Siebel Connector for SAP R/3

Version 7.7, Rev. A
June 2004

Siebel Systems, Inc., 2207 Bridgepointe Parkway, San Mateo, CA 94404

Contents

Chapter 8: EAI Queue

Chapter 9: Upgrading from v6.x to v7.x

Chapter 10: Modifying Siebel Interfaces After an SAP Upgrade

Appendix A: Data Types

Appendix B: SAP Field Mappings

Appendix C: SAP Code Page Mappings

Appendix D: Troubleshooting Siebel Connector for SAP R/3

Appendix E: Creating Integration Touch Points

Index

1 What's New in This Release

What's New in Siebel Connector for SAP R/3, Version 7.7, Rev. A

Table 1 lists changes described in this version of the documentation to support Release 7.7 of the software.

Table 1. What's New in Siebel Connector for SAP R/3, Version 7.7, Rev. A

Topic	Description
"Generating Partner Profiles" on page 31	Added note after Step 8.
"Manually Creating Partner Profiles" on page 32	Step 9: ■ Added note after Output Mode. ■ Added instructions for entering data into Seg. release in IDoc type field.
"Using the Business Service Simulator" on page 103	Added Step 4.
"The Send Transaction Service" on page 150	Certain Send Transaction Service configuration options in Tables 38 and 39 are available as user properties only, not component parameters.

What's New in Siebel Connector for SAP R/3, Version 7.7

Table 2 lists changes described in this version of the documentation to support Release 7.7 of the software.

Table 2. What's New in Siebel Connector for SAP R/3, Version 7.7

Topic	Description
"SAP Code Page Configuration" on page 36	Siebel 7.7 supports SAP Latin-2 and Cyrillic.
Chapter 4, "Standard Integrations"	Siebel 7.7 supports SAP 4.6C out of the box. The new mappings use 4.6C BAPI and IDOC integration objects.
"Modifying Standard Integration Interfaces" on page 72	Business data flow values reflect 4.6C BAPI and IDOC integration objects.
Chapter 10, "Modifying Siebel Interfaces After an SAP Upgrade"	Siebel interfaces to SAP R/3 can be affected by a SAP upgrade.

Table 2. What's New in Siebel Connector for SAP R/3, Version 7.7

Topic	Description
Appendix B, "SAP Field Mappings"	New mappings use 4.6C BAPI and IDOC integration objects.
Dropped appendix, "SAP R/3 v3.1H Mappings"	SAP R/3 version 3.1H is no longer supported.

2 About SAP R/3

The Siebel Connector for SAP R/3 provides integration between Siebel applications and SAP R/3. This integration offers capabilities designed to meet your sales, marketing, and customer service requirements. The Siebel Connector for SAP R/3 extends Siebel applications to integrate with back office data and processes.

Using the Siebel Connector for SAP R/3

The Siebel Connector for SAP R/3 supports both synchronous and asynchronous transactions across application boundaries. The resulting consistency of data provides efficient coordination between front and back-office operations. For example, sales and service professionals can enter sales orders in Siebel applications and receive real-time feedback on inventory availability from the SAP R/3 database. The sales or service professional can then fulfill the sales order using SAP's Sales and Distribution module in the back office, without ever leaving the Siebel application interface.

You can approach your work with the Siebel Connector for SAP R/3 in the following ways, depending upon your specific needs:

- Use the Siebel standard integrations for SAP as provided.
- Modify the standard integration to suit your business needs.
- Create customized integrations to support your own business needs.

You can learn how to use the standard integrations, as is, from this guide. You can also find guidance on modifying these Standard Integrations. Some information on creating customized integrations is included, but you also need to consult *Overview: Siebel eBusiness Application Integration Volume I*.

Features and Capabilities

The Siebel Connector for SAP R/3 comprises two kinds of integrations, which are based on different interfaces supported by the SAP R/3 application:

- BAPI-based Integration, which uses the SAP Business API to integrate Siebel applications with SAP R/3.

