

Generating WSDL for Request/Response Service

Perform the following steps:

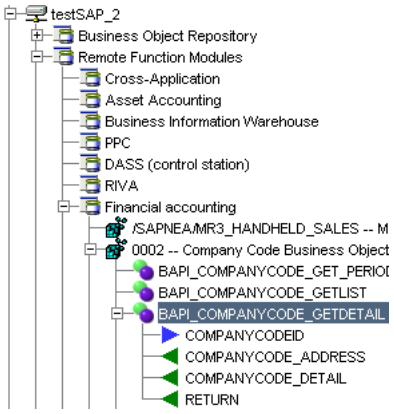
1. Start **Application Explorer** and connect to a defined SAP target (a J2CA configuration).

   See "Defining a Target to SAP" on page 2-10 for more information on defining a target and connecting to SAP.

2. Expand the SAP target to which you are connected.

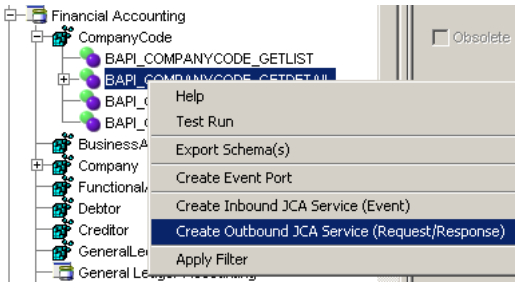
3. Expand **Remote Function Modules**, **Financial Accounting**, **0002 -- Company Code Business Object**, and then select **BAPI_COMPANYCODE_GETDETAIL**.

   The following image shows a connected and expanded target.



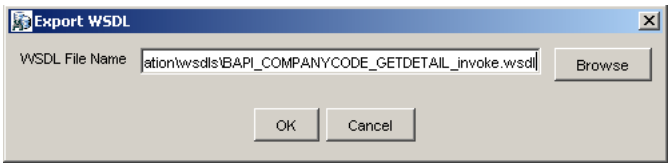
4. Right-click the **BAPI_COMPANYCODE_GETDETAIL** node.

   The following menu is displayed:



5. Click **Create Outbound JCA Service (Request/Response)**.

The Export WSDL dialog box is displayed.



6. Click **OK**.

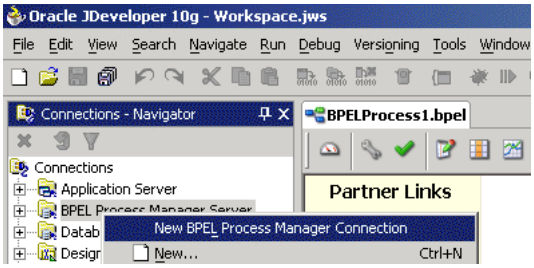
You can now design a BPEL process in JDeveloper.

Creating a BPEL PM Server Connection in JDeveloper

Before you design an outbound BPEL process, you must create a connection to your BPEL Server using JDeveloper. To create a server connection:

1. Open **JDeveloper**.
2. To display the connections, click the **Connections** tab at the bottom of the upper left pane in JDeveloper.

   The following menu is displayed.

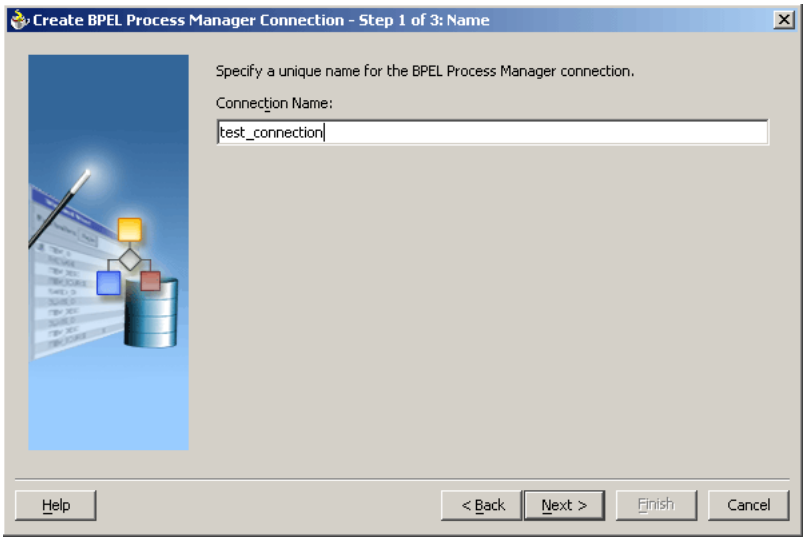


3. Right-click **BPEL Process Manager Server** and select **New BPEL Process Manager Connection**.

   The Create BPEL Process Manager Connection - Welcome dialog box is displayed.

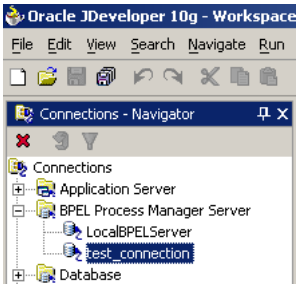
4. Click **Next**.

   The Create BPEL Process Manager Connection dialog box is displayed.



5. Specify a unique name for your BPEL Server connection and click **Next**.
6. Specify a valid host name and port number for the BPEL PM Server you wish to connect to.
7. Click **Finish**.

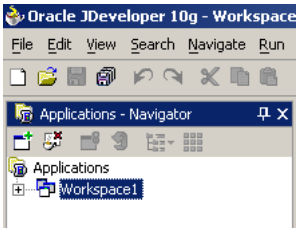
   Your newly created server connection is displayed in the Connections tab under the BPEL Process Manager Server node.



Creating a BPEL Project for a Synchronous BPEL Process

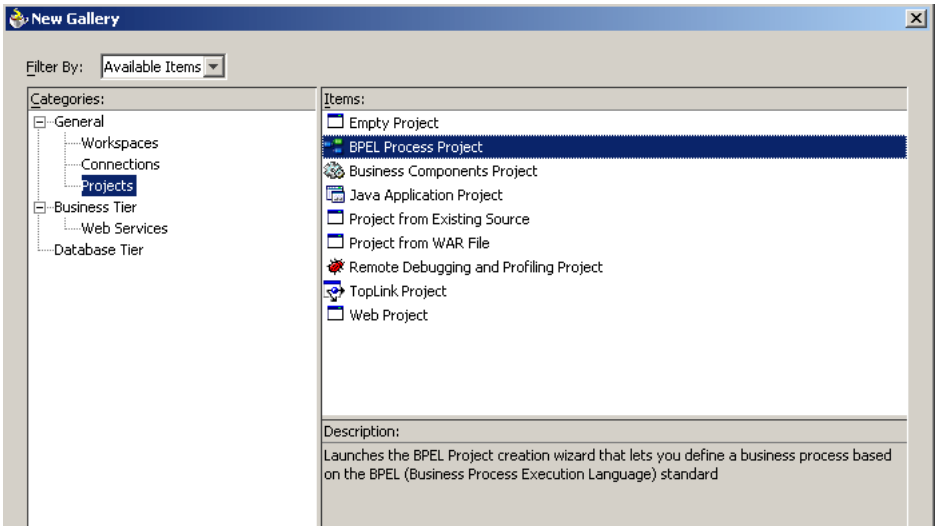
To create a BPEL Project for a synchronous BPEL process:

1. At the bottom of the upper left pane, click the **Applications** tab and select a workspace for your project.



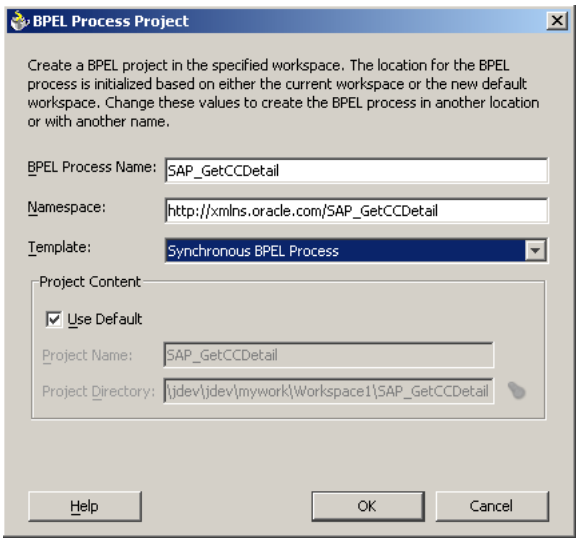
2. Right-click the workspace and select **New Project**.

   The New Gallery window displays a list of available items.



3. From the Items list, select **BPEL Process Project** and click **OK**.

The BPEL Process Project dialog box is displayed.



4. Perform the following steps:
   a. Specify a name for the BPEL process, for example, **SAP_GetCCDetail**.

      The Namespace field is updated automatically.

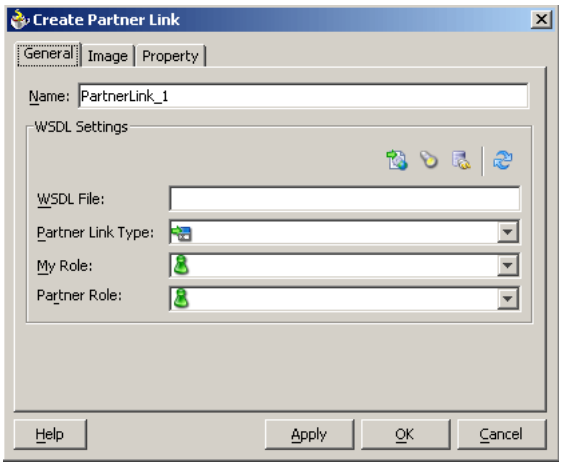
   b. From the Template list, select **Synchronous BPEL Process**.
5. Click **OK**.

Designing the BPEL Process for BAPI_COMPANYCODE_GETDETAIL

To design the BPEL Process:

1. From the Process Activities pane on the right, drag and drop a PartnerLink to the visual editor.

   The Create Partner Link dialog box is displayed.

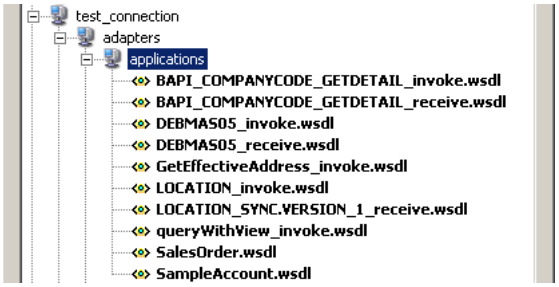


2. Click the **WSIL browser** icon (second icon from the left preceding the **WSDL File** field).

The WSDL Chooser dialog box is displayed.

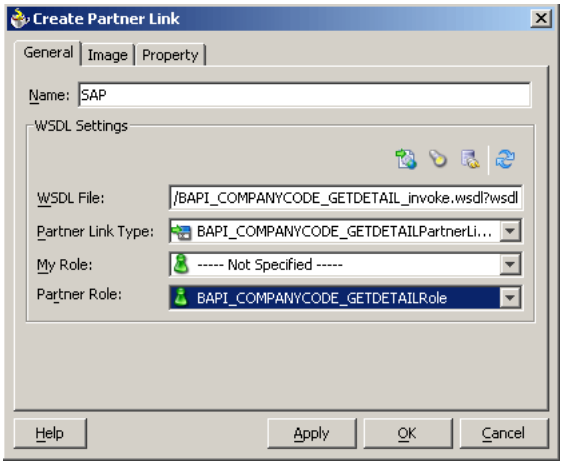
3. Expand your new server connection, then expand **adapters**, and then **applications**.

   The WSDL Chooser dialog box is displayed.



4. Select **BAPI_COMPANYCODE_GETDETAIL_invoke.wsdl** and click **OK**.

   The **WSDL File** field in the Create Partner Link dialog box displays the name and location of the selected WSDL file. The **Partner Link Type** field specifies the PartnerLink defined in the WSDL file.



   Perform the following steps:

   a. Leave the **My Role** field unspecified. The role of the PartnerLink is null, as it will be synchronously invoked from the BPEL process.

   b. From the **Partner Role** list, select the default value **BAPI_COMPANYCODE_GETDETAILRole**. This is the role of the BPEL process.

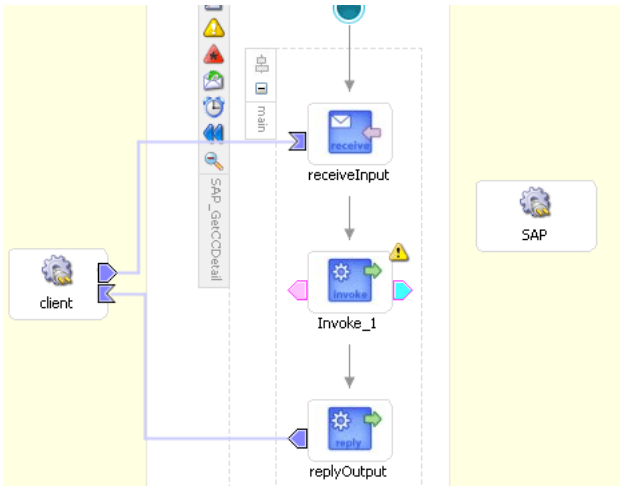
5. Click **OK**.

   The new PartnerLink appears in the visual editor.

6. Select **Save** from the File menu.

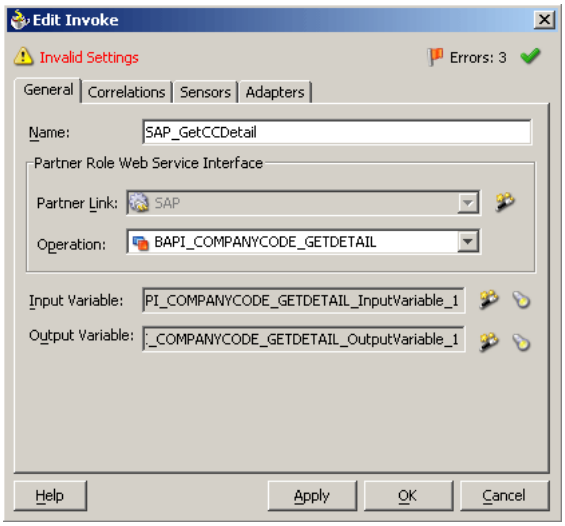
7. From the **Process Activities** pane on the right, drag an **Invoke** activity to the visual editor and place it between the Receive activity (`receiveInput`) and the Reply activity (`replyOutput`).

   The Invoke process activity is shown in the diagram view.



8. Drag the **blue arrow** from Invoke_1 and connect it to the **SAP** PartnerLink.

   The Edit Invoke dialog box is displayed.



   Perform the following steps:

   a. In the **Name** field, enter Get_CCDetail.

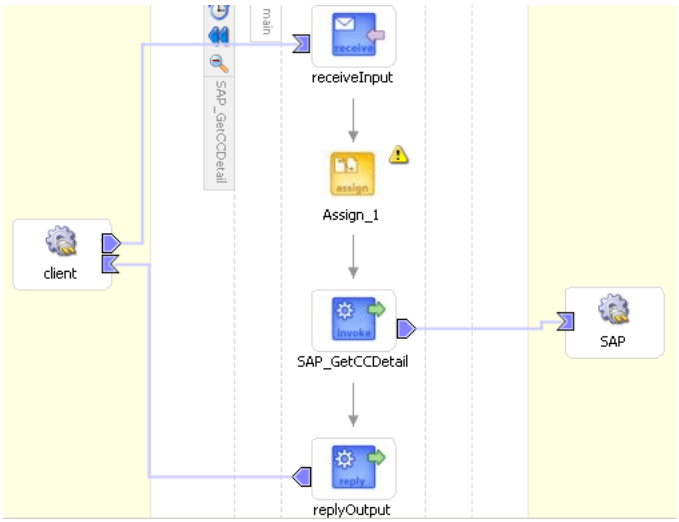
   b. Click the first icon to the right of the **Input Variable** field, then click **OK** in the Create Variable window that is displayed.

   c. Repeat the previous step to create a default variable for Output Variable.

9. Click **OK**.

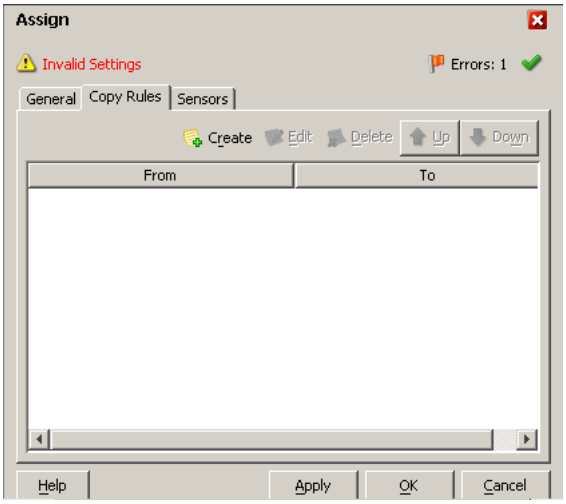
10. Drag an Assign process activity and drop it between `receiveInput` and `Get_CCDetail`.

The following image shows the new Assign activity in JDeveloper visual editor.



11. Double-click the **Assign** activity icon.

    The Assign dialog box is displayed.



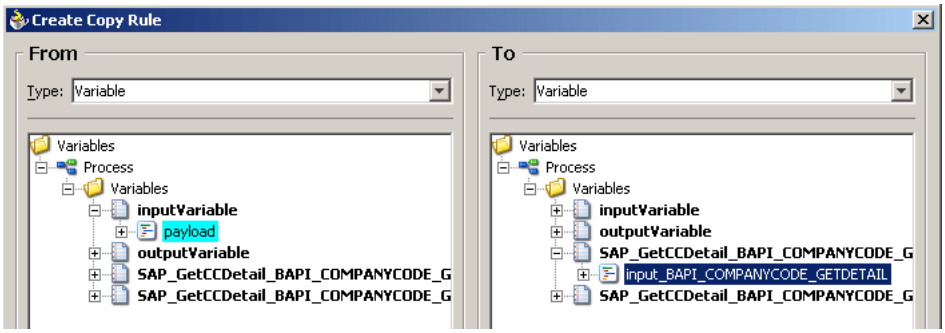
12. In the Copy Rules tab, click **Create**.

    The Create Copy Rule dialog box is displayed.

    a. In the **From** pane, expand **Variables**, then **inputVariable**, and then highlight **payload**.

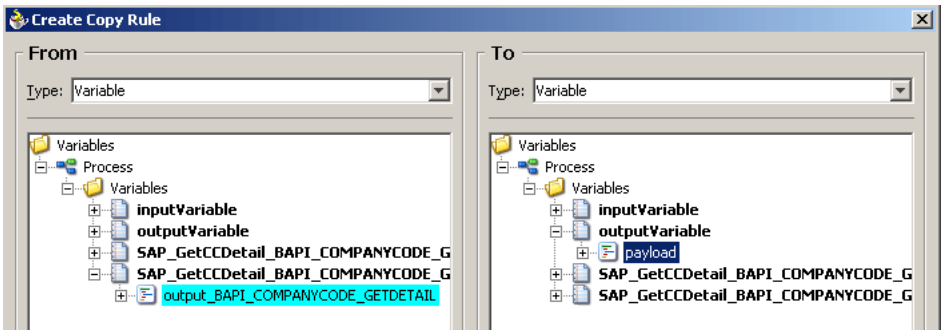
    b. In the **To** pane, expand **Variables**, then **GetCompanyDetail_BAPI_COMPANYCODE_GETDETAIL_InputVariable**, and then highlight **input_BAPI_COMPANYCODE_GETDETAIL**.

Your Create Copy Rule dialog box should look as follows:



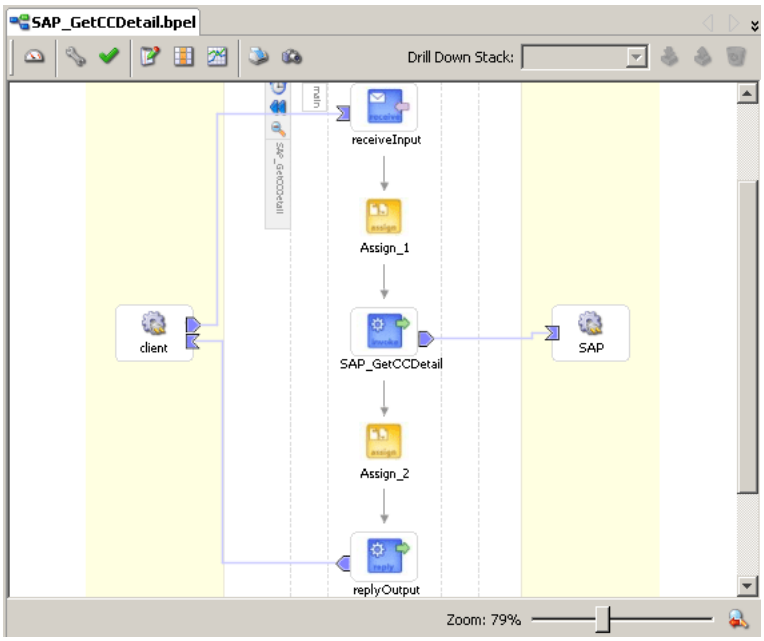
13. To close the Create Copy Rule dialog box and the Assign dialog box, click **OK**.
14. From the **Process Activities** pane on the right, drag another **Assign** activity to the visual editor and place it between the Invoke activity (`SAP_GetCCDetail`) and the Reply activity (`replyOutput`).
15. Double-click the **Assign** activity icon and click **Create**.
16. Map **Get_CCDetail_BAPI_COMPANYCODE_GETDETAIL_OutputVariable**, **output_BAPI_COPMANYCODE_GETDETAIL** to **outputVariable**, **payload**.

    Verify that you have mapped all variables as follows:



17. Click **OK**, then click **OK** again.
18. Select **Save** from the File menu.

You have completed the design of this BPEL process.



Deploying the BPEL Process for BAPI_COMPANYCODE_GETDETAIL

JDeveloper deploys the outbound BPEL process for BAPI_COMPANYCODE_GETDETAIL directly to Oracle BPEL Console.

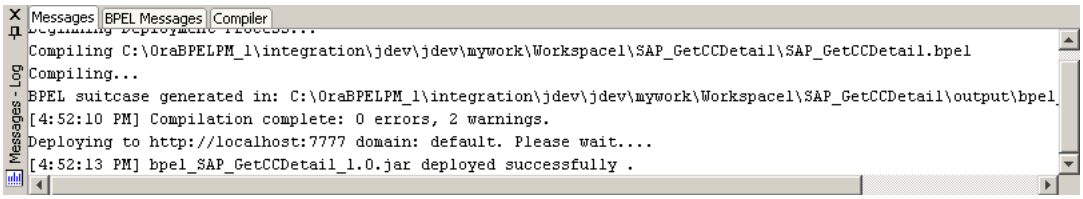
To deploy your BPEL process in JDeveloper:

1. Right-click your process flow in the Applications - Navigator pane.
2. Select **Deploy**, then ***Your BPEL PM Server connection***, and then **Deploy to default domain**.

   The Password Prompt dialog box is displayed.
3. Enter your BPEL PM Server password in the Password Prompt dialog box.

   The deployment process starts automatically after you enter the correct password.
4. Observe the **Messages** log at the bottom of the window.

   The Messages log displays the deployment status. In this example, it shows a successful deployment message for the process.



If deployment was not successful, click the **Compiler** tab to view all error and warning messages generated during the deployment process.

Runtime Configuration

To invoke the BAPI_COMPANYCODE_GETDETAIL process from Oracle BPEL Console:

1. Start the Oracle BPEL Console by entering the following URL in a browser:

   `http://host:port/BPELConsole`

2. Select a domain and provide a valid password.

   The Oracle BPEL Console main page is displayed.

3. Click the **BPEL Processes** tab.

4. Click the SAP process link, **BAPI_COMPANYCODE_GETDETAIL**.

5. Click **Initiate**.

   The Initiate tab enables you to test your BPEL process.

   Perform the following steps:

   a. From the **Initiating a test instance** menu, select **XML Source**.

   b. Enter the following code in the text area provided for XML input:

```
<?xml version="1.0" encoding="UTF-8"?>
<CompanyCode.GetDetail SERVICENAME="CompanyCodeGetDetail" METHODNAME="BAPI_
COMPANYCODE_GETDETAIL" LICENSE="test" CompanyCodeId="1000"/>
```

6. Click **Post XML Message**.

   The response received from the SAP system is displayed in the Initiate window.

SAP Event Integration

This example demonstrates how OracleAS Adapter for SAP integrates with SAP to receive event data. In this example, an SAP event occurs when a customer record is added to an SAP system.

The design-time and runtime procedures are outlined in the following sections.

Design-Time Configuration

You must create a separate channel for every inbound J2CA service and select that channel when you generate WSDL for inbound interaction using Application Explorer.

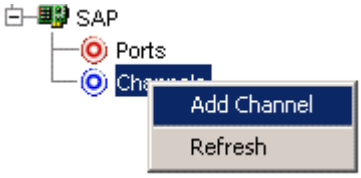
> **Note:** If two or more events share the same channel, event messages may not be delivered to the right BPEL process.

Creating a Channel

To create a channel:

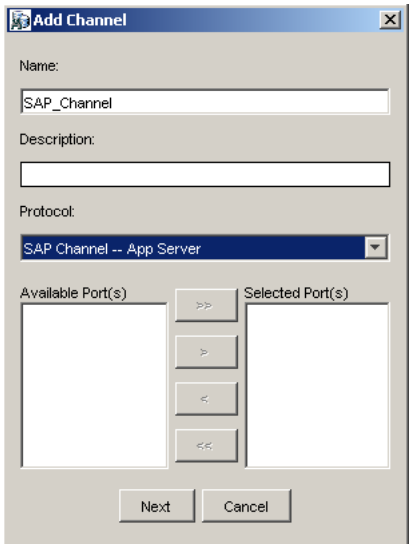
1. Start **Application Explorer** and connect to a J2CA configuration.
2. In the left pane, expand the **Events** node.
3. Expand the **SAP** node.

