# Oracle® Application Server Adapter

for SAP R/3 User's Guide

10*g* Release 2 (10.1.2)

**Part No. B14061-01**

November 2004

ORACLE®

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

Part No. B14061-01



Primary Author: Stefan Kostial

Contributing Author: Meera Srinivasan

Contributor: Arvind Jain

# Contents

## 3 Deployment and Integration



## 4 Examples



## 5 Troubleshooting and Error Messages



## 6 Advanced Topics



## A Configuring SAP for Inbound and Outbound Processing

# Send Us Your Comments

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

**Part No. B14061-01**

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: appserverdocs_us@oracle.com
- FAX: (650) 506-7375. Attn: Oracle Application Server Documentation Manager
- Postal service:

  Oracle Corporation
  Oracle Application Server Documentation
  500 Oracle Parkway, Mailstop 1op6
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, please contact your local Oracle Support Services.

# Preface

This guide explains how to use the Oracle Application Server Adapter for SAP R/3 to access SAP BAPIs, RFCs, and IDocs. In this guide you will learn how to define a delivery channel for SAP and add an interaction to generate native events, which are XML instances defined by XML schemas.

This preface contains these topics:



## 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. 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 additional information, visit the Oracle Accessibility Program Web site at

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

**Accessibility of Code Examples in Documentation** JAWS, a Windows screen reader, 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, JAWS 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.

## Intended Audience

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

- Create delivery channels and interactions with an SAP system
- Maintain applications

To use this document, you need some knowledge of SAP BAPIs, RFCs, and IDocs.

## Organization

This document contains:

### Chapter 1, "Introduction to Oracle Application Server Adapter for SAP R/3"

This chapter describes the Oracle Application Server Adapter for SAP R/3 and the hardware and software requirements. It also provides instructions on adding custom function modules into an R/3 system. The custom modules enhance browsing performance during the selection of interactions from an R/3 system.

### Chapter 2, "Adapter Configuration Using Application Explorer"

This chapter provides instructions for starting Application Explorer, for creating projects, establishing a connection to SAP, and creating schemas and Web services. It also explains how to configure the Event Adapter.

### Chapter 3, "Deployment and Integration"

This chapter describes Oracle Containers for J2EE (OC4J) deployment and integration with Oracle Application Server Integration InterConnect.

### Chapter 4, "Examples"

This chapter contains examples.

### Chapter 5, "Troubleshooting and Error Messages"

This chapter describes how to troubleshoot and interpret error messages.

### Chapter 6, "Advanced Topics"

This chapter includes advanced topics for expert users.

### Appendix A, "Configuring SAP for Inbound and Outbound Processing"

This appendix describes how to enable inbound and outbound SAP processing.

## Related Documentation

For more information, see these Oracle resources:

- *Oracle Application Server Adapter Concepts*
- *Oracle Application Server Adapters Installation Guide*
- *Oracle Application Server Administrator's Guide*
- *Oracle Application Server Concepts*
- *Oracle Application Server Containers for J2EE User's Guide*

- *Oracle Application Server Integration InterConnect User's 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 username 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 various conventions in text to help you more quickly identify special terms. The following table describes those conventions and 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*<br>Ensure that the recovery catalog and target database do *not* reside on the same disk. |
| `UPPERCASE monospace (fixed-width) font` | Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles. | You can specify this clause only for a `NUMBER` column.<br>You can back up the database by using the `BACKUP` command.<br>Query the `TABLE_NAME` column in the `USER_TABLES` data dictionary view.<br>Use the `DBMS_STATS.GENERATE_STATS` procedure. |

| Convention | Meaning | Example |
|---|---|---|
| lowercase monospace (fixed-width) font | Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values.<br>**Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown. | Enter `sqlplus` to open SQL*Plus.<br>The password is specified in the `orapwd` file.<br>Back up the datafiles and control files in the `/disk1/oracle/dbs` directory.<br>The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table.<br>Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true`.<br>Connect as `oe` user.<br>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*.<br>Run U*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 |
|---|---|---|
| [ ] | Brackets enclose one or more optional items. Do not enter the brackets. | `DECIMAL (`*digits* `[ ,` *precision* `])` |
| { } | Braces enclose two or more items, one of which is required. Do not enter the braces. | `{ENABLE \| DISABLE}` |
| \| | A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar. | `{ENABLE \| DISABLE}`<br>`[COMPRESS \| NOCOMPRESS]` |
| ... | Horizontal ellipsis points indicate either:<br>■ That we have omitted parts of the code that are not directly related to the example<br>■ That you can repeat a portion of the code | `CREATE TABLE ... AS` *subquery*`;`<br>`SELECT` *col1*`, ` *col2*`, ... , ` *coln* `FROM employees;` |

| Convention | Meaning | Example |
|---|---|---|
| . . . | Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example. | SQL> SELECT NAME FROM V$DATAFILE;<br>NAME<br>----------<br>/fsl/dbs/tbs_01.dbf<br>/fs1/dbs/tbs_02.dbf<br>.<br>.<br>.<br>/fsl/dbs/tbs_09.dbf<br>9 rows selected. |
| Other notation | You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown. | acctbal NUMBER(11,2);<br>acct CONSTANT NUMBER(4) := 3; |
| *Italics* | Italicized text indicates placeholders or variables for which you must supply particular values. | CONNECT SYSTEM/*system_password*<br>DB_NAME = *database_name* |
| 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. However, because these terms are not case sensitive, you can enter them in lowercase. | SELECT last_name, employee_id FROM employees;<br>SELECT * FROM USER_TABLES;<br>DROP TABLE hr.employees; |
| lowercase | Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.<br>**Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown. | SELECT last_name, employee_id FROM employees;<br>sqlplus hr/hr<br>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 |
|---|---|---|
| Choose Start > | How to start a program. | To start the Database Configuration Assistant, choose Start > Programs > Oracle - *HOME_NAME* > Configuration and Migration Tools > 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 file name begins with \\, then Windows assumes it uses the Universal Naming Convention. | c:\winnt"\"system32 is the same as C:\WINNT\SYSTEM32 |

| Convention | Meaning | Example |
|---|---|---|
| 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 scott/tiger TABLES=emp QUERY=\"WHERE job='SALESMAN' and sal<1600\"<br>C:\>imp SYSTEM/*password* FROMUSER=scott TABLES=(emp, dept) |
| *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. For Windows, the default location was C:\orant.<br><br>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. 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\orann, where *nn* is the latest release number. The Oracle home directory is located directly under *ORACLE_BASE*.<br><br>All directory path examples in this guide follow OFA conventions.<br><br>Refer to *Oracle Database Platform Guide for Windows* for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories. | Go to the ORACLE_BASE\ORACLE_HOME\rdbms\admin directory. |

# 1

# Introduction to Oracle Application Server Adapter for SAP R/3

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



## Adapter Features

The OracleAS Adapter for SAP provides a means to exchange real-time business data between SAP systems and other application, database, or external business partner systems. The **adapter** enables external applications for inbound and outbound processing with SAP. The OracleAS Adapter for SAP can be deployed as a JCA 1.0 resource adapter. This deployment is referred to as the OracleAS Adapter JCA. It can also be deployed as a Web services servlet and as such is referred to as the Oracle Application Server Adapter Business Services Engine (BSE).

The adapter uses XML messages to enable non-SAP applications to communicate and exchange transactions with SAP using services and events.

- Services: Applications use this capability to invoke an SAP business object or business operation.
- Events: Applications use this capability to access SAP data only when an SAP business event occurs.

To support event functionality, 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 only port supported in this release is Remote Method Invocation (RMI). It is used for integration with Oracle Application Server Integration InterConnect.

- 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.

The OracleAS Adapter for SAP provides:

- Support for bidirectional message interactions.
- Oracle Application Server 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 JCA 1.0 resource adapter.
- Web services for the Business Services Engine.

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

## SAP Certification

SAP has certified the OracleAS Adapter for SAP for use with all versions of SAP, including mySAP.com solutions.

The adapter provides state-of-the-art middleware solutions for SAP Basis and SAP Web application server-based systems. This adapter has achieved 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, Oracle Application Server Adapter for SAP R/3 expedites the conversion, import, and export of critical intermediate documents (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 ALE (Application Link Enabling) 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 pulled or pushed into XML formats to or from SAP solution-based systems over the Internet.

## Supported Platforms

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

- SAP Web Application Server Version 6.2 and higher
- SAP R/3 4.0 and higher
- SAP Enterprise R/3 4.7
- All or portions of: MySAP.com technology solutions (SAP BW, SAP APO, SAP CRM, SAP SRM, SAP EBP, SAP SEM, SAP WP, SAP KW)

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

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

- Windows 2000/2003/XP
- Solaris Version 2.8, 2.9, and higher
- Linux x86 Redhat Advanced Server Version 2.1 and higher, Suse Version 8.1 and higher, United Linux Version 1.0 and higher
- IBM AIX Version 5.1 and higher
- HP-UX Version 11i and higher
- HP Tru64 Version 5.1a and higher

# SAP Concepts

The OracleAS Adapter for SAP is designed to provide standard access to SAP business objects such as Remote Function Call (RFC) modules, BAPIs (Business Application Programming Interfaces), and IDocs (Intermediate Documents) 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 invokes its remote requests. These objects work in the following ways:

- **Business Application Programming Interfaces (BAPI)** 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.
- **Remote Function Call (RFC) Modules** are SAP application interfaces that enable clients to invoke 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 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 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 your SAP Service Marketplace.

- **Intermediate Documents (IDocs)** are the "logical messages" that correspond to different business processes. They 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.

## Integration with SAP

You can use the OracleAS Adapter for SAP to invoke 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 SAP system can be configured to send an IDoc or RFC out to a logical system when a certain event occurs, in this case, to the adapter. 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, the OracleAS Adapter for SAP can send requests to SAP using the BAPI, RFC, or IDoc interfaces.

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

- Eliminating 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.

## Adapter Architecture

The OracleAS Adapter for SAP works in conjunction with the following components:

- Application Explorer
- Business Services Engine (BSE)
- Enterprise Connector for J2EE Connector Architecture (JCA)

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 the Business Services Engine or with the Enterprise Connector for J2EE Connector Architecture (JCA). When working in a JCA 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 Architecture

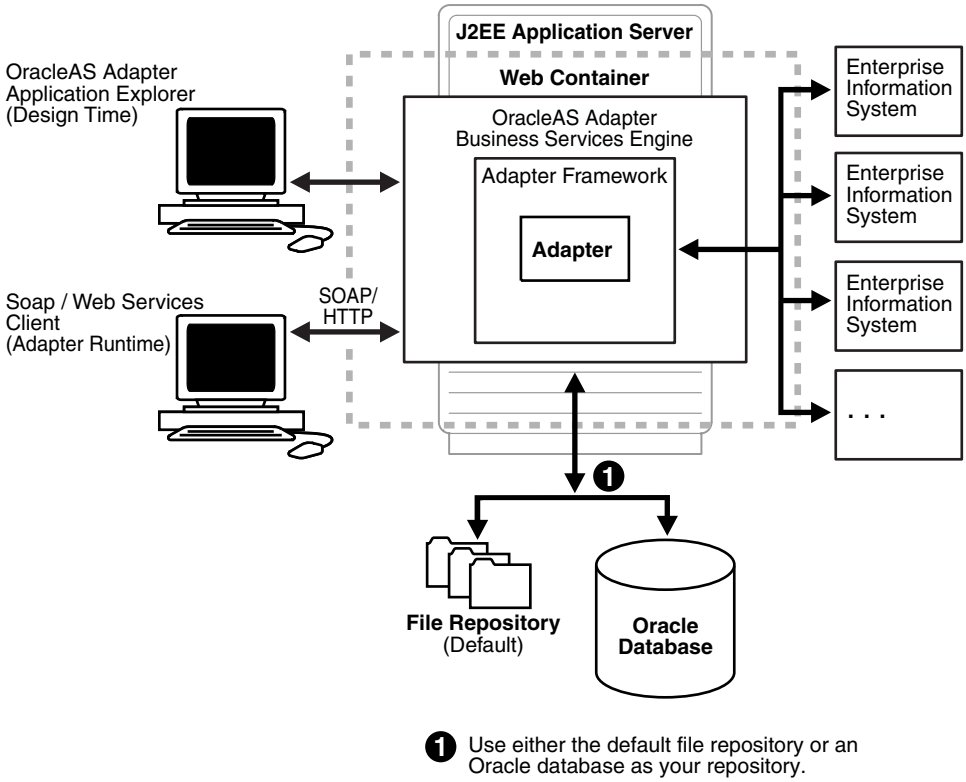
Figure 1–1 shows the generic architecture for the Oracle Application Server Adapter Business Services Engine (BSE) for packaged applications. The OracleAS Adapter Application Explorer works in conjunction with the Oracle Application Server Adapter Business Services Engine (BSE), as deployed to the OC4J (Oracle Containers for J2EE) container of the 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 consuming requests from clients, interacting with the EIS, and sending responses from the EIS back to clients.

BSE uses RMI as a protocol to send the events received from the EIS (SAP) application to OracleAS Integration InterConnect. BSE supports both a file-based and an Oracle database repository. The BSE repository stores the EIS connection information and the WSDL for adapter services. A single instance of BSE can connect to multiple EIS applications.

*Figure 1–1 Oracle Application Server Adapter Business Services Architecture*

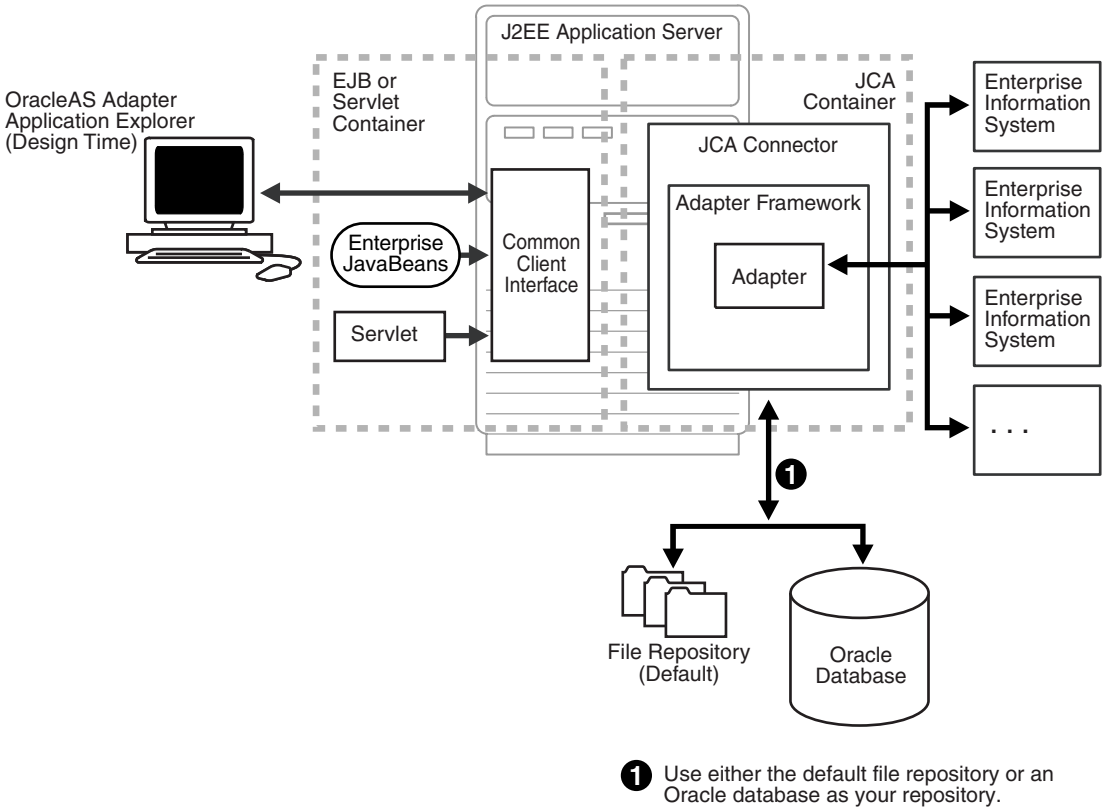



### Oracle Application Server Adapter Generic JCA Architecture

Figure 1–2 shows the generic architecture for the OracleAS Adapter JCA for packaged applications. This is a pure JCA 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 JCA 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-ra.xml` file.

***Figure 1–2 Oracle Application Server Adapter Generic JCA Architecture***




**See Also:**

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

# 2

# Adapter Configuration Using Application Explorer

This chapter describes how to use 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

The server must be started where Application Explorer is deployed.

To start Application Explorer:

1. Ensure the server is started where Application Explorer is deployed.
2. On Windows, invoke `iaexplorer.exe`, found under *`OracleAS_home`*`\adapters\application\tools` or on UNIX, invoke the iwae script, `iwae.sh`, found under *`OracleAS_home`*`/adapters/application/tools`.

**OracleAS_home**
Is the directory where the Oracle Application Server is installed.

Application Explorer opens. You are ready to define new targets to your SAP system.

## Configuring Settings for BSE or JCA

You need not configure the OracleAS Adapter Business Services Engine (BSE) for a file-based repository because it is configured during the Oracle installation. You also need not configure the OracleAS Adapter JCA because the `ra.xml` file is configured automatically during installation.

## Configuring the OracleAS Adapter Business Services Engine

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.

To configure BSE:

1. Open the following page in your browser:

   ```
   http://hostname:port/ibse
   ```

   **hostname**
   Is the hostname of the Oracle Application Server.

   **port**
   Is the HTTP port for the Oracle Application Server.

   For example,

   ```
   http://localhost:7777/ibse
   ```

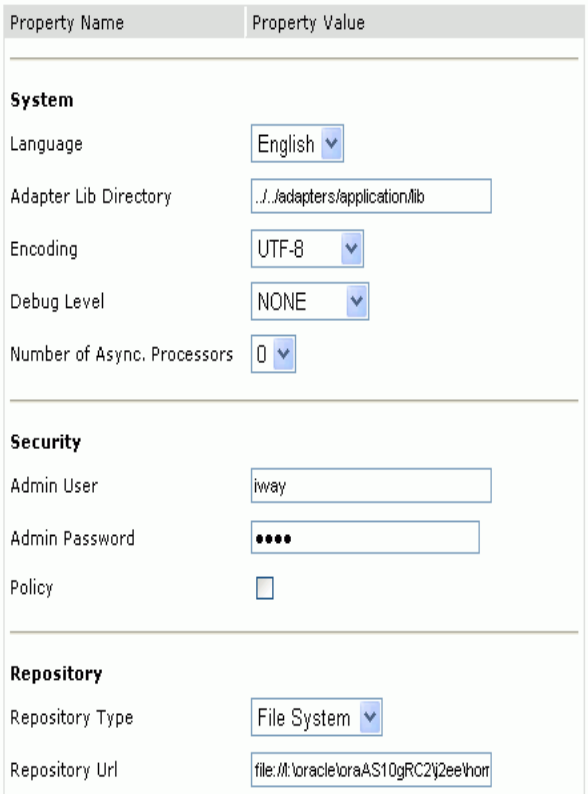
   **Note:** The first time you access this page, it may take time to load.

2. When prompted, log on.

   When first installed, the user ID and the password are:

   - User name: iway
   - Password: iway

The BSE configuration page opens.



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

   ```
   OracleAS_home\adapters\application\lib
   ```

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

4. For security purposes, type 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 Business Services (BSE) system settings:

1. Open the **BSE configuration** page by entering the following URL:

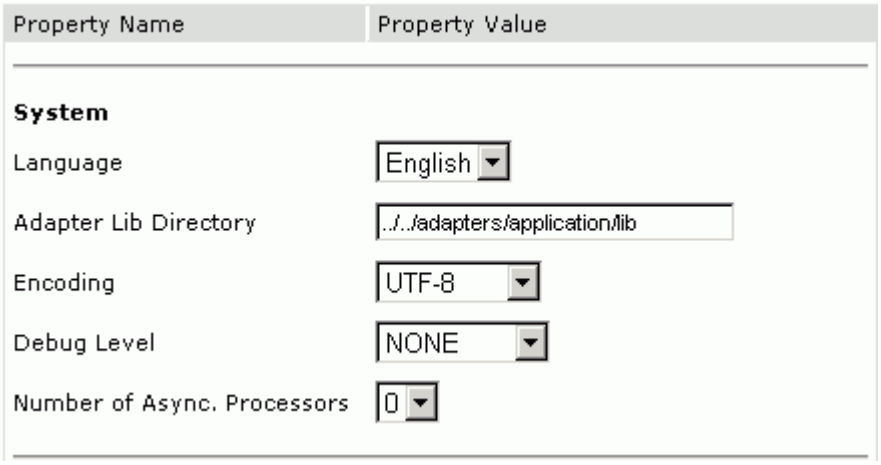
   ```
   http://hostname:port/ibse/IBSEConfig
   ```

**hostname**
Is the machine where BSE is installed.

**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 opens.

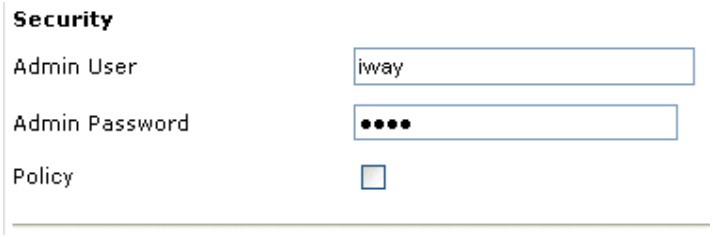



2. Configure the system settings.

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

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

The following image shows the Security pane.