  For example, using BAPI-based integration, you can check an account's sales order history and you can submit sales orders in real time to SAP. You can submit and query sales orders individually or in batches by way of the Siebel Server.

- IDOC-based integration, which uses Siebel applications to integrate Siebel data with SAP R/3 data. IDOC Integration has been implemented using SAP's Application Link Enabling (ALE) interface.

  For example, using IDOC-based integration, you can create Siebel Accounts from SAP Customers and Siebel Products from SAP Materials.

Terminology

Siebel applications and SAP use different terminology for the same objects. This guide always uses the Siebel application term when referring to Siebel software and data elements. (See Table 3.) It always uses the SAP term when referring to SAP software and data elements. When a description requires both terms, the Siebel application term occurs first and the SAP term occurs second. Example: Account/Customer.

Table 3. Siebel Applications and SAP Terminology

Siebel Applications Term	SAP Term
Account	Customer
Contact	Contact
Sales Order	Sales Order
Product	Material

This guide uses many abbreviations and specialized terms. Table 4 provides a list of many of these terms.

Table 4. Siebel Connector for SAP R/3 Terminology

Term	Meaning
ABAP	SAP programming language, Advanced Business Application Programming.
ALE	Application Link Enabling. This is SAP's technology for transmitting IDOC data containers to and from external applications using Transactional RFC.
BAPI	Business Application Programming Interface. This is the function call interface to SAP's business object methods. These function calls are often referred to as BAPIs. The terms BAPI and RFC are very similar and are often used in place of one another.
Dialog Process	An SAP process that handles immediate user activity and ABAP execution.
EAI	eBusiness Application Integration. A toolkit of Siebel-supplied utilities you can use to create data adapters between Siebel applications and external applications.
IDOC	Intermediate Document. This is the hierarchical data container used by ALE.
RFC	Remote Function Call. This is the technology used by SAP to allow external applications to call ABAP functions defined in SAP and allow SAP to call functions defined in external applications. The terms RFC and BAPI are very similar and are often used in place of one another.

Table 4. Siebel Connector for SAP R/3 Terminology

Term	Meaning
Synchronous RFC	Two-way RFC call into or out of SAP.
tRFC	Transactional RFC. This is a one-way RFC call into or out of SAP with guaranteed delivery.

Table 5 contains a list of Siebel Connector for SAP R/3 business services and their common names in this document.

Table 5. Business Service Names

Name in Document	Business Service Name
BAPI Adapter	EAI SAP BAPI Adapter
BAPI Workflow Service	EAI SAP BAPI Workflow Service
IDOC Adapter	EAI SAP IDOC Adapter
IDOC MQ AMI Adapter	EAI SAP IDOC MQ AMI Adapter
IDOC MQ AMI Workflow Processor	EAI SAP IDOC MQ AMI Workflow Processor
IDOC Workflow Processor	EAI SAP IDOC Workflow Processor
IDOC Workflow Service	EAI SAP IDOC Workflow Service
Process Transaction Service	EAI SAP Process Transaction Service
Send Transaction Service	EAI SAP Send Transaction Service
tRFC BAPI Adapter	EAI SAP BAPI Adapter (tRFC)
tRFC BAPI Receiver	EAI SAP BAPI Receiver (tRFC)

Standard Integrations

The Siebel Connector for SAP R/3 includes prebuilt standard integrations, which may be used as-is, or tailored to meet specific SAP interface business requirements. These standard integrations include Siebel screens, views, applets, workflows, Business Services, and Business Service data maps. Together these elements provide a link between Siebel eBusiness applications and SAP R/3. Table 6 lists the standard integrations.

Table 6. Standard Integration Summary

Data Class	Objects Integrated
Master Data	Accounts
	Contacts
	Products
Transaction Data	Sales Orders

The Business Service data maps provide preconfigured mappings between equivalent entities in both applications. Preconfigured mappings simplify configuration and consist of the most commonly required data, and take into account the complicated entity relationships contained in both applications. These mappings provide a foundation for any desired customization work and are modifiable.