   The Ports and Channels nodes appear in the left pane.



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

   The Add Channel dialog box is displayed.

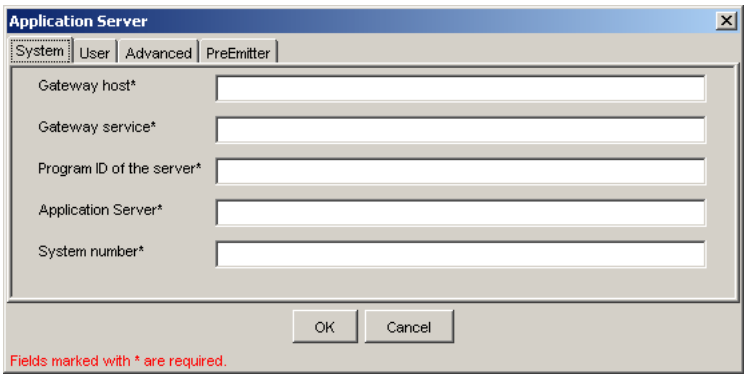


   Perform the following steps:

   a. Enter a name for the channel, for example, `SAP_Channel`.
   b. Enter a brief description (optional).
   c. From the **Protocol** list, select **SAP Channel--App Server**.

5. Click **Next**.

The Application Server dialog box is displayed.



The following tabs are available:

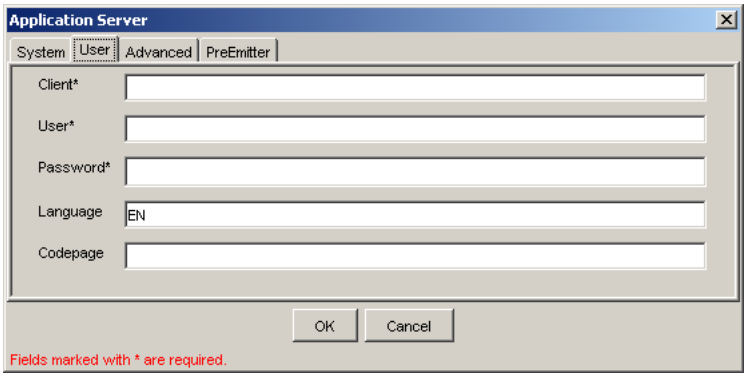
- System (Required)
- User (Required)
- Advanced
- PreEmitter

6. In the **System** tab, enter the appropriate information for your SAP channel, based on the information in the following table.

Table 5–1 System Tab Parameters

Target Parameter	Description
Gateway host	A host name for the SAP Gateway.
Gateway service	A service for the SAP Gateway.
Program ID of the server	An SAP program ID you want to use for this channel.
Application Server	A host name for the application server.
System number	A system number for SAP.

7. Click the **User** tab.



8. For the **User** tab, enter the appropriate information for your SAP channel, based on the information in the following table.

Table 5–2 User Tab Parameters

Target Parameter	Description
Client	The client number defined for the SAP application for client communications.
User	A valid user ID for the SAP application.
Password	A valid password for the SAP application.
Language	A language key. EN (English) is the default.
Code page	A character code page value.

9. For the **Advanced** tab (optional), enter the appropriate information for your SAP channel, based on the information in the following table.

Table 5–3 Advanced Tab Parameters

Target Parameter	Description
IDoc Format	Select an IDoc enter from the list.
User Defined Function Modules	Enter the path to the user-defined function module you created.
SAP trace	Select this check box if you want to enable SAP traces for troubleshooting purposes.
Unicode	Select this check box if you are expecting your response in Unicode format.
Processing Mode	Select the type of synchronous processing from the list.

10. Click **OK**.

    The channel appears under the channels node in the left pane. An X over the icon indicates that the channel is currently disconnected.

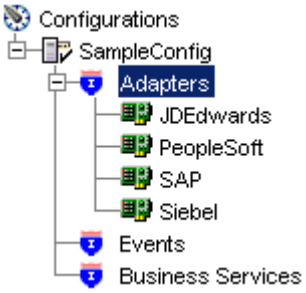
> **Note:** Do not start the channel, as it is managed by BPEL PM Server. If you start the channel for testing and debugging purposes, stop it before runtime.

Generating WSDL for Event Notification

After you create a channel and verify that it is not started, you must generate WSDL for the event using Application Explorer.

1. Start Application Explorer.
2. Expand the **Adapters** node.

A list of all adapters is displayed.



Perform the following steps:

a. Expand the **SAP** node.

A list of your available targets is displayed.

b. Click a target name under the **SAP** node, for example, **SAPTarget**..

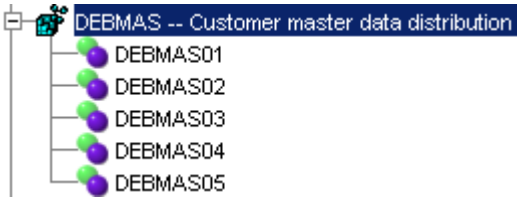
The Connection dialog box displays the saved parameters.

3. Verify your connection parameters.
4. Provide the required password.
5. Right-click the target name and select **Connect**.

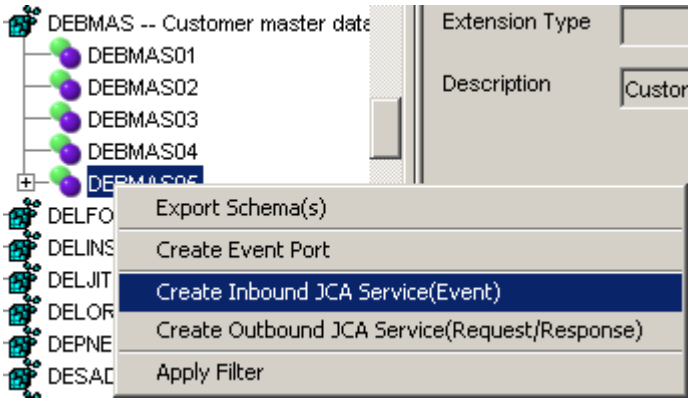
The x icon disappears, indicating that the node is connected.

6. Expand the **ALE(IDOCS)** node and select **DEBMAS**.

The DEBMAS list is displayed.

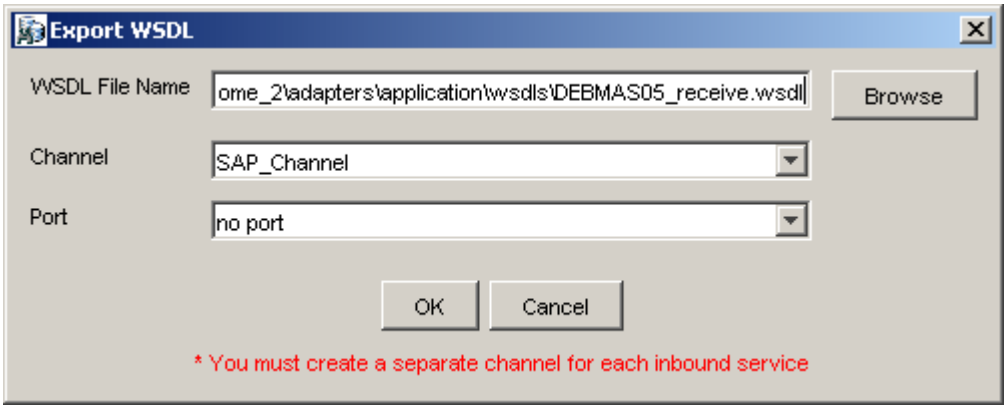


7. Right-click **DEBMAS05** from the **DEBMAS** list.



8. Select **Create Inbound JCA Service (Event)**.

   The Export WSDL dialog box is displayed.



   Perform the following steps:

   a. In the **WSDL File Name** field, specify a name and location of the WSDL file.

   b. In the **Channel** field, select the channel you created for this inbound service.

      Important: You must create a separate channel for every event. Verify that the channel is stopped before runtime.

   c. **If you are using the optional port feature**, you must select a port from the **Port** list. See "J2CA Filtering Port Option for Integration with BPEL Process Manager" on page 2-24 for more information.

      If you are not using event ports for schema validation, skip this step. In this case, the default value of no port is selected automatically.

9. Click **OK**.

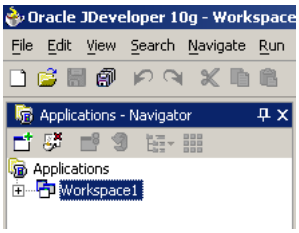
Creating a BPEL PM Server Connection in JDeveloper

Before you design a BPEL process using the WSDL you generated in Application Explorer, you must create a connection to your BPEL Server using JDeveloper. See "Creating a BPEL PM Server Connection in JDeveloper" on page 5-3 for details on how to create the server connection.

Designing the BPEL Process for the SAP_DEBMAS05 Event

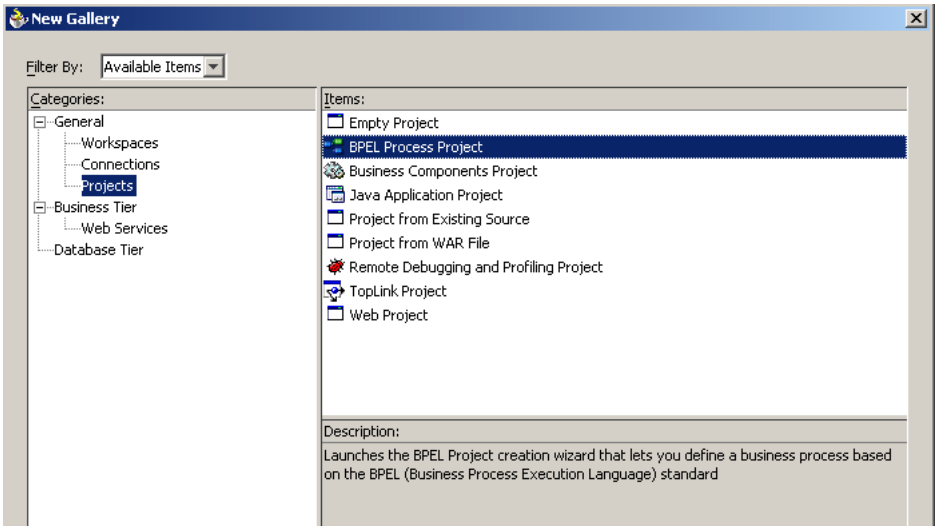
To design a BPEL process for inbound interaction:

1. Click the **Applications** tab and select a workspace for your project.



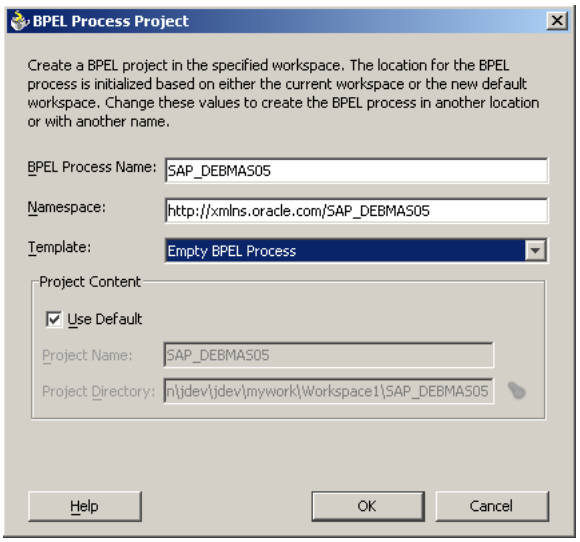
2. Right-click the workspace and select **New Project**.

   The New Gallery window is displayed.



3. From the Items list, select **BPEL Process Project** and click **OK**.

   The BPEL Process Project dialog box is displayed.

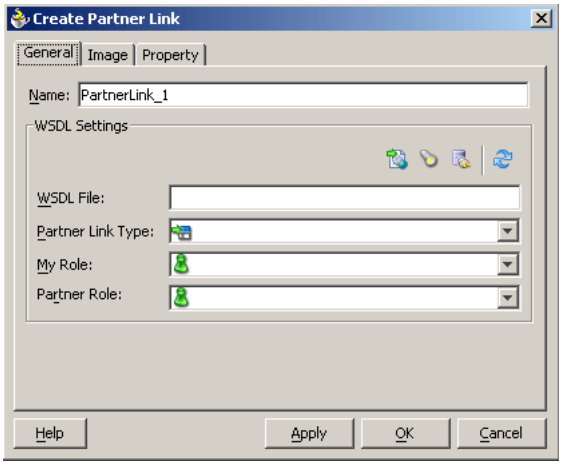


4. Perform the following steps:
   a. Specify a name for the process.

      The Namespace field is updated automatically.
   b. From the Template list, select **Empty BPEL Process**.
   c. Click **OK**.

   An empty BPEL process project template is created.
5. From the Process Activities pane on the right, drag and drop a **PartnerLink** to the visual editor.

   The Create Partner Link dialog box is displayed.

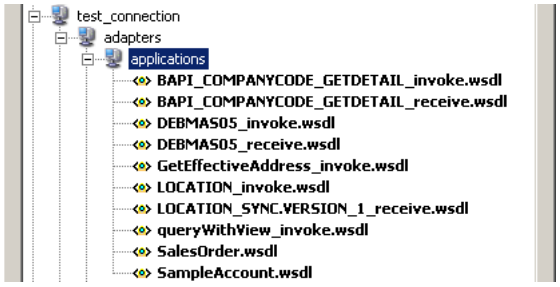


6. Click the **WSIL browser** icon (second icon from the left preceding the **WSDL File** field).

   The WSDL Chooser dialog box is displayed.

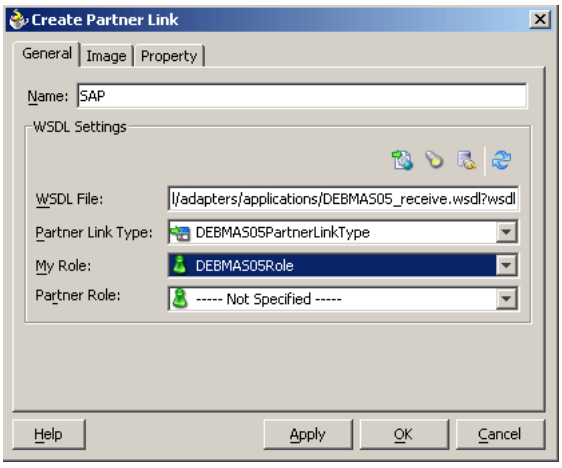
7. Expand your BPEL Server connection, then expand **adapters**, and then **applications**.

   The WSDL Chooser dialog box is displayed.



8. Select **DEBMAS05_receive.wsdl** and click **OK**.

The Create Partner Link dialog box is displayed.



The **WSDL File** field displays the name and location of the selected WSDL file. The **Partner Link Type** field specifies the PartnerLink defined in the WSDL file.

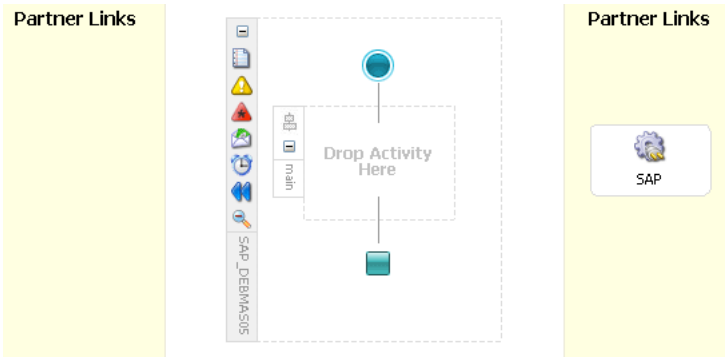
Perform the following steps:

a. From the **My Role** list, select the default value **DEBMAS05Role**.

b. Leave the **Partner Role** field unspecified.

9. Click **Apply**, and then **OK**.

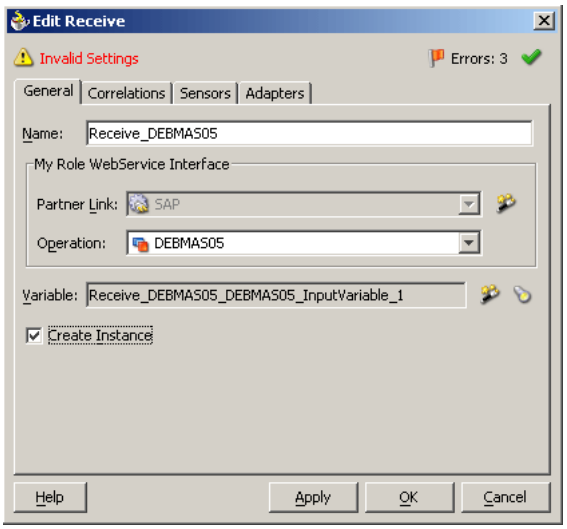
The new SAP PartnerLink appears in the visual editor.



10. From the **Process Activities** pane on the right, drag a **Receive** activity to the visual editor and place it in the designated placeholder labeled **Drop Activity Here**.

11. Connect the Receive activity to the SAP PartnerLink.

The Edit Receive dialog box is displayed.



Perform the following steps:

a. Specify a name for the Receive Activity, for example, **Receive_DEBMAS05**.

b. Click the first icon to the right of the **Variable** field, then click **OK** in the Create Variable dialog box that is displayed.

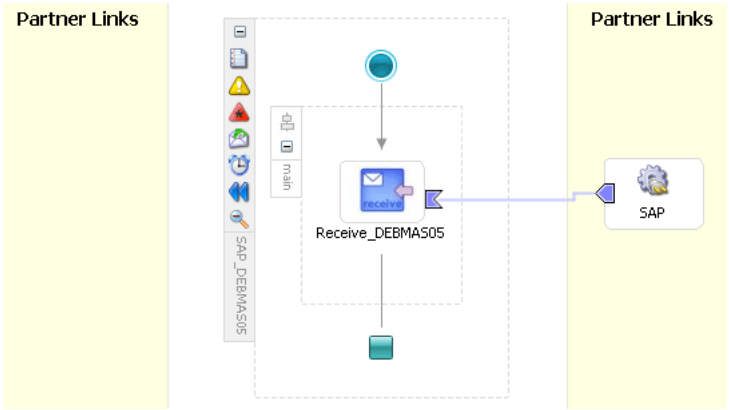
c. Verify that the **Create Instance** check box is selected.

12. Click **Apply**.

    The Receive dialog box should no longer display any warnings or errors.

13. Click **OK**.

    Your completed process looks as follows.



14. Select **Save** from the **File** menu.

Deploying the BPEL Process for the SAP_DEBMAS05 Inbound Service

1. Right-click your process flow in the Applications - Navigator pane.
2. Select **Deploy**, then ***Your BPEL PM Server connection***, and then **Deploy to default domain**.
3. When prompted, enter your BPEL PM server password and click **OK**.

The deployment process starts automatically after you enter the correct password.

Runtime Configuration

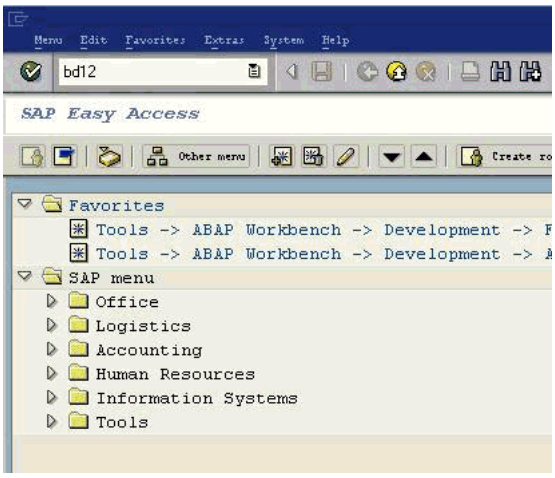
Events are generated as a result of activity in an application system. For example, SAP may generate an event as customer information is updated in the system. For more information on events, see "Configuring an Event Adapter" on page 2-21

Triggering an Event in SAP

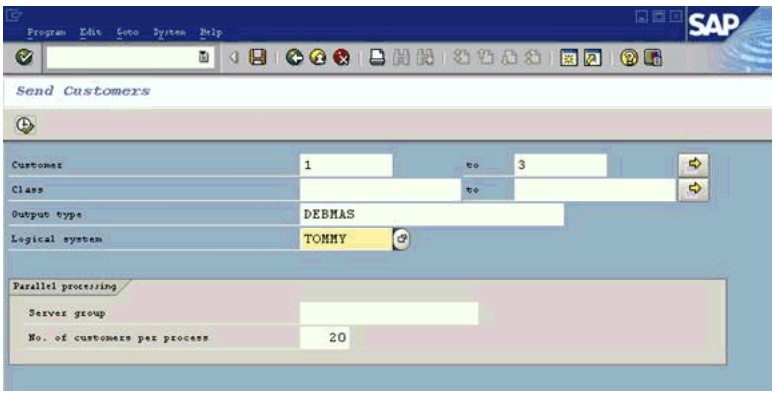
The following topics describe how to trigger an event in SAP and verify event integration using OracleAS Adapter for SAP.

To trigger an event in SAP:

1. Start the SAP Workbench and log in to the SAP system.



2. Run the **bd12** transaction.



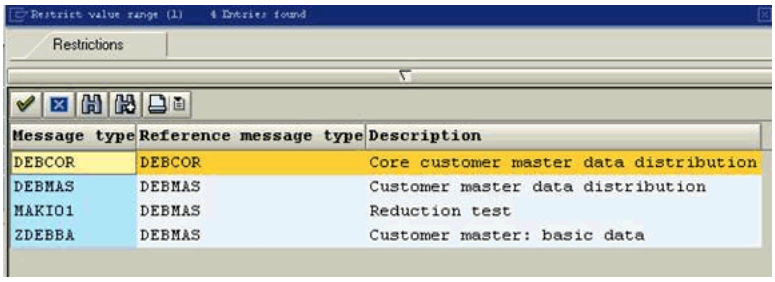
Enter the following information in the Send Customers window:

a. In the Customer field, enter a customer number with a range from 1 to 3.

b. In the Output type field, enter **DEBMAS**.

c. In the Logical system field, specify the logical system you are using with SAP.

3. Click the **check mark** icon in the upper left-hand corner.



4. Ensure **DEBMAS** appears in the Message type column.
5. Click the **Execute** button.

   Customer master data is sent to the logical system specified. If a channel in Application Explorer defined the Program ID with the same value, the channel receives this customer master data from SAP.

Verifying the Results

To verify your results:

1. Log in to Oracle BPEL Console at

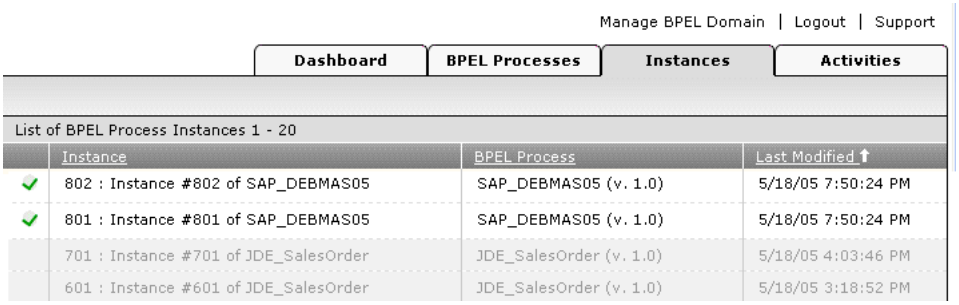
   `http://host:port/BPELConsole`

2. Enter the password for your BPEL domain.

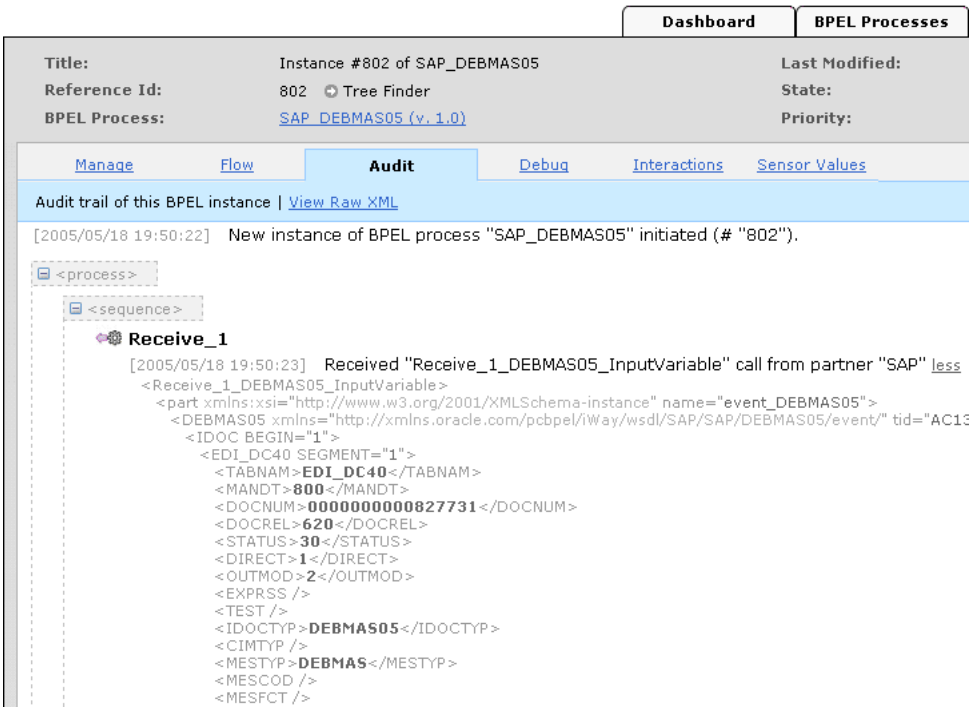
   The default password is `bpel`.

3. Click the **Instances** tab.

   Recently received runtime events are displayed in the Instances tab.



4. Click the **SAP_DEBMAS05** instance, then click **Audit** to see the received SAP_DEBMAS05 event message.



6

InterConnect Integration Examples

This chapter contains the following examples:

- Adapter Configuration in Application Explorer
- SAP Service Integration
- SAP Event Integration

The scenarios shown in this chapter require the following prerequisites.

