Oracle9*i*AS InterConnect Adapter for SAP R/3

Installation and User's Guide

Release 2 (9.0.2)

February 2002

Part No. A95438-01

Oracle9*i*AS InterConnect Adapter for SAP R/3 Installation and User’s Guide, Release 2 (9.0.2)

Part No. A95438-01

Contents

Send Us Your Comments vii

Preface ix

1 Introduction

What is SAP? 1-2
- Required Software 1-2
- Supported Platforms 1-2

2 Installation and Configuration

Installating the SAP Adapter 2-2
- Preinstallation Tasks 2-2
- Installation Tasks 2-2

SAP Adapter Configuration 2-4
- Using the Application Parameter 2-5
- adapter.ini Initialization Paramter File 2-6
  - Hub.ini 2-6
  - Agent Connection Paramters 2-7
  - SAP Adapter-Specific Parameters 2-11

Starting the SAP Adapter 2-12

3 Supported SAP Interfaces

Exception Fields 3-2

Inbound to SAP 3-2

Creating an Application Link Enabling Implemented Procedure.. 3-3
Importing Attributes from SAP 3-8
Creating an Application Link Enabling Subscribed Event.. 3-12
Creating a Remote Function Call Implemented Procedure... 3-17
Creating a Remote Function Call Subscribed Event... 3-22
Outbound From SAP.. 3-27
Creating an Outbound Application Link Enabling Invoked Procedure 3-27
Creating an Application Link Enabling Published Event.. 3-32
Creating a Remote Function Call Invoked Procedure... 3-37
Creating a Remote Function Call Published Event.. 3-42

4 Application Link Enabling

Frequently Used Application Link Enabling Transactions 4-2
Application Line Enabling Terminology.. 4-3
Logical System... 4-3
Intermediate Documents Type 4-4
Message Type 4-5
Oracle9*i*AS InterConnect Application Acting as a Client... 4-6
Oracle9*i*AS InterConnect Application Acting as a Server 4-6
Application Link Enabling Subdirectories—Queue and Cache... 4-7
Queuing Inbound Intermediate Documents 4-8
Application Link Enabling General Settings 4-9
Inbound Intermediate Documents 4-11
Outbound Intermediate Documents.. 4-12
R/3 Application Link Enabling Configuration.. 4-12
Application Link Enabling—Exploring Intermediate Document Types 4-22
Manually Downloading an IDOC.. 4-23
Enhance Application Link Enabling Remote Browsing 4-25
How to Install the Remote Browsing Function Modules.. 4-26
Why a Function Group? 4-26
Create Four Structures 4-26
Create a Function Group 4-31
Create Two Function Modules 4-32

5 Remote Function Call

Introduction to Remote Function Call 5-2
SAP Adapter Interaction with R/3 5-2
SAP Adapter Application Acting as a Client 5-2
SAP Adapter Application Acting as a Server 5-2
Remote Function Call Configuration 5-3
Calling From SAP adapter to R/3 5-3
Default Login to R/3 5-3
Calling from R/3 to the SAP Adapter 5-4
Optimize Remote Function Call Function Modules 5-4
Enhance Remote Function Call Function Module Remote Browsing 5-6
Why a Function Group? 5-6
Creating a Function Group 5-6
Creating the Z_RFC_GET_AREAS Function Module 5-7
Attributes 5-7
Table Parameters/Exceptions Interface 5-7
Documentation 5-8
Uploading the Function Module Source Code 5-8
Creating the Z_RFC_GET_GROUPS Function Module 5-9
Attributes 5-9
Import/Export Parameter Interface 5-10
Table Parameters/Exceptions Interface 5-11
Uploading the Function Module Source Code 5-12
Set Global Data 5-13
Clean Your R/3 System 5-14
Erasing Function Groups and Function Modules 5-14
Erasing a Table 5-15

6 Runtime

Configuration Editor 6-2
Creating SAP Host Definitions in Global Settings 6-3
Default Login to R/3 6-6
Reference Login to R/3 6-8
Inbound to R/3 6-9
Outbound from R/3 6-10

Common Settings 6-11
Exiting Configuration Editor 6-12

Index

Send Us Your Comments

Oracle9*i*AS InterConnect Adapter for SAP R/3 Installation and User's Guide, Release 2 (9.0.2)

Part No. A95438-01

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: iasdocs_us@oracle.com
- FAX: 650-506-7407 Attn: Oracle9*i* Application Server Documentation Manager
- Postal service:
  Oracle Corporation
  Oracle9*i* Application Server Documentation
  500 Oracle Parkway, M/S 2op3
  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
- Documentation Accessibility
- Intended Audience
- Organization
- Related Documentation
- Conventions

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 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:

- *Oracle9iAS InterConnect User Guide* in the Oracle9*i* Application Server Documentation Library
- *Oracle9i Application Server Installation Guide*
- *Oracle9iAS InterConnect Adapter Configuration Editor User's Guide*

In North America, printed documentation is available for sale in the Oracle Store at

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

Customers in Europe, the Middle East, and Africa (EMEA) can purchase documentation from

`http://www.oraclebookshop.com/`

Other customers can contact their Oracle representative to purchase printed documentation.

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/admin/account/membership.html`

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/docs/index.htm`

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	`CREATE TABLE ... AS` *`subquery`*`;`
	■ That you can repeat a portion of the code	`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 SAP adapter. This chapter discusses the following topics:

- What is SAP?

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: *Oracle9i Application Server Installation Guide*, Appendix C for hardware requirements

Supported Platforms

The following platforms support the SAP adapter:

- Windows NT 5.0
- Solaris 2.6
- Solaris 7 (2.7)
- Solaris 8 (2.8)
- HP-UX B.11.00

2 Installation and Configuration

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

- Installating the SAP Adapter
- SAP Adapter Configuration
- Starting the SAP Adapter

Installing the SAP Adapter

This section contains these topics:

- Preinstallation Tasks
- Installation Tasks

Preinstallation Tasks

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

- An existing Oracle9*i* Application Server Oracle home
- An existing Oracle9*i* Application Server Infrastructure Database Oracle home
- An existing Oracle9*i*AS InterConnect Oracle home
- A new Oracle home (the installer creates this for you)

Consult the *Oracle9i Application Server Installation Guide* before proceeding with SAP adapter installation. This guide includes information on:

- CD-ROM mounting
- Oracle Universal Installer startup
- Oracle9*i*AS InterConnect software, hardware, and system requirements
- Oracle9*i*AS InterConnect installation

Note: Oracle9*i*AS InterConnect Hub is installable through the Oracle9*i*AS InterConnect Hub installation type. You must install the Oracle9*i*AS InterConnect Hub before proceeding with the SAP adapter installation.

Installation Tasks

To install the SAP adapter:

1. Click **Next** on the Welcome page.

   The File Locations page displays.

2. Enter the following information in the Destination fields:

   - Name—The Oracle home name.
   - Path—The full path to the Oracle home in which to install the SAP adapter.

Note: Do not change the path specified in the Source field. This is the location on the CD-ROM from which to install the SAP adapter.

3. Click **Next**.

   The Installation Types page displays.

4. Select Oracle9*i*AS InterConnect Adapters and click **Next**.

   The Available Product Components page displays.

5. Select Oracle9*i*AS InterConnect SAP Adapter and click **Next**.

6. If the SAP adapter is not being installed on the same computer as Oracle9*i*AS InterConnect Hub and another adapter is not installed in the current Oracle home, the Oracle9*i*AS InterConnect Hub Database page displays. Enter the following information about the Oracle9*i*AS InterConnect Hub to use:

   - Host Name—The hostname of the computer on which Oracle9*i*AS InterConnect Hub is installed
   - Port Number—The port number of the computer
   - Database SID—The system identifier (SID) of the Oracle9*i*AS InterConnect Oracle9*i*AS Metadata Repository
   - Password—The password for the Oracle9*i*AS Metadata Repository schema

   The Oracle9*i*AS Metadata Repository stores metadata used by Oracle9*i*AS InterConnect to coordinate communication between components.

7. Click **Next**. 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`.

8. Click **Next**. Complete the fields for any other components selected for installation, such as other adapters. When finished, the Summary page displays.

9. 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\9.0.2\adapters\`*`Application`*
UNIX	`$ORACLE_HOME/oai/9.0.2/adapters/`*`Application`*

Application is the value you specified in Step 8 on page 2-3.

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/9.0.2/adapters/`*`Application`*
Windows	`%ORACLE_HOME%\oai\9.0.2\adapters\`*`Application`*

Table 2–2 Executable Files

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

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 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 *i*Studio under the Applications folder.

If you are using pre-packaged metadata, after importing the pre-packaged metadata into the repository, start up *i*Studio to find the corresponding application (under the Applications folder in *i*Studio) 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 Paramter File

This section contains these topics:

- Hub.ini
- Agent Connection Paramters
- SAP Adapter-Specific Parameters

Hub.ini

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.

Parameter	Description	Example
`hub_username`	The name of the hub database schema (or username). Possible values are valid hub database username. There is no default value.	`hub_username=myhub`
`hub_password`	The password for the hub database user. Possible values are the valid password for the hub database user. There is no default value.	`hub_password=manager`
`hub_host`	The name of the machine hosting the hub database. Possible values are the valid machine name. There is no default value.	`hub_host=mpjoshipc`
`hub_instance`	The valid SID of the hub database. There is no default value.	`hub_instance=orcl`
`hub_port`	The TNS listener port number for the HUB database instance. There is no default value.	`hub_port=1521`
`repository_name`	The valid name of the repository this adapter talks to. There is no default value.	`repository_name=myrepo`

Agent Connection Paramters

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

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`

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 Oracle9*i*AS InterConnect-specific DTD for all messages sent to the hub as other Oracle9*i*AS 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 Oracle9*i*AS InterConnect DTD. Only set to `true` if a Oracle9*i*AS 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`

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

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 Oracle9*i*AS 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 Oracle9*i*AS 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. 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. The default value is 60000.	agent_persistence_ retry_interval=60000
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\ora902\bin
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\ ora902\oai\902\lib\ oai.jar;%JREHOME%\lib\ i18n.jar;D:\oracle\ora 902\jdbc\classes12.zip
service_class	The entry class for the Windows NT 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

Parameter	Description	Example
`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`
`service_jdk_version`	Windows only. The JDK version the adapter Java VM should use. The default value is `1.3.1`.	`service_jdk_version=1.3.1`
`service_jdk_dll`	Windows only. The dll the adapter Java VM should use. The default value is `jvm.dll`.	`service_jdk_dll=jvm.dll`

SAP 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.oai.TxAgent`

Starting the SAP Adapter

Start the SAP adapter using the `start` script in the directory named after the SAP adapter on Windows NT, UNIX, or HP.

On Windows NT or Windows 2000, start it from the Service 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 *OracleHome9iASInterConnectAdapter-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.

3 Supported SAP Interfaces

This chapter provides an overview about SAP-specific information to assist you in working with the SAP 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.

Consider an example where you have a setup with SAP on one side, an Oracle9*i*AS InterConnect hub in the middle and a Web front end the other side. Suppose the Web front-end tries to add a record to the SAP side, however the a record with the same primary key already exists in SAP. In this case, you have a non-retryable error. 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

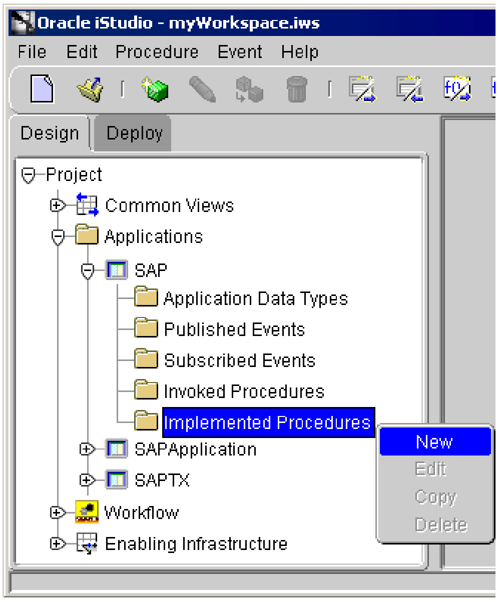
Sending messages inbound means the SAP adapter is the client and SAP is the server. To send messages to the SAP adapter, ensure that the host definition and login information is set for connecting to the SAP system using the Configuration Editor.

See Also: *Oracle9iAS InterConnect Configuration Editor User's Guide*

Creating an Application Link Enabling Implemented Procedure

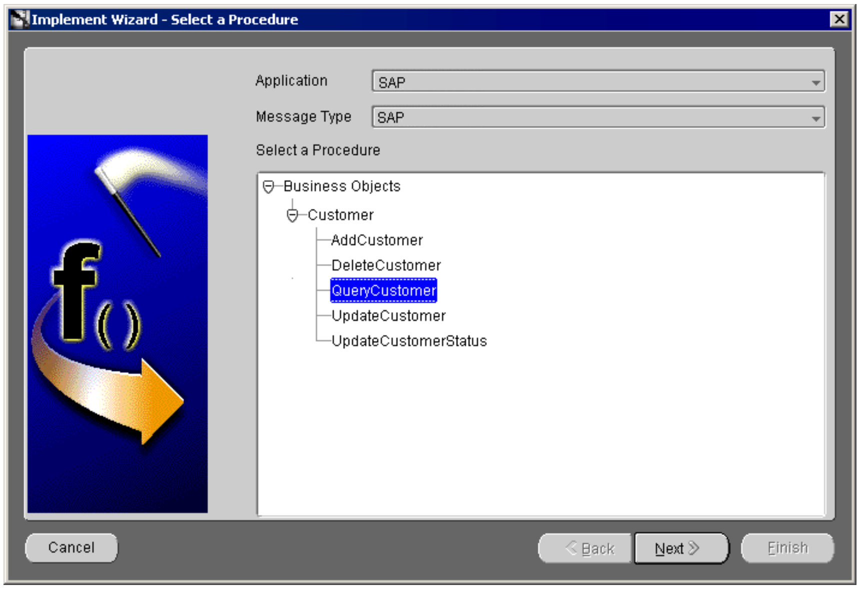
1. Start iStudio and open your project.
2. Expand the Applications folder.
3. Expand your Application.
4. Right-click **Implemented Procedures** and select **New**.

Figure 3–1 Creating an Implemented Procedure



The Implement Wizard—Select a Procedure dialog displays.

Figure 3–2 Implement Wizard —Selecting a Procedure



Implement Wizard - Select a Procedure

Application: SAP

Message Type: SAP

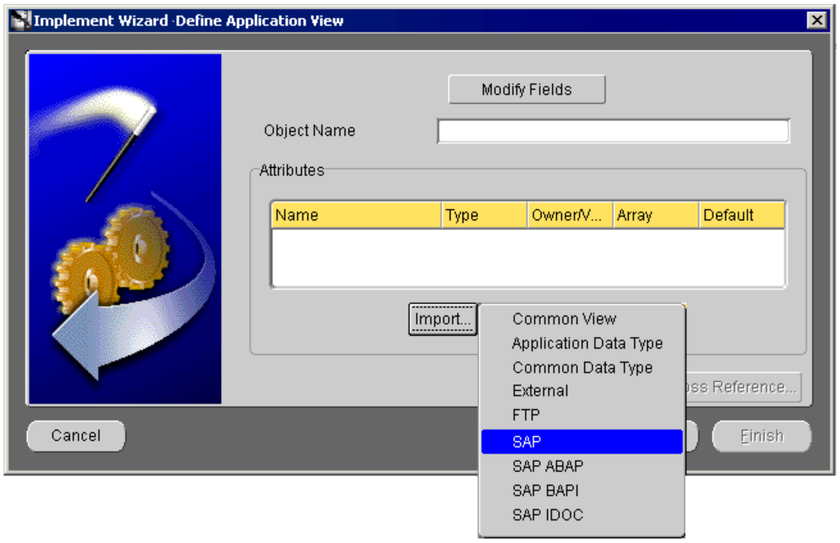
Select a Procedure

```
Business Objects
  Customer
    AddCustomer
    DeleteCustomer
    QueryCustomer
    UpdateCustomer
    UpdateCustomerStatus
```

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.

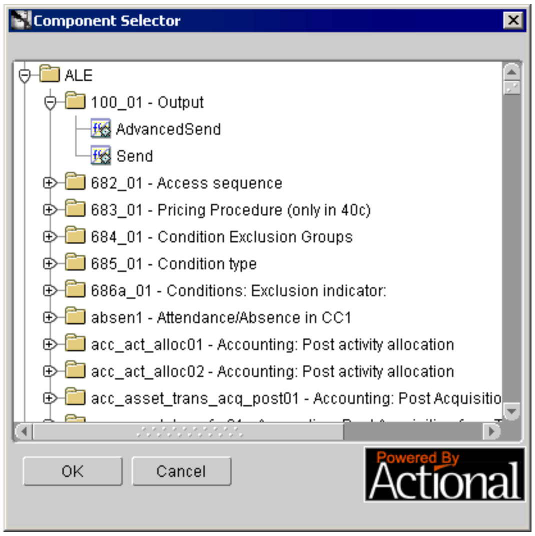
   If this is the initial login for this machine, enter the correct information.

   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

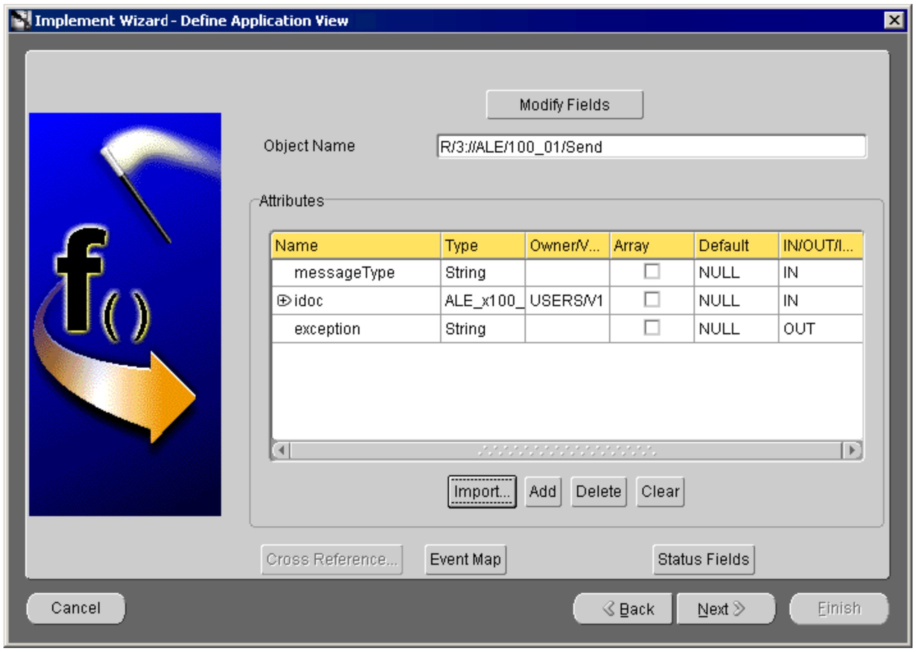

Component Selector
ALE
100_01 - Output
AdvancedSend
Send
682_01 - Access sequence
683_01 - Pricing Procedure (only in 40c)
684_01 - Condition Exclusion Groups
685_01 - Condition type
686a_01 - Conditions: Exclusion indicator:
absen1 - Attendance/Absence in CC1
acc_act_alloc01 - Accounting: Post activity allocation
acc_act_alloc02 - Accounting: Post activity allocation
acc_asset_trans_acq_post01 - Accounting: Post Acquisitio
OK
Cancel
Powered By Actional

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



Object Name: R/3://ALE/100_01/Send

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
messageType	String		☐	NULL	IN
⊕ idoc	ALE_x100_	USERS/V1	☐	NULL	IN
exception	String		☐	NULL	OUT

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.

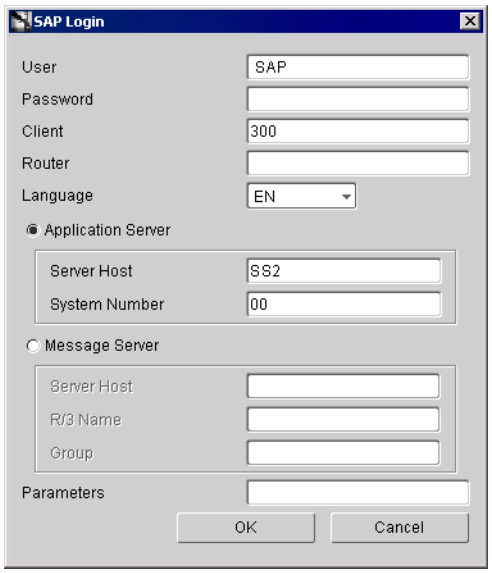
To import attributes from SAP:

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



User	SAP
Password	
Client	300
Router	
Language	EN

● Application Server

Server Host	SS2
System Number	00

○ Message Server

Server Host	
R/3 Name	
Group	

Parameters	

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 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.
R3NAME	Name of the SAP R/3 system, if using Remote Function Call session management.

Table 3–1 Identification Keys

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.

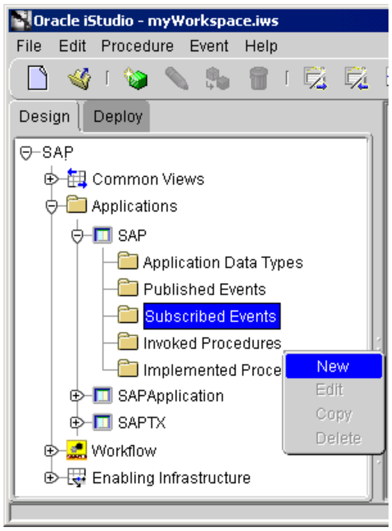
3. Click **OK** to accept your selections and continue to the Component Selector.

Creating an Application Link Enabling Subscribed Event

To create an Application Link Enabling subscribed event using iStudio:

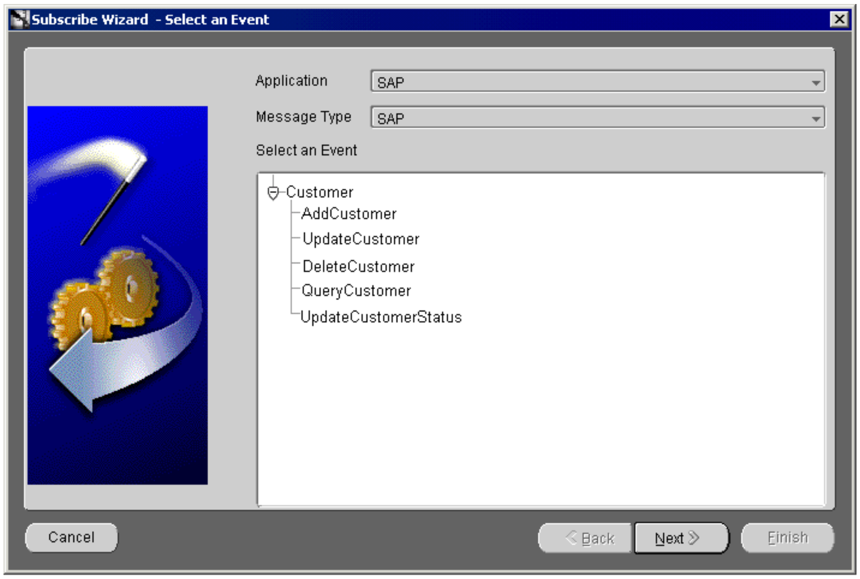
1. Start iStudio
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Subscribed Events** and select **New**.

Figure 3–7 Creating a Subscribed Event



The Subscribe Wizard—Select an Event dialog displays.

Figure 3–8 Subscribe Wizard - Selecting an Event



Application: SAP

Message Type: SAP

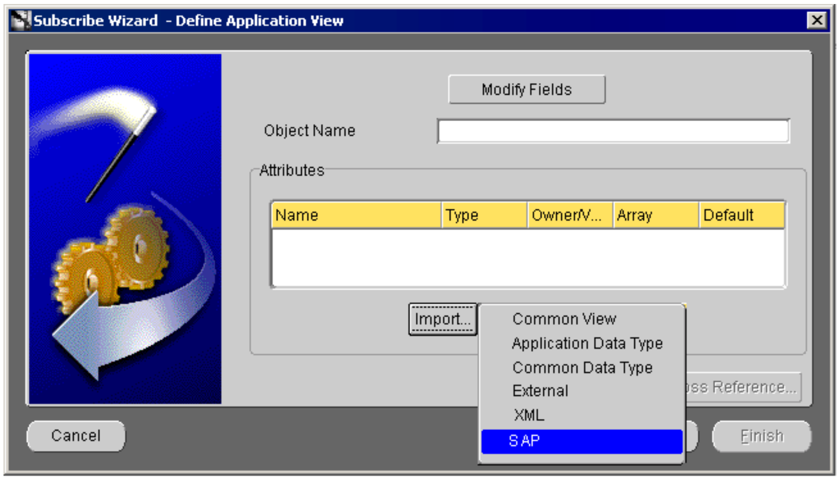
Select an Event

- Customer
  - AddCustomer
  - UpdateCustomer
  - DeleteCustomer
  - QueryCustomer
  - UpdateCustomerStatus

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.

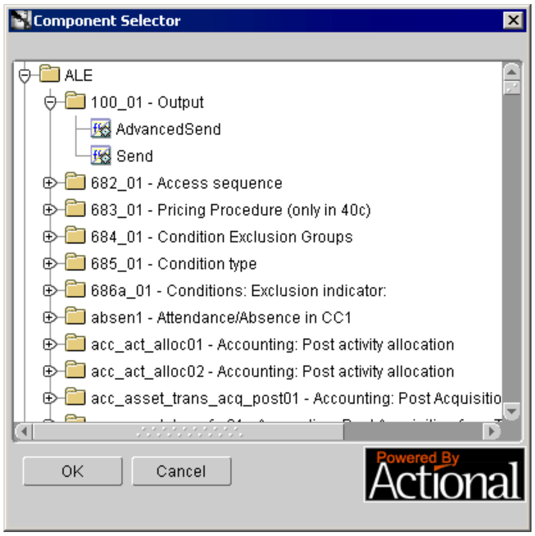
If this is the initial login for this machine, enter the correct information.

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


Component Selector

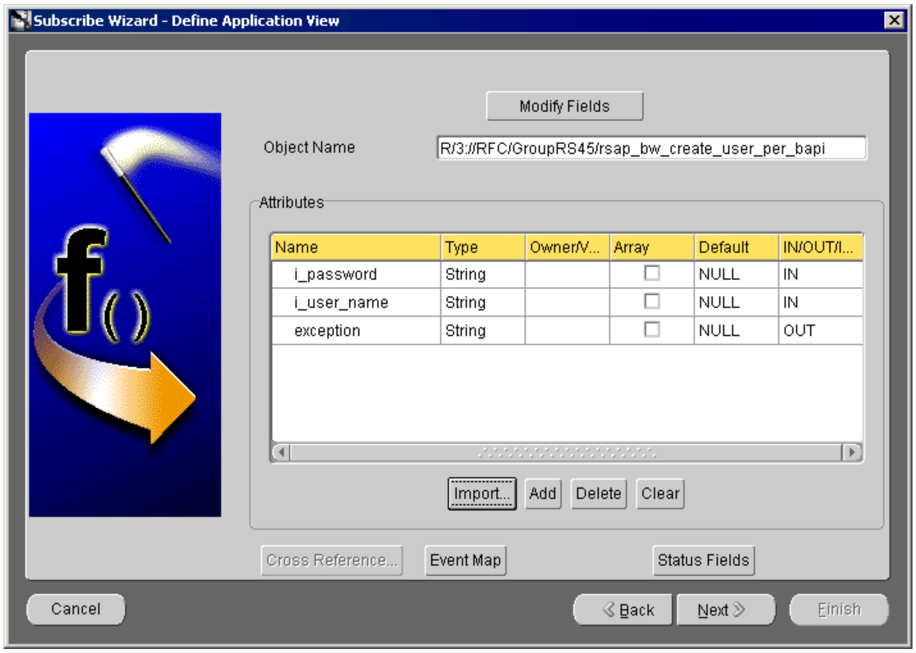
ALE
- 100_01 - Output
  - AdvancedSend
  - Send
- 682_01 - Access sequence
- 683_01 - Pricing Procedure (only in 40c)
- 684_01 - Condition Exclusion Groups
- 685_01 - Condition type
- 686a_01 - Conditions: Exclusion indicator:
- absen1 - Attendance/Absence in CC1
- acc_act_alloc01 - Accounting: Post activity allocation
- acc_act_alloc02 - Accounting: Post activity allocation
- acc_asset_trans_acq_post01 - Accounting: Post Acquisitio

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



Object Name: R/3://RFC/GroupRS45/rsap_bw_create_user_per_bapi

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
i_password	String		☐	NULL	IN
i_user_name	String		☐	NULL	IN
exception	String		☐	NULL	OUT

11. Click **Next**.

The Define Mappings dialog displays.

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

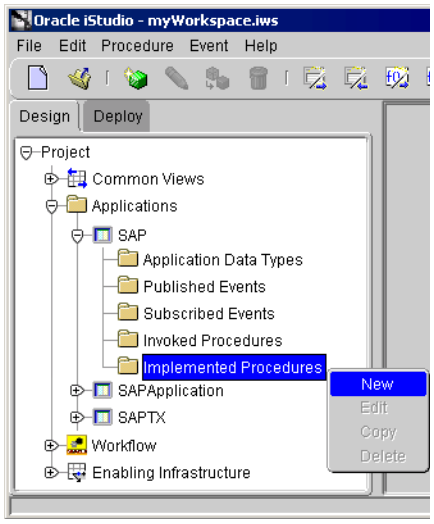
The created event displays in the right panel of iStudio.

Creating a Remote Function Call Implemented Procedure

To create a Remote Function Call implemented procedure:

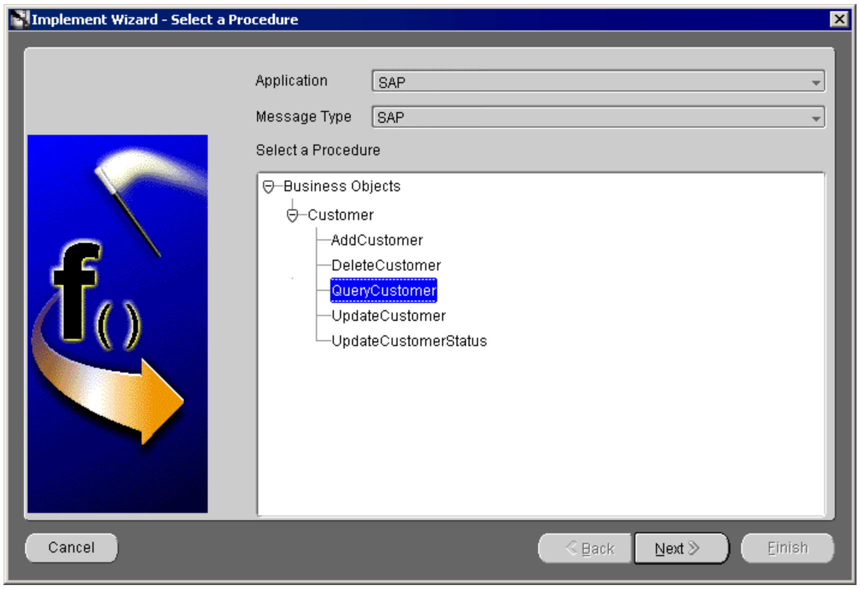
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Implemented Procedures** and select **New**.

Figure 3–12 Creating an Implemented Procedure



The Implement Wizard—Select a Procedure dialog displays.

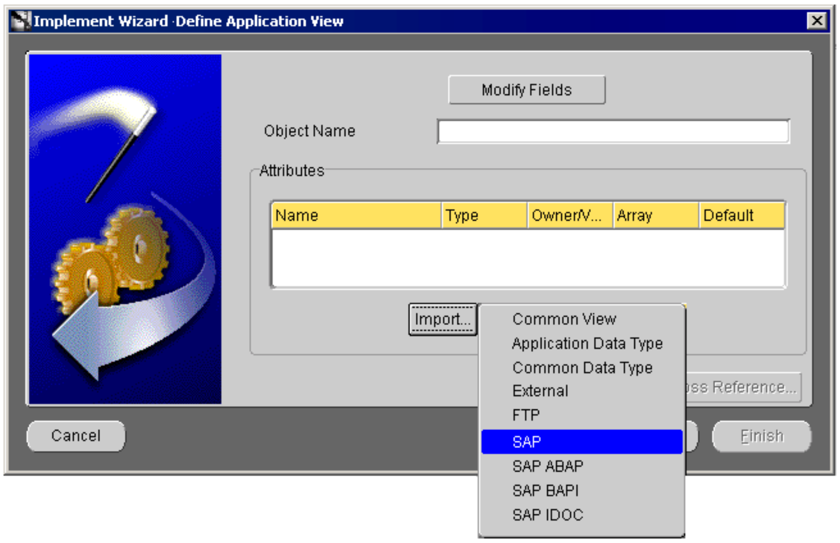
Figure 3–13 Implement Wizard - Selecting a Procedure



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.

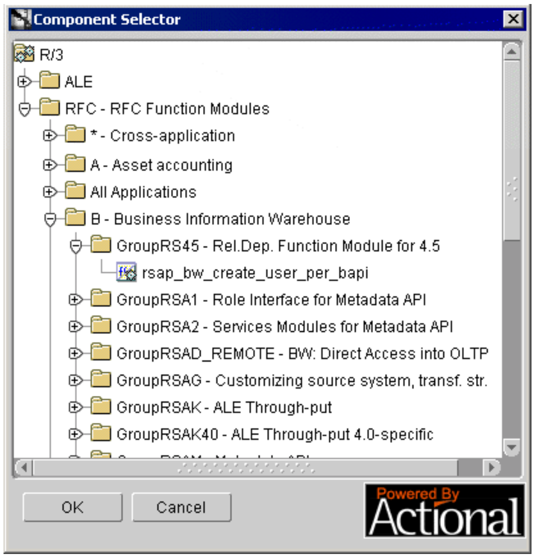
If this is the initial login for this machine, enter the correct information.

> **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



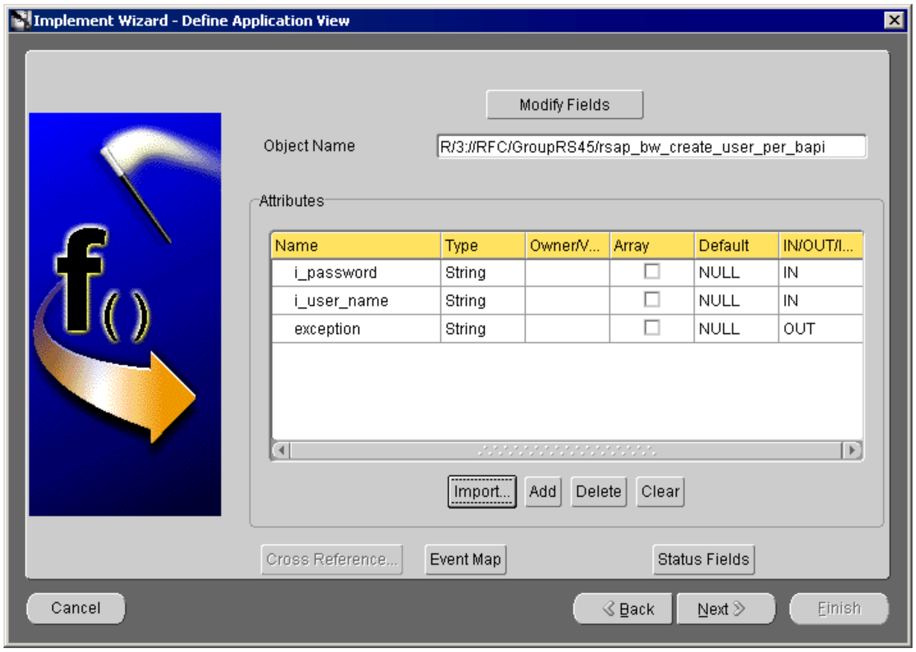
Component Selector

R/3
ALE
RFC - RFC Function Modules
* - Cross-application
A - Asset accounting
All Applications
B - Business Information Warehouse
GroupRS45 - Rel.Dep. Function Module for 4.5
rsap_bw_create_user_per_bapi
GroupRSA1 - Role Interface for Metadata API
GroupRSA2 - Services Modules for Metadata API
GroupRSAD_REMOTE - BW: Direct Access into OLTP
GroupRSAG - Customizing source system, transf. str.
GroupRSAK - ALE Through-put
GroupRSAK40 - ALE Through-put 4.0-specific

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



Object Name: R/3://RFC/GroupRS45/rsap_bw_create_user_per_bapi

Attributes

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
i_password	String		☐	NULL	IN
i_user_name	String		☐	NULL	IN
exception	String		☐	NULL	OUT

11. Click **Next**.

The Define Mappings dialog displays.

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

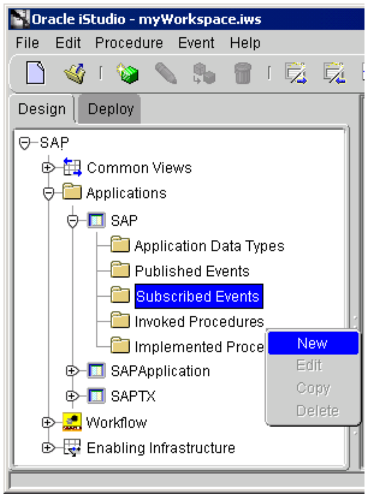
The new populated procedure display in the right panel of iStudio.

Creating a Remote Function Call Subscribed Event

To create a Remote Function Call subscribed event:

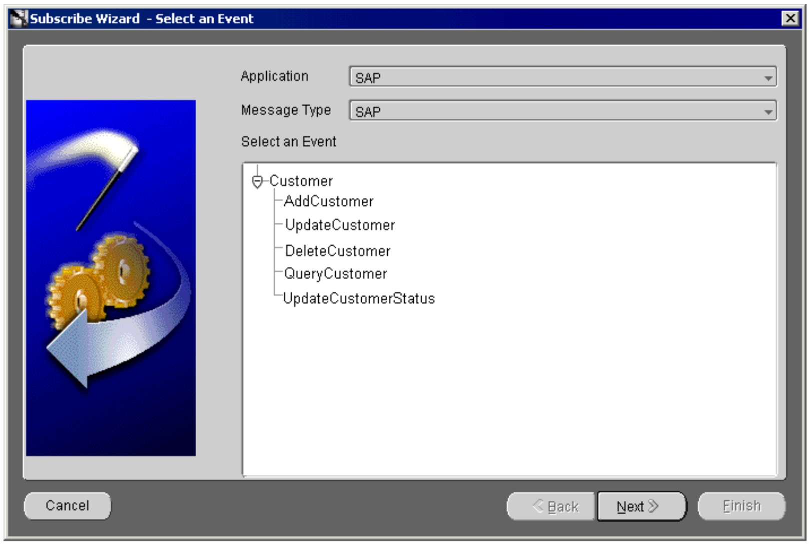
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Subscribed Events** and select **New**.

Figure 3–17 Creating a Subscribed Event



The Subscribe Wizard—Select an Event dialog displays.

Figure 3–18 Subscribe Wizard - Selecting an Event



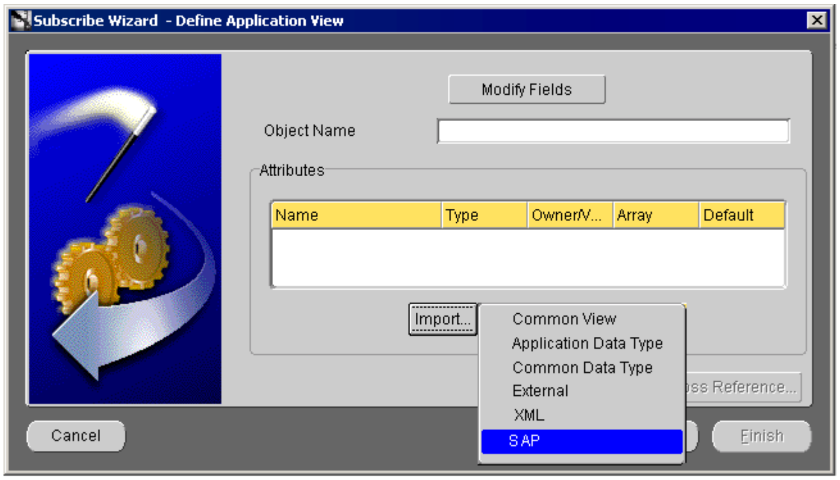
Application: SAP
Message Type: SAP
Select an Event

```
Customer
 ├AddCustomer
 ├UpdateCustomer
 ├DeleteCustomer
 ├QueryCustomer
 └UpdateCustomerStatus
```

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.

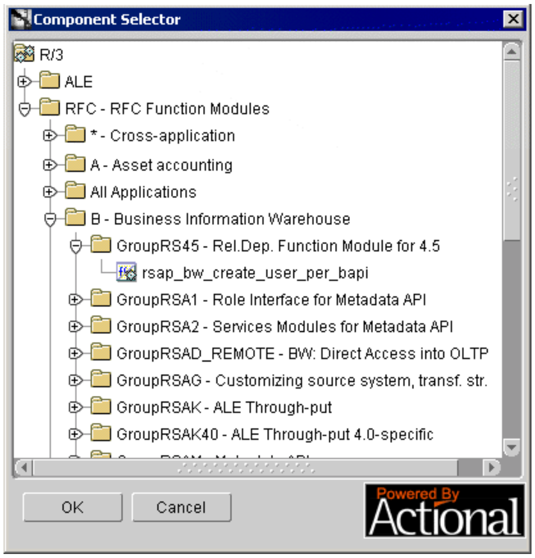
   If this is the initial login for this machine, enter the correct information.

   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



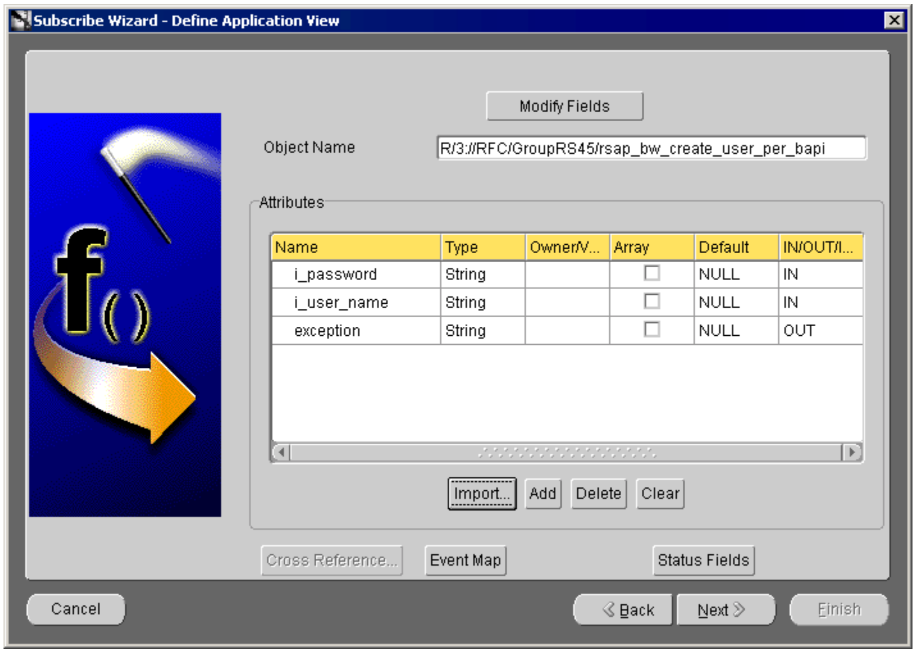
R/3
ALE
RFC - RFC Function Modules
* - Cross-application
A - Asset accounting
All Applications
B - Business Information Warehouse
GroupRS45 - Rel.Dep. Function Module for 4.5
rsap_bw_create_user_per_bapi
GroupRSA1 - Role Interface for Metadata API
GroupRSA2 - Services Modules for Metadata API
GroupRSAD_REMOTE - BW: Direct Access into OLTP
GroupRSAG - Customizing source system, transf. str.
GroupRSAK - ALE Through-put
GroupRSAK40 - ALE Through-put 4.0-specific

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



Object Name: R/3://RFC/GroupRS45/rsap_bw_create_user_per_bapi

Attributes

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
i_password	String		☐	NULL	IN
i_user_name	String		☐	NULL	IN
exception	String		☐	NULL	OUT

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.

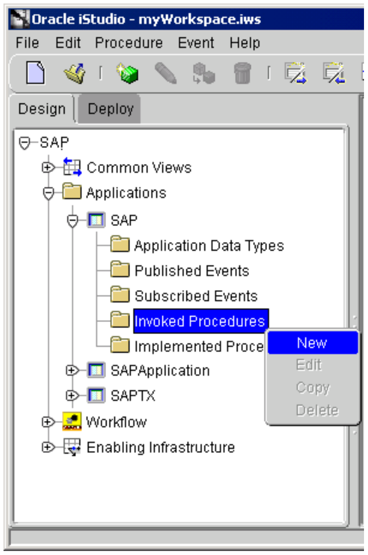
Outbound From SAP

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

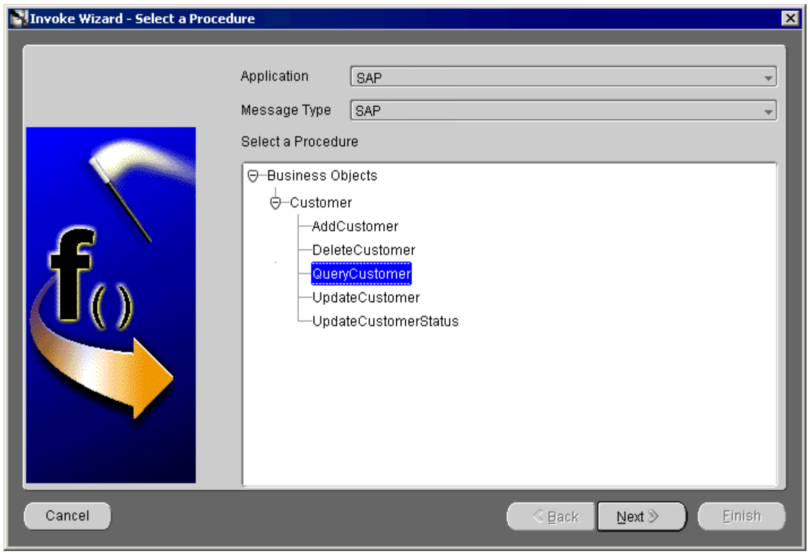
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Invoked Procedures** and select **New**.

Figure 3–22 Creating an Implemented Procedure



The Invoke Wizard—Select a Procedure dialog displays.

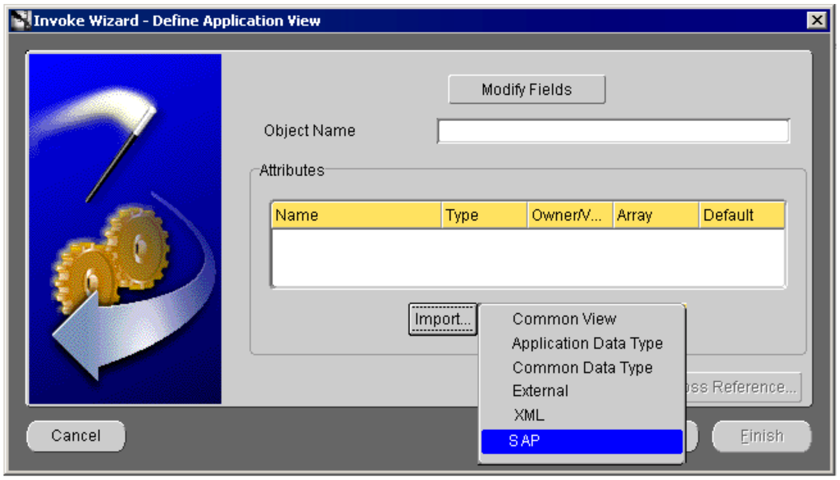
Figure 3–23 Invoke Wizard - Selecting a Procedure



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.

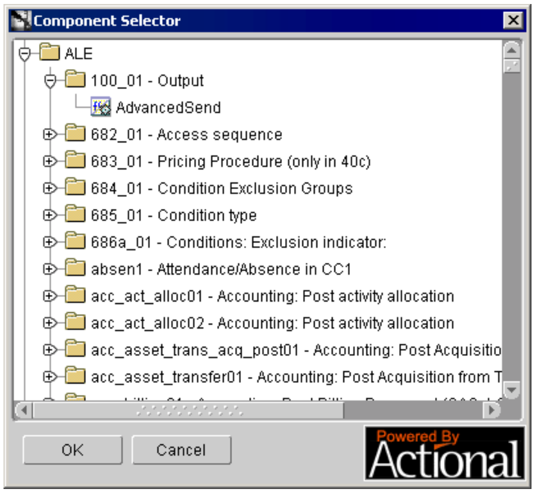
   If this is the initial login for this machine, enter the correct information.

   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.

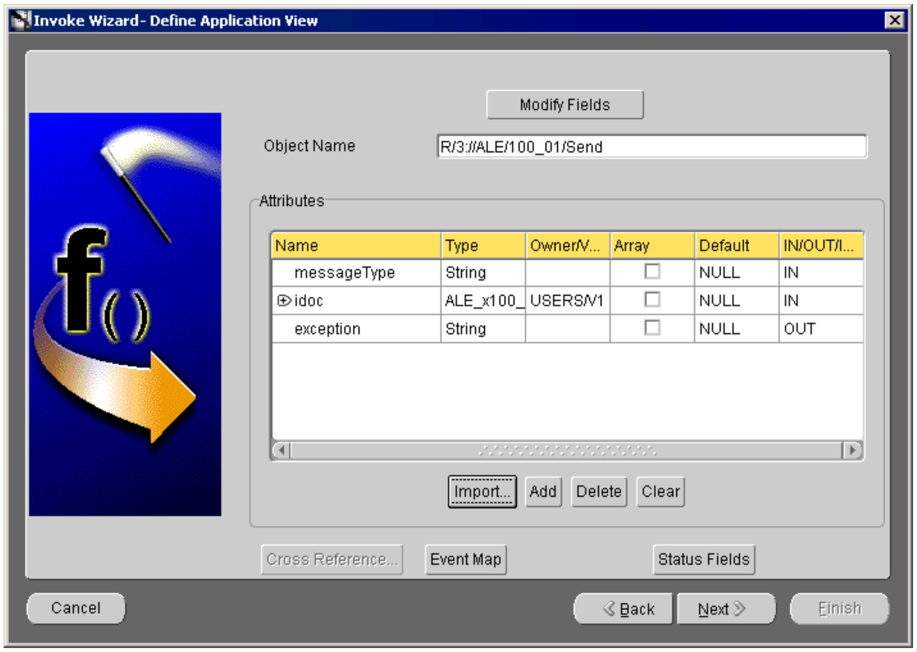
Figure 3–25 Component Selector - Application Link Enabling


Component Selector
ALE
100_01 - Output
AdvancedSend
682_01 - Access sequence
683_01 - Pricing Procedure (only in 40c)
684_01 - Condition Exclusion Groups
685_01 - Condition type
686a_01 - Conditions: Exclusion indicator:
absen1 - Attendance/Absence in CC1
acc_act_alloc01 - Accounting: Post activity allocation
acc_act_alloc02 - Accounting: Post activity allocation
acc_asset_trans_acq_post01 - Accounting: Post Acquisitio
acc_asset_transfer01 - Accounting: Post Acquisition from T
OK
Cancel
Powered By Actional

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



Object Name: R/3://ALE/100_01/Send

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
messageType	String		☐	NULL	IN
⊕ idoc	ALE_x100_	USERS/V1	☐	NULL	IN
exception	String		☐	NULL	OUT

11. Click **Next**.

    The Define Mappings dialog displays.

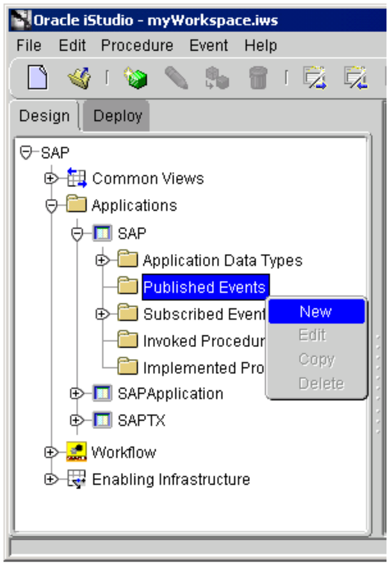
12. Click **New** to define mappings and click **Finish**.

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

Creating an Application Link Enabling Published Event

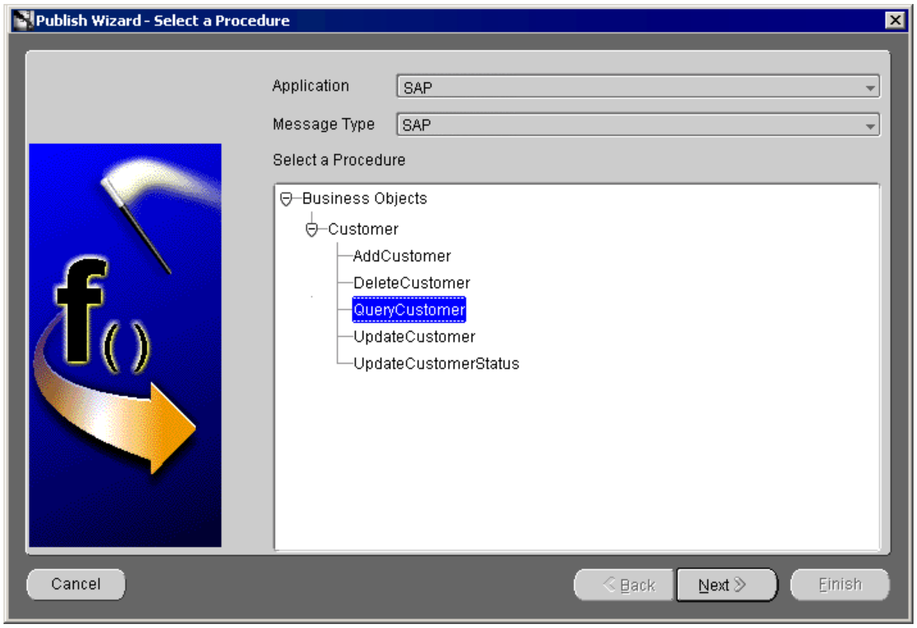
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Published Events** and select **New**.

Figure 3–27 Creating a Published Event



The Publish Wizard—Select an Procedure dialog displays.

Figure 3–28 Publish Wizard - Selecting an Event


Publish Wizard - Select a Procedure

Application: SAP

Message Type: SAP

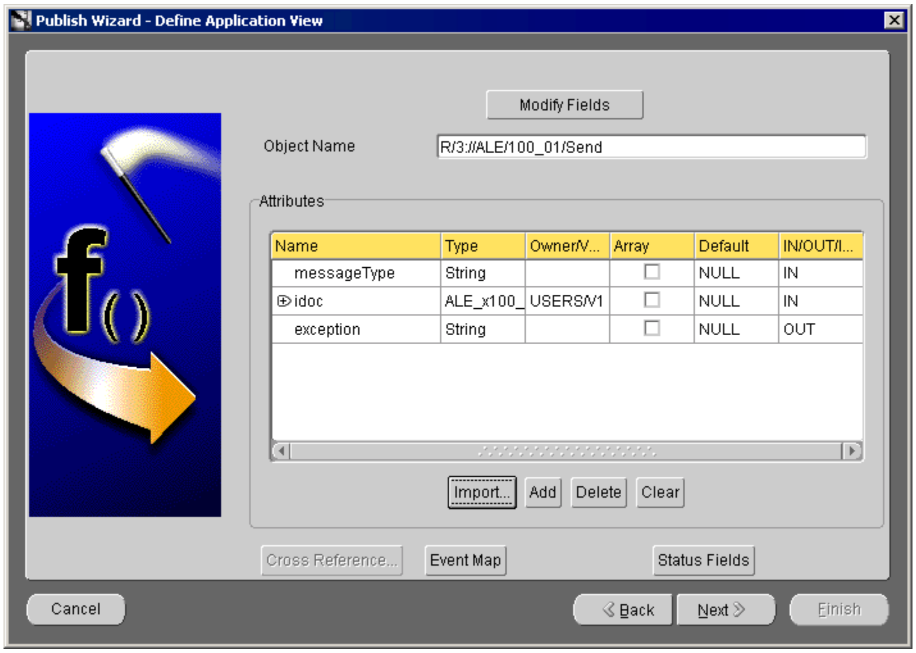
Select a Procedure

- Business Objects
  - Customer
    - AddCustomer
    - DeleteCustomer
    - QueryCustomer
    - UpdateCustomer
    - UpdateCustomerStatus

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



Object Name: R/3://ALE/100_01/Send

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
messageType	String		☐	NULL	IN
⊕ idoc	ALE_x100_	USERS/V1	☐	NULL	IN
exception	String		☐	NULL	OUT

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

The SAP Login dialog displays.

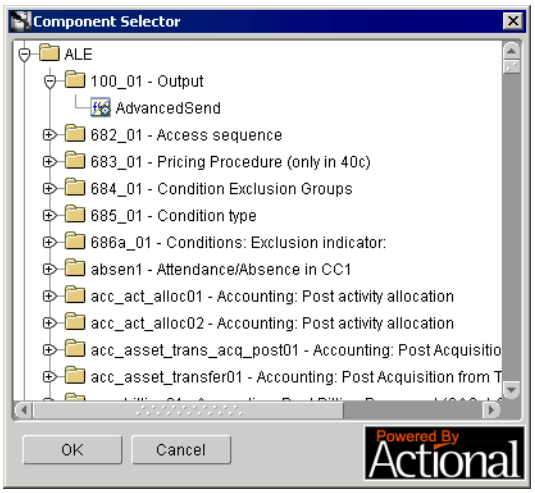
If this is the initial login for this machine, enter the correct information.

> **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


Component Selector

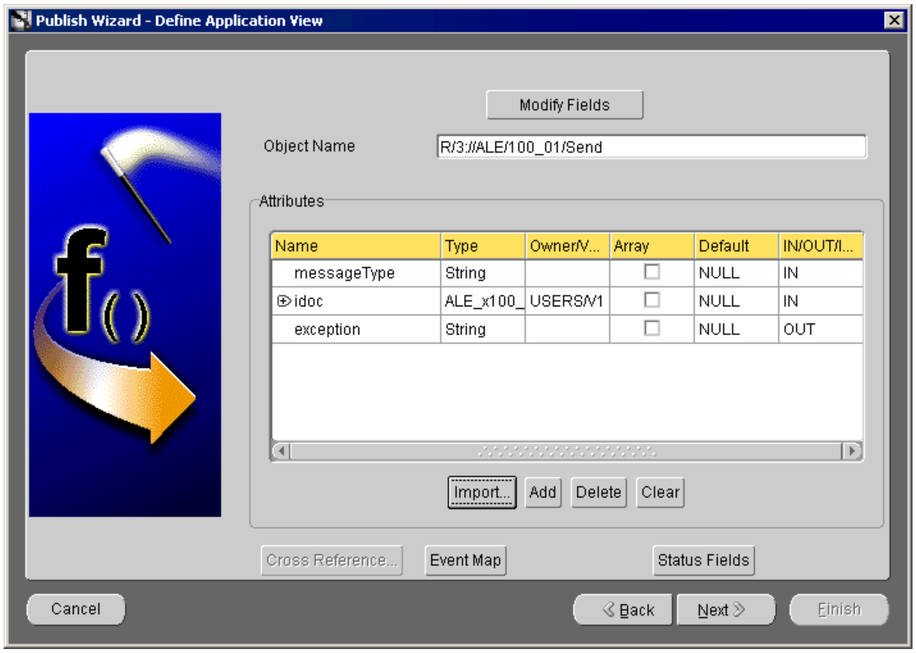
- ALE
  - 100_01 - Output
    - AdvancedSend
  - 682_01 - Access sequence
  - 683_01 - Pricing Procedure (only in 40c)
  - 684_01 - Condition Exclusion Groups
  - 685_01 - Condition type
  - 686a_01 - Conditions: Exclusion indicator:
  - absen1 - Attendance/Absence in CC1
  - acc_act_alloc01 - Accounting: Post activity allocation
  - acc_act_alloc02 - Accounting: Post activity allocation
  - acc_asset_trans_acq_post01 - Accounting: Post Acquisitio
  - acc_asset_transfer01 - Accounting: Post Acquisition from T

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


Publish Wizard - Define Application View

Object Name: R/3://ALE/100_01/Send

Attributes

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
messageType	String		☐	NULL	IN
⊕ idoc	ALE_x100_	USERS/V1	☐	NULL	IN
exception	String		☐	NULL	OUT

11. Click **Next**.

The Define Mappings dialog displays.

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

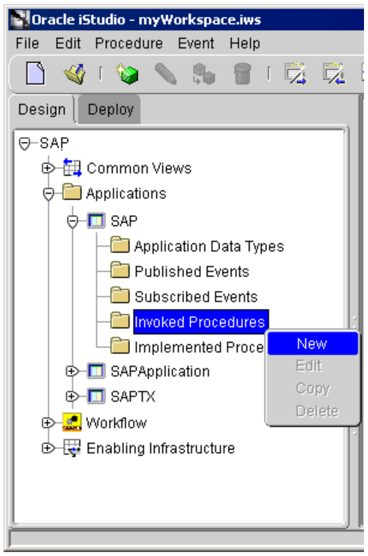
The new populated event will display in the right panel in iStudio.

Creating a Remote Function Call Invoked Procedure

To create a Remote Function Call invoked procedure in iStudio:

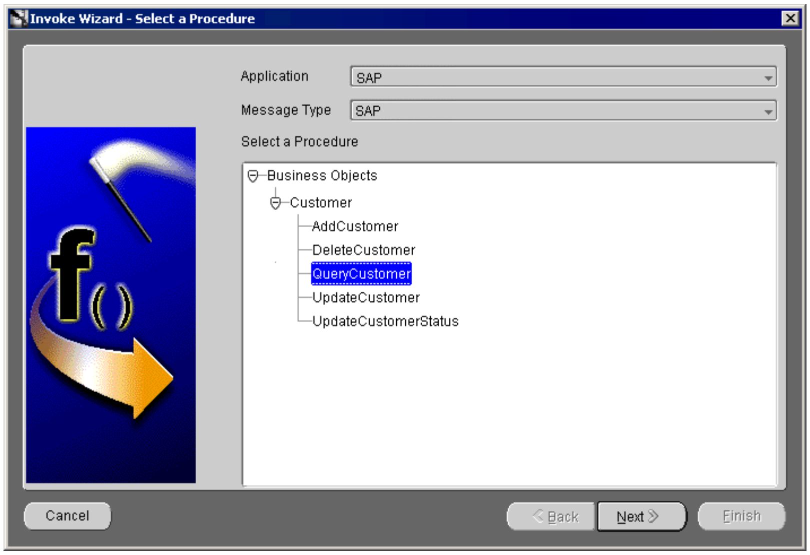
1. Start iStudio.
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Invoked Procedures** and select **New**.

Figure 3–32 Creating an Invoked Procedure



The Invoke Wizard—Select a Procedure dialog displays.

Figure 3–33 Invoke Wizard - Selecting a Procedure



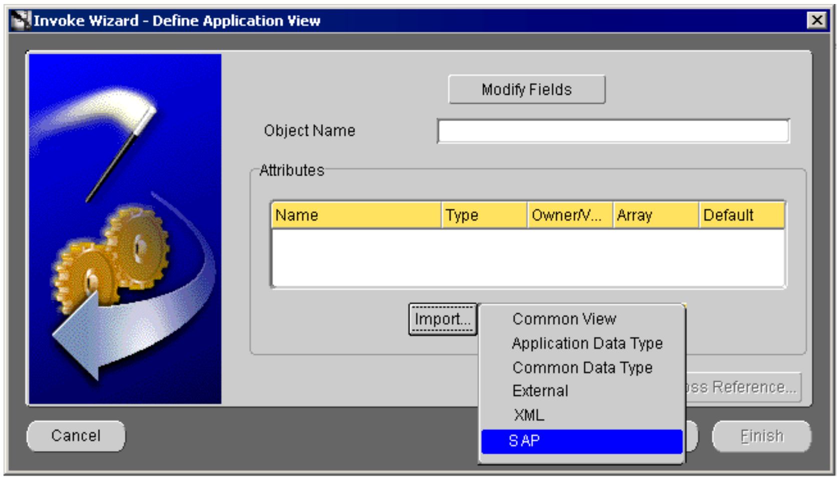
Application: SAP
Message Type: SAP
Select a Procedure

- Business Objects
  - Customer
    - AddCustomer
    - DeleteCustomer
    - QueryCustomer
    - UpdateCustomer
    - UpdateCustomerStatus

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.

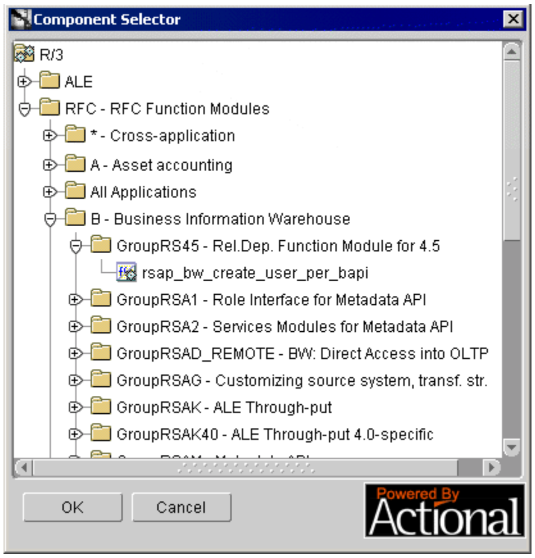
If this is the initial login for this machine, enter the correct information.

> **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.

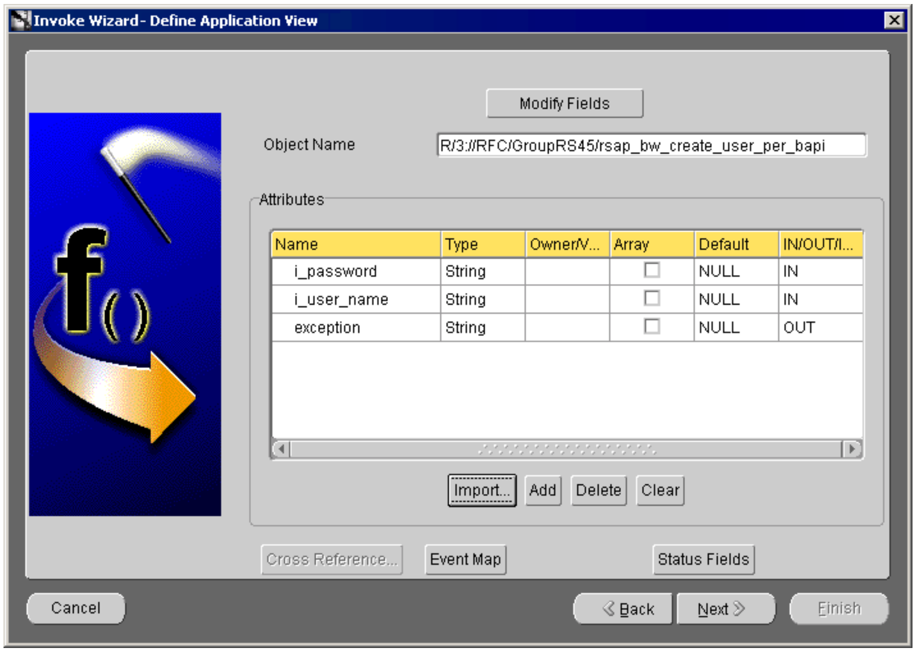
Figure 3–35 Component Selector - Remote Function Call


Component Selector
R/3
ALE
RFC - RFC Function Modules
* - Cross-application
A - Asset accounting
All Applications
B - Business Information Warehouse
GroupRS45 - Rel.Dep. Function Module for 4.5
rsap_bw_create_user_per_bapi
GroupRSA1 - Role Interface for Metadata API
GroupRSA2 - Services Modules for Metadata API
GroupRSAD_REMOTE - BW: Direct Access into OLTP
GroupRSAG - Customizing source system, transf. str.
GroupRSAK - ALE Through-put
GroupRSAK40 - ALE Through-put 4.0-specific
OK
Cancel
Powered By Actional

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



Object Name: R/3://RFC/GroupRS45/rsap_bw_create_user_per_bapi

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
i_password	String		☐	NULL	IN
i_user_name	String		☐	NULL	IN
exception	String		☐	NULL	OUT

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

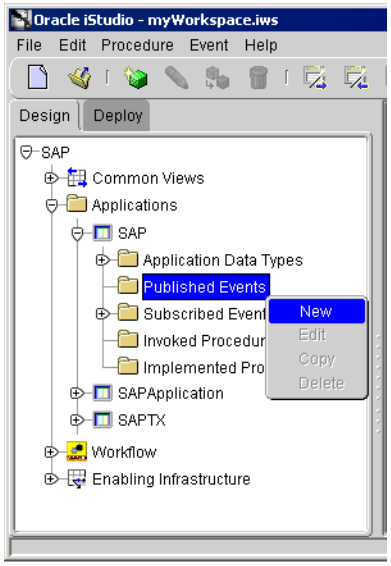
    The new populated event displays in the right panel of iStudio.

Creating a Remote Function Call Published Event

To create a Remote Function Call published event in iStudio:

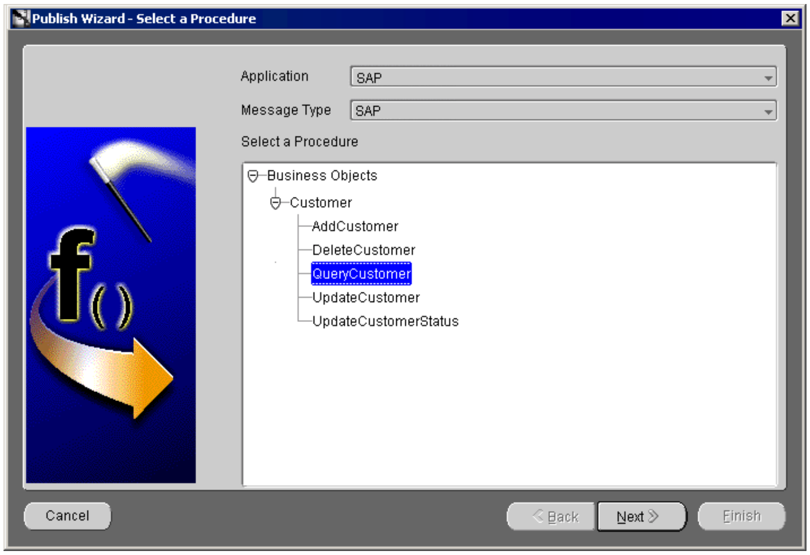
1. Start iStudio
2. Open your project.
3. Expand the Applications folder.
4. Expand your application.
5. Right-click **Published Events** and select **New**.

Figure 3–37 Creating a Published Event



The Publish Wizard—Select a Procedure dialog displays.

Figure 3–38 Publish Wizard - Selecting a Procedure



Application: SAP

Message Type: SAP

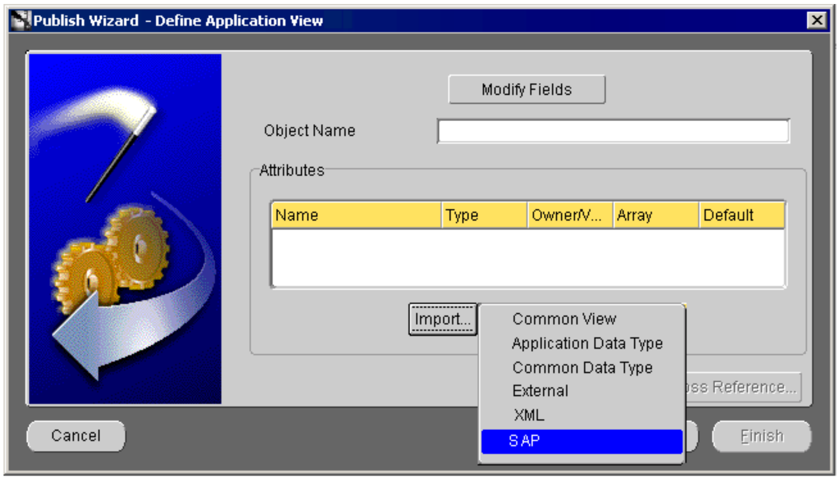
Select a Procedure

- Business Objects
  - Customer
    - AddCustomer
    - DeleteCustomer
    - QueryCustomer
    - UpdateCustomer
    - UpdateCustomerStatus

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.

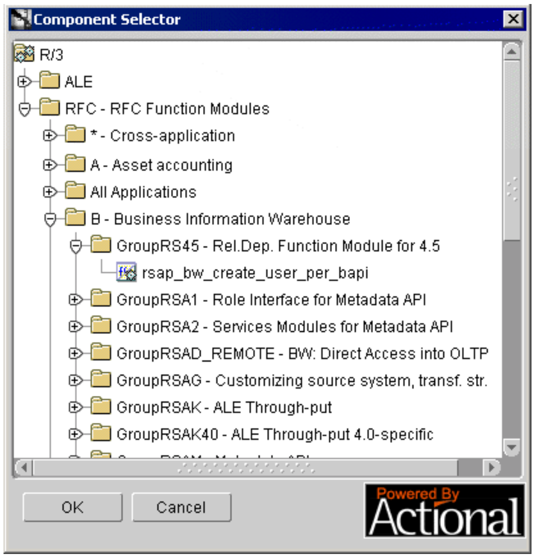
   If this is the initial login for this machine, enter the correct information.

   > **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

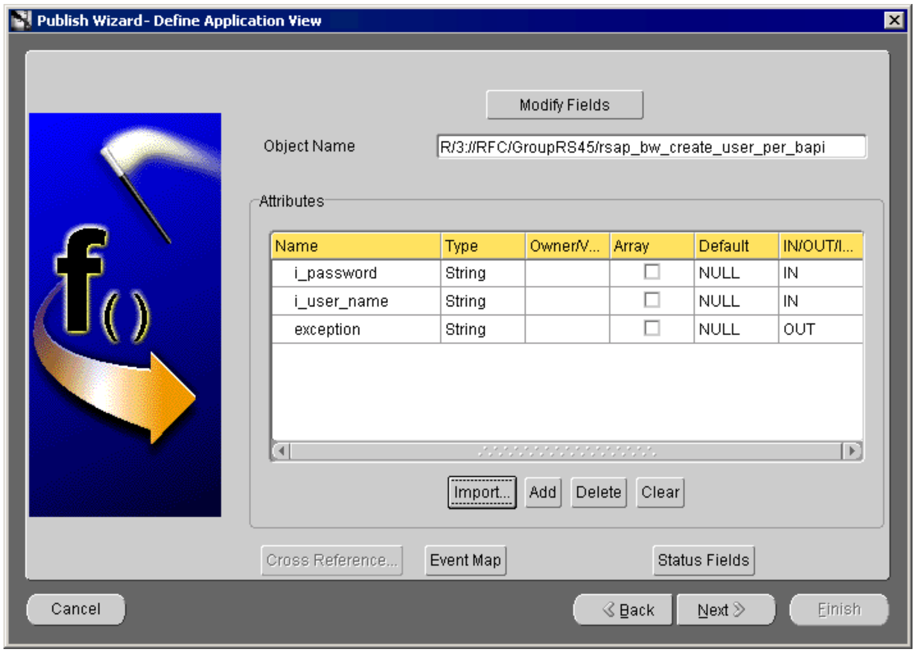


Component Selector
R/3
ALE
RFC - RFC Function Modules
* - Cross-application
A - Asset accounting
All Applications
B - Business Information Warehouse
GroupRS45 - Rel.Dep. Function Module for 4.5
rsap_bw_create_user_per_bapi
GroupRSA1 - Role Interface for Metadata API
GroupRSA2 - Services Modules for Metadata API
GroupRSAD_REMOTE - BW: Direct Access into OLTP
GroupRSAG - Customizing source system, transf. str.
GroupRSAK - ALE Through-put
GroupRSAK40 - ALE Through-put 4.0-specific
OK
Cancel
Powered By Actional

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.



Object Name: R/3://RFC/GroupRS45/rsap_bw_create_user_per_bapi

Attributes

Name	Type	Owner/V...	Array	Default	IN/OUT/I...
i_password	String		☐	NULL	IN
i_user_name	String		☐	NULL	IN
exception	String		☐	NULL	OUT

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
- Application Line Enabling Terminology
- Application Link Enabling Subdirectories—Queue and Cache
- Inbound Intermediate Documents
- Outbound Intermediate Documents
- R/3 Application Link Enabling Configuration
- Application Link Enabling—Exploring Intermediate Document Types
- Manually Downloading an IDOC
- Enhance Application Link Enabling Remote Browsing

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
- Oracle9iAS InterConnect Application Acting as a Client
- Oracle9iAS InterConnect 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.

Oracle9*i*AS InterConnect Application Acting as a Client

If you want to make your Oracle9*i*AS 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.

Oracle9*i*AS InterConnect Application Acting as a Server

You can have your Oracle9*i*AS 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 Oracle9*i*AS 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 Oracle9*i*AS InterConnect message will be constructed and sent to the Oracle9*i*AS 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.

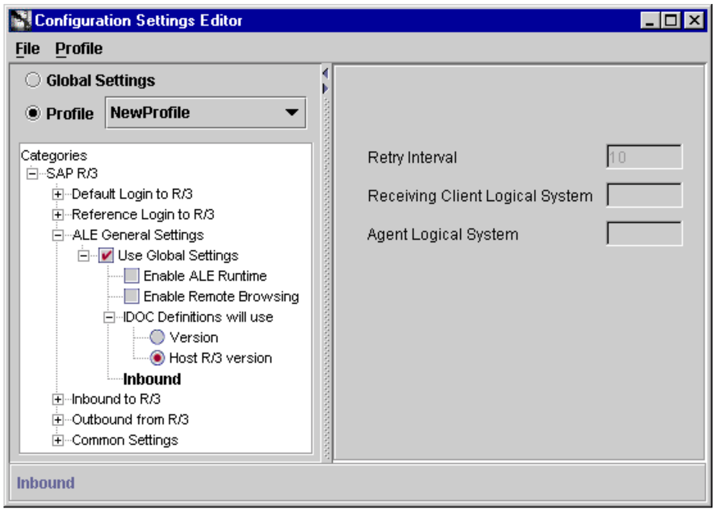
See Also: "Manually Downloading an IDOC" on page 4-23

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



Retry Interval: 10
Receiving Client Logical System:
Agent Logical System:

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 Oracle9*i*AS 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 Oracle9*i*AS 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.

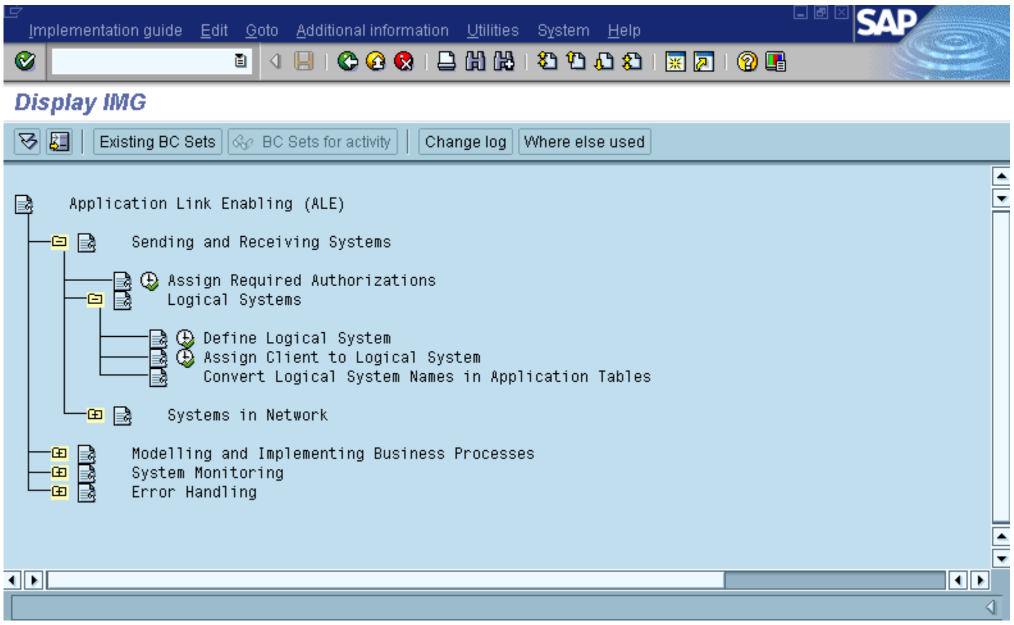
> **See Also:** *SAP Implementation Guide for ALE* contains specific R/3 customization instructions on how to create, or find, an existing logical system

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)



```
Application Link Enabling (ALE)
    Sending and Receiving Systems
        Assign Required Authorizations
        Logical Systems
            Define Logical System
            Assign Client to Logical System
            Convert Logical System Names in Application Tables
        Systems in Network
    Modelling and Implementing Business Processes
    System Monitoring
    Error Handling
```

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

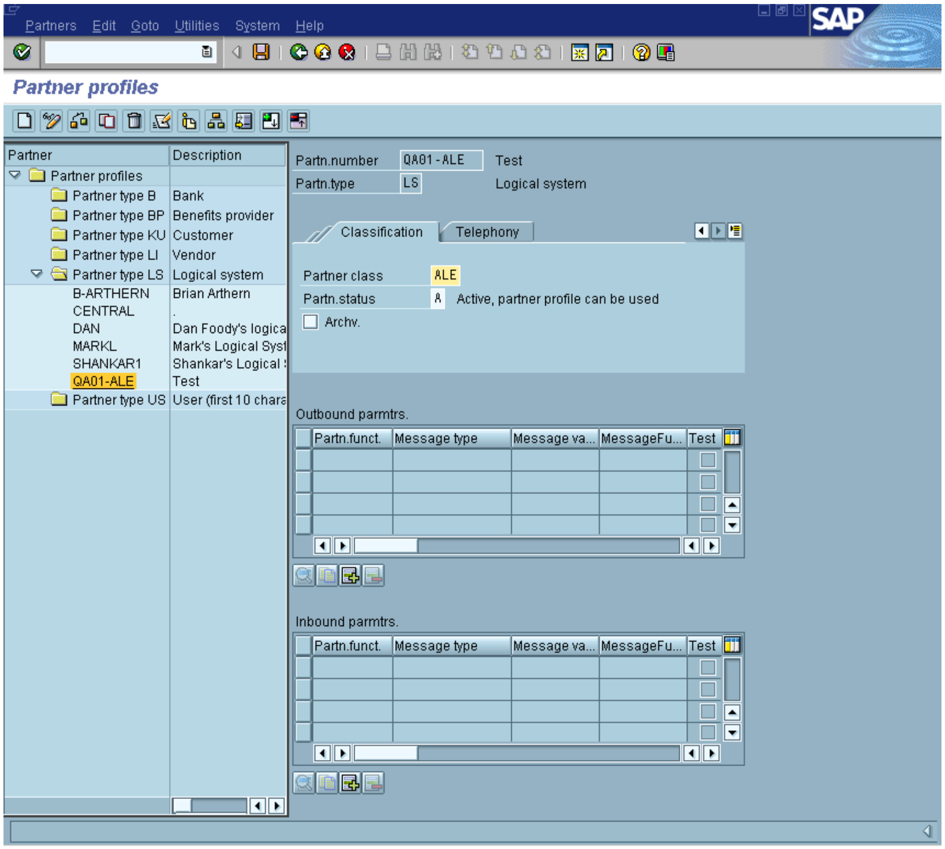
- For example, SAP recommends the naming standard for the base logical system as `XXXCLNTyyy`.

  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



Partner	Description
Partner profiles	
Partner type B	Bank
Partner type BP	Benefits provider
Partner type KU	Customer
Partner type LI	Vendor
Partner type LS	Logical system
B-ARTHERN	Brian Arthern
CENTRAL	.
DAN	Dan Foody's logica
MARKL	Mark's Logical Syst
SHANKAR1	Shankar's Logical :
QA01-ALE	Test
Partner type US	User (first 10 chara

Partn.number	QA01-ALE	Test
Partn.type	LS	Logical system

Partner class	ALE	
Partn.status	A	Active, partner profile can be used
☐ Archv.		

Outbound parmtrs.

Partn.funct.	Message type	Message va...	MessageFu...	Test
				☐
				☐
				☐
				☐

Inbound parmtrs.

Partn.funct.	Message type	Message va...	MessageFu...	Test
				☐
				☐
				☐
				☐

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.

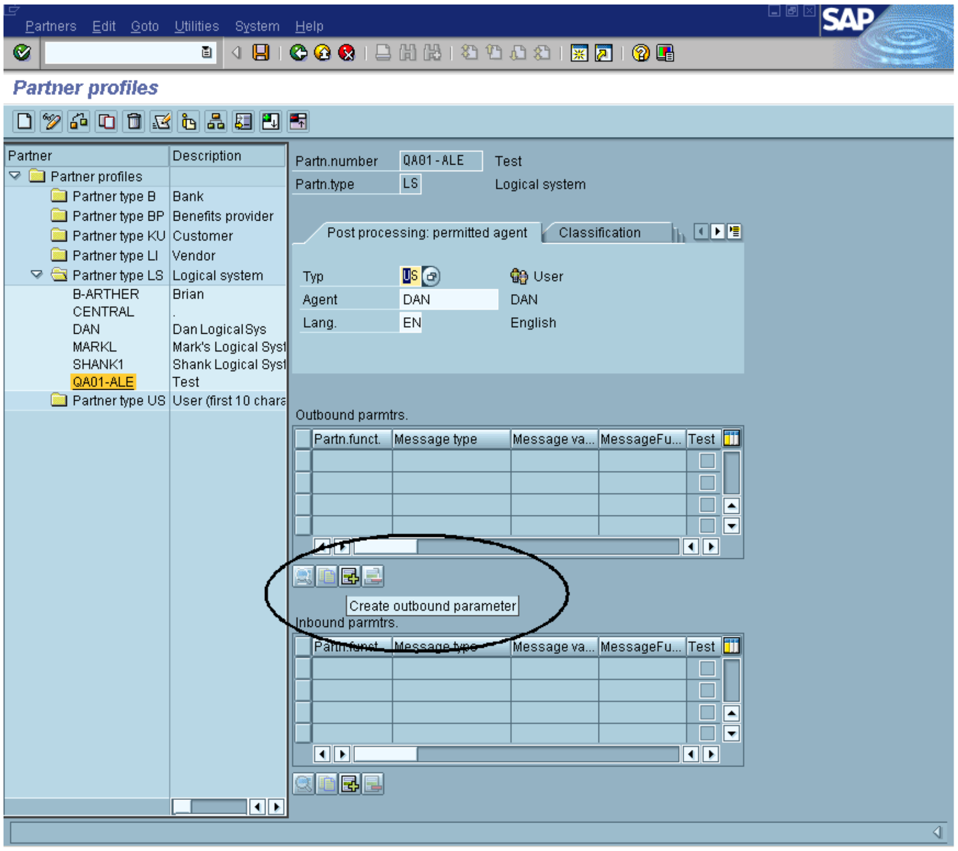
7. Enter **A** in the Partn. status field.
8. Click **Save** to create the partner.

Step 6 Maintaining Outbound Parameters

To maintain the outbound parameters:

1. Click **Outbound Parameters**.

Figure 4–4 CreateOutbound Parameters Button


Partners Edit Goto Utilities System Help
SAP
Partner profiles
Create outbound parameter

Partner	Description
Partner profiles	
Partner type B	Bank
Partner type BP	Benefits provider
Partner type KU	Customer
Partner type LI	Vendor
Partner type LS	Logical system
B-ARTHER	Brian
CENTRAL	.
DAN	Dan LogicalSys
MARKL	Mark's Logical Syst
SHANK1	Shank Logical Syst
QA01-ALE	Test
Partner type US	User (first 10 chara

Partn.number QA01-ALE Test
Partn.type LS Logical system

Post processing: permitted agent | Classification

Typ US User
Agent DAN DAN
Lang. EN English

Outbound parmtrs.

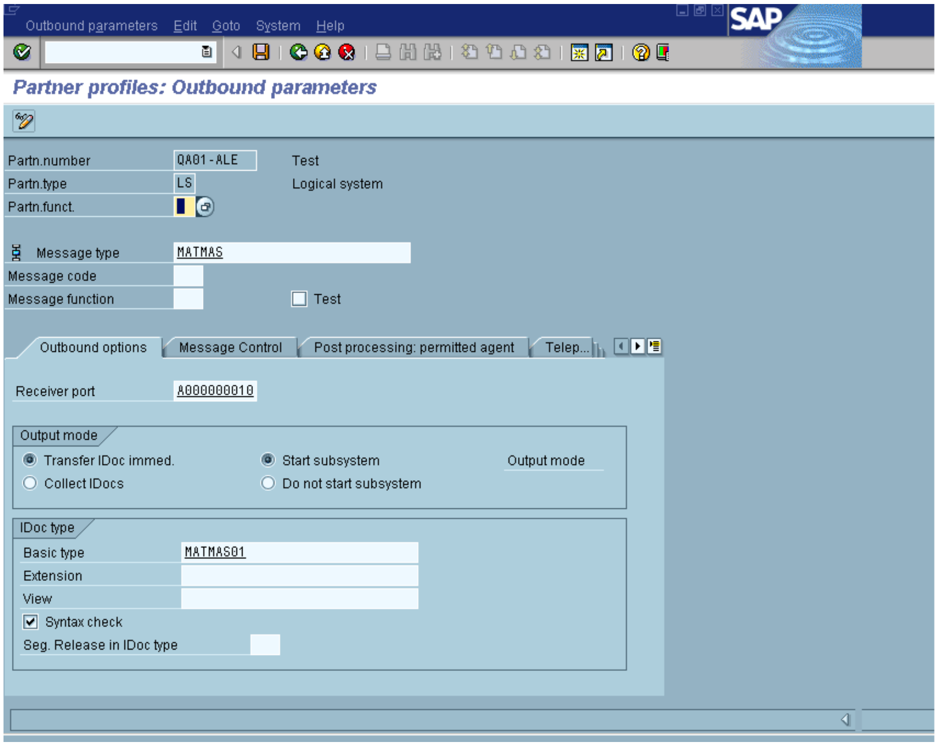
Partn.funct.	Message type	Message va...	MessageFu...	Test

Inbound parmtrs.

Partn.funct.	Message type	Message va...	MessageFu...	Test

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

Figure 4–5 Enter the Information



Partner profiles: Outbound parameters

Partn.number	QA01-ALE	Test
Partn.type	LS	Logical system
Partn.funct.		
Message type	MATMAS	
Message code		
Message function		☐ Test

Outbound options | Message Control | Post processing: permitted agent | Telep...

Receiver port A000000010

Output mode
- (●) Transfer IDoc immed.
- ( ) Collect IDocs
- (●) Start subsystem
- ( ) Do not start subsystem
- Output mode

IDoc type

Basic type	MATMAS01
Extension	
View	
☑ Syntax check	
Seg. Release in IDoc type	

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 ledidtyp.
   ```

   > **See Also:** "How to Install the Remote Browsing Function Modules" on page 4-26 if `include ledidtyp` 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 Oracle9*i*AS 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

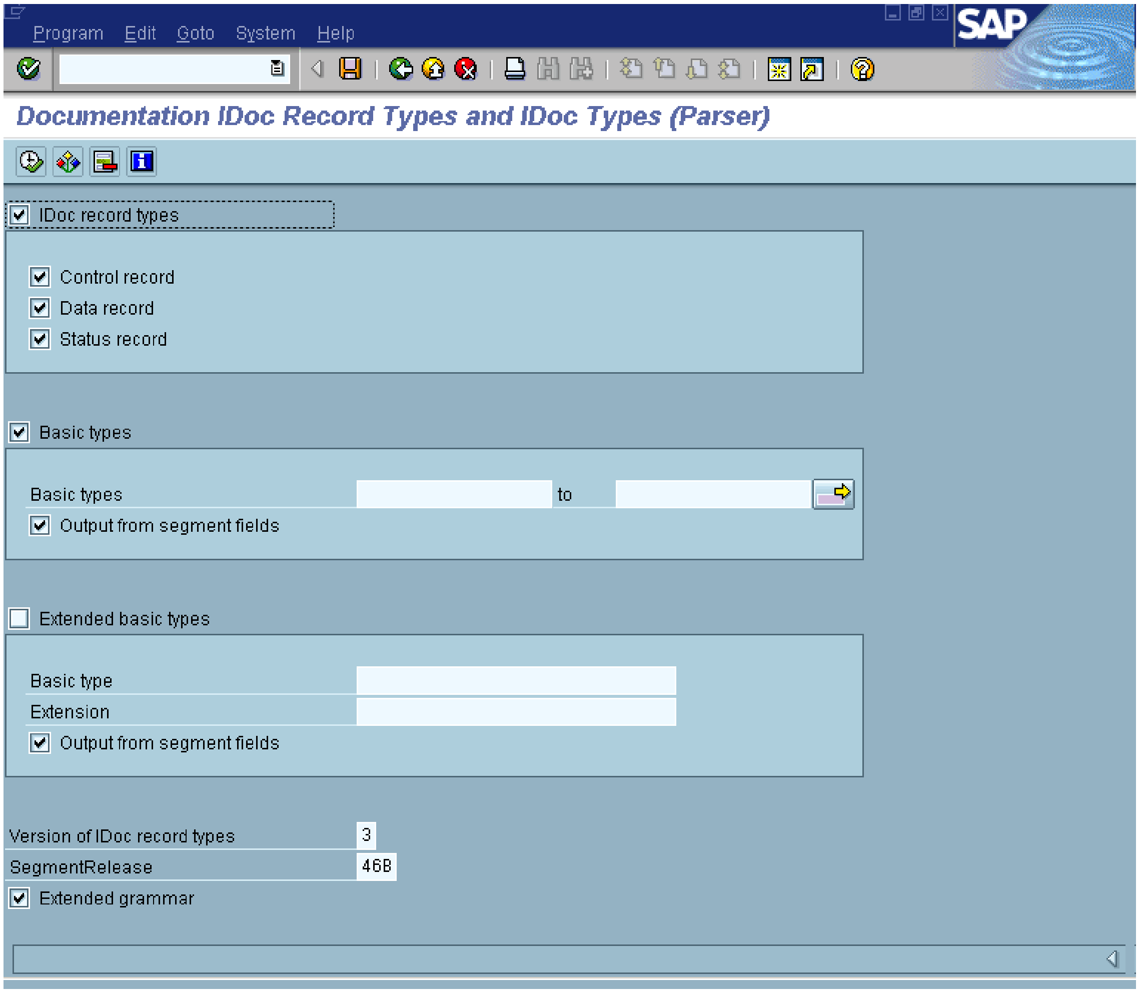
4. Log into an R/3 System.

   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)



Documentation IDoc Record Types and IDoc Types (Parser)

☑ IDoc record types

☑ Control record
☑ Data record
☑ Status record

☑ Basic types

Basic types to
☑ Output from segment fields

☐ Extended basic types

Basic type
Extension
☑ Output from segment fields

Version of IDoc record types	3
SegmentRelease	46B

☑ Extended grammar

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 ledidtype` 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

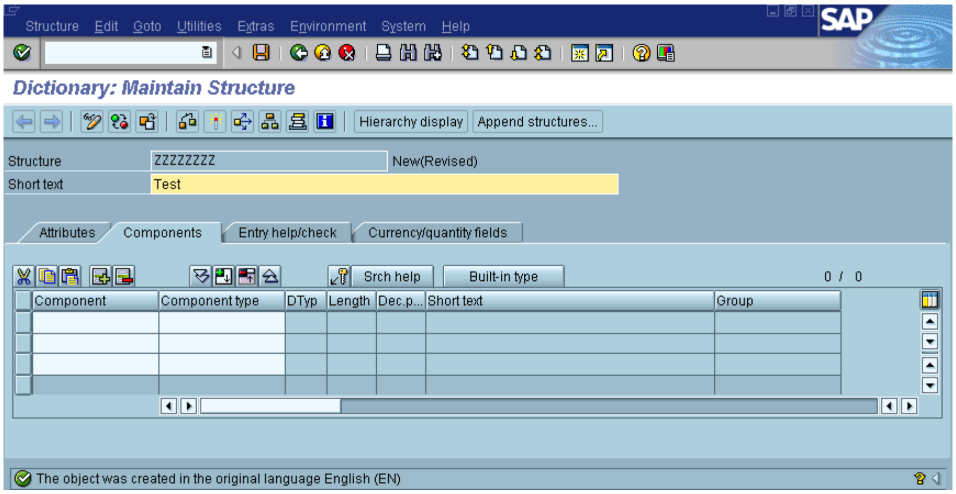
Create Four Structures

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



Dictionary: Maintain Structure

Structure	ZZZZZZZZ	New(Revised)
Short text	Test	

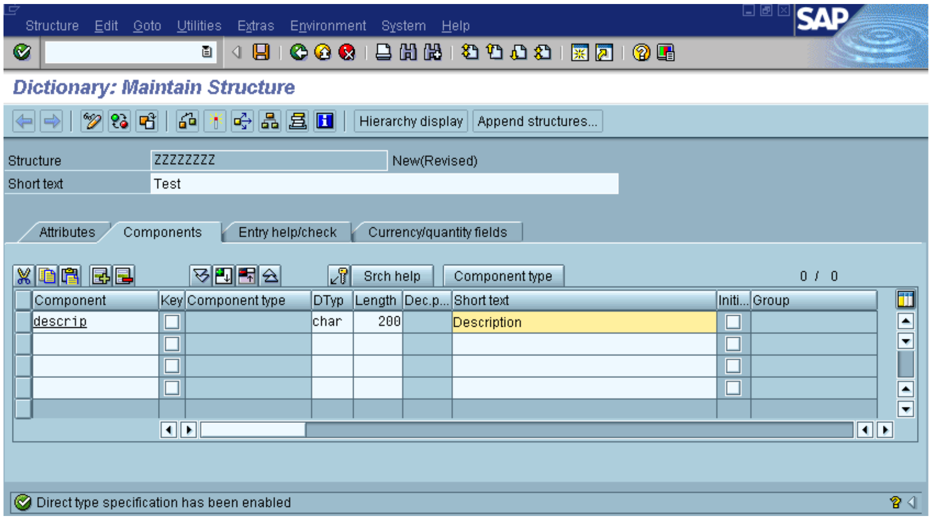
Component	Component type	DTyp	Length	Dec.p...	Short text	Group

The object was created in the original language English (EN)

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



Structure ZZZZZZZZ New(Revised)
Short text Test

0 / 0

Component	Key	Component type	DTyp	Length	Dec.p...	Short text	Initi...	Group
descrip	☐		char	200		Description	☐	
	☐						☐	
	☐						☐	
	☐						☐	

Direct type specification has been enabled

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.

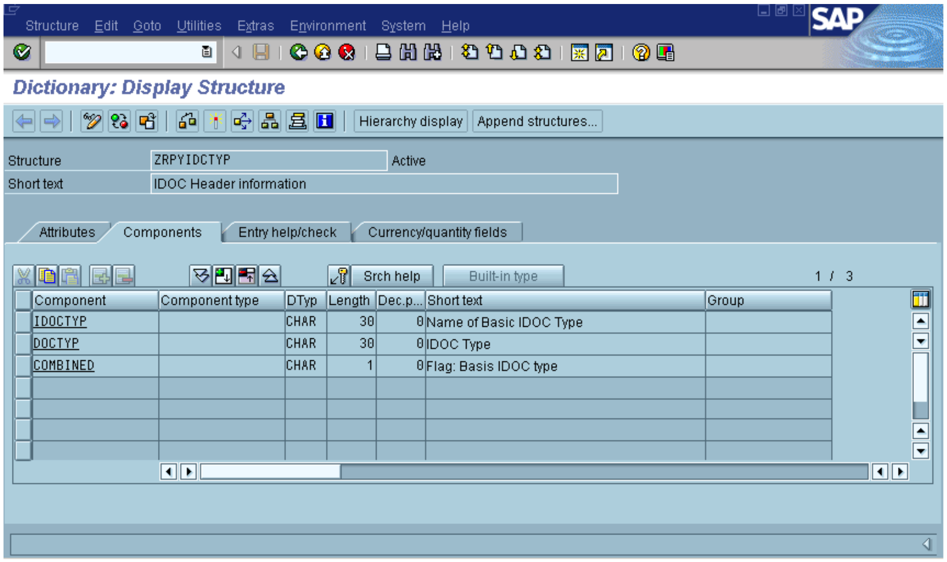
17. Press **Ctrl+F2** to check the consistency of the structure.

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 Field name and Data elem fields. Table 4–3 lists the table structure and display fields.

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



Structure ZRPYIDCTYP Active
Short text IDOC Header information

Component	Component type	DTyp	Length	Dec.p...	Short text	Group
IDOCTYP		CHAR	30	0	Name of Basic IDOC Type	
DOCTYP		CHAR	30	0	IDOC Type	
COMBINED		CHAR	1	0	Flag: Basis IDOC type	

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

Name	ZRPYIDCTYP		
Short text	Intermediate Document Header information		
Field name	**Type**	**Length**	**Short Text**
IDOCTYP	CHAR	30	Name of Basic intermediate document type
DOCTYP	CHAR	30	Intermediate Document Type
COMBINED	CHAR	1	Flag: Basis intermediate document type

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

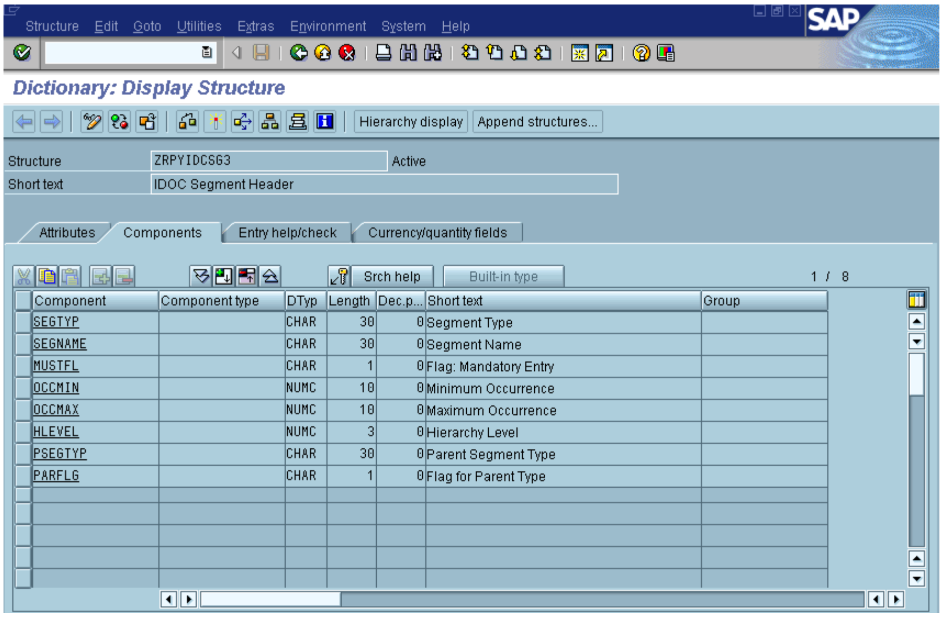
Name	ZRPYIDCSG3		
Short text	Intermediate Document Segment header		
Field name	**Type**	**Length**	**Short Text**

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

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	1	Flag for parent segment

Create the ZRPYIDCTYP Structure Figure 4–10 describes creating the ZRPYIDCTYP structure.

Figure 4–10 Dictionary: Table/Structure Display



Dictionary: Display Structure

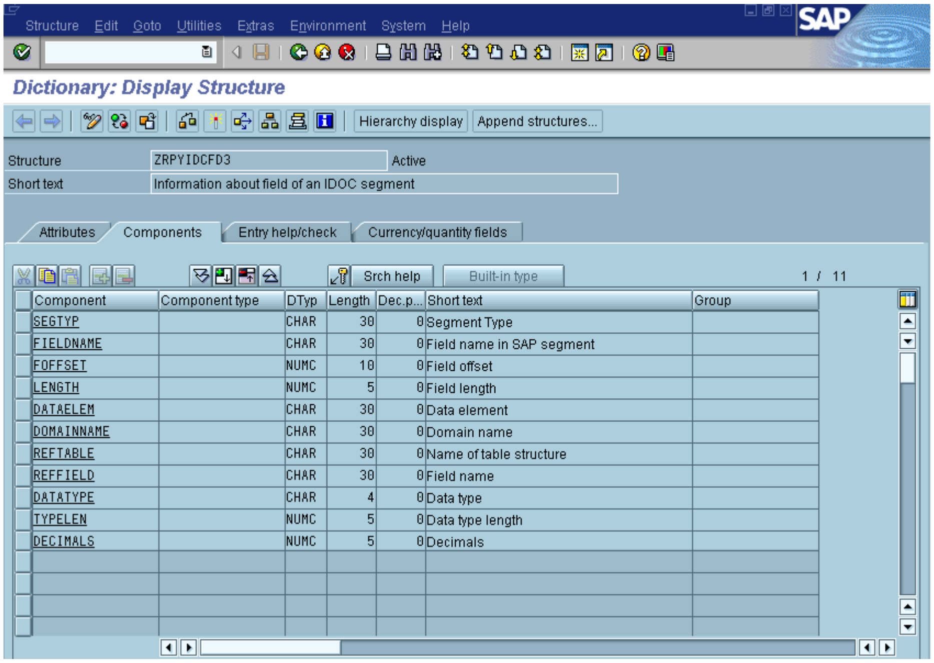
Structure ZRPYIDCSG3 Active
Short text IDOC Segment Header

1 / 8

Component	Component type	DTyp	Length	Dec.p...	Short text	Group
SEGTYP		CHAR	30	0	Segment Type	
SEGNAME		CHAR	30	0	Segment Name	
MUSTFL		CHAR	1	0	Flag: Mandatory Entry	
OCCMIN		NUMC	10	0	Minimum Occurrence	
OCCMAX		NUMC	10	0	Maximum Occurrence	
HLEVEL		NUMC	3	0	Hierarchy Level	
PSEGTYP		CHAR	30	0	Parent Segment Type	
PARFLG		CHAR	1	0	Flag for Parent Type	

Create the ZRPYIDCFD3 Structure Figure 4–11 describes creating the ZRPYIDCFD3 structure.

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



Structure ZRPYIDCFD3 Active
Short text Information about field of an IDOC segment

Component	Component type	DTyp	Length	Dec.p...	Short text	Group
SEGTYP		CHAR	30	0	Segment Type	
FIELDNAME		CHAR	30	0	Field name in SAP segment	
FOFFSET		NUMC	10	0	Field offset	
LENGTH		NUMC	5	0	Field length	
DATAELEM		CHAR	30	0	Data element	
DOMAINNAME		CHAR	30	0	Domain name	
REFTABLE		CHAR	30	0	Name of table structure	
REFFIELD		CHAR	30	0	Field name	
DATATYPE		CHAR	4	0	Data type	
TYPELEN		NUMC	5	0	Data type length	
DECIMALS		NUMC	5	0	Decimals	

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

  Table 4–5 lists object components.

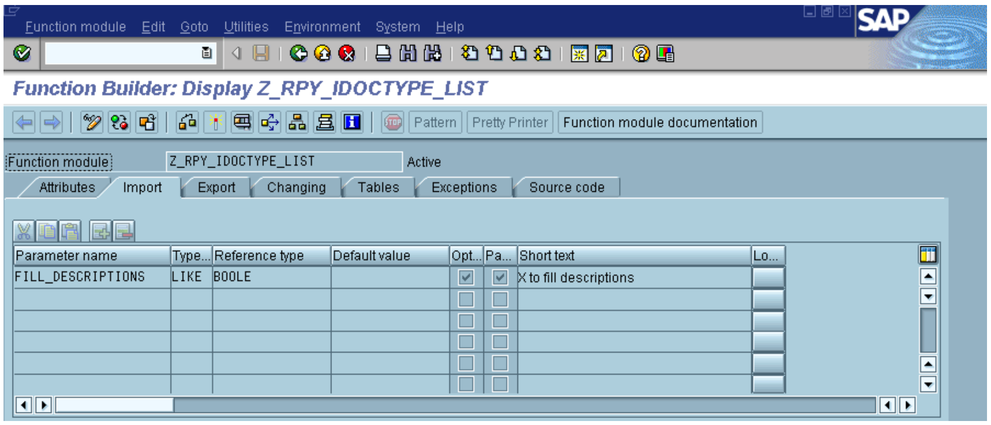
Table 4–5 Object Components

Classification	Function Group:	ZMAS
	Application:	Z
	ShortText:	Retrieve details about all released intermediate documents.
Processing type	Remote Function 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



Function Builder: Display Z_RPY_IDOCTYPE_LIST

Function module: Z_RPY_IDOCTYPE_LIST Active

Parameter name	Type...	Reference type	Default value	Opt...	Pa...	Short text	Lo...
FILL_DESCRIPTIONS	LIKE	BOOLE		✓	✓	X to fill descriptions	

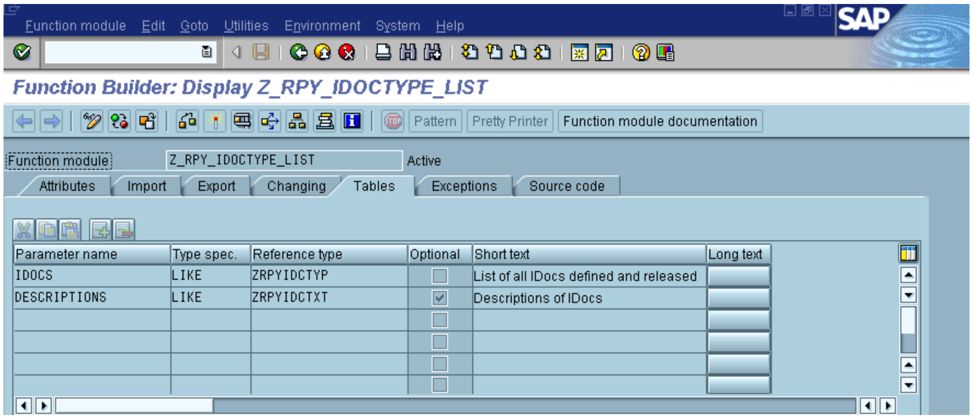
Table 4–6 Function Module Display:Import/Export Parameters Z_RPY_IDOCTYPE_LIST

Import parameter	Reference field	Proposal	Optional	Short Text
FILL_DESCRIPTIONS	BOOLE	X		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



Function Builder: Display Z_RPY_IDOCTYPE_LIST

Function module Z_RPY_IDOCTYPE_LIST Active

Parameter name	Type spec.	Reference type	Optional	Short text	Long text
IDOCS	LIKE	ZRPYIDCTYP	☐	List of all IDocs defined and released	
DESCRIPTIONS	LIKE	ZRPYIDCTXT	✓	Descriptions of IDocs	

Table 4–7 Function Module Display:Import/Export Parameters Z_RPY_IDOCTYPE_LIST

Table Parameters	Ref. Structure	Optional	Short Text
IDOCS	ZRPYIDCTYP		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

  Table 4–8 lists the classification, function group, and ZMAS for the `Z_RPY_IDOCTYPE_READ_DEFN3` function module.

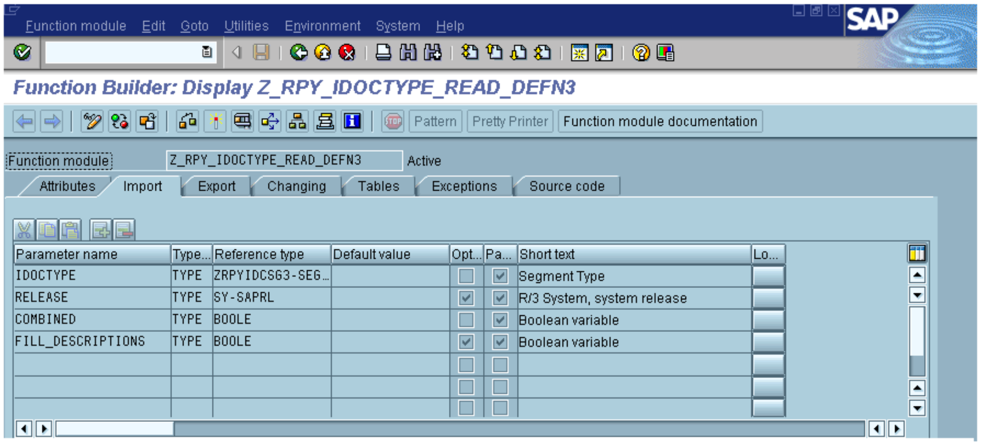
Table 4–8 Z_RPY_IDOCTYPE_READ_DEFN3 function module

Classification	Function Group:	ZMAS
	Application:	Z
	ShortText:	Retrieve details about one intermediate document type.
Processing type	Remote Function Call supported	
	Immediate Start	

- Import/Export Parameter Interface

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

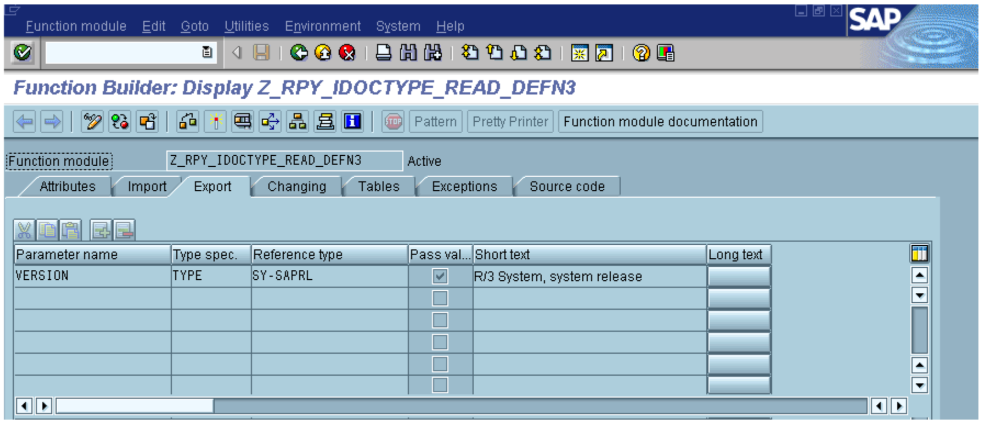
Figure 4–15 Function Module Display:Import/Export Parameters Z_RPY_IDOCTYPE_READ_DEFN2


Function module Edit Goto Utilities Environment System Help
Pattern
Pretty Printer
Function module documentation
Attributes
Import
Export
Changing
Tables
Exceptions
Source code

Function Builder: Display Z_RPY_IDOCTYPE_READ_DEFN3

Function module: Z_RPY_IDOCTYPE_READ_DEFN3 Active

Parameter name	Type...	Reference type	Default value	Opt...	Pa...	Short text	Lo...
IDOCTYPE	TYPE	ZRPYIDCSG3-SEG...		☐	✓	Segment Type	
RELEASE	TYPE	SY-SAPRL		✓	✓	R/3 System, system release	
COMBINED	TYPE	BOOLE		☐	✓	Boolean variable	
FILL_DESCRIPTIONS	TYPE	BOOLE		✓	✓	Boolean variable	
				☐	☐		
				☐	☐		
				☐	☐		


Function module Edit Goto Utilities Environment System Help
Pattern
Pretty Printer
Function module documentation
Attributes
Import
Export
Changing
Tables
Exceptions
Source code

Function Builder: Display Z_RPY_IDOCTYPE_READ_DEFN3

Function module: Z_RPY_IDOCTYPE_READ_DEFN3 Active

Parameter name	Type spec.	Reference type	Pass val...	Short text	Long text
VERSION	TYPE	SY-SAPRL	✓	R/3 System, system release	
			☐		
			☐		
			☐		
			☐		
			☐		

Table 4–9 Function Module Display: Import/Export Parameters Z_RPY_IDOCTYPE_READ_DEFN2

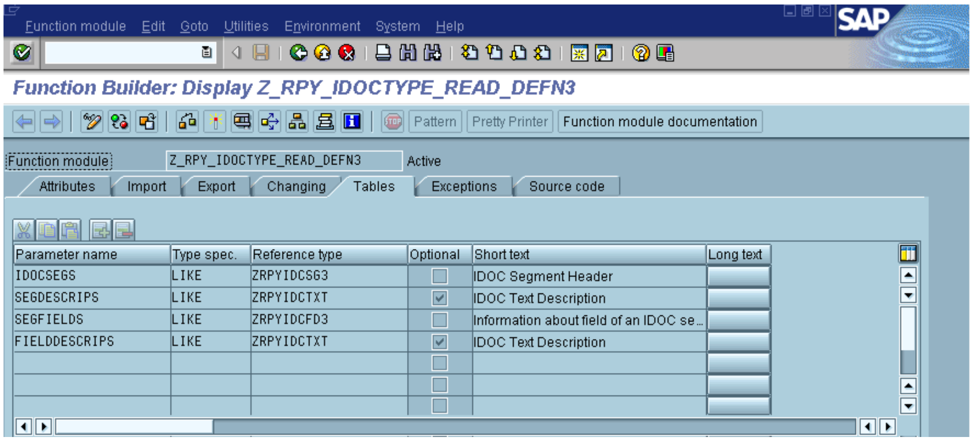
Import parameter	Reference field	Proposal	Optional	Short Text
`IDOCTYPE`	ZRPYIDCSG3-SEGTYP			Segment
`RELEASE`	SY-SAPRL	SY-SAPRL	X	R/3 Systems, system release
`COMBINED`	BOOLE		X	Boolean Variable
`FILL_DESCRIPTIONS`	BOOLE	X	X	Boolean Variable
Export Parameters	**Reference field**			
`VERSION`	SY-SAPRL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


Function module Edit Goto Utilities Environment System Help
Pattern
Pretty Printer
Function module documentation
Attributes
Import
Export
Changing
Tables
Exceptions
Source code

Function Builder: Display Z_RPY_IDOCTYPE_READ_DEFN3

Function module: Z_RPY_IDOCTYPE_READ_DEFN3 Active

Parameter name	Type spec.	Reference type	Optional	Short text	Long text
IDOCSEGS	LIKE	ZRPYIDCSG3	☐	IDOC Segment Header	
SEGDESCRIPS	LIKE	ZRPYIDCTXT	☑	IDOC Text Description	
SEGFIELDS	LIKE	ZRPYIDCFD3	☐	Information about field of an IDOC se...	
FIELDDESCRIPS	LIKE	ZRPYIDCTXT	☑	IDOC Text Description	

Table 4–10 Function Module Display: Import/Export Parameters Z_RPY_IDOCTYPE_READ_DEFN3

Table Parameters	Ref. Structure	Optional	Short Text
IDOCSEGS	ZRPYIDCSG3		IDOCSegment Header
SEGDESCRIPS	ZRPYIDCTXT	X	IDOC Text Description
SEGFIELDS	ZRPYIDCFD3		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 ledidtyp.
   ```

   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 ledidtyp.
     ```

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
- Remote Function Call Configuration
- Optimize Remote Function Call Function Modules
- Enhance Remote Function Call Function Module Remote Browsing
- Clean Your R/3 System

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 Control Broker 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

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 Control Broker 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 Control Broker 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/9.0.2/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

Table 5–3 describes attributes classifications.

Table 5–3 Attributes Classifications

Classification	Function Group:	ZMAS
	Application:	Z
	ShortText:	Retrieve areas for remote callable Remote Function Call functions
Processing type	Remote Function Call supported	
	Immediate Start	

Table Parameters/Exceptions Interface

Table 5–4 describes table parameters.

Table 5–4 Table Parameters

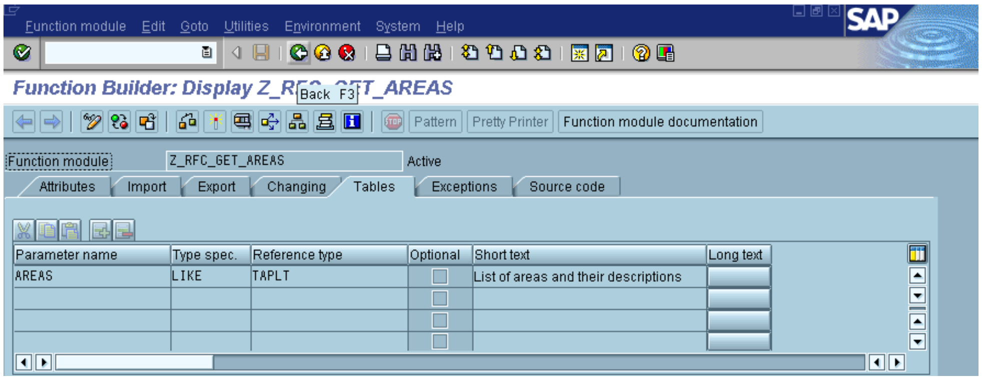
Table Parameters	Ref. Structure	Short Text
AREAS	TAPLT	Lot of areas and descriptions

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



Function module: Z_RFC_GET_AREAS Active

Parameter name	Type spec.	Reference type	Optional	Short text	Long text
AREAS	LIKE	TAPLT	☐	List of areas and their descriptions	
			☐		
			☐		
			☐		

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 Control Broker using **Utilities** > **More Utilities** > **UpLoad/DownLoad** >**UpLoad**.
5. Enter the path and file name, `install_directory/oai/9.0.2/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

Table 5–5 describes attributes definitions.

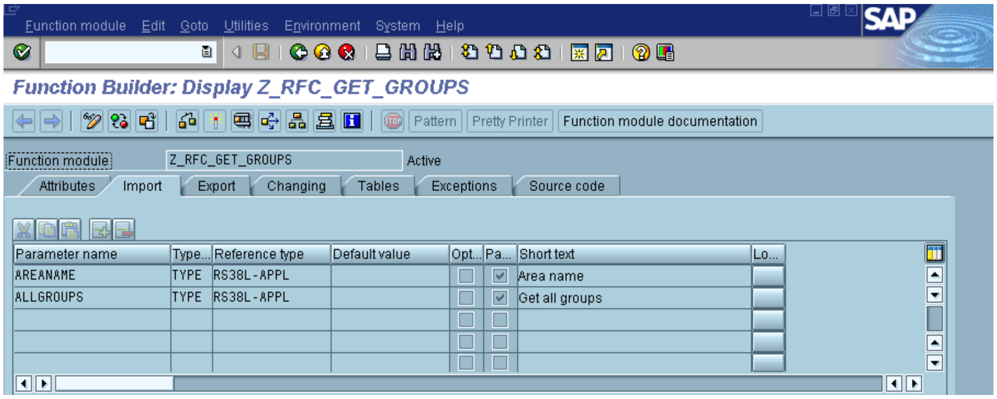
Table 5–5 Attributes Definitions

Classification	**Function Group:**	**ZMAS**
	Application:	Z
	ShortText:	Retrieve groups for remote callable Remote Function Call functions
Processing type	Remote Function 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



Function Builder: Display Z_RFC_GET_GROUPS

Function module Z_RFC_GET_GROUPS Active

Parameter name	Type...	Reference type	Default value	Opt...	Pa...	Short text	Lo...
AREANAME	TYPE	RS38L-APPL		☐	✓	Area name	
ALLGROUPS	TYPE	RS38L-APPL		☐	✓	Get all groups	

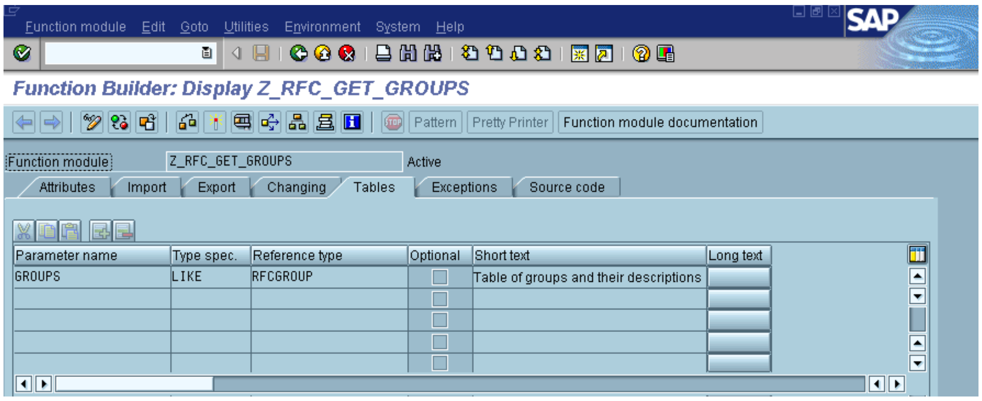
Table 5–6 Import parameter definitions

Import parameter	Reference field	Proposal	Optional	Short Text
AREANAME	RS38L-APPL			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



Function Builder: Display Z_RFC_GET_GROUPS

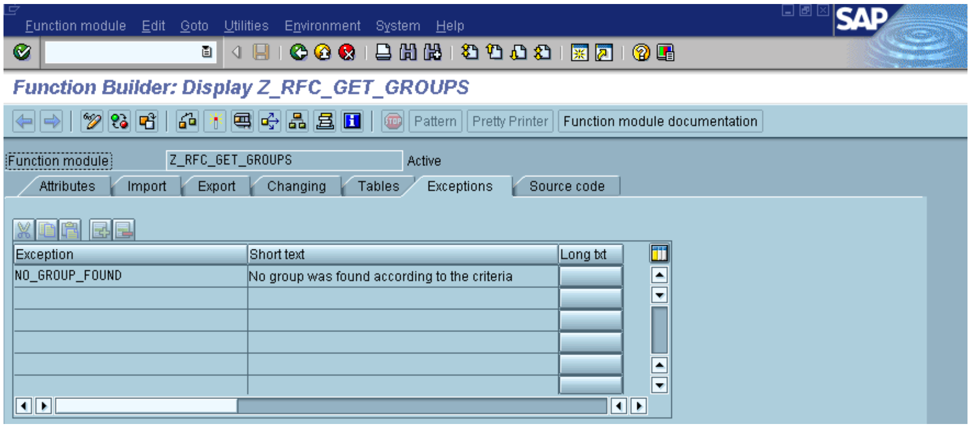
Function module Z_RFC_GET_GROUPS Active

Parameter name	Type spec.	Reference type	Optional	Short text	Long text
GROUPS	LIKE	RFCGROUP	☐	Table of groups and their descriptions	

Table 5–7 Table Parameter definitions

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

Figure 5–4 Remote Function Call Exception Group



Function Builder: Display Z_RFC_GET_GROUPS

Function module Z_RFC_GET_GROUPS Active

Exception	Short text	Long txt
NO_GROUP_FOUND	No group was found according to the criteria	

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 Control Broker 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/9.0.2/sap/rfc_files/groups3x.asc
   ```

   - OR -

   R/3 Version 4.x

   ```
   install_directory/oai/9.0.2/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 *Oracle9iAS 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 Control Broker, 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.

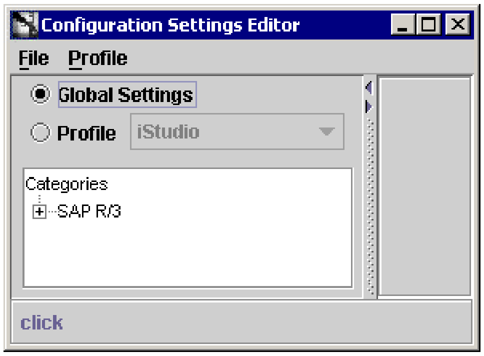
Note: Before using any BAPI interfaces, you must configure the Remote Function Call. BAPI and Remote Function Call share their configuration information.

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



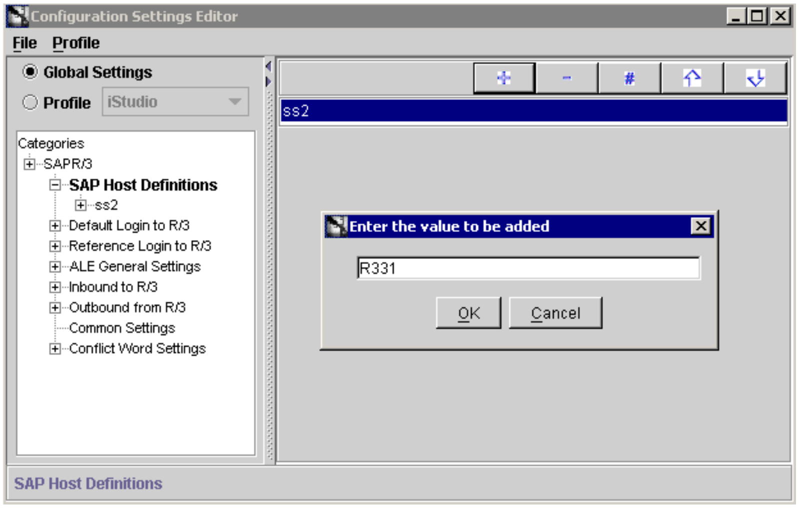
Creating SAP Host Definitions in Global Settings

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



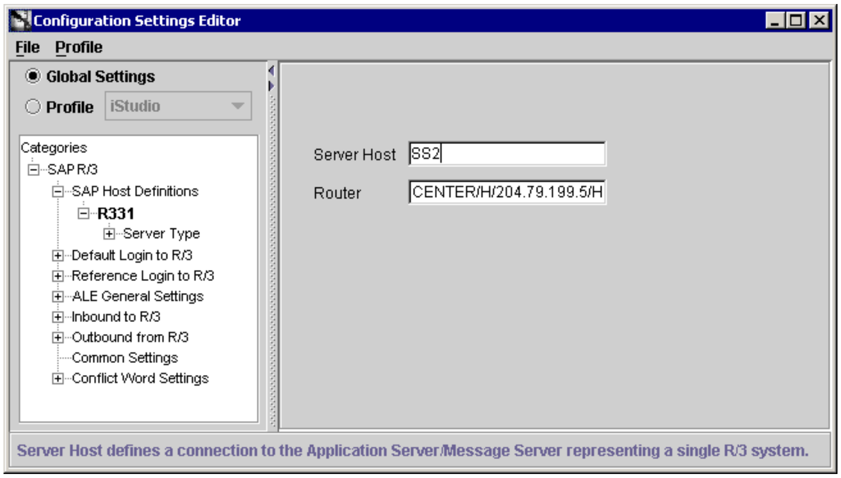
3. Click **PLUS (+)** on the right panel to add a host.
4. Type the name of the new Host definition.

   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



Server Host: SS2

Router: CENTER/H/204.79.199.5/H

Server Host defines a connection to the Application Server/Message Server representing a single R/3 system.

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.

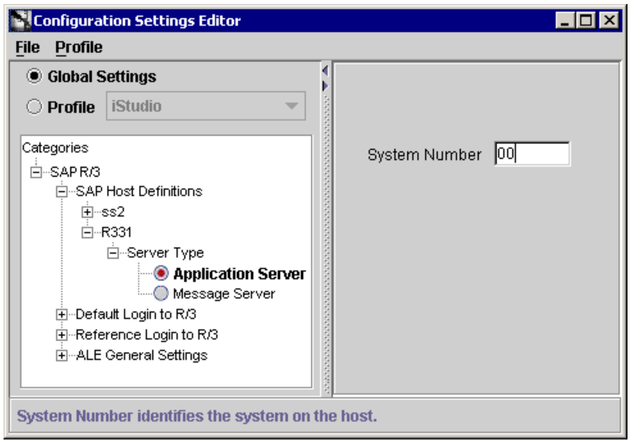
   This is a Destination router to connect to the application server or Message Server, for example `/H/UNICENTER/H/204.79.199.5/H`.

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



System Number 00

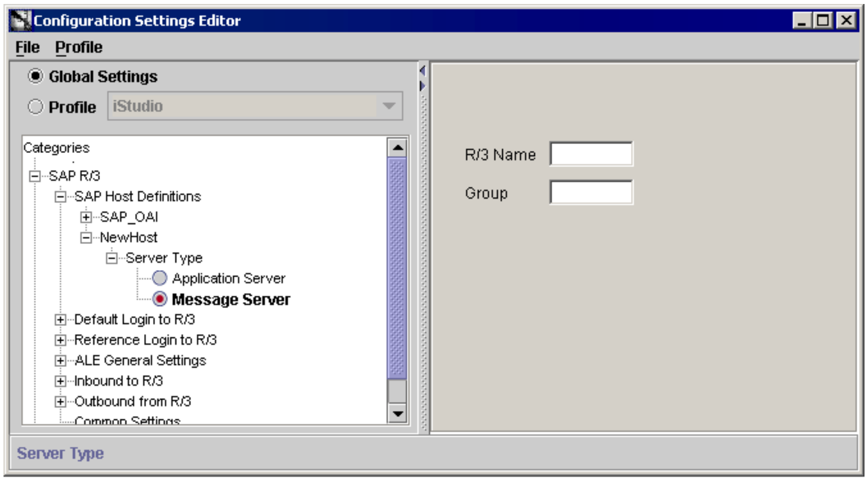
System Number identifies the system on the host.

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:

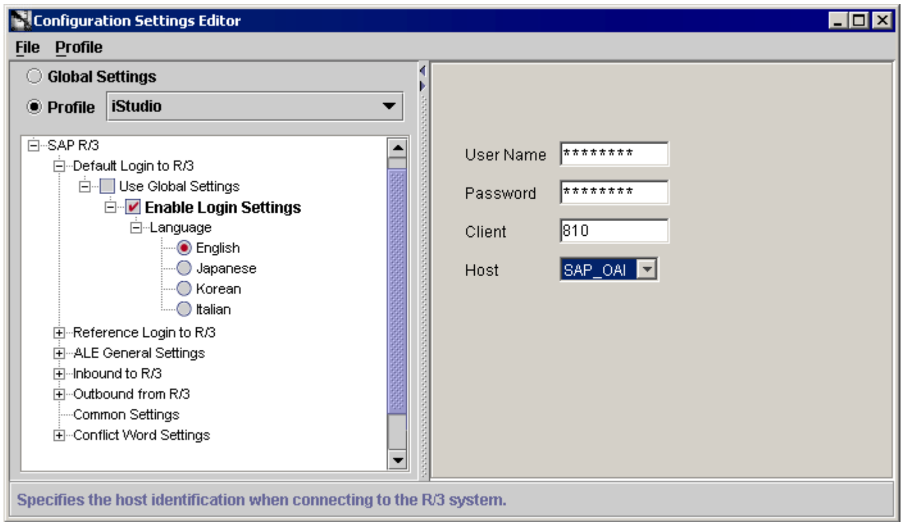
1. Click **Profile** and select **iStudio**.

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



User Name ********
Password ********
Client 810
Host SAP_OAI

Specifies the host identification when connecting to the R/3 system.

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 Oracle9*i*AS 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.

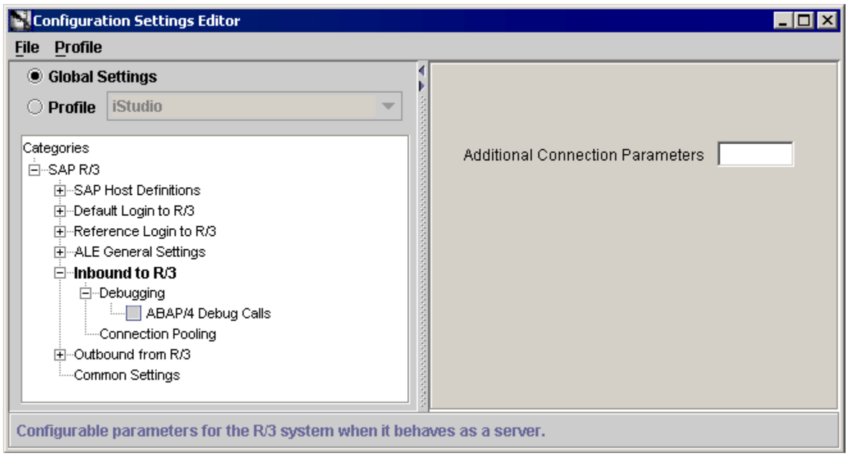
Inbound to R/3

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

1. 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



Configurable parameters for the R/3 system when it behaves as a server.

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

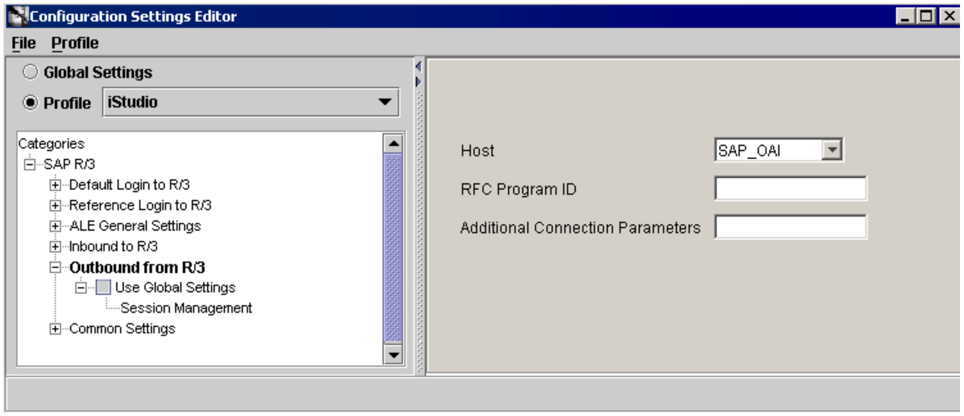
The Outbound from R/3 group contains parameters pertaining to the R/3 system when R/3 is calling other systems through SAP adapter.

1. 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



Host SAP_OAI
RFC Program ID
Additional Connection Parameters

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 Control Broker 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

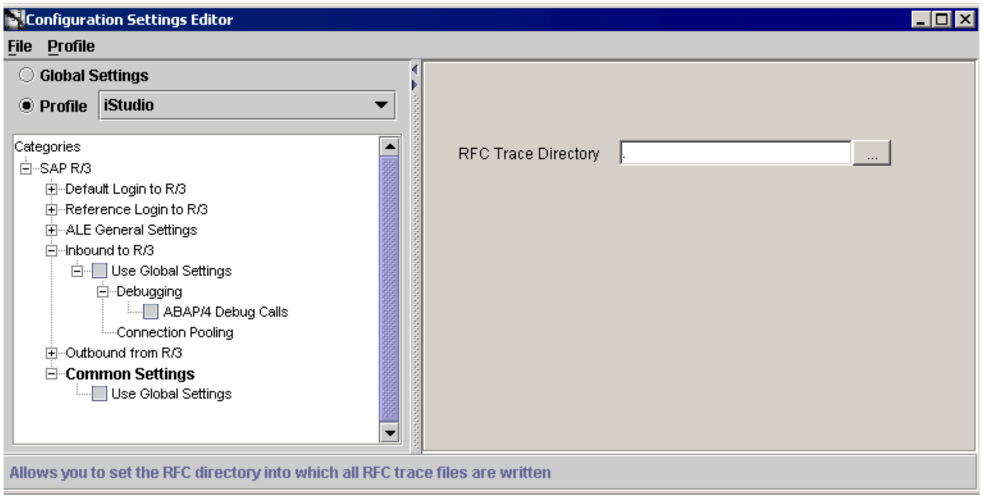
The Common Remote Function Call Settings group allows you to set the Remote Function Call directory into which all Remote Function Call trace files are written. For example, all `dev_rfc.trc`, all `rfc .trc` files.

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



Allows you to set the RFC directory into which all RFC trace files are written

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

A

advanced queuing adapter
- installation, 2-2

ALE
- application acting as a server, 4-6
- define base logical system, 4-12
- define partner logical system, 4-16
- download intermediate documents manually, 4-23
- enhance remote browsing, 4-25
- exploring IDOC types, 4-22
- exploring intermediate document types, 4-22
- general settings, 4-9
- logical system, 4-3
- queue directory, cache directory, 4-7
- r/3 configuration, 4-12
- remote browsing, browsing function module, 4-26
- sending material master intermediate document, 4-14
- transactional port, 4-15
- writing a client with oracle9ias interconnect, 4-6

ale
- frequently used transactions, 4-2
- terminology, 4-3

application parameter, 2-5

attributes
- importing, 3-8

C

common settings, 6-11

configuration, 2-4
- adapter.ini initialization parameter file, 2-6
- agent connection parameters, 2-7
- directories, 2-5
- executable files, 2-4
- files, 2-4
- hub.ini, 2-6
- sap adapter parameters, 2-11

configuration editor, 6-2
- exiting, 6-12

creating
- event, 3-22, 3-32
- invoked procedure, 3-27, 3-37
- partner profile, 4-17
- procedure, 3-3, 3-17
- rfc published event, 3-42
- subscribed ale event, 3-12

customer distribution model, 4-21

D

default login, 6-6
- r/3, 5-3

default Login to R/3, 6-6

E

event
- create published ALE, 3-32
- creating rfc published, 3-42
- creating rfc subscribed, 3-22
- creating subscribed ale, 3-12

exception fields, 3-2

F

function group, 4-26, 4-31, 5-6
- creating, 5-6

function groups
- erasing, 5-14

function module, 4-32
- creating, 5-7, 5-9
- upload source code, 5-12
- upload the source code, 5-8

function modules
- erasing, 5-14

H

host ID, 6-8

I

importing
- attributes, 3-8

inbound
- intermediate documents, 4-11
- to R/3, 6-9

inbound to r/3, 6-9

installation, 2-2
- preinstallation, 2-2
- tasks, 2-2

intermediate documents
- browsing remote modules, 4-26
- inbound, 4-11
- outbound, 4-12
- type, 4-4
- without connecting to R/3, 4-8

L

login
- sap, 3-8

M

message type, 4-5

O

outbound, 3-27
- intermediate documents, 4-12
- maintain parameters, 4-19

outbound from R/3
- host, 6-10

outbound to r/3, 6-10

P

parameters
- maintain outbound, 4-19

partner profile
- creating, 4-17

platforms
- supported, 1-2

preinstallation, 2-2

procedure
- creating, 3-3
- creating an invoked, 3-27
- creating implemented, 3-17
- invoked rfc, 3-37

R

r/3
- clean your system, 5-14

reference login, 6-8

reference login to R/3, 6-8

remote ALE browsing, 4-25

remote function call
- calling from sap to r/3, 5-3
- configuration, 5-3
- enhance module remote browsing, 5-6
- introduction, 5-2
- optimize modules, 5-4
- set global data, 5-13

S

sap
- calling from r/3 to sap adapter, 5-4
- installation, 2-2
- interaction with r/3, 5-2
- outbound, 3-27

starting, 2-12
what is, 1-2
sap adapter
configuration, 2-4
sap host definitions
creating in global settings, 6-3
software
required, 1-2
supported platforms, 1-2

T

table
erasing, 5-15