SAP Master Data

Table 7 lists the standard integrations provided for SAP Master Data.

Table 7. Standard Integrations for SAP Master Data

Integration Data	Description
Account to Customer	The Account to Customer Standard Integration uses synchronous BAPI communication as well as SAP's ALE communication services to create and update SAP customer information from Siebel Accounts. Using Siebel views you can enter SAP customer information directly into the Siebel application, including customer sales area specific data.
Customer to Account	The Customer to Account Standard Integration uses SAP's ALE communication services to accept SAP Customer IDOCs and transform them into Siebel Accounts. This provides not only the initial load of data between SAP and Siebel applications, but also the periodic high volume synchronization of data.
Material to Product	The Material to Product Standard Integration uses ALE to accept Material IDOCs from SAP R/3 and transform them into Siebel Products. Also transformed are product division and sales area information. The Material to Product standard integration can provide an initial load of data from SAP to the Siebel application, as well as support periodic high volume synchronization of data.

SAP Transaction Data

Table 8 lists the standard integrations provided for SAP Transaction Data. For more information, read Chapter 4, "Standard Integrations."

Table 8. Standard Integrations for SAP Transaction Data

Integration Name	Description
Account Order History	Use this to view and import sales orders for a given account from SAP into a Siebel application in real time. Imported sales orders are also updated automatically with the latest SAP data.
Siebel Sales Order to SAP Sales Order	Use this to create sales orders within Siebel applications and submit those sales orders to SAP. A real-time synchronous BAPI interface provides an immediate response from SAP upon the submission of the sales order. SAP checks both credit and product availability. Pricing may be done by either SAP or Siebel applications. The status of these sales orders may be obtained in real time or in the background after they are created.
Sales Order Updates	The Sales Order Updates standard integration automatically updates the Sales Order in the Siebel database when it changes in SAP. This integration uses SAP's ALE interface, which allow SAP to trigger the status update by sending a sales order change IDOC message to the Siebel applications.
Quote to Sales Order	Use this process to create Siebel and SAP orders from a single quote. Siebel quotes are first converted to a Siebel order. The Siebel order can then be submitted to SAP. A real-time synchronous BAPI interface provides an immediate response from SAP upon the submission of the sales order. SAP checks both credit and product availability. Pricing may be done by either SAP or Siebel applications.

Management of End-to-End Data Transfers

The Siebel Connector for SAP R/3 provides the infrastructure to manage data transfers between SAP and Siebel applications. The Siebel Server runs BAPI and ALE/IDOC Adapters that exchange data between Siebel objects and the respective SAP interfaces. These adapters eliminate the need for coding to SAP's proprietary transports. IDOCs are processed by Siebel Server components, which deliver the data to the Siebel base tables. These Siebel Server processes may be administered and monitored remotely through the Siebel Server Manager process.

Mobile User Support

Incoming transactions from SAP to the Siebel application are routed to mobile users according to the visibility rules defined in the Siebel repository. In addition, Siebel mobile users' transactions are queued in their local databases. When these mobile users synchronize with the Siebel Server, their transactions are placed in the server queue and subsequently completed with SAP through the interfaces described in this guide.

Architectural Overview

The Siebel EAI Architecture provides the foundation for the development, modification and execution of both standard integrations and custom integrations. This section provides an overview of this architecture. For more information, read *Overview: Siebel eBusiness Application Integration Volume I*.

Business Data Flow

Each Standard Integration or Custom Integration is based on the creation of Business Data Flows. A Business Data Flow controls the transformation of an SAP data object to a Siebel data object and a Siebel data object to an SAP data object. Figure 1 illustrates inbound and outbound Business Data Flows. The outbound dataflow starts when the EAI Siebel Adapter extracts a record from the database, processes it through a Business Service data map, and then uses a BAPI or an IDOC to pass the record to SAP. SAP processes the record, and returns it through a BAPI or an IDOC to the inbound dataflow, which uses a Business Service data map, and the EAI Siebel Adapter.