Prerequisites

The following are installation and configuration requirements:

- OracleAS Adapter for SAP must be installed on Oracle Application Server.
- SAP must be configured for outbound processing. See Appendix A, "Configuring SAP for Inbound and Outbound Processing" for more information.
- OracleAS Database adapter must be deployed and properly configured.
- OracleAS Integration InterConnect Adapter Plugin for EIS must be installed and running.

> **See Also:** *Oracle Application Server Adapters Installation Guide*

The examples in this chapter present the configuration steps necessary for demonstrating service and event integration with SAP. Prior to using this material, you must be familiar with the following:

- How to configure OracleAS Adapter for SAP for services and events. For more information, see Chapter 2, "Configuring OracleAS Adapter for SAP".
- How to configure OracleAS Integration InterConnect iStudio (iStudio) for service and event interactions. All required steps are shown in the examples.

> **See Also:** *Oracle Application Server Integration InterConnect User's Guide*

Overview of InterConnect Integration

InterConnect provides a comprehensive application integration framework. OracleAS Adapter for SAP used in conjunction with InterConnect 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 OracleAS InterConnect is a two-step process:

1. **Design Time:** OracleAS Adapter for SAP is configured in Application Explorer for services and events, as described in Chapter 2, "Configuring OracleAS Adapter for SAP". Integration logic is modeled in iStudio. Metadata are stored in repositories.
2. **Runtime:** The underlying platform treats this metadata as runtime instructions to enable the communication between participating applications.

Adapter Configuration in Application Explorer

The following example describes how to establish a connection to your SAP system, and how to create a port and a channel for DEBMAS05.

Connecting to SAP

1. Start Application Explorer.
2. Expand the **Adapters** node.

   Perform the following steps:

   a. Expand the **SAP** node.

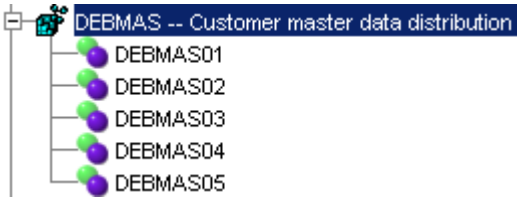
   b. Click the target name, for example, SAPTarget, under the **SAP** node.

   The Connection dialog box opens, populated with values you have previously entered.

3. Verify your connection parameters.
4. Provide the required password.
5. Right-click the target name and select **Connect**.

   The X over the icon disappears, indicating that the node is connected.

6. Expand the **ALE(IDOCS)** node and select **DEBMAS**.

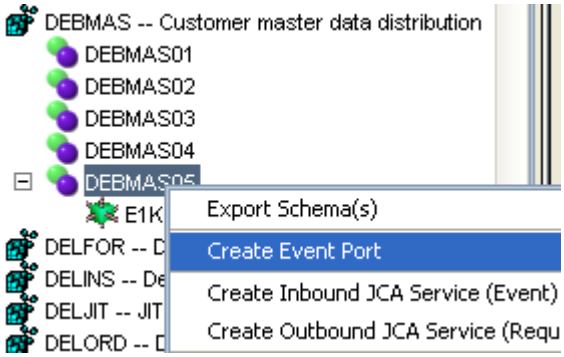


7. Expand the **DEBMAS** list and select **DEBMAS05**.

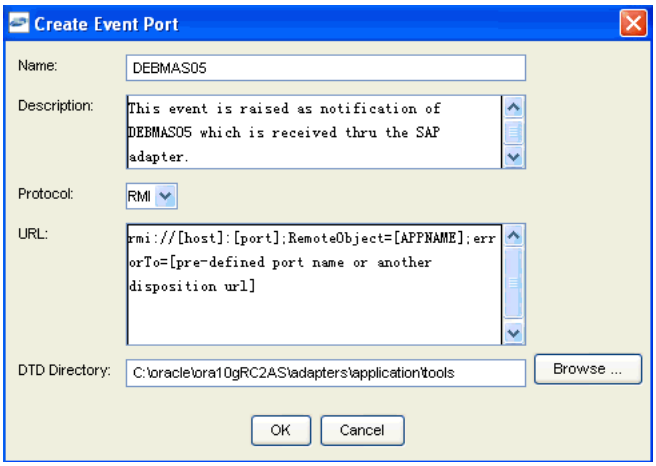
Creating an Event Port for DEBMAS05

To create an event port for DEBMAS05:

1. Right-click the **DEBMAS05** node and select **Create Event Port**.



   The Create Event Port dialog box is displayed.



   Perform the following steps:

   a. Type a name for the event port and provide a brief description.

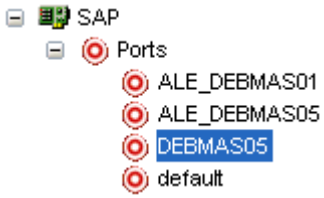
   b. From the list, select the required disposition, for example, RMI.

   c. Type the disposition URL.

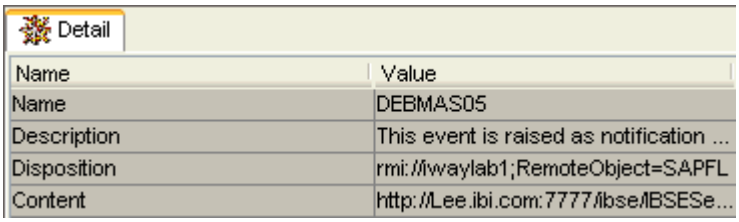
   d. Type (or browse to) the path containing the DTD directory.

2. Click **OK**.

   The port appears under the ports node in the left pane.



In the right pane, a table appears that summarizes the information associated with the event port you created.



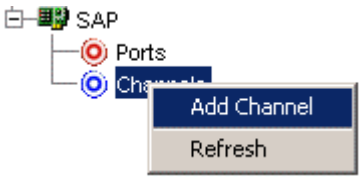
You can now associate the event port with a channel.

Creating a Channel

To create a channel:

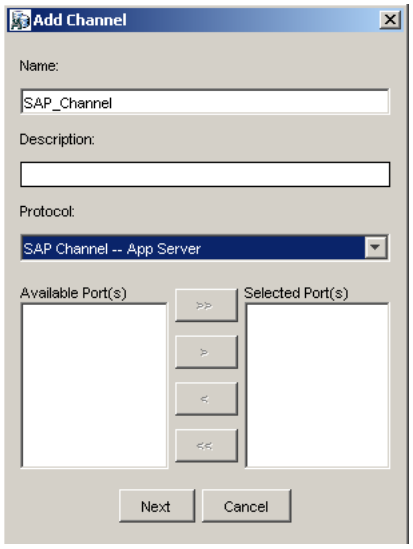
1. In the left pane, click the **Events** node.
2. Expand the **SAP** node.

   The ports and channels nodes appear in the left pane.



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

   The Add Channel dialog box opens.



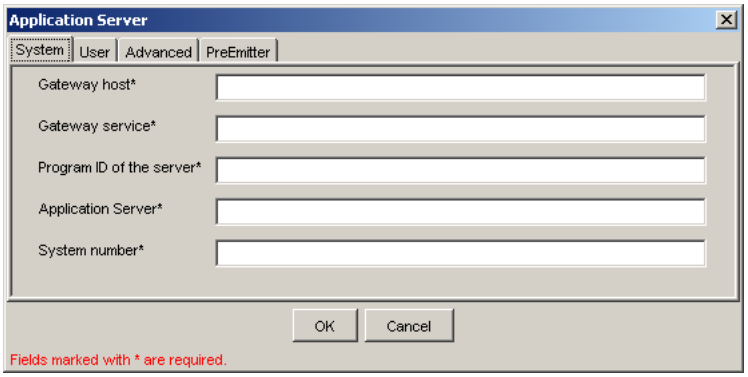
   Perform the following steps:

   a. Type a name for the channel, for example, `TEST_CHANNEL`.
   b. Type a brief description.
   c. From the **Protocol** list, select **SAP Channel**.
   d. Select an event port from the list of available ports. To select more than one, hold down the **Ctrl** key and click the ports.

e. To transfer the ports to the list of selected ports, click the **double right (>>)** arrow button.

4. Click **Next**.

   The Application Server dialog box opens.



   The following tabs are available:

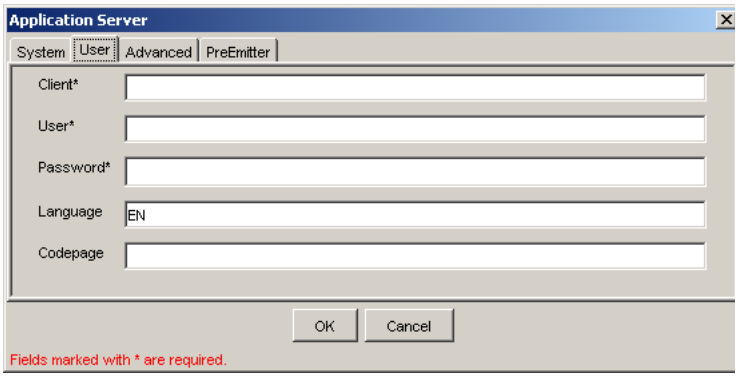
   - System (Required)
   - User (Required)
   - Advanced

5. For the **System** tab, type the appropriate information for your SAP channel, based on the information in the following table.

Table 6–1 System Tab Parameters

Target Parameter	Description
Gateway host	A host name for the SAP Gateway.
Gateway service	A service for the SAP Gateway.
Program ID of the server	An SAP program ID you want to use for this channel.
Application Server	A host name for the application server.
System number	A system number for SAP.

6. Click the **User** tab.



7. For the **User** tab, type the appropriate information for your SAP channel, based on the information in the following table.

Table 6–2 User Tab Parameters

Target Parameter	Description
Client	The client number defined for the SAP application for client communications.
User	A valid user ID for the SAP application.
Password	A valid password for the SAP application.
Language	A language key. EN (English) is the default.
Code page	A character code page value.

8. For the **Advanced** tab (optional), type the appropriate information for your SAP channel, based on the information in the following table.

Table 6–3 Advanced Tab Parameters

Target Parameter	Description
IDoc Format	Select an IDoc type from the list.
User Defined Function Modules	Enter the path to the user-defined function module you created.
SAP trace	Select this check box if you want to enable SAP traces for troubleshooting purposes.
Unicode	Select this check box if you are expecting your response in Unicode format.
Synchronous Processing	Select the type of synchronous processing from the list.

9. Click **OK**.

   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.

10. Right-click the channel node and select **Start**.

    The channel you created becomes active.

    The x over the icon in the left pane disappears.

11. To stop the channel, right-click the connected channel node and select **Stop**.

SAP Service Integration

This topic illustrates SAP service integration. It describes the design-time and runtime configuration steps.

Design-Time Configuration

The following procedures describe how to start the InterConnect repository, create a common view, and define invoked and implemented procedures. Then, it describes how to export PL/SQL code from iStudio.

Starting the Repository

To start the InterConnect repository, double-click the `start.bat` file located in the following directory:

InterConnect_home`\repository\start.bat`

Where `InterConnect_home` is the directory where InterConnect is installed, for example, `C:\OraHome_1\integration\interconnect\`.

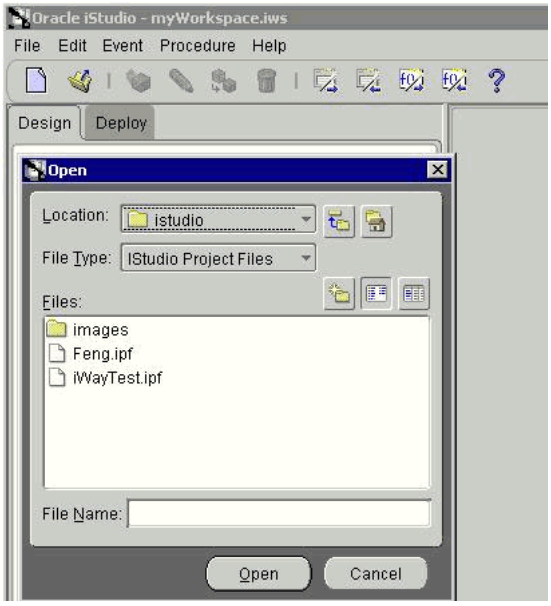
Creating a Common View

To create a Common View:

1. Start iStudio by double-clicking the `start.bat` file located in the following directory:

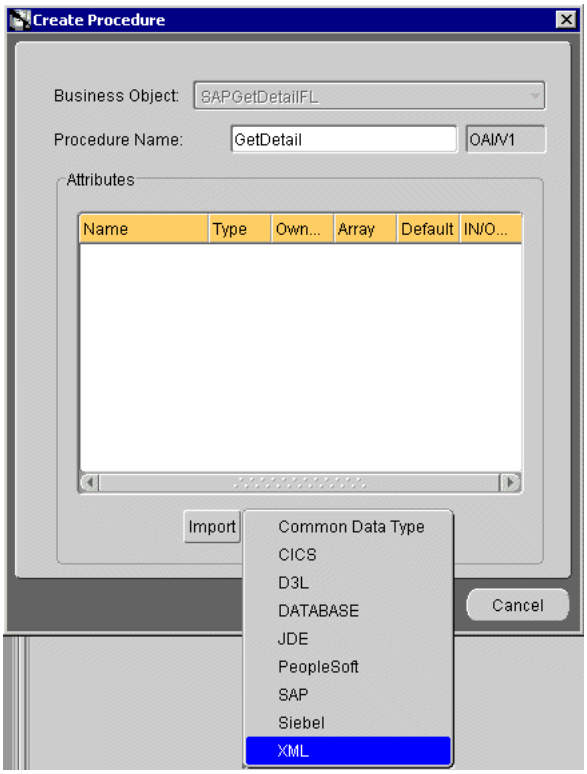
   InterConnect_home`\istudio\iStudio.bat`

   iStudio starts.



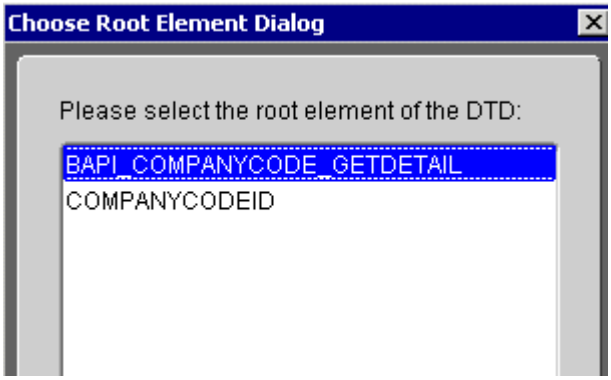
2. Open a project.
3. Open **Common Views** and **Business Objects**.

4. Create a Business Object called **SAPGetDetailFL**.



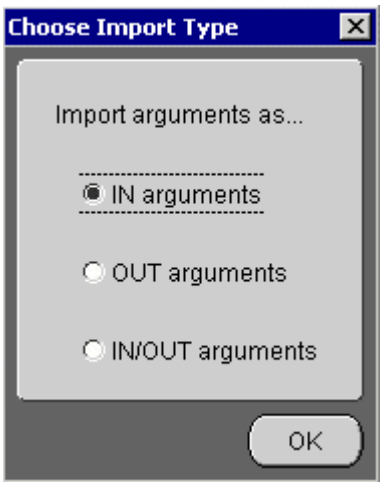
5. Create a new procedure under `SAPGetDetailFL` and type `GetDetail` as the procedure name.
6. Open the DTD generated from Application Explorer and load it.

   The Choose Root Element dialog box is displayed.

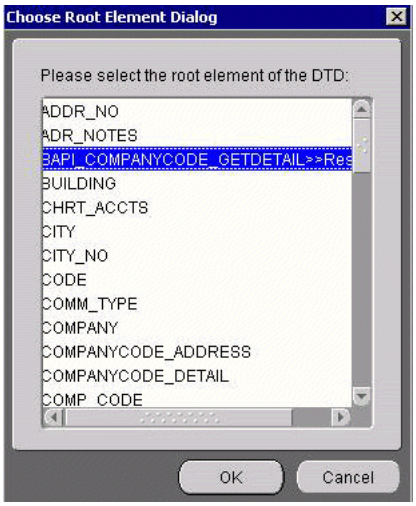


7. Select the root element, `BAPI_COMPANYCODE_GETDETAIL`, for this example.

8. Click **OK**.

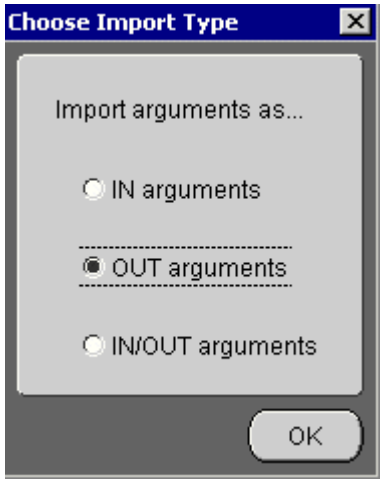


9. Select **IN arguments** as the import type for the request DTD and click **OK**.

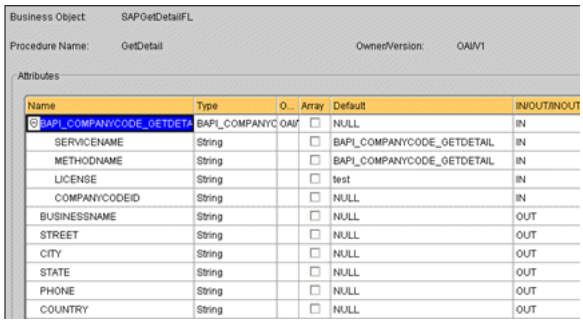


10. Import the response DTD, select the root element, and click **OK**.

    The Choose Import Type dialog box is displayed.



11. Select **OUT arguments** as the import type for the response DTD and click **OK**.



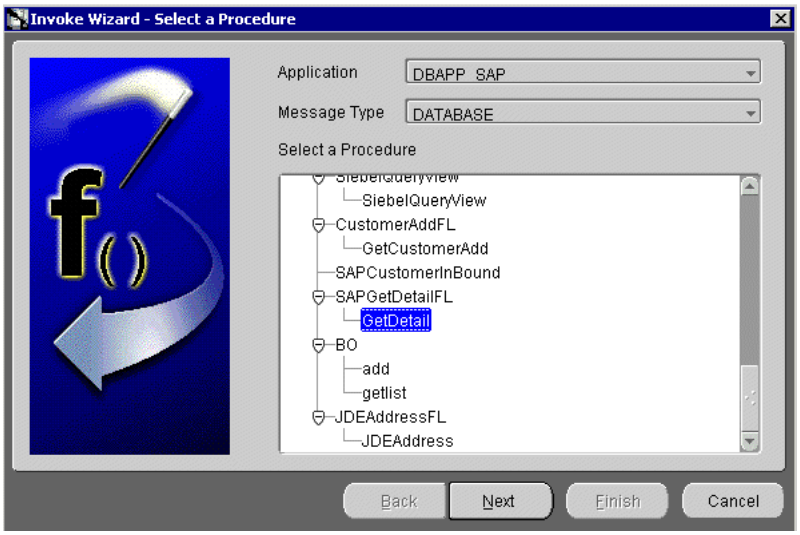
12. Manually enter all the OUT parameters as shown in the previous image.
13. To save the new procedure, click **Save**.

Creating an Invoked Procedure

To create an invoked procedure:

1. Create a new application called **DBAPP_SAP**.
2. Right-click **Invoked Procedures** and select **New**.

   The Invoke Wizard - Select a Procedure window is displayed.

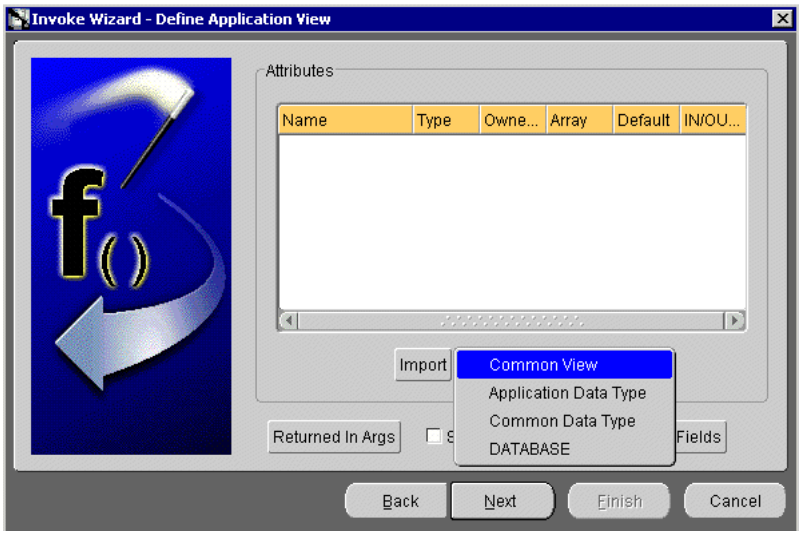


   Perform the following steps:

   a. From the **Message Type** list, select **DATABASE**.
   b. Expand the `SAPGetDetailFL` business object as the event and select **GetDetailFL**.

3. Click **Next**.

The Invoke Wizard - Define Application View window is displayed.

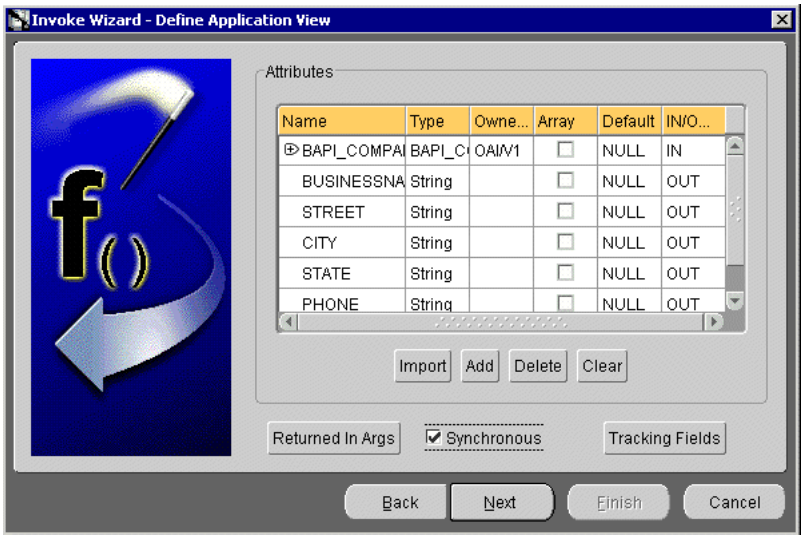


Perform the following steps:

a. Click **Import**.

b. Select **Common View**.

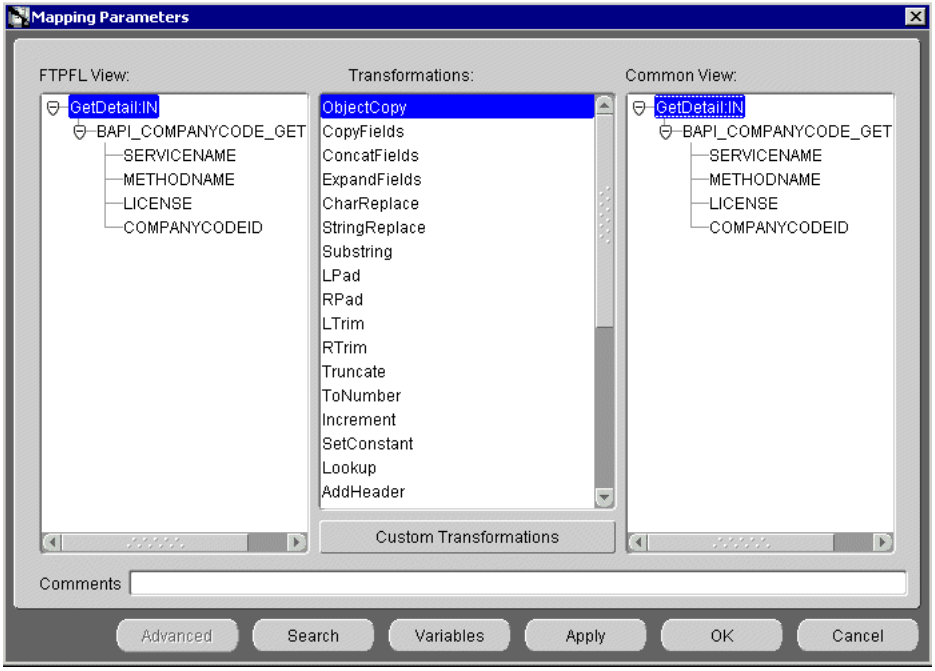
Information appears in the right pane.



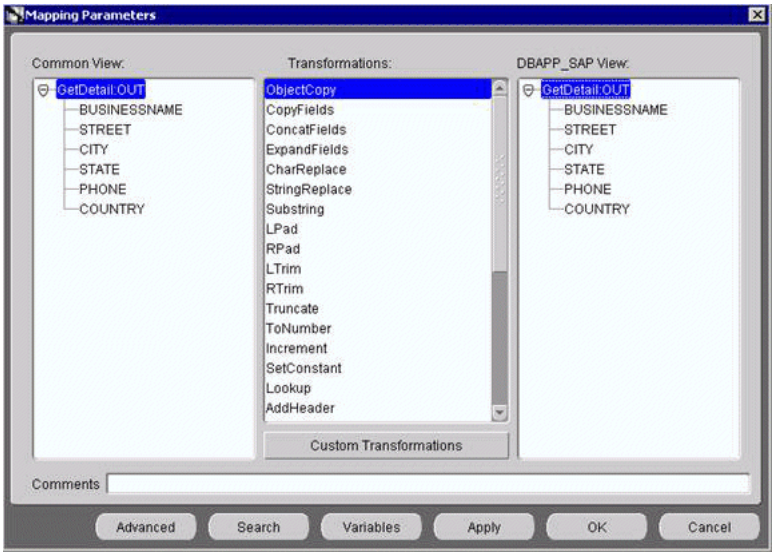
c. Select the **Synchronous** check box, because this is a request and a response.