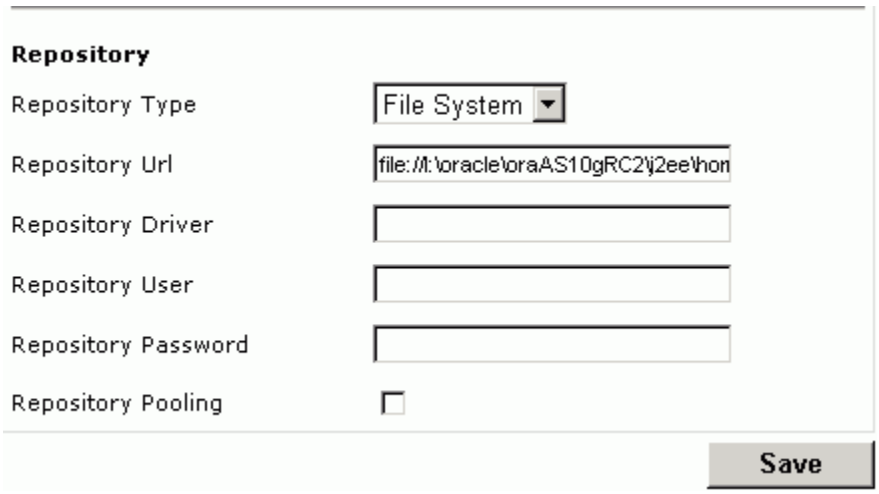



3. Configure the security settings.

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

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

The following image shows all of the fields and the check box 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 2-5 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:<br>Oracle<br>File |
| Repository URL | Type 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 | Type the user ID to use when opening a connection to the database. |
| Repository Password | Type 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.

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 execute 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, see 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. Type the following command:

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

# Creating a Repository Project

Before you use Application Explorer with the OracleAS Adapter for SAP, you must create a repository project. You can create two kinds of repository projects, Web services and JCA, depending on the container to which the adapter is deployed.

At 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 JCA repository is created for the default ManagedConnectionFactory. The name of this project is jca_sample.

Web services and the OracleAS Adapter Business Services Engine (BSE) refer to the same type of deployment. See "Adapter Features" on page 1-1 for more information.

## Creating a Web Service Project for the Web Service Adapter

To create a Web service project for the Web service adapter using Application Explorer, you must first define a new configuration.

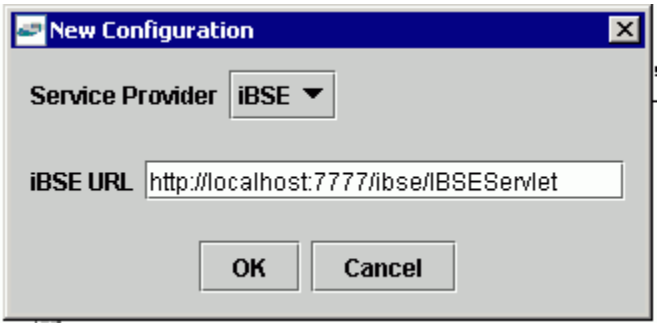
### Defining a New Configuration for BSE

To define a new configuration for BSE:

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

The New Configuration dialog box opens.

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




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

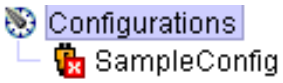
   `http://`*`hostname`*`:`*`port`*`/ibse/IBSEServlet`

**hostname**
Is the machine where your application server resides.

**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.



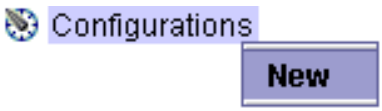

The Web service repository project file is stored in *`OracleAS_home`*`\j2ee\home\applications\ws-app-adapter\ibse.`

## Creating a Repository Project for OracleAS Adapter JCA

To create a repository project for the OracleAS Adapter JCA using Application Explorer, you must first define a new configuration.

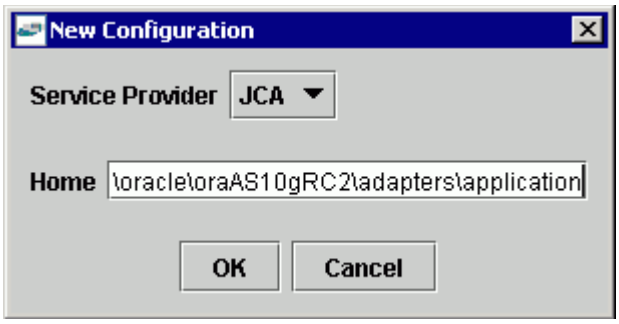
### Defining a New Configuration for JCA

To define a new configuration for JCA:




1. Right-click **Configurations** and select **New.**
2. In the dialog box, type a name for the new configuration, for example, SampleConfig, and click **OK.**

   The following dialog box opens.

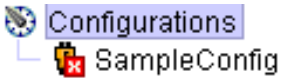



3. From the **Service Provider** list, select **JCA.**
4. In the **Home** field, enter a path to your JCA configuration directory where the repository, schemas, and other information is stored, for example:

   *OracleAS_home*\adapters\application

5. Click **OK.**

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




The OracleAS Adapter JCA repository project file is stored in *OracleAS_home*\adapters\application\config*configuration_name*

*configuration_name*
Is the name of the configuration you created; for example, SampleConfig.

## Connecting to a New 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).


- 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 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 are now ready to define new targets to SAP.

# Establishing a Connection (Target) for SAP

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

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.

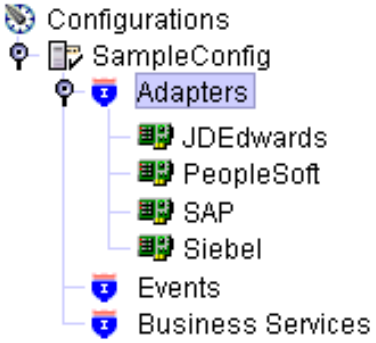
## Connecting to SAP

To connect to SAP for the first time, you must define a new target.

### Defining a Target for SAP

To define a target:

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




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


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

   The Add Target dialog box opens.

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

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

   c. From the **Target Type** list, select a target type.

3. Click **OK.**

The Application Server dialog box appears.

The following tabs are available:

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

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

***Table 2–1 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. |

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

***Table 2–2 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. |
| SAP Trace | Select this option to enable traces. |

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

***Table 2–3 Advanced Tab Parameters***

| Target Parameter | Description |
|---|---|
| Connection pool size | Enter the number of connections you want to make available to SAP. |
| Connection pool name | Enter the name of your SAP connection pool. |
| BAPI Exception Handling | Select **Throws Exception** or **Creates Error Document** from the list in the event of a BAPI exception. |
| Commit with wait | This option is disabled by default. |

7. For the **Security** tab (optional), type the appropriate information for your SAP target based on the information in the following table.

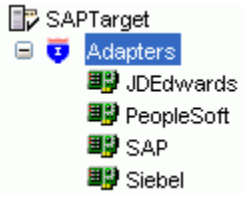
*Table 2–4 Security Tab Parameters*

| Target Parameter | Description |
|---|---|
| Logon ticket (SSO2) | If you are using a Secure Network Communications (SNC) adapter with SAP, enter the name of the SSO2 logon ticket you are using. |
| Logon ticket (X.509) | If you are using an SNC adapter with SAP, enter the name of the X.509 logon ticket you are using. |
| 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. |
| SNC library path | Enter the path to the SNC library. |

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

8. When you have supplied all the required information for your target, click **Finish.**

   After the extraction finishes, the new target, **SAPTarget**, appears under the sap node.




   See "Creating an XML Schema" on page 2-13 for information on how to create schemas for the adapter.

## Connecting to a Defined SAP Target

To connect to a target:

1. Expand the **Adapters** node.
2. Expand the **SAP** node.
3. Under the **SAP** node, click the target name, for example, SAPTarget.

   The Connection dialog box opens, populated with values you entered for the connection parameters.
4. Verify your connection parameters. If required, provide the password.
5. 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.
- Delete a connection that is no longer required.

### Disconnecting from a Connection to SAP

1. Expand the **Service 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.

### Editing a Target

To edit a target:

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

   The Edit pane opens on the right.
3. Modify the target information.
4. Click **OK.**

### Deleting a Connection to SAP

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

   The node disappears from the list of available connections.

# Viewing Application System Objects

**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 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 your SAP Service Marketplace.

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

# Creating an XML Schema

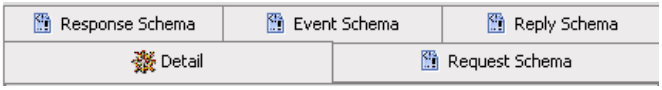
After you explore the SAP business function library and select an object, you can use Application Explorer to create the XML request schema and the XML response schema for that function.

## Creating a Request and a Response Schema

The following procedure explains how to create request and response schemas for an SAP business function. Application Explorer enables you to create XML schemas for this function.

1. Connect to an SAP target as described in "Connecting to a Defined SAP Target" on page 2-11.
2. Expand the **Business Object Repository** node.
3. Click the icon to the left of the **Financial Accounting** group.
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.

# Creating and Testing a Web Service or Business Service

You can generate a Web service (also known as a **business service**). You can explore the business function repository and generate Web services 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 (JCA) implementation of the adapter, Web services are not available. When the adapter is deployed to use the OracleAS Adapter JCA, the Common Client Interface provides integration services using the adapter.

### Creating a Web Service or Business Service

To create a Web service for an SAP business function:

1. Expand the **sap** node and then 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 Business Service.**

   The Create Web Service dialog box opens.

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

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

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

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

   d. Select one of the available licenses.
4. Click **Next.**

   The License and Method dialog box opens.

   a. In the **License** 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, type a descriptive name for the method.

   c. In the **Description** field, type 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 or Business 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. If you are not on the Business Services node of Application Explorer, click the node to access Web services.
2. If it is not expanded, expand the list of Web services under **Business Services.**
3. Expand the **Services** node.
4. Select the name of the business service you want to test.

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

5. 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.

6. Enter the appropriate input.

7. Click **Invoke.**

   Application Explorer displays the results.

# 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.

- Port

  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 an Event Port" on page 2-15 for more information.

- Channel

  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 2-18 for more information.

---

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

---

## Creating an Event Port

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

### Creating an Event Port From the Adapters Node

You can bypass the Events node and create an event port directly from the Adapters node.

To create an event port from the Adapters node:

1. Select the SAP object for which you want to create an event port.

2. Right-click the node and select **Add Port.**

   The Add Port dialog box opens.

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

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

   c. Type the disposition url.

3. Click **OK.**

   See "Creating an Event Port From the Events Node" on page 2-16 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 JCA 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 2-16 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 opens.

   a. Type 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]
   ```

   d. From the **Disposition** protocol list, select **RMI.**

   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. |

| Parameter | Description |
|---|---|
| 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 are ready to associate the event port with a channel. See "Creating a Channel Using Application Explorer" on page 2-18 for more information.

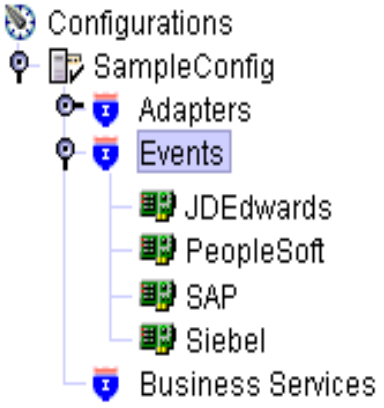
## Modifying an Event Port

The following procedures describe how to edit and delete an RMI event port using Application Explorer.

### Editing an Event Port

To edit an event port using Application Explorer:

1. Expand the **Events** node.




2. Expand the **SAP** node.
3. Right-click the event port you want to edit and select **Edit.**

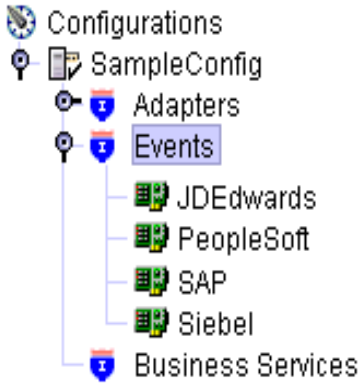
   The Edit Port pane opens.

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

### Deleting an Event Port

To delete an event port using Application Explorer:

1. Expand the **Events** node.




2. Expand the **SAP** node.
3. Right-click the event port you want to delete and select **Delete.**

   A confirmation dialog box opens.
4. To delete the event port you selected, click **OK.**

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

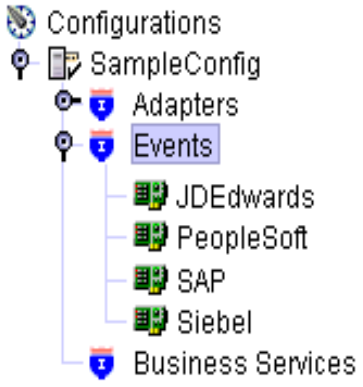
## Creating a Channel Using Application Explorer

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

### 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 opens.

   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 Message 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 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. |
| 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, type 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), type the appropriate information for your SAP channel based on the information in the following table.

***Table 2–7 Advanced Tab Parameters***

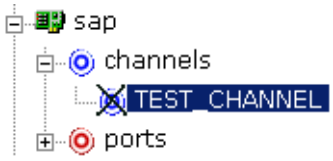
| Target Parameter | Description |
|---|---|
| IDOC Format | Select an IDoc type from the list. |
| User Defined Function Modules | Enter the path to the user-defined function module you created. |
| SAP trace | Select this check box if you want to enable SAP traces for troubleshooting purposes. |

*Table 2–7 (Cont.) Advanced Tab Parameters*

| Target Parameter | Description |
|---|---|
| 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. |

8. Click **OK.**

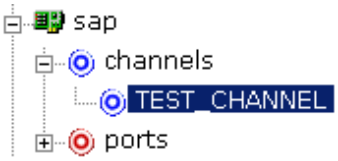
   The channel appears under the channels node in the left pane.




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

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

   The channel you created becomes active.




   The X that was over the icon in the left pane disappears.

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

### Editing a Channel

To edit a channel:

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

   The Edit channels pane opens.

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 opens.

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

   The channel disappears from the list in the left pane.

# 3

# Deployment and Integration

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

This chapter discusses the following topics:

- Oracle OC4J Integration
- OracleAS Adapter BSE Integration with OracleAS Integration InterConnect

**See Also:**

- *Oracle Application Server Integration InterConnect User's Guide*
- *Oracle Application Server Containers for J2EE User's Guide*

## Oracle OC4J Integration

The following topic shows the basic commands for using CCI with packaged application adapters.

**See Also:**

- "OC4J Containers" in *Oracle Application Server Adapter Concepts*
- "Deployment and Integration through J2CA" in *Oracle Application Server Adapter Concepts*

### Application Development Using the CCI API

The following example shows the code structure for using CCI with packaged application adapters. The code sample is shown in five 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. |
| 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 JCA supports only basic security mapping. The DEBUG log level provides detaild information on the mapping behavior. It functions as follows:

- If the userName and password are not set, and no security is provided by the application server, the OracleAS Adapter JCA will still let it pass and rely on the adapter configuration security information.
- If userName and password are set, these values will overwrite the adapter configuration. The OracleAS Adapter JCA 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 invoke 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 invoked, in this case SAP, and the name of a target defined in Application Explorer. See "Complete Code Sample" on page 3-3 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 Execute Interaction

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

A standard JCA 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());
```

*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.
 *
 * Author: Marcelo Borges
 * Date: August, 2004
 */
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()

}
```

### Creating a Managed Connection Factory

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 are defined 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, the OracleAS Adapter for SAP has a preconfigured jca_sample configuration in the 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, "Adapter Configuration Using Application Explorer" 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, "Adapter Configuration Using Application Explorer" 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, "Adapter Configuration Using Application Explorer" for more information. |
| loglevel | It overwrites the level set by the ManagedConnectionFactory property. |

### Creating Multiple Managed Connection Factories

To establish Multiple Managed Connection Factories, you must edit the `OC4J-ra.xml` file to add the required information. The file can contain more than one `<connector-factory>` element. By adding more `<connector-factory>` elements, you can create Multiple Managed connection factories. 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 Connection Factories, you must add new <connector-factory> nodes in the file. For example:

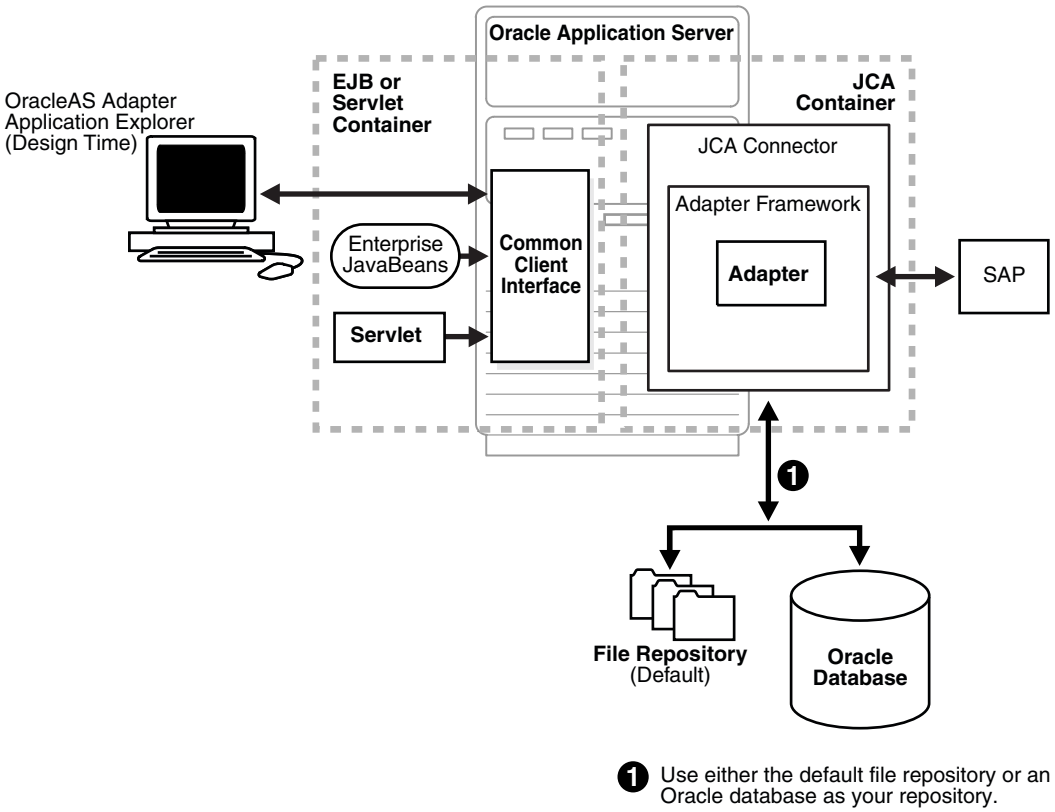
```
<?xml version="1.0"?>
```