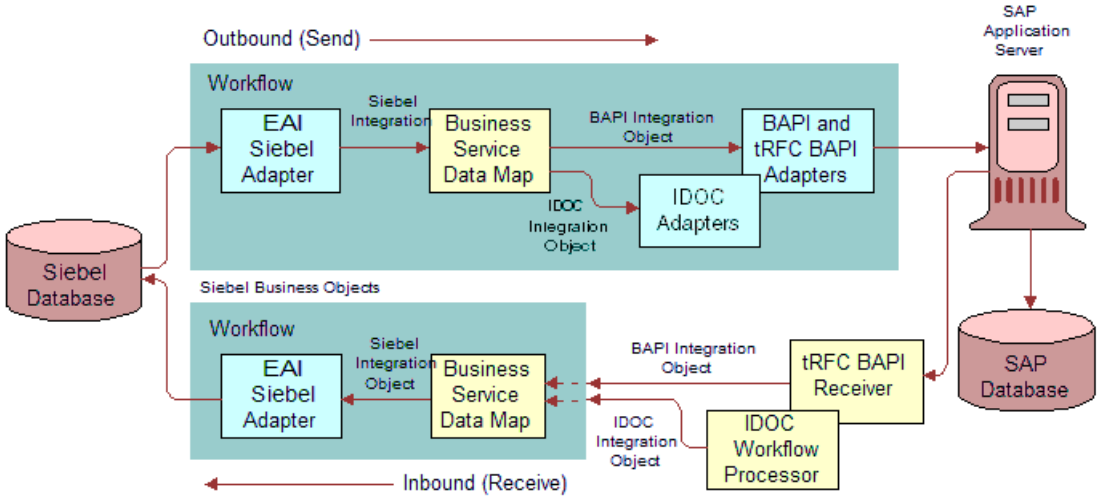


Figure 1. General Architecture of the Integration Process

As Figure 1 shows there are two types of Business Data Flows possible, Outbound to SAP (Send) and Inbound from SAP (Receive). The processing flow for each is largely contained within a Siebel Workflow.

Outbound Business Data Flows generally contain a call to the EAI Siebel Adapter to first extract data from the Siebel database corresponding to a Siebel Business Object. This data is then used to populate a corresponding Siebel Integration Object. The Siebel Integration Object is passed to a Business Service whose job is to transform the Siebel Integration Object structure into an SAP Integration Object structure. The data transformation within the Business Service Data Map can be written using Siebel eScript or can be done through the use of Siebel Data Mapper. SAP integration objects can represent either an IDOC structure in SAP or a BAPI function interface. The IDOC Adapter converts the IDOC data into a special format and then passes that data to the BAPI Adapter. The BAPI Adapter then interfaces to the SAP Application Server directly. All data passing into SAP passes through the BAPI Adapter.

Inbound Business Data Flows must start with the tRFC BAPI Receiver Component. The tRFC BAPI Receiver runs in the background continuously waiting for data from SAP. This data can be in the form of an IDOC or RFC function call. When the tRFC BAPI Receiver receives an IDOC, it calls the IDOC Workflow Processor to convert the raw SAP IDOC data into an IDOC Integration Object. The IDOC Workflow Processor then invokes a workflow to process the data further. When the tRFC BAPI Receiver receives an RFC call, it creates a BAPI Integration Object that it sends to a workflow for processing. The workflow typically contains a Business Service Data Map to transform the data into a Siebel Integration Object. The Siebel Integration Object is then passed to the EAI Siebel Adapter where it can be processed as a business object into the Siebel database.