4. Click **Next**.

5. Click **New** to create a mapping between the Common View and the Application View for the IN parameters.

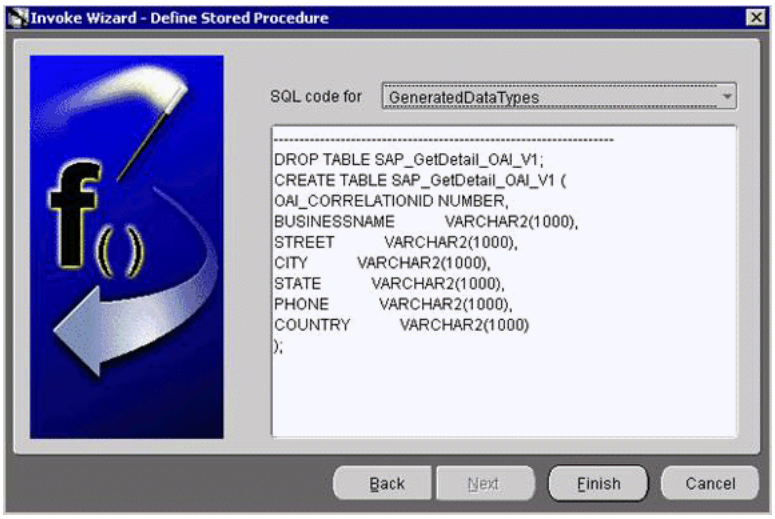


In this example, the Application View and the Common View have the same structure. All the attributes can be mapped by using ObjectCopy Transformation.



6. Click **Apply** and then **OK**.

iStudio generates SQL code.



7. Click **Finish**.

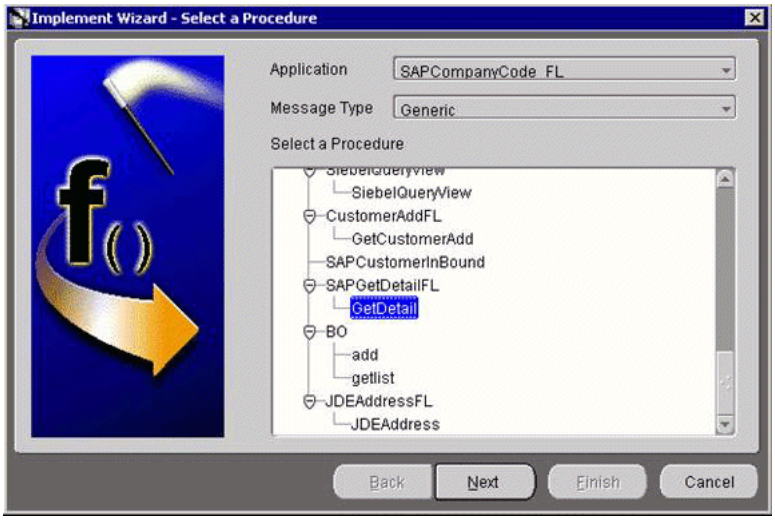
   The application definition for the invoked procedure is now complete.

Defining an Implemented Procedure

To define an implemented procedure:

1. Create a new application called `SAPCompanyCodeFL`.
2. Expand **SAPCompanyCode_FL**.
3. Right-click **Implemented Procedure** and select **New**.

   The Implement Wizard - Select a Procedure window is displayed.



   Perform the following steps:

   a. From the **Message Type** list, select **Generic**.

   b. Expand the `SAPGetDetailFL` business object and select **GetDetail** as the procedure.

4. Click **Next**.

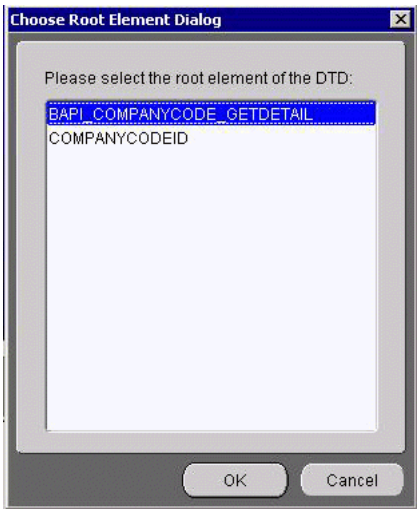
Perform the following steps:

a. Click **Import**.

b. Select **XML**.

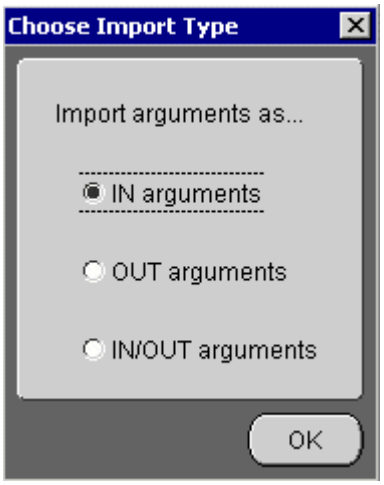
c. Select the request and response DTDs generated by Application Explorer.

5. Import the request and response DTDs into iStudio.



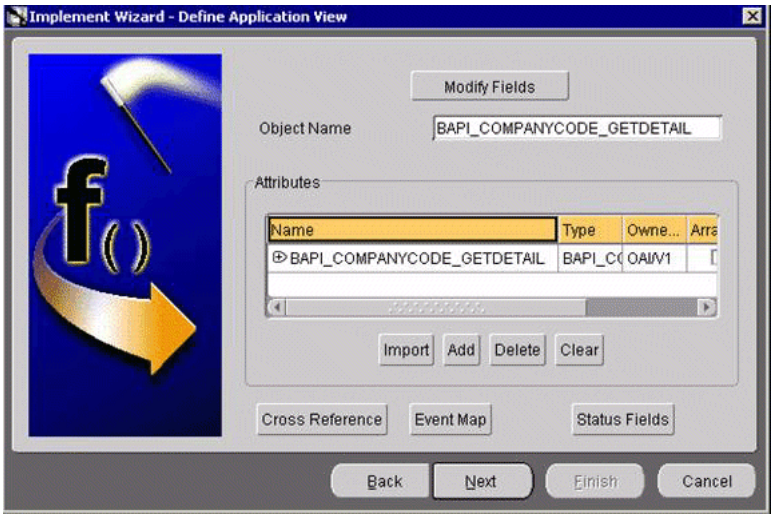
6. Select **BAPI_COMPANYCODE_GETDETAIL** as the root element of the request DTD.

   The Choose Import Type dialog box is displayed.



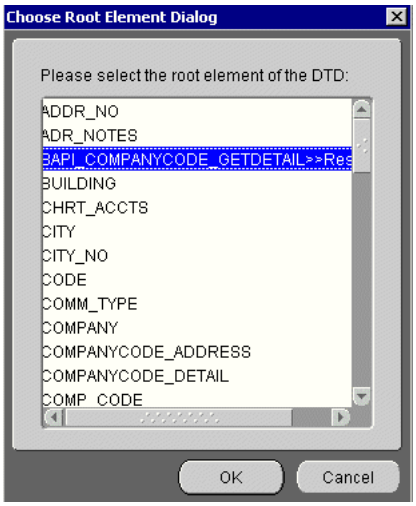
7. Select **IN arguments** as the import type for the request DTD and click **OK**.

The Implement Wizard - Define Application View window is displayed.

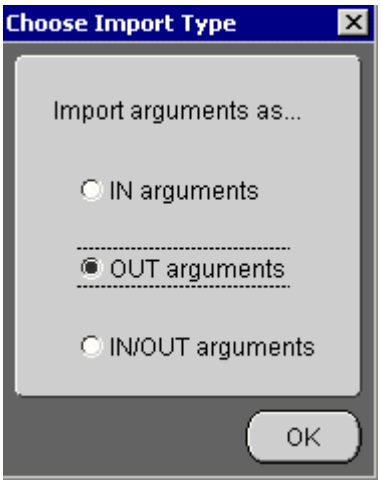


8. Type the root element of the request DTD in the Object Name field, if it is not automatically populated after importing the request DTD and click **Import**.

   The Choose Root Element dialog box is displayed.

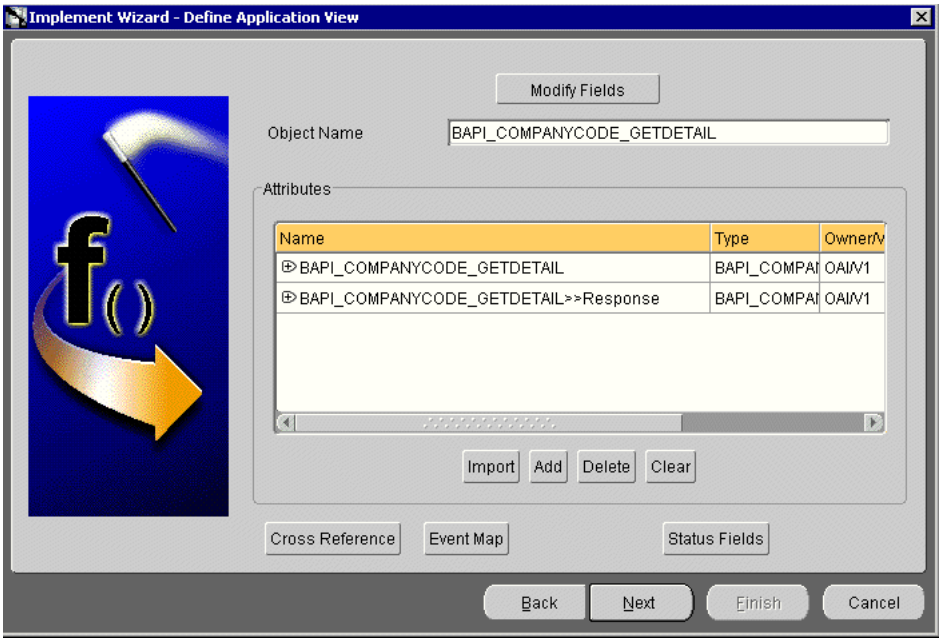


9. Select **BAPI_COMPANYCODE_GETDETAIL** as the root element and click **OK**.

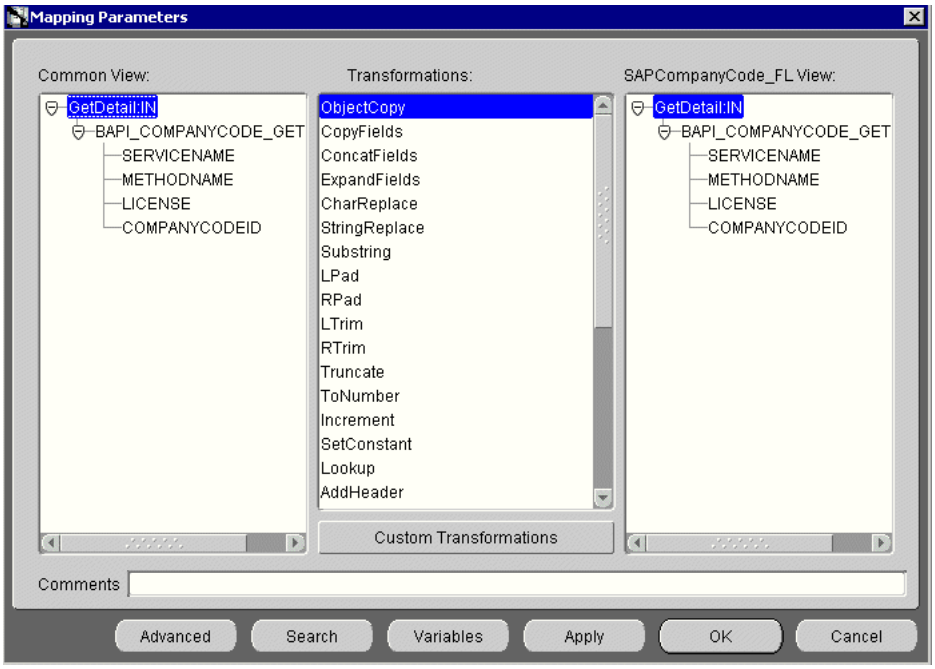


10. Select **OUT arguments** as the import type for the response DTD and click **OK**.

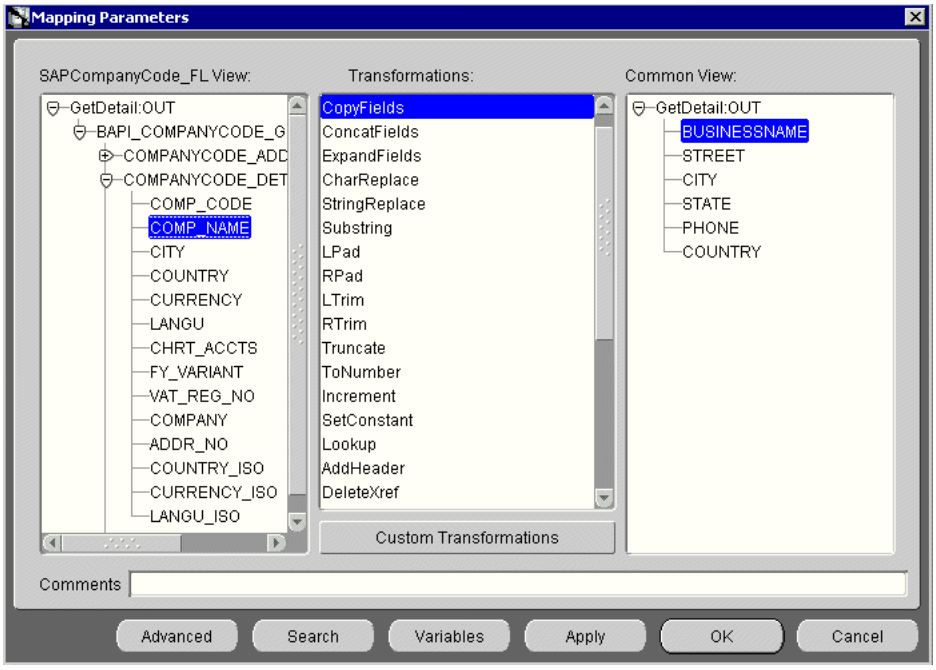
    Both the request and response DTDs are now imported into iStudio.



11. To define a mapping between the Application View and the Common View, click **Next** and then **New**.



Because the Common View IN and the Application View IN have the same structure, ObjectCopy transformation is used for the mapping.



12. Click **Apply** and then **OK**.

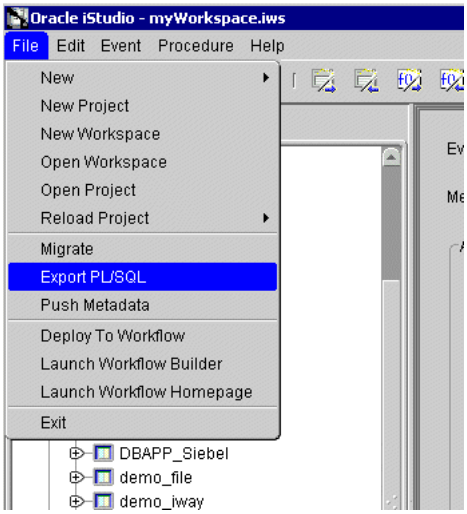
13. To complete the definition of the implemented procedure, click **Next** and then **Finish**.

Exporting PL/SQL Code from iStudio

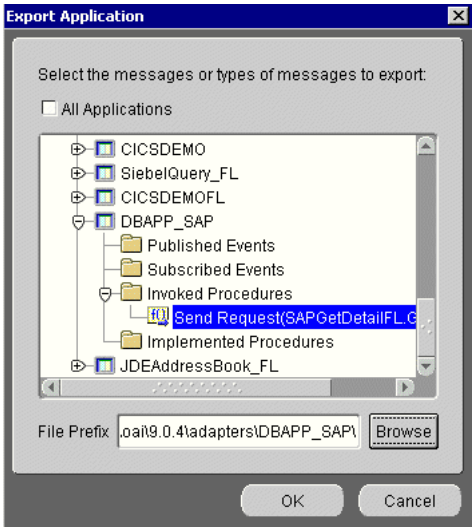
You must export the PL/SQL code created in "Defining an Implemented Procedure" on page 6-13 and run it against the appropriate schema. In this example, the schema used is DBAPP_SAP.

To export PL/SQL code from iStudio:

1. In iStudio, click **File** and **Export PL/SQL**.



The Export Application dialog box is displayed.



Perform the following steps:

a. Select the application from which to export PL/SQL.

b. Type or browse to the file prefix (path to the application).

2. Click **OK**.

In this example, two SQL scripts are created:

- `DBAPP_SAP_SAPGetDetailFLTYPES.sql`
- `DBAPP_SAP_SAPGetDetailFL.sql`

3. Log on to the database with the appropriate privileges (in this example, DBAPP_SAP) and run the following in the order given:

   a. DBAPP_SAP_SAPGetDetailFLTYPES.sql

   b. DBAPP_SAP_SAPGetDetailFL.sql

4. Create another stored procedure, COMPANYGETDETAIL_EXE, in the same schema. It is carried out at runtime to create the database message that is sent to the hub.

```
CREATE OR REPLACE  PROCEDURE "DBAPP_SAP"."COMPANYGETDETAIL_EXE" (
servicename LONG,
methodname  LONG,
license     LONG,
customerid  LONG
)
AS
  moid NUMBER;
  aoid NUMBER;
  coid NUMBER;
  businessname LONG;
  address LONG;
  city LONG;
  state LONG;
  phone LONG;
  country LONG;
  detailid NUMBER;
BEGIN
  SAPGetDetailFL.crMsg_GetDetail_OAI_V1(moid, aoid);
  detailid := SAPGetDetailFL.cr_BAPI_COMPANYCODE_GETDETAIL_
(servicename,methodname,license,customerid,moid,aoid);
  coid := SAPGetDetailFL.inv_GetDetail_OAI_V1(moid,'DBAPP_
SAP','',businessname,address,city,state,phone,country);
COMMIT;
END;
```

Editing the adapter.ini File

The adapter.ini file located under *InterConnect_home*\adapters*appname* may require editing as described in the following procedure. If the following information is missing in adapter.ini, you must add it manually.

1. Open the adapter.ini file located under

   InterConnect_home\adapters*appname*

   Where appname is the name of the application you created.

2. Add the following two lines to adapter.ini:

```
// Bridge class
bridge_class=com.iwaysoftware.iwbridge.IWBridge

// IBSE URL
ibse_url=http://hostname:port/ibse/IBSEServlet/XDSOAPRouter
```

   Where hostname is the name of the Oracle Application Server host and port is the server port number. The default server port number for Oracle Application Server is 7777.

Runtime Configuration

The following topic describes how to verify service integration using OracleAS Adapter for SAP.

Verifying Service Integration

To verify service integration:

1. Start Oracle Application Server or ensure that the server is running.
2. If necessary, restart OC4J by issuing the following command:

   ```
   OracleAS_home\opmn\bin\opmnctl  stopproc process-type=home
   OracleAS_home\opmn\bin\opmnctl  startproc process-type=home
   ```

3. Check the status of OC4J by issuing the following command:

   ```
   OracleAS_home\opmn\bin\opmnctl  status
   ```

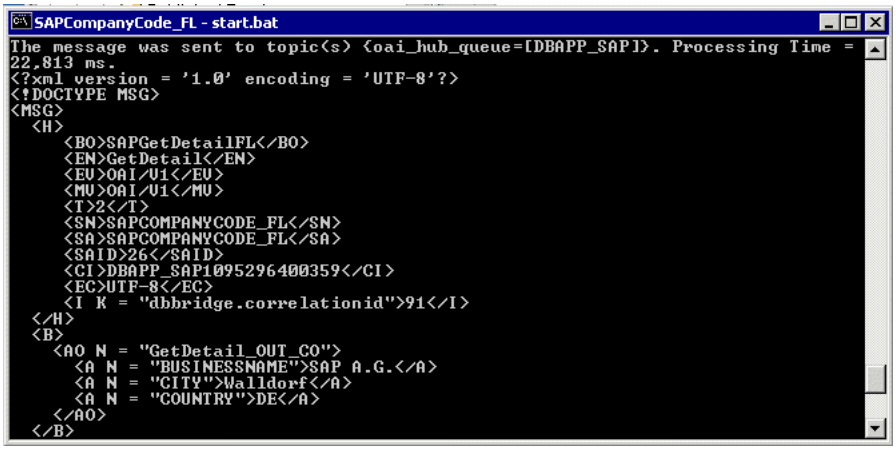
4. Start the adapter by issuing the following commands:

   ```
   InterConnect_home\adapters\SAPCompanyCode_FL\start.bat
   InterConnect_home\adapters\DBAPP_SAP\start.bat
   ```

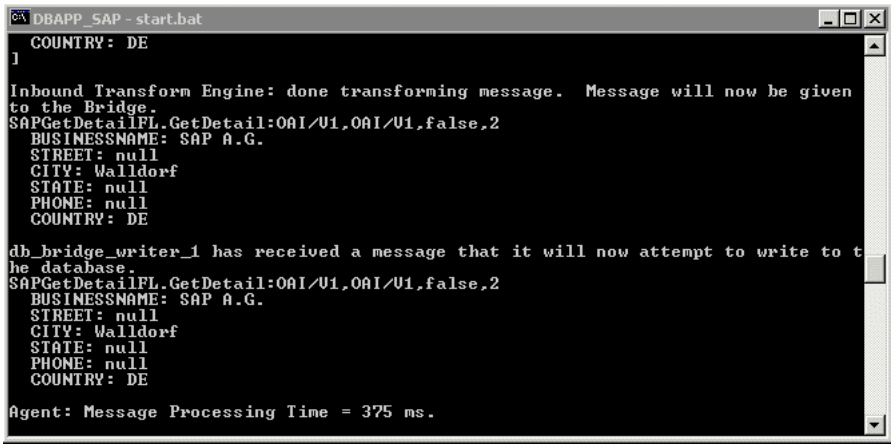
5. Log on to SQL*Plus with `DBAPP_SAP` and run the following command:

   ```
   exec
   companygetdetail_exe
   ('BAPI_COMPANYCODE_GETDETAIL','BAPI_COMPANYCODE_GETDETAIL','test','0001');
   ```

   The following image shows the `SAPCompanyCode_FL` example. It receives a reply from SAP and returns the reply to the hub.



The following image shows the DBAPP_SAP example. It receives a reply from the hub and writes the data to the database table.



SAP Event Integration

This example demonstrates how OracleAS Adapter for SAP integrates with SAP to receive event data. In this example, an SAP event occurs when a customer record is added to an SAP system. The adapter receives the SAP event customer data and disposes the data to an RMI event port. The RMI server resides on the OracleAS Integration InterConnect Hub. An OracleAS Database adapter on the OracleAS Integration InterConnect Hub subscribed to this event receives the customer data, transforms the event data, and then inserts the data into a database table. Design-time and runtime configuration is described in the following sections.

Design-Time Configuration

The following procedures describe how to start the repository and create a common view and then publish and subscribe an event.

Starting the Repository

To start the InterConnect repository, double-click the `start.bat` file located in the following directory:

InterConnect_home`\repository\start.bat`

Creating a Common View

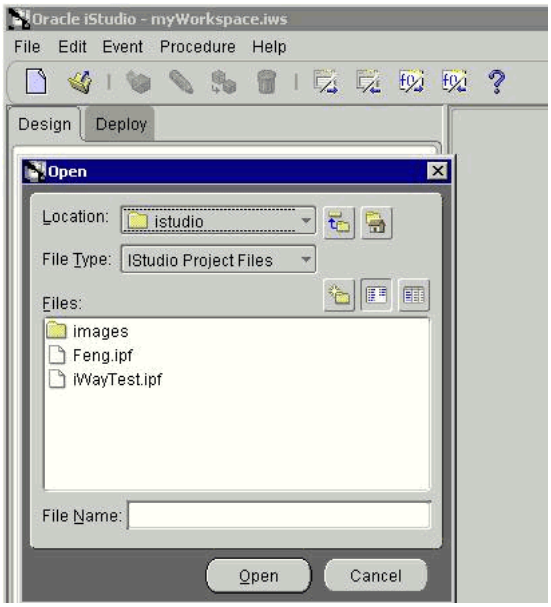
To create a Common View:

1. Start iStudio.

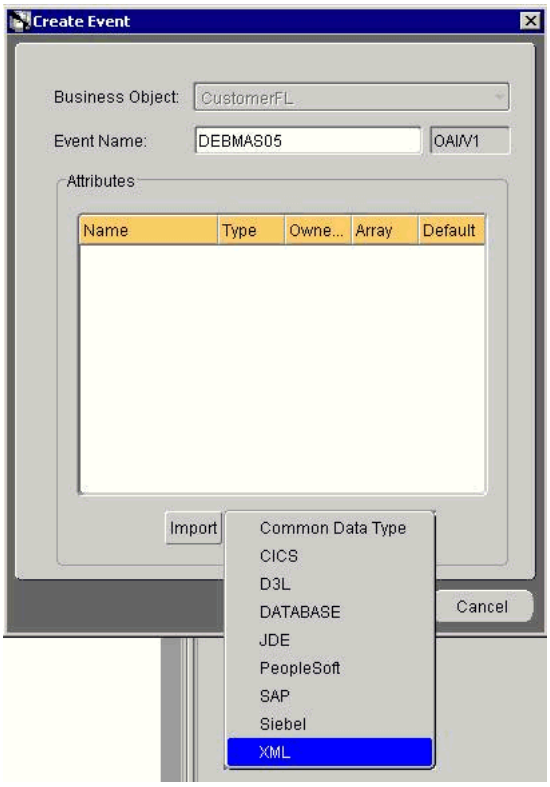
   The iStudio start.bat file is located in the following directory:

   InterConnect_home`\istudio\iStudio.bat`

iStudio starts.