```
<!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>
```

### Oracle Application Server Adapter JCA Architecture

Figure 3–1 shows deployment of the 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 JCA 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 JCA Architecture***



# OracleAS Adapter BSE Integration with OracleAS Integration InterConnect

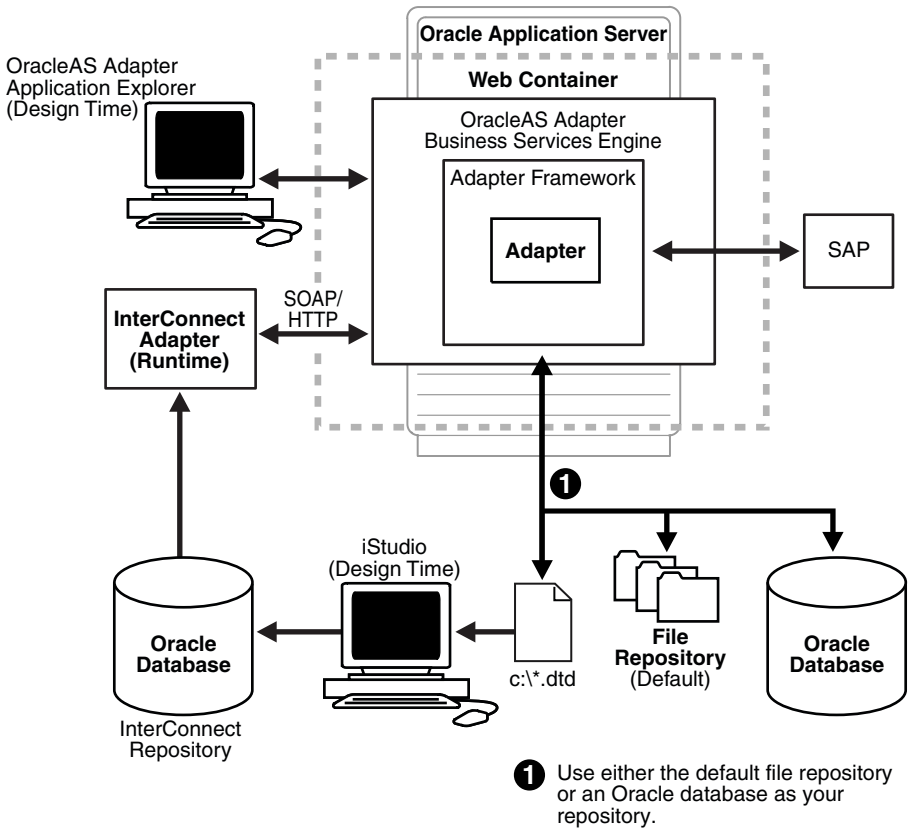
**See Also:** "Deployment and Integration through OracleAS Web Services" in *Oracle Application Server Adapter Concepts*

### BSE Architecture as Deployed to Oracle Application Server

Figure 3–2 shows adapter framework deployment with BSE to OracleAS Integration InterConnect. In a runtime service scenario, the OracleAS Integration InterConnect EIS Adapter Plugin (EIS Adapter Plugin) receives DTD-compliant XML from the **agent** component of the EIS Adapter Plugin. The EIS Adapter Plugin strips runtime information from the XML, wraps the XML in a SOAP envelope, and sends the result to BSE, including the runtime information in the SOAP request. BSE receives the request, removes the envelope, retrieves Web service method metadata, including adapter and connection information from the repository, and makes the adapter request.

BSE receives the adapter response, wraps the response XML in a SOAP envelope, and returns it to the EIS Adapter Plugin. The EIS Adapter Plugin then strips the SOAP envelope, strips the namespace prefix, if present, and passes the DTD-compliant XML to the agent component of the EIS Adapter Plugin.

***Figure 3–2 BSE Architecture as Deployed to Oracle Application Server***




Upon installation of the Oracle Web Services Adapter, an `adapter.ini` file is created. The file consists of all the initialization parameters that the adapter reads at

startup. You can configure the OracleAS Integration InterConnect EIS Adapter Plugin by modifying these parameters.

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

# 4

# Examples

This chapter contains the following examples:

- Creating Events
- SAP Event Integration
- SAP Service Integration

The event and service scenarios shown depend on the following prerequisites and configuration steps:

### Prerequisites

- OracleAS Adapter for SAP installed on the Oracle Application Server.
- OracleAS Database adapter deployed and configured.
- OracleAS Integration InterConnect Adapter Plugin for EIS installed and running.

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

### Configuration Steps

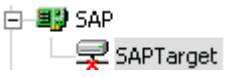
The examples present all the configuration steps necessary for demonstrating service and event integration with SAP. See the following for more information:

1. Configure SAP for outbound processing. See Appendix A, "Configuring SAP for Inbound and Outbound Processing" for more information.
2. Configure the OracleAS Adapter for SAP for services and events. See Chapter 2, "Adapter Configuration Using Application Explorer" for more information.
3. Configure OracleAS Integration InterConnect iStudio for service and event interactions. For more information, see the service and event steps that follow.

## Creating Events

The following example describes how to create events for DEBMAS05 and how to create a channel and a port.

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




   a. Expand the **SAP** node.

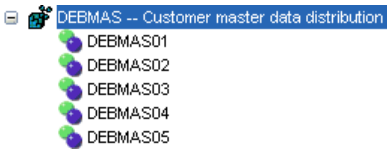
**b.** Click the target name, for example, SAPTarget, under the **SAP** node.

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

3. Verify your connection parameters. If required, provide the password.
4. Right-click the target name and select **Connect.**

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

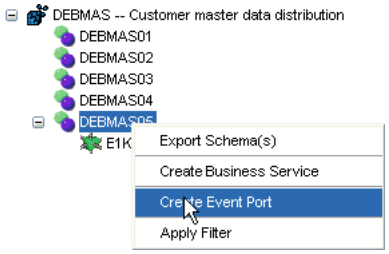

5. From the expanded Adapter list, expand the **SAP** node, expand **SAP46C**, and then select **ALE(IDOCS).**
6. Expand the **ALE(IDOCS)** node and select **DEBMAS.**




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

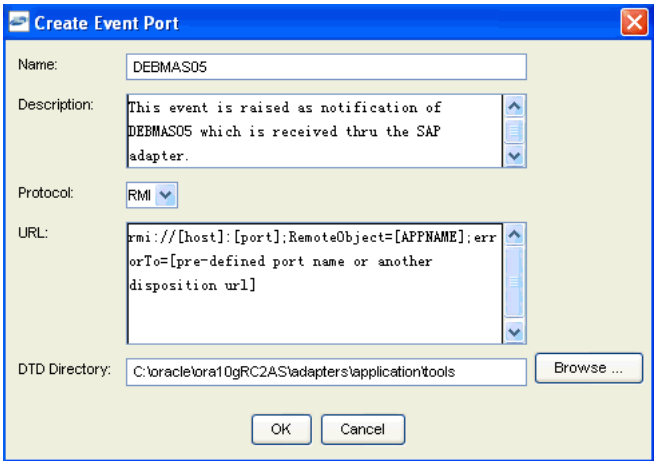
## Creating Events for DEBMAS05

To create events for DEBMAS05:




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

   The Create Event Port dialog box opens.



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

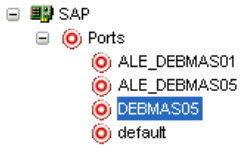
b. From the list, select the required disposition, for example, RMI.

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.

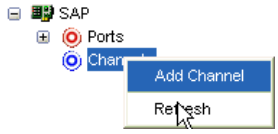
## Creating a Channel

To create a channel:

1. In the left pane, click the **Events** node.

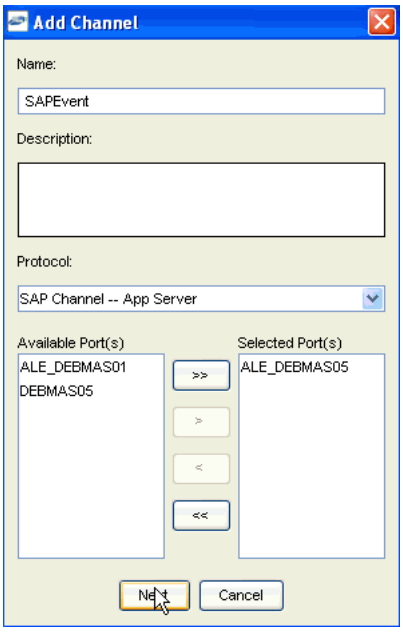
2. Expand the **SAP** node.

   The ports and channels nodes appear in the left pane.




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

   The Add Channel dialog box opens.




**a.** Type a name for the channel, for example, `TEST_CHANNEL`.

**b.** Type a brief description.

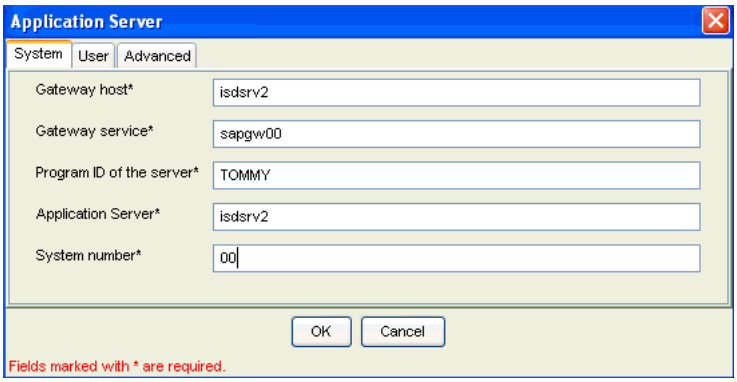
**c.** From the **Protocol** list, select **SAP Channel.**

**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:

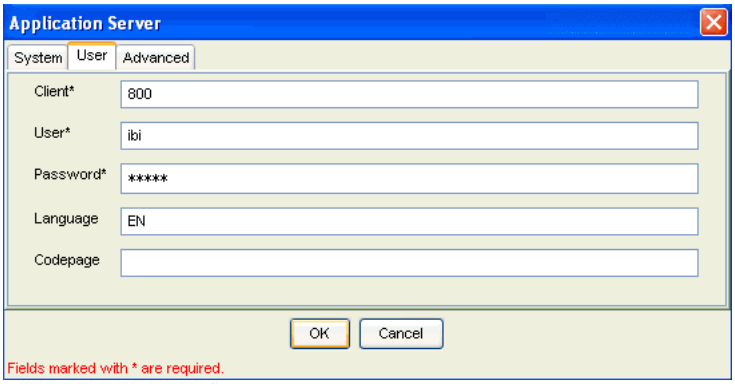
- System (Required)
- User (Required)
- Advanced

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

*Table 4–1 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 4–2 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 4–3 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 that was over the icon in the left pane disappears.

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

# SAP Event Integration

This topic illustrates how the OracleAS Adapter for SAP R/3 integrates with SAP to receive event data. The procedures describe design time and runtime. The example demonstrates how the 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. The design time and runtime procedures are outlined in the following sections.

## OracleAS Integration InterConnect Design Time

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

### Starting the Repository

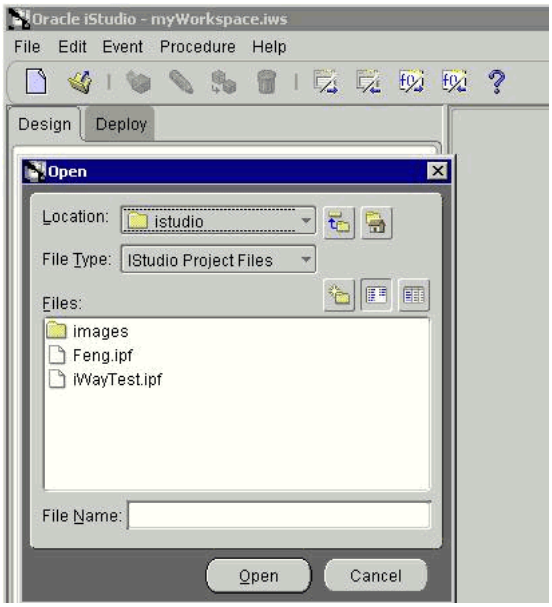
To start the repository, double-click the `start.bat` file located in the following directory:

```
D:\oracle\ora92InterCon\oai\9.0.4\repository\start.bat
```

### Creating a Common View

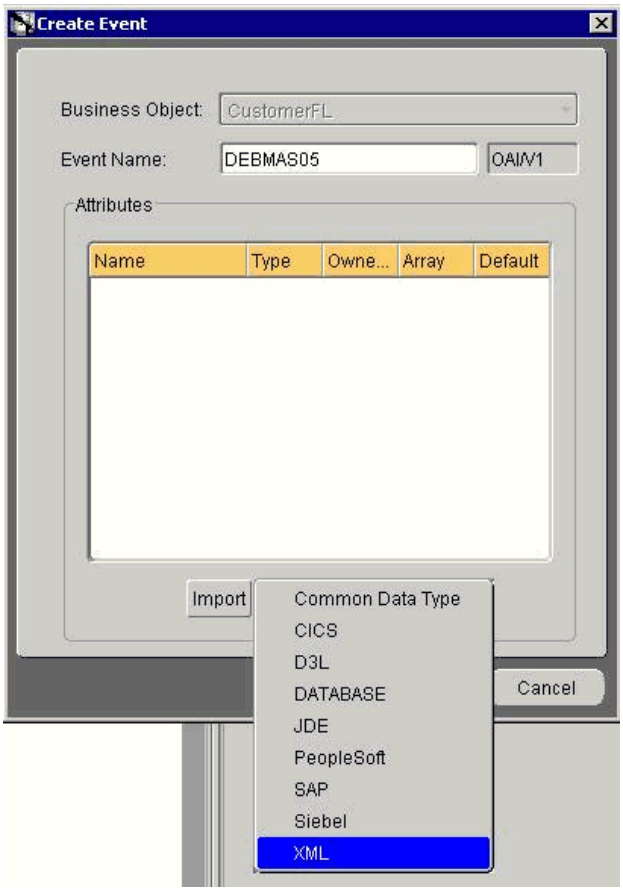
To create a Common View:

1. Start Oracle iStudio.

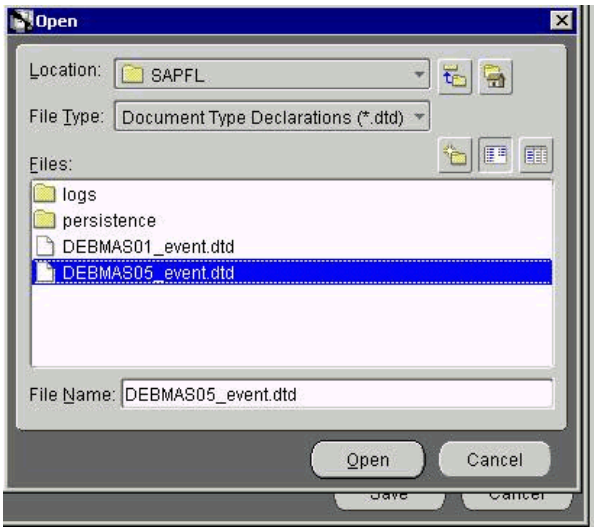



2. Open a new project.
3. Open **Common Views** and **Business Objects.**
4. Create a Business Object called **CustomerFL** and a new event under `CustomerFL`.

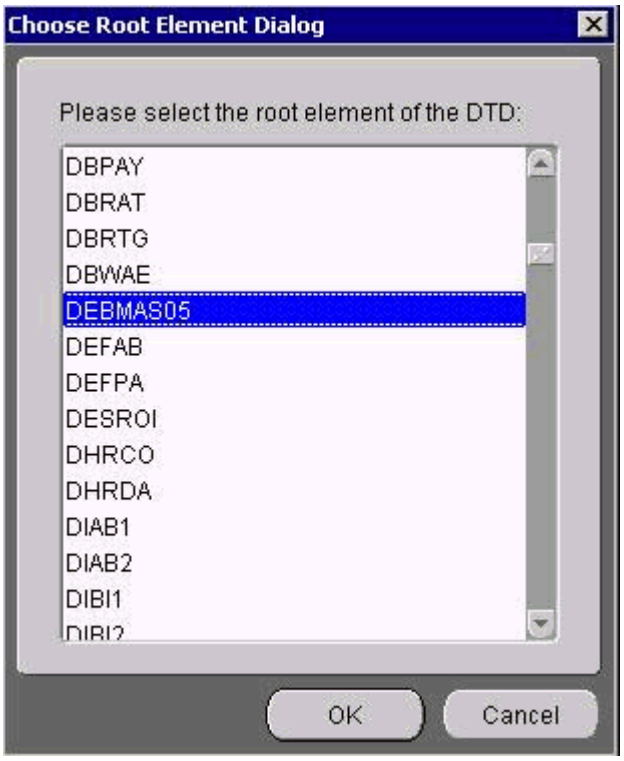
> **Note:** The event name must be the root element of the DTD generated from Application Explorer. In this example, the root element in the DTD is `DEBMAS05`.



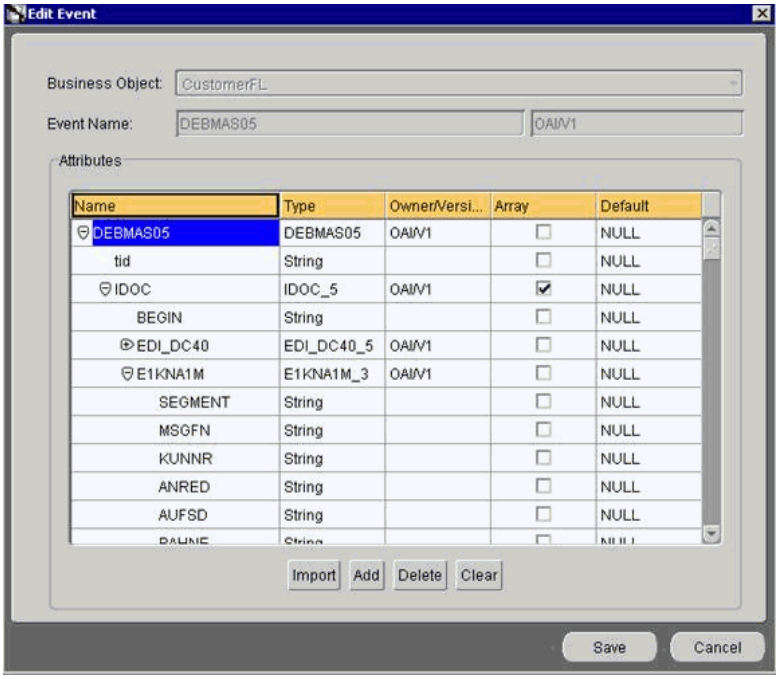

5. Click **Import** and select **XML** from the list.




6. Open the DTD generated from Application Explorer.




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



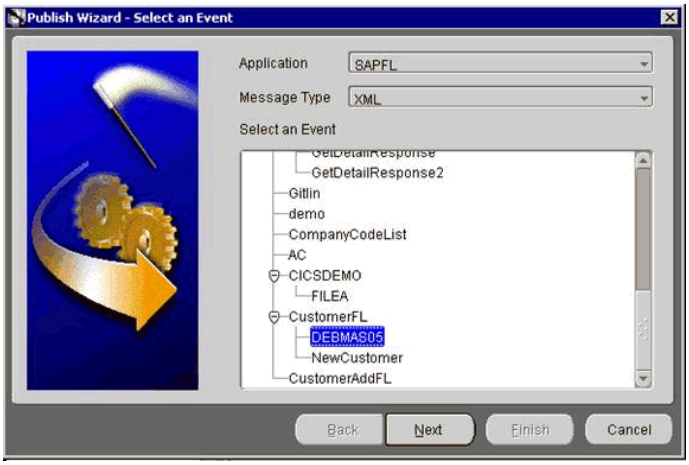

8. Click **Save.**

### Publishing an Event

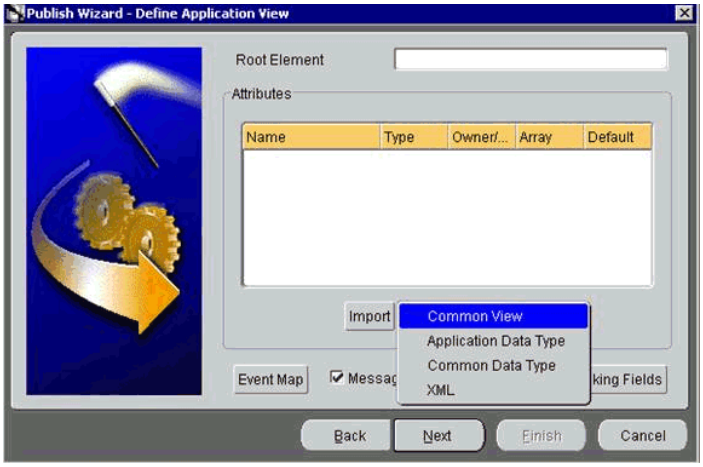
To publish an event:

1. Create a new application called **SAPFL.**

2. Expand **SAPFL.**
3. Right-click **Publish Events** and select **New.**

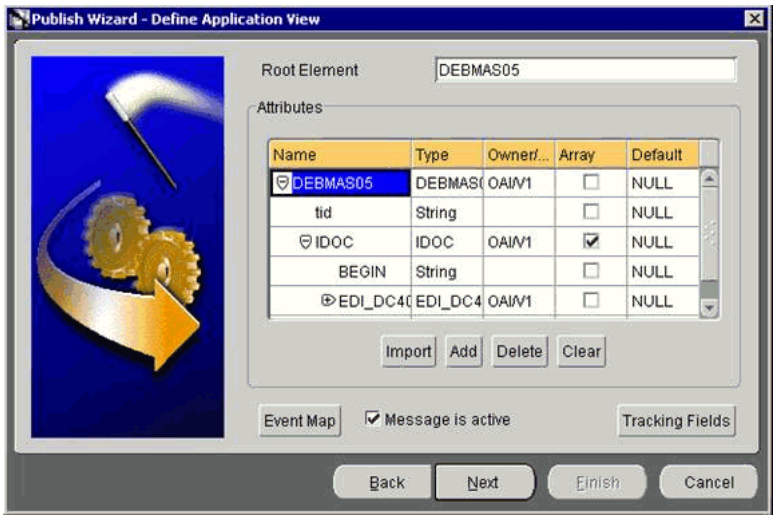



4. Select **XML** as the message type and select **DEBMAS05** under the CustomerFL business object as the event.
5. Click **Next.**



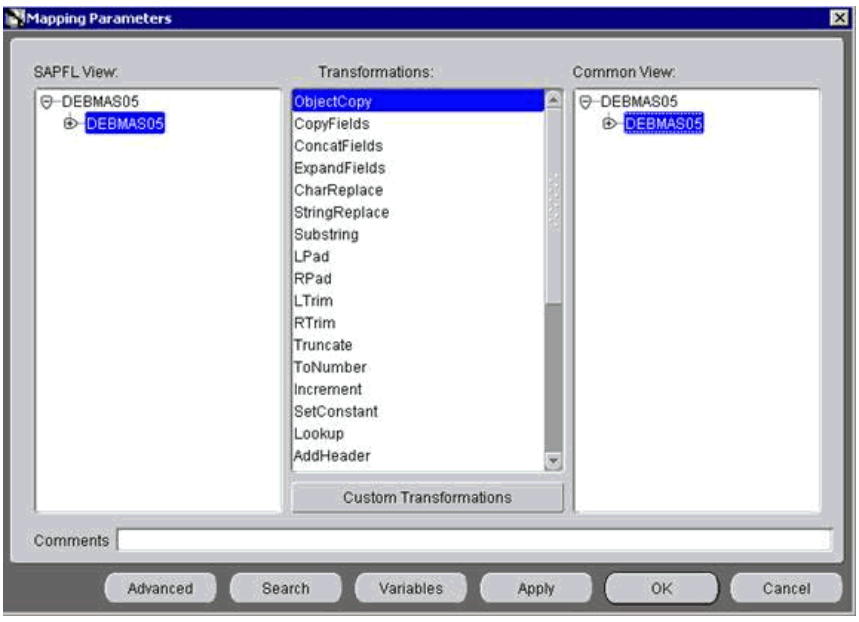

6. Click **Import** and select **Common View** from the list.

   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.**

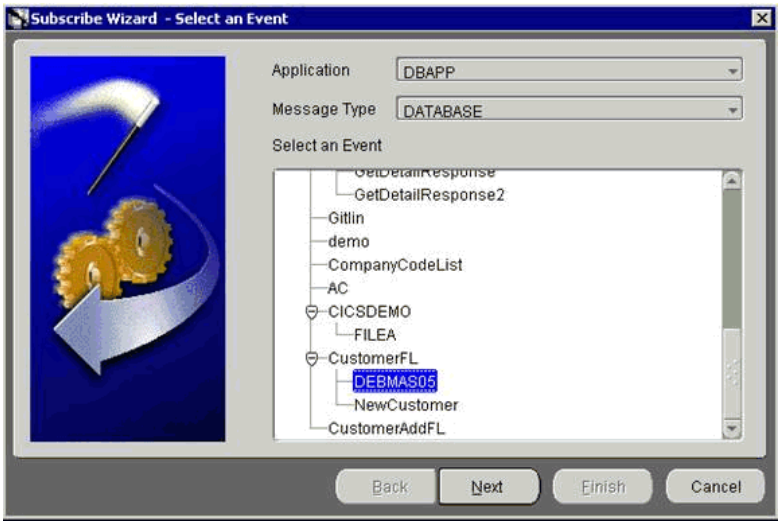
    The application definition for a published event is now complete.

## Subscribing an Event

To subscribe an event:

1. Create a new application called **DBAPP.**
2. Expand **DBAPP.**

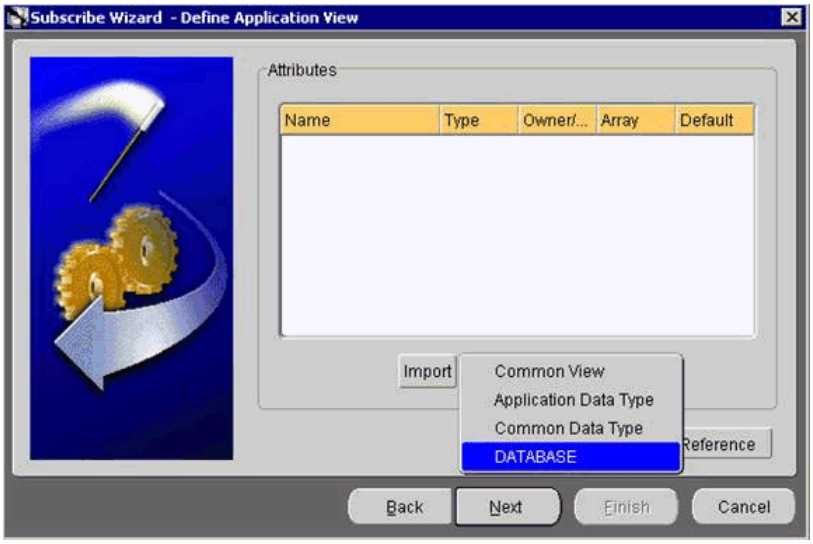
3. Right-click **Subscribe Events** and select **New.**




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

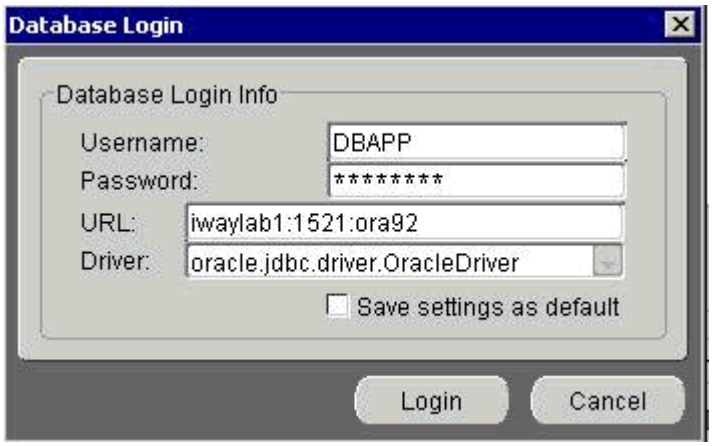
   b. Under the `CustomerFL` business object, select **DEBMAS05** as the event.

4. Click **Next.**

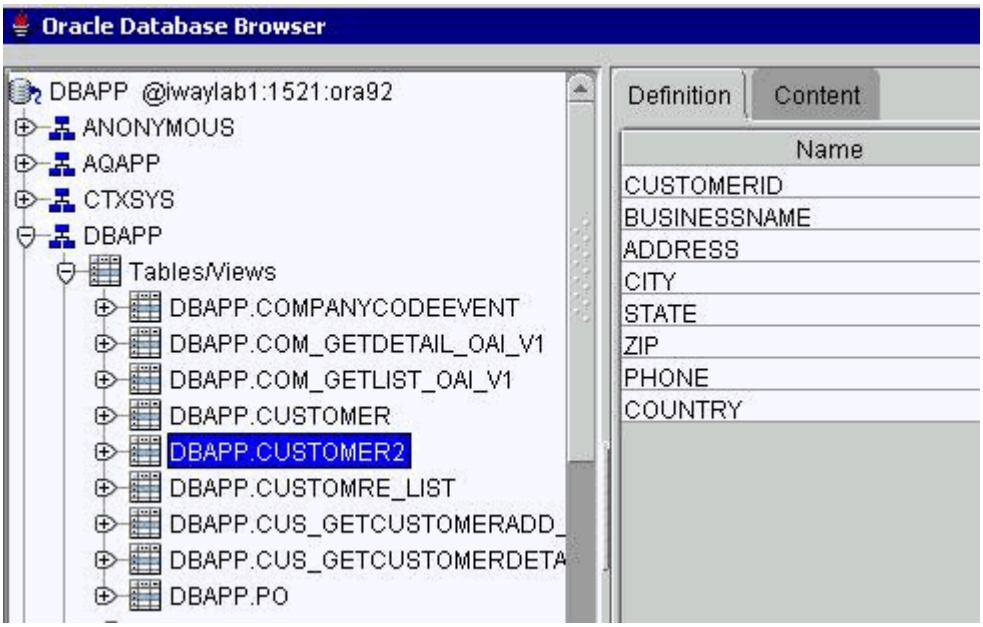



5. Click **Import** and select **Database** from the list.

   The Database Login dialog box opens.




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

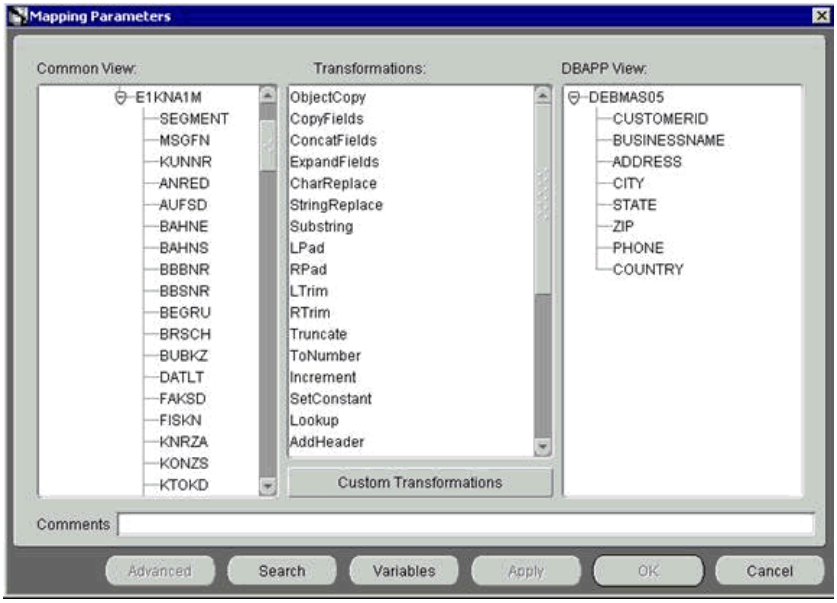



   a. Expand the **DBAPP** schema.

   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.




**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 opens.

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

The procedure details display.

**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. This script must be executed to insert event data into the database table.


10. Click **Finish.**

    The application definition for a subscribed event is now complete.

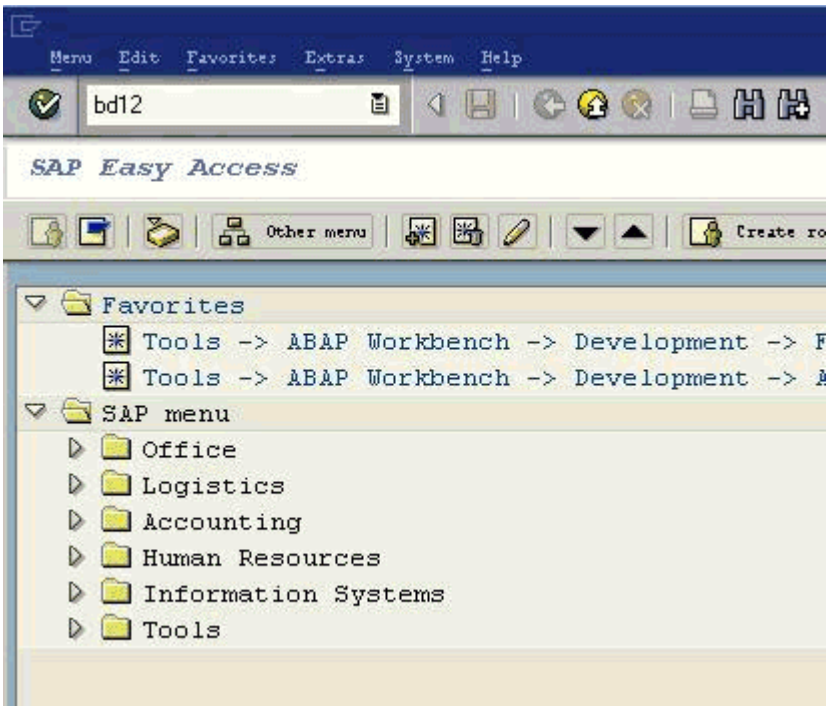
## OracleAS Integration InterConnect Runtime

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

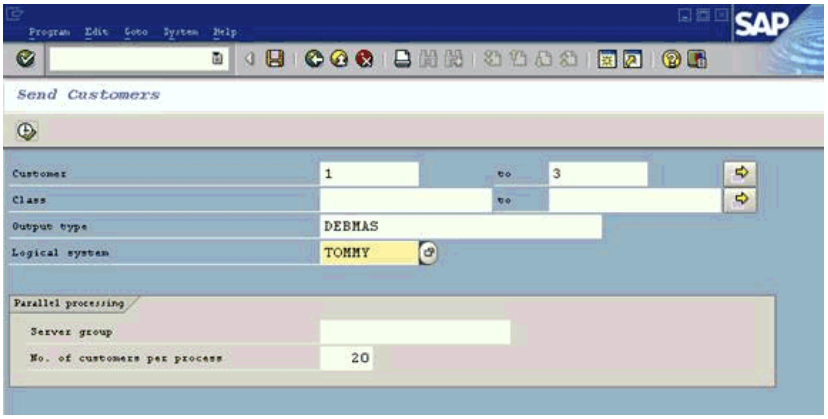
### Triggering an Event

To trigger an event in SAP:

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

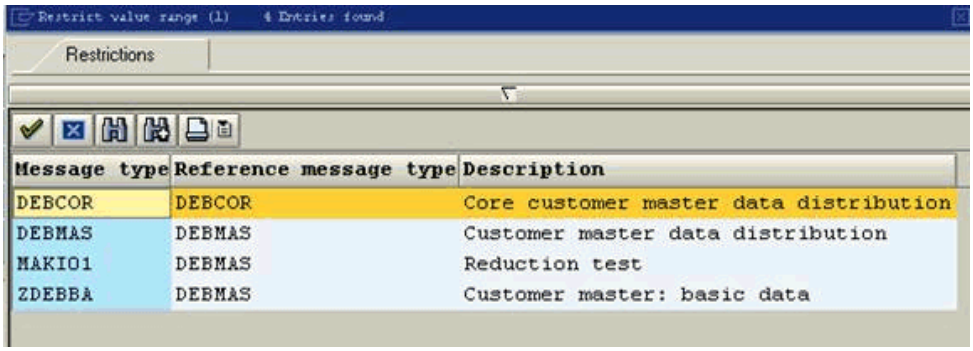



2. Execute the **bd12** transaction.




   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.

# SAP Service Integration

This topic illustrates SAP service integration. The procedures describe design time and runtime.

## OracleAS Integration InterConnect Design Time

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

### Starting the Repository

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