Workflow Integration

Siebel Workflow is the center of the Business Data Flow. Workflows control the flow and transformation of data into and out of the Siebel applications. You create them using a graphical user interface provided within the Siebel applications called the Workflow Designer. Workflow provides many capabilities beyond what is described in this guide. For more information about Siebel Workflow, read *Siebel Business Process Designer Administration Guide*.

Integration Objects

Integration Objects are the data containers used within the Workflow environment. They represent the data structure of either a Siebel Business Object or an external application's data object. You can create Integration Objects with the Integration Object Wizard provided in Siebel Tools. The Integration Object wizard can create Siebel Integration Objects from Siebel Business Objects, IDOC Integration Objects from SAP IDOCs, and BAPI Integration Objects from SAP BAPI Interfaces. This document describes how to create IDOC and BAPI Integration Objects with the wizard. For more information on the Integration Object Wizard, read *Integration Platform Technologies: Siebel eBusiness Application Integration Volume II*.

Business Services

Business services execute predefined or custom actions in a workflow process. Examples of business services include the Siebel Adapter, BAPI Adapter, IDOC Workflow Processor, Business Service Data Map, and tRFC BAPI Receiver. These business services act on Integration Objects passed to them. They perform such functions as interfacing to the Siebel database, interfacing to SAP, or transforming one integration object into another. Siebel Systems, Inc., provides many business services but you can also create your own.

Although business services can be used to perform many different functions, they all have a standard interface. Business services have object-like qualities, such as methods, method arguments, and user properties. These elements define how a business service can be used. Business services are defined in Siebel Tools. This guide describes those business services used to interface to SAP. For more information on business services in general, read *Integration Platform Technologies: Siebel eBusiness Application Integration Volume II*.

Server Components

Some Business Services run within the context of Server Components. Two such components are the tRFC BAPI Receiver Component (BAPIRcvr) and the Business Integration Manager (BusIntMgr). For information on other components that control the use of the EAI Queue, read Chapter 8, "EAI Queue."

tRFC BAPI Receiver Component

The BAPIRcvr Server Component executes as a background task that calls a method of the tRFC BAPI Receiver Service repeatedly to look for data SAP may be sending. Component parameters may be set prior to the start of the server to control processing. Component parameters may have the same names as the underlying Business Service User Properties or Method Arguments, providing many different options for configuration. The relationship between User Properties, Method Arguments and Component Parameters for SAP Components and Business Services is defined more fully later in this document. For more information on Server Components, read *Integration Platform Technologies: Siebel eBusiness Application Integration Volume II* and *Siebel System Administration Guide*.

Business Integration Manager Component

The Business Integration Manager is a server component that processes requests from Siebel applications in real time. The BAPI Adapter Business Service and workflow usually execute within the context of the Business Integration Manager Server Component. In this guide, Business Integration Manager is covered only with respect to the execution of the BAPI Adapter. For more information about the Business Integration Manager, read *Overview: Siebel eBusiness Application Integration Volume I*.

3 Installation and Configuration

This chapter covers the installation and configuration of the Siebel Connector for SAP R/3.

Installation Overview

The following sections describe the basic installation and configuration process for executing standard integrations and their underlying business services. It may be necessary to perform additional configuration within SAP or the Siebel application to support specific customer requirements. For additional information on business services configurations, read Chapter 5, "Customizing Integrations."

NOTE: Before using the Siebel Connector for SAP R/3 make sure you have installed `librfc32.dll` version 4.6C, which is part of the SAP RFC Software Development Kit (SDK). The RFC SDK can be installed during installation of the SAP GUI by enabling the Development Tools check box.

The following list summarizes the necessary steps for installation and configuration of the SAP Connector. To install the SAP Connector infrastructure follow Step 1 through Step 5. Step 7 and Step 8 are required for use of the SAP Connector standard integrations.

1. Modify the SAP configuration file (read "Modifying SAP Configuration Files" on page 19).
2. Modify Siebel configuration files such as `siebel.cfg`, `uagent.cfg`, and `tools.cfg` (read "Modifying Siebel Configuration Files" on page 23).
3. Configure ALE/IDOC (read "Configuring ALE/IDOC Connectivity" on page 26).
4. Configure components. Enable component groups and components, and set parameters (read "Component Configuration" on page 34).
5. Check connectivity (read "Checking Connectivity" on page 38).
6. Configure for MQSeries (if you desire to use MQSeries) (read "Configuring the SAP Connector for Use with IBM MQSeries" on page 40).
7. Configure SAP for standard integrations (read "SAP Configuration for Standard Integrations" on page 45).
8. Configure the Siebel application for standard integrations (read "Siebel Configuration for Standard Integrations" on page 47).
9. Activate workflows (read "Activating Workflows" on page 54).

Modifying SAP Configuration Files

This section describes the SAP configuration files you must modify as part of this installation.

Modifying the saprfc.ini File

The saprfc.ini file contains information that Siebel applications use to connect with SAP. Depending upon your installation and configuration, you may have saprfc.ini files in the following folders: siebsrvr/bin, tools/bin, and seaw/bin. The siebsrvr/bin folder is used when SAP business services execute in the Siebel server. The tools/bin folder is used when SAP integration object wizards are executed in Siebel Tools. The seaw/bin folder is used when SAP business services execute from the Siebel Mobile Web Client. SAP business services execute in the client only during execution of the Workflow Simulator and EAI Business Service Simulator.

You may modify and use these files, or you may elect to use an already existing file you may have set up for your SAP R/3 installation. In any case, you must set the RFC_INI environment variable to point to your saprfc.ini file. This must be set on every Siebel installation that executes the server or Tools applications. In a test environment, you also need to set this environment variable on any machine on which you are executing the Workflow Simulator or EAI Business Service Simulator in the Siebel Mobile Web Client.

The saprfc.ini file contains a series of SAP Destination definitions. Each definition contains the connection information necessary for an external application to connect to SAP in some manner. Each entry begins with the name of the Destination: DEST=*name*. The name in the destination entry is important, as it is this name that must be entered in the Siebel application so that the Siebel Business Services can retrieve the connection information in this entry.

There are two types of entries in the file that are important for connection to SAP. The first is indicated by a TYPE=A entry in the saprfc.ini file destination entry. This type of destination entry indicates that the external application (Siebel eBusiness) connects as a client to the SAP R/3 Application Server. The BAPI adapter uses this type of entry. The second destination type is indicated by a TYPE=R entry in the saprfc.ini file destination entry. This type of destination entry indicates that the external application (Siebel eBusiness) connects as a server to the SAP R/3 Application Server. The tRFC BAPI Receiver uses this type of entry.

To create a destination entry for the BAPI adapter

1. Open the saprfc.ini file in a text editor.
2. Search for the text TYPE=A to find the sample TYPE=A entry.

   This entry has the following required lines:

   ```
   DEST=DEST_NAME
   TYPE=A
   AHOST=HOST_NAME
   SYSNR=XX
   ```

3. Edit DEST= to replace DEST_NAME with your own destination name.

   Any name can be used, but do not use spaces. You use this name as the destination in the SAPRfcConnectString component parameters for the Business Integration Manager component. For more information, read "Setting Server Component Parameters" on page 35.

4. Edit ASHOST= to replace HOST_NAME with your SAP application server host name.

5 Edit SYSNR= to replace XX with your SAP system number.

For example your entry might now look something like this:

```
DEST=DEV_Outbound
TYPE=A
ASHOST=devserver
SYSNR=00
RFC_TRACE=0
ABAP_DEBUG=0
USE_SAPGUI=0
```

6 Save the file and close the editor.

7 Be sure to set the RFC_INI environment variable to point to the saprfc.ini file.

8 To define connections to other SAP implementations use additional TYPE=A entries.

For load balancing, you may use TYPE=B entries instead of TYPE=A.

```
A TYPE=B entry contains the following values:
DEST=DEST_NAME
TYPE=B
R3NAME=Name of the R/3 system. This is optional. The default is the destination.
MSHOST=Host name of the message server
GROUP=Application servers group name. This is optional. The default is PUBLIC.
```

Refer to your SAP documentation for more information on this type of entry.

To create a destination entry for the tRFC BAPI receiver

1 Open the saprfc.ini file in the text editor.

2 Search for the text TYPE=R to find the sample TYPE=R entry.

This entry has the following required lines:

```
DEST=DEST_NAME
TYPE=R
PROGID=PROGRAM_ID
GWHOST=HOST_NAME
GWSERV=sapgwXX
```

3 Edit the DEST= line to replace the DEST_NAME with your own destination name.

Any name can be used, but do not use spaces. You use this name as the value of the SAPRfcDestEntry component parameter for the tRFC BAPI Receiver component.

4 Edit the PROGID= line to replace PROGRAM_ID with the your program ID.

This can be any name you decide upon, but it must be defined within SAP as part of the RFC Destination defined in SAP transaction sm59; read "Creating RFC Destinations" on page 28.

5 Edit the GWHOST= line to replace HOST_NAME with the SAP gateway server host name.

6 Edit the GWSERV= line to replace the sapgwXX with your SAP gateway server name.

Usually it is in this form: `sapgw` followed by the two-digit system number. For example, your entry might now look something like this:

```
DEST=DEV_Inbound
TYPE=R
PROGID=myprogramid
GWHOST=devserver
GWSERV=sapgw00
RFC_TRACE=0
```

For debugging purposes you may want to set the values of RFC_TRACE or ABAP_DEBUG. For more information, read "Troubleshooting SAP Connection Problems" on page 181.

Setting the RFC_INI Environment Variable

You need to set the RFC_INI environment variable in Windows NT so that Siebel Tools or the Siebel Server can access the `saprfc.ini` file. Siebel applications open this file prior to connecting to SAP.

To set the environment variable

1 Click the Windows Start button, and then choose Settings > Control Panel > System.

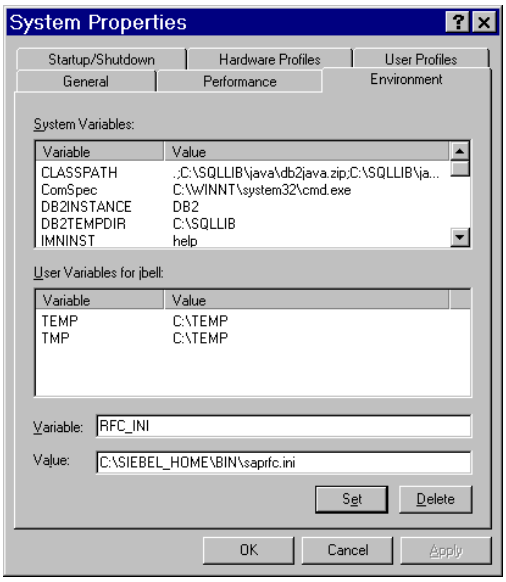
The System Properties window opens.

2 Navigate to the Environment fields.

- **a** In Windows NT, click the Environment tab.
- **b** In Windows 2000, click the Advanced tab, and then click Environment Variables.

3 In the System Variable field, enter RFC_INI.

The following figure shows the Environment tab with the existing System variables and User Variables, and the entry of the new RFC_INI variable.



4 In the Value field, enter `%SIEBEL_HOME%\BIN\saprfc.ini`, where `%SIEBEL_HOME%` is the root directory of your Siebel installation.

5 Click Set.

6 Click OK.

7 Restart Windows.

Modifying Siebel Configuration Files

Siebel configuration files are used by the Siebel Tools and Siebel Mobile Web Client applications. The Siebel Tools configuration file (`tools.cfg`) is located in the `tools/bin/`*`language`* folder. If you intend to customize integrations, you must modify this configuration file. The Siebel Client configuration files are located in the `siebsrvr/bin/`*`language`* folder. The names of these files differ with the client application you have purchased, such as `siebel.cfg` or `uagent.cfg`. These files are used only by the Siebel Mobile Web Client running in disconnected mode.

Your SAP system administrator must create a CPIC user for your connection to SAP. You need this user name and password for this procedure.

To modify Siebel Tools

1 Open the `tools.cfg` file in a text editor.

2 Search for this text:

```
[SAPSubsys]
```

This starts the section defining the parameters for use with SAP. It has these entries:

```
[SAPSubsys]
SAPRfcUserName           = CHANGE_ME
SAPRfcPassword           = CHANGE_ME
SAPRfcConnectString      = DEST=CHANGE_ME CLIENT=CHANGE_ME LANG=CHANGE_ME
SAPCodepage               = CHANGE_ME
```

3. Replace CHANGE_ME after SAPRfcUserName = with the CPIC user name.
4. Replace CHANGE_ME after SAPRfcPassword = with the password for the user name entered.
5. Edit SAPRfcConnectString = as follows:
   a. Replace CHANGE_ME after DEST= with a destination name as defined in a TYPE=A entry of the `saprfc.ini` file being used.
   b. Replace CHANGE_ME after CLIENT= with the SAP client number.
   c. Replace CHANGE_ME after LANG= with the login language character (E for English).
   d. Be sure that there are no spaces on either side of the equals signs for DEST=, CLIENT= and LANG= and use at least one space after each entered value.
6. Replace CHANGE_ME after SAPCodepage = with the Siebel code page value for your SAP implementation. You can find this value in Table 11 on page 36.

   For example, your entry might look something like this:

```
[SAPSubsys]
SAPRfcUserName = auser
SAPRfcPassword = apassword
SAPRfcConnectString = DEST=DEV_Outbound CLIENT=555 LANG=E
SAPCodepage    = CP1252
```

7. Save and close the `tools.cfg` file.

To test new or modified workflows using the Business Service Simulator or the Workflow Simulator in a Siebel client modify your Siebel client configuration file.

Your SAP system administrator must create a CPIC user for your connection to SAP. The same user name used for your Siebel Tools configuration can be used here also.

NOTE: The Siebel Client configuration file is used only by the Siebel Mobile Web Client running in disconnected mode. As such the following procedure applies only to the use of this client in this mode.

To modify the Siebel Client configuration file

1. Open the appropriate configuration file for your Siebel client application.

   The name depends upon the application you use. For example, the sales application uses siebel.cfg and the call center application uses uagent.cfg.

2. Search for this text:

```
[SAPSubsys]
```

This starts the section defining the parameters for use with SAP. It has these entries.

```
[SAPSubsys]
SAPRfcUserName          = CHANGE_ME
SAPRfcPassword          = CHANGE_ME
SAPRfcConnectString     = DEST=CHANGE_ME CLIENT=CHANGE_ME LANG=CHANGE_ME
SAPRfcDestEntry         = CHANGE_ME
```

3. Replace CHANGE_ME after SAPRfcUserName = with the CPIC user name.
4. Replace CHANGE_ME after SAPRfcPassword = with the password for the user name entered.
5. Edit SAPRfcConnectString = as follows:
   a. Replace CHANGE_ME after DEST= with a destination name as defined in a TYPE=A entry of the `saprfc.ini` file being used.
   b. Replace CHANGE_ME after CLIENT= with the SAP client number.
   c. Replace CHANGE_ME after LANG= with the login language character (E for English).
   d. Be sure that there are no spaces on either side of the equals signs for DEST=, CLIENT= and LANG= and use at least one space after each entered value.
6. Replace CHANGE_ME after SAPRfcDestEntry = with the destination name of a TYPE=R entry in the `