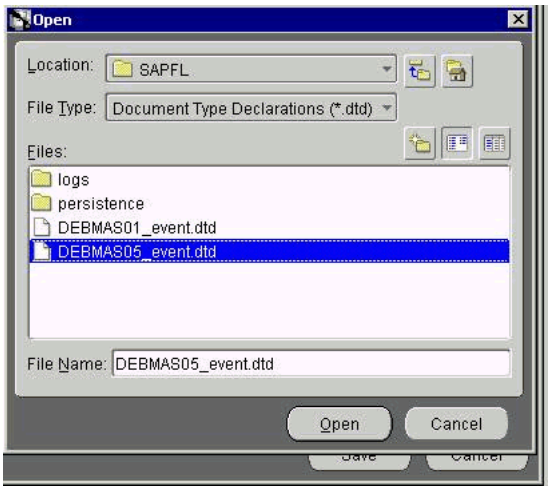


2. Open a new project.
3. Open **Common Views** and **Business Objects**.
4. Create a Business Object called **CustomerFL** and a new event under `CustomerFL`.

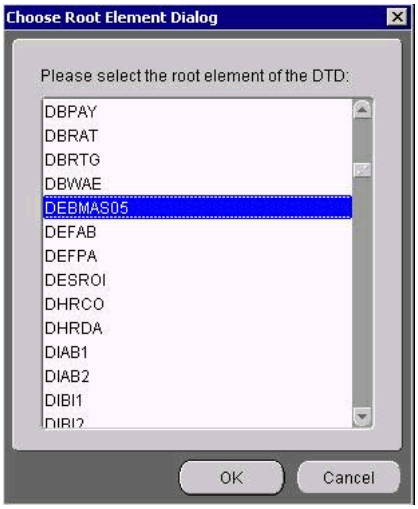


Note: The event name must be the root element of the DTD generated from Application Explorer. In this example, the root element in the DTD is `DEBMAS05`.

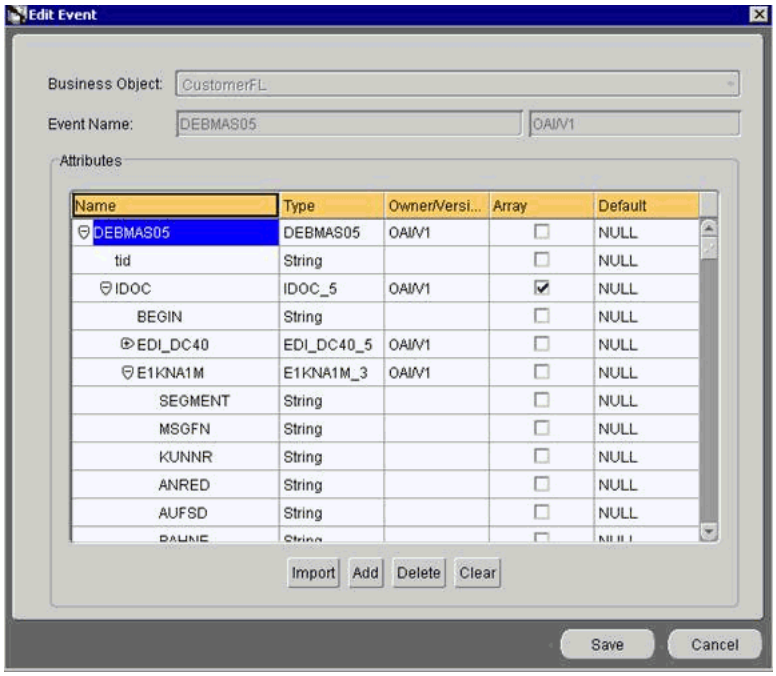
5. Click **Import** and select **XML** from the list.



6. Open the DTD generated from Application Explorer.



7. Select the root element, which must be identical to the event name specified earlier.

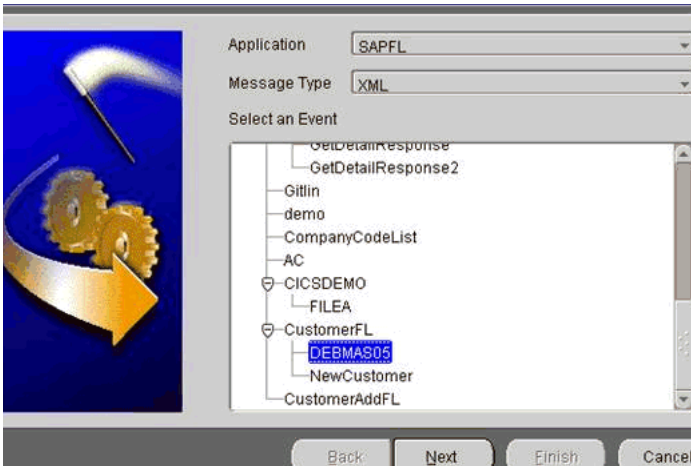


8. Click **Save**.

Publishing an Event

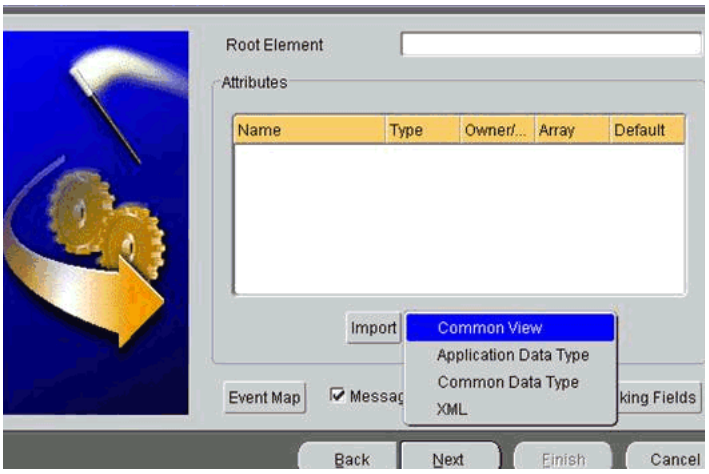
To publish an event:

1. Create a new application called **SAPFL**.
2. Expand **SAPFL**.
3. Right-click **Publish Events** and select **New**.



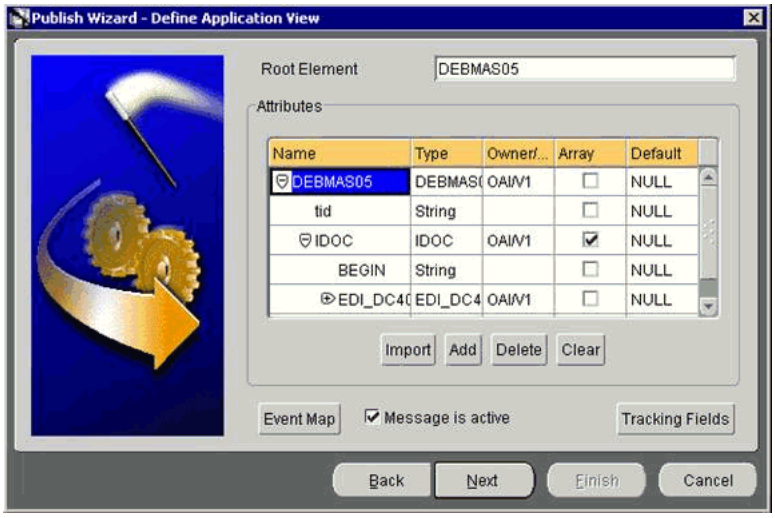
4. Select **XML** as the message type and select **DEBMAS05** under the CustomerFL business object as the event.

5. Click **Next**.



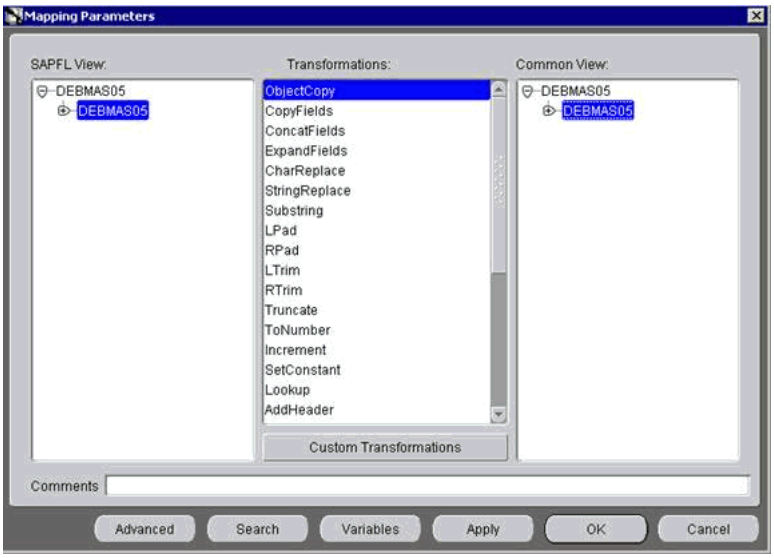
6. Click **Import** and select **Common View** from the list.

   The structure from the selected Common View loads.



7. Ensure you enter the root element of the XML message, for example, `DEBMAS05`.
8. Click **Next**.
9. Click **New** to create a mapping between the Common View and Application View.

In this example, the Application View and Common View have the same structure. All the attributes can be mapped by using the ObjectCopy transformation.



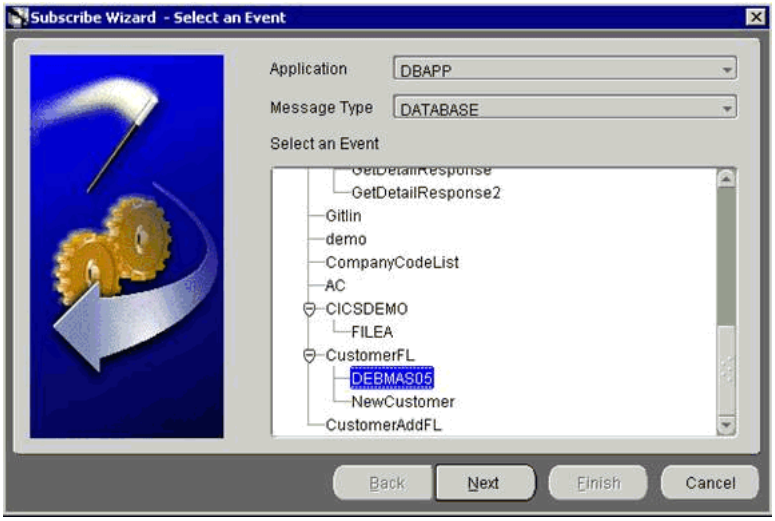
10. Click **Apply**, **OK**, and then **Finish**.

    The application definition for a published event is now complete.

Subscribing an Event

To subscribe an event:

1. Create a new application called **DBAPP**.
2. Expand **DBAPP**.
3. Right-click **Subscribe Events** and select **New**.

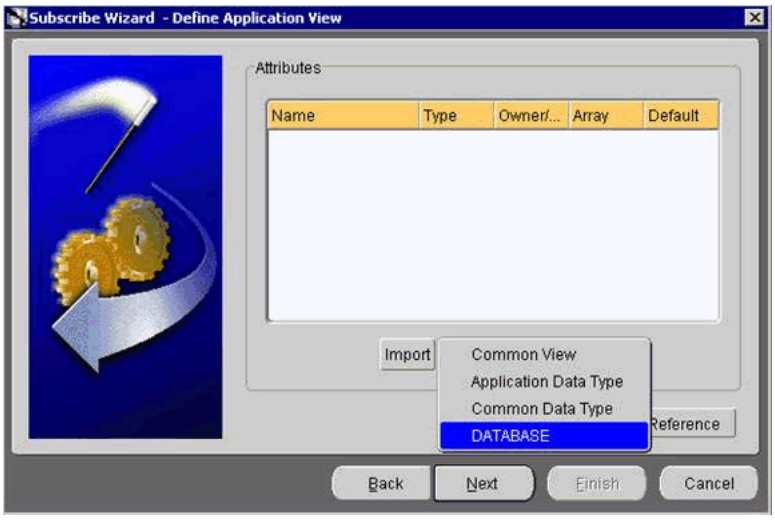


Perform the following steps:

a. Select **Database** as the message type.

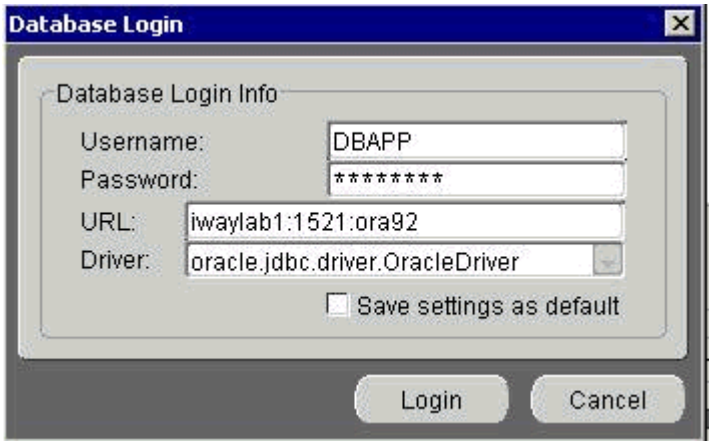
b. Under the `CustomerFL` business object, select **DEBMAS05** as the event.

4. Click **Next**.

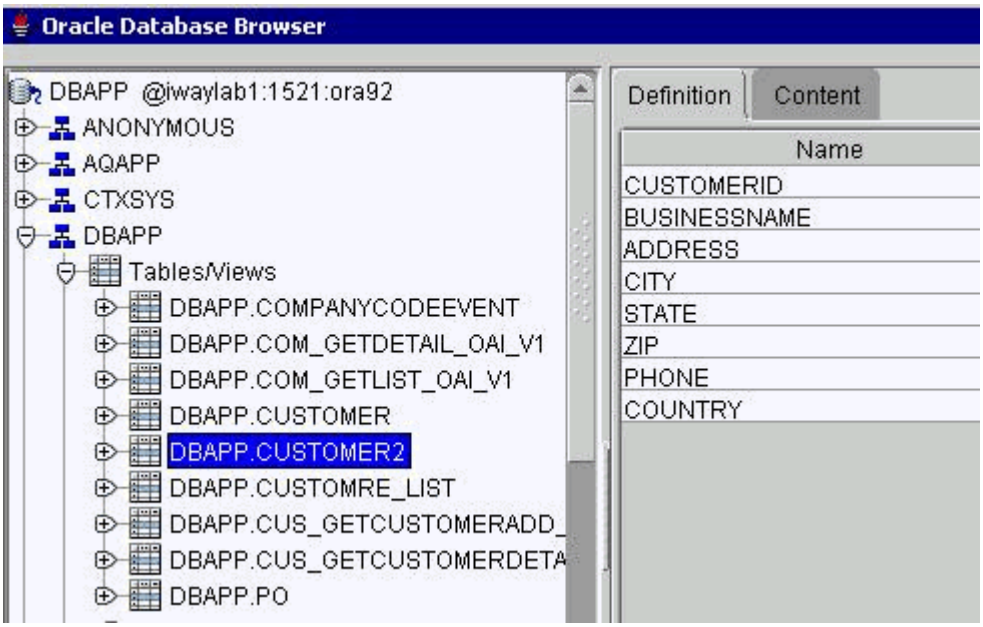


5. Click **Import** and select **Database** from the list.

   The Database Login dialog box is displayed.



6. Type the appropriate login credentials and click **Login**.



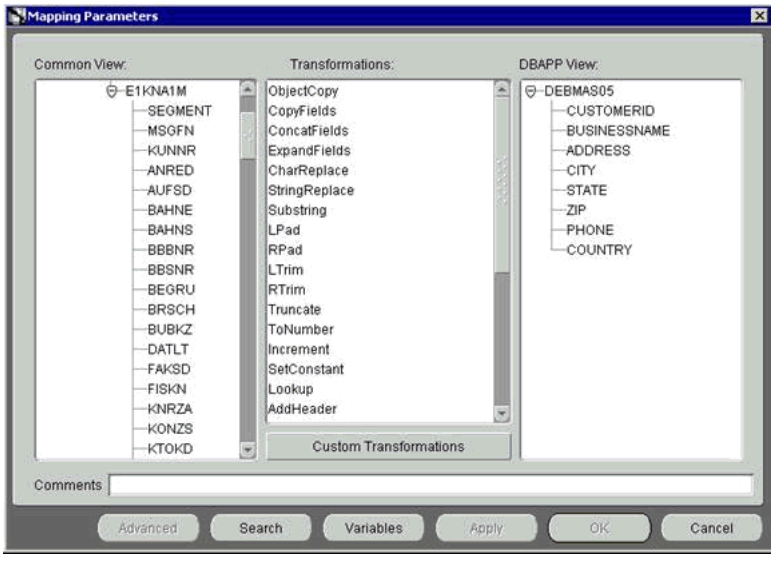
Perform the following steps:

a. Expand the **DBAPP** schema.

b. Select the previously defined table, in this example, `DBAPP.CUSTOMER2`.

7. Click **Next**.

8. Click **New** to define mapping between the Application View and the Common View.



Perform the following steps:

a. Define the mapping for each field.

b. After you map each field, click **Apply**.

9. When you have finished the mapping process, click **OK**.

The SQL code window is displayed.

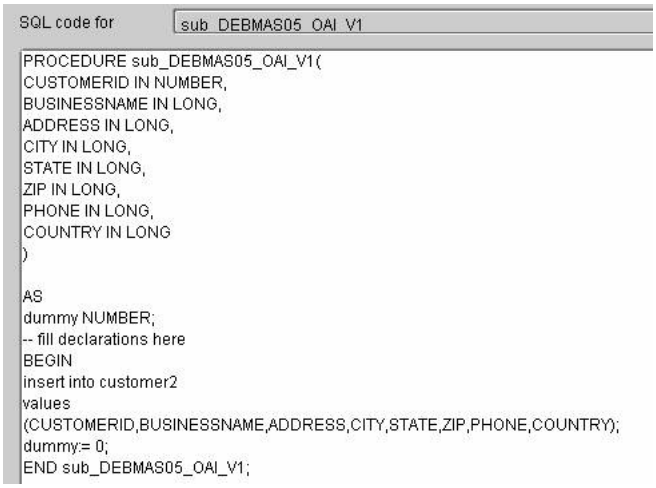
a. Select **sub_DEBMAS05_OAI_V1** from the **SQL** code list.

The procedure details are displayed.

b. Enter the appropriate code, if necessary.

In the following example, the INSERT statement is entered between BEGIN and END to insert received event data into the Customer2 table.

You must run this script to insert event data into the database table.



10. Click **Finish**.

The application definition for a subscribed event is now complete.

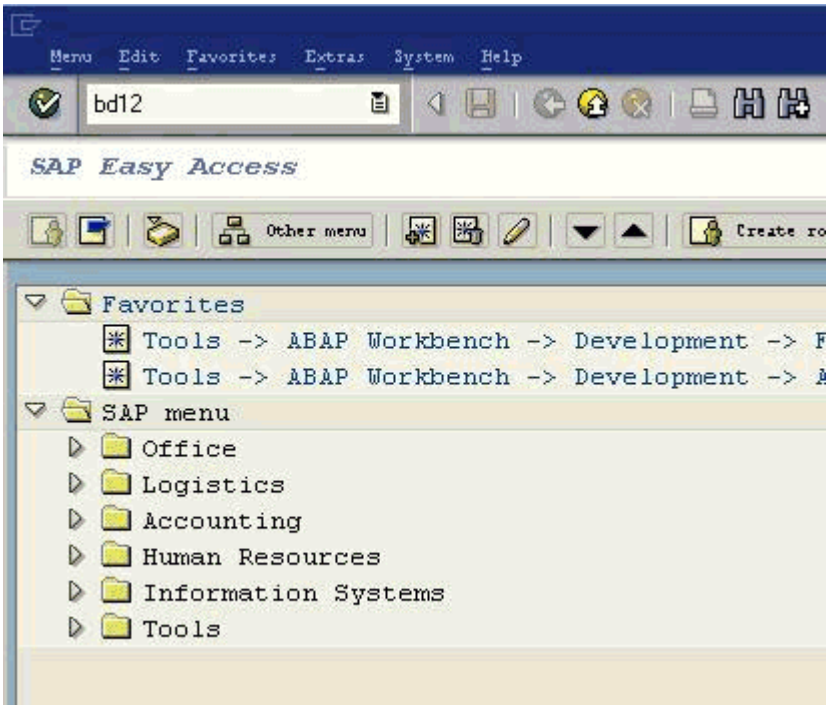
Runtime Configuration

The following topic describes how to trigger an event in SAP and verify event integration using OracleAS Adapter for SAP.

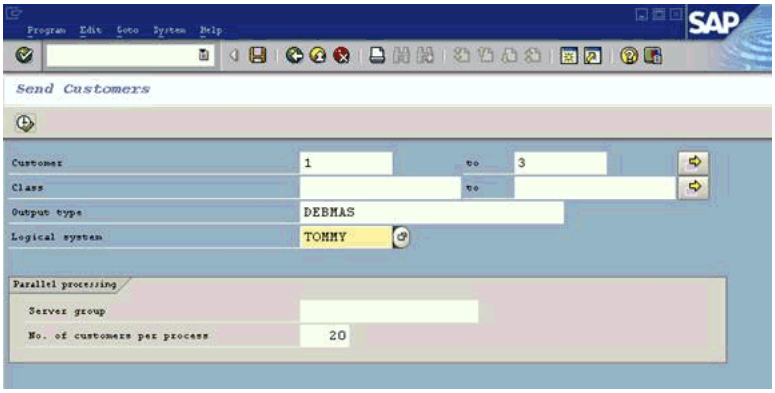
Triggering an Event in SAP

To trigger an event in SAP:

1. Start the SAP Workbench and log in to the SAP system.



2. Carry out the **bd12** transaction.



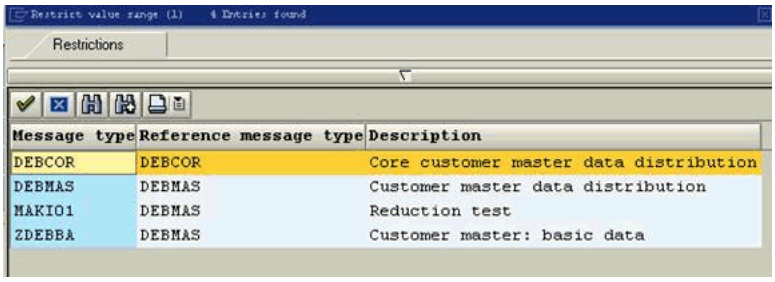
Perform the following steps:

a. In the Customer field, enter a customer number with a range from 1 to 3.

b. In the Output type field, enter **DEBMAS**.

c. In the Logical system field, specify the logical system you are using with SAP.

3. Click the **check mark** icon in the upper left-hand corner.



4. Ensure **DEBMAS** appears in the Message type column.

   Customer master data is sent to the logical system specified. If a channel in Application Explorer defined the Program ID with the same value, the channel receives this customer master data from SAP.

7

Troubleshooting and Error Messages

This chapter explains the limitations and workarounds when connecting to SAP. The following topics are discussed:

- Troubleshooting
- BSE Error Messages

The adapter-specific errors listed in this chapter can arise whether using the adapter with an OracleAS Adapter J2CA or with an OracleAS Adapter Business Services Engine (BSE) configuration.

Troubleshooting

This topic provides troubleshooting information for SAP, separated into four categories:

- Application Explorer
- SAP
- OracleAS Adapter J2CA
- BSE

> **Note:** Log file information that can be relevant in troubleshooting can be found in the following locations:
>
> - The OracleAS Adapter J2CA trace information can be found under the *OracleAS_home*\opmn\logs directory.
> - BSE trace information can be found under the *OracleAS_home*\j2ee\home\applications\ws-app-adapter\ibse\ibselogs directory.
> - The log file for Application Explorer can be found under the *OracleAS_home*\adapters\application\tools directory.

Application Explorer

To use Application Explorer on **Windows** for debugging or testing purposes, load the batch script ae.bat, found under:

OracleAS_home\adapters\application\tools

On **UNIX**, load the shell script ae.sh, found under:

OracleAS_home/adapters/application/tools

Error	Solution
SAP does not appear in the Application Explorer Adapter node list.	Ensure that the `sapjco.jar` and `sapjcorfc.dll` files are added to the `lib` directory. Ensure the `librfc32.dll` file is added to the Windows system32 folder.
Cannot connect to OracleAS Adapter for SAP from Application Explorer. `Problem activating adapter`	Ensure that: ■ SAP is running. ■ The Application Server name, System Number, and Client Number are correct. ■ The SAP user ID and password are correct.
Cannot connect to the SAP target through Application Explorer. The following error message appears: `Error getting target [SAP] - java.lang.Exception: Error Logon to SAP System`	Ensure that you enter the correct connection parameters when connecting to the SAP target.
Cannot connect to your SAP system through Application Explorer. The following error message appears: `Problem activating adapter. (com.ibi.sapr3.SapAdapterException : com.sap.mw.jco.JCO$Exception: (102) RFC_ERROR_COMMUNICATION: Connect to SAP gateway failed Connect_PM GWHOST=isdsrv8, GWSERV=sapgw00, ASHOST=isdsrv8, SYSNR=00 LOCATION CPIC (TCP/IP) on local host ERROR partner not reached (host isdsrv8, service 3300) TIME Fri Aug 27 11:49:14 2004 RELEASE 620 COMPONENT NI (network interface) VERSION 36 RC -10 MODULE ninti.c LINE 979 DETAIL NiPConnect2 SYSTEM CALL SO_ERROR ERRNO 10061 ERRNO TEXT WSAECONNREFUSED: Connection refused COUNTER 1). Check logs for more information`	Ensure that SAP is running and that you are using the correct parameter values to connect to your application server.
Cannot connect to your SAP system through Application Explorer even though SAP is running. The following error message appears: `Problem activating adapter. (com.ibi.sapr3.SapAdapterException : java.lang.ExceptionInInitializerError: JCO.classInitialize(): Could not load middleware layer 'com.sap.mw.jco.rfc.MiddlewareRFC' JCO.nativeInit(): Could not initialize dynamic link library sapjcorfc [no sapjcorfc in java.library.path]. java.library.path`	Ensure that the `sapjcorfc.dll` file is added to the lib directory and the `librfc32.dll` file is added to the Windows system32 folder.

