Oracle® Application Server InterConnect

Adapter for SAP R/3 Installation and User's Guide

10*g* (9.0.4)

Part No. B10408-01

August 2003

ORACLE®

Oracle Application Server InterConnect Adapter for SAP R/3 Installation and User's Guide, 10*g* (9.0.4)

Part No. B10408-01

Contents

3 Supported SAP Adapter Interfaces

4 Application Link Enabling

5 Remote Function Call

Send Us Your Comments

Oracle Application Server InterConnect Adapter for SAP R/3 Installation and User's Guide, 10*g* (9.0.4)

Part No. B10408-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. 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?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, 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, M/S 1op6
  Redwood Shores, CA 94065
  USA

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

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

Preface

This preface contains these topics:

Intended Audience

This guide is intended for those who perform the following tasks:

- install applications
- maintain applications

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 Corporation 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 Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Organization

This document contains:

Chapter 1, "Introduction"

This chapter describes the SAP adapter and the hardware and software requirements.

Chapter 2, "Installation and Configuration"

This chapter provides installation and configuration for the SAP adapter.

Chapter 3, "Supported SAP Adapter Interfaces"

This chapter describes the supported interfaces for the SAP adapter.

Chapter 4, "Application Link Enabling"

This chapter describes applicaiton link enabling for the SAP adapter.

Chapter 5, "Remote Function Call"

This chapter describes remote function call for the SAP adapter.

Chapter 6, "Runtime"

This chapter provides runtime information for theSAP adapter.

Related Documentation

For more information, see these Oracle resources:

- *Oracle Application Server InterConnect User's Guide*
- *Oracle Application Server InterConnect Installation Guide*
- *Oracle Application Server InterConnect Adapter Configuration Editor 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://otn.oracle.com/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://otn.oracle.com/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 Microsoft 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.	*Oracle9i Database Concepts* 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. You can back up the database by using the `BACKUP` command. Query the `TABLE_NAME` column in the `USER_TABLES` data dictionary view. Use the `DBMS_STATS.GENERATE_STATS` procedure.

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. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter `sqlplus` to open SQL*Plus. The password is specified in the `orapwd` file. Back up the datafiles and control files in the `/disk1/oracle/dbs` directory. The `department_id`, `department_name`, and `location_id` columns are in the `hr.departments` table. Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true`. Connect as `oe` user. The `JRepUtil` class implements these methods.
`lowercase italic monospace (fixed-width) font`	Lowercase italic monospace font represents placeholders or variables.	You can specify the *`parallel_clause`*. Run `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}` `[COMPRESS | NOCOMPRESS]`

Convention	Meaning	Example
`...`	Horizontal ellipsis points indicate either: ■ That we have omitted parts of the code that are not directly related to the example ■ That you can repeat a portion of the code	`CREATE TABLE ... AS subquery;` `SELECT col1, col2, ... , coln FROM employees;`
. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	
Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	`acctbal NUMBER(11,2);` `acct CONSTANT NUMBER(4) := 3;`
Italics	Italicized text indicates placeholders or variables for which you must supply particular values.	`CONNECT SYSTEM/system_password` `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;`
		`SELECT * FROM USER_TABLES;`
		`DROP TABLE hr.employees;`

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

Conventions for Microsoft Windows Operating Systems

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

Convention	Meaning	Example
Choose Start >	How to start a program.	To start the Oracle 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`
`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>`

	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\"`
		`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`

Convention	Meaning	Example
ORACLE_HOME and *ORACLE_BASE*	In releases prior to Oracle8*i* release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level *ORACLE_HOME* directory that by default used one of the following names: ■ `C:\orant` for Windows NT ■ `C:\orawin95` for Windows 95 ■ `C:\orawin98` for Windows 98 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 Oracle9*i* release 1 (9.0.1) on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is `C:\oracle\ora90`. The Oracle home directory is located directly under *ORACLE_BASE*. All directory path examples in this guide follow OFA conventions.	Go to the *ORACLE_BASE*`\`*ORACLE_HOME*`\rdbms\admin` directory.

1

Introduction

Oracle connects to SAP through the Oracle Application Server InterConnect Adapter for SAP R/3 (SAP adapter). This chapter provides an overview of the SAP adapter.

What is SAP?

SAP is a vendor of enterprise management software. The business application, R/3, automates and manages enterprise business process; for example, inventory control, customer master file maintenance, invoicing, and accounting. It is both a business application product and a large-scale application development platform.

Required Software

The following lists the system to which the SAP adapter connects:

Table 1–1 List of systems to which the SAP adapter connects

Component Support	Required Software
SAP	4.6B

See Also: *Oracle Application Server InterConnect Installation Guide* for hardware requirements

Supported Platforms

The following platforms support the SAP adapter:

- Windows NT 4.0 with SP 6 or above
- Windows 2000 with SP 1 or above
- Solaris 8
- HP-UX 11.0

2

Installation and Configuration

This chapter describes installation and configuration of the SAP adapter. This chapter discusses the following topics:

- Installing the SAP Adapter Adapter
- SAP Adapter Configuration
- Starting the SAP Adapter
- Stopping the SAP Adapter

Installing the SAP Adapter Adapter

This section contains these topics:

Preinstallation Tasks

The SAP adapter must be installed in one of the following Oracle homes:

- An existing Oracle Application Server InterConnect (OracleAS InterConnect) Oracle home for this release
- A new Oracle home (the installer creates this for you)

Consult the following guides before proceeding with SAP adapter installation:

- *Oracle Application Server InterConnect Installation Guide*, which includes information on:
  - CD-ROM mounting
  - Oracle Universal Installer startup
  - OracleAS InterConnect software, hardware, and system requirements
  - OracleAS InterConnect installation

Note: OracleAS InterConnect Hub is installable through the OracleAS InterConnect Hub installation type. You must install the OracleAS InterConnect Hub before proceeding with the SAP adapter installation.

Installation Tasks

To install the SAP adapter:

1. On the Available Product Components page of the OracleAS InterConnect installation, select SAP adapter, then select **Next**.

   Consider the following scenarios:

- If installing the SAP adapter in an independent Oracle home, please make sure that the OracleAS InterConnect Hub has been installed, not necessarily in the same Oracle home. Continue to step 2.
- If installing the SAP adapter in an existing Oracle home, please make sure that it is a home directory to one of the OracleAS InterConnect component. Continue to step 3.

Note: The hub database information, such as the SID, host, port, and username/password from the Hub installation is needed for step 2.

---

2. If installing OracleAS InterConnect for the first time on this machine, complete the following steps to enter the hub database information:
   a. On the Welcome page, select **Next**. The Database Configuration page displays. Enter information in the following fields:
      * Host Name—The host name of the machine where the hub database is installed.
      * Port Number—The TNS listener port for the hub database.
      * Database SID—The SID for the hub database.
   b. Click **Next**. The Database User Configuration page displays. Enter information in the following fields:
      * User Name—The hub database user name. Make sure the OracleAS InterConnect Hub is installed. If the Hub is not installed, complete the installation and note the user name and password.
      * Password—The password for the hub database user.
3. Click **Next**. The Adapter Configuration page displays. Enter the application to be defined or already defined in iStudio in the Application Name field. White spaces or blank spaces are not permitted. The default value is `mySAPApp`.
4. Click **Next**. Complete the fields for any other components selected for installation, such as other adapters. When finished, the Summary page displays.
5. Click **Install** to install the SAP adapter and other selected components. The SAP adapter is installed in the following directory:

Platform	Directory
Windows	%ORACLE_HOME%\oai\9.0.4\adapters*Application*
UNIX	$ORACLE_HOME/oai/9.0.4/adapters/*Application*

Application is the value you specified in Step 3 on page 2-3.

6. Click **Exit** at the End of Installation page to exit the SAP adapter installation.

Post Installation Tasks

Enabling iStudio

After installing the SAP adapter and the iStudio, complete the following iStudio post installation steps to fully enable the SAP adapter.

> **See Also:** *Oracle Application Server InterConnect Installation Guide* for information on installing iStudio along with OracleAS InterConnect Development Kit

1. Update the PATH environment variable to include the following directory:

   On Windows: *ORACLE_HOME*\oai\9.0.4\bin

   On UNIX: *ORACLE_HOME*/oai/9.0.4/bin

   This procedure is required to run the utilities in this directory.

2. The SAP adapter requires the librfc32.dll SAP library for browsing the SAP system in iStudio and for run time. If the SAP graphical interface is installed on a machine, the librfc32.dll can be found in the following directories:

   ```
   <SAP install directory>SAPpc\SapGui\RFCSDK\lib
   <SAP install directory>SAPpc\SapGui\RFCSDK\bin
   ```

   The version information for the library is librfc32.dll 4640.5.734.3319. Copy this library file to the following directory:

   On Windows: *ORACLE_HOME*\oai\9.0.4\bin

   On UNIX: *ORACLE_HOME*/oai/9.0.4/bin

If you do not have the library available at your site, please refer to R/3 note number 0413708. In this note you can find the information regarding procurement of the libraries.

3. To enable dynamic access to IDocs and enhanced use of RFCs, use the files provided in the following directory and upload the source code on to the SAP server:

   On Windows: *ORACLE_HOME*\oai\9.0.4\bin

   On UNIX: *ORACLE_HOME*/oai/9.0.4/bin

   This directory has the following folders:

   - a. ale_files—The files in this folder are used on the SAP server to enable dynamic access to IDocs at both design time and runtime. Without uploading the source code in these files, you will not be able to dynamically access IDocs. In that case you need to manually download the IDoc definitions from the SAP server.

     See Also: File IdocBrow.txt for instructions on accessing IDocs

   - b. rfc_files—The files in this folder are used on the SAP server or enhanced access to RFCs at both design time and runtime. RFC browsing and runtime calls will be slower if the source code in these files are not uploaded to the SAP server.

     See Also: File rfcbrows.txt for instructions on accessing RFCs

- Set the configuration settings for the SAP adapter using the Configuration Editor before using the SAP adapter for runtime. The Configuration Editor is a Java application and is launched by running the configeditor.bat file in the ORACLE_HOME/oai/9.0.4/config/ directory:

  On Windows:*ORACLE_HOME*\oai\9.0.4\config

  On UNIX: *ORACLE_HOME*/oai/9.0.4/config

Registering the License for the SAP Adapter (Windows only)

Before using the SAP adapter, or if using the browser in iStudio, you need to register the license using the license registration tool. This tool uses the following files:

- `licreg.exe`—Located in the *`ORACLE_HOME`*`\oai\9.0.4\bin` directory.
- `acboai.lic`—The license file located in the *`ORACLE_HOME`*`\oai\9.0.4\config` directory.

To register the license, complete the following:

1. Update the `PATH` environment variable to include the *`ORACLE_HOME`*`\oai\9.0.4\bin` directory. This is required for running the utilities in this directory.
2. Double click on `licreg.exe` to display the License Manager dialog.
3. Navigate to the *`ORACLE_HOME`*`\oai\9.0.4\config` directory.
4. Select `acboai.lic` and click **OK**.

`Licreg.exe` silently registers the license. To verify the registration, from a command prompt, launch `logdump.exe` located in *`ORACLE_HOME`*`\oai\9.0.4\bin` directory. This prints the log messages regarding license registration.

SAP Adapter Configuration

Table 2–2, Table 2–3, and Table 2–4 describe executable files, configuration files, and directories. These files and directories are accessible from the directory shown in Table 2–1:

Table 2–1 Advanced Queuing Adapter Directory

On...	Go to...
UNIX	`$ORACLE_HOME/oai/9.0.4/adapters/`*`Application`*
Windows	`%ORACLE_HOME%\oai\9.0.4\adapters\`*`Application`*

Table 2–2 Executable Files

File	Description
`start.bat` (Windows)	Takes no parameters, starts the adapter.
`start` (UNIX)	Takes no parameters, starts the adapter.
`stop.bat` (Windows)	Takes no parameters; stops the adapter.
`stop` (UNIX)	Takes no parameters; stops the adapter.
`ignoreErrors.bat` (Windows)	If an argument is specified, then the given error code will be ignored. If no argument is specified, then all error codes specified in the `ErrorCodes.ini` will be ignored.
`ignoreErrors` (UNIX)	If an argument is specified, then the given error code will be ignored. If no argument is specified, then all error codes specified in the `ErrorCodes.ini` will be ignored.

Note: Running `stop.sh` on UNIX does not stop the SAP adapter. Use `<CTRL>C` to stop the SAP adapter.

Table 2–3 Configuration Files

File	Description
`ErrorCodes.ini` (Windows and UNIX)	Should contain one error code per line.
`adapter.ini` (Windows and UNIX)	Consists of all the initialization parameters which the adapter reads at startup. Refer to Appendix A for a typical `adapter.ini` file.

Table 2–4 Directories

File	Description
`persistence`	The messages are persisted in this directory. This directory or its contents should not be edited.
`logs`	The logging of adapter activity is done in subdirectories of the log directory. Each new run of the adapter creates a new subdirectory in which logging is done in an `oailog.txt` file.

Using the Application Parameter

Adapters do not have integration logic. The SAP adapter has a generic transformation engine that processes metadata from the repository as runtime instructions to do transformations. The application defines for an adapter what its capabilities are. For example, it can define what messages it can publish, what messages it can subscribe to, and what are the transformations to perform. The application parameter allows the adapter to become smart in the context of the application to which it is connected. It allows the adapter to retrieve from the repository only that metadata that is relevant to the application. The application parameter must match the corresponding application that will be defined in iStudio under the Applications folder.

If you are using pre-packaged metadata, after importing the pre-packaged metadata into the repository, start up iStudio to find the corresponding application (under the Applications folder in iStudio) to use as the application for the adapter you are installing (unless the package you are using provides directions for what the application should be).

Adapter.ini Initialization Parameter File

This section contains these topics:

Hub.ini Parameters

The SAP adapter connects to the hub database using parameters from the `hub.ini` file located in the hub directory. The following table lists the parameter name, a description for each parameter, the possible and default values, and an example.

Table 2–5 Hub.ini Parameters

Parameter	Description	Example
hub_username	The name of the hub database schema (or username). The default value is oaihub904.	hub_username=oaihub904
hub_password	The password for the hub database user. There is no default value. You input the hub_password value during installation.	hub_password=manager
hub_host	The name of the machine hosting the hub database. There is no default value. You input the hub_host value during installation.	hub_host=mpmypc
hub_instance	The system identification number (SID) of the hub database. There is no default value. You input the hub_instance value during installation.	hub_instance=orcl
hub_port	The transparent network services (TNS) listener port number for the HUB database instance. There is no default value. You input the hub_port value during installation.	hub_port=1521
repository_name	The valid name of the repository this adapter talks to. The default value is InterConnectRepository.	repository_name=InterConnectRepository

Real Application Clusters-specific Hub.ini Parameters

When a hub is installed on a Real Application Clusters (RAC) database, parameters listed in Table 2–6 represent information on additional nodes used for connection and configuration. These parameters are added on top of the default parameters which represent the primary node. In Table 2–6, *x* represent the node number, which varies between 2 and the number of nodes. For example, if the Real Application Clusters setup contains 4 nodes, *x* can take a value between 2 and 4.

Table 2–6 Real Application Cluster-specific hub.ini Parameters

Parameter	Description	Example
hub_num_nodes	Number of nodes in Real Application Clusters.	hub_num_nodes=4
hub_host*x*	The host where the Real Application Clusters database is installed.	hub_host2=dsunram13
hub_instance*x*	The instance on the respective node.	hub_instance2=orcl2
hub_port*x*	The port on which the listener is listening.	hub_port2=1521

Agent Connection Parameters

The SAP adapter connects to the spoke application using parameters from the `adapter.ini` file. Table 2–7 lists the parameter name, description, the possible and default values, and example of each parameter.

Table 2–7 Adapter.ino Parameters

Parameter	Description	Example
`application`	The name of the application this adapter connects to. This must match with the name specified in iStudio during creating of metadata. Any alphanumeric string can be used. There is no default value.	`application=aqapp`
`partition`	The partition this adapter handles as specified in iStudio. Any alphanumeric string is a possible value. There is no default value.	`partition=germany`
`instance_number`	To have multiple adapter instances for the given application with the given partition, each adapter should have a unique instance number. Possible values are any integer greater than 1. There is no default value.	`instance_number=1`
`agent_log_level`	Specifies the amount of logging necessary. Possible values are: 0=errors only 1=status and errors 2=trace, status, and errors The default value is 1.	`agent_log_level=2`
`agent_ subscriber_name`	The subscriber name used when this adapter registers its subscription. The possible value is a valid Oracle Advanced Queuing subscriber name and there is no default value.	`agent_subscriber_ name=aqapp`
`agent_message_ selector`	Specifies conditions for message selection when registering its subscription with the hub. The possible value is a valid Oracle Advanced Queuing message selector string. There is no default value.	`agent_message_ selector=recipient_ list like '%aqapp,%'`
`agent_reply_ subscriber_name`	The subscriber name used when multiple adapter instances for the given application with the given partition are used. Optional if there is only one instance running. The possible value is application name (`parameter: application`) concatenated with instance number (`parameter: instance_number`). There is no default value.	If `application=aqapp, instance_number=2,` then, `agent_reply_ subscriber_ name=aqapp2`

Table 2–7 Adapter.ino Parameters

Parameter	Description	Example
`agent_reply_message_selector`	Used only if multiple adapter instances for the given application with the given partition. The possible value is a string built using concatenating application name (`parameter:application`) with instance number (`parameter:instance_number`). There is no default value.	If `application=aqapp, instance_number=2`, then `agent_reply_message_selector=receipient_list like '%,aqapp2,%'`
`agent_tracking_enabled`	Specifies if message tracking is enabled. Set to false to turn off all tracking of messages. Set to true to track messages with tracking fields set in iStudio. Possible values are `true` or `false`. The default value is `true`.	`agent_tracking_enabled=true`
`agent_throughput_measurement_enabled`	Specifies if throughput measurement is enabled. Set to true to turn on all throughput measurements. Possible values are `true` or `false`. The default value is `true`.	`agent_throughput_measurement_enabled=true`
`agent_use_custom_hub_dtd`	Specifies if a custom DTD should be used for the common view message when handing it to the hub. By default adapters use an OracleAS InterConnect-specific DTD for all messages sent to the hub as other OracleAS InterConnect adapters will be retrieving the messages from the hub and know how to interpret them. Set to `true` if for every message, the DTD imported for the message of the common view is to be used instead of the OracleAS InterConnect DTD. Only set to `true` if a OracleAS InterConnect adapter is not receiving the messages from the hub. Possible values are `true` or `false`. There is no default value.	`agent_use_custom_hub_dtd=false`
`agent_metadata_caching`	Specifies the metadata caching algorithm. Possible values are: ▪ `startup`—Cache everything at startup. This may take a while if there are a lot of tables in the repository. ▪ `demand`—Cache metadata as it is used. ▪ `none`—No caching. This slows down performance. The default value is `demand`.	`agent_metadata_caching=demand`

Table 2–7 Adapter.ino Parameters

Parameter	Description	Example
`agent_dvm_table_caching`	Specifies the DVM caching algorithm. Possible values are: ■ `startup`—Cache all DVM tables at startup. This may take a while if there are a lot of tables in the repository. ■ `demand`—Cache tables as they are used. ■ `none`—No caching. This slows down performance. The default value is `demand`.	`agent_dvm_table_caching=demand`
`agent_lookup_table_caching`	Specifies the lookup table caching algorithm. Possible values are: ■ `startup`—Cache all lookup tables at startup. This may take a while if there are a lot of tables in the repository. ■ `demand`—Cache tables as they are used. ■ `none`—No caching. This slows down performance. The default value `demand`.	`agent_lookup_table_caching=demand`
`agent_delete_file_cache_at_startup`	With any of the agent caching methods enabled, metadata from the repository is cached locally on the file system. Set this parameter to `true` to delete all cached metadata on startup. Note: After changing metadata or DVM tables for this adapter in iStudio, you must delete the cache to guarantee access to the new metadata or table information. Possible values are `true` or `false`. The default value is `false`.	`agent_delete_file_cache_at_startup=false`
`agent_max_ao_cache_size`	Specifies the maximum number of application objects' metadata to cache. Possible values are any integer greater than 1. The default value is `200`.	`agent_max_ao_cache_size=200`
`agent_max_co_cache_size`	Specifies the maximum number of common objects' metadata to cache. Possible values are any integer greater than 1. The default value is `100`.	`agent_max_co_cache_size=100`
`agent_max_message_metadata_cache_size`	Specifies the maximum number of messages' metadata to cache (publish/subscribe and invoke/implement). Possible values are any integer greater than 1. The default value is `200`.	`agent_max_message_metadata_cache_size=200`

Table 2–7 Adapter.ino Parameters

Parameter	Description	Example
`agent_max_dvm_table_cache_size`	Specifies the maximum number of DVM tables to cache. Possible values are any integer greater than 1. The default value is `200`.	`agent_max_dvm_table_cache_size=200`
`agent_max_lookup_table_cache_size`	Specifies the maximum number of lookup tables to cache. Possible values are any integer greater than 1. The default value is `200`.	`agent_max_lookup_table_cache_size=200`
`agent_max_queue_size`	Specifies the maximum size that internal OracleAS InterConnect message queues can grow. Possible values are any integer greater than 1. The default value is `1000`.	`agent_max_queue_size=1000`
`agent_persistence_queue_size`	Specifies the maximum size that internal OracleAS InterConnect persistence queues can grow. Possible values are any integer greater than 1. The default value is `1000`.	`agent_persistence_queue_size=1000`
`agent_persistence_cleanup_interval`	Specifies how often the persistence cleaner thread should run. Possible values are any integer greater than 30000 milliseconds. The default value is `60000`.	`agent_persistence_cleanup_interval=60000`
`agent_persistence_retry_interval`	Specifies how often the persistence thread should retry when it fails to push a Oracle9*i*AS InterConnect message. Possible values are any integer greater than 5000 milliseconds. The default value is `60000`.	`agent_persistence_retry_interval=60000`
`agent_pipeline_to_hub`	Specifies how to turn on or off the pipeline for messages from the Bridge towards the hub. If you set the pipeline to `false`, the file persistence is not used in that direction.	`agent_pipeline_to_hub=false`
`agent_pipeline_from_hub`	Specifies how to turn on or off the pipeline for messages from the hub towards the Bridge. If you set the pipeline to `false`, the file persistence is not used in that direction.	`agent_pipeline_from_hub=false`
`service_path`	Windows only. The value that the environment variable PATH should be set to. Path is set to the specified value before forking the Java VM. Typically, all directories containing all necessary DLLs should be listed here. Possible values are the valid path environment variable setting. There is no default value.	`service_path=%JREHOME%\bin;D:\oracle\ora904\bin`

Table 2–7 Adapter.ino Parameters

Parameter	Description	Example

`service_classpath`	The classpath used by the adapter Java VM. If a custom adapter is developed and as a result, the adapter is to be used to pick up any additional jars, add the jars to the existing set of jars being picked up. Possible values are the valid classpath. There is no default value.	`service_classpath=D:\oracle\ora904\oai\904\lib\oai.jar;`
		`%JREHOME%\lib\i18n.jar`
		`D:\oracle\ora904\jdbc\classes12.zip`

`service_class`	The entry class for the Windows service. The possible value is `oracle/oai/agent/service/AgentService`. There is no default value.	`service_class=oracle/oai/agent/service/AgentService`
`service_max_java_stack_size`	Windows only. The maximum size to which the Java VM's stack can grow. Possible values are the valid Java VM maximum native stack size. The default value is the default for the Java VM.	`service_max_java_stack_size=409600`
`service_max_native_stack_size`	Windows only. The maximum size to which the Java VM's native stack can grow. Possible values are the valid Java VM maximum native stack size. The default value is the default for the Java VM.	`service_max_native_size=131072`
`service_min_heap_size`	Windows only. Specifies the minimum heap size for the adapter Java VM. Possible values are the valid Java VM heap sizes. The default value is the default Java VM heap size.	`service_min_heap_size=536870912`
`service_max_heap_size`	Windows only. Specifies the maximum heap size for the adapter Java VM. Possible values are any valid Java VM heap sizes. The default value is `536870912`.	`service_max_heap_size=536870912`
`service_num_vm_args`	Windows only. The number of `service_vm_arg<number>` parameters specified. Possible values are the number of `service_vm_arg<number>` parameters. There is no default value.	`service_num_vm_args=1`
`service_vm_arg<number>`	Windows only. Specifies any additional arguments to the Java VM. For example, to get line numbers in any of the stack traces, set `service_vm_arg1=java.compiler=NONE`. If there is a list of arguments to specify, use multiple parameters as shown in the example by incrementing the last digit starting with 1. Be sure to set the `service_num_vm_args` correctly. Possible values are any valid Java VM arguments. There is no default value.	`service_vm_arg1=java.compiler=NONE` `service_vm_arg2=oai.adapter=.aq`

Table 2–7 Adapter.ino Parameters

Parameter	Description	Example
`service_jdk_version`	Windows only. The JDK version the adapter Java VM should use. The default value is `1.4.1`.	`service_jdk_version=1.4.1`
`service_jdk_dll`	Windows only. The dll the adapter Java VM should use. The default value is `jvm.dll`.	`service_jdk_dll=jvm.dll`

`nls_date_format`	Format for date fields expressed as string. The following pattern letters are defined. All other characters from A to Z and from a to z are reserved.
	Letter – Date or Time – Component Examples
	`G` – Era designator – `AD`
	`y` – Year – `1996; 96`
	`M` – Month in year – `July; Jul; 07`
	`w` – Week in year – `27`
	`W` – Week in month – `2`
	`D` – Day in year – `189`
	`d` – Day in month – `10`
	`F` – Day of week in month – `Number 2`
	`E` – Day in week – `Tuesday; Tue`
	`a` – A.M./P.M. marker – `P.M.`
	`H` – Hour in day (0-23) – `0`
	`k` – Hour in day (1-24) – `24`
	`K` – Hour in A.M/P.M. (0-11) – `0`
	`h` – Hour in A.M./P.M. (1-12) – `12`
	`m` – Minute in hour – `30`
	`s` – Second in minute – `55`
	`S` – Millisecond – `978`
	`z` – Time zone – `Pacific`
	The default date format is `EEE MMM dd HH:mm:ss zzz yyyy`.
	Note: This parameter specifies date format. It is applicable for the date format only.
		Date format pattern `dd/MMM/yyyy` can represent 01/01/2003.
		`nls_date_format=dd-MMM-yy`
		Multiple date format can be specified as `num_nls_formats=2`
		`nls_date_format1=dd-MMM-yy`
		`nls_date_format2=dd/MMM/yy`

Table 2–7 Adapter.ino Parameters

Parameter	Description	Example
`nls_country`	This parameter is a valid ISO Country Code. These upper-case and two-letter codes are defined by ISO-3166. You can find a full list of these codes at a Web site, such as, `http://www.chemie.fu-berlin.de/diverse/doc/ISO_3166.html` The default Country code is `US`. **Note**: This parameter specifies date format. It is applicable for the date format only.	`US`
`nls_language`	This parameter is a valid ISO Language Code. These lower-case and two-letter codes are defined by ISO-639. You can find a full list of these codes at a Web site, such as, `http://www.ics.uci.edu/pub/ietf/http/related/iso639.txt` The default language code is `en`. **Note**: This parameter specifies date format. It is applicable for the date format only.	`nls_language=en`
`encoding`	Character encoding for published messages. The adapter uses this parameter to generate encoding information in encoding tag of transformed OracleAS InterConnect message. OracleAS InterConnect represents messages internally as an XML document. The default encoding of the XML document is `UTF-8`. However, this encoding can be configured using this parameter, which is typically used when the OracleAS InterConnect message consists of characters not supported by `UTF-8` and when the `XMLParser` is unable to handle them.	`encoding=JA16SJIS`
`corba_port_number`	The CORBA port number on which the adapter CORBA service listens. Generally, this port is allocated dynamically. However, it can be configured to enable access across firewall.	`corba_port_number=14000`

SAP Adapter Adapter-Specific Parameters

The following table lists the parameters specific to the SAP adapter.