```
D:\oracle\ora92InterCon\oai\9.0.4\repository\start.bat
```

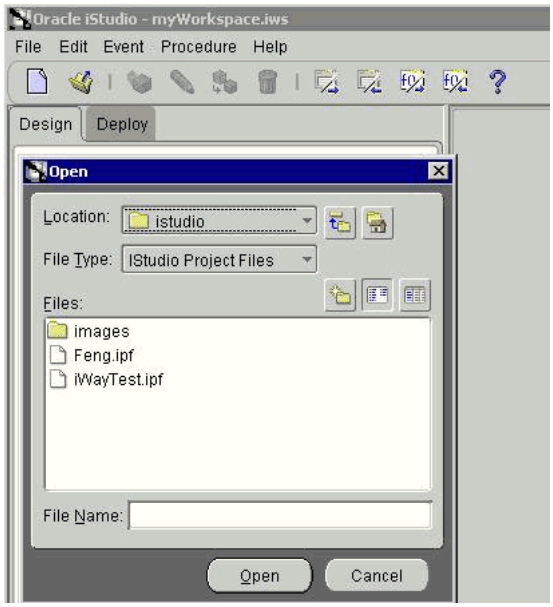
### Creating a Common View

To create a Common View:

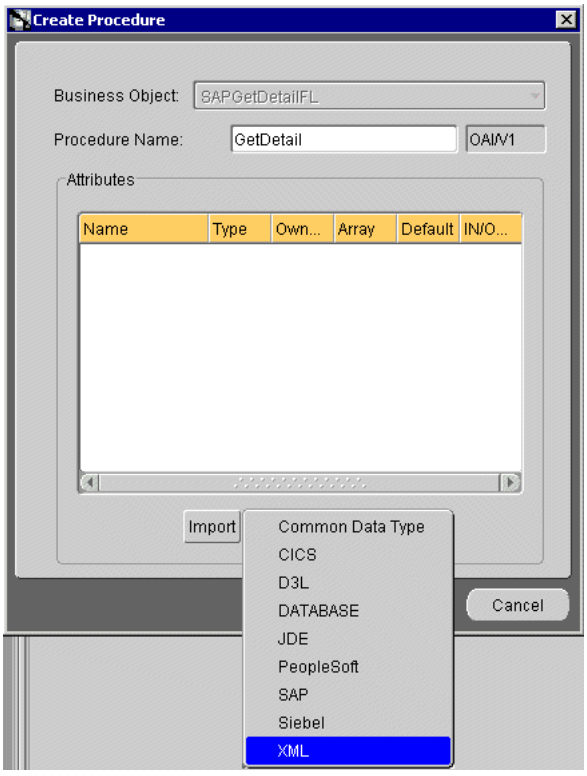
1. Start Oracle iStudio by double-clicking the `start.bat` file located in the following directory:

   ```
   D:\oracle\ora92iStudio\oai\9.0.4\istudio\iStudio.bat
   ```

   iStudio opens.

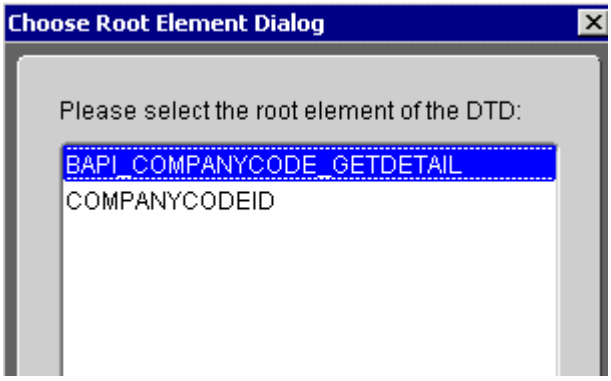



2. Open a project.
3. Open **Common Views** and **Business Objects.**
4. Create a Business Object called **SAPGetDetailFL.**

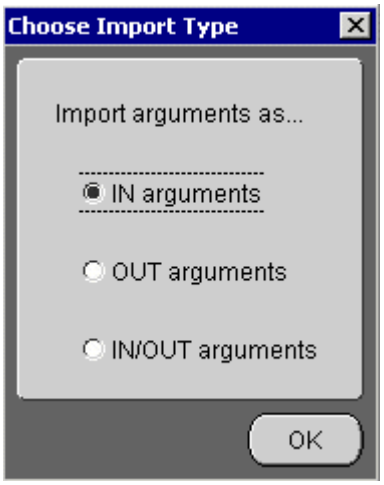



5. Create a new procedure under `SAPGetDetailFL` and type `GetDetail` as the procedure name.
6. Open the DTD generated from Application Explorer and load it.

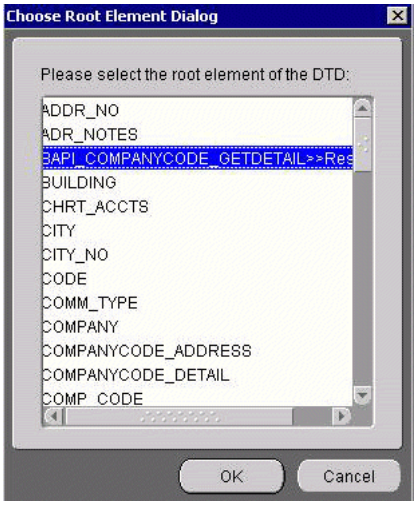
   The Choose Root Element dialog box opens.




7. Select the root element, `BAPI_COMPANYCODE_GETDETAIL`, for this example.
8. Click **OK.**

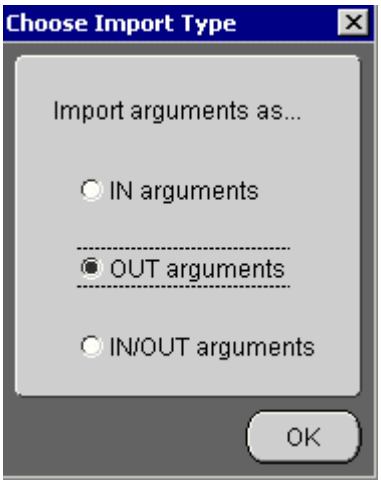



9. Select **IN arguments** as the import type for the request DTD and click **OK.**

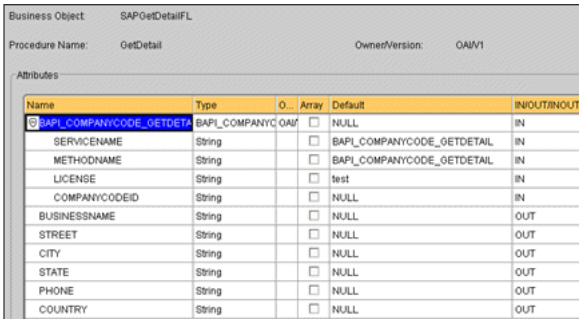



10. Import the response DTD, select the root element, and click **OK.**

    The Choose Import Type dialog box opens.




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



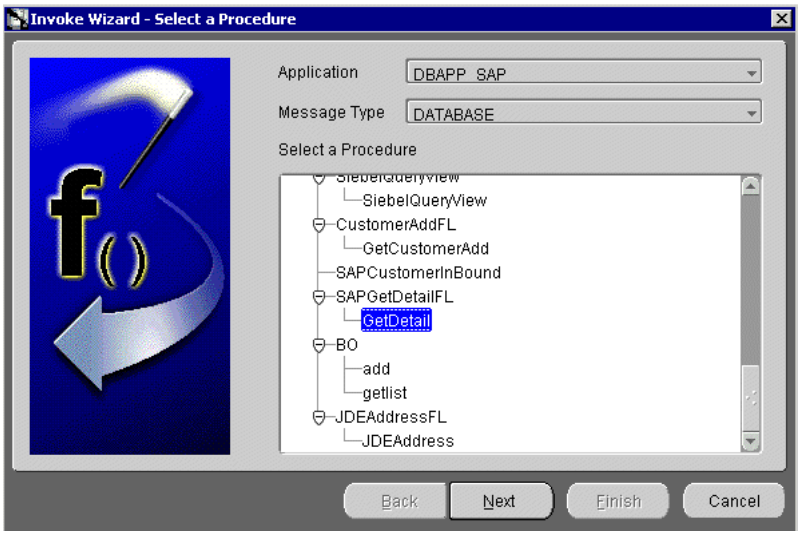

12. Manually enter all the OUT parameters as shown in the previous image.
13. To save the new procedure, click **Save.**

### 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 opens.

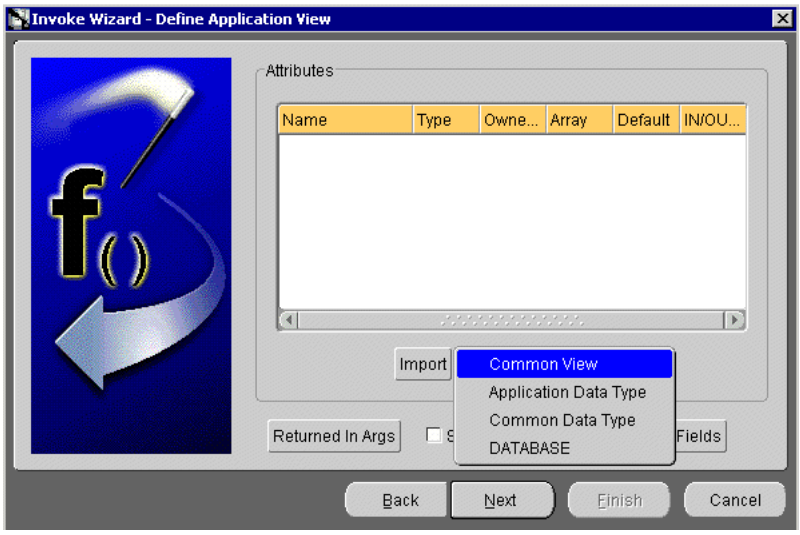



a. From the **Message Type** list, select **DATABASE.**

b. Expand the `SAPGetDetailFL` business object as the event and select **GetDetailFL.**

3. Click **Next.**

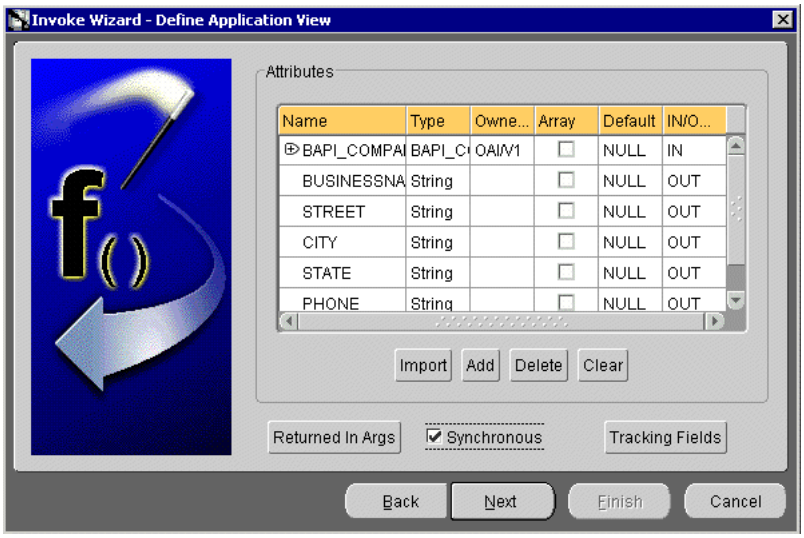
The Invoke Wizard - Define Application View window opens.




a. Click **Import.**

b. Select **Common View.**

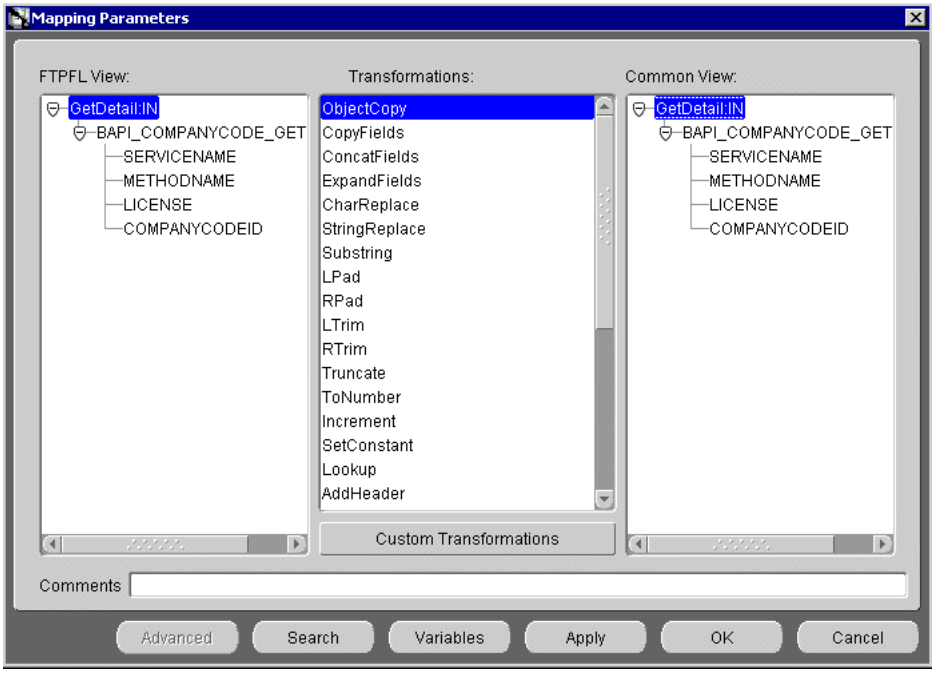
Information appears in the right pane.



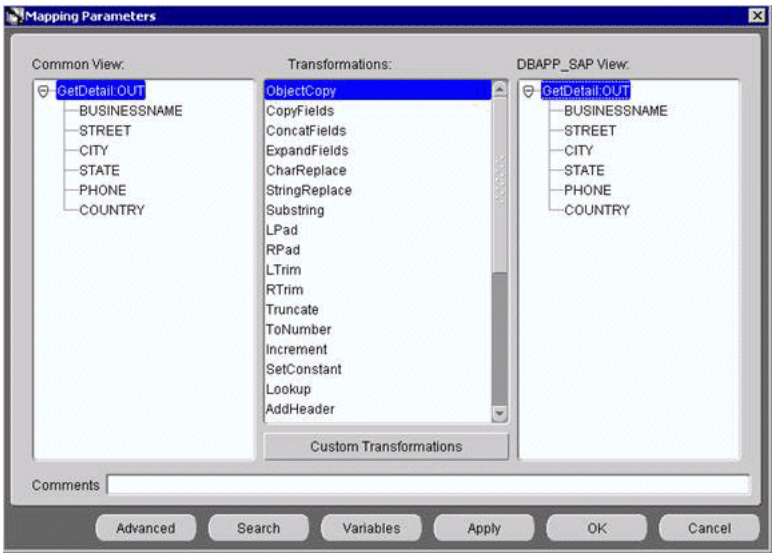

**c.** Select the **Synchronous** check box, because this is a request and a response.

4. Click **Next.**

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

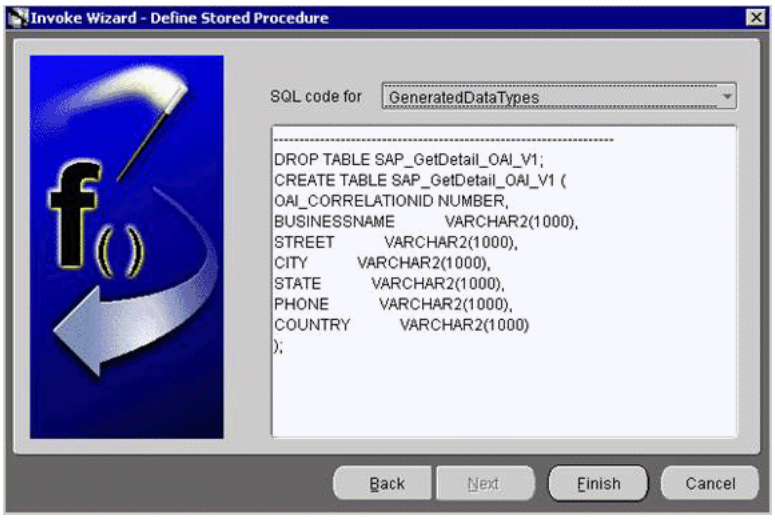



In this example, the Application View and the Common View have the same structure. All the attributes can be mapped by using ObjectCopy Transformation.




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

   iStudio generates SQL code.




7. Click **Finish.**

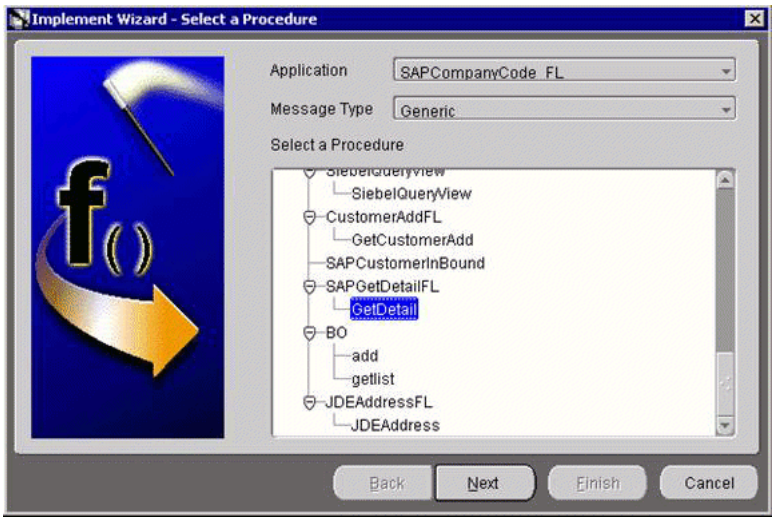
   The application definition for the invoked procedure is now complete.

## 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 opens.




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

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

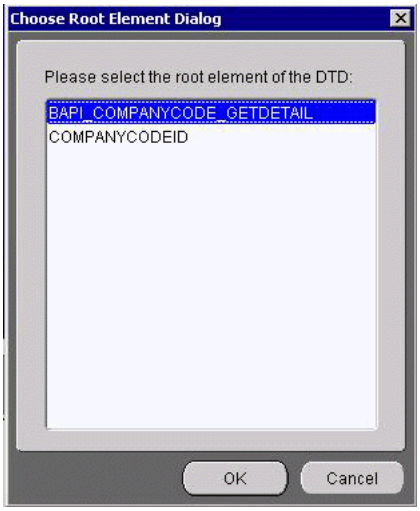
4. Click **Next.**

a. Click **Import.**

b. Select **XML.**

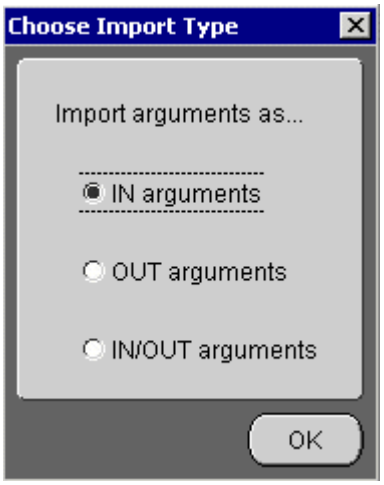
c. Select the request and response DTDs generated by Application Explorer.

5. Import the request and response DTDs into iStudio.



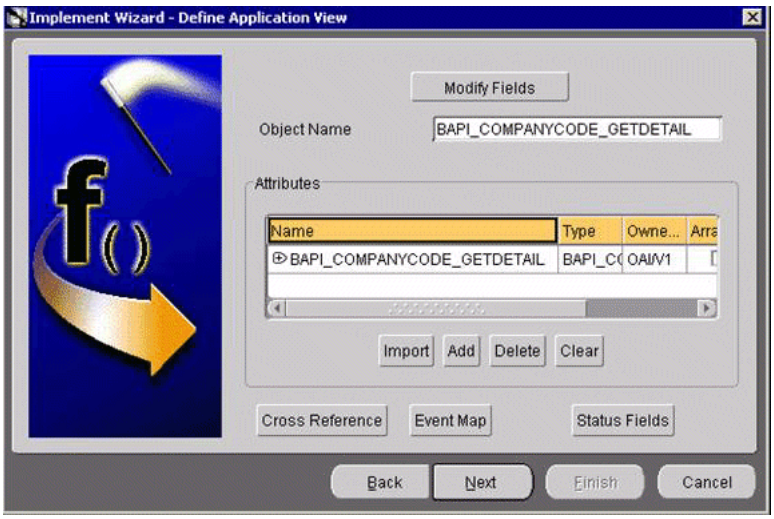

6. Select **BAPI_COMPANYCODE_GETDETAIL** as the root element of the request DTD.

The Choose Import Type dialog box opens.



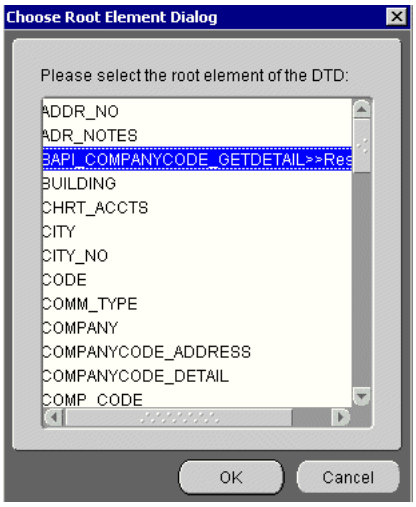

7. Select **IN arguments** as the import type for the request DTD and click **OK.**

   The Implement Wizard - Define Application View window opens.

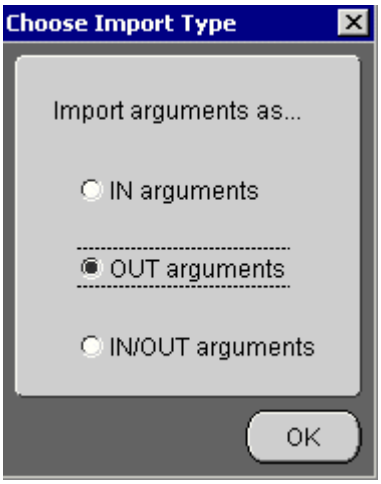



8. Type the root element of the request DTD in the Object Name field, if it is not automatically populated after importing the request DTD and click **Import.**

   The Choose Root Element dialog box opens.

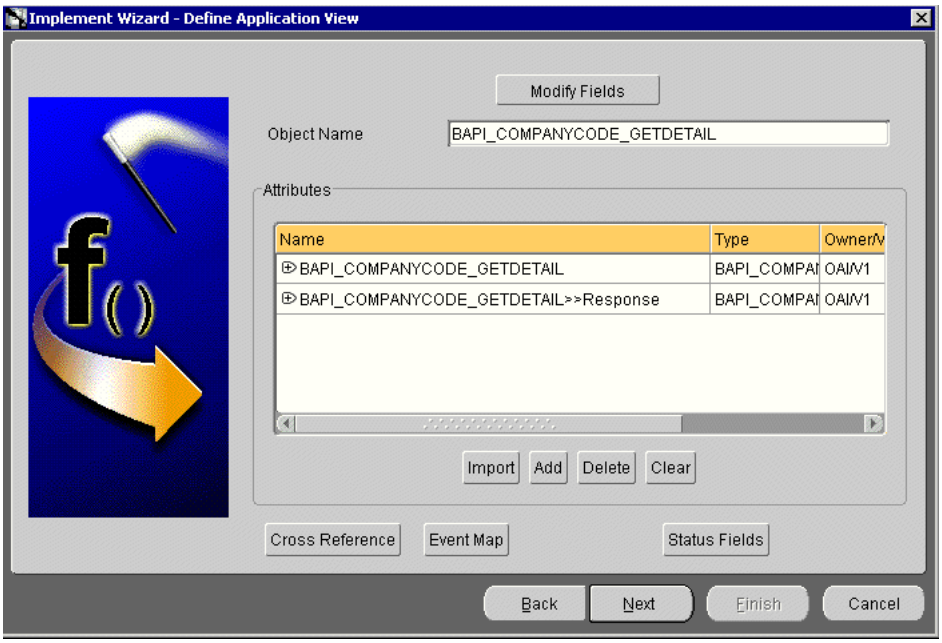



9. Select **BAPI_COMPANYCODE_GETDETAIL** as the root element and click **OK.**

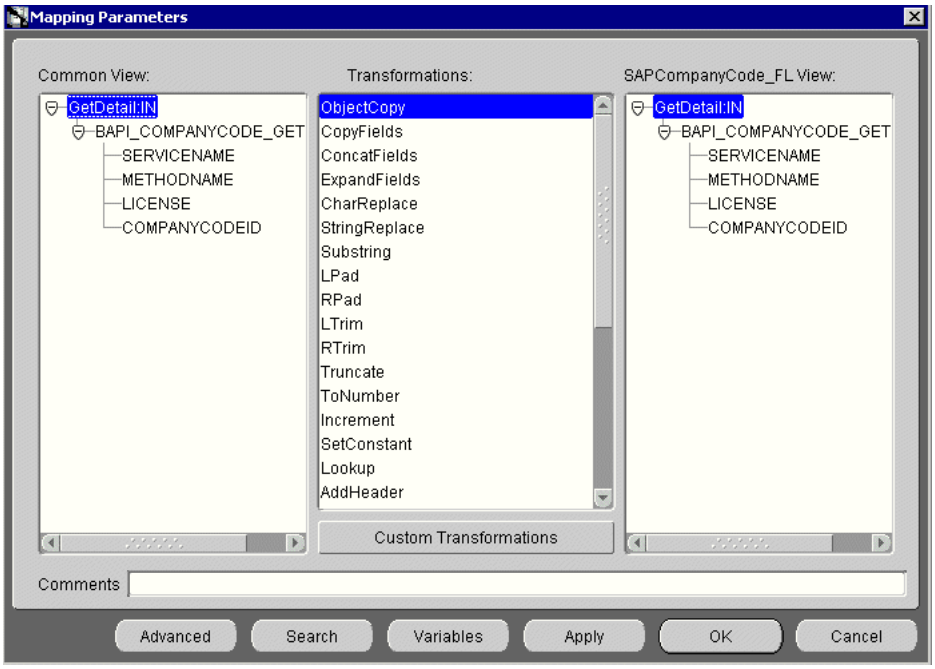



10. Select **OUT arguments** as the import type for the response DTD and click **OK.**

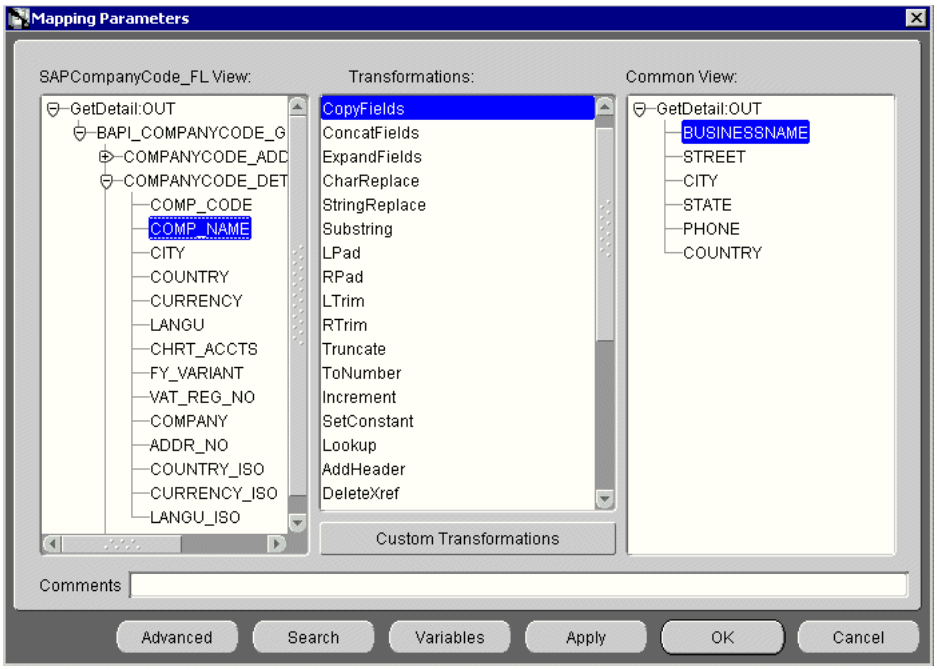
    Both the request and response DTDs are now imported into iStudio.




**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.



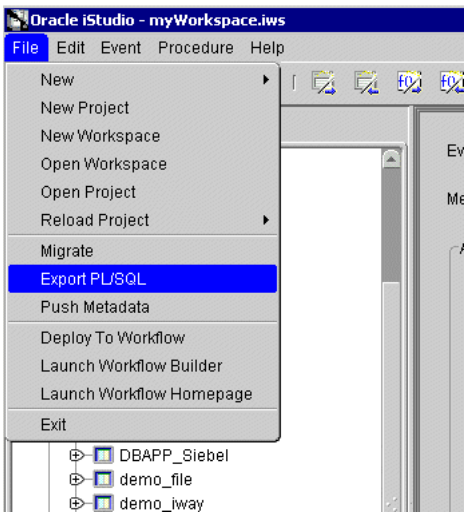

12. Click **Apply** and then **OK.**

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

## Exporting PL/SQL Code from iStudio

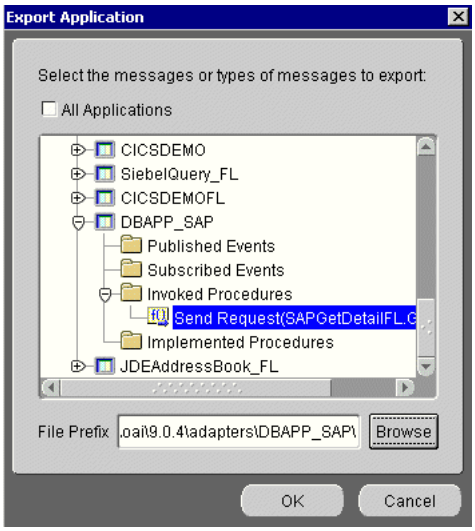
You must export the PL/SQL code created in "Defining an Implemented Procedure" on page 4-23 and execute it against the appropriate schema. In this example, the schema used is DBAPP_SAP.

To export PL/SQL code from iStudio:




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

   The Export Application dialog box opens.




**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 execute 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 executes 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

To edit the adapter.ini file:

1. Open the `adapter.ini` file.
2. Add the following two lines to `adapter.ini` for the adapter:

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

// IBSE URL
ibse_url=http://lee.ibi.com:7777/ibse/IBSEServlet/XDSOAPRouter
```

**lee.ibi.com**
Is the URL of the server.

**7777**
Is the port number.

## OracleAS Integration InterConnect Runtime

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

### Verifying Service Integration

To verify service integration:

1. Start the Oracle Application Server or ensure that the server is running.
2. Restart OC4J, if required, by executing 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 executing the following command:

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

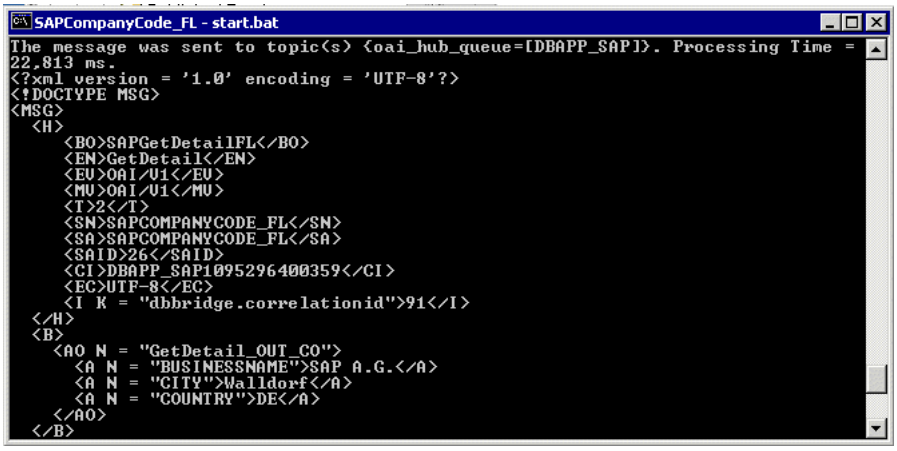
4. Invoke and implement the adapter by executing the following commands:

```
\InterConnect_HOME\oai\9.0.4\adapters\SAPCompanyCode_FL\start.bat
\InterConnect_HOME\oai\9.0.4\adapters\DBAPP_SAP\start.bat
```

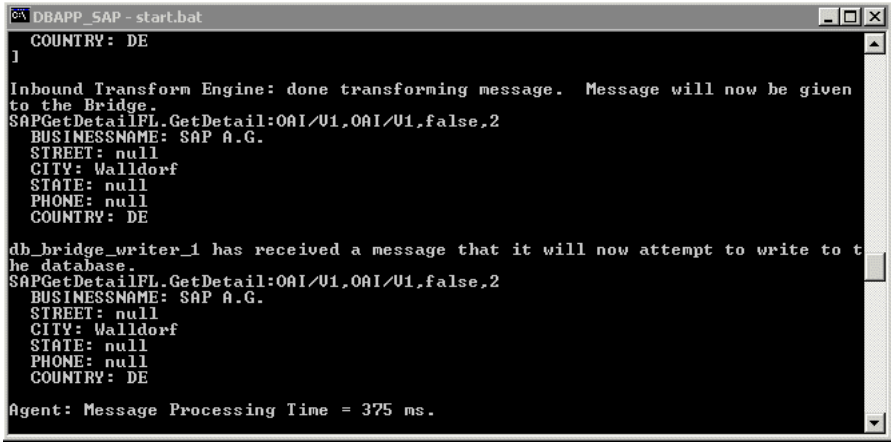
5. Log on to SQL*Plus with `DBAPP_SAP` and execute the following command:

```
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.



# 5

# 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 JCA or with a BSE configuration.

## Troubleshooting

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

- Application Explorer
- SAP
- OracleAS Adapter JCA
- BSE

> **Note:** Log file information that can be relevant in troubleshooting can be found in the following locations:
>
> - The OracleAS Adapter JCA 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, invoke the ae batch script, ae.bat, found under *OracleAS_home*\adapters\application\tools or on UNIX invoke the ae script, ae.sh, found under *OracleAS_home*/adapters/application/tools.

| Error | Solution |
|---|---|
| Cannot connect to the OracleAS Adapter for SAP from Application Explorer. | Ensure that:<br>SAP is running.<br>The Application Server name, System Number, and Client Number are correct.<br>The SAP user ID and password are correct. |
| Cannot connect to the SAP target through Application Explorer. The following error message appears:<br>`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. |
| 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 your SAP system through Application Explorer. The following error message appears:<br>`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 the parameter values to connect to your application server are correct. |
| Cannot connect to your SAP system through Application Explorer even though SAP is running. The following error message appears:<br>`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 JCA are installed on the same server). The following error message appears:<br>`com.ibi.sapr3.SapAdapterException: java.lang.ExceptionInInitializerError: JCO.classInitialize(): Could not load middleware layer 'com.sap.mw.jco.rfc.MiddlewareRFC'`<br>`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` | Put `sapjco.jar` in the server classpath. |

## SAP

| Error | Solution |
| --- | --- |
| When executing a request, the following error message appears:<br>`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 and is available in SAP. |
| When executing a request, the following error message appears:<br>`AdapterException: java.lang.Exception: Object type unknown for business object: CUST` | Check the syntax of your input XML document and make sure the Object type is correct. |
| When executing a request, the following error message appears:<br>`AdapterException: java.lang.Exception: Unable to retrieve BAPI name for: CUSTOMER.DETAIL2` | Check the syntax of your input XML document and make sure the name of the BAPI is correct and is available in SAP. |
| When executing a request, the following error message appears:<br>`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 make sure the IDoc extension is correct and is available in SAP. |
| When executing a request, the following error message appears:<br>`AdapterException: java.lang.Exception: BapiError/BapiAbort: You are not authorized to display customers.` | Make sure your user ID has the correct permissions configured in SAP. Consult your SAP administrator for more information. |

**OracleAS Adapter JCA**

| Error | Solution |
|---|---|
| In Application Explorer, the following error message appears when you attempt to connect to an OracleAS Adapter JCA configuration:<br>`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 |

# BSE Error Messages

This topic discusses the different types of errors that can occur when processing Web services through the Business Services Engine (BSE).

## General Error Handling in BSE

The Business Services Engine (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 execution time, 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 when you encounter some of the more common error conditions on an adapter-specific basis.

Usually the SOAP gateway (**agent**) inside BSE passes a SOAP request message to the adapter required for the Web service. If an error occurs, 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 execution, 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

When the OracleAS Adapter for SAP receives a SOAP request message that does not conform to the WSDL for the Web services being executed, 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

When the OracleAS Adapter for SAP executes an SAP object using input parameters passed in the SOAP request message that do not match records in SAP, 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

When the OracleAS Adapter for SAP cannot connect to SAP when executing a Web service, 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

When the OracleAS Adapter for SAP receives a SOAP request message that does not conform to the WSDL for the Web services being executed, 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

When the OracleAS Adapter for SAP executes a SOAP request using input parameters passed that do not match records in the target system, 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>
```

# 6

# Advanced Topics

This chapter includes the following topics:



## Using 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 Policy-Based Security

Web services provide a layer of abstraction between the back-end business logic they invoke, and the user or application running the Web service. This enables easy application integration but raises the issue of controlling the use and execution 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 checked 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 execute the Business Service.

When a policy is not applied, the default value for a Business Service is to "grant all". For example, anybody can execute 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.

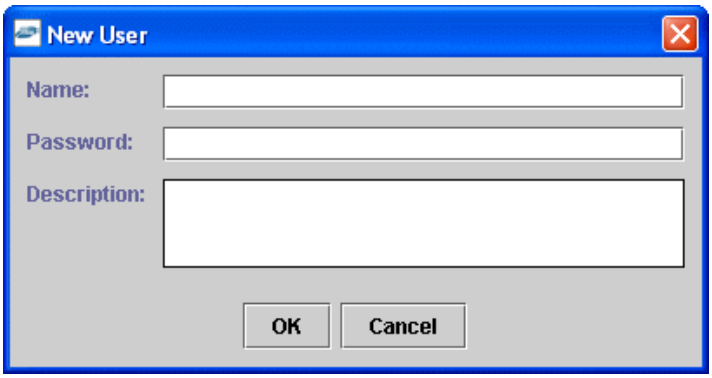
1. Open Application Explorer.
2. Right-click the configuration to which you want to connect, for example, `SampleConfig`. See Chapter 2, "Adapter Configuration Using Application Explorer" for information on creating a new configuration.
3. Select **Connect.**

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


   a. Expand the **Business Services** node by clicking the plus (+) symbol.
   b. Expand the **Configuration** node by clicking the plus (+) symbol.
   c. Expand the **Security** node by clicking the plus (+) symbol.
   d. Expand the **Users and Groups** node by clicking the plus (+) symbol.


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

   The New User dialog box opens.




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

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

c. In the **Description** field, type 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, "Adapter Configuration Using Application Explorer" 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 by clicking the plus (+) symbol.

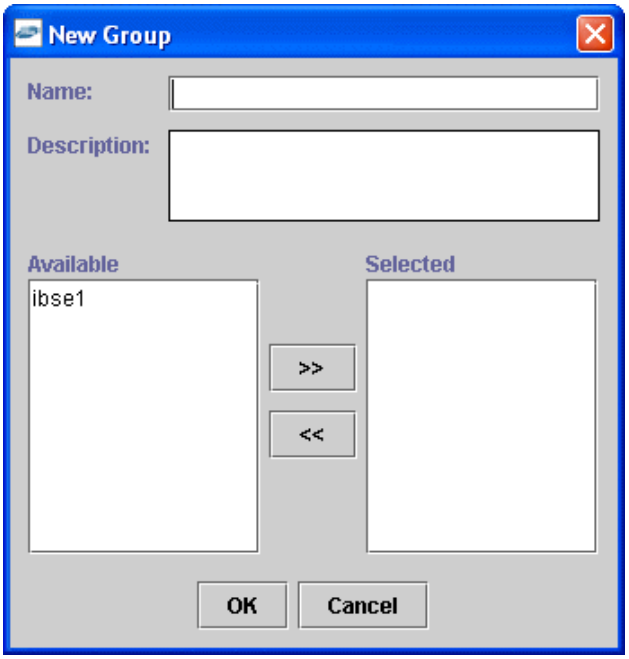
b. Expand the **Configuration** node by clicking the plus (+) symbol.

c. Expand the **Security** node by clicking the plus (+) symbol.

d. Expand the **Users and Groups** node by clicking the plus (+) symbol.

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

   The New Group dialog box opens.




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

   b. In the **Description** field, type 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 execute 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, "Adapter Configuration Using Application Explorer" 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 by clicking the plus (+) symbol.

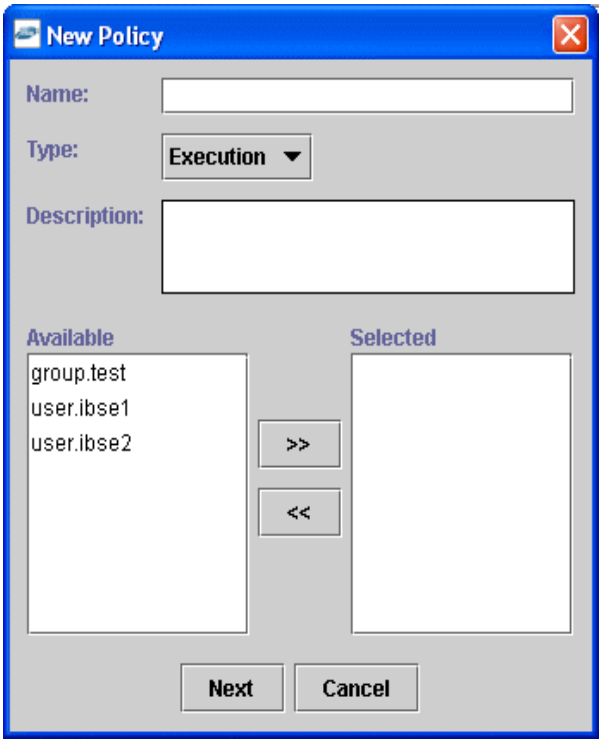
b. Expand the **Configuration** node by clicking the plus (+) symbol.

c. Expand the **Security** node by clicking the plus (+) symbol.

d. Expand the **Policies** node by clicking the plus (+) symbol.


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

   The New policy dialog box opens.




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

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

c. In the **Description** field, type 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.

---

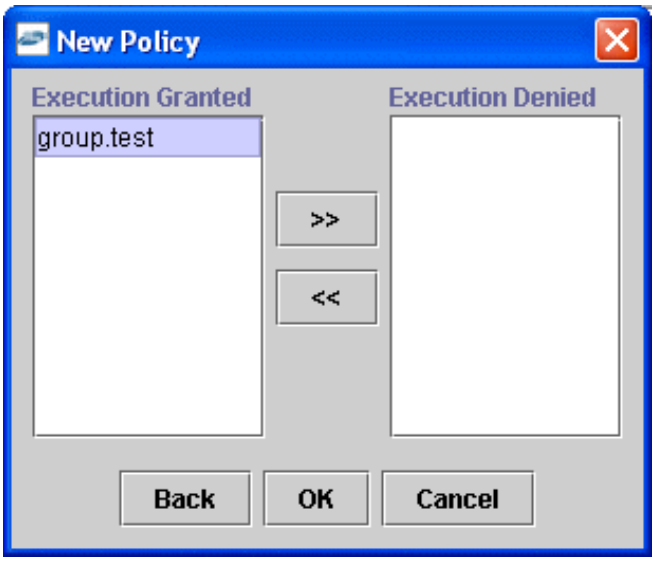
**Note:** This user ID is checked against the value in the user ID element of the SOAP header sent to BSE in a SOAP request.

---

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

6. Click **Next.**

The New Policy permissions dialog box opens.




7. To grant permission to a user or group to execute a Business Service, select the user or group and move them into the **Execution Granted** list by selecting the double left-facing arrow.
8. To deny permission to a user or group to execute a Business Service, select the user or group and move them into the **Execution Denied** list by selecting the double right-facing arrow.
9. 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.

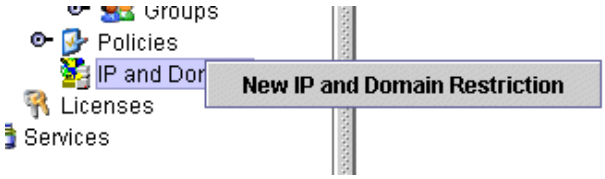
1. Open Application Explorer.
2. Right-click the configuration to which you want to connect, for example, SampleConfig. See Chapter 2, "Adapter Configuration Using Application Explorer" for information on creating a new configuration.
3. Select **Connect.**

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

   a. Expand the **Business Services** node by clicking the plus (+) symbol.

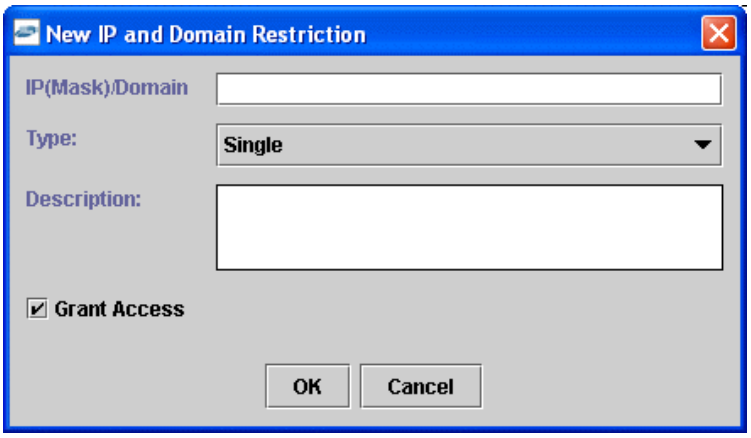
b. Expand the **Configuration** node by clicking the plus (+) symbol.

c. Expand the **Security** node by clicking the plus (+) symbol.




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

   The New IP and Domain Restriction dialog box opens.




   a. In the **IP(Mask)/Domain** field, type 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, for example, yahoo.com.

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

   c. In the **Description** field, type 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 JCA 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.

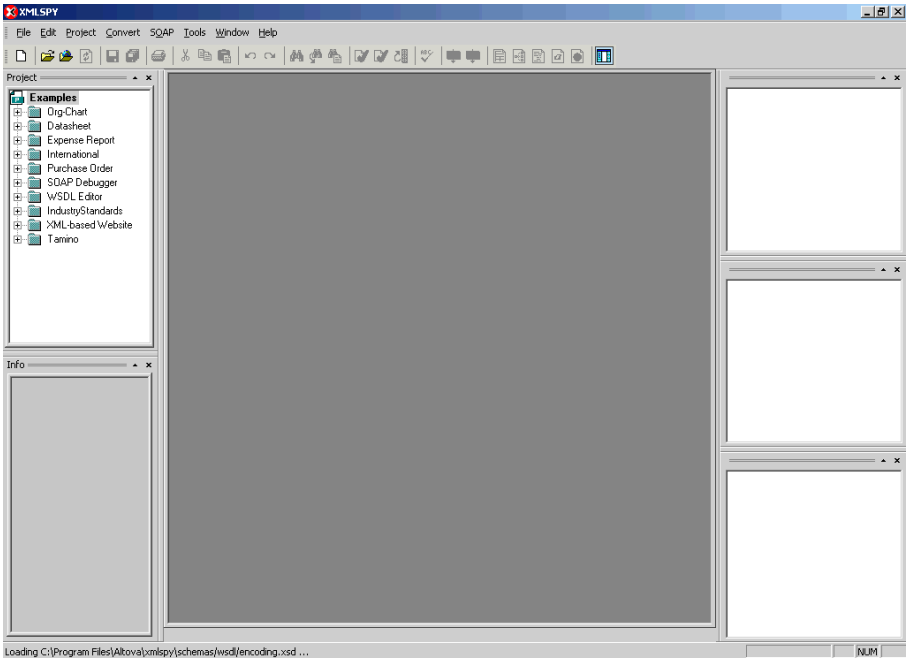
## Migrating a BSE Repository

To migrate a BSE repository:

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

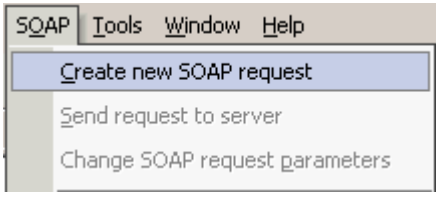
   ```
   http://localhost:7777/ibse/IBSEServlet/admin/iwcontrol.ibs
   ```

2. Open a third-party XML editor, for example, XMLSPY.



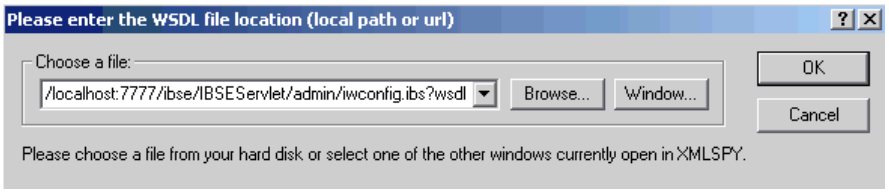

3. In the menu bar, click **SOAP.**

   A list of options appears.




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

   The WSDL file location dialog box opens.



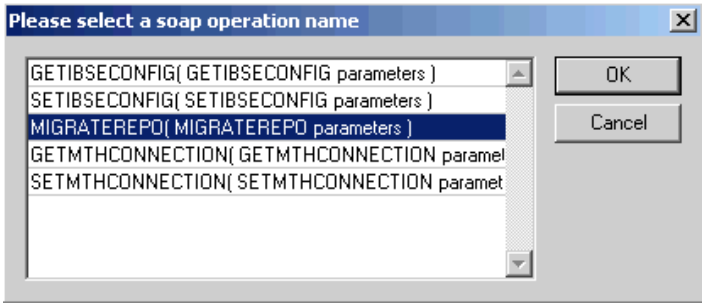

   a. In the **Choose a file** field, paste the BSE control service URL.

   b. Append **?wsdl** to the URL, for example:

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

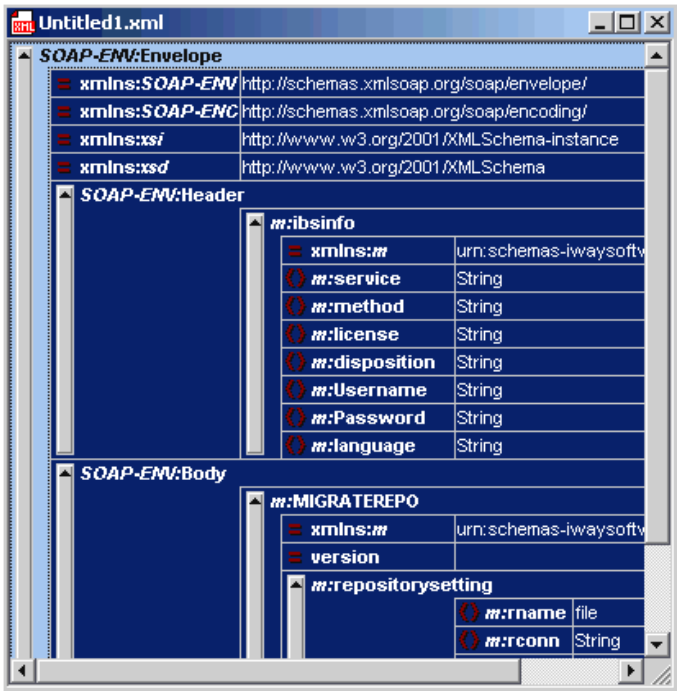
5. Click **OK.**

   The soap operation name dialog box opens and lists the available control methods.




6. Select the **MIGRATEREPO(MIGRATEREPO parameters)** control method and click **OK.**

   The following window opens, which shows 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 the **Text view** icon.

   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>
```

   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.

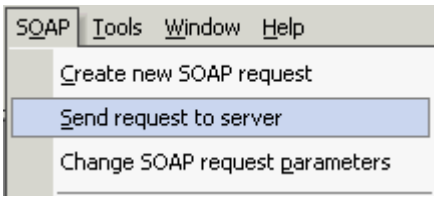
- If you want to migrate a single Web service from the current BSE repository, enter the Web service name in the `<m:servicename>` tag, for example:

```
<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. In 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 JCA Repository

To migrate a JCA repository:

1. Navigate to the location of your JCA configuration directory where the repository schemas and other information is stored, for example:

   *OracleAS_home*\adapters\application

2. Locate and copy the `repository.xml` file.

3. Place this file in a new JCA configuration directory to migrate the existing repository.

   Your JCA repository is migrated to the new JCA 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.

The IDoc is "accepted", that is, saved in the database, if the upstream system is recognized. If your partner is defined with the corresponding message in your partner profiles, the IDoc is then processed further. This is done independently in a 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

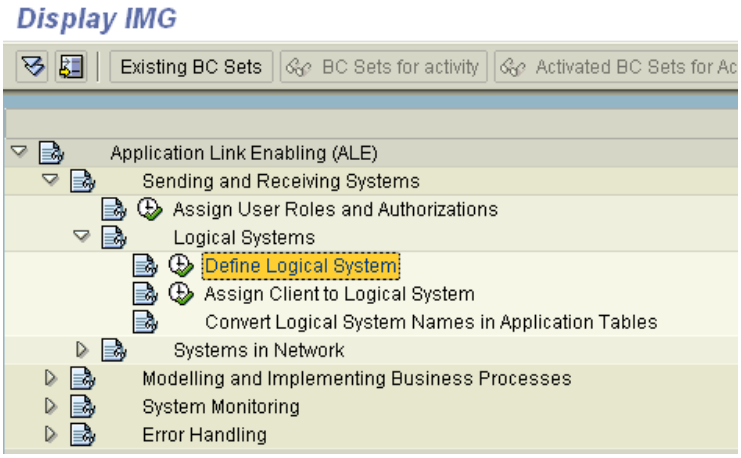
In any distributed environment, each participating system must have a unique ID to avoid confusion. In SAP, the name of the logical system is used as the unique ID. This name is assigned explicitly to one client in an SAP system.

### Defining a Logical System

To define a logical system:

1. Execute the **sale** transaction.


   The Display IMG window opens.



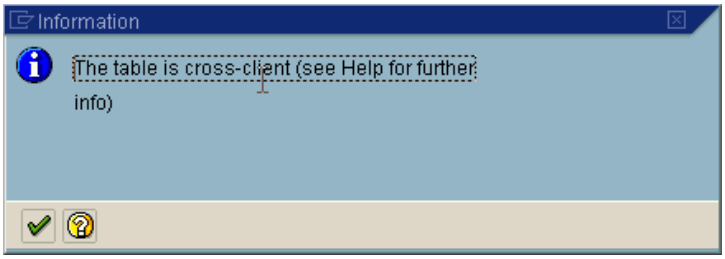

   a. Expand **Sending and Receiving Systems.**

   b. Expand **Logical Systems.**

   c. Select **Define Logical System.**

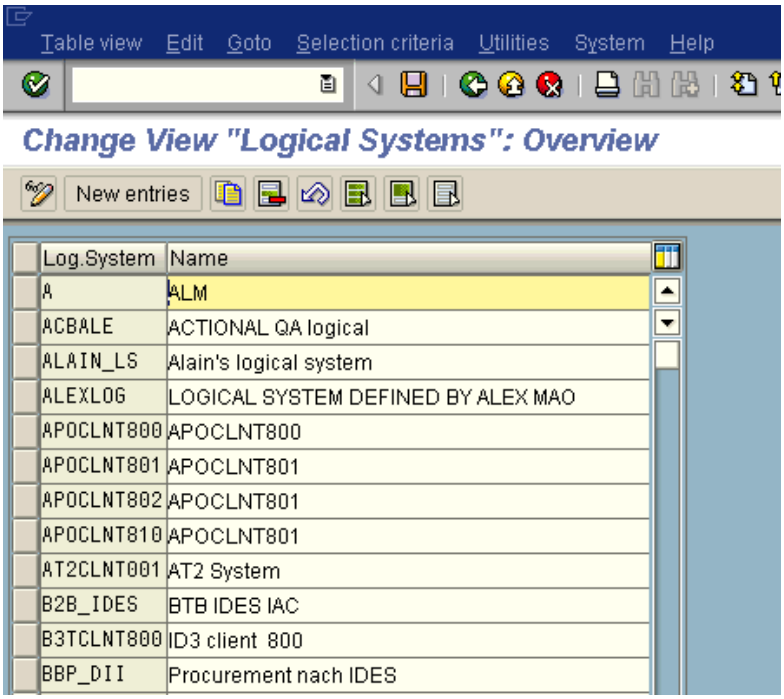
2. Click the **IMG - Activity** icon.

   A message window opens indicating that the table is cross-client.



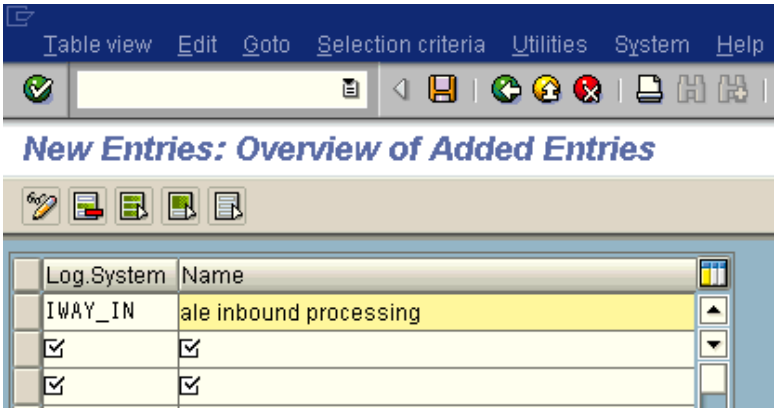

3. Click the **check mark** button to continue.

   The Change View "Logical Systems": Overview window opens.



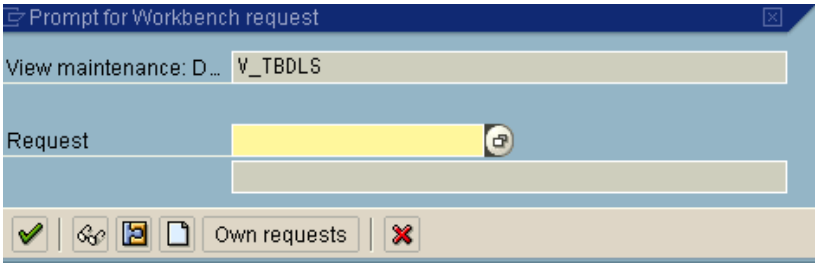

4. Click the **New Entries** button.

   The New Entries: Overview of Added Entries window opens.



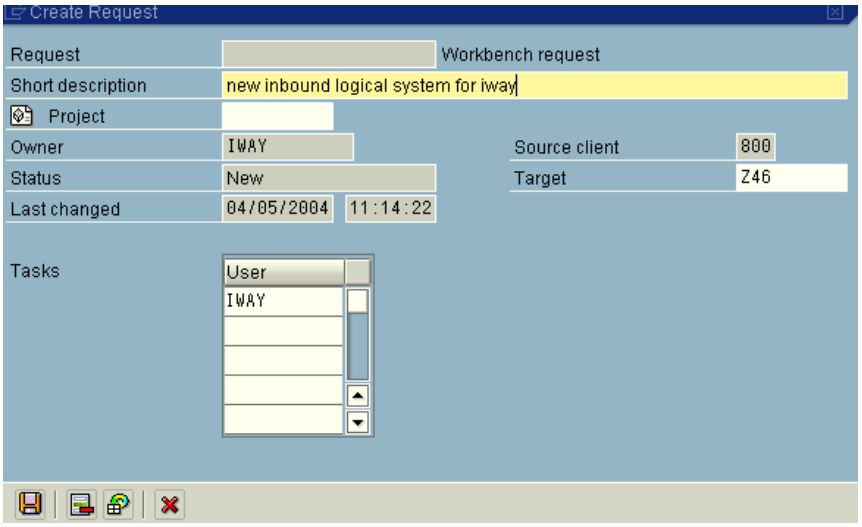

5. Type the Logical System, for example, ORACLETDS, in the **Log.System** column and type a description in the **Name** column.
6. Click **Save.**

   The Prompt for Workbench request dialog box opens.



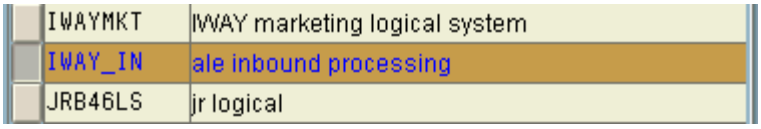

7. Click the **Create Request** icon.

The Create Request dialog box opens.




8. Type a name and description for your request and click **Save.**

   The logical system you configured, for example, ORACLETDS, is now added to the list.



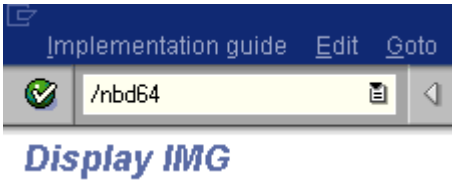

## Configuring a Distribution Model

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.

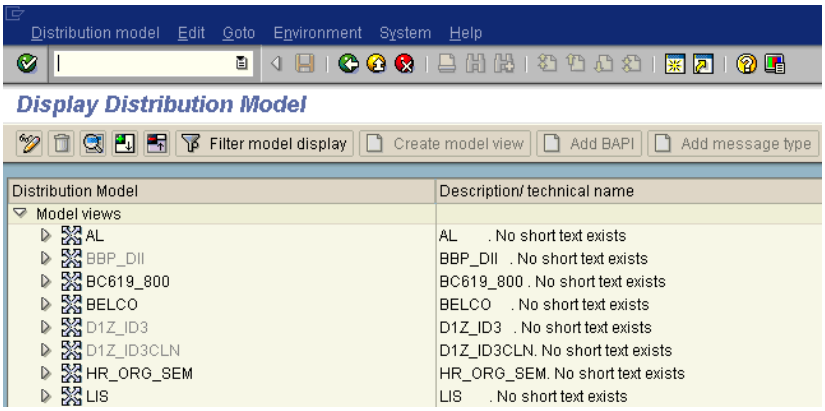
### Defining a Distribution Model

To define a distribution model:

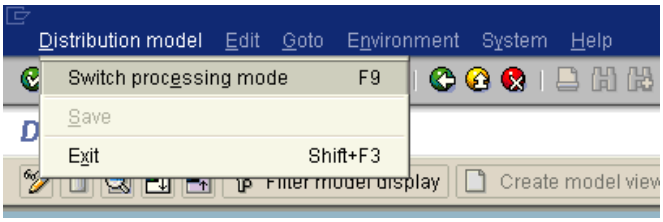
1. Execute the **bd64** transaction.




   The Display Distribution Model window opens.

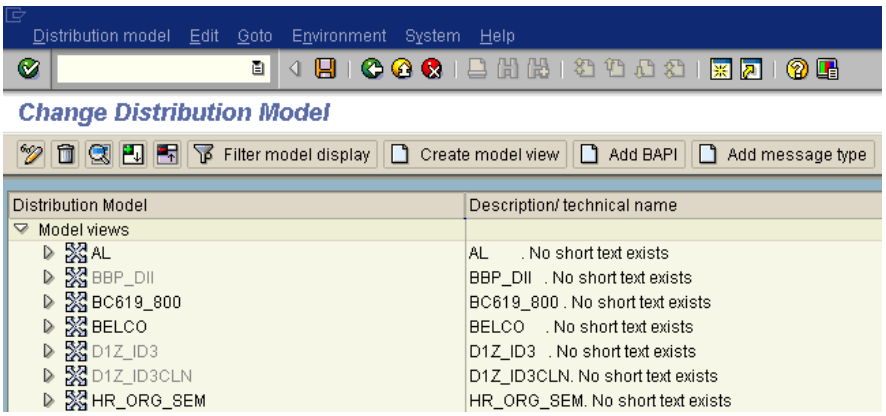



2. Click **Distribution Model** from the menu bar.



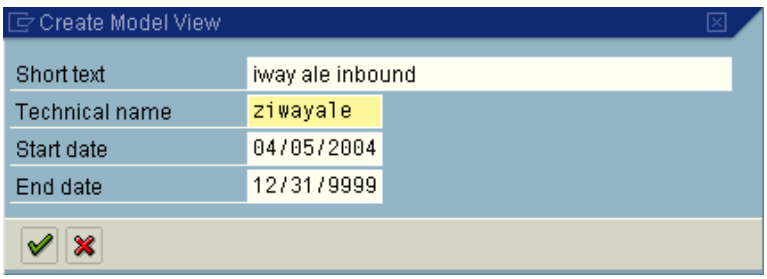

3. Select **Switch processing mode.**

   The Display Distribution Model window is switched to Change Distribution Model.



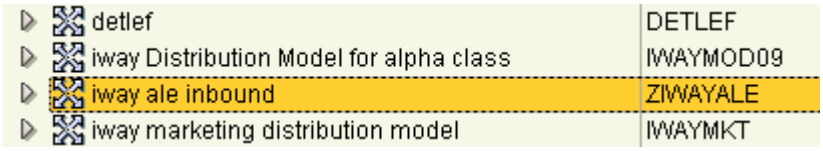

4. Click the **Create model view** button.

   The Create Model View dialog box opens.



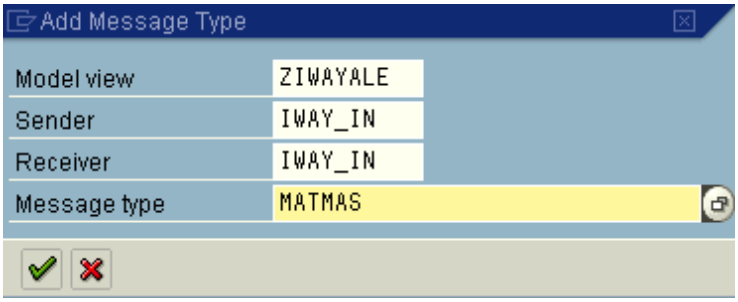
5. Type a model view name in the **Short text** field and a name in the **Technical name** field, which also serves as a description.
6. Click the **check mark** button to enter the information.

   You are returned to the main Change Distribution Model window. The distribution model you configured is now added to the list.




7. Click the **Add message type** button.

   The Add Message Type dialog box opens.




   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** button to enter the information.

   You are returned to the main Change Distribution Model window.

9. Click **Save.**

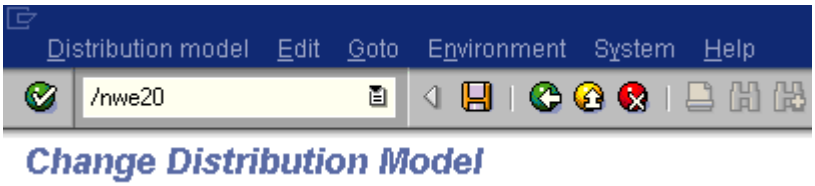
## Defining a Partner Profile

Partner profiles are a prerequisite for data exchange. This involves defining who can exchange messages with the SAP system and using which port.

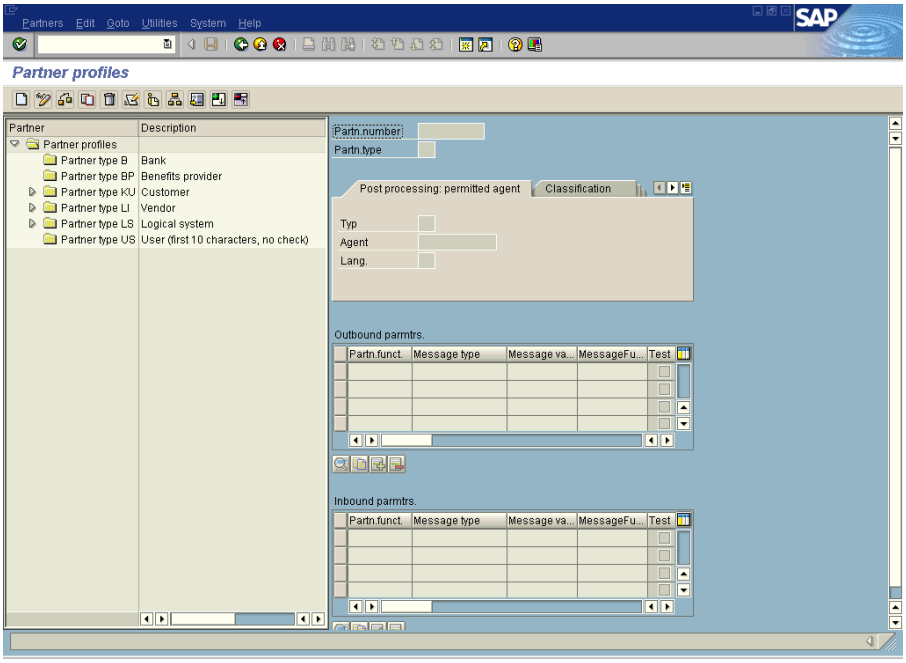
### Defining a Partner Profile

To define a partner profile:

1. Execute the **we20** transaction.

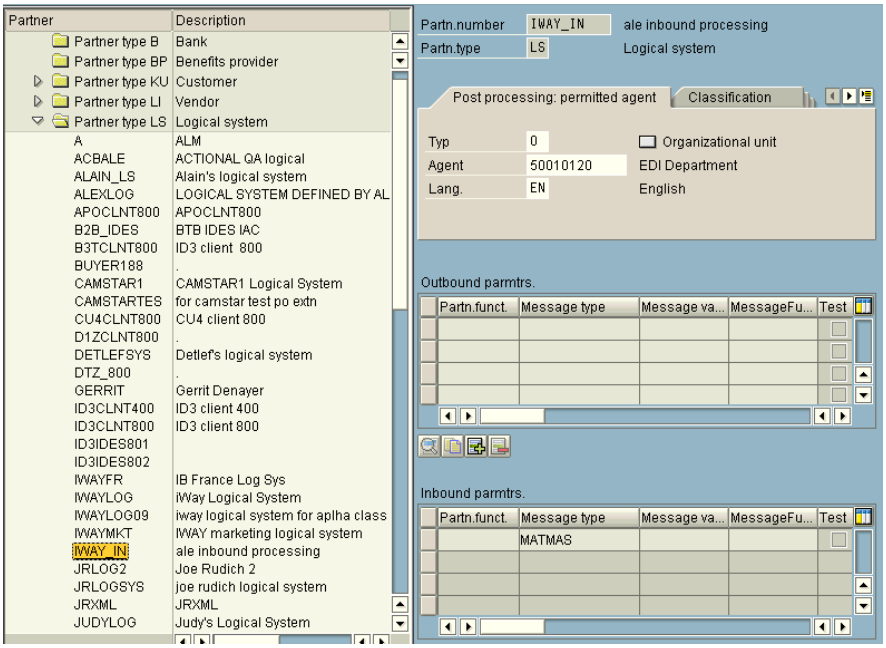



The Partner profiles window opens.



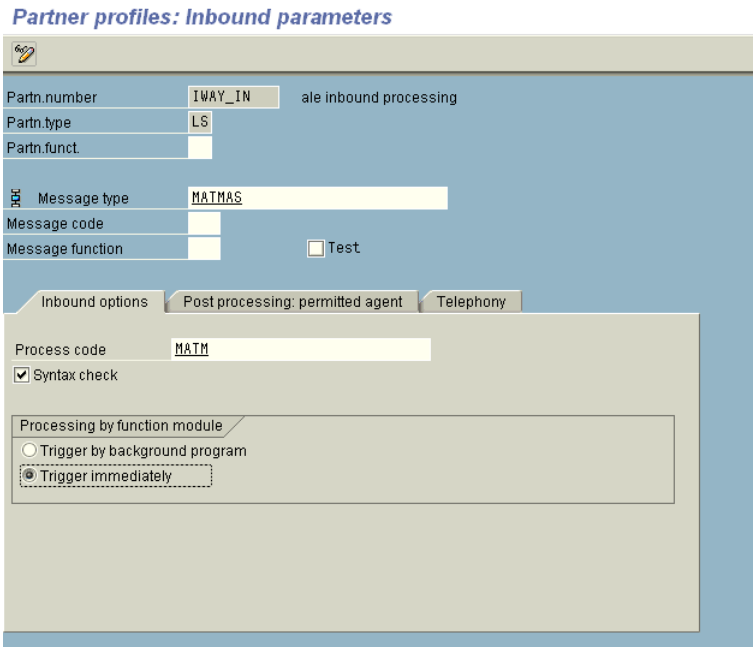

2. In the left pane, expand **Partner type LS** and select the logical system you configured from the list, for example, ORACLETDS.

   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 opens.




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 events that 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 execute the function that is provided by an RFC server. The functions that can be executed 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, 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.

### 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)

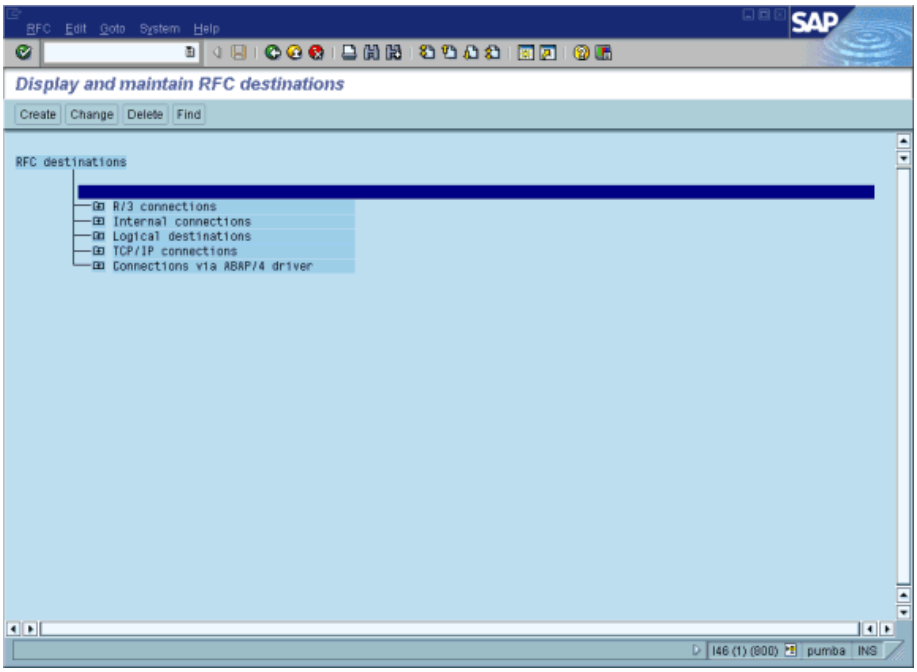
The RFC destination is a symbolic name (for example, ORACLETDS) that is used to direct events to a target system, masking the program ID. The Program ID is configured in both SAPGUI and the event adapter.

### Registering Your Program ID

To register your program ID:

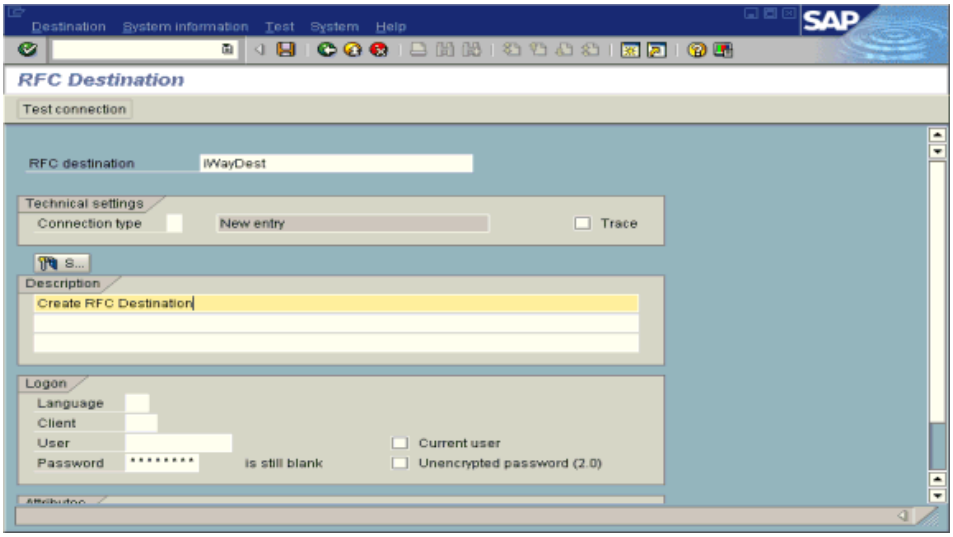
1. Launch the SAP GUI and log in to the SAP system.
2. Select **Tools, Administration, Network,** and then **RFC destination.**
3. Execute the **SM59** transaction.

   The Display and maintain RFC destinations window opens.




4. Select **TCP/IP connections** and click **Create.**

   The RFC Destination window opens.




   a. In the **RFC destination** field, type a name, for example, ORACLETDS.

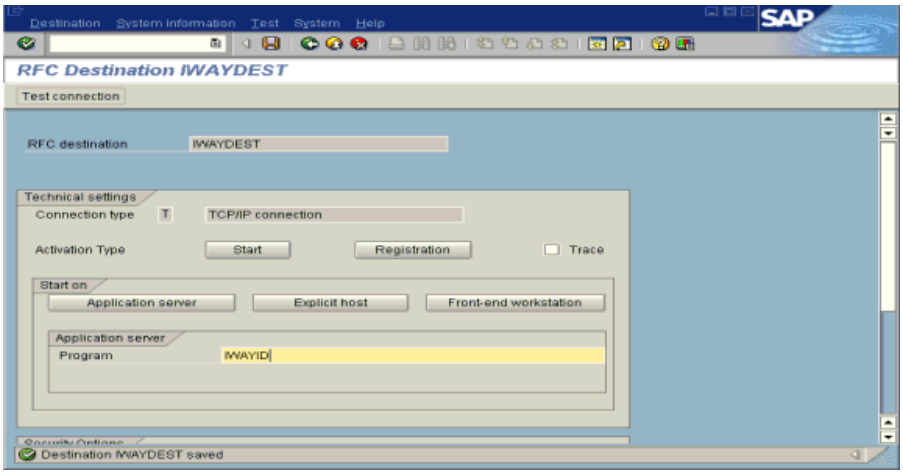
      The value you enter in this field is case sensitive.

   b. In the **Connection type** field, enter **T** for destination type TCP/IP.

   c. In the **Description** field, type a brief description.

5. Click **Save** from the tool bar or select **Save** from the Destination menu.

   The RFC Destination ORACLETDS window opens.




a. For the **Activation Type**, click the **Registration** button.

b. In the **Program** field, type **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 the OracleAS Adapter for SAP are communicating.
9. Click **TestConnection.**

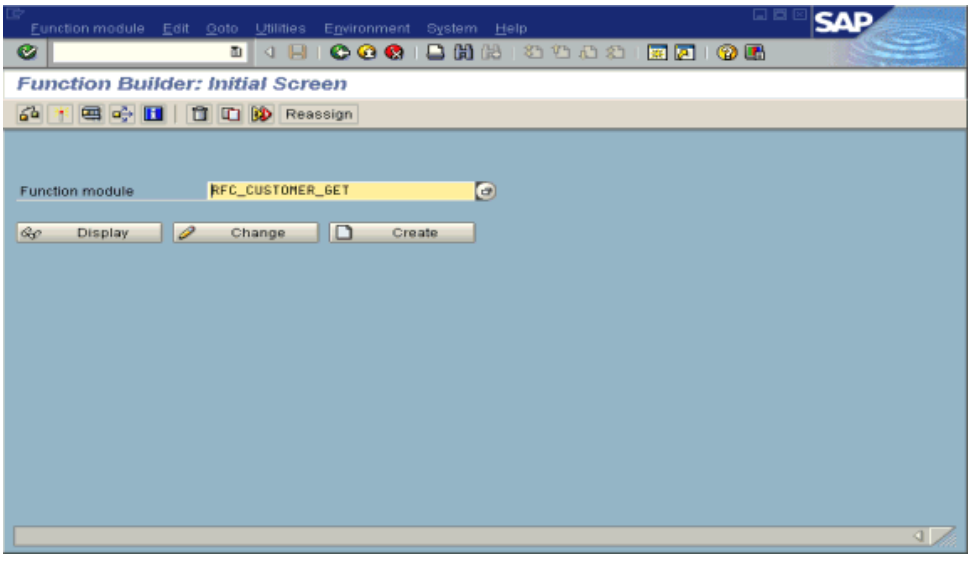
## Testing the SAP Event Adapter

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. See "Registering Your Program ID" on page A-10 for more information on RFC destination.

### Testing the SAP Event Adapter by Sending an RFC or a BAPI Manually

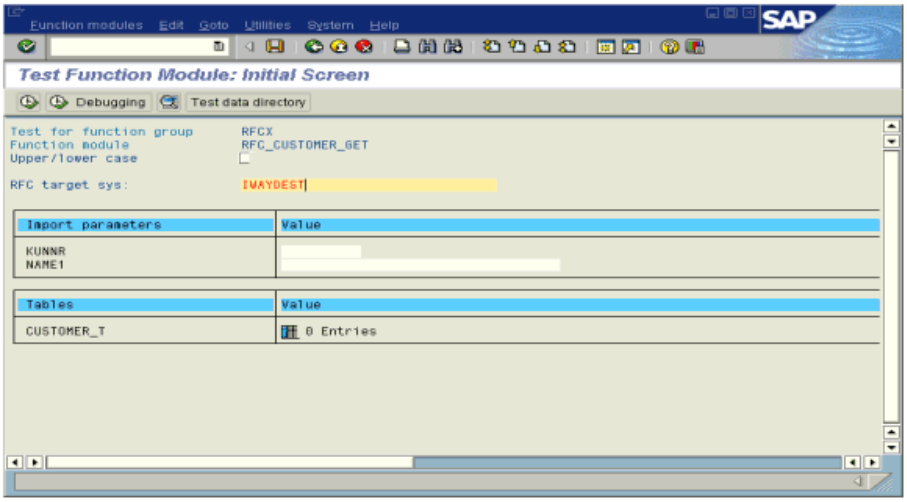
To test the SAP event adapter:

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 opens.




6. Enter data into the SAP GUI and click the **Execute** button.

   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.

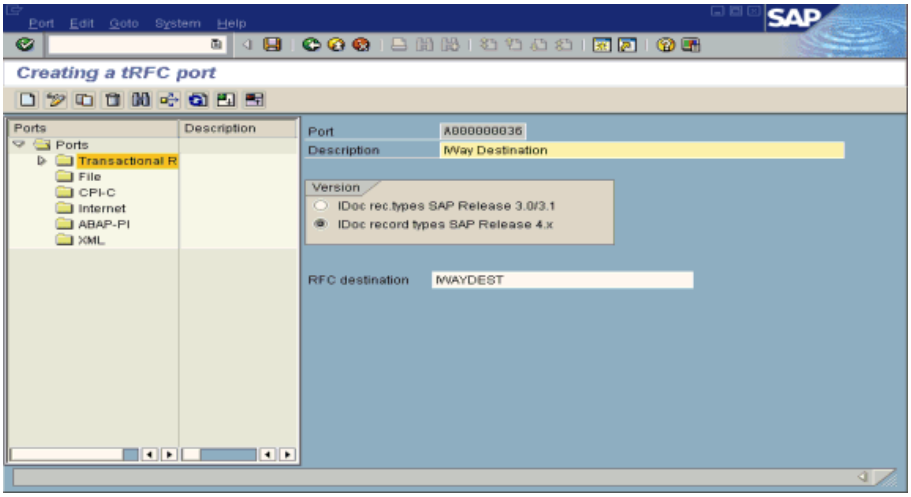
### Defining a Port

To define a port:

1. In the ALE configuration, choose **Tools, Business Communications, IDocs Basis, IDoc,** and then **Port Definition.**

You can also execute the WE21 transaction.

The Creating a tRFC port window opens.




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

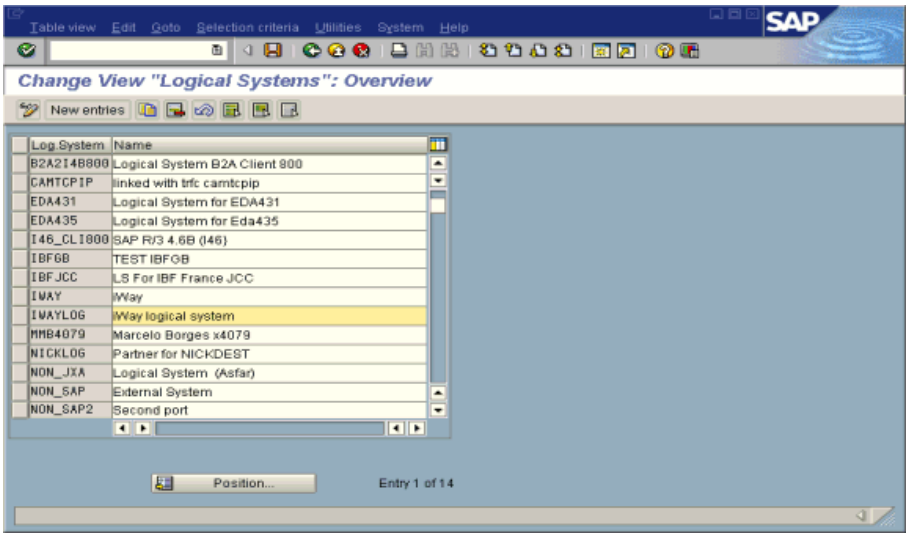
One type of partner is a logical system. A logical system manages one or more RFC destinations.

### Creating a Logical System

To create a logical system called ORACLETDS:

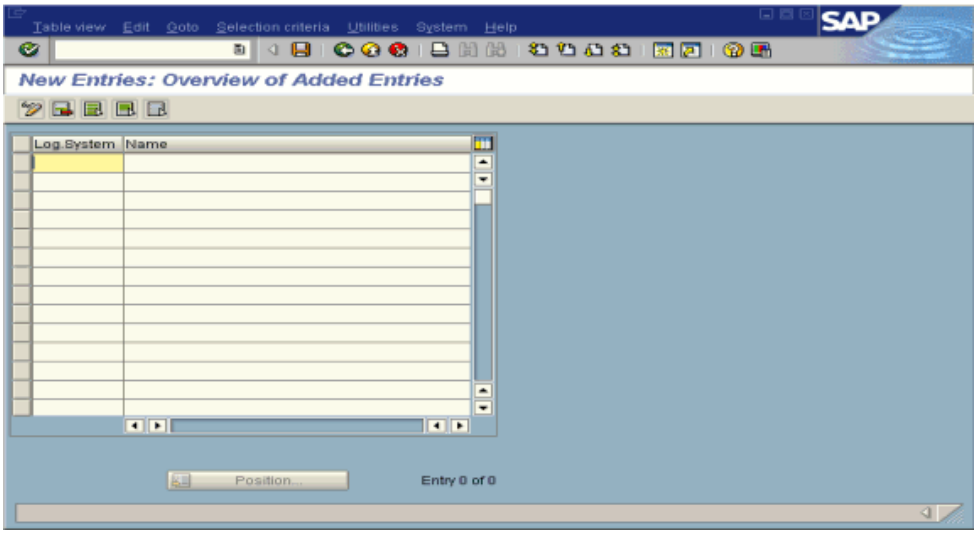
1. In the ALE configuration, enter the area menu selection **SALE** transaction.
2. Select **SAP Reference IMG.**
3. Expand the following nodes: **Basis Components, Application Link Enabling (ALE), Sending and Receiving Systems, Logical Systems,** and **Define Logical System.**
4. Click the check mark beside **Define Logical System.**

   The Change View "Logical Systems": Overview window opens displaying a list of logical systems and their names.




5. Click **New entries.**

   The New Entries: Overview of Added Entries window opens 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, type 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.

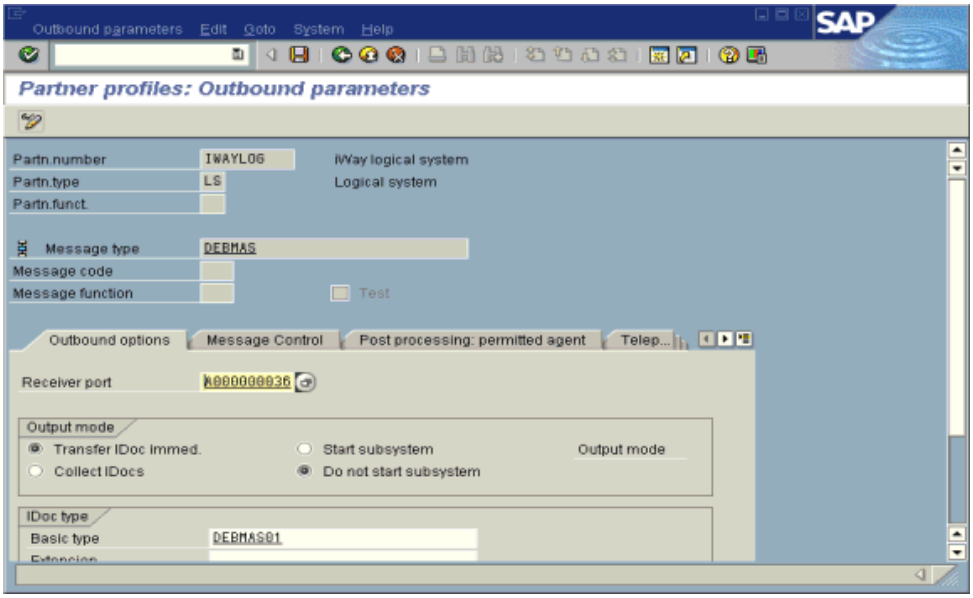
### Creating a Partner Profile

To create a partner profile:

1. In SAP GUI, choose **Tools, Business Communication, IDoc Basis,** and **Partner profile.**

   You can also execute the WE21 transaction.

   The Partner profiles: Outbound parameters window opens with fields for specifying details for the partner profile.




   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.**

   Partn.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.

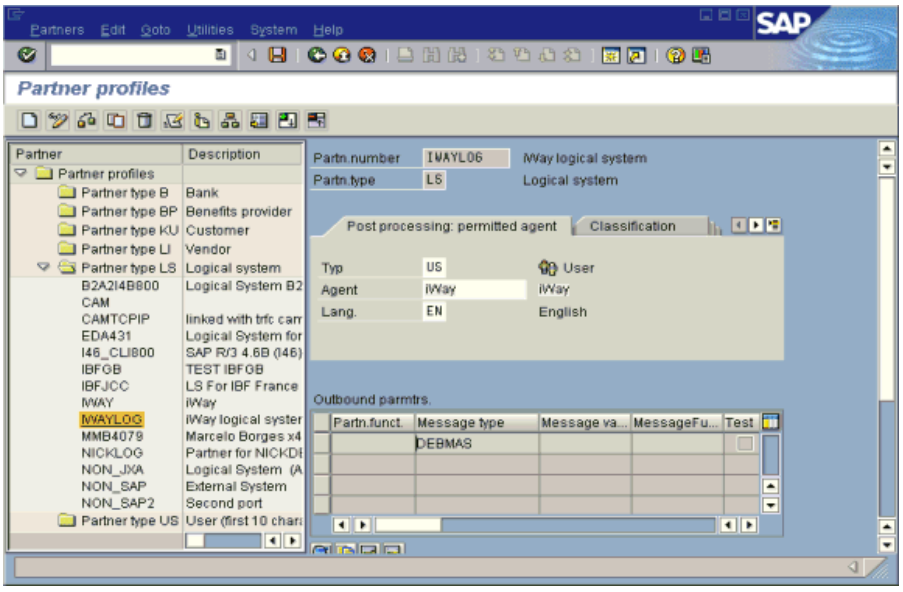
   a. Depending on your performance requirements, click **Transfer IDoc Immed** or **Collect IDocs.**

   b. For the IDoc, type a message type, for example, DEBMAS.

   c. Type a receiver port, for example, A000000036.

7. Click **Save** to save the session.

   The Partner profiles summary window opens and displays 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 execute the BD64 transaction.

   The Display Distribution Model window opens.

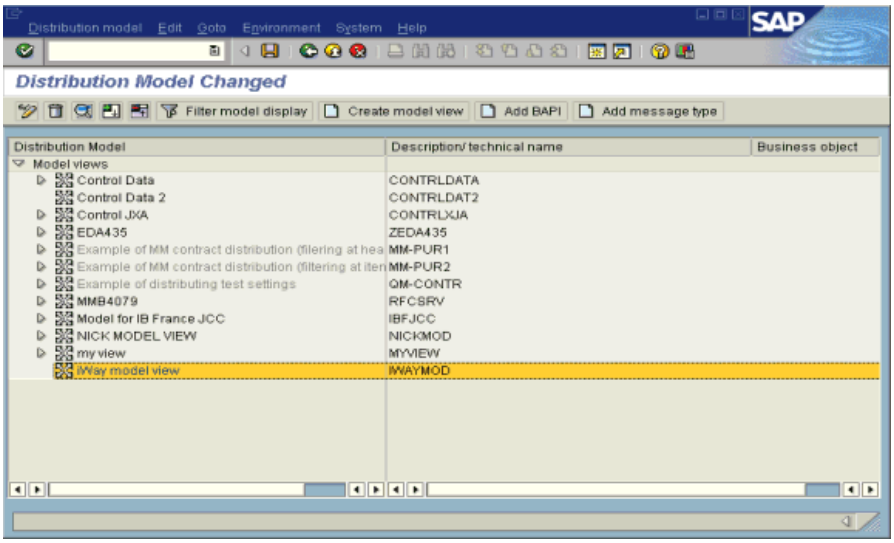
2. Select **Create model view.**

   If required, switch the processing mode to edit within Distribution Model/Switch Processing Mode.

3. Type a short text string and a technical name for your new model view.

4. Click **Save.**

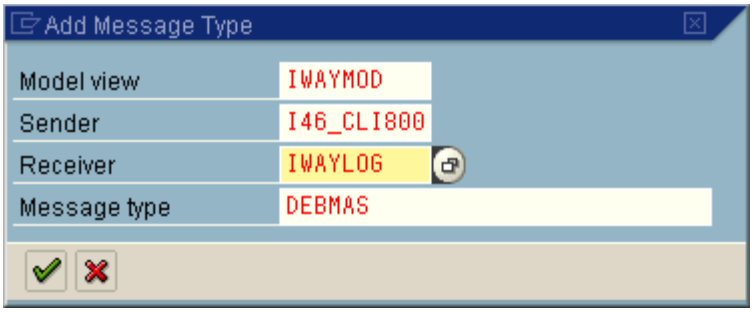
   The Distribution Model Changed window opens with a tree structure of the distribution model.




a. In the Distribution Model tree, select a new model view.

b. On the right, select **Add message type.**

The Add Message Type box opens displaying fields for specifying the sender and receiver of the message, as well as the message type.




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.

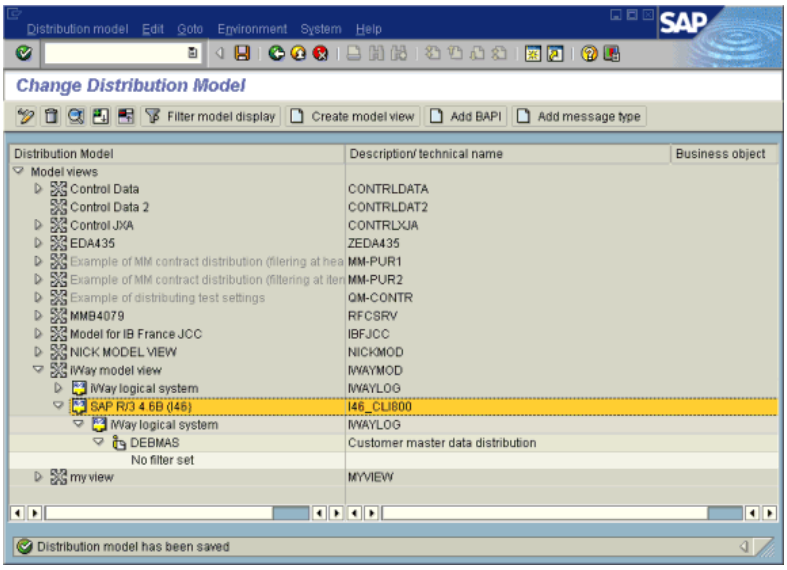
b. In the **Receiver** field, provide the logical system, for example, ORACLETDS.

c. In the **Message type** field, provide the type of IDoc, for example, DEBMAS.

5. Click the check mark.

6. Click **Save.**

The Change Distribution Model window opens displaying 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.

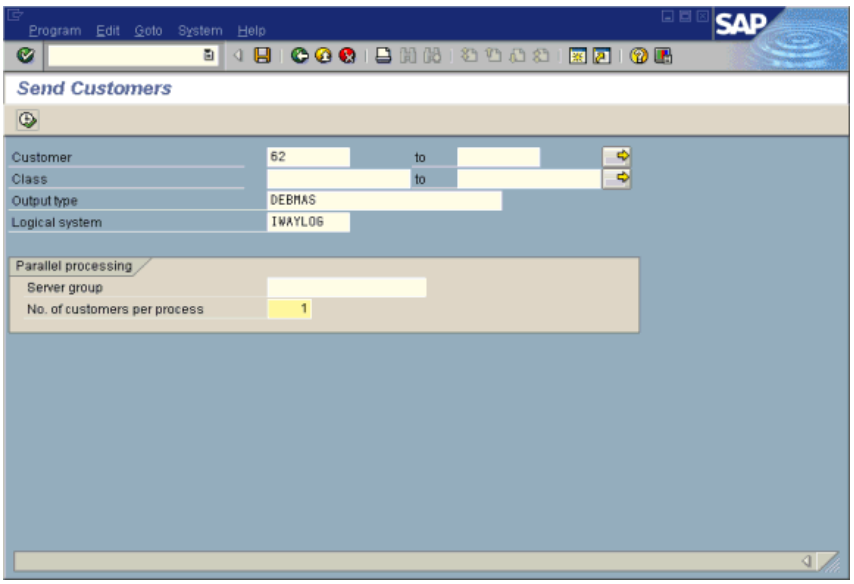
## Testing the SAP ALE Configuration

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:

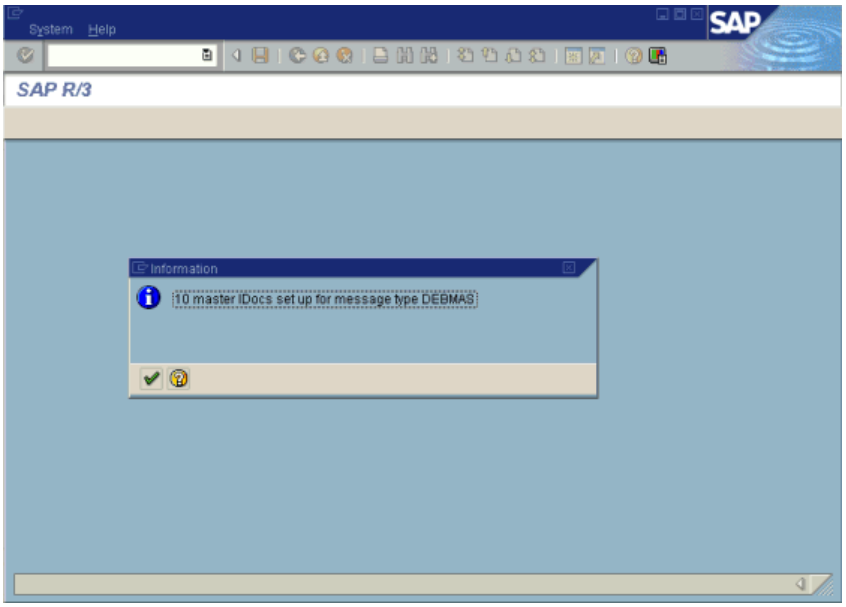
1. In the Send Customers window, type the IDoc message type, for example, DEBMAS in the **Output type** field.




2. In the **Logical system** field, type 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 displays.



# 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



## W



## X