Error	Solution

The dll is loaded in another classloader (BSE and J2CA are installed on the same server). The following error message appears:
`com.ibi.sapr3.SapAdapterException: java.lang.ExceptionInInitializerError: JCO.classInitialize(): Could not load middleware layer 'com.sap.mw.jco.rfc.MiddlewareRFC'`
`JCO.nativeInit(): Could not initialize dynamic link library sapjcorfc [Native Library F:\iWay55.008.0628\lib\sapjcorfc.dll already loaded in another classloader]. java.library.path`
	Add `sapjco.jar` to the server classpath.

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 machine.
Logon failure error at runtime	If the password for connecting to your SAP system is not specified when creating a target or with the Edit option in Application Explorer, you will be unable to connect to SAP. 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.bat`): `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`

SAP

Error	Solution
When executing a request, the following error message appears: `AdapterException: java.lang.Exception: Function module CUSTOMER_GETDETAIL2 does NOT exist.`	Check the syntax of your input XML document and make sure the name of the Remote Function module is correct, available in SAP, and activated.
When executing a request, the following error message appears: `AdapterException: java.lang.Exception: Object type unknown for business object: CUST`	Check the syntax of your input XML document and verify that the Business Object type exists in SAP and is correct and activated. Also verify that the name of your document is correct.
When executing a request, the following error message appears: `AdapterException: java.lang.Exception: Unable to retrieve BAPI name for: CUSTOMER.DETAIL2`	Check the syntax of your input XML document and verify that the name of the BAPI is correct, available in SAP, and activated.
When executing a request, the following error message appears: `java.lang.RuntimeException: com.sap.mw.jco.JCO$AbapException: (126) OBJECT_UNKNOWN: Basic type or extension does not exist.`	Check the syntax of your input XML document and verify that the IDoc type or extension type is correct, available in SAP, and activated.
When executing a request, the following error message appears: `AdapterException: java.lang.Exception: BapiError/BapiAbort: You are not authorized to display customers.`	Verify that your user ID has the correct permissions configured in SAP. Consult your SAP administrator for more information.

Note: Activation is an important SAP concept. If an object does not exist in an activated state, it may appear as present on the system, but is not available for use. This is especially important for IDOCs and SAP Business Objects. Consult your SAP documentation for further information.

OracleAS Adapter J2CA

Error	Solution
In Application Explorer, the following error message appears when you attempt to connect to an OracleAS 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, *`OracleAS_home`*`\adapters\application`

BPEL Process Manager

Error	Solution
Endpoint activation error on deployment of SAP event handling project (inbound) in JDeveloper	Verify that the channel used for this inbound J2CA service is stopped in Application Explorer. If you have started this channel for testing or debugging purposes, you must stop it before starting BPEL PM Server. Endpoint activation is managed by BPEL Process Manager.
The following error message appears in BPEL PM Server Console: `Process "TestSAP" (revision "1.0") compilation failed. <2005-05-18 10:49:53,285> <ERROR><default.collaxa.cube.engine.deployment> <Cube ProcessLoader::create> Failed to read wsdl. Error happened when reading wsdl at "http://127.0.0.1:7777/BPELConsole/wsil/adapters/applications/CUSTOMER_invoke.wsdl?wsdl", because "WSDLException: faultCode=INVALID_WSDL: Invalid XML in document at: http://127.0.0.1:7777/BPELConsole/wsil/adapters/applications/CUSTOMER_invoke.wsdl?wsdl: The element type "P" must be terminated by the matching end-tag "</P>".`	Verify that the specified WSDL file exists at that URL and that the file is valid. Workaround: Change the WSDL location to `localhost:7777`. The default is `127.0.0.1:7777`. Alternative workaround: Add the IP address to the `Dhttp.nonProxyHosts` list found in `obsetenv.bat` (Windows) or `obsetenv.sh` (Unix)
The following exception is thrown in JDeveloper during deployment of the BPEL process: `java.io.FileNotFoundException: \BPELConsole\wsil\adapters\applications\DEBMAS01_receive.wsdl?wsdl (The system cannot find the path specified)`	Verify that you have all the required patches installed. The required patches are listed and updated on the Oracle Technology Network Web site.

BSE Error Messages

This topic discusses the different types of errors that can occur when processing Web services through BSE.

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 runtime, various conditions can cause errors in BSE when Web services that use adapters are running. 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 if 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, 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.

Adapter-Specific Error Handling

When an adapter raises an exception during runtime, 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 or not 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, 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.

OracleAS Adapter for SAP Invalid SOAP Request

If OracleAS Adapter for SAP receives a SOAP request message that does not conform to the WSDL for the Web services being carried out, 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>Error processing agent [XDSapIfrAgent] - XD[FAIL] SapIFRException:
java.sql.SQLException:
com.ibi.sapjco.SapCallableStatement: execute() j
java.util.NoSuchElementException</faultstring>
  </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Empty Result From SOAP Request

If OracleAS Adapter for SAP carries out an SAP object using input parameters passed in the SOAP request message that do not match records in SAP, 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>Error processing agent [XDSapIfrAgent] - XD[FAIL] SapIFRException:
java.sql.SQLException: com.ibi.sapjco.SapCallableStatement: execute()
java.sql.SQLException: JCO Error Key: NO_RECORD_FOUND Short Description:
com.sap.mw.jco.JCO$AbapException: (126) NO_RECORD_FOUND: NO_RECORD_
FOUND</faultstring>
   </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
  </SOAP-ENV:Envelope>
```

Failure to Connect to SAP

If OracleAS Adapter for SAP cannot connect to SAP when executing a Web service, 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>Error processing agent [XDSapIfrAgent] - XD[RETRY]
Connect to SAP gateway failed Connect_PM GWHOST=ESDSUN, GWSERV=sapgw00,
ASHOST=ESDSUN,
SYSNR=00 LOCATION CPIC (TCP/IP) on local host ERROR partner not reached (host
ESDSUN, service 3300)
TIME Mon Jun 30 16:01:02 2003 RELEASE 620 COMPONENT NI (network interface) VERSION
36 RC -10 MODULE ninti.c LINE 976 DETAIL NiPConnect2
SYSTEM CALL SO_ERROR ERRNO 10061 ERRNO TEXT WSAECONNREFUSED: Connection refused
COUNTER 1</faultstring>
  </SOAP-ENV:Fault>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Invalid SOAP Request

If OracleAS Adapter for SAP receives a SOAP request message that does not conform to the WSDL for the Web services being carried out, 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>
```

Empty Result From OracleAS Adapter for SAP Request

If OracleAS Adapter for SAP carries out 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>
```

8

Advanced User Tools

This chapter includes the following topics:

Web Services Policy-Based Security

Application Explorer provides a security model called Web services policy-based security. The following topics describe how the feature works and how to configure it.

Web services provide a layer of abstraction between the back-end business logic and the user or application running the Web service. This enables easy application integration but raises the issue of controlling the use and implementation of critical and sensitive business logic that is run as a Web service.

Application Explorer controls the use of Web services that use adapters, using a feature called policy-based security. This feature enables an administrator to apply "policies" to Business Services (Web services) to deny or permit their execution.

A policy is a set of privileges dealing with the execution of a Business Service (BS) that can be applied to an existing or new BS. When you set specific rights or privileges inside a policy, you do not have to re-create privileges for every BS that has security concerns in common with other Business Services. Instead, you reuse a policy on multiple Business Services.

The goal of the feature is to secure requests at both the transport and the SOAP request level transmitted on the wire. Some of the policies do not deal with security issues directly, but do affect the runtime behavior of the Web services to which they have been applied.

The Business Services administrator creates an "instance" of a policy type, names it, associates individual users or groups (a collection of users), and then applies that policy to one or more Business Services.

You can assign a policy to a Business Service, or to a method within a Business Service. If a policy is only applied to a method, other methods in that Business Service will not be governed by it. However, if a policy is applied to the Business Service, all methods are governed by it. At runtime, the user ID and password that are sent to BSE in the SOAP request message are verified against the list of users for all policies applied to that specific Business Service. The policy type that is supported is Resource Execution, which dictates who can or cannot carry out the Business Service.

When a policy is not applied, the default value for a Business Service is to "grant all". For example, anybody can run the Business Service, until the Resource Execution policy is associated to the Business Service. At that time, only those granted execution

permissions, or users not part of the group that has been denied execution permissions, have access to the Business Service.

Configuring Web Services Policy-Based Security

The following procedures describe how to configure Web services policy-based security.

Creating and Associating a User with a Policy

Before you create instances of policies, you must have a minimum of one user or one group to associate to an instance. You can create users and groups using Application Explorer.

1. Open Application Explorer.
2. Right-click the configuration to which you want to connect, for example, `SampleConfig`. See Chapter 2, "Configuring OracleAS Adapter for SAP" for information on creating a new configuration.
3. Select **Connect**.

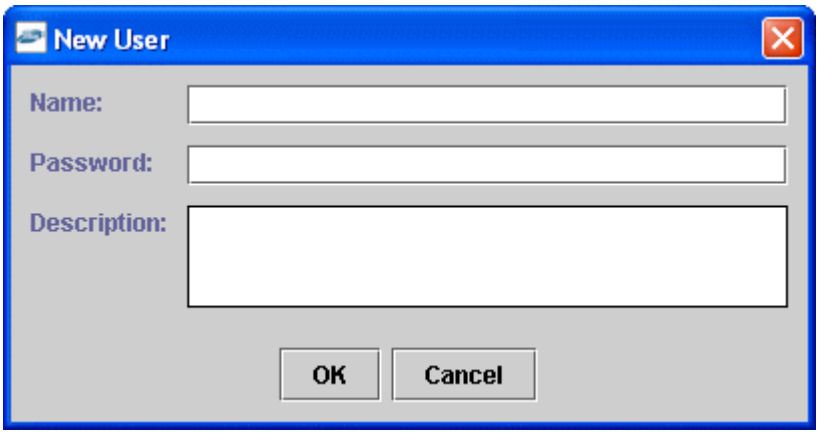
   Nodes appear for Adapters, Events, and Business Services (also known as Web services).

   Perform the following steps:

   a. Expand the **Business Services** node.
   b. Expand the **Configuration** node.
   c. Expand the **Security** node.
   d. Expand the **Users and Groups** node.

4. Right-click **Users** and click **New User**.

The New User dialog box is displayed.



Provide the following information:

a. In the **Name** field, enter a user ID.

b. In the **Password** field, enter the password associated with the user ID.

c. In the **Description** field, enter a description of the user (optional).

5. Click **OK**.

The new user is added under the Users node.

Creating a Group to Use With a Policy

To create a group to use with a policy:

1. Open Application Explorer.
2. Right-click the configuration to which you want to connect, for example, SampleConfig. See Chapter 2, "Configuring OracleAS Adapter for SAP" for information on creating a new configuration.
3. Select **Connect**.

   Nodes appear for Adapters, Events, and Business Services (also known as Web services).

Perform the following steps:

a. Expand the **Business Services** node.

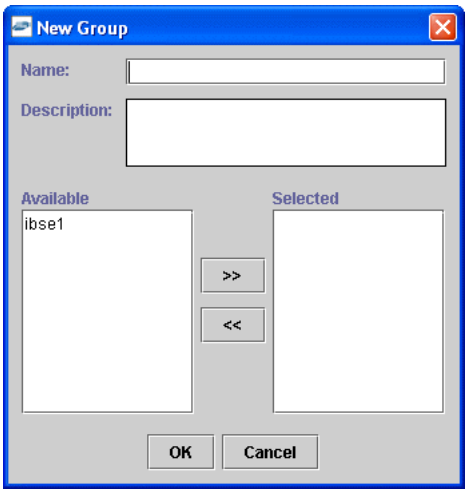
b. Expand the **Configuration** node.

c. Expand the **Security** node.

d. Expand the **Users and Groups** node.

4. Right-click **Groups** and select **New Group**.

The New Group dialog box is displayed.



Provide the following information:

a. In the **Name** field, enter a name for the group.

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

c. From the available list of users in the left pane, select one or more users and add them to the **Selected** list by clicking the double right-facing arrow.

5. When you have selected at least one user, click **OK**.

The new group is added under the Group node.

Creating an Execution Policy

An execution policy governs who can run the Business Services to which the policy is applied.

To create an execution policy:

1. Open Application Explorer.
2. Right-click the configuration to which you want to connect, for example, SampleConfig. See Chapter 2, "Configuring OracleAS Adapter for SAP" for information on creating a new configuration.
3. Select **Connect**.

Nodes appear for Adapters, Events, and Business Services (also known as Web services).

Perform the following steps:

a. Expand the **Business Services** node.

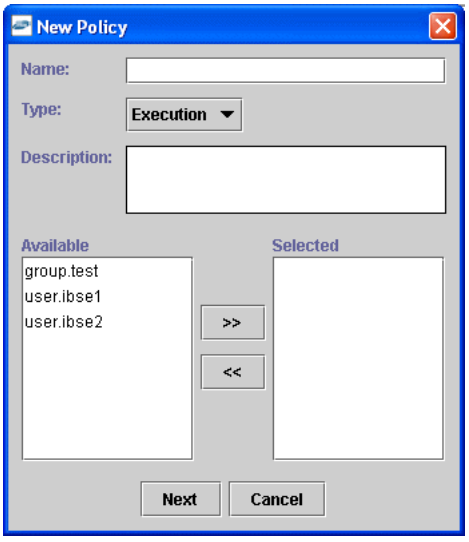
b. Expand the **Configuration** node.

c. Expand the **Security** node.

d. Expand the **Policies** node.

4. Right-click **Policies** and select **New Policy**.

The New policy dialog box is displayed.



Provide the following information:

a. In the **Name** field, enter a name for the policy.

b. From the **Type** list, select **Execution**.

c. In the **Description** field, enter a description for the policy (optional).

d. From the available list of users in the left pane, select one or more users and add them to the **Selected** list by clicking the double right-facing arrow.

Note: This user ID is verified against the value in the user ID element of the SOAP header sent to BSE in a SOAP request.

---

5. When you have selected at least one user selected, click **OK**.
6. Click **Next**.

   The New Policy permissions dialog box is displayed.

   - To grant permission to a user or group to run a Business Service, select the user or group and move them into the **Execution Granted** list by selecting the double left-facing arrow.
   - To deny permission to a user or group to run a Business Service, select the user or group and move them into the **Execution Denied** list by selecting the double right-facing arrow.

7. Click **OK**.

   The following pane summarizes your configuration.

- **Name** test
- **Type** Execution
- **Description**
- **User and Group Restrictions**
  - group.test Execution Granted

Using the IP and Domain Restrictions Policy Type

You configure the IP and Domain Restriction policy type slightly differently from other policy types. The IP and Domain Restriction policy type controls connection access to BSE and therefore need not be applied to individual Web services. You need not create a policy; however, you must enable the Security Policy option in Application Explorer.

1. Open Application Explorer.
2. Right-click the configuration to which you want to connect, for example, SampleConfig. See Chapter 2, "Configuring OracleAS Adapter for SAP" for information on creating a new configuration.

3. Select **Connect**.

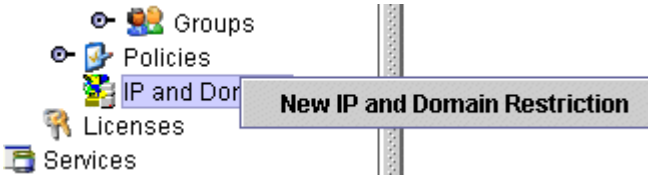
   Nodes appear for Adapters, Events, and Business Services (also known as Web services).

   a. Expand the **Business Services** node.

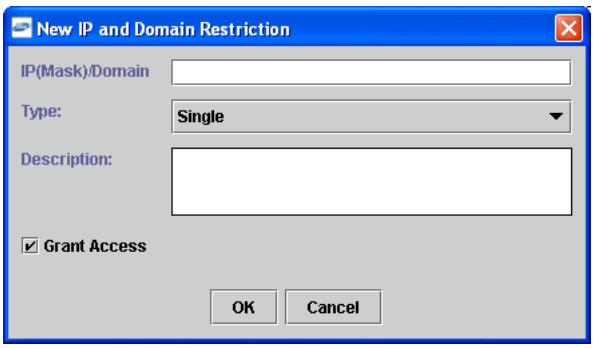
   b. Expand the **Configuration** node.

   c. Expand the **Security** node.

4. Right-click **IP and Domain** and select **New IP and Domain Restriction**.



   The New IP and Domain Restriction dialog box is displayed.



   Provide the following information:

   a. In the **IP(Mask)/Domain** field, enter the IP or domain name using the following guidelines.

      If you select **Single** (Computer) from the **Type** list, you must provide the IP address for that computer. If you only know the DNS name for the computer, click **DNS Lookup** to obtain the IP Address based on the DNS name.

      If you select **Group** (of Computers), you must provide the IP address and subnet mask for the computer group.

      If you select **Domain**, you must provide the domain name.

   b. From the **Type** list, select the type of restriction.

   c. In the **Description** field, enter a description (optional).

   d. To grant access, select the **Grant Access** check box.

5. Click **OK**.

   The new domain is added under the IP and Domain node.

The following pane summarizes your configuration.

- **IP Address(Mask)/Domain** www.yahoo.com
- **Type** Domain
- **Access** Denied
- **Description**

Migrating Repositories

During design time, the Oracle 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 runtime. For management purposes, you can migrate BSE and J2CA repositories that are configured for Oracle to new destinations without affecting your existing configuration. For example, you may want to migrate a repository from a test environment to a production environment.

Migrating a BSE Repository

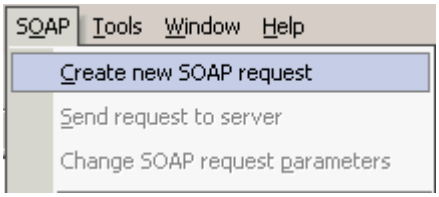
To migrate a BSE repository:

1. Copy the BSE control service URL, for example:

   ```
   http://localhost:7777/ibse/IBSEServlet/admin/iwcontrol.ibs
   ```

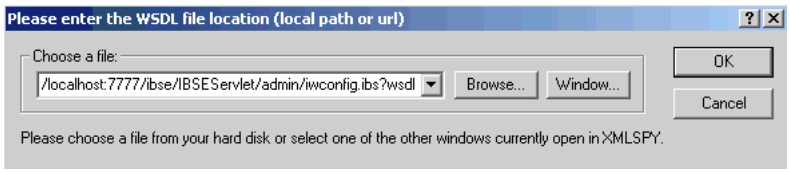
2. Open a third-party XML editor, for example, XMLSPY.
3. From the menu bar, click **SOAP**.

   A list of options appears.



4. Select **Create new SOAP request**.

   The WSDL file location dialog box is displayed.



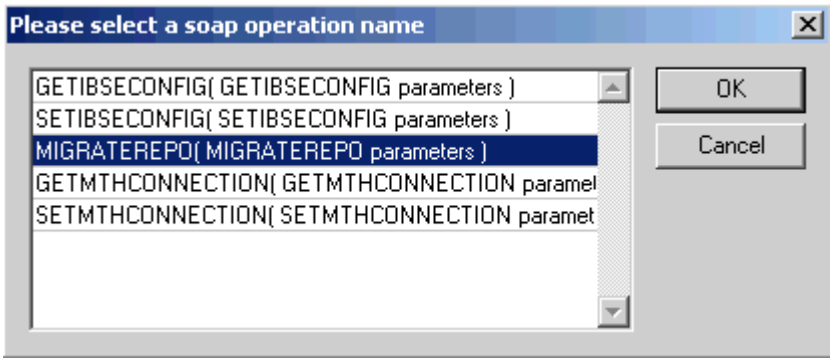
   Perform the following steps:

   a. In the **Choose a file** field, paste the BSE control service URL.
   b. Append **?wsdl** to the URL, for example:

      ```
      http://localhost:7777/ibse/IBSEServlet/admin/iwcontrol.ibs?wsdl
      ```

5. Click **OK**.

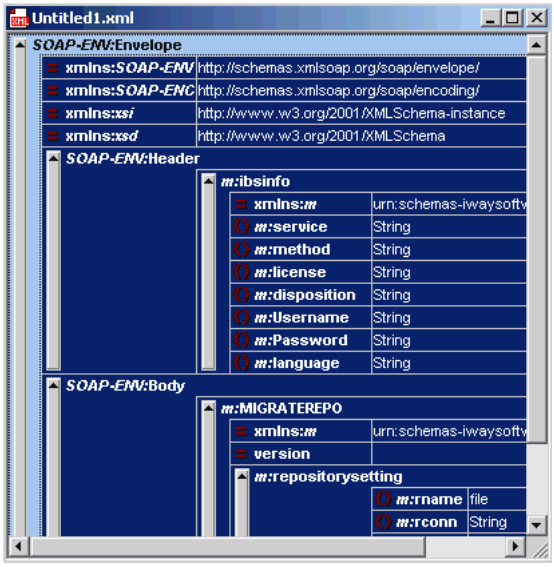
The soap operation name dialog box is displayed and the available control methods are listed.



6. Select the **MIGRATEREPO(MIGRATEREPO parameters)** control method and click **OK**.

> **Note:** The **MIGRATEREPO(MIGRATEREPO parameters)** control method is available from the BSE administration console. This control method migrates all Web services to the new (empty) repository. You can choose to migrate select Web services only.

The following window is displayed, showing the structure of the SOAP envelope.



7. Locate the **Text view** icon in the toolbar.

8. To display the structure of the SOAP envelope as text, click **Text view**.

   The `<SOAP-ENV:Header>` tag is not required and can be deleted from the SOAP envelope.

9. Locate the following section:

```
<m:MIGRATEREPO xmlns:m="urn:schemas-iwaysoftware-com:jul2003:ibse:config"
version="">
  <m:repositorysetting>
    <m:rname>oracle</m:rname>
    <m:rconn>String</m:rconn>
    <m:rdriver>String</m:rdriver>
    <m:ruser>String</m:ruser>
    <m:rpwd>String</m:rpwd>
  </m:repositorysetting>
  <m:servicename>String</m:servicename>
</m:MIGRATEREPO>
```

   Perform the following steps:

   a. For the `<m:rconn>` tag, replace the String placeholder with a repository URL where you want to migrate your existing BSE repository.

      The Oracle repository URL has the following format:

      ```
      jdbc:oracle:thin:@[host]:[port]:[sid]
      ```

   b. For the `<m:rdriver>` tag, replace the String placeholder with the location of your Oracle driver.

   c. For the `<m:ruser>` tag, replace the String placeholder with a valid user name to access the Oracle repository.

   d. For the `<m:rpwd>` tag, replace the String placeholder with a valid password to access the Oracle repository.

10. Perform one of the following migration options.

   - If you want to migrate a single Web service from the current BSE repository, enter the Web service name in the `<m:servicename>` tag, for example:

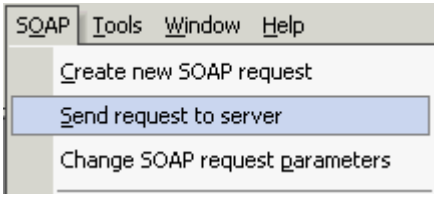
     ```
     <m:servicename>SAPService1</m:servicename>
     ```

   - If you want to migrate multiple Web services from the current BSE repository, duplicate the `<m:servicename>` tag for each Web service, for example:

     ```
     <m:servicename>SAPService1</m:servicename>
     <m:servicename>SAPService2</m:servicename>
     ```

   - If you want to migrate all Web services from the current BSE repository, remove the `<m:servicename>` tag.

11. From the menu bar, click **SOAP** and select **Send request to server**.



   Your BSE repository and any Web services you specified are now migrated to the new Oracle repository URL you specified.

Migrating a J2CA Repository

To migrate a J2CA repository:

1. Navigate to the location of your J2CA configuration directory where the repository schemas and other information is stored, for example:

   OracleAS_home\adapters\application\config*JCA_CONFIG*

   Where JCA_CONFIG is the name of your J2CA configuration.

2. Locate and copy the repository.xml file.

3. Place this file in a new J2CA configuration directory to migrate the existing repository.

   Your J2CA repository is migrated to the new J2CA configuration directory.

A

Configuring SAP for Inbound and Outbound Processing

During inbound (client) processing, IDocs are transferred to the interface and stored in the R/3 System. The document data is generated in a second step, also in the course of a workflow.

Outbound processing in SAP involves event handling. An event in SAP is defined as an occurrence of a status change in an object. Events are created when the relevant status change occurs.

The following topics describe how to enable inbound and outbound SAP processing.

Configuring SAP Inbound Processing

SAP inbound processing requires the upstream system to transfer an IDoc to the IDoc interface through the R/3 System port. For this reason, you do not have to specify a port in the inbound partner profiles; the IDoc interface only must recognize the upstream system as a port. A port definition, which provides a unique ID for the upstream system, must be available for the port. The technical parameters of this port definition can (and usually are) overwritten by the upstream system.

If the upstream system is recognized, then the IDoc is saved in the database. If a partner is defined with the corresponding message in partner profiles, the IDoc is then processed further. This is done independently in the second step. This ensures that the external system can receive the data quickly and reliably (automatically).

You must perform the following steps to configure SAP for inbound IDoc processing:

1. Configure a logical system.
2. Configure a distribution model.
3. Define an inbound partner profile.

Configuring a Logical System

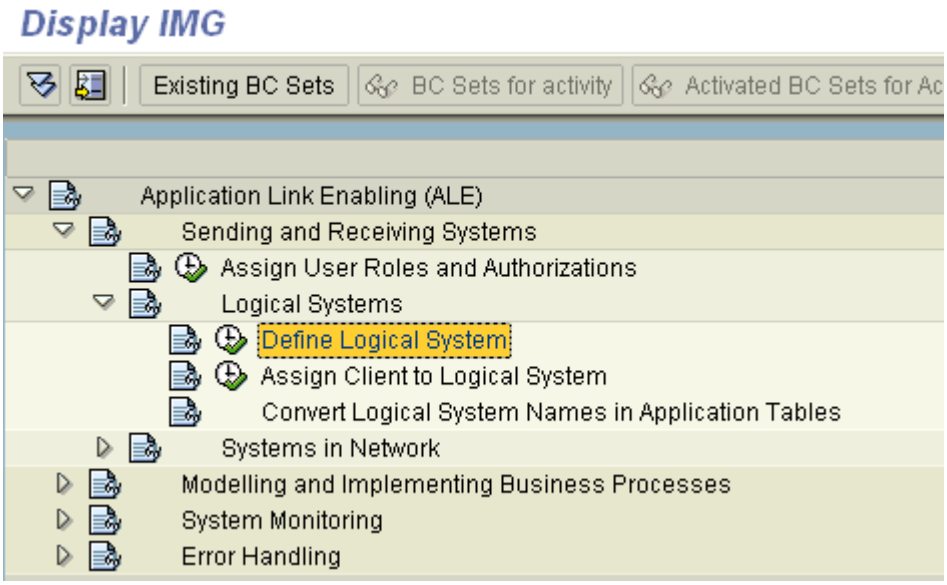
In any distributed environment, each participating system must have a unique ID to avoid confusion. In SAP, the name of the logical system is used as the unique ID. This name is assigned explicitly to one client in an SAP system.

Defining a Logical System

To define a logical system:

1. Run the **sale** transaction.

The Display IMG window is displayed.



Perform the following steps:

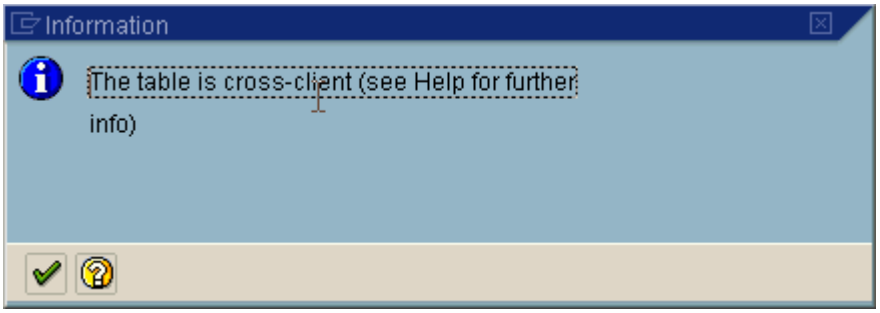
a. Expand **Sending and Receiving Systems**.

b. Expand **Logical Systems**.

c. Select **Define Logical System**.

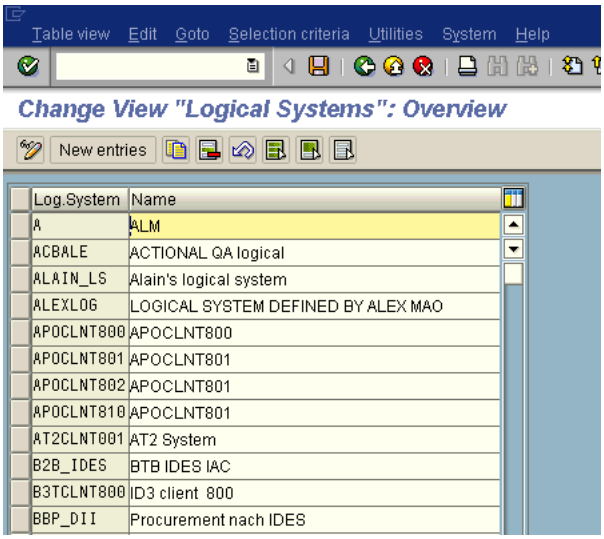
2. Click the **IMG - Activity** icon.

A message window is displayed. It indicates that the table is cross-client.



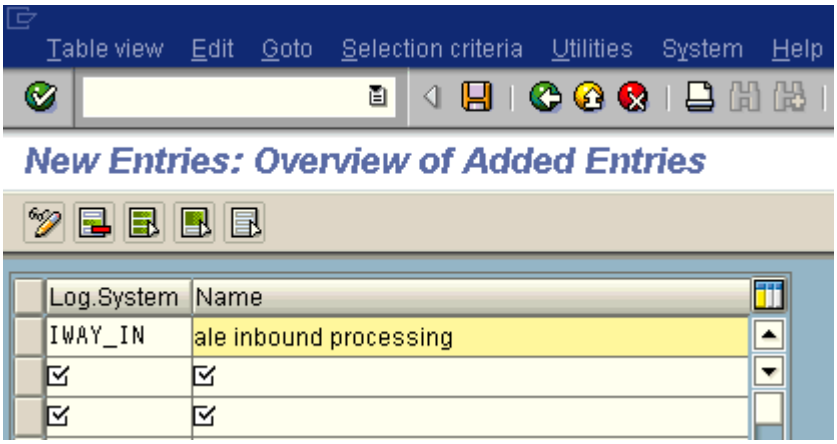
3. Click the **check mark** icon to continue.

The Change View "Logical Systems": Overview window is displayed.



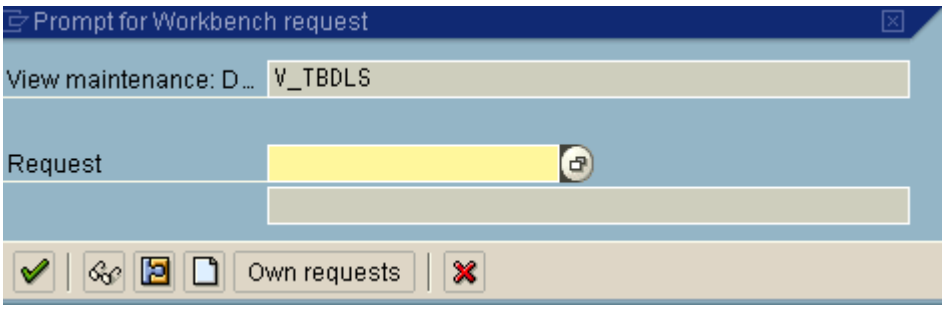
4. Click **New Entries**.

   The New Entries: Overview of Added Entries window is displayed.



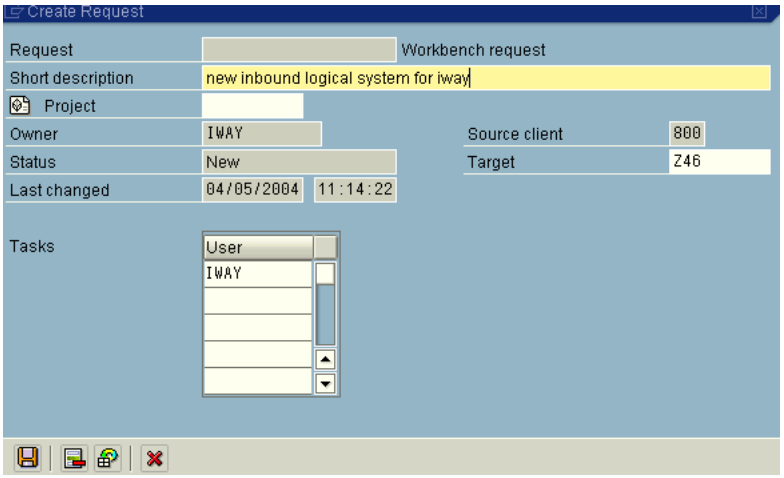
5. Enter the Logical System, for example, ORACLETDS, in the **Log.System** column and provide a description in the **Name** column.
6. Click **Save**.

   The Prompt for Workbench request dialog box is displayed.



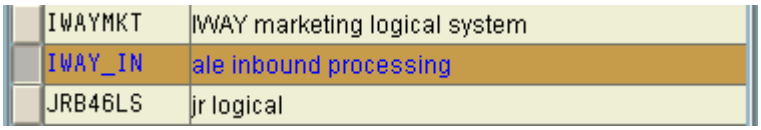
7. Click the **Create Request** icon.

The Create Request dialog box is displayed.



8. Enter a name and description for your request and click **Save**.

   The logical system you configured, for example, ORACLETDS, is now added to the list.



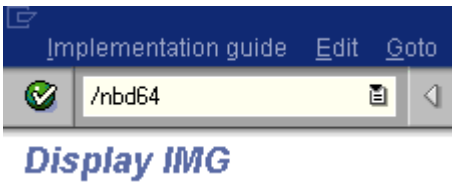
Configuring a Distribution Model

A distribution model is used to describe the ALE message flow between logical systems. Business objects are distributed to connected recipients according to a unique distribution model that can contain rules of varying complexity depending on the type of business objects involved.

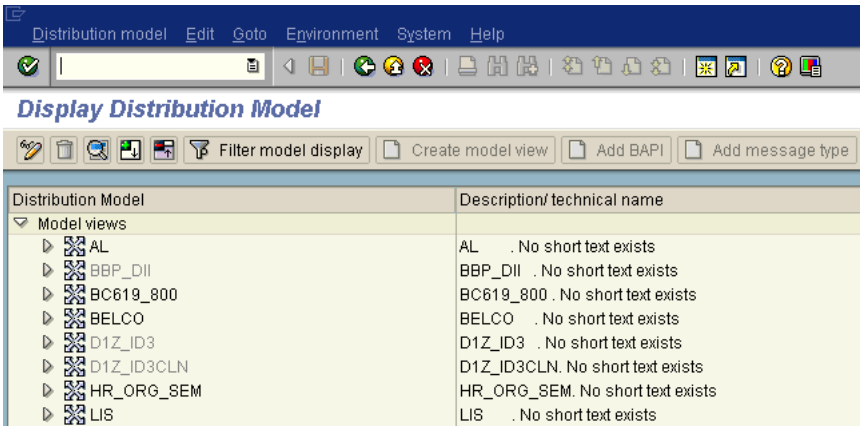
Defining a Distribution Model

To define a distribution model:

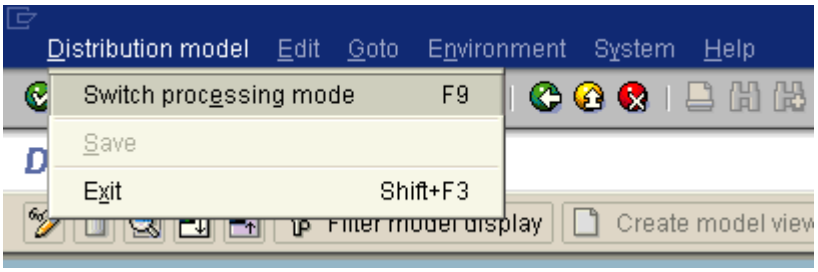
1. Run the **bd64** transaction.



The Display Distribution Model window is displayed.

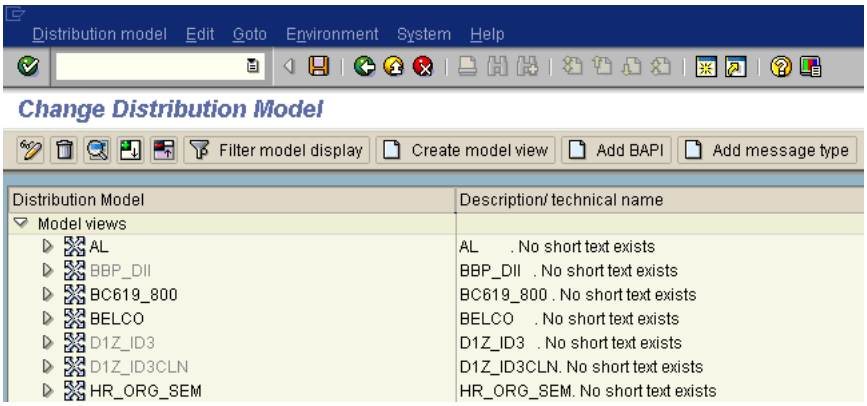


2. Click **Distribution Model** from the menu bar.



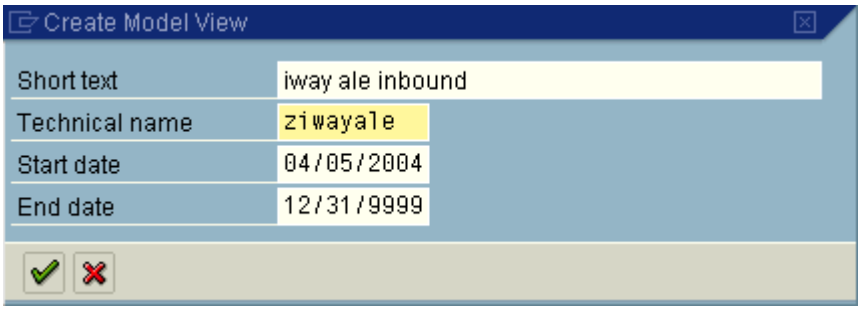
3. Select **Switch processing mode**.

   The Display Distribution Model window is switched to Change Distribution Model.



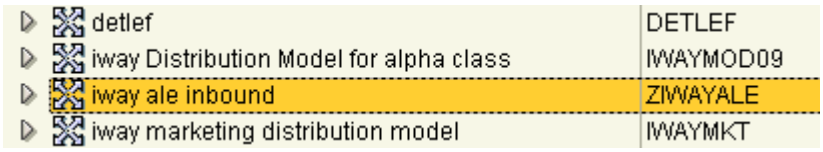
4. Click **Create model view**.

The Create Model View dialog box is displayed.



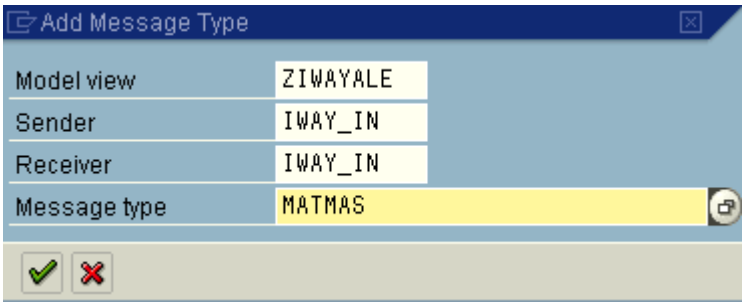
5. Enter a model view name in the **Short text** field and a name in the **Technical name** field, which also serves as a description.
6. Click the **check mark** icon to enter the information.

   You are returned to the main Change Distribution Model window. The distribution model you configured is now added to the list.



7. Click **Add message type**.

   The Add Message Type dialog box is displayed.



   Perform the following steps:

   a. In the **Sender** and **Receiver** fields, enter the logical system you configured, for example, ORACLETDS.

      You can click the icon to the right of each field to browse from a list of logical systems.

   b. In the **Message type** field, enter the message type you want to use, for example, MATMAS.

      You can click the icon to the right of each field to browse from a list of available message types.

8. Click the **check mark** icon to enter the information.

   You are returned to the main Change Distribution Model window.

9. Click **Save**.

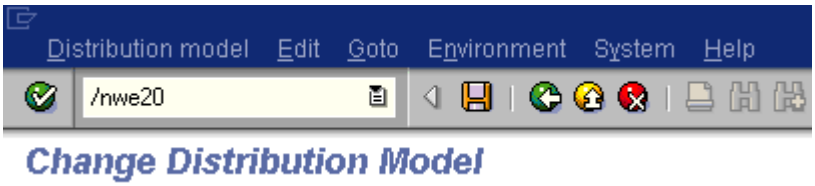
Defining a Partner Profile

Partner profiles are a prerequisite for data exchange. This involves defining who can exchange messages with the SAP system and using which port.

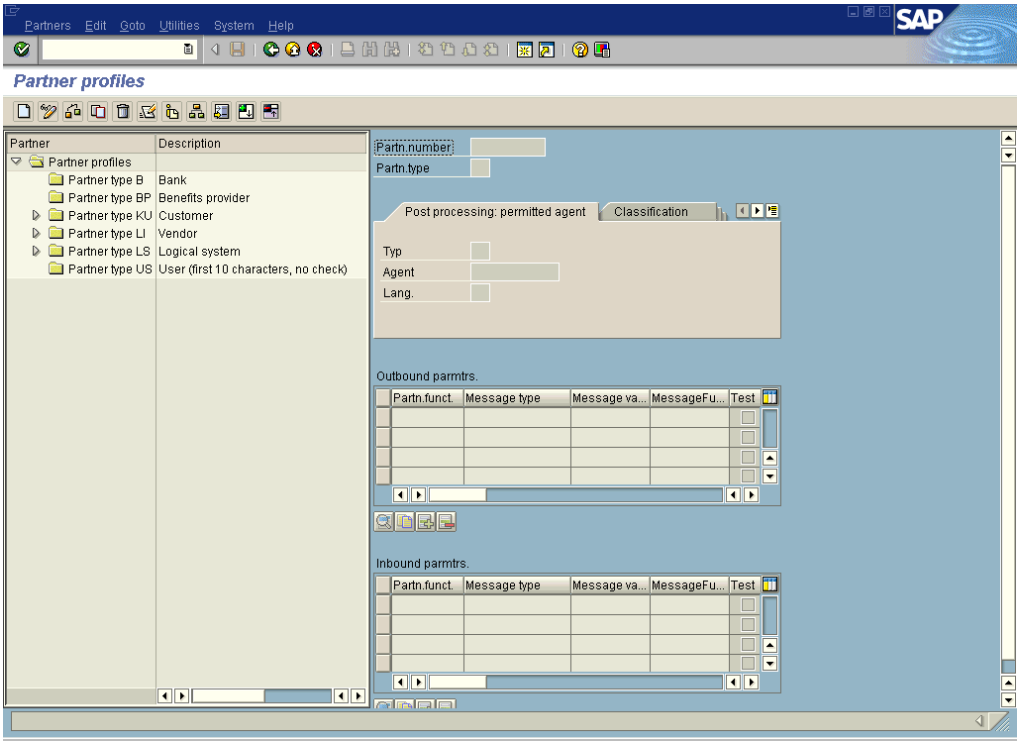
Defining a Partner Profile

To define a partner profile:

1. Run the **we20** transaction.

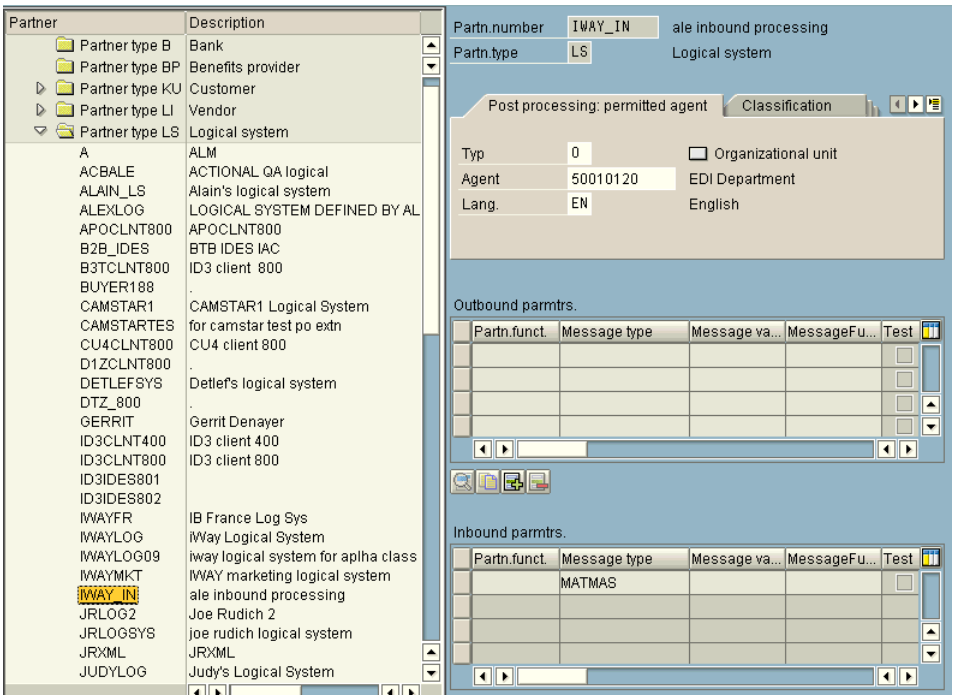


The Partner profiles window is displayed.



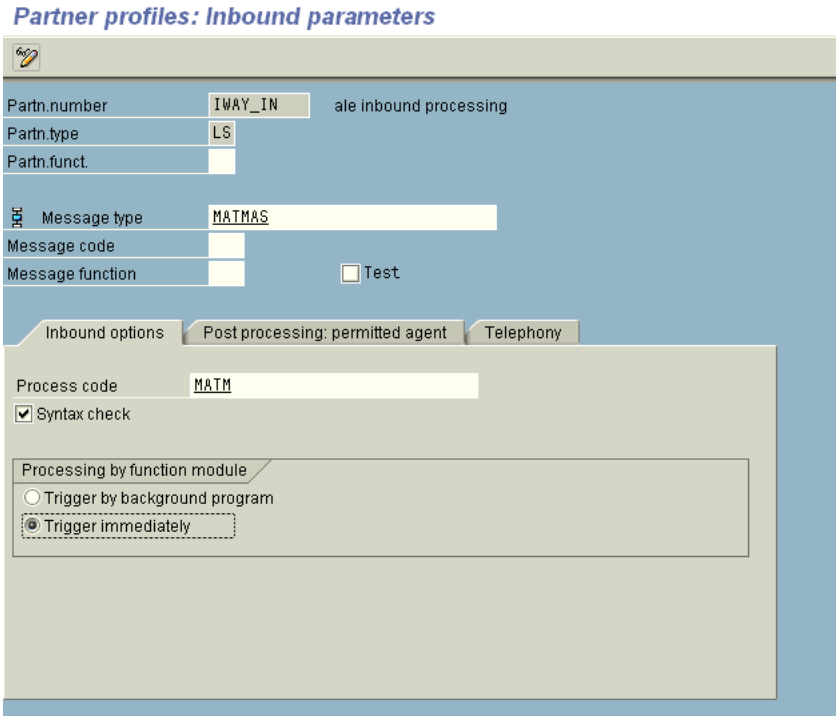
2. In the left pane, expand **Partner type LS** and select the logical system you configured from the list, for example, ORACLETDS.

In the right pane, the Partn.number field refers to the name of the logical system.



3. Click **Save**.
4. From the **Inbound** parameters table, click the **Create inbound parameter** icon.

   The Partner profiles: Inbound parameters window is displayed.



5. In the **Message type** field, enter the message type you want to use, for example, MATMAS.

You can click the icon to the right of each field to browse from a list of available message types.

The Inbound options tab is selected by default.

6. In the **Process code** field, enter the process code you want to use, for example, MATM.

   You can click the icon to the right of each field to browse from a list of available process codes.

7. In the **Processing by function module** area, select one of the following options:

   - Trigger by background program.

     In this case the adapter writes IDocs to the SAP database, which is processed immediately.

   - Trigger immediately.

     In this case, the adapter waits for the SAP system to process IDocs. This can take anywhere from 1 to 15 minutes.

8. Click **Save**.

Configuring SAP Outbound Processing

Event creation must be implemented by you or by SAP. An event is created from specific application programs (the event creator) and then published systemwide. Any number of receivers can respond to the event with their own response mechanisms. An event is usually defined as a component of an object type.

SAP pseudo events are not processed by the SAP Event manager, but are called from an ABAP program or Remote Function Call (using the Destination parameter).

Related Concepts and Terminology

The following topic lists and defines specific terminology related to SAP and SAP event handling.

Client and Server Programs

RFC programs for non-SAP systems can function as either the caller or the called program in an RFC communication. There are two types of RFC programs:

- RFC Client
- RFC Server

The RFC client is the instance that calls the RFC to run the function that is provided by an RFC server. The functions that can be called remotely are called RFC functions, and the functions provided by the RFC API are called RFC calls.

SAP Gateway

The SAP Gateway is a secure application server. No connections are accepted unless they have been preregistered previously from the SAP presentation Client. A server connection presents itself to the Gateway and exposes a Program Identifier. If the Program Identifier is found in the list of registered Program IDs, the Gateway server then offers a connection to the server, which "Accepts" a connection. This ProgramID is then linked with an RFC Destination within SAP, which enables SAP Function Modules and ALE documents (IDocs or BAPI IDocs) to be routed to the destination. The RFC Destination functions as a tag to mask the Program ID to SAP users.

An RFC server program can be registered with the SAP gateway and wait for incoming RFC call requests. An RFC server program registers itself under a Program ID at an SAP gateway and not for a specific SAP system.

In SAPGUI, the destination must be defined with transaction SM59, using connection type T and Register Mode. Moreover, this entry must contain information on the SAP gateway at which the RFC server program is registered.

Program IDs and Load Balancing

If the Gateway Server has a connection to a particular server instance and another server instance presents itself to the gateway, then the gateway offers the connection and then begins functioning in Load Balancing mode. Using a proprietary algorithm, the Gateway sends different messages to each server depending on demand and total processing time. This may cause unpredictable results when messages are validated by schema and application.

When configuring multiple events in the Oracle Application Server using a single SAP program ID, SAP load balances the event data. For example, if multiple remote function calls or BAPIs use the same program ID (for example, ORACLETDS) and multiple SAP listeners are configured with this progamID, then SAP sends one request to one listener and the next to another listener, and so on.