Parameter	Description	Example
`bridge_class`	This indicates the entry class for the SAP adapter. Do not modify this value. A possible value is `com.actional.oai.TxAgent`. There is no default value.	`bridge_ class=com.actional.oa i.TxAgent`

Starting the SAP Adapter

On UNIX, start the SAP adapter using the `start` script in the following directory:

`$ORACLE_HOME/oai/9.0.4/adapters/`*`Application`*

Type **start**, then press **Enter**.

On Windows, start the adapter from the Services window available from the Start menu.

1. Access the Services window from the Start menu:

On...	Choose...
Windows NT	Start > Settings > Control Panel > Services
Windows 2000	Start > Settings > Control Panel > Administrative Tools > Services

   The Services window displays.

2. Select the *OracleHomeOracleASInterConnectAdapter-Application* service.
3. Start the service based on your operating system:

On...	Choose...
Windows NT	Choose Start.
Windows 2000	Right click the service and choose Start from the menu that displays.

The SAP adapter, in turn, automatically starts the publishing engine, a tool for notifying foreign applications of additions, deletions, or updates to the native application.

> **See Also:** *Oracle Application Server InterConnect Adapter Publishing Engine User's Guide*

On Windows only, If you are using the SAP adapter or your browser in iStudio, but you fail to initialize the SAP adapter, you may not have the keys for `JavaHome` and `RuntimeLib`, or these keys do not point to the correct JDK. In this case, the iStudio browser will not display or the data from the backend system cannot be imported.

To created these keys, use the Windows `regedit` tool. To access the regedit tool:

1. Click **Start** and select **Run**.
2. Enter **regedit** and click **OK**.

The following example displays the values for these keys when the SAP adapter is installed under the `C:\Oracle\Ora90 directory`:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\JavaSoft\Java Runtime Environment]

[HKEY_LOCAL_MACHINE\SOFTWARE\JavaSoft\Java Runtime Environment\1.3]
"CurrentVersion"="1.3"

[HKEY_LOCAL_MACHINE\SOFTWARE\JavaSoft\Java Runtime Environment\1.3]
"JavaHome"="c:\\oracle\\ora90\\jdk\\jre"
"MicroVersion"="1"
"RuntimeLib"="C:\\Oracle\\Ora90\\jdk\\jre\\bin\\hotspot"
```

Stopping the SAP Adapter

On UNIX, stop the SAP adapter using the `stop` script in the following directory:

`$ORACLE_HOME/oai/9.0.4/adapters/`*`Application`*

Type **stop**, then press **Enter**.

On Windows, stop the adapter from the Services window available from the Start menu.

1. Access the Services window from the Start menu:

On...	Choose...
Windows NT	Start > Settings > Control Panel > Services
Windows 2000	Start > Settings > Control Panel > Administrative Tools > Services

The Services window displays.

2. Select the *OracleHomeOracleASInterConnectAdapter-Application* service.
3. Stop the service based on your operating system:

On...	Choose...
Windows NT	Choose Stop.
Windows 2000	Right click the service and choose Stop from the menu that displays.

You may verify the stop status by viewing the `oailog.txt` files in the appropriate time stamped subdirectory of the `log` directory within the adapter directory.

3

Supported SAP Adapter Interfaces

This chapter provides an overview about SAP Adapter-specific information to assist you in working with the SAP Adapter adapter. The following topics are discussed:

- Exception Fields
- Inbound to SAP
- Outbound From SAP

Exception Fields

An exception field is added by the SAP adapter when a function is imported into iStudio.

If an error happens during a call, the exception field generally contains a detailed description of the error that occurred. You can then propagate this error string to the calling application.

For example, setup an SAP R/3 system on one side, an OracleAS InterConnect hub in the middle, and a Web front end the other side. If the Web front-end tries to add a record to the SAP R/3 side and a record with the same primary key already exists in SAP R/3, a non-retryable error occurs. The exception field contains the exception data. This data may be propagated back to the Web front-end. The following is an example of an exception message:

```
exception: E-OAI0003: Exception occurred during call to
AddWidget@OAI://Messages/WidgetStore
User defined exception
Exception occurred:
    Source: WidgetStore::AddWidget
Cause: OAI://Messages/exception=MsgAgentException (Unique ID none)
    Exception occurred:
    Source: WidgetStore::AddWidget
    Cause: OAI://Messages/exception=MsgAgentException (Unique ID none)
Exception data:
    struct MsgAgentExceptionData =
    String Source = OAIsgProcessMessage(WidgetStore::AddWidget)
    String ErrorText = Row exists in ADD or DATAENTRY mode (81,10)
    String Explanation = The specified search keys resulted in an existing
        level 0 row found when in ADD or DATAENTRY mode.
```

Inbound to SAP

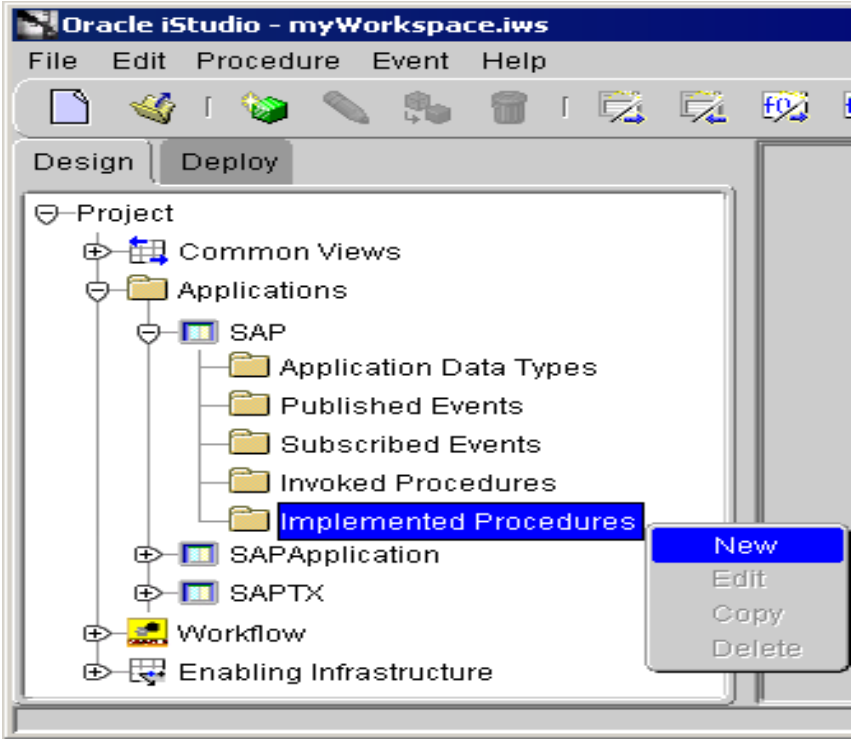
Sending messages inbound means the SAP adapter is the client and SAP is the server. To send messages to the SAP adapter, ensure that the host definition and login information is set for connecting to the SAP system using the Configuration Editor.

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

Creating an Application Link Enabling Implemented Procedure

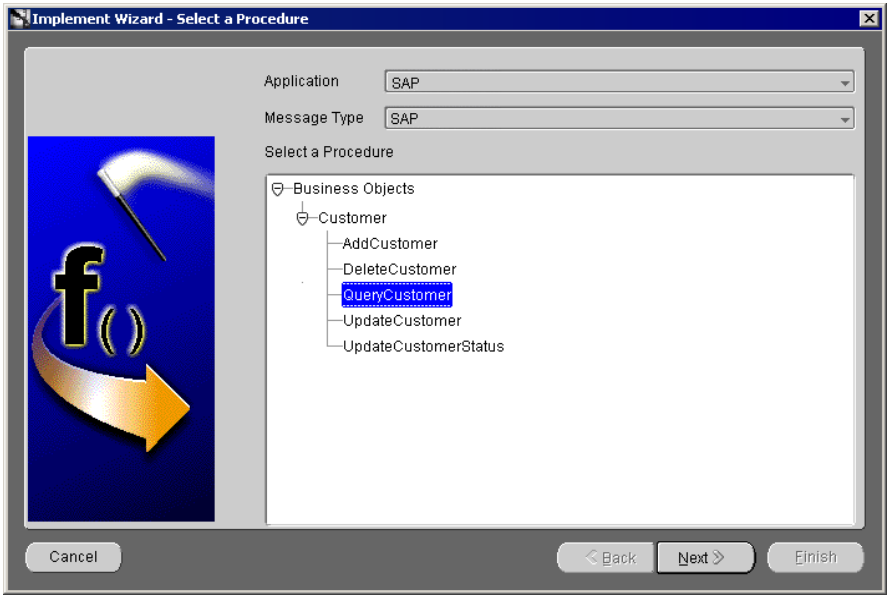
1. Start iStudio and open your project.
2. Expand the Applications folder.
3. Expand your Application.
4. Right-click **Implemented Procedures** and select **New**.

Figure 3–1 Creating an Implemented Procedure



The Implement Wizard—Select a Procedure dialog displays.

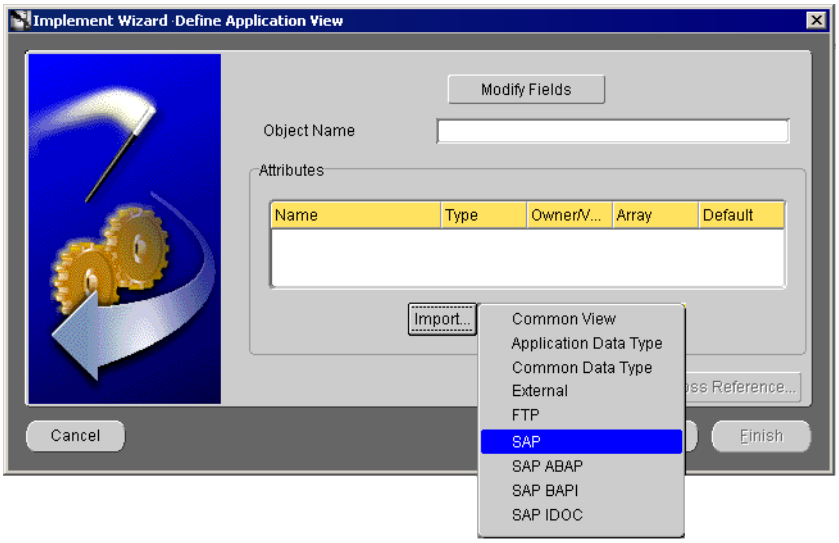
Figure 3–2 Implement Wizard —Selecting a Procedure



5. Select the Application and Message Type from the dropdown menus.
6. Select a procedure and click **Next**.

The Implement Wizard—Define Application View dialog displays.

Figure 3–3 Implement Wizard - Define Application View - Importing SAP



7. Click **Import** and select **SAP** from the dropdown menu.

   The SAP Login dialog displays.

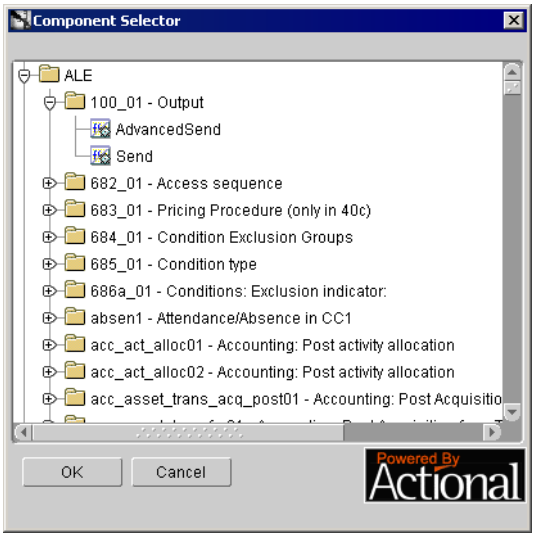
   If this is the initial login for this machine, enter the correct information.

   See Also: "Importing Attributes from SAP" on page 3-8

   If this machine has been logged in to SAP before, enter the password on the SAP Login dialog and click **OK**.

Once logged in to SAP, the Component Selector dialog displays.

Figure 3–4 Component Selector for Application Link Enabling

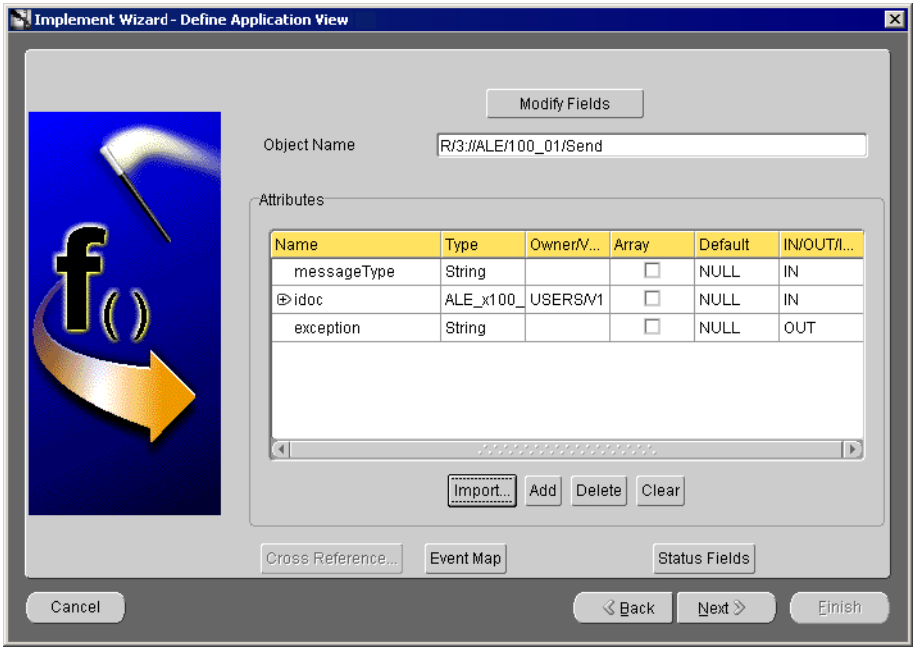


8. Expand the ALE tree until the correct component displays for selection.
9. Select **AdvancedSend** or **Send** and click **OK**.

   The `Send` method populates the control record of the intermediate document from the parameters set up in the SAP R/3 configuration editor. The `AdvancedSend` method allows more flexibility. When you use this method, you must pass the control data to the method.

The Define Application View dialog displays with the selected component and its attributes.

Figure 3–5 Populated Implement Wizard - Define Application View dialog



10. Click **Next**.

    The Define Mappings dialog displays.

11. Define the mappings and click **Finish**.

    The new populated procedure displays in the right panel.

Importing Attributes from SAP

When you use iStudio to import attributes from SAP, you must first log in to SAP. When logging into SAP from iStudio, the login fields automatically populate, leaving the Password field the only field that requires input.

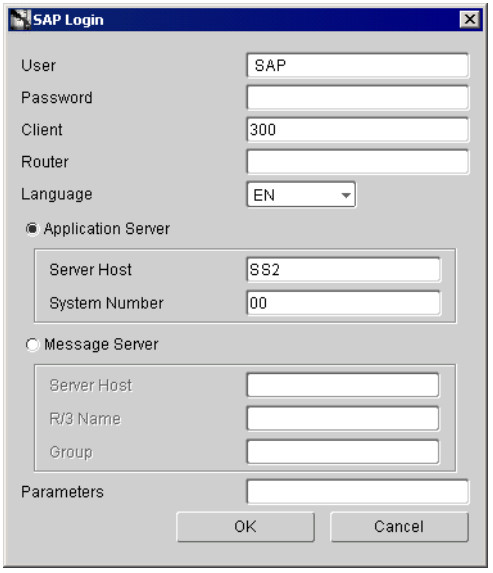
To import attributes from SAP:

1. Click **Import** and select **SAP** on the Define Applications View dialog.

   The SAP Login dialog displays.

   The first time you log in from a new workstation, you are required to enter information in every field that is required for your setup. Every subsequent login from that workstation only requires a password to log in. For every iStudio session, only one login is required.

Figure 3–6 Initial SAP Login dialog



2. Enter information in the following fields:
   - User—The user ID for the SAP R/3 system.
   - Password—The user password for the SAP R/3 system.
   - Client—The client number for the SAP R/3 system.

- Router—A destination router used to connect to the Application Server or Message Server. For example: `/H/UNICENTER/H/204.79.199.5/H`.
- Language—The language required by SAP R/3 system. By default the Language parameter retrieves the language information from the users operating system.
- Application Server—Select if using the Application Server and enter information the following fields:
  * Server Host—The identification of your SAP R/3 system. This value defines a connection to an Application Server representing a single SAP R/3 system.
  * System Number—The SAP System number identifying the system on the host. This number specifies the TCP/IP service of the Remote Function Call Gateway containing the registered Agent.

    System Number further identifies the Host to a specified Service level. The service is the TCP/IP service name (a port number through `\winnt\system32\drivers\etc\services`). For example, using `ss1:00` as the connecting host in the browser, the `00` is what SAP calls the system number. When specifying a service name, `sapgw00`, the `00` also represents the system number. That is, if an SAP R/3 system uses system number `23`, then `ss1:23` is in the login dialog and uses `sapgw23` as the service number for the SAP Agent. `sapgw` is a name assigned on installation to identify the gateway machine.
- Message Server—Select if using the Message Server and enter information in the following fields:
  * Server Host—The Server type which identifies the Message and provides the Server host name. This value defines a connection to a message server acting as a load-balancing server redirecting the login to an application server. The message server option is only valid for inbound calls. For example, `hs0016.WDF.SAP-AG-DE`.
  * SAP R/3 Name—The System ID identifying the SAP System. For example, `D15`.
  * Message Server Group—If your message servers belong to a group, enter the message server group. For example, `PUBLIC`.

- Parameters—The host identification parameter.

  A route string that contains a substring for each SAP router and for the target server. The route string syntax is: `/H/host/S/service/W/pass` that is, it comprises any number of substrings of the form `/H/host/S/service/W/pass`. For example, a connection from `hostA` to `hostB`, `port 3333` via the saprouter host `hostR` with SAProuter password summer has the route string `/H/hostR/S/3299/W/summer/H/hostB/S/3333`.

Table 3–1 lists the possible host identification keys and definitions extracted from Remote Function Call 4.0 documentation.

See Also: SAP Remote Function Call documentation for more information about establishing Remote Function Call connections

Table 3–1 Identification of Keys

Key	Definition
ABAP_DEBUG	Specifies whether to run the function modules within the ABAP debugger. Can be either zero (0) for no debugger, or 1 for running within the debugger. Default is zero (0). In the context of the product, ABAP_DEBUG may be useful for debug or diagnostic purposes. However, it is of limited use in a production environment since the ABAP debugger would be invoked on the server's machine, not the client machine.
ASHOST	Host name of a specific application server, if not using session management.
CLIENT	Login client. Although this key is automatically appended by the product, it can be specified in the host identification, thus forcing a specified use instead of the one provided by the user or client. This is most useful at run time if it is desired to force all client applications to login with a specific client.
DEST	Destination in saprfc.ini.
GROUP	Name of the group of application servers, if using session management.
LANG	Login language (1-character SAP language or 2-character ISO 639 language).
MSHOST	Host name of the Message Server, if using Remote Function Call session management.
PASSWD	Login password. Similar comment as Client.

Table 3–1 Identification of Keys(Cont.)

Key	Definition
R3NAME	Name of the SAP R/3 system, if using Remote Function Call session management.
RFC_TRACE	Specifies whether Remote Function Call tracing should be enabled. Can be either zero (0) for disabling tracing or 1 for enabling it. Default is zero (0). When enabled, the Remote Function Call library writes trace entries in a trace file—rfc id.trc—in the current directory, or in the directory identified by the RFC_TRACE_DIR environment variable. id represents the Remote Function Call connection, meaning that there is one Remote Function Call trace file created per connection. Note that errors are always written to trace files. The RFC_TRACE keyword only affects the logging of other general trace messages.
SNC_LIB	Path and name of the Secure Network Communication library.
SNC_MODE	Specifies whether to work with Secure Network Communication. Can be either zero (0) for not working with Secure Network Communication, or 1 for working with Secure Network Communication. Default is zero (0).
SNC_MYNAME	Own Secure Network Communication name if the default one is not appropriate.
SNC_PARTNERNAME	Secure Network Communication name of the Secure Network Communication partner (Remote Function Call server) or Secure Network Communication name of the message server (session management).
SNC_QOP	Secure Network Communication quality of service. Default: 8 (RFC_SNC_QOP_DEFAULT).
SYSNR	SAP R/3 system number, if not using session management.
USE_SAPGUI	Specifies whether a SAPGUI is allowed to be invoked in the context of the Remote Function Call connection. Can be either zero (0), 1, or 2. Zero (0), the Default setting, specifies that no SAPGUI should be invoked. 1 specifies that a SAPGUI should be invoked. 2 is similar to 1, except that the SAPGUI is hidden between two Remote Function Call functions.
USER	Login user. Similar comment as Client.

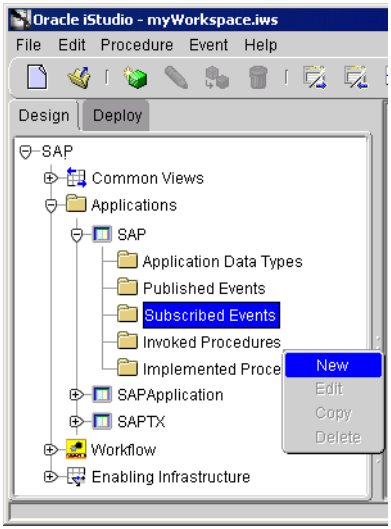
3. Click **OK** to accept your selections and continue to the Component Selector.

Creating an Application Link Enabling Subscribed Event

To create an Application Link Enabling subscribed event using iStudio:

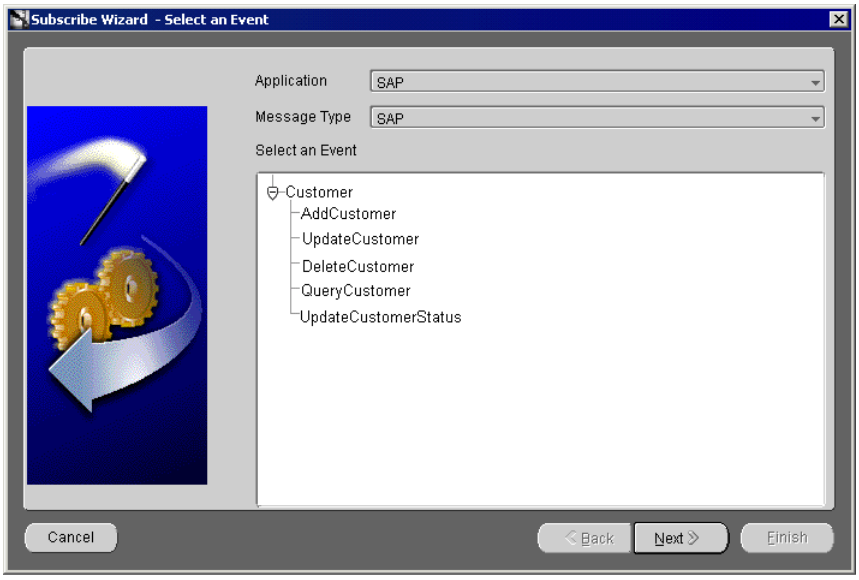
1. Start iStudio
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Subscribed Events** and select **New**.

Figure 3–7 Creating a Subscribed Event



The Subscribe Wizard—Select an Event dialog displays.

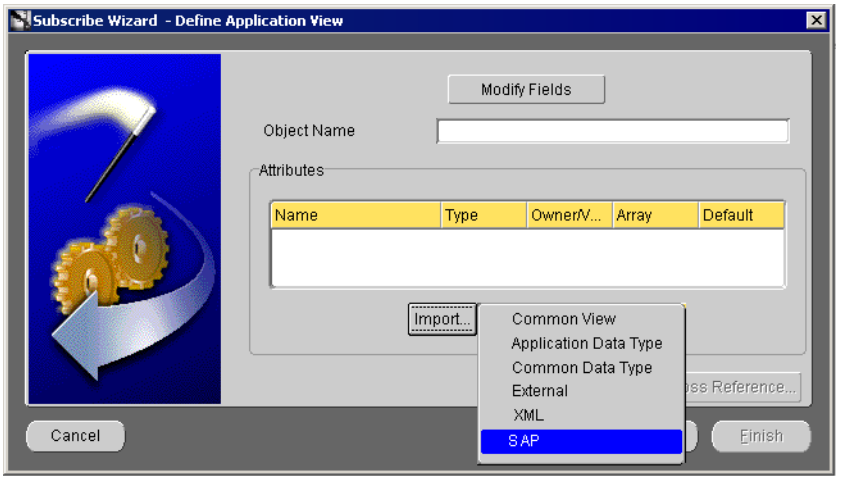
Figure 3–8 Subscribe Wizard - Selecting an Event



6. Select the Application and Message Type from the dropdown menus.
7. Select an event and click **Next**.

The Define Application View dialog displays.

Figure 3–9 Subscribe Wizard - Define Application View



8. Click **Import** and select **SAP**.

   The SAP Login dialog displays.

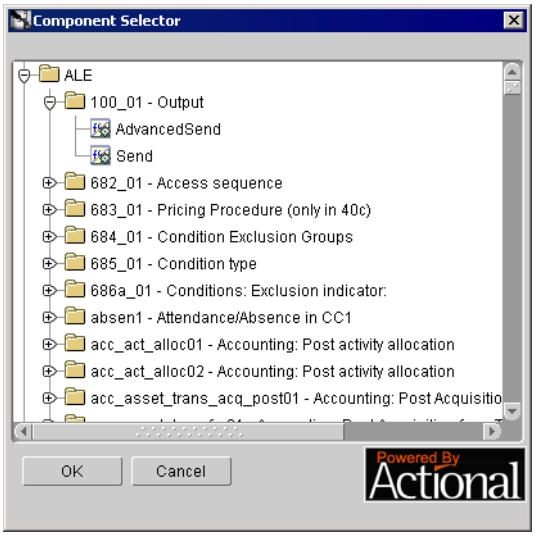
   If this is the initial login for this machine, enter the correct information.

   See Also: "Importing Attributes from SAP" on page 3-8

   If this machine has been logged in to SAP before, enter the password on the SAP Login dialog and click **OK**.

The Component Selector dialog displays.

Figure 3–10 Component Selector

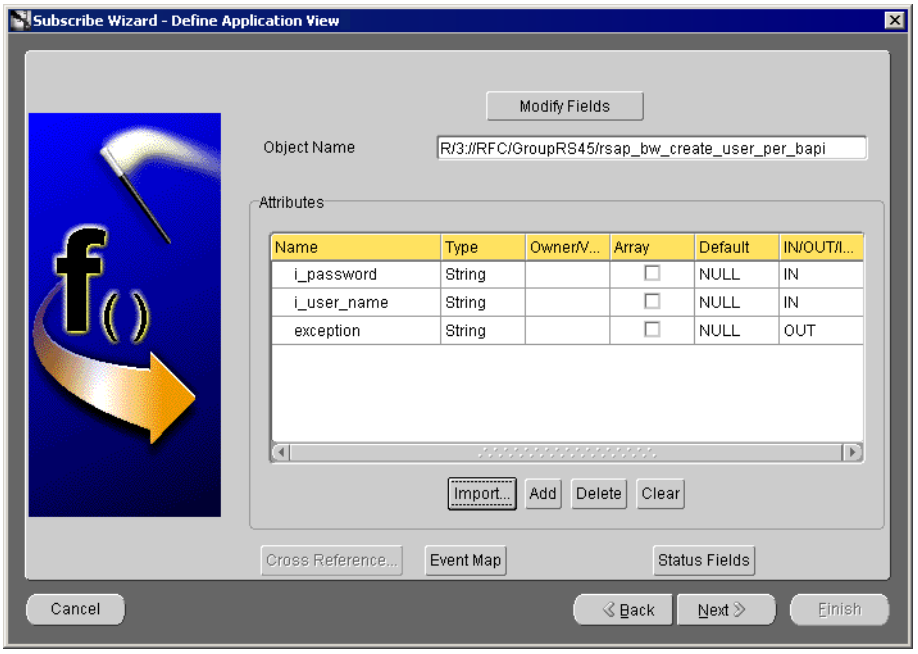


9. Expand the ALE tree until the correct component displays for selection.
10. Select **AdvancedSend** or **Send** and click **OK**.

    The `Send` method populates the control record of the intermediate document from the parameters set up in the SAP R/3 configuration editor. The `AdvancedSend` method allows more flexibility. When you use this method, you must pass the control data to the method.

The populated Define Applications View dialog displays.

Figure 3–11 Subscribed Wizard - Define Application View



11. Click **Next**.

The Define Mappings dialog displays.

12. Click **New** to define the mappings, then click **Finish**.

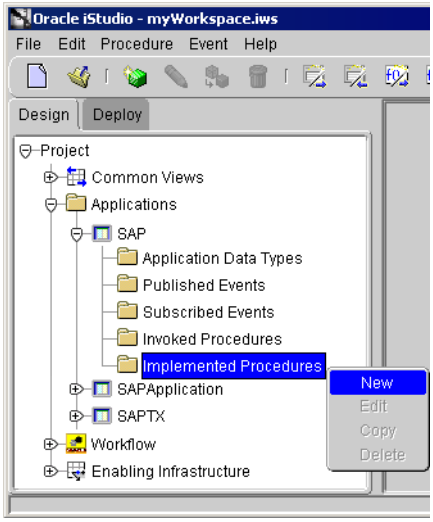
The created event displays in the right panel of iStudio.

Creating a Remote Function Call Implemented Procedure

To create a Remote Function Call implemented procedure:

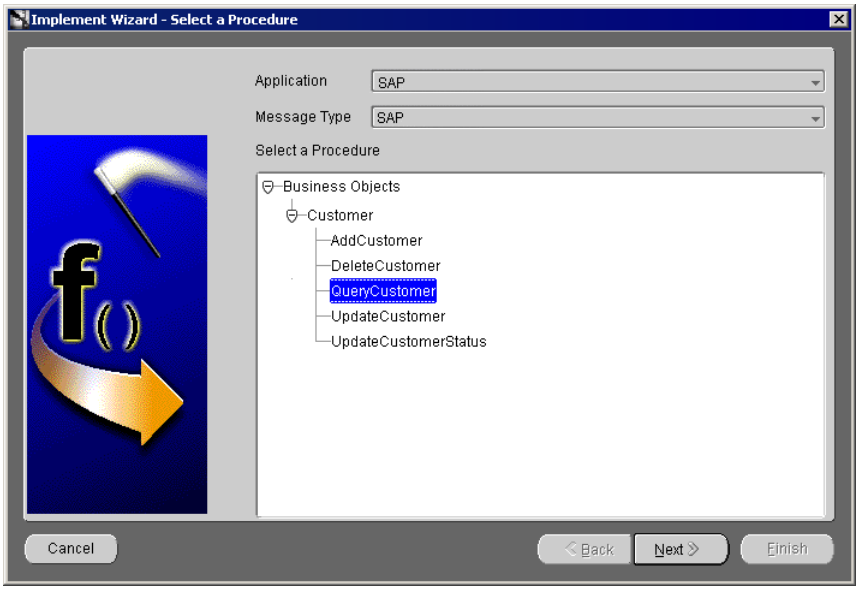
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Implemented Procedures** and select **New**.

Figure 3–12 Creating an Implemented Procedure



The Implement Wizard—Select a Procedure dialog displays.

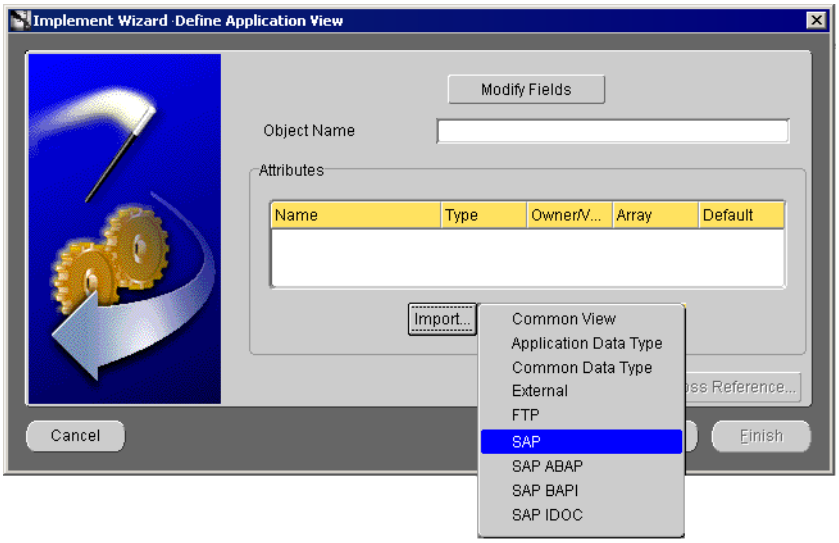
Figure 3–13 Implement Wizard - Selecting a Procedure



6. Select the Application and Message Type from the dropdown menus.
7. Select a procedure and click **Next**.

The Define Application View dialog displays.

Figure 3–14 Implement Wizard - Define Application View



8. Click **Import** and select **SAP**.

   The SAP Login dialog displays.

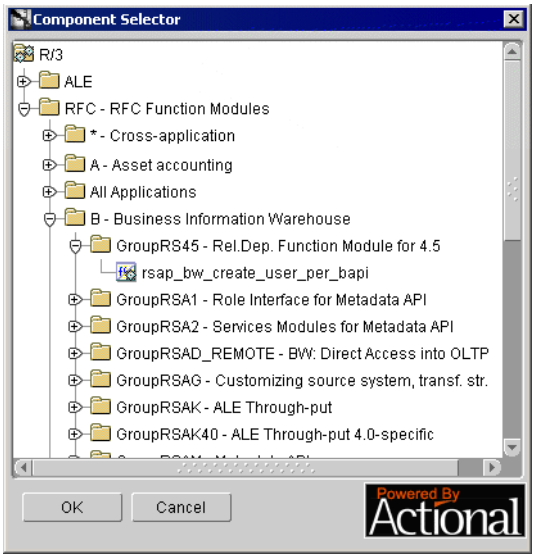
   If this is the initial login for this machine, enter the correct information.

   See Also: "Importing Attributes from SAP" on page 3-8

   If this machine has been logged in to SAP before, enter the password on the SAP Login dialog and click **OK**.

The Component Selector dialog displays.

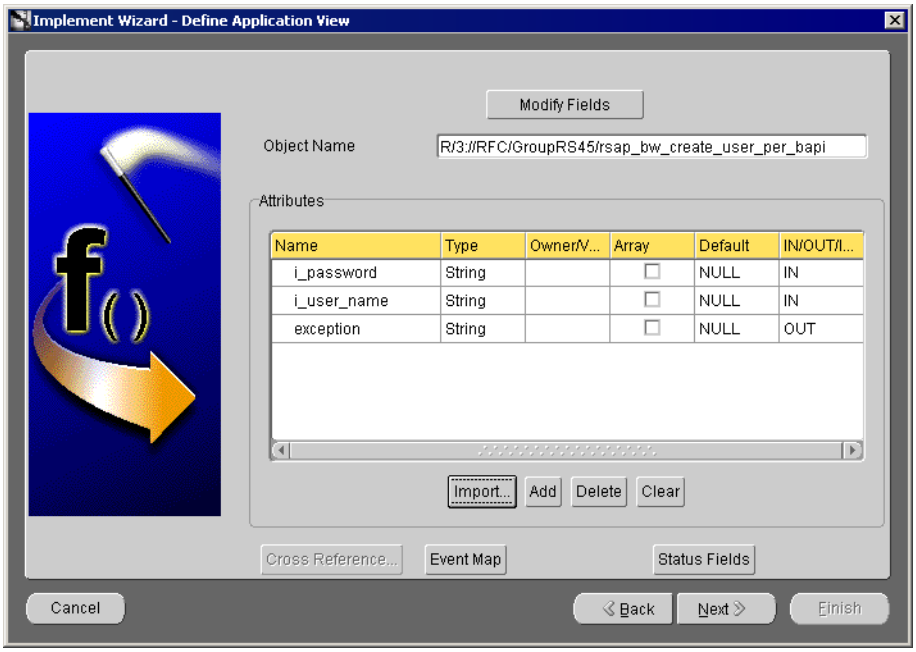
Figure 3–15 Component Selector with RFC - RFC Function Module sub-folders



9. Expand the RFC - RFC Function Modules tree until the correct component displays for selection.
10. Select a component and click **OK**.

The populated Define Application View dialog displays.

Figure 3–16 Implement Wizard - Define Application View



11. Click **Next**.

    The Define Mappings dialog displays.

12. Click **New** to define mappings and click **Finish**.

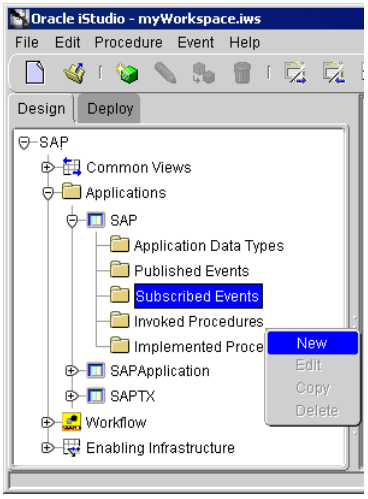
    The new populated procedure display in the right panel of iStudio.

Creating a Remote Function Call Subscribed Event

To create a Remote Function Call subscribed event:

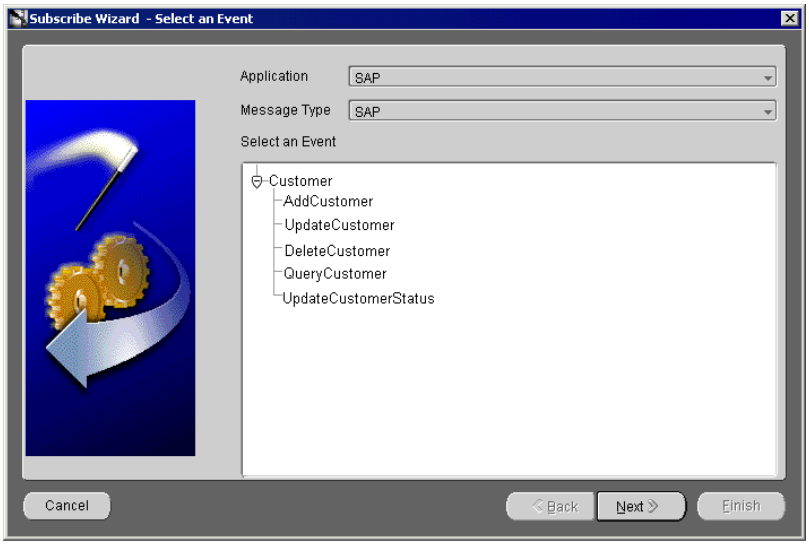
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Subscribed Events** and select **New**.

Figure 3–17 Creating a Subscribed Event



The Subscribe Wizard—Select an Event dialog displays.

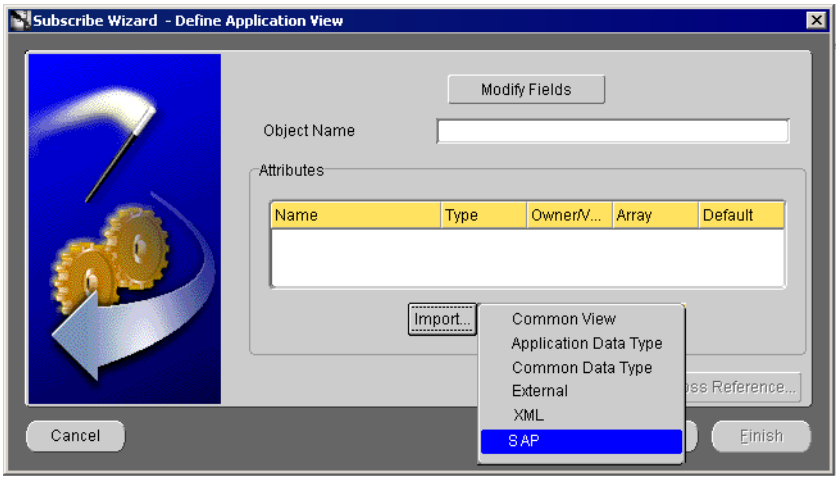
Figure 3–18 Subscribe Wizard - Selecting an Event



6. Select the Application and Message Type from the dropdown menus.
7. Select an event and click **Next**.

The Define Application View dialog displays.

Figure 3–19 Subscribe Wizard - Define Application View



8. Click **Import** and select **SAP**.

   The SAP Login dialog displays.

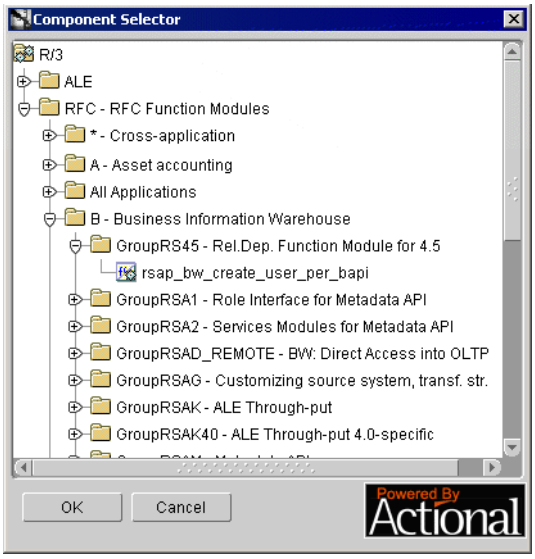
   If this is the initial login for this machine, enter the correct information.

   See Also: "Importing Attributes from SAP" on page 3-8

   If this machine has been logged in to SAP before, enter the password on the SAP Login dialog and click **OK**.

The Component Selector dialog displays.

Figure 3–20 Component Selector - RFC - RFC Function Modules

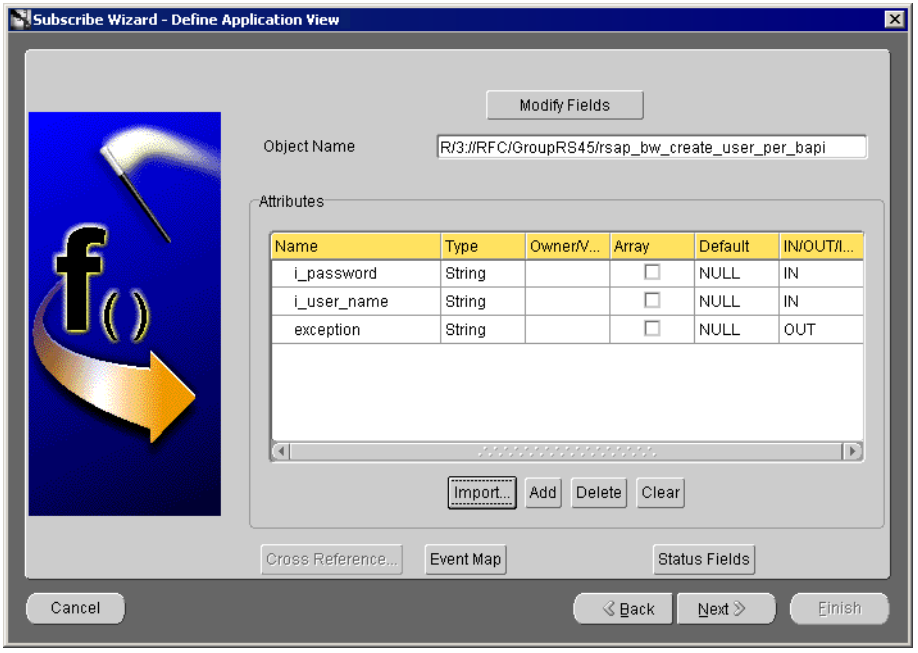


9. Expand the RFC - RFC Function Modules tree until the correct component displays for selection.

10. Select a component and click **OK**.

    The populated Define Application View dialog displays.

Figure 3–21 Subscribe Wizard - Populated Define Application View



11. Click **Next**.

    The Define Mappings dialog displays.

12. Click **New** to define mappings and click **Finish**.

    The new populated event displays in the right panel of iStudio.

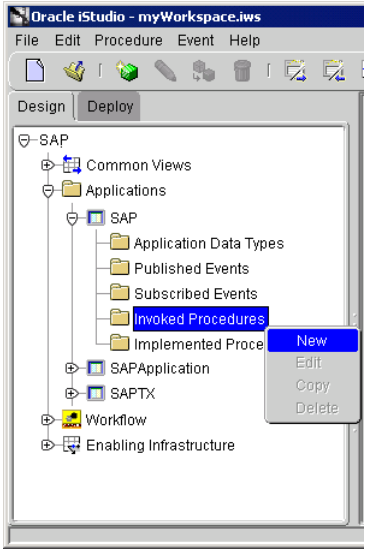
Outbound From SAP

Outbound from R/3 is used when the SAP R/3 system is sending messages to your application. The Remote Function Call Program ID must be set. The Remote Function Call Program ID is used to register with the SAP R/3 system.

Creating an Outbound Application Link Enabling Invoked Procedure

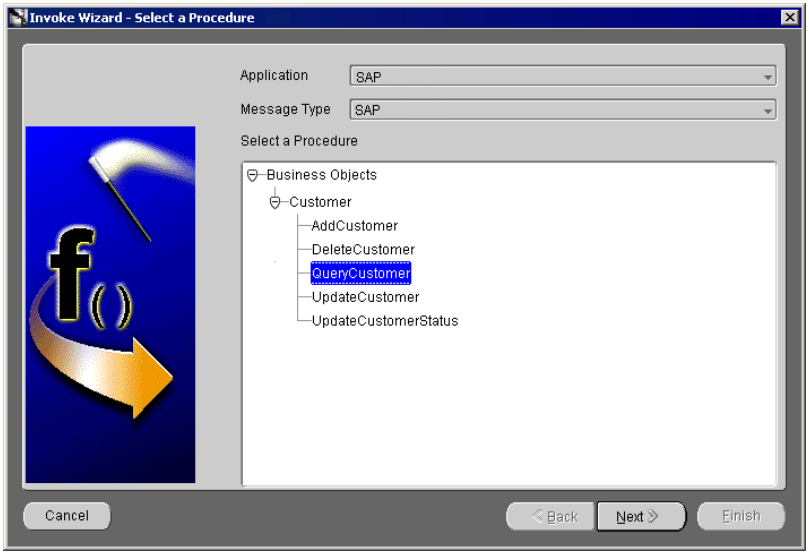
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Invoked Procedures** and select **New**.

Figure 3–22 Creating an Implemented Procedure



The Invoke Wizard—Select a Procedure dialog displays.

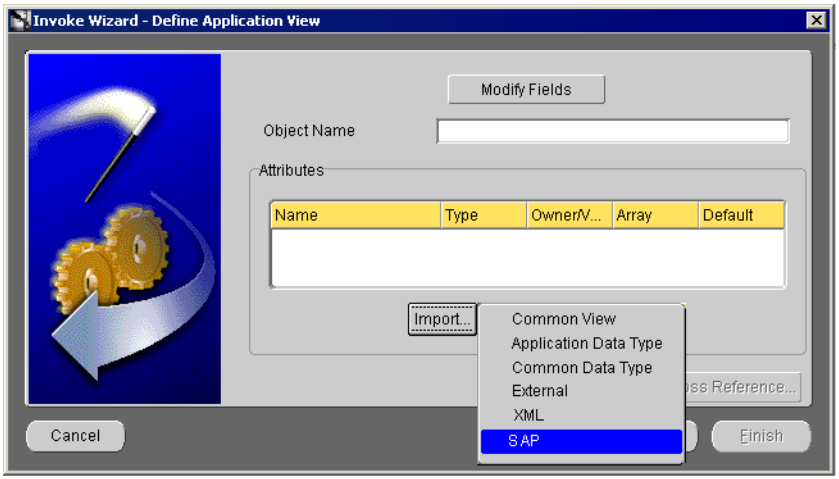
Figure 3–23 Invoke Wizard - Selecting a Procedure



6. Select the Application and Message type from the dropdown menus.
7. Select a procedure, and click **Next**.

The Define Application View dialog displays.

Figure 3–24 Invoked Wizard - Define Application View



8. Click **Import** and select **SAP**.

   The SAP Login dialog displays.

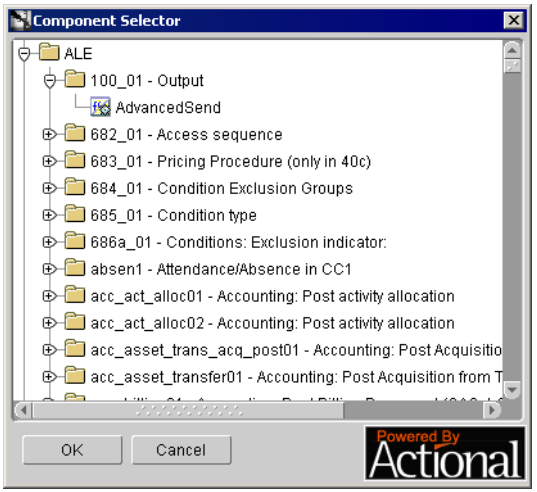
   If this is the initial login for this machine, enter the correct information.

   See Also: "Importing Attributes from SAP" on page 3-8

   If this machine has been logged in to SAP before, enter the password on the SAP Login dialog and click **OK**.

The Component Selector displays.

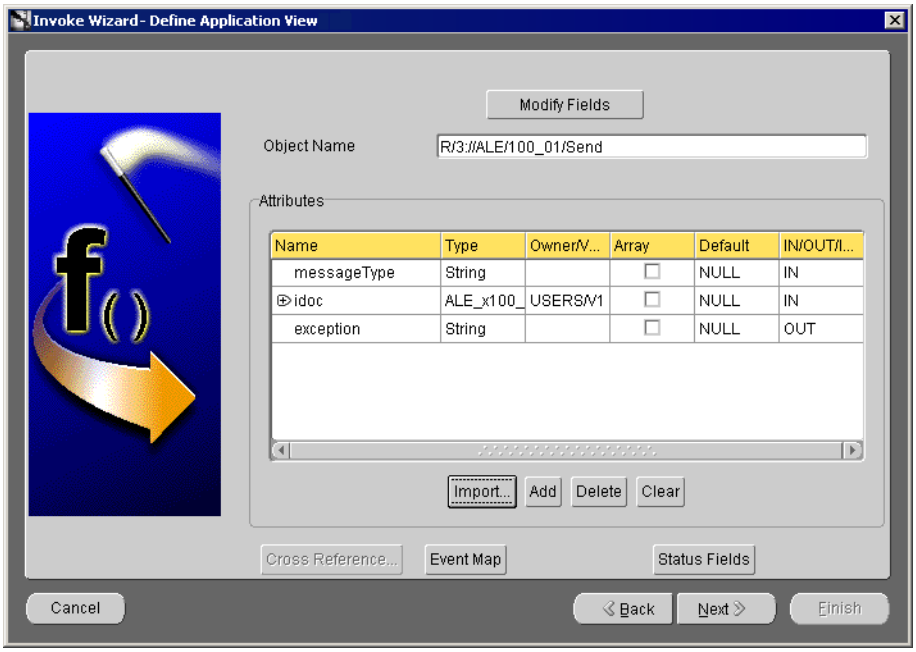
Figure 3–25 Component Selector - Application Link Enabling



9. Expand the ALE tree until the correct component displays for selection.
10. Select a component and click **OK**.

The populated Define Application View dialog displays.

Figure 3–26 Invoke Wizard - Populated Define Applications View



11. Click **Next**.

    The Define Mappings dialog displays.

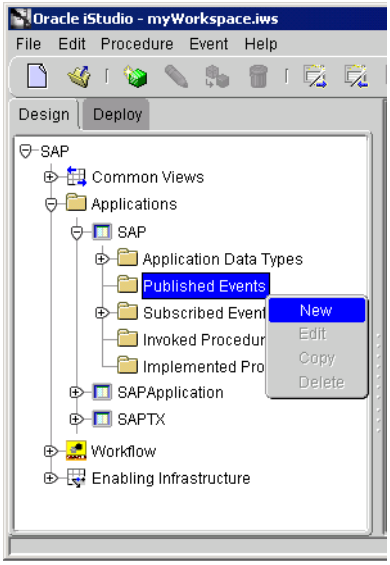
12. Click **New** to define mappings and click **Finish**.

    The new populated event displays in the right panel of iStudio.

Creating an Application Link Enabling Published Event

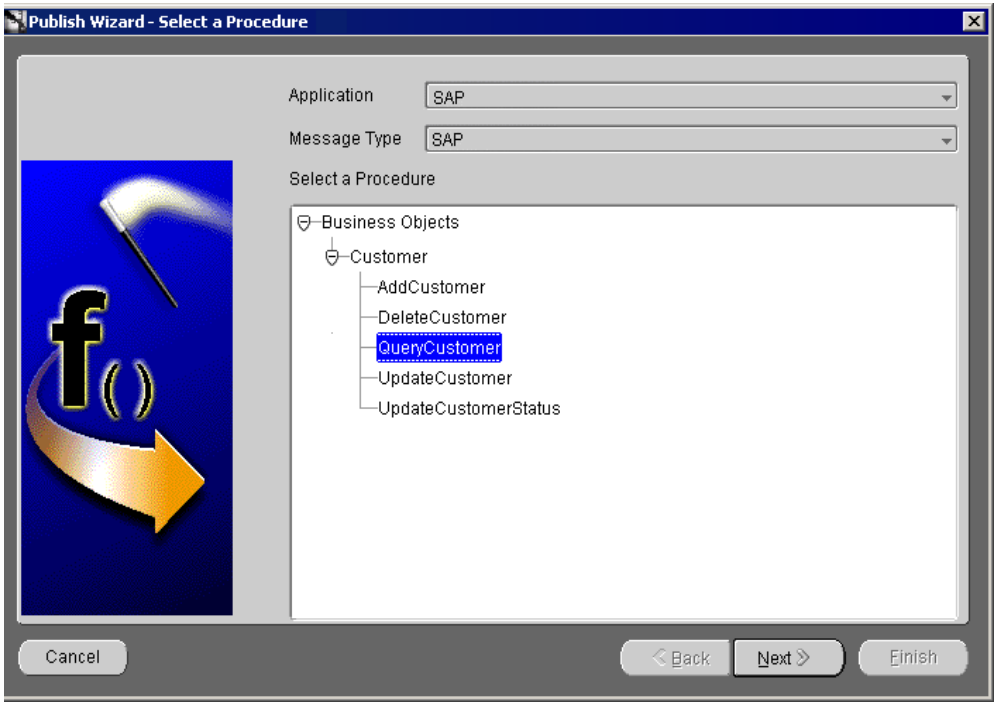
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Published Events** and select **New**.

Figure 3–27 Creating a Published Event



The Publish Wizard—Select an Procedure dialog displays.

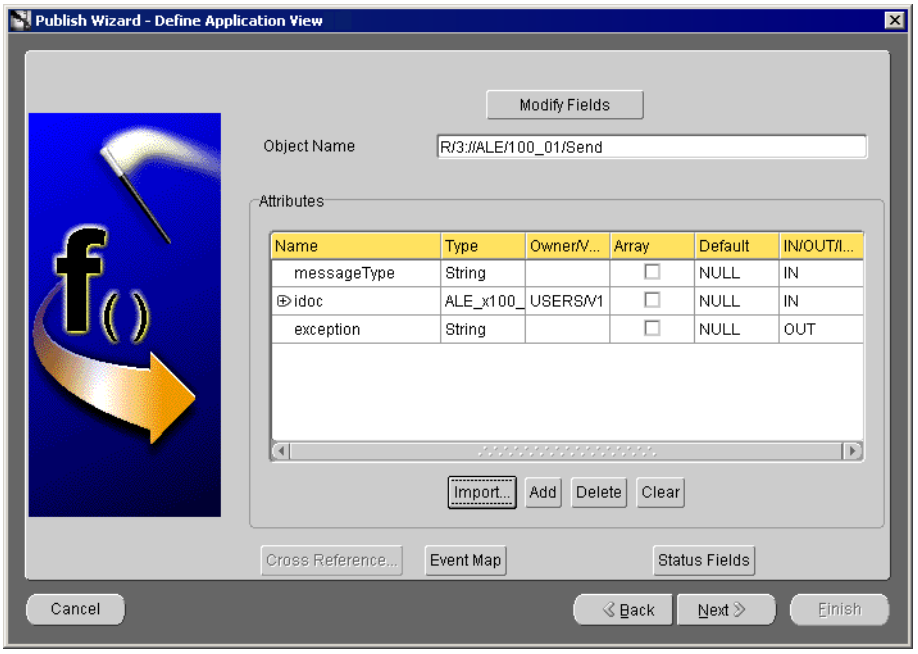
Figure 3–28 Publish Wizard - Selecting a Procedure



6. Select the Application and Message Type from the dropdown menus.
7. Select an event and click **Next**.

The Define Application View dialog displays.

Figure 3–29 Publish Wizard - Define Application View



8. Click **Import** and select **SAP**.

   The SAP Login dialog displays.

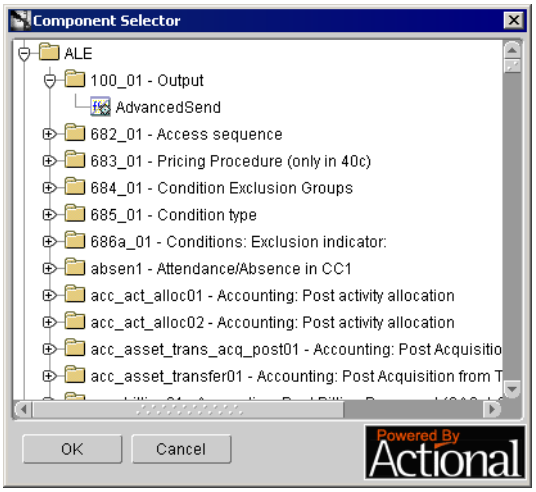
   If this is the initial login for this machine, enter the correct information.

   See Also: "Importing Attributes from SAP" on page 3-8

   If this machine has been logged in to SAP before, enter the password on the SAP Login dialog and click **OK**.

The Component Selector displays.

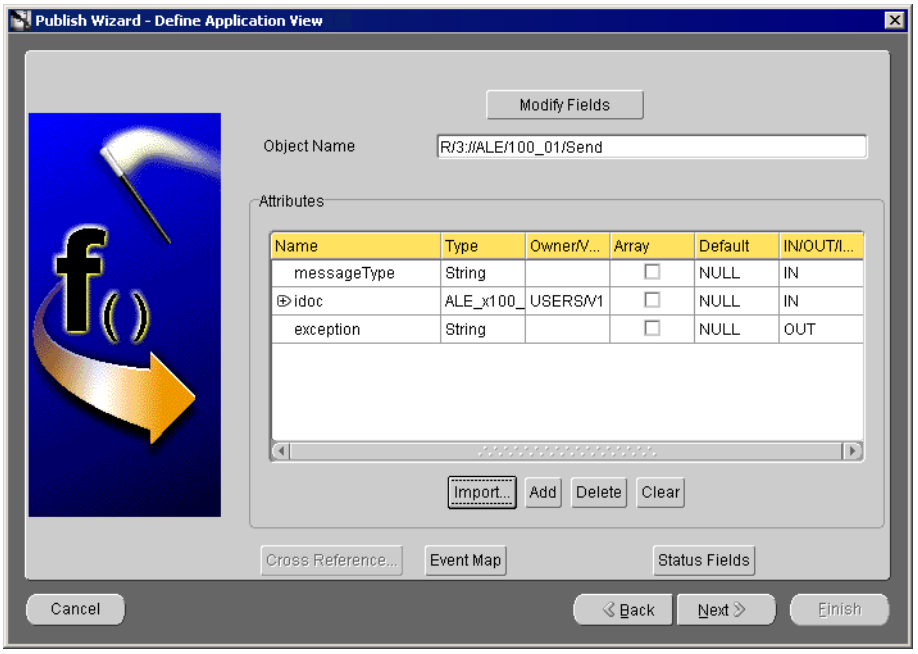
Figure 3–30 Component Selector - Application Link Enabling AdvancedSend



9. Expand the ALE tree until the correct component displays for selection.
10. Select a component and click **OK**.

The populated Define Application View dialog displays.

Figure 3–31 Publish Wizard - Populated Define Application View



11. Click **Next**.

The Define Mappings dialog displays.

12. Click **New** to define mappings and click **Finish**.

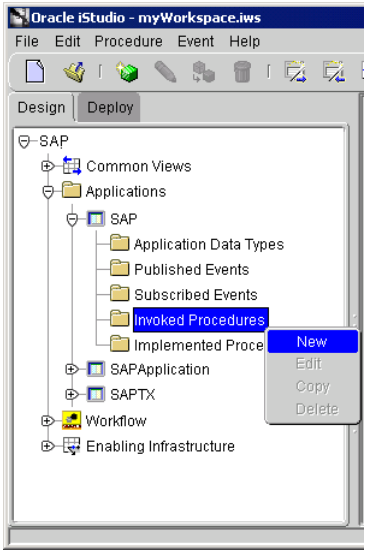
The new populated event will display in the right panel in iStudio.

Creating a Remote Function Call Invoked Procedure

To create a Remote Function Call invoked procedure in iStudio:

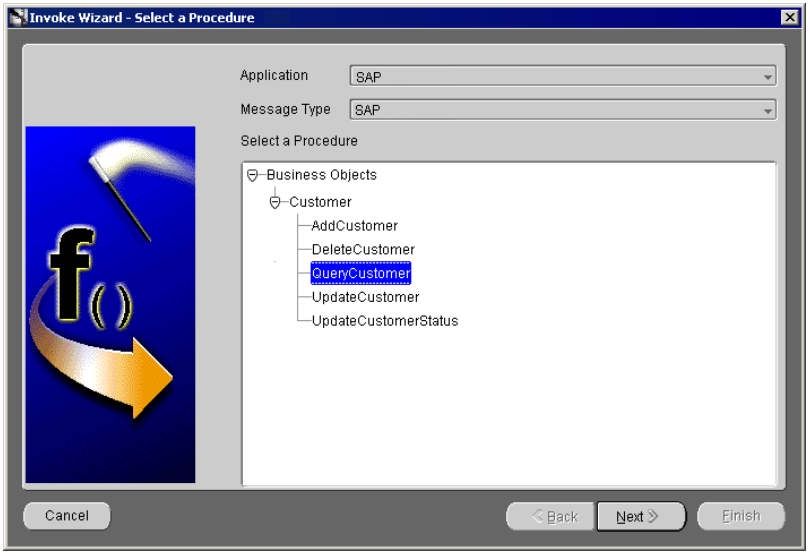
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Invoked Procedures** and select **New**.

Figure 3–32 Creating an Invoked Procedure



The Invoke Wizard—Select a Procedure dialog displays.

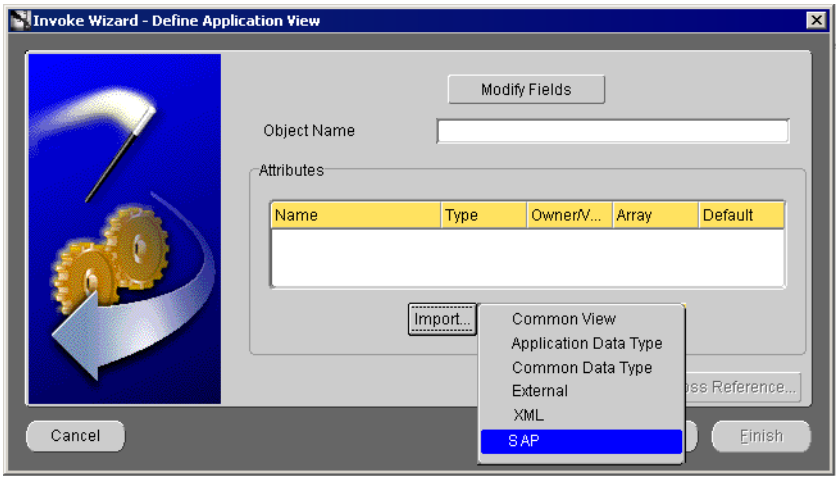
Figure 3–33 Invoke Wizard - Selecting a Procedure



6. Select the Application and Message Type from the dropdown menus.
7. Select a procedure and click **Next**.

The Define Application View dialog displays.

Figure 3–34 Invoke Wizard - Define Application View



8. Click **Import** and select **SAP**.

   The SAP Login dialog displays.

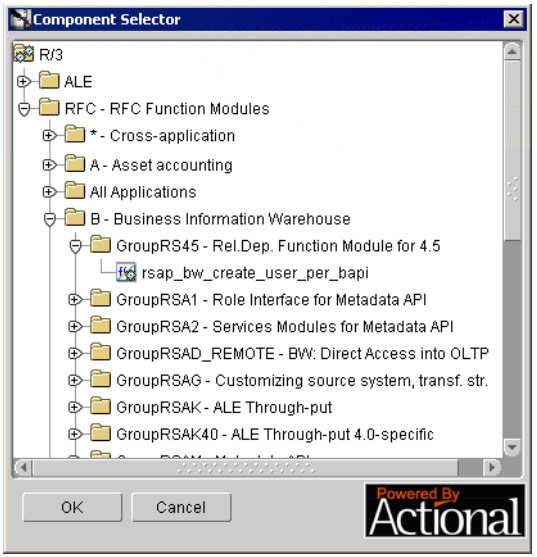
   If this is the initial login for this machine, enter the correct information.

   See Also: "Importing Attributes from SAP" on page 3-8

   If this machine has been logged in to SAP before, enter the password on the SAP Login dialog and click **OK**.

The Component Selector displays.

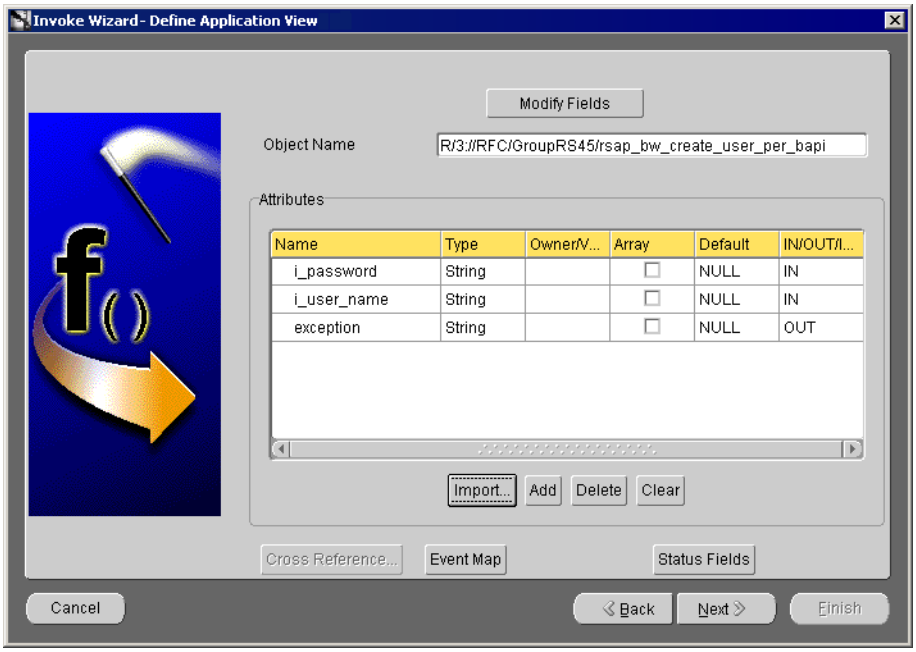
Figure 3–35 Component Selector - Remote Function Call



9. Expand the RFC - RFC Function Modules tree until the correct component displays for selection.
10. Select a component and click **OK**.

The populated Define Application View dialog displays.

Figure 3–36 Invoke Wizard - Define Application View



11. Click **Next**.
12. Click **New** to define mappings and click **Finish**.

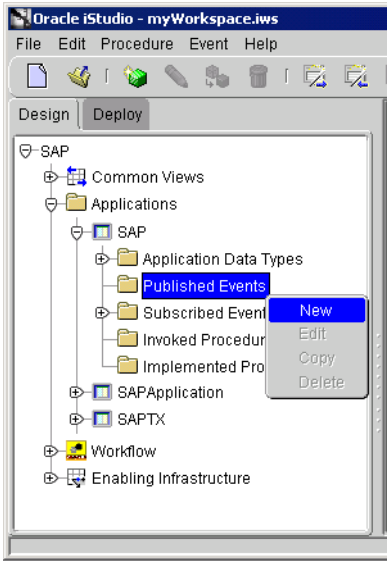
    The new populated event displays in the right panel of iStudio.

Creating a Remote Function Call Published Event

To create a Remote Function Call published event in iStudio:

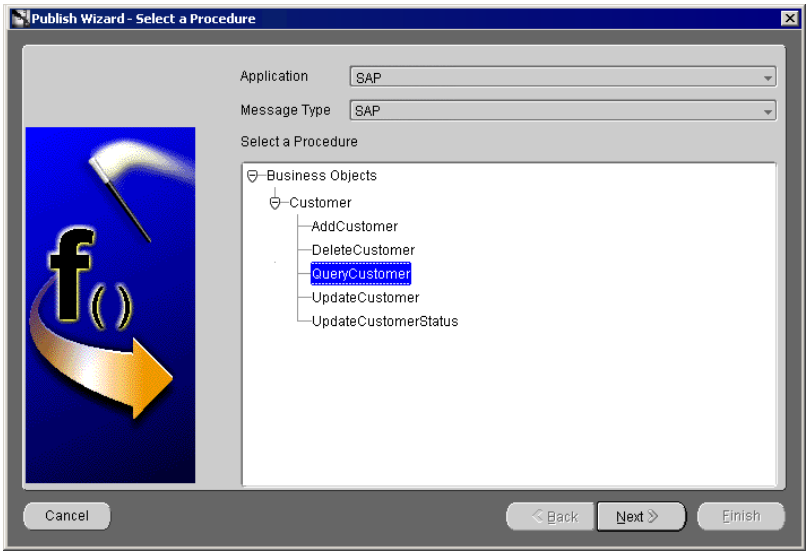
1. Start iStudio
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Published Events** and select **New**.

Figure 3–37 Creating a Published Event



The Publish Wizard—Select a Procedure dialog displays.

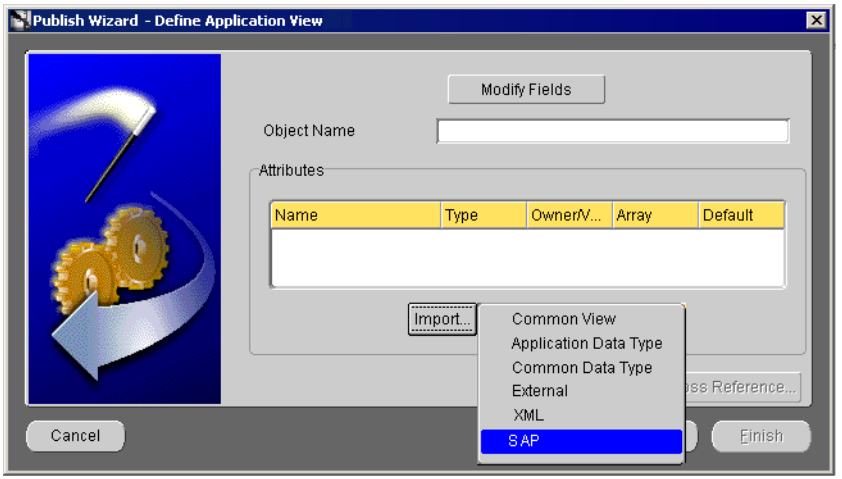
Figure 3–38 Publish Wizard - Selecting a Procedure



6. Select the Application and Message Type from the dropdown menus.
7. Select an event and click **Next**.

The Define Application View dialog displays.

Figure 3–39 Publish Wizard - Define Application View



8. Click **Import** and select **SAP**.

   The SAP Login dialog displays.

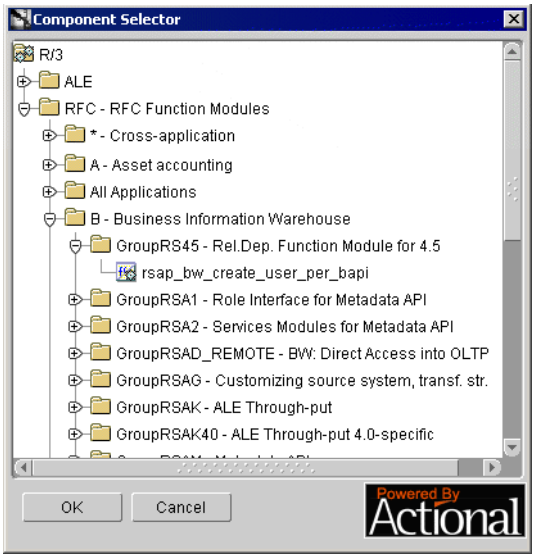
   If this is the initial login for this machine, enter the correct information.

   See Also: "Importing Attributes from SAP" on page 3-8

   If this machine has been logged in to SAP before, enter the password on the SAP Login dialog and click **OK**.

The Component Selector displays.

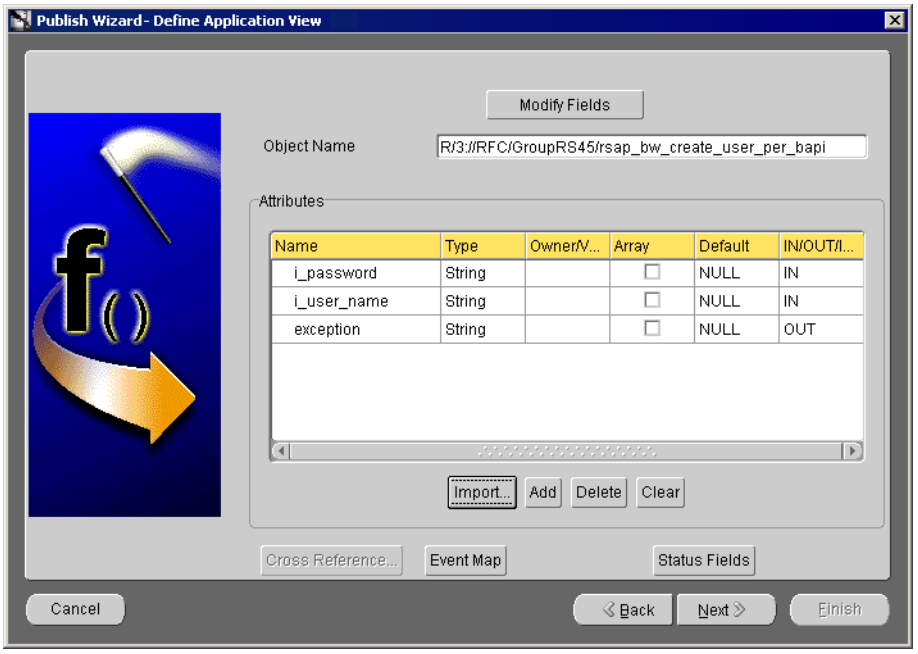
Figure 3–40 Component Selector - Remote Function Call



9. Expand the RFC - RFC Function Modules tree until the correct component displays for selection.
10. Select a component and click **OK**.

The populated Define Application View dialog displays.

Figure 3–41 Publish Wizard - Define Application View



11. Click **Next**.
12. Click **New** to define mappings and click **Finish**.

    The new populated event displays in the right panel of iStudio.

4

Application Link Enabling

Application Link Enabling (ALE) handles the exchange of messages across independent R/3 systems or between external systems and R/3. Application Link Enabling uses intermediate documents (IDOC) as a universal container for the information. Intermediate documents are used to upload to, or download data from, other systems.

This chapter discusses the following topics:

Frequently Used Application Link Enabling Transactions

Table 4–1 displays a list of frequently used Application Link Enabling transactions.

Table 4–1 Frequently Used Application Link Enabling Transactions

Transaction	Description
SALE	Application Link Enabling Customizing
BD21	Analyze change pointers - create intermediate documents from change pointer
BD12	Send customer master
BD61	Activate change pointer generally
BD54	Maintain logical systems
BD64	Maintain distribution model
BD71	Distribute customer model
BDM2	Cross-system intermediate documents reporting
WE02	Intermediate document Display
WE05	Intermediate document List
WE20	Maintain partner profile
WE21	Maintain port definition
WE30	Develop intermediate document types
WE31	Maintain intermediate document segment
WE60	Intermediate Documents Documentation - Intermediate document types
BDM7	Application Link Enabling Audit - statistical analyses
WE14	Process (dispatch) intermediate documents through port - RSEOUT00
WE16	Inbound file
WE42	Process code inbound
SARA	Central intermediate documents archive
WE47	Status code maintenance

Transaction	Description
WE82	Assign intermediate documents to message type
SM59	Maintain Remote Function Call destinations
SM37	Display batch jobs - job overview
SM50	Process overview
SLG1	Evaluate application log
SM21	System log
SM58	Transactional Remote Function Call monitoring
RZ12	Remote Function Call Server Group maintenance

Application Line Enabling Terminology

The following terms are described:

- Logical System
- Intermediate Documents Type
- Message Type
- OracleAS InterConnect Application Acting as a Client
- Application Acting as a Server

Logical System

A logical system is your R/3 representation. This is your R/3 address where you can distribute data to and from an R/3 system. Logical systems start with a base logical system.

See Also: "Define a Base Logical System" on page 4-12

The base contains your main address. From the base logical system, an SAP administrator creates partner logical systems.

See Also: "Define a Partner Logical System" on page 4-16

A base system uses the case sensitive Remote Function Call (RFC). To browse the Remote Function Call destinations from the SAP interface:

1. Click **Tools >Administration**.
2. Select **Network**.
3. Select **RFC destination**.
4. Select **TCP/IP Connections**.
5. Select the Remote Function Call destination to use.

Make sure the Remote Function Call points to the correct computer using the **System Information > Target System**. You can also verify your connection using **Test Connection**.

Ask the administrator of the logical system which RFC Destination to use.