There is a load-balancing algorithm present in the SAP Gateway Server. This mechanism is proprietary to SAP application development and might work by comparing total throughput of the connection, the number of times in wait state, and so on. This means one connection might receive nine messages and a second connection might receive one message. If five of the nine messages are rejected for schema validation and the one message on the other connection is rejected for schema validation, you might suspect that you are missing SAP event handling messages.

Load balancing in server (inbound to adapter from SAP) situations is handled by connecting multiple instances of the adapter to the SAP system. The SAP system will then load balance the connections. You cannot tune this performance.

Load balancing in client (outbound from adapter to SAP) situations is handled only by the SAP application design. If your system supports a Message Server, then you can load balance in client situations. If you have only one application server, you cannot load balance except by application server tuning, such as maximum number of connections permitted or time of day limits on connections.

The SAP system default limit is 100 RFC (communication) or adapter users. Each user takes up more than 2 MB of memory on the application server of the SAP system, and more or less on the adapter depending on the workload.

Connection Pooling

A connection pool is a set of client connections to a specific destination. The pool may automatically create new connections to the specified remote system or return an already existing connection. It also provides methods to return a connection back to the pool when it is no longer needed.

A connection pool can check which connections are no longer in use and can be closed to save system resources. The time period after which the pool checks the connections as well as the time after which a connection will time out can be configured by the calling application.

A pool is always bound to one user ID and password, meaning that all connections taken from this pool will also use these credentials. An SAP connection is always bound to an SAP user ID and an SAP Client number.

If you log on with a pool size that is set to 1, no connection pool is created (1 userid – 1 process thread). If you log on with a pool size that is greater than 1, a pool is created with a size of n, which is the number you specified.

For more information about connection pooling, see the SAP JCO API documentation.

Registering Your Program ID in SAPGUI

To enable your SAP system to issue the following calls or interfaces to the SAP event adapter, you must register your program ID under an RFC destination.

- Remote Function Calls (RFC)
- Business Application Programming Interfaces (BAPI)
- Intermediate Documents (IDoc)

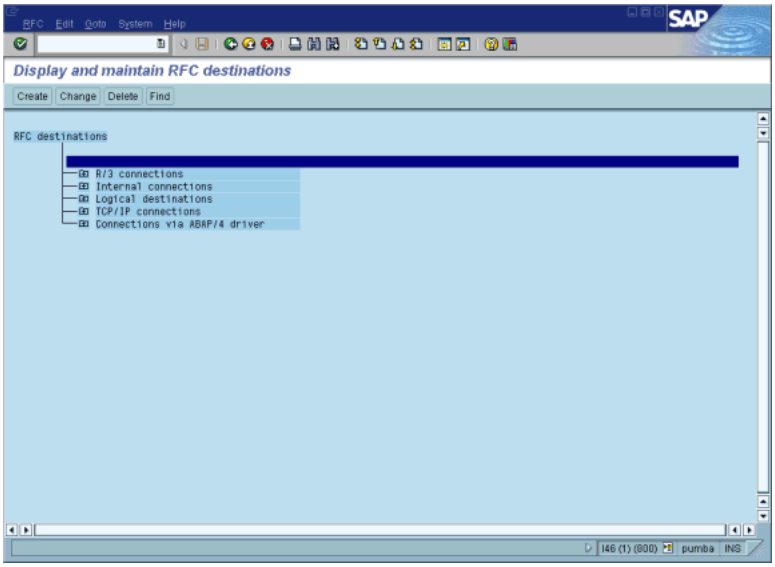
The RFC destination is a symbolic name (for example, ORACLETDS) that is used to direct events to a target system, masking the program ID. The Program ID is configured in both SAPGUI and the event adapter.

Registering Your Program ID

To register your program ID:

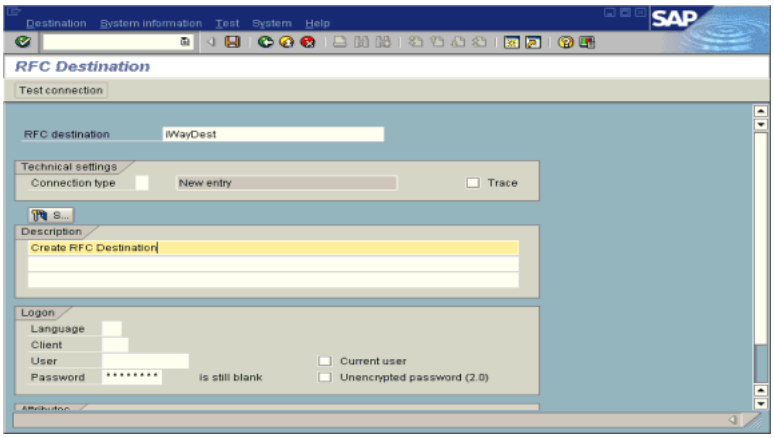
1. Launch the SAP GUI and log in to the SAP system.
2. Select **Tools**, **Administration**, **Network**, and then **RFC destination**.
3. Run the **SM59** transaction.

   The Display and maintain RFC destinations window is displayed.



4. Select **TCP/IP connections** and click **Create**.

The RFC Destination window is displayed.



Provide the following information:

a. In the **RFC destination** field, enter a name, for example, **ORACLETDS**.

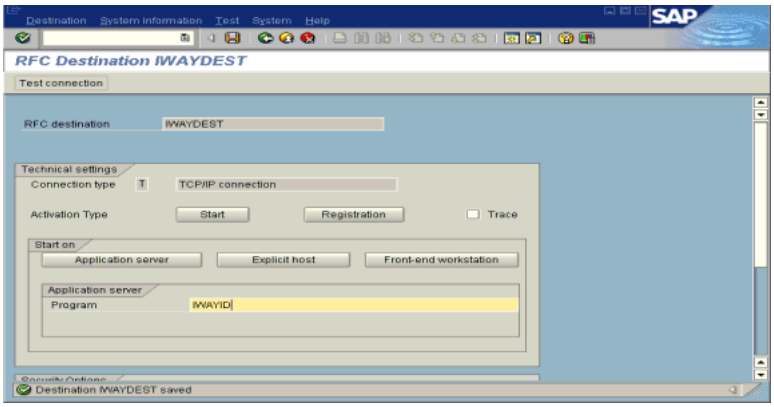
   The value you enter in this field is case sensitive.

b. In the **Connection type** field, enter **T** for destination type TCP/IP.

c. In the **Description** field, enter a brief description.

5. Click **Save** from the tool bar or select **Save** from the Destination menu.

   The RFC Destination ORACLETDS window is displayed.



Perform the following steps:

a. For the **Activation Type**, click **Registration**.

b. In the **Program** field, enter **ORACLETDS**.

6. Click **Save** from the tool bar or select **Save** from the Destination menu.
7. Ensure your event adapter is running.
8. Verify that the SAP system and OracleAS Adapter for SAP are communicating.
9. Click **TestConnection**.

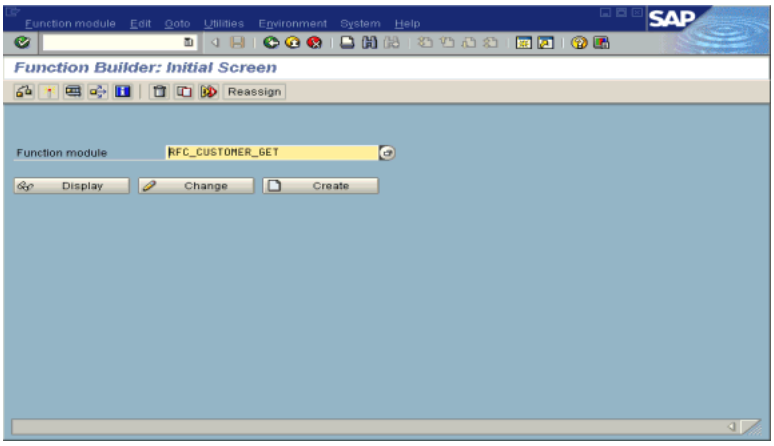
Testing the SAP Event Adapter

In the SAP Server, the SE37 transaction enables you to send an RFC (Remote Function Call) or a BAPI (Business Application Programming Interface) to any RFC destination. For more information on RFC destination, see Registering Your Program ID in SAPGUI on page A-11.

Testing the SAP Event Adapter by Sending an RFC or a BAPI Manually

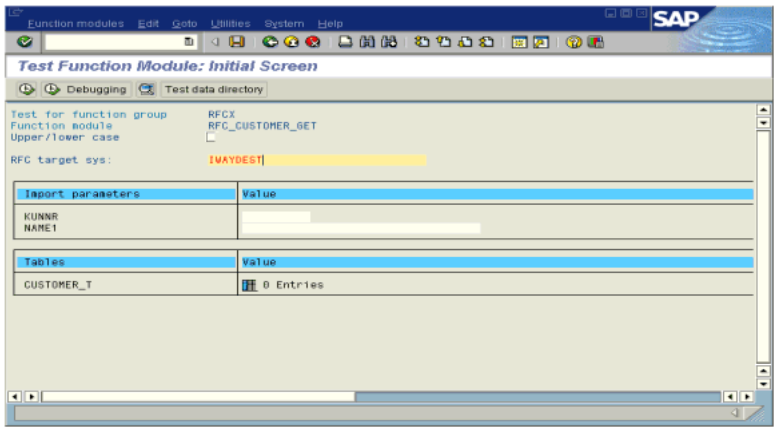
To test the SAP event adapter:

1. In the Function Builder, select a function module, for example, `RFC_CUSTOMER_GET`.



2. To choose single test, press **F8** and click the **Single Test** icon or choose **Function module**, select **Test** and then **Single Test**.
3. Enter an RFC target system, for example, ORACLETDS.
4. Enter input data for the particular RFC modules, for example, AB*.
5. To execute, press **F8**.

   The Test Function Module: Initial Screen window is displayed.



6. Enter data into the SAP GUI and click **Execute**.

   The function name and input data are transferred through RFC to create an XML document on the Oracle Application Server with the parameters input in SAPGUI.

Application Link Embedding Configuration for the Event Adapter

The SAP event adapter receives IDocs (Intermediate Documents) from SAP. To configure an SAP system to send IDocs to the SAP event adapter, use the ALE (Application Link Embedding) configuration to:

1. Register your program ID in SAPGUI.
2. Define a port.
3. Create a logical system.
4. Create a partner profile.
5. Create a distribution model for the partner and message type.
6. Test the SAP event adapter.

Defining a Port

A port identifies where to send messages. This port can be used only if an RFC destination was created previously.

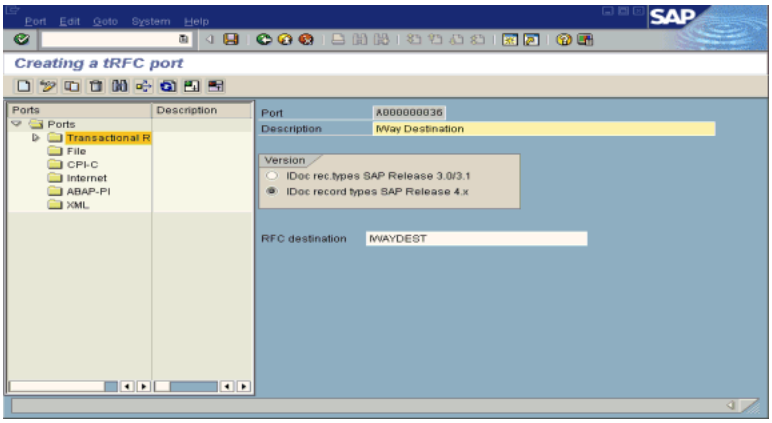
Defining a Port

To define a port:

1. In the ALE configuration, choose **Tools**, **Business Communications**, **IDocs Basis**, **IDoc**, and then **Port Definition**.

   You can also run the WE21 transaction.

   The Creating a tRFC port window is displayed.



2. In the left pane under **Ports**, select **Transactional RFC** and click **Create**.
3. Select **Generate port name**.

   The system generates the port name.
4. Enter the IDoc version you want to send through this port.
5. Click the destination you created, for example, ORACLETDS.
6. Save the session, making note of the system-generated RFC port.

Creating a Logical System

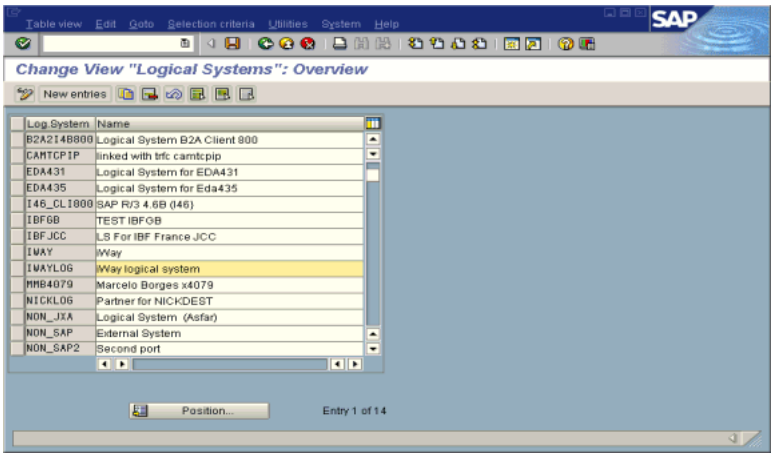
One type of partner is a logical system. A logical system manages one or more RFC destinations.

Creating a Logical System

To create a logical system called ORACLETDS:

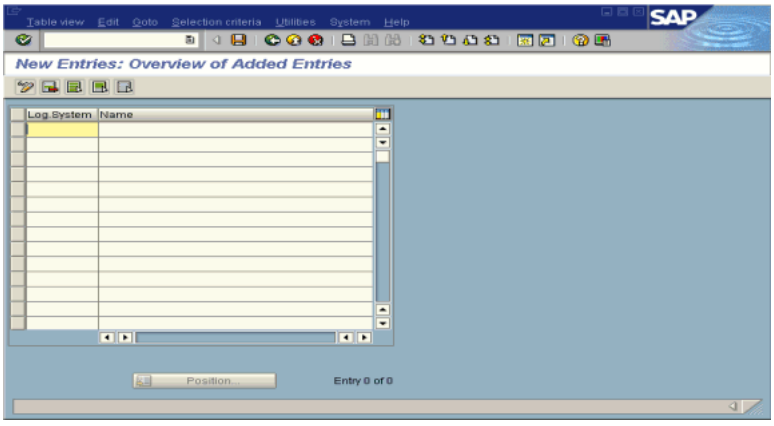
1. In the ALE configuration, enter the area menu selection **SALE** transaction.
2. Select **SAP Reference IMG**.
3. Expand the following nodes: **Basis Components**, **Application Link Enabling (ALE)**, **Sending and Receiving Systems**, **Logical Systems**, and **Define Logical System**.
4. Click the check mark beside **Define Logical System**.

   The Change View "Logical Systems": Overview window displays a list of logical systems and their names.



5. Click **New entries**.

   The New Entries: Overview of Added Entries window is displayed with Log.System and Name columns for new log system.



6. Type an entry for Log System, for example, ORACLETDS.
7. In the **Name** column, enter a name (description) for the partner profile.
8. Click **Save** to save the session.

Creating a Partner Profile

A partner profile is a definition of parameters for the electronic interchange of data with a trading partner using the IDoc interface.

To communicate with a partner using the IDoc interface, you must create a partner profile.

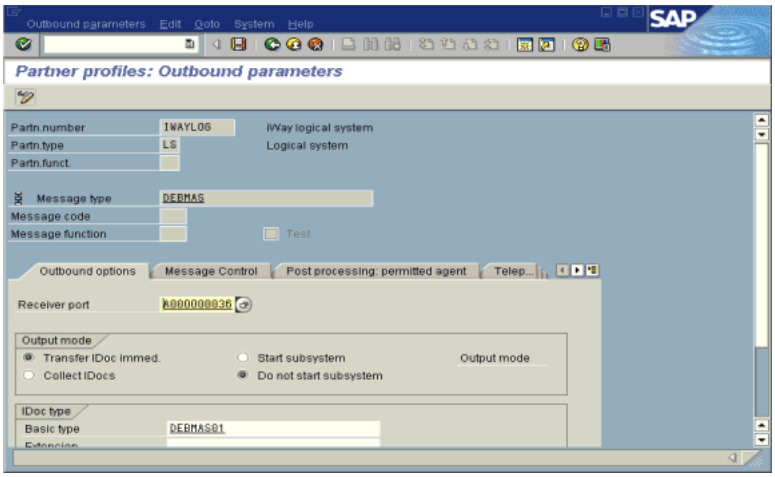
Creating a Partner Profile

To create a partner profile:

1. In SAP GUI, choose **Tools**, **Business Communication**, **IDoc Basis**, and **Partner profile**.

   You can also run the WE21 transaction.

   The Partner profiles: Outbound parameters window is displayed and shows fields for specifying details for the partner profile.



   Perform the following steps:

   a. Select Partner type **LS** (Logical system).

   b. Press **F5** (Create).

2. For Type, enter **USER**.

3. For Agent, enter the current user ID, or you may select another agent type.

4. Under the outbound parameter table control, select **Create outbound parameter**.

   Partner type is **LS,** and the Message type is **DEBMAS**, which is the IDoc document type.

5. Leave **Partn.funct** blank.

6. Click the **Outbound options** tab.

   Provide the following information:

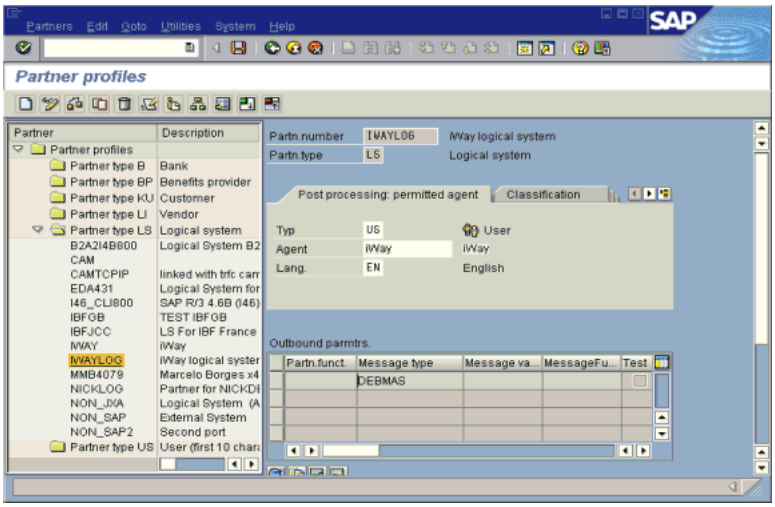
   a. Depending on your performance requirements, click **Transfer IDoc Immed** or **Collect IDocs**.

   b. For the IDoc, enter a message type, for example, DEBMAS.

   c. Enter a receiver port, for example, A000000036.

7. Click **Save** to save the session.

The Partner profiles summary window is displayed. It contains information for the logical system that you created.



Collected IDocs

When using collected IDocs on any platform during inbound processing (service mode), if the DOCNUM field does not have a unique document number for each IDoc, the system creates an IDoc for each header record in the collected IDoc file and duplicates the data for each IDoc.

Make sure the DOCNUM field is included in the EDI_DC40 structure and that each IDoc has a unique sequence number within the collected IDoc file.

Creating a Distribution Model for the Partner and Message Type

You must create a distribution model for the partner and message type you designated.

Creating a Distribution Model

To create a distribution model called ORAMOD:

1. In SAP GUI, choose **Tools**, **AcceleratedSAP**, **Customizing**, and then **Project Management**.

   You can also run the BD64 transaction.

   The Display Distribution Model window is displayed.

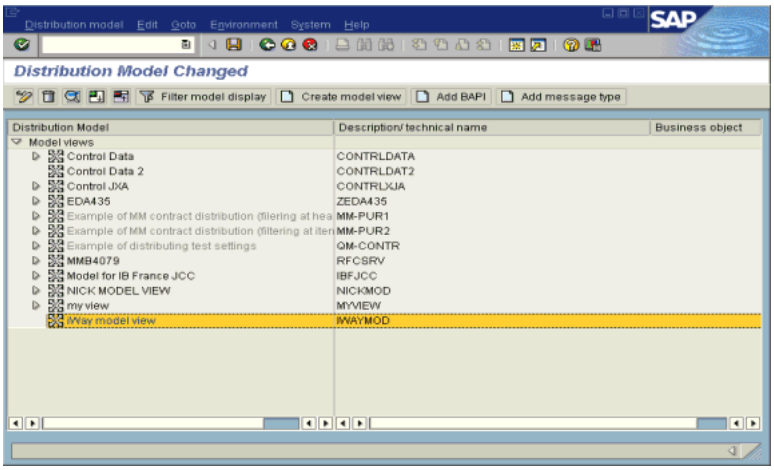
2. Select **Create model view**.

   If required, switch the processing mode to edit within Distribution Model/Switch Processing Mode.

3. Enter a short text string and a technical name for your new model view.

4. Click **Save**.

The Distribution Model Changed window is displayed, showing a tree structure of the distribution model.

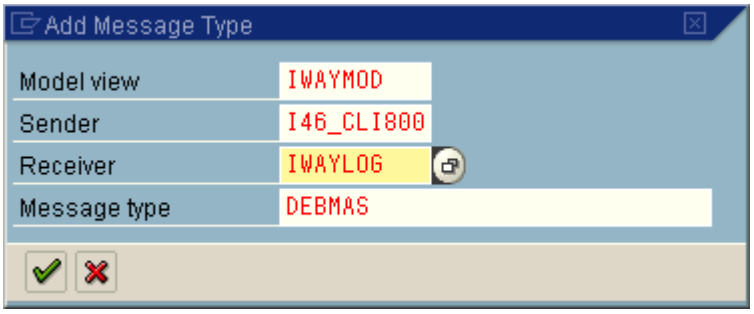


Perform the following steps:

a. In the Distribution Model tree, select a new model view.

b. On the right, select **Add message type**.

The Add Message Type box is displayed. It contains fields for specifying the sender and receiver of the message, as well as the message type.



Provide the following information:

a. In the **Sender** field, provide the sender that points to the SAP system, which sends the IDoc, for example, `I46_CLI800`.

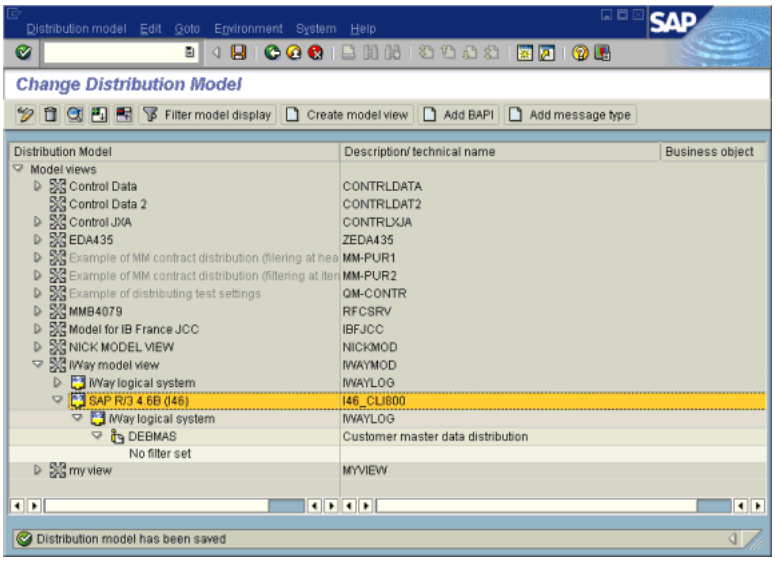
In this case, the sender is an SAP 4.6B system.

b. In the **Receiver** field, provide the logical system, for example, ORACLETDS.

c. In the **Message type** field, provide the type of IDoc, for example, DEBMAS.

5. Click the **check mark** icon.
6. Click **Save**.

The Change Distribution Model window displays the new model view to use to send message type, DEBMAS, from the I46_CLI800 SAP system to the ORACLETDS logical system.



You are now ready to test the connection to the logical system.

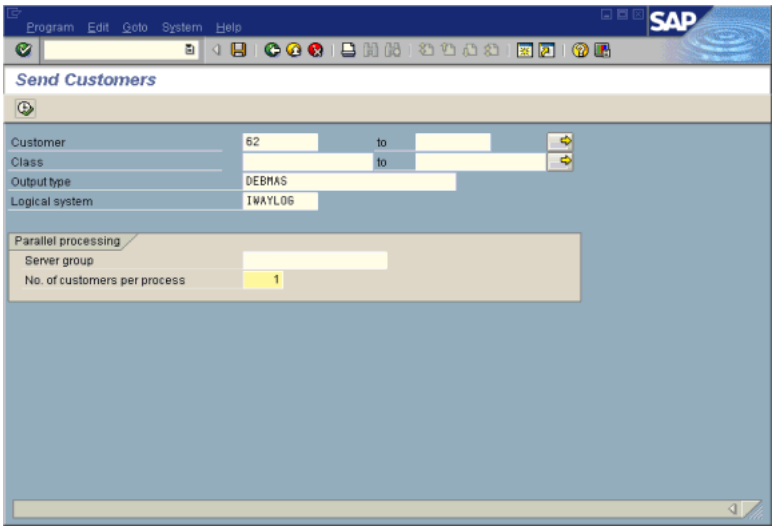
Testing the SAP ALE Configuration

In the SAP Server, the BD12 transaction enables you to send IDocs to any logical system, for example, to an event adapter.

Testing the SAP ALE Configuration

To test the SAP Application Link Embedding (ALE) configuration:

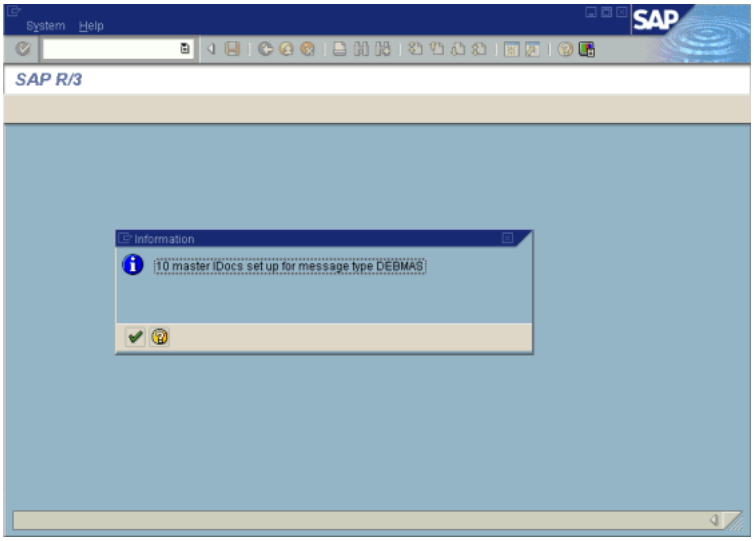
1. In the Send Customers window, enter the IDoc message type, for example, DEBMAS in the **Output type** field.



2. In the **Logical system** field, enter the logical system, for example, ORACLETDS.
3. Click **Run**.

The SAP event adapter receives the IDoc in XML format. No response is expected from the event adapter.

A confirmation window is displayed.



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

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X