Intermediate Documents Type

An intermediate document type represents the structure of the data associated with a message type. An intermediate document is a component with the data of a particular message type in it. Intermediate documents are data containers with intelligence built in. Each intermediate document contains only one business type.

Before a the development machine can send or receive intermediate documents of a certain type, it needs to know the intermediate document structure. An intermediate document consists of the following types:

- Control Record—Every intermediate document has one control record. The control record contains information about the intermediate document. For example, it contains the type of intermediate document, the message type, sender and receiver information, and direction (inbound or outbound). This information provides control data on an outbound intermediate documents and processing options on an inbound intermediate document.
- Data Record—An intermediate document contains one or more data records containing application data and consists of one or more data records. Its sequence and structure are dictated by the sequence and structure of segments in a given intermediate document type. For an outbound interface, Application Link Enabling function modules populate these segments with application data. For inbound Application Link Enabling interfaces, the application modules process the data contained in the segments.

- Status Record—With a length of 2 bytes, the status record contains information on the state of the intermediate document as it passes through various stages of processing. SAP assigns values between 01 to 41 for outbound intermediate documents and assigns values between 50 to 73 for inbound intermediate documents. The status record is a history of the intermediate document states containing dates and time-stamps.

Intermediate documents are identified by a unique intermediate document number (IDOCNUM) assigned by SAP. However, it is possible to manually assign a number range of intermediate documents.

Message Type

The message type represents the data exchanged between R/3 and an external system. A message type characterizes the data being sent across systems and relates to the structure of the data: an intermediate document type. For example, `MATMAS` is a message type for Material Master, and `IVOIC` is a message type for an Invoice. There are over 200 message types supported by Application Link Enabling in an R/3 system.

Access logical message types using the `/nwedi` transaction or by completing the following steps:

1. Select **Development**.
2. Select **IDOC types**.

   Using **Environment > Message types** retrieves a list of available message types.

To access an assignment of logical message types to intermediate document types, complete the following steps:

1. Select **Environment**.
2. Select **IDOC types/message**.

The main transaction in the R/3 system for intermediate documents handling is `/nwedi`. SAP documentation is available for intermediate document types and intermediate document segment types. The **IDOC>IDOC lists** menu accesses the list of intermediate documents created and received in an R/3 system.

OracleAS InterConnect Application Acting as a Client

If you want to make your OracleAS InterConnect application acts as a client sending intermediate documents, create a subscribed event or an implemented procedure. It is preferable to create a subscribed event because intermediate documents are more similar to events than request/reply pairs. When this event/procedure is triggered, an intermediate document is sent to the SAP system.

You must set up the Application Link Enabling general settings using the R/3 configuration editor to send an intermediate document to R/3.

If you browse the SAP system in iStudio, a pair of methods associated with each intermediate document displays. These methods are called `Send` and `AdvancedSend`. Events/procedures can be built around either of these. If you call the `Send` method it populates the control record of the intermediate document from the parameters set in the R/3 configuration editor. `AdvancedSend` allows more flexibility; if you use this method you must pass the control data to the method.

Application Acting as a Server

You can have your OracleAS InterConnect application act as a server receiving R/3 intermediate documents. R/3 sends an intermediate document to the development machine's Program ID. In order to receive an Application Link Enabling intermediate document, you must first register the RFC program ID. This is done by setting the RFC program ID in the R/3 configuration editor.

You can create a published event or an invoked procedure (events are preferred) to be triggered when an intermediate document is sent to your OracleAS InterConnect application. You must use the `AdvancedSend` method associated with that Application Link Enabling intermediate document to define your event/procedure. When an intermediate document is sent to your application, an appropriate OracleAS InterConnect message will be constructed and sent to the OracleAS InterConnect hub.

Application Link Enabling Subdirectories—Queue and Cache

The Application Link Enabling `Cache` and `Queue` directories, located under `<install_path>\...\config\ALE\<profileName>`, are created after Application Link Enabling parameters are set in the Configuration Editor. The `Cache` and `Queue` directories are required when manually downloading intermediate document structures (the `.mtd` file) from the SAP system to the local machine.

The `Queue` directory contains the queue of requests that were not sent. The requests are re-sent every `[RetryInterval]` minutes.

The `Cache` directory contains local descriptions of Application Link Enabling messages.

- Files with the extension of `.mtd`, for example, `<IDOCName>_<SAPVersionNumber>.mtd`, are created when downloaded using `RSEIDOC3`, with only the Display structure and Display segment fields set and one intermediate document type generated. Files of this name are automatically converted to `.ido` files by the SAP adapter. However, `.mtd` files do not convert to `.ido` files if an `.ido` file with the same name exists.
- Files named `.ido` are binary files containing the local representation of intermediate document messages. These are either downloaded from the R/3 system or they are built from `.mtd` files as needed.

Intermediate documents can be accessed at runtime by setting the Application Link Enabling Enable Remote Browsing parameter in the development machine's Configuration Editor only if the Enhanced Browsing Function Modules have been uploaded.

This parameter is used mostly for casual browsing as the `.ido` files are not saved locally and it can be slow. The preferred method is to use the SAP Parser method and download the `.mtd` files locally to generate the `.ido` file.

If, at runtime, an intermediate document definition is needed but no `.ido` file exists, then the development machine downloads the `.ido` file from an available R/3 system. However, if the R/3 system is down, nothing will work. In this case, for reliability, pre-download the intermediate document definitions to an `.mtd` file.

`The.mtd` files create the `.ido` files. The `.ido` files are compiled versions of the `.mtd` file. Delete the `.mtd` file after creating the `.ido` file. Calling the intermediate document, either by viewing them in the development machine's Browser or being called by the development machine, creates the `.ido` file from the `.mtd` only on the initial call.

If an `.ido` file exists in your `CACHE` directory for a specific intermediate document, the development machine uses the existing `.ido` file. If the `.mtd` file is then updated, the `.ido` file does not automatically update. To manually update the `.ido` file, delete the old `.ido` file. Calling the intermediate document causes the updated `.mtd` file to generate a new `.ido` file. If you customize an intermediate document definition locally, do not forget to update the intermediate document definition in R/3 and inform users of this change in the definition structure. Otherwise, the next time an intermediate document is sent, it uses the old definitions and conversion errors will occur. Users of the intermediate document definition must delete the `.mtd` and the `.ido` files for that intermediate document and download the `IDOC.mtd` to compile a new `.ido` file.

See Also:

- "How to Install the Remote Browsing Function Modules" on page 4-26
- "Manually Downloading an IDOC" on page 4-23

Queuing Inbound Intermediate Documents

If R/3 is down, the development machine cannot send inbound intermediate documents. It saves the intermediate documents for later transmission.

If you have not downloaded an `.ido` definition into an `.mtd` file, the development machine cannot transmit or queue your intermediate document.

See Also: "Manually Downloading an IDOC" on page 4-23

When sending Application Link Enabling intermediate documents to R/3, the runtime code retrieves, and uses, a connection from its connection pool. If R/3 cannot be contacted, for example, no connection is available, the intermediate document is queued to re-send later. If a copy of the development machine is running with the same profile used by the sending client, the agent scans the `Queue` directory. The SAP Agent sends the queued intermediate documents according to the user-specified retry interval.

Application Link Enabling General Settings

The General Settings panel of the Configuration Editor defines the general Application Link Enabling settings. It is available from either the Global Settings or a user-defined profile. Table 4–2 provides a description of the fields in the Configuration Editor.

Figure 4–1 Configuration Settings Editor - Application Link Enabling General Settings

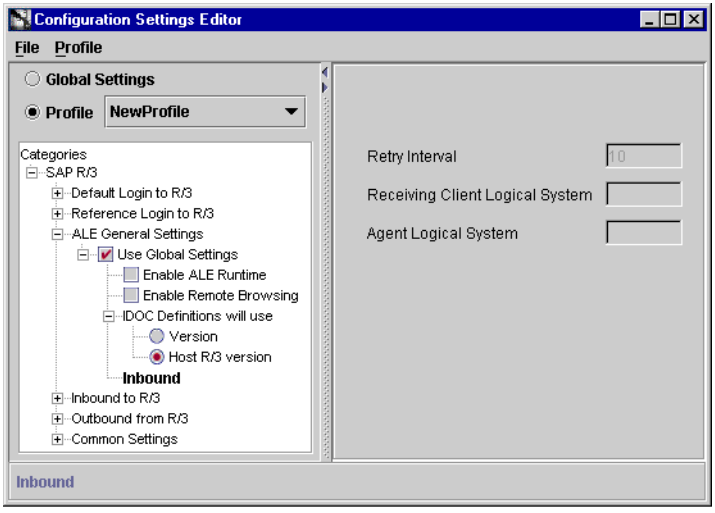


Table 4–2 Configuration Settings Editor - Application Link Enabling General Settings fields

Field	Description
Enable ALE Runtime	Activates the Application Link Enabling connections. Enables or disables the ability of the SAP adapter to be sent intermediate documents via the Remote Function Call destination defined by the Host.
Enable Remote Browsing	When unchecked, the Application Link Enabling adapter will not go to the R/3 system to retrieve the definition of Application Link Enabling messages. The SAP adapter only browses intermediate document definitions that were manually downloaded from the R/3 system. For more information, refer to the Manually Downloading an IDOC on page 4-23. **Pre-Requisite:** Upload the intermediate document browsing function modules provided into the SAP system. Application Link Enabling checks a setting before attempting to retrieve an intermediate document definition from R/3.
DOC Definitions will use Host R/3 version	A three letter, uppercase string used to specify that the intermediate document definitions should be those of the specified release of R/3. If the setting is blank (internally) this indicates that the latest R/3 version must use.
IDOC Definitions will use Version	A three letter, uppercase string used to specify that the intermediate document definitions should be those of the specified release of R/3. The first time the R/3 system version is required at runtime, the R/3 system is queried and the result is stored back into this key. The setting changes from **Use Latest Available** to the R/3 version.

Table 4–2 Configuration Settings Editor - Application Link Enabling General Settings fields

Field	Description
Inbound - Retry Interval	This is a 32-bit integer with valid values ranging from 1 to 2*24*60 to two days. It represents the number of minutes between retry attempts when re-sending a message to R/3.
Receiving Client Logical System ID	A 10-character string representing the SAP System Base logical system ID for the recipient of your intermediate document (created in the SAP System by a System Administrator). You are sending an intermediate document. This is the logical system ID associated with the SAP client to whom you are sending the intermediate document. If you use `AdvancedSend` in your code, you can set the parameters in control structure passed to the `AdvancedSend` method. If you use the `Send` method in your code, the development machine uses the values set in the ALE General Settings.
Agent Logical System ID	A 10-character string representing the development machine. This identification is created in the SAP System for the development machines by the System Administrator. You are sending an intermediate document. This is the logical system ID associated with the intermediate document source (the development machine) in R/3. If you use `AdvancedSend` in your code, you can set the parameters in the `AdvancedSend` method. If you use the `Send` method in your code, the development machine uses the values set in the ALE General Settings.

Inbound Intermediate Documents

If sending an inbound intermediate document from an OracleAS InterConnect application to R/3, set the following using the Configuration Editor:

- Receiving Client Logical System ID
- Agent Logical System ID
- Intermediate Document Version
- Default Login to R/3 - Host

Outbound Intermediate Documents

If sending an outbound intermediate document from R/3 to OracleAS InterConnect application, set the following using the Configuration Editor:

- Default Login to R/3 - Host
- Host and Program ID settings in Outbound From R/3 group

R/3 Application Link Enabling Configuration

Complete the following steps to configure the R/3 system to use Application Link Enabling functionality.

Step 1 Define a Base Logical System

To use Application Link Enabling functionality you must configure both SAP and the development machine. The first step is to identify a base logical system in your R/3 system. Using the SAPGUI ALE customizing menu, set up your client's base logical system.

To access the ALE customizing menu, either use the `/nSALE` transaction or the **Implementation Guide for R/3 Customizing (IMG) > Cross-Applications Components > Distribution (ALE)** menu selection and expand **Basic Configuration**. The logical system you create is the `sender` in outbound interfaces and the `receiver` in inbound interfaces. An SAP system administrator creates the base logical system as follows:

1. Expand the **Set up Logical System**.
2. Execute **Maintain Logical System**.
3. Click **New Entries**.
4. Enter the name and description of the logical system and save your data in the Change Request Query Data dialog. The table is client independent.

   For example, SAP recommends the naming standard for the base logical system as `XXXCLNTyyy`.

   where:

   - `xxx` is the instance.
   - `CLNT` is an identification name, for example, a client name.

- `yyy` furthers the client identification. For example, if the same client handles different `IDOC` structures you can differentiate them using numbers (`CLNT01`, `CLNT02`).

5. In the dialog box requesting a change, select an existing request if you have one open, or create a new one by clicking **Create Request** and entering a short description.

After setting your base logical system, assign the logical system to the client of the base logical system, thus creating partner logical systems. Access the panel using the `/nSCC4` transaction.

See Also: "Creating a Partner Profile" on page 4-17

To assign the logical system to the client of the base logical system:

1. Execute **Allocate Logical System to the Client**.
2. Find the entry of your client, `yyy`.
3. Double-click the row to select it and click the entry name for details.
4. Enter **`xxxCLNTyyy`** in the field for logical system and save your entry.

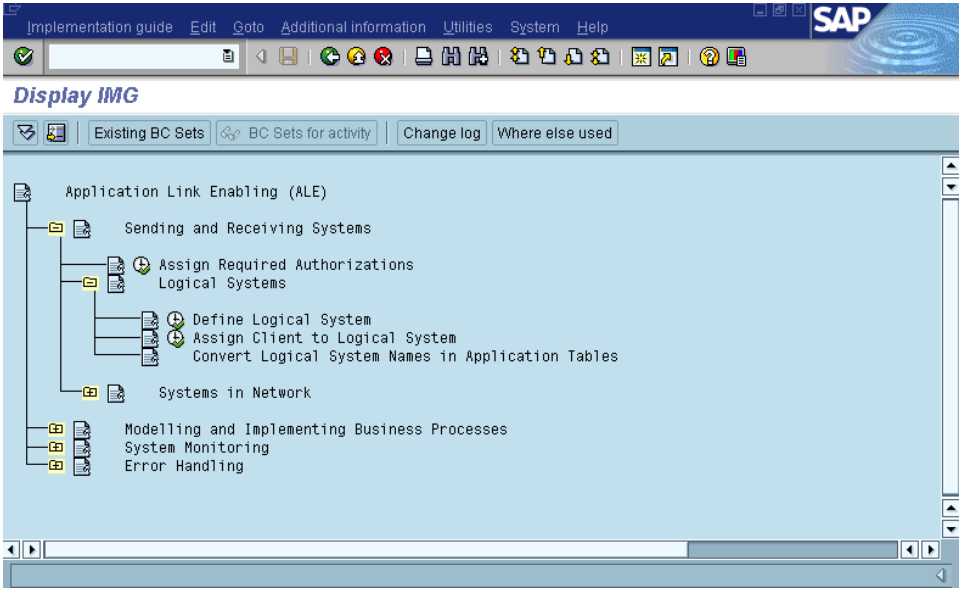
See Also: *SAP Implementation Guide for ALE* contains specific R/3 customization instructions on how to create, or find, an existing logical system

You can use either the `/nSALE` transaction or the menu:

1. Select **Tools**.
2. Select **Accelerated SAP > Customizing**.
3. Go to **Edit Project**.
4. Select **SAP Reference IMG**.
5. Open the **Cross-Application Components**.

6. Find **Distribution (ALE)** and expand the subsequent branches.

Figure 4–2 Display Structure: Distributed (ALE)



To use an existing logical system, you can access the current logical system list using the following:

- **Tools > ALE > Master Data**

Step 2 Sending a Material Master Intermediate Document

To send a material master intermediate document, use the `/nBD10` transaction or select from the menu:

- **Tools > ALE > Master Data Distribution > Cross-Application > Material > Send**

Step 3 Creating a Transactional Port

To communicate outside of the R/3 system, you need a transactional Remote Function Call port and a communication level through a Remote Function Call destination. A port is an SAP logical representation of a communication channel for intermediate documents. There are four types of ports that Application Link Enabling can use to distribute intermediate documents:

- Transactional Remote Function Call
- File
- R/2
- Internet

To create a transactional port:

1. Highlight the branch for the type of port you wish to define.
2. Click **Create** or **F7**.
3. Click the popup dialog or press Enter.

   The List of Port dialog displays.
4. Click **New Entries**.
5. Enter a description in the **Description** field, for example, Task Port.
6. Press **F4** in Logical destination to access the popup RFC Destination dialog.

   You are linking this port to a logical Remote Function Call destination to invoke certain processing on a server. Use the `/nsm59` transaction to create a Remote Function Call destination of type TCP/IP connection.
7. Double-click an existing Remote Function Call to display in the Logical destination field.
8. Click **OK** to accept your selection.

Step 4 Define a Partner Logical System

Based on an existing logical system, a partner profile is an identifier for a system used for communicating messages. There are four types of partner profiles of which `LS` (logical system) is used for Application Link Enabling communications.

A partner profile defines parameters of communication between two or more systems. Other than general information, you must maintain inbound parameters and message control. The main parameters are:

- Message types
- Intermediate document types
- Process codes
- Partner function
- Application identifier
- Message function
- Output type
- Port

There are parameters that also determine the mode of processing and error handling.

Partner profiles are the gateway for Application Link Enabling communications. They route specified messages through defined intermediate document types to a given port. This is after invoking the appropriate function modules for outbound processing. During this time, it receives intermediate documents of a specific type and identifies modules to post data to the application databases for inbound messages.

To maintain partner profiles use the following transactions:

- `/nwe20`
- `/sale`

To define a partner logical system:

1. Select the `/sale` transaction.
2. Select **Modeling and Implementing Business Process** > **Partner Profiles and Time of Processing** > **Maintain Partner Profiles Manually**.

3. Highlight the LS branch and press **F7**, or click **Create**.

   All Application Link Enabling partner profiles use `LS` as the partner type. `LS` is used for Application Link Enabling communications.

4. Enter a part number.

5. Select a **Base Logical System** and **LS** for Partner Type in the Partner Type and Number fields.

   Each client has its own base that represents it to the outside world. To send Application Link Enabling messages you need to start with a base.

Step 5 Creating a Partner Profile

Complete the following steps to create a partner profile:

1. Enter **`xxxCLNTyyy`** in the Partn. number field using the `/nsale` transaction.

   This is either the base logical system you created, or an existing logical system. Every partner profile used for Application Link Enabling must be based on an existing logical system.

   See Also: "Define a Base Logical System" on page 4-12

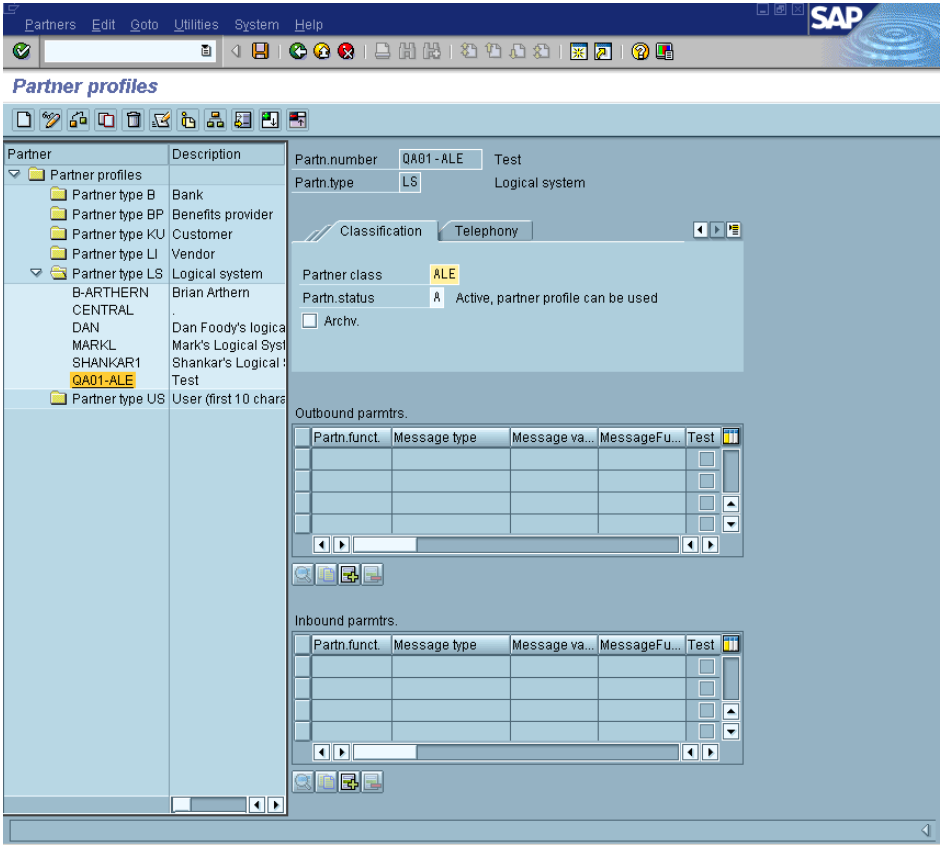
   For example, SAP recommends the naming standard for the base logical system as `XXXCLNTyyy`.

   where:

   - `xxx` is the instance.
   - `CLNT` is an identification name, for example, a client name.
   - `yyy` furthers the client identification, for example, if the same client handles different intermediate document structures you can differentiate them using numbers (`CLNT01, CLNT02`).

2. Enter the code or use the dropdown menu to select from the existing Partner's listing to set your partner number, **Partn. number**.

Figure 4–3 Partner Profiles: Initial Screen



3. Enter **LS** in the Partn. type field.

   All Application Link Enabling partner profiles use LS as the partner type. LS is used for Application Link Enabling communications.

4. Click the Classification tab.

5. Click **Create** or **F7**.

6. Enter **ALE** in the Partner class field.

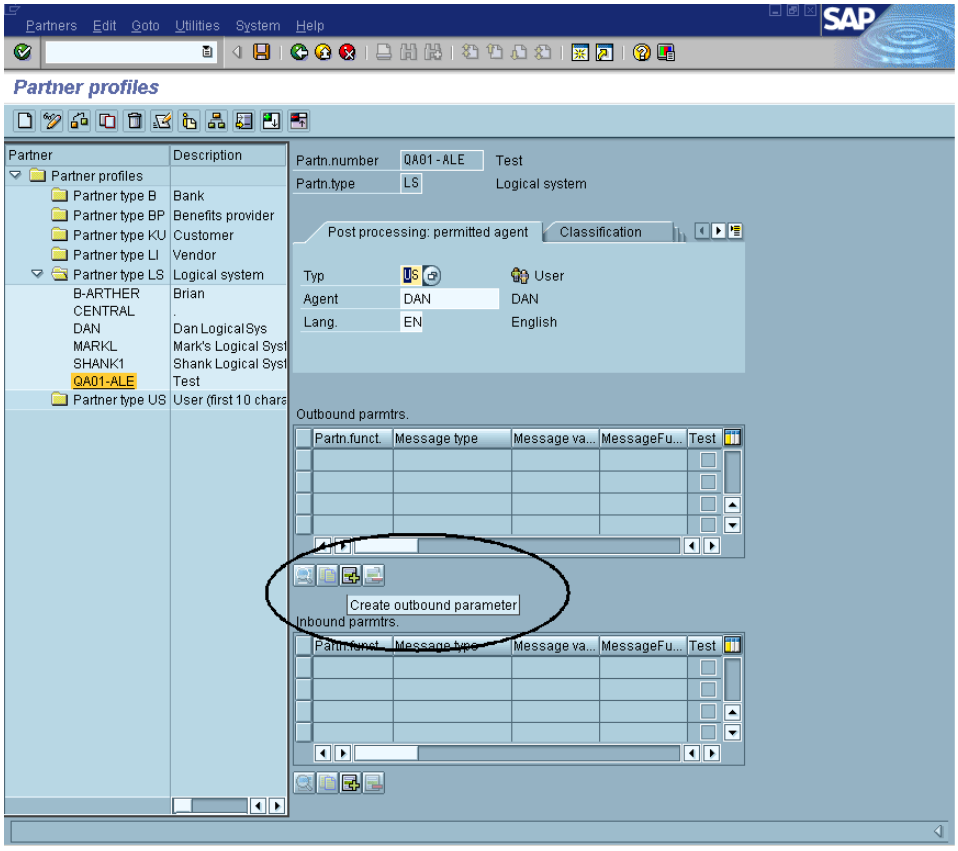
7. Enter **A** in the Partn. status field.
8. Click **Save** to create the partner.

Step 6 Maintaining Outbound Parameters

To maintain the outbound parameters:

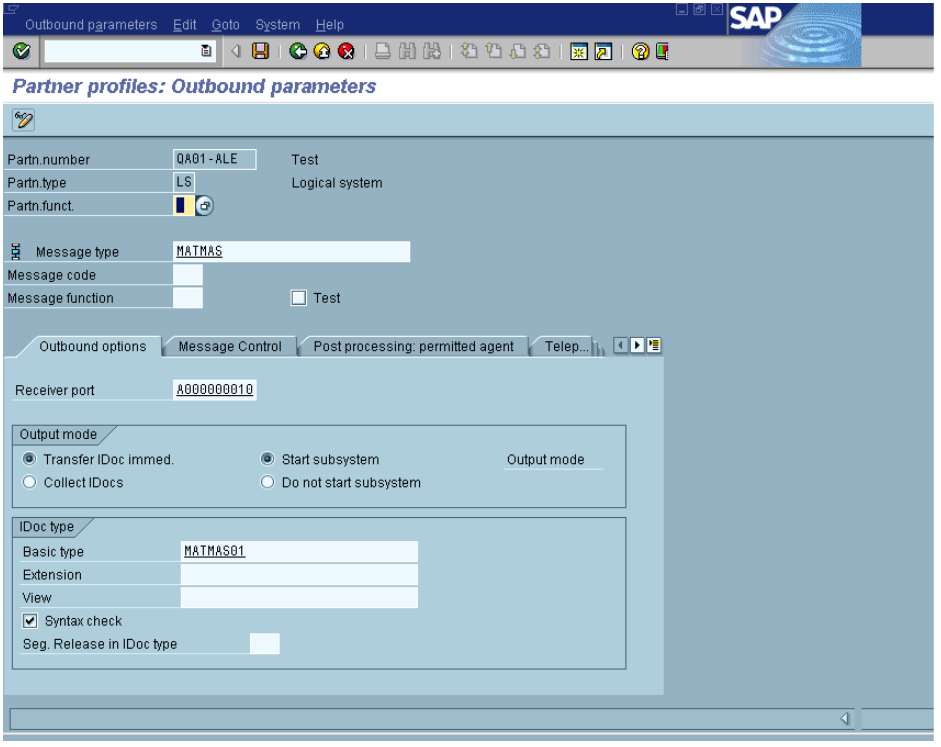
1. Click **Outbound Parameters**.

Figure 4–4 CreateOutbound Parameters Button



2. Enter your message type in the relevant input fiends, for example, Message type > MATMAS.

Figure 4–5 Enter the Information



3. Enter the transactional port previously created in the Receiver Port field.
4. Set the pack size.

   The pack size is the number of intermediate documents sent in a single dispatch.
5. Check **Transfer IDOC immed.** in Output Mode.

   You can also select **Collect IDocs** or **Do not start subsystem**. The first parameter instructs Application Link Enabling communication layer to collect all intermediate documents until further processing is requested. The second parameter is used to invoke third-party translation software.

6. Access the popup screen using the down arrow in the IDoc Basic Type field.

   Intermediate Document Basic type browsing displays available intermediate documents for that message type. Enter a basic type, for example, MATMAS02.

   Using this dialog, you can specify multiple message types.

7. Save your selections.

8. Press **F3** to return to the previous screen and view your settings.

Step 7 Customer Distribution Model

The Customer Distribution Model stores information about the flow of messages across various systems. It stores data that dictates which messages (messages types) flow to which logical system. Many messages can flow to one logical system, and one message can flow to several systems.

To create a Customer Distribution Model in the R/3 system with the client's base logical system as the sender logical system, either use the `/nBD64` transaction or complete the following to use the menu:

1. Select **Tools > Accelerated SAP > Customizing > Edit Project**.

2. Press **F6** for the Enterprise IMG.

3. Expand **Basic > Distribution (ALE) > Modeling and Implementing Business Processes > Cross-Application Settings**.

4. Open **Maintain Distribution Model and Distribute Views**.

5. Select **Transaction** and double-click **Maintain Customer distribution model directly**.

6. Click **Outbound** parameters.

7. Continue with the SAP dialogs to define your parameters.

Application Link Enabling—Exploring Intermediate Document Types

There are two ways to use the development machine and R/3 to explore intermediate documents. You can manually download the intermediate documents to your local machine, or you can use the development machine's Enhance Browsing steps.

If you only work with a few intermediate documents, it is recommended that you manually download the intermediate documents.

See Also: "Manually Downloading an IDOC" on page 4-23

If you use multiple intermediate document structures, you can enhance Application Link Enabling browsing by adding a few items provided with the development machine in your SAP system. By uploading the development machine source code into your SAP system, you can download all of the intermediate document definitions from your SAP system to your local machine for automatic browsing.

Uploading the development machine source code into the function modules may have been done. Perform a simple check by completing the following:

1. Navigate through **Tools > ABAP/4 Workbench > Development Function Builder**.
2. Set **Function module** to Z_RPY_IDOCTYPE_READ_DEFN3.
3. Click **Global data**.
4. Click **Display**.
5. Verify that the screen displays the following:

```
function-pool zmas. MESSAGE-ID ..
include lediдtyp.
```

See Also: "How to Install the Remote Browsing Function Modules" on page 4-26 if `include lediдtyp` does not display

The development machine includes the text files for `Z_RPY_IDOCTYPE_LIST (idoclist.asc)` and `Z_RPY_IDOCTYPE_READ_DEFN3 (idocread.asc)` in the `install_directory\SAP\ALE_Files` directory. Use these files to upload the source code into the function modules.

Manually Downloading an IDOC

To download intermediate documents, you must have the `Cache` and `Queue` directories under `\install_directory\config\ALE\profileName`.

If you do not have these directories, use the **Configuration Editor-> R/3-> ALE-> General Settings** menu and check **Enable ALE Runtime**. Re-starting the OracleAS InterConnect application creates the `Cache` and `Queue` directories.

To manually download intermediate document definitions from an R/3 system to your the development machine server, complete the following steps:

1. Select **R/3 Settings > ALE General Settings** in the development machine Configuration Editor.
2. Verify that **Enable ALE Runtime** is selected and that **Enable Remote Browsing** is not checked.
3. Save the settings and exit the Configuration Editor.

   Note: Delete any existing `.ido` files for that intermediate document from your cache directory

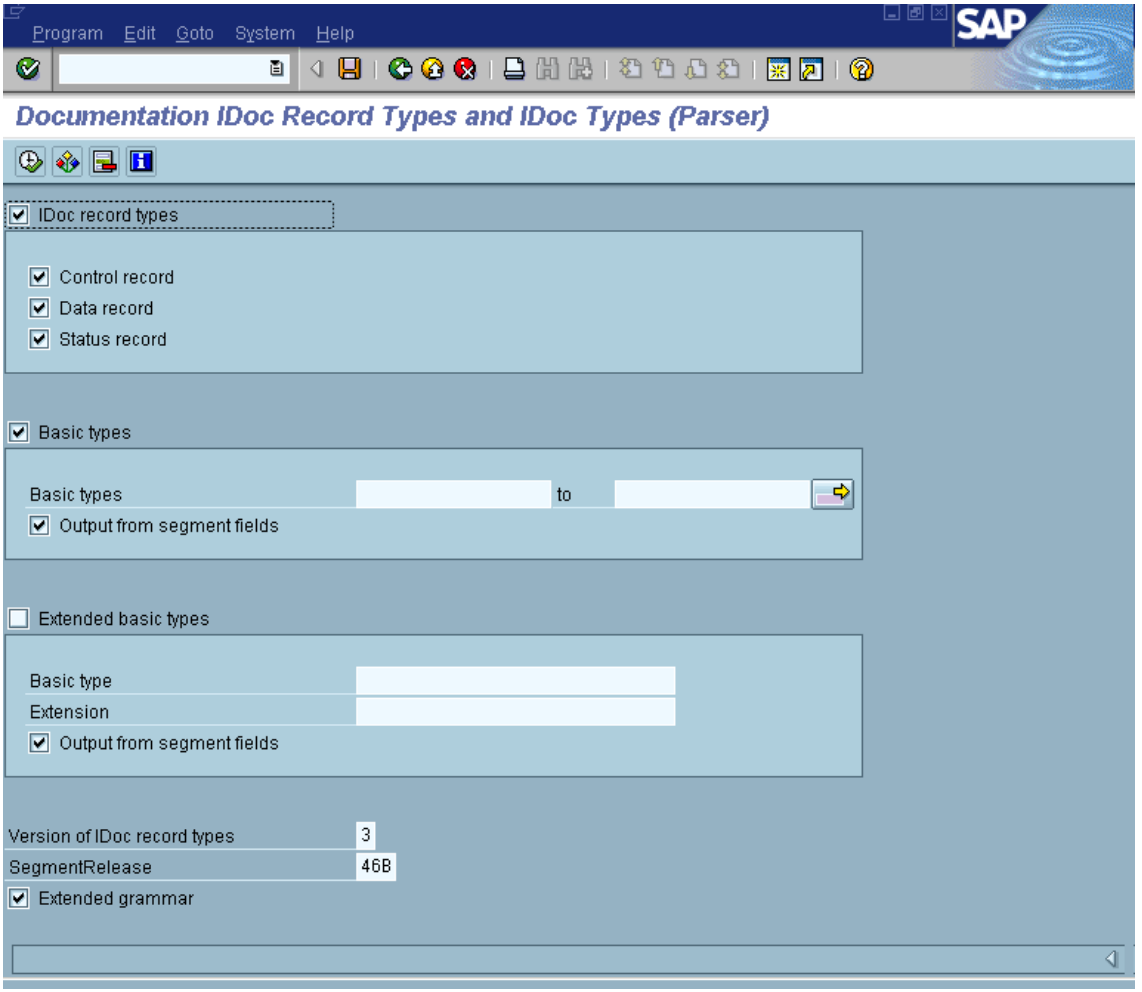
4. Log into an R/3 System.

   The SAPGUI Easy Access dialog displays.
5. From the main R/3 menu, expand **Tools > Business Communications > IDOC**.

   The Process technology tree displays.
6. Expand **IDOC > IDOC Basis**.

7. Expand **Documentation > IDOC type (parser).**

Figure 4–6 Documentation IDoc Record Types and IDoc Types (Parser)



8. Select **MATMAS01** from the Sel. of IDOC types popup dialog in Idoc Basic types.
9. Click the **check mark** to accept and load your selection.
10. Press **F8** or **Execute** to run.

11. Select **List > Download** in the Documentation IDoc Record types and IDOC Types (parser).
12. Select **unconverted** in the Save list in file popup menu.
13. Enter the following in the Transfer List to a Local File type `matmas02_31H.mtd`:

    ```
    install_path\config\ALE\profileName\Cache\IDOCName_SAPVers.mtd
    ```

14. Click **OK**.
15. Exit the R/3 session.

    In the iStudio browser, when you browse Application Link Enabling intermediate documents, the downloaded information, retrieved from your machine, displays without logging into an R/3 session.

Enhance Application Link Enabling Remote Browsing

To enhance your Application Link Enabling remote browsing, the development machine includes the text files for `Z_RPY_IDOCTYPE_LIST (idoclist.asc)` and `Z_RPY_IDOCTYPE_READ_DEFN3 (idocread.asc)` in the `install_directory\SAP\ALE_Files` directory.

This procedure is optional. If you only work with a few intermediate documents, it is recommended that you manually download the definitions (`.mtd` files) for use with the development machine.

See Also: "Manually Downloading an IDOC" on page 4-23

If you use multiple intermediate document structures, you can use the following to enhance Application Link Enabling remote browsing. `IDOCName_SAPVersionNumber.ido` files download automatically at runtime if you have selected Enable Remote Browsing and you are using the development machine's enhanced browsing function modules. Downloading `IDOCName_SAPVersionNumber.ido` files at runtime is time-consuming. For example, running the Application Link Enabling sample generates the `matmas_31H.ido` automatically (unless it was already manually downloaded because `.ido` files are over-written and the original file remains). Also, if you send or receive intermediate documents and the R/3 system goes down, you do not receive an error message; the message is queued and the message is sent the next time the system is functional.

Note: Remove any existing `.ido` and `.mtd` files from your local system as the structure is changed and any existing `.ido` will not be updated with the new structure.

How to Install the Remote Browsing Function Modules

The following steps create:

- Four data dictionary structures
- A function group
- Two function modules: `Z_RPY_IDOCTYPE_LIST` and `Z_RPY_IDOCTYPE_READ_DEFN3`

Why a Function Group?

The function modules must be in the same function group (usually custom built) and the global data (shared by the entire function group) must contain the `include lediditype` statement.

To begin, create the following four Data dictionary structures:

- `ZRPYIDCTXT`—IDOC Text Description
- `ZRPYIDCTYP`—IDOC Header information
- `ZRPYIDCSG3`—IDOC Segment header
- `ZRPYIDCFD3`—Information about field of an intermediate document segment

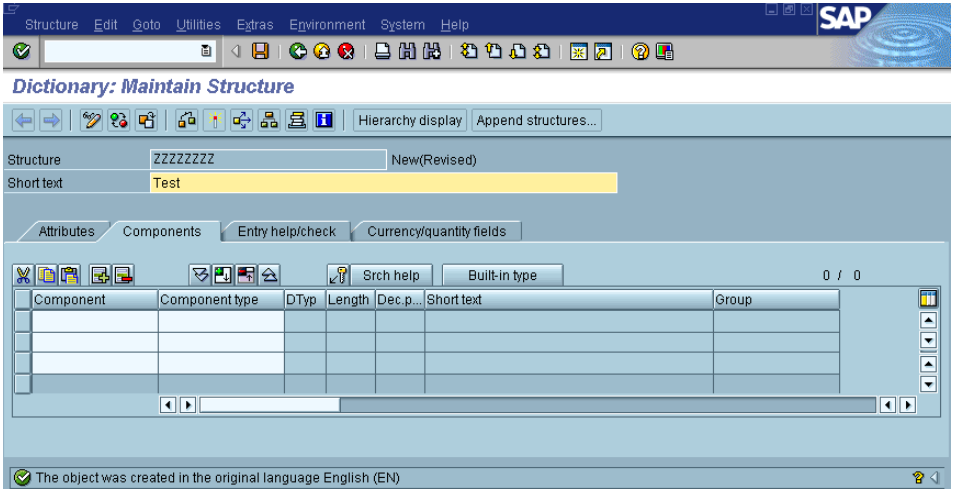
Create Four Structures

To create a structure, use the following pattern for each structure, for example using `ZRPYIDCTXT` in the SAPGUI, execute the `/nse11` transaction, or complete the following steps:

1. Select **Tools > ABAP/4 Workbench > Development > ABAP/4 Dictionary**.
2. Click **Data Type**.
3. Enter a table name in the Object name field, for example, `ZRPYIDCTXT`.
4. Click **Create** or **F5**.
5. Select **Structure**.
6. Enter a description in the Short text field, for example, IDOC Text Description.

7. Click **Client Type Entry** to ensure the transaction is in direct type entry mode. (Data Element input fields are disabled.)

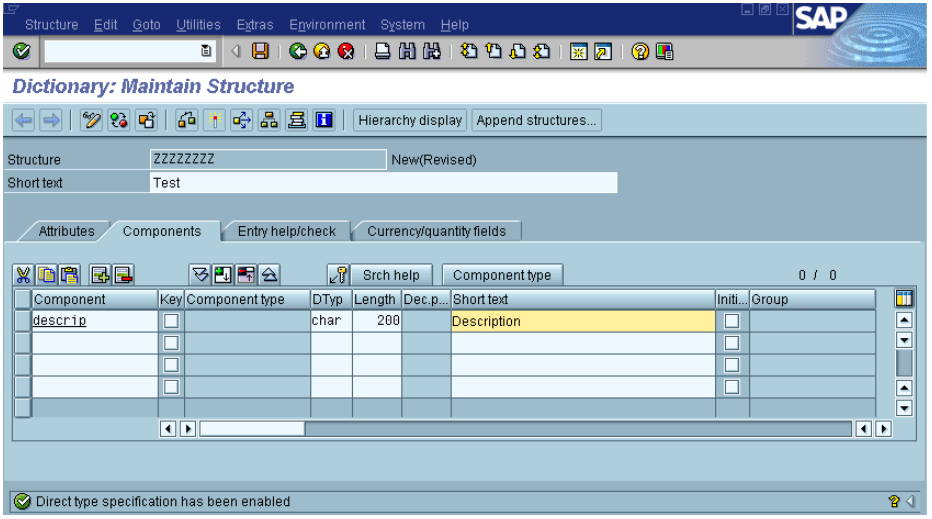
Figure 4–7 Dictionary:Table/Structure Change Fields



8. Click **Built-in Type** to switch to Direct Type Entry.
9. Enter the Component, for example, `DESCRIP`.
10. Enter DTyp, for example, CHAR.
11. Enter Length, for example 200.
12. Enter Short Text, for example, Description.

13. Click **Enter** to finish creating the field after entering the information from the table.

Figure 4–8 Dictionary:Table/Structure:Display Fields



14. Press **F11** to save the table.

    The Create object catalog entry dialog displays.

15. Complete the necessary fields.

16. Verify how to store the component with your system administrator. Select one of the following:

    - Local object—A non-transportable temporary component.
    - Development classes—Allow entities to take part in the R/3 transport mechanism. Changes to components in these classes are recorded and can be transported to other systems.

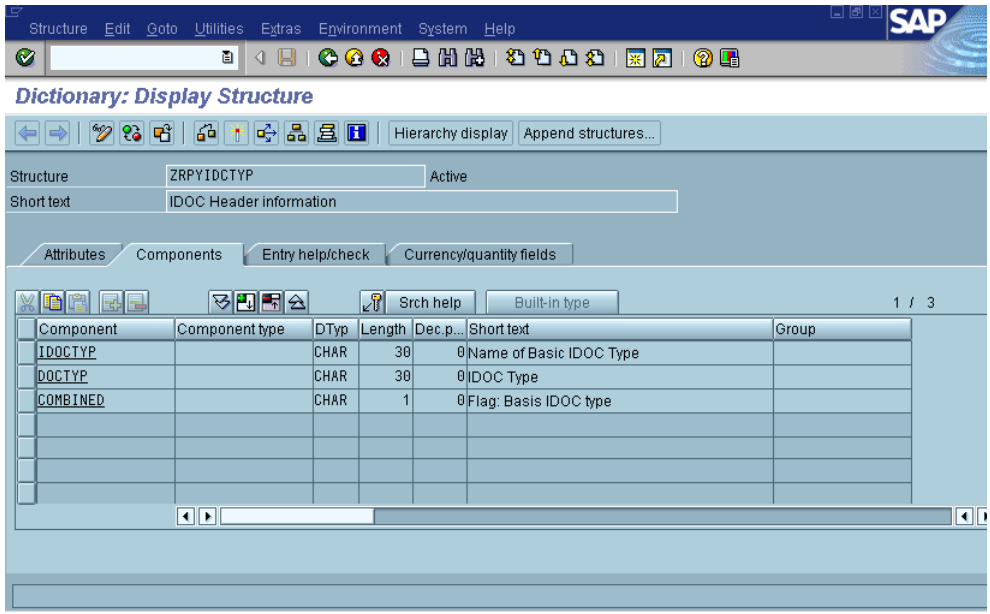
17. Press **Ctrl+F2** to check the consistency of the structure.

18. Press **Ctrl+F3** to activate the structure.

19. Press **F3** to return to the previous screen.

Repeat these steps for the other three structures, ZRPYIDCTYP, ZRPYIDCSG3, and ZRPYIDCFD3. Use the following information for all Filed name and Data elem fields.

Figure 4–9 Dictionary:Table/Structure:Display Fields



Create the ZRPYIDCTYP Structure Figure 4–10 and Table 4–3 describe creating the ZRPYIDCTYP structure.

Figure 4–10 Dictionary: Display Structure

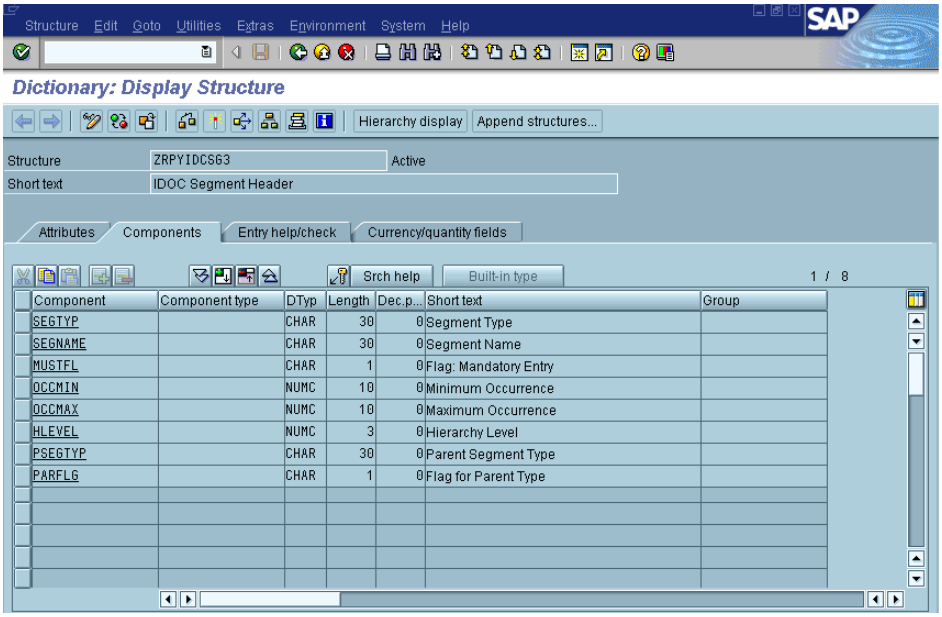


Table 4–3 Dictionary:Table/Structure:Display Fields

Field name	Type	Length	Short Text
IDOCTYP	CHAR	30	Name of Basic intermediate document type
DOCTYP	CHAR	30	Intermediate Document Type
COMBINED	CHAR	N/A	Flag: Basis intermediate document type

Create the ZRPYIDCFD3 Structure Figure 4–11 and Table 4–4describe creating the ZRPYIDCFD3 structure.

Figure 4–11 Dictonary:Table/Structure:Display Fields

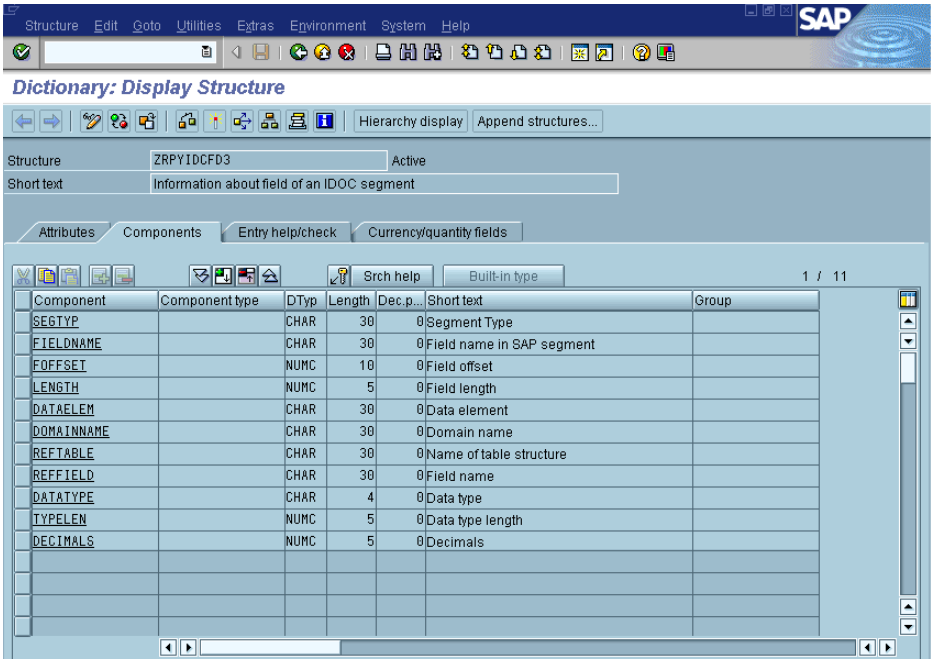


Table 4–4 Dictionary:Table/Structure:Display Fields

Field name	Type	Length	Short Text
SEGTYP	CHAR	30	Segment type
SEGNAME	CHAR	30	Segment name
MUSTFL	CHAR	1	Flag: Mandatory entry
OCCMIN	NUMC	10	Minimum occurrence
OCCMAX	NUMC	10	Maximum occurrence
HLEVEL	NUMC	3	Hierarchy level
PSEGTYP	CHAR	30	Parent segment type
PARFLG	CHAR	N/A	Flag for parent segment

Create a Function Group

To create a function group using the SAPGUI:

1. Enter the `/nse37` transaction.
2. Select **Goto > Function groups > Create Group**.
3. Enter the function group name in the Function group field. For example, ZMAS.
4. Enter the group description in the Short text field.
5. Click **Save**.

   The Create object catalog entry dialog displays.
6. Complete the necessary fields for this dialog.
7. Verify how to store the component with your system administrator. Select **Local Object** or select a **Development Class**.

Create Two Function Modules

To create the `Z_RPY_IDOCTYPE_LIST` and `Z_RPY_IDOCTYPE_READ_DEFN3` function modules in the SAPGUI:

1. Enter the `/nse37` transaction.
2. Select **Function Library**.

   The Function Library: Initial Screen dialog displays.
3. Enter the function module name in the Function module field, for example, `Z_RPY_IDOCTYPE_LIST`.
4. Click **Create**.
5. Enter the following values for each of the Object components selections: **Attributes**, **Import**, **Export**, **Changing**, **Tables**, **Exceptions**, and **Source Code**.

Figure 4–12 Object Components

Create a Z_RPY_IDOCTYPE_LIST Function Module The following section describes creating the Z_RPY_IDOCTYPE_LIST function module.

- Administration
  - Beside Classification > Function Group, enter the following object components:
    * Enter **ZMAS**.
    * Under Classification > Application, enter **Z**.
    * Under Classification > ShortText, enter **Retrieve details about one intermediate document type**.
  - Under Processing Type, check the following boxes:
    * **Remote Functional Call supported**
    * **Immediate Start**
- Import/Export Parameter Interface

  Figure 4–13 displays the function module display for the `Z_RPY_IDOCTYPE_LIST`.

Figure 4–13 Function Module Display:Import/Export Parameters Z_RPY_IDOCTYPE_LIST

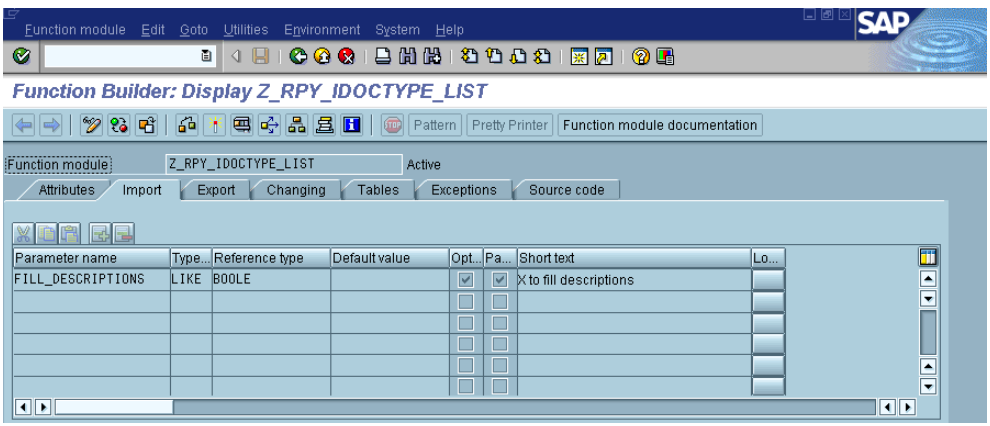


Table 4–5 Function Module Display:Import/Export Parameters Z_RPY_IDOCTYPE_LIST

Import parameter	Reference field	Proposal	Optional	Short Text
FILL_DESCRIPTIONS	BOOLE	X	N/A	x to fill descriptions

- Table Parameters/Exceptions Interface

  Figure 4–14 displays the function module display for the Z_TPY_IDOCTYPE_LIST.

Figure 4–14 Function Module Display:Import/Export Parameters Z_TPY_IDOCTYPE_LIST

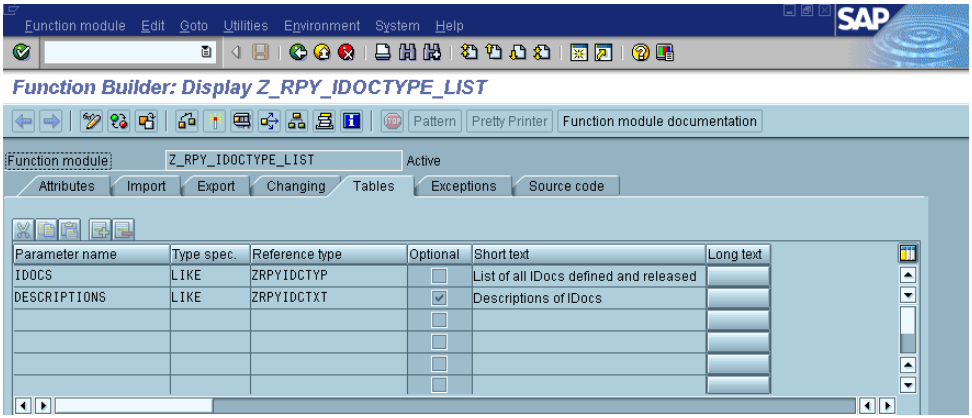


Table 4–6 Function Module Display:Import/Export Parameters Z_RPY_IDOCTYPE_LIST

Table Parameters	Ref. Structure	Optional	Short Text
IDOCS	ZRPYIDCTYP	N/A	List of intermediate document's defined and released.
DESCRIPTIONS	ZRPYIDCTXT	X	Description of intermediate documents.

- Documentation

  To access documentation:

  1. Click the **Source Code** tab.

     Upload the Source Code provided with the development machine using SAP's command, **Utilities > More Utilities > UpLoad/DownLoad > UpLoad**.

     R/3 Version 3.x—Enter the path and file name, `install_directory\SAPALE_Files\idoclist.asc`.

     R/3 Version 4.x—Enter the path and file name, `install_directory\SAP\ALE_Files\idoclist40.asc`.

  2. Press **Ctrl+F3** to activate the function module.

Create a Z_RPY_IDOCTYPE_READ_DEFN3 Function Module This section describes creating a `Z_RPY_IDOCTYPE_READ_DEFN3` function module.

- Administration
  - Beside Classification > Function Group, enter the following object components:
    * Under Classification > Application, enter **Z**.
    * Under Classification > ShortText, enter **Retrieve details about one intermediate document type**.
  - Under Processing Type, check the following boxes:
    * **Remote Function Call support**
    * **Immediate Start**

- Import/Export Parameter Interface

  Figure 4–15 displays the function module display for the Z_RPY_IDOCTYPE_READ_DEFN3 function module.

Figure 4–15 Function Module Display:Import/Export Parameters Z_RPY_IDOCTYPE_READ_DEFN2

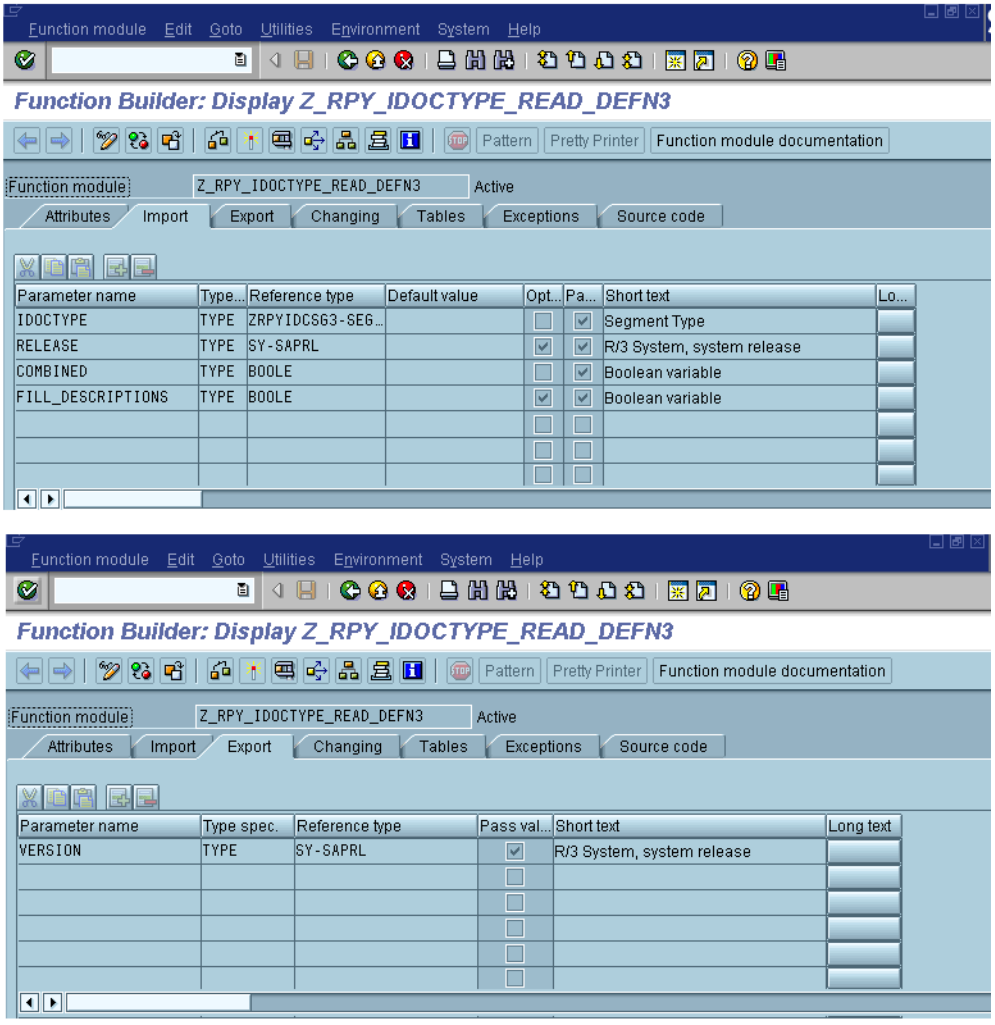


Table 4–7 Function Module Display: Import Parameters Z_RPY_IDOCTYPE_READ_DEFN2

Import parameter	Reference field	Proposal	Optional	Short Text
IDOCTYPE	ZRPYIDCSG3-SEGTYP	N/A	N/A	Segment
RELEASE	SY-SAPRL	SY-SAPRL	X	R/3 Systems, system release
COMBINED	BOOLE	N/A	X	Boolean Variable
FILL_DESCRIPTIONS	BOOLE	X	X	Boolean Variable

Table 4–8 Function Module Display: Export Parameters Z_RPY_IDOCTYPE_READ_DEFN2

Export Parameter	Reference Field	Short Text
VERSION	SY-SAPRL	R/3 System, system release

If you get an error and cannot continue, ensure all reference fields have been activated.

- Table Parameters/Exceptions Interface

  Figure 4–16 displays the table parameters and exceptions for the Z_RPY_IDOCTYPE_READ_DEFN3 function module.

Figure 4–16 Function Module Display: Table Parameters/Exceptions: Z_RPY_IDOCTYPE_READ_DEFN3

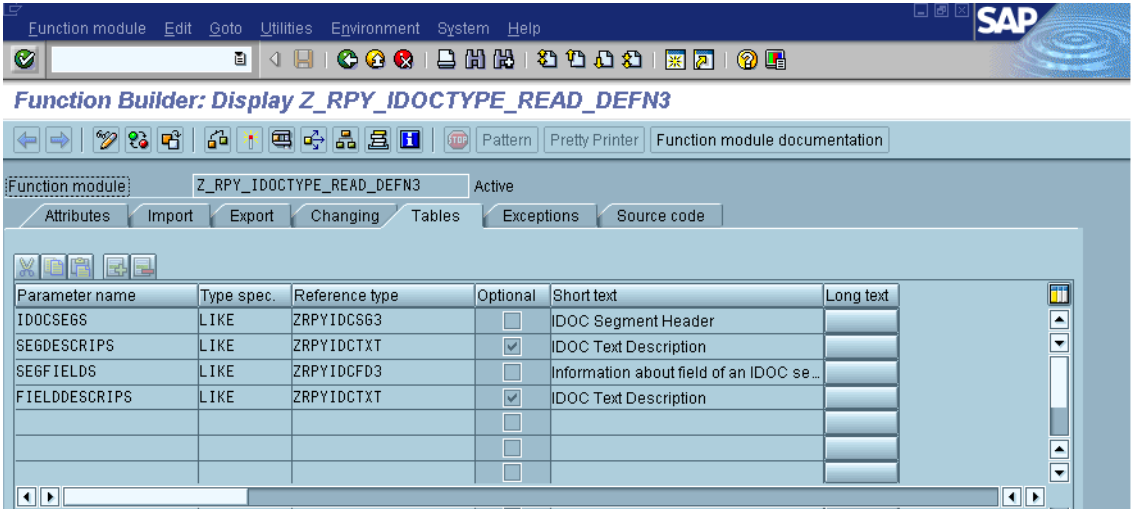


Table 4–9 Function Module Display:Import/Export Parameters Z_RPY_IDOCTYPE_READ_DEFN3

Table Parameters	Ref. Structure	Optional	Short Text
IDOCSEGS	ZRPYIDCSG3	N/A	IDOCSegment Header
SEGDESCRIPS	ZRPYIDCTXT	X	IDOC Text Description
SEGFIELDS	ZRPYIDCFD3	N/A	Information about Field of IDOC
FIELDDESCRIPS	ZRPYIDCTXT	X	IDOC Field Description

Upload the Source Code

To upload the source code:

1. Click the **Source Code** tab.

2. Upload the Source Code provided with the development machine using SAP's command, **Utilities > More Utilities > UpLoad/DownLoad > UpLoad**.
3. Enter the path and file name, `install_directory\SAP\ALE_Files\idocread.asc`.
4. Press **Ctrl+F3** to activate the function module.
5. Select **Global Data** and click **Change**.
6. Locate the following line:

   ```
   function-pool zmas.MESSAGE-ID .,
   ```

7. Insert the following:

   ```
   include ledidityp.
   ```

   To verify you have access to this global data, perform a simple check:

   * Navigate through **Tools > ABAP/4 Workbench > Function Builder**.
   * Set **Function module** to `Z_RPY_IDOCTYPE_READ_DEFN3`.
   * Check **Global data**.
   * Click **Display**.
   * Verify the following displays:

     ```
     function-pool zmas.MESSAGE-ID ..
     include ledidityp.
     ```

5

Remote Function Call

Remote Function Call is a feature of SAP R/3 that allows function modules to be invoked locally or remotely. This chapter describes how the SAP adapter may be integrated with SAP R/3 Remote Function Call.

This chapter discusses the following topics:

Introduction to Remote Function Call

A function module is a unit of functionality in SAP. Remote Function Call is a feature of R/3 that allows you to invoke function modules remotely. This allows the R/3 system to be integrated with other systems. The interface of every function is maintained by the SAP system in its data dictionary. A key concept in Remote Function Call is the Remote Function Call Program ID. This is a symbolic ID associated with an end point that services Remote Function Calls. The receiving server first needs to register the ID with the calling SAP system. Users in the SAP system may now invoke a call to a function at this Remote Function Call destination.

SAP Adapter Interaction with R/3

The following two sections describe the interaction between the SAP adapter and R/3 Remote Function Call.

SAP Adapter Application Acting as a Client

To make your SAP adapter application act as a client calling a Remote Function Call interface, you must define a subscribed event or an implemented procedure in your application. It is preferable to use implemented procedures for this application. Triggering of your event/procedure fires a call to the underlying SAP Remote Function Call interface.

Before you can define these events or procedures, you need to do is to set up a default login into the R/3 system.

SAP Adapter Application Acting as a Server

To make your SAP adapter application act as a server implementing a Remote Function Call interface, you must define a published event or an invoked procedure in your application. For use with Remote Function Call, an invoked procedure is preferred. A call to this Remote Function Call interface triggers your event/procedure and causes an appropriate message to be sent to the SAP adapter hub.

Note: The interfaces you define must already be in the SAP system's data dictionary. The SAP system does not need to provide an implementation for these interfaces; however, it must know the signatures. You must setup the default login parameters. Another parameter you need to set up is the Remote Function Call program ID. These parameters are set in the R/3 Configuration Editor.

Remote Function Call Configuration

The following configuration parameters must set using the Configuration Editor for working with Remote Function Call.

Calling From SAP Adapter to R/3

This section describes default login parameters for R/3.

Default Login to R/3

Default Login to R/3 allows you to program your development application to automatically connect to R/3 servers.

Default Login to R/3 authenticates a user's runtime credentials. This group only appears under a user-defined profile. All the parameters on this page are identical to those that appear when logging into R/3 for a regular session.

Table 5–1 Login to R/3 Field Descriptions

Field	Description
Enable Login Settings	Enables or disables the selected login feature.
Client	Enter your client number for the R/3 system.
User	Enter your user ID for the R/3 system.
Password	Specifies your user password for the R/3 system.
Host	Specifies the Host ID when connecting to the R/3 system.
Language	Required by R/3. By default this parameter retrieves the language information from the user's operating system.
Additional Connection Parameters	Passes additional string connection parameters when OracleAS InterConnect acts as a Remote Function Call client connecting to an R/3 server.

Table 5–1 Login to R/3 Field Descriptions

Field	Description
Debugging - ABAP/4	This feature is useful for debugging or diagnostic purposes. However, it is of limited use in a production environment, as the message does not display on the client machine. ABAP/4 Debug Calls are also known as Remote Function Call Debug Calls. Use this selection when you are debugging Function Modules. Selecting ABAP/4 Debug Calls automatically sets the `ABAP_DEBUG` connection parameter allowing Function Module calls to go through the SAPGUI debugger.

Calling from R/3 to the SAP Adapter Adapter

Default login to R/3 needs to be set.

See Also: "Default Login to R/3" on page 5-3

Table 5–2 Calling from R/3 to SAP adapter

Field	Description
Host	Specifies the TCP/IP host running the SAP gateway.
RFC Program ID	Specifies the Remote Function Call Program ID that the SAP Agent uses to register itself with the Remote Function Call Gateway.
Additional Connection Parameters	Passes additional string connection parameters to `RfcAccept` when OracleAS InterConnect acts as a Remote Function Call Server to an R/3 Client.

Optimize Remote Function Call Function Modules

In an unoptimized SAP environment, the Remote Function Call table retrieval is slow. To build the Remote Function Call namespace, R/3 downloads three separate tables: `area`, `groups`, and `functions`. Of the three tables, only the function tables contain any relationship about which group and area it belongs to. The areas and groups tables contain extra areas and groups that do not belong to the function. The groups table contains more than 5,000 entries, while the final usable groups are around 700 entries. The browser, working back from the function table, removes the unused groups and areas. Accessing a local R/3 system takes around 5 to 6 seconds for all the tables to be built. However, remotely accessing an R/3 system could take up 4 to 5 minutes.

To reduce the time spent downloading information at development time, functions are provided that allow selective retrieval of areas, groups, and functions. In support of lookup-on-demand for the groups and the areas, the SAP adapter has two custom function modules: one to retrieve the areas and the second to download groups for a specific area.

If you decide not to upload the browsing enhancement functions, the retrieval preloads the table. While preloading the Remote Function Call table is time consuming on a remote R/3 system, it provides advantages for local system since the whole table is prebuilt one time. For this reason, a registry setting key `RFCTablePreloadEnabled` is added in the FM Setting area. Selecting this function prebuilds the tables despite the existence of OracleAS InterConnect custom functions.

At runtime, the area and groups tables are not required and these two processes can be skipped to optimize the download process. On a local R/3 system, the preloading of the function table is fast. However, on a remote R/3 system, the preloading of the function table takes time for the table to completely download. To enhance the performance, the runtime use of preload or lookup-on-demand are based on either populating the preloaded table or using the setting `TablePreloadEnabled`.

Although long group names exist (starting from version 4.0) the native Remote Function Call function (that retrieves the list of functions from a long name group) does not differentiate between itself and a short name version that matches the first part of its name. If you query for functions belonging to a short name group, or a long name group which matches a short name group's name, the returned function list is a union of both groups' functions.

The SAP adapter can build the groups table with the short name groups or long names. The short name groups are preferred as they also call functions from the long name group. R/3 release 4.5 has the proper support for long name group and no longer returns the functions that belong to the groups that have the same short format name.

Note: Because 4.0x support for long name groups is not complete, using a group exposed on 4.0x and running it against a 4.5 machine might cause the function to be undefined since the long, or the short, name group on 4.5 contains only the functions that belongs to them and not the union of all the functions belong to both groups.

Enhance Remote Function Call Function Module Remote Browsing

To enhance your Remote Function Call Function Module Remote browsing, the SAP adapter includes the text files for Z_RFC_GET_AREAS (areas.asc) and Z_RFC_GET_GROUPS (ginstall_directorySAPALE_Filesroups3x.asc or groups4x.asc) in the install_directory/oai/9.0.4/sap/rfc_files directory.

Why a Function Group?

The function modules must be in the same function group (usually custom built) and the global data (shared by the entire function group) must contain the following statement:

```
tables:tfdir, taplt, tlibt
```

Creating a Function Group

Use the following steps to create the ZMAS function group.

1. Select **Tools > ABAP/4 Workbench > Function Builder** in the SAPGUI.
2. Select **Goto > Function groups > Create group**.
3. Enter the function group name in the Function field, for example, ZMAS.
4. Enter the group description in Short field and click **Save**.

   The Create object catalog entry dialog displays.
5. Complete the necessary fields for this dialog.
6. Verify where to store the object with your System Administrator. You can select one of the following:
   - Local object—A non transportable temporary object.
   - Development classes—Allow entities to take part in the R/3 transport mechanism. Changes to objects in these classes are recorded and can be transported to other systems.
7. Create two function modules: Z_RFC_GET_AREAS and Z_RFC_GET_GROUPS.

Creating the Z_RFC_GET_AREAS Function Module

Use the following steps to create the Z_RFC_GET_AREAS function module.

1. Select **Tools > ABAP/4 Workbench** in the SAPGUI.
2. Select **Function Builder**.

   The Function Library: Initial Screen dialog displays.
3. Enter the function module name, **Z_RFC_GET_AREAS**, in the Function module field and click **Create**.
4. Type in the following values for each of the object components selections:

```
Attributes
Import/Export Parameter Interface
Table Parameters/Exceptions Interface
Documentation
```

Attributes

The following describes attributes classifications:

- Beside Classification > Function Group, enter the following object components:
  - Enter **ZMAS**.
  - Under Classification > Application, enter **Z**.
  - Under Classification > ShortText, enter **Retrieve details about one intermediate document type**.
- Under Processing Type, check the following boxes:
  - **Remote Functional Call supported**
  - **Immediate Start**

Table Parameters/Exceptions Interface

Table 5–3 describes table parameters.

Table 5–3 Table Parameters

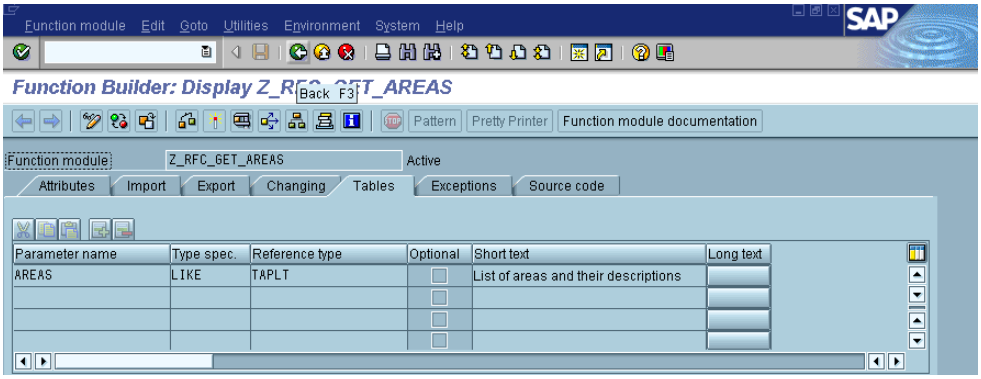
Table Parameters	Ref. Structure	Short Text
AREAS	TAPLT	Lot of areas and descriptions

Documentation

Figure 5–1 displays the Z_RFC_GET_GROUPS function module display. Enter the following values:

- Parameter Name—AREAS
- Short Text—List of areas and their descriptions
- Parameter—Table

Figure 5–1 Function Module Display:Z_RFC_GET_GROUPS



Uploading the Function Module Source Code

Use the following steps to upload the function module source code:

1. Select **Back** or **F3** to return to the previous dialog.
2. Click the **Source Code** tab.
3. Click **Change**.
4. Upload the Source Code provided with OracleAS InterConnect using **Utilities > More Utilities > UpLoad/DownLoad >UpLoad**.
5. Enter the path and file name, `install_directory/oai/9.0.4/sap/rfc_files/areas.asc`.
6. Click **Ctrl+F3** to activate the function module.

Creating the Z_RFC_GET_GROUPS Function Module

Use the following steps to create the Z_RFC_GET_GROUPS function module.

1. Select **Tools->ABAP/4 Workbench** in the SAPGUI.
2. Select **Function Builder**.

   The Function Library: Initial Screen dialog displays.
3. Enter the function module name, Z_RFC_GET_GROUPS, in the Function module field.
4. Click **Create**.
5. Type in the following values for each of the Object components selections:

```
Attributes
Import/Export Parameter Interface
Table Parameters/Exceptions Interface
Documentation
```

Attributes

The following describes attributes classifications:

- Beside Classification > Function Group, enter the following object components:
  * Enter **ZMAS**.
  * Under Classification > Application, enter **Z**.
  * Under Classification > ShortText, enter **Retrieve details about one intermediate document type**.
- Under Processing Type, check the following boxes:
  * **Remote Functional Call supported**
  * **Immediate Start**

Import/Export Parameter Interface

Figure 5–2 displays the Z_RFC_GET_GROUPS import and export parameters.

Figure 5–2 Function Module Display:Import/Export Parameters Z_RFC_GET_GROUPS

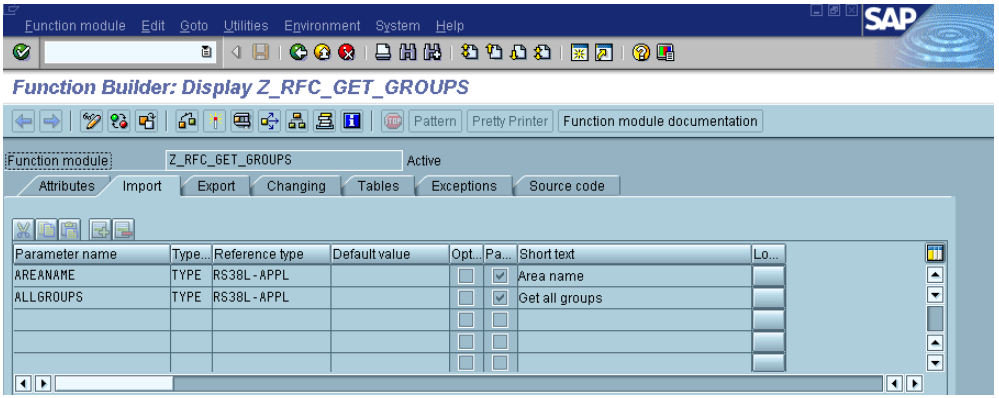


Table 5–4 Import parameter definitions

Import parameter	Reference field	Proposal	Optional	Short Text
AREANAME	RS38L-APPL	N/A	N/A	Get all Groups
ALLGROUPS	RS38L-APPL	SPACE	X	Area Name

Table Parameters/Exceptions Interface

Figure 5–3 displays the Remote Function Call parameters group.

Figure 5–3 Remote Function Call Table Parameters Group

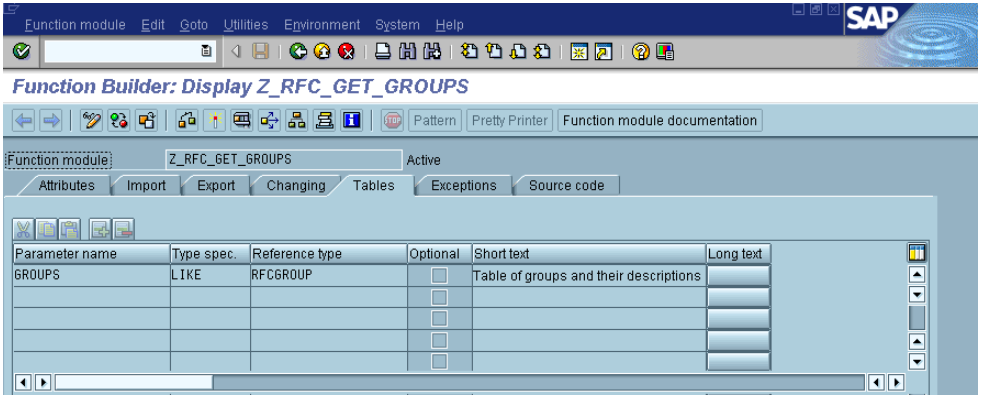
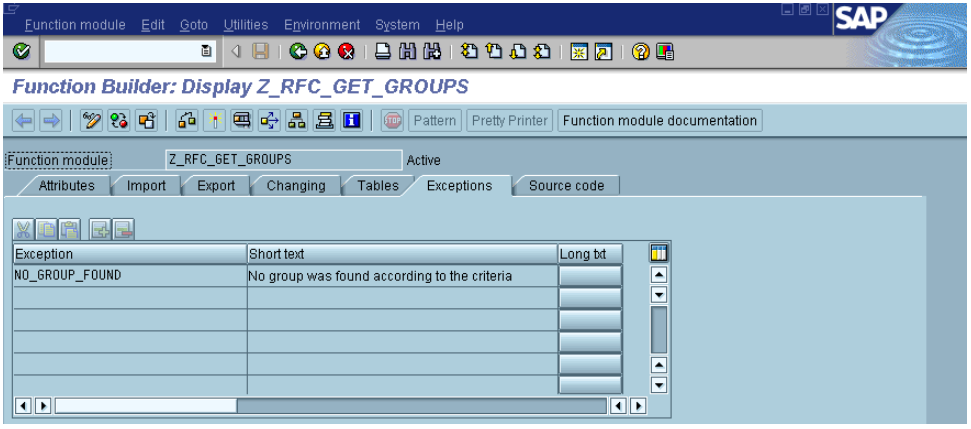


Table 5–5 Table Parameter Definitions

Table Parameters	Ref. Structure	Short Text
GROUPS	RFCGROUP	Table of Groups and their descriptions
Exception	N/A	N/A
NO_GROUP_FOUND	N/A	No Group was Found according to the Criteria

Figure 5–4 Remote Function Call Exception Group



Uploading the Function Module Source Code

Use the following steps to upload the function module source code:

1. Click the **Source Code** tab.
2. Click **Change**.
3. Upload the Source Code provided with OracleAS InterConnect using the SAP command **Utilities > More Utilities > UpLoad/DownLoad >UpLoad**.
4. Enter the path and file name:

   R/3 Version 3.x

   ```
   install_directory/oai/9.0.4/sap/rfc_files/groups3x.asc
   ```

   or

   R/3 Version 4.x

   ```
   install_directory/oai/9.0.4/sap/rfc_files/groups4x.asc
   ```

5. Click **Ctrl+F3** to activate the function module.

Set Global Data

Use the following steps to set global data:

1. Select **Back** or **F3** to return to the previous screen.
2. Select **Global Data**.
3. Click **Change**.
4. Locate the line:

   ```
   function-pool zmas.MESSAGE-ID .,
   ```

   and insert the following:

   ```
   tables:tfdir, taplt, tlibt.
   ```

5. Verify you have access to this global data by performing a simple check:
   a. Click **Tools->ABAP/4 Workbench->Function Builder**.
   b. Set the function module to Z_RFC_GET_AREAS.
   c. Check **Global data**.
   d. Click **Display**.
   e. Verify that the following displays:

   ```
   function-pool zmas.MESSAGE-ID ..
   tables:tfdir, taplt, tlibt.
   ```

Clean Your R/3 System

It is recommended that you remove all tables you insert in an R/3 system so that in the future you can create tables for this sample. For example, to remove ZORDERS and ZCOMMISS tables, you must first erase the function groups and the function modules.

Erasing Function Groups and Function Modules

To erase function groups and modules:

1. Open the SAPGUI initial dialog.
2. Select **Tools->ABAP Workbench** to erase the Z_ACCNT_DEPT function group.
3. Click **Function Builder** to display the Function Builder: Initial Screen dialog.
4. Select **Goto->Function groups->Delete group**.

   The Change Function Group dialog displays.
5. Enter Z_ACCNT_DEPT in the Function group field and click the **check mark**.

   The Delete Function: Group: Delete Function dialog displays. It shows the two function modules belonging to the group.
6. Click **Delete** to delete the Z_COMMISS_ADD and the Z_COMMISS_UPDATE function modules.
7. Repeat steps 3 and 4 to erase any other function group and any other function modules.

Erasing a Table

After erasing the function groups and modules, erase a table using the SAPGUI with the following steps:

1. Press **F3** to return to the ABAP Workbench dialog.
2. Click **Dictionary**.
3. Type `ZORDERS` in the Object name field.
4. Click **trash can**. A confirmation dialog displays.
5. Repeat steps 3 and 4 to erase the `ZCOMMISS` table.

> **Note:** Remember to type `ZCOMMISS` in the object name text box.

6

Runtime

This chapter describes how to use the Configuration Editor to configure the SAP adapter. The Configuration Editor is only used at runtime. The following topics are discussed:

- Configuration Editor
- Creating SAP Host Definitions in Global Settings
- Default Login to R/3
- Common Settings
- Exiting Configuration Editor

See Also: Chapter 4, "Application Link Enabling"

Note: Profiles and deployment are sensitive to the Master Key setting. If using a shared machine, before accessing the Configuration Editor, ensure the Master Key is set to either that of User1 or create a new Master Key for your profiles. Refer to the *Oracle Application Server InterConnect Configuration Editor User's Guide* for more information on the Master Key.

Before editing any settings in the Configuration Editor, check that the profile is named iStudio. If iStudio has been run from the runtime machine, and logged into R/3 using OracleAS InterConnect, profile iStudio is automatically created. If you have not run iStudio on the runtime machine, you must create a profile called iStudio on the Configuration editor and set that profile as default.

Configuration Editor

> **Note:** Throughout this section, reference to launching the Configuration Editor is expressed as: Type configeditor and press Enter. However, if you are using a UNIX machine, you must type **configeditor.sh** and press Enter.

Using the R/3 Configuration Editor, you can customize the settings to specify how your development machine and components interact with your R/3 system.

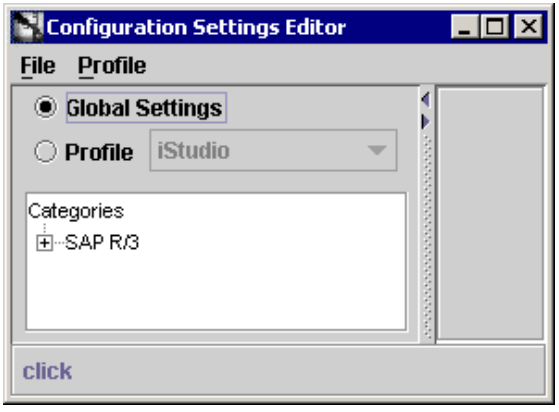
> **Note:** Before using any BAPI interfaces, you must configure the Remote Function Call. BAPI and Remote Function Call share their configuration information.

You can make changes to the login, ALE, Inbound, and Outbound to R/3 settings in the Configuration Editor. To access the Configuration Editor, from a command line:

1. Change directories to the Configuration Editor installation directory.
2. Type configeditor and press **Enter**.

   The Configuration Editor displays.

Figure 6–1 Configuration Settings Editor



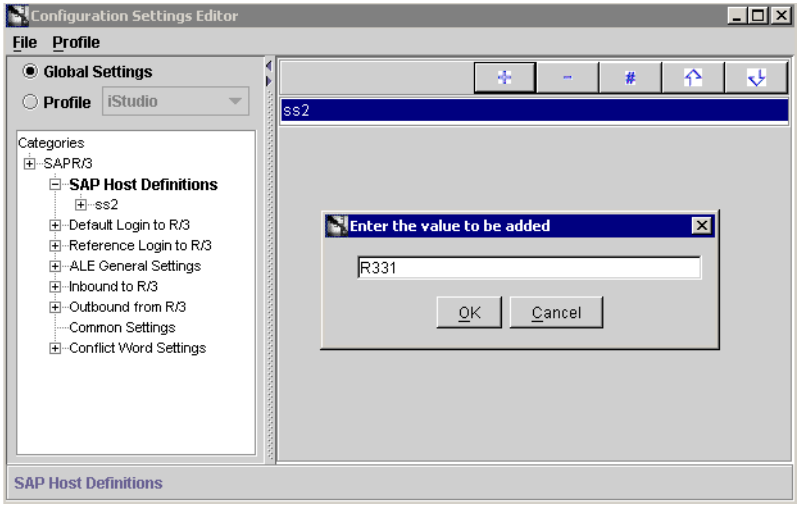
Creating SAP Host Definitions in Global Settings

Before specifying the settings for the adapter, you must create SAP host definitions under Global Settings. When the Configuration Editor is launched, by default, the radio button for Global Settings is selected. To create an SAP host:

1. Double-click **SAP R/3**.
2. Select **SAP Host Definitions**.

   Control functions display in the right panel.

Figure 6–2 Configuration Editor Host Definition



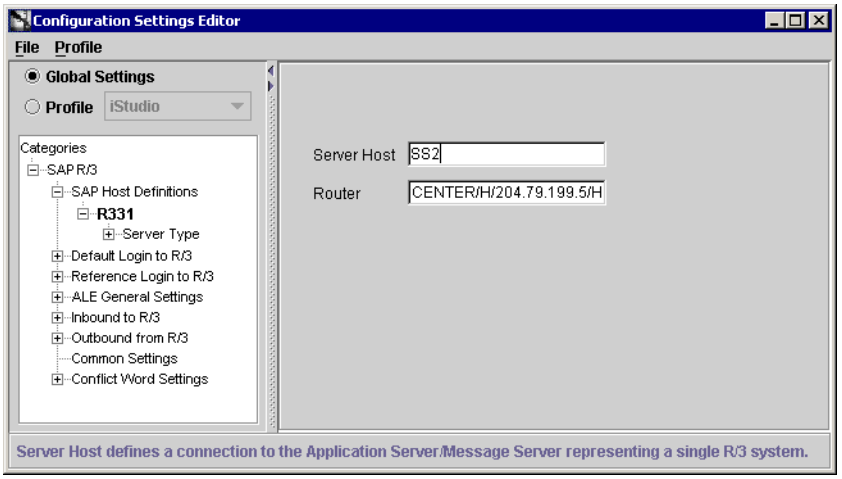
3. Click **PLUS (+)** on the right panel to add a host.
4. Type the name of the new Host definition.

   This can be a descriptive name recognizable as being set for a specific system, for example, `R331 is` for an R/3 Version 3.1 system.
5. Click **OK**.

   New host name displays in the right panel.
6. Expand **SAP Host Definitions** in the left panel.

7. Click the **Server Host name**.

Figure 6–3 Configuration Settings Editor Server Type



8. Enter the **Server Host identification** in the Server Host field.

   This is the actual link to the server.

9. Specify a **Router**, if required.

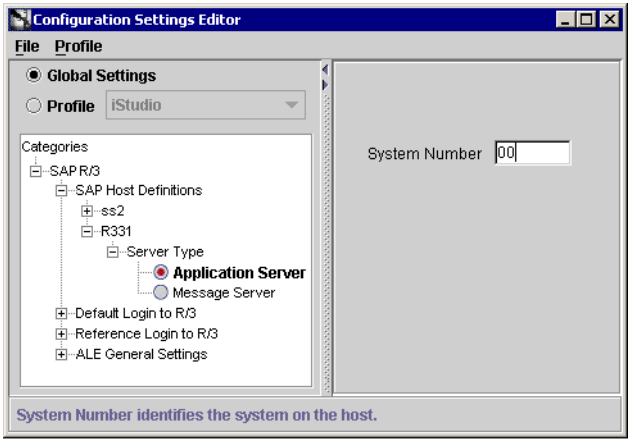
   This is a Destination router to connect to the application server or Message Server, for example `/H/UNICENTER/H/204.79.199.5/H`.

10. Expand **Server Type** and select your server type.

11. Specify the **system number** if the server type is Application Server in the System Number field.

The system number further identifies the Host to a specified Service level. The service is the TCP/IP service name (a port number through `c:\winnt\system32\drivers\etc\services`). For example, using `ss1:00` as the connecting host, the `00` is what SAP calls the system number.

Figure 6–4 Configuration Settings Editor Application Server

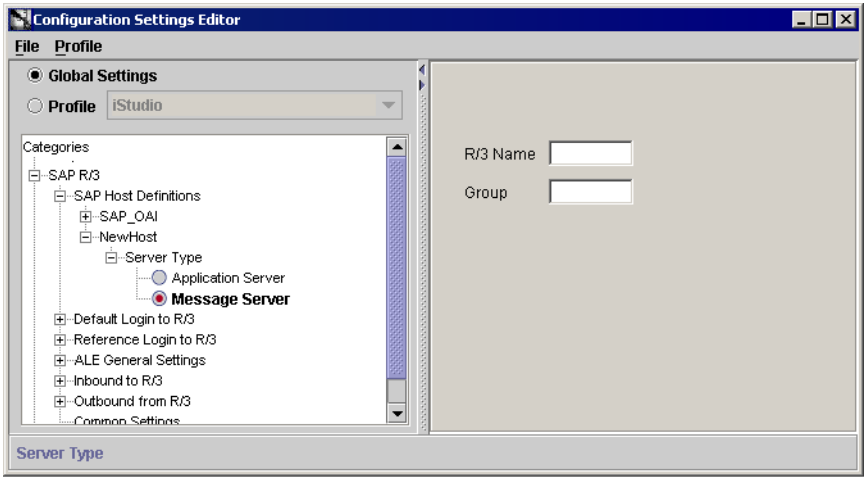


If the server type is Message Server, specify the following:

- R/3 Name—The system ID that identifies the SAP System, for example D15.

- Group—The Message Server Group if your message servers belong to a group, for example, PUBLIC.

Figure 6–5 Configuration Settings Editor Message Server



Default Login to R/3

The Default Login to R/3 group allows you to program your development application to automatically connect to R/3 servers. The Default Login to R/3 authenticates your runtime credentials. From the Configuration Editor main menu:

1. Click **Profile** and select **iStudio**.

Note: Under some circumstances you may wish to run your adapter under a profile other than iStudio. This may be needed for example, in case you want to run two instances of the SAP adapter on the same machine. You may want to have two instances of the same type of adapter if these instances need to connect to different backend system installations. To accomplish this you need to create a new profile using the configuration editor and fill in the settings for this new profile. The name of the new profile should be the same as the name of the application. For example if your application is called APP2, create a profile called APP2. Now APP2 will use the settings in the profile called APP2 whenever it runs.

---

2. Expand the **SAP R/3** tree.
3. Expand **Default Login to R/3**.
4. Expand **Use Global Settings**.
5. Expand **Enable Login Settings** and check that box. The right panel displays the default login fields to specify.

Figure 6–6 Configuration Editor Enable Login

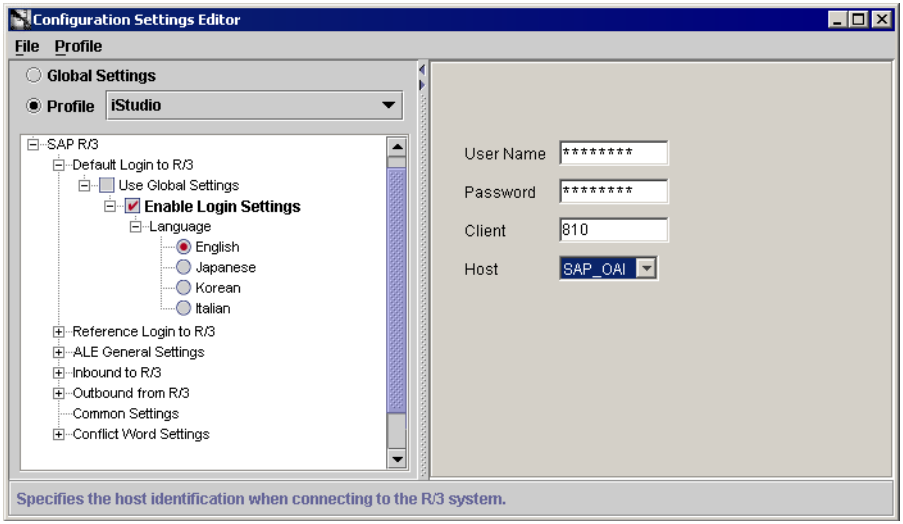


Table 6–1 Enable Login Settings Panel Configuration Editor

Enable Login Settings Panel Field	Field Description
User Name	Your user ID for the R/3 system.
Password	Your user password for the R/3 system.
Client	Your client number ID for the R/3 system.

Table 6–1 Enable Login Settings Panel Configuration Editor

Enable Login Settings Panel Field	Field Description
Host	Specifies the Host ID when connecting to the R/3 system. Select a Host ID from the dropdown list. All the Host IDs created for the `SAP Host Definition` setting in Global Settings are shown in this list. For Inbound to R/3: the value of Host is that of the Application Server or the Message Server to be contacted.
Language	Required by R/3. By default, the Language parameter retrieves the language information from your operating system.

Reference Login to R/3

The Reference Login to R/3 authenticates your runtime credentials. All the parameters for this group are identical to those logging into R/3 for a regular session.

When using OracleAS InterConnect with multiple R/3 systems, it is possible to have one of the systems act as a reference system while calling into other systems. This means that data elements, function signatures, and BAPI parameters are taken from the reference system rather than from the one that you are calling. This is useful in cases where different systems are running different versions of R/3. For example, suppose you have the following three systems:

```
billing:00 running R/3 version 3.1H
billing:01 running R/3 version 3.1I
billing:02 running R/3 version 4.0C
```

Previously, you would have needed one set of clients to call the 3.1 systems, and another client to call the 4.6 system, because of new parameters added to the 4.6 signature of certain function modules. You can use the reference login feature to indicate that the repository information should always be read from only one of the machines. In this particular example, you might select `billing:00` since it is the oldest machine. If you set `billing:00` as your reference machine, calls to `billing:01` or `billing:02` are made according to the information in `billing:00's` repository. As long as the changes have been made in a backward-compatible manner (with optional parameters, for instance), the same client is usable with all three machines. Reference a local server but make client calls against a remote server on a slow connection.

Inbound to R/3

The Inbound group contains configurable parameters pertaining to the R/3 system when it behaves as a server.

Click to expand **Inbound to R/3**.

The Inbound to R/3 selection is highlighted in the left pane and Additional Connection Parameters field displays in the right panel.

Figure 6–7 Inbound to R/3 Configuration Settings Editor

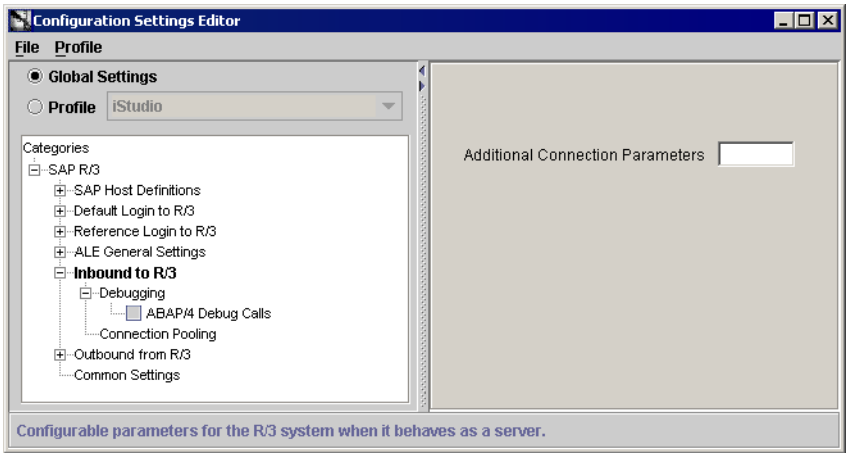


Table 6–2 Inbound to R/3 Configuration Settings Editor

Inbound to R/3 Settings Values	Value Descriptions
Debugging	This feature is useful for debugging or diagnostic purposes. However, it is of limited use in a production environment, as the message does not display on the client machine. ABAP/4 Debug Calls are also known as Remote Function Call Debug Calls. Use this selection when you are debugging Function Modules. Selecting ABAP/4 Debug Calls automatically sets the ABAP_DEBUG connection parameter allowing Function Module calls to go through the SAPGUI debugger.
Connection Pooling - Max Concurrent Connections	The default value is `50`. This setting controls the maximum concurrent connections to the SAP R/3 system.

Table 6–2 Inbound to R/3 Configuration Settings Editor

Inbound to R/3 Settings Values	Value Descriptions
Additional Connection Parameters	Passes additional string connection parameters to `RfcOpenEx` when Control Broker acts as an Remote Function Call Client connecting to the R/3 Server.

Outbound from R/3

The Outbound from R/3 group contains parameters pertaining to the R/3 system when R/3 is calling other systems through SAP adapter.

Click to expand **Outbound from R/3**.

Outbound from R/3 is highlighted in the left panel. The Host, RFC Program ID, and Additional Connection Parameters fields display in the right panel.

See Also: Table 6–3

Figure 6–8 Outbound from R/3 Configuration Settings Editor

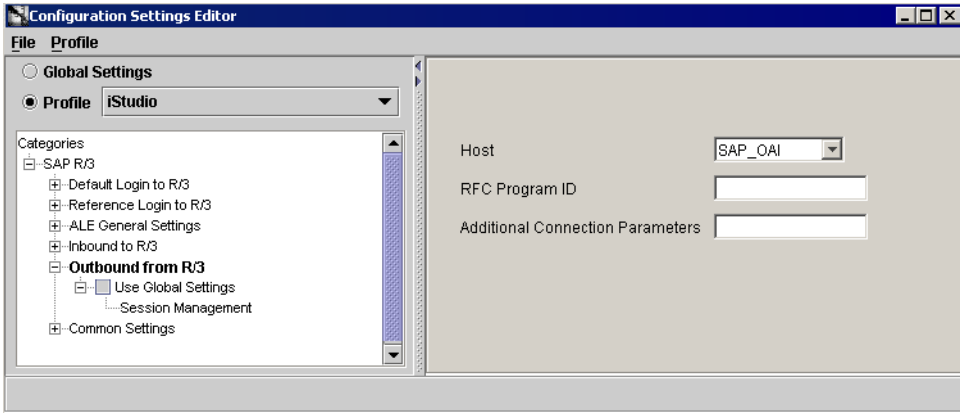


Table 6–3 Outbound from R/3 Configuration Settings Editor

Outbound from R/3 Settings Fields Panel	Field Descriptions
Host	The host is used in the login process to an R/3 system. You select a Host ID from the drop down list. All the Host IDs created for `SAP Host Definition` setting in **Global Settings** are shown in this list. The value of Host specifies the TCP/IP host running the Remote Function Call Gateway containing the registered Agent, for example, usually it is the machine where the SAP System is installed.
Remote Function Call Program ID	Specifies the Remote Function Call Program ID that the OracleAS InterConnect acting as an Remote Function Call server uses to register itself with the Remote Function Call Gateway. A unique identification assigned to an SAP Server to partition the application. Each Destination Host on the SAPGUI has a corresponding Program ID assigned by the System Administrator. This name is case-sensitive. For example, the program ID is a named port into R/3 corresponding to an Remote Function Call destination. When writing an R/3 application, the destination must be specified in order to send requests.
Additional Connection Parameters	Passes additional string connection parameters to `RfcAccept` when the SAP adapter acts as an Remote Function Call Server to an R/3 Client.

Common Settings

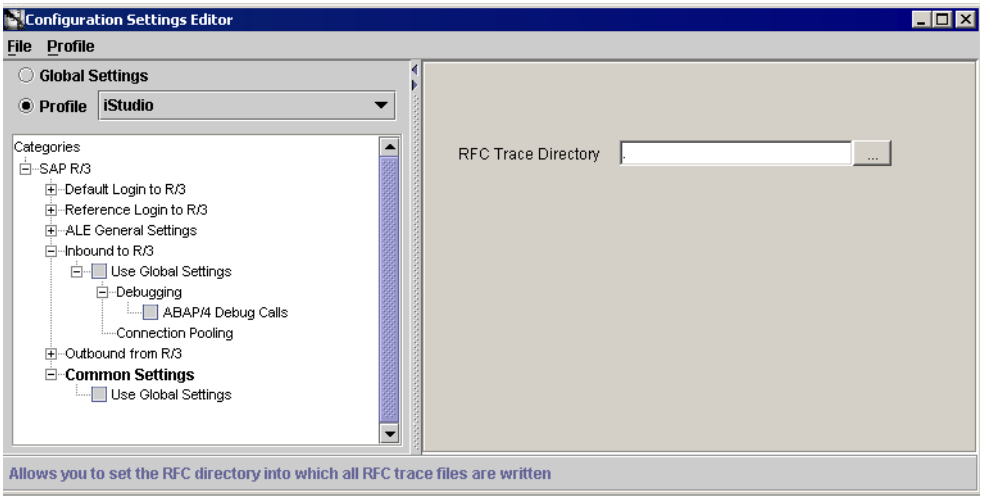
The Common Remote Function Call Settings group allows you to set the Remote Function Call directory into which all Remote Function Call trace files are written. For example, all `dev_rfc.trc`, all `rfc .trc` files.

The SAP adapter writes trace messages in trace files whose name are of the form `rfc?????_????.trc`, where each ? is a digit between 0 and 9. Each Remote Function Call Connection creates a different trace file.

1. Expand to **Common Settings**.

Common Settings is highlighted in the left panel and the RFC Trace Directory field displays in the right panel.

Figure 6–9 Configuration Editor Common Settings



The RFC Trace File Directory specifies the full path of the Remote Function Call trace file. In the RFC Trace Directory field, enter a temporary path to hold your temporary files. You also use the browse button to activate a directory selection dialog to select a temporary directory.

By default, trace files are written into the current working directory.

Exiting Configuration Editor

When the correct parameters are entered, the Configuration Editor can be exited. When the Configuration Editor is exited, the parameters entered are saved. You can also select **File -> Save settings** to save your changes before exiting the program.

To exit the Configuration Editor:

1. Click the **X** in the upper right corner.

   The following prompt displays:

   ```
   Some of the settings in have been changed in this session. Would you Like to
   save the changes?
   ```

2. Click **YES**.

   The following prompt displays:

   ```
   The settings you've changed will take affect after restart.
   ```

3. Click **OK**.

   The program terminates and closes.

Index

H

I

L

M

O

P

R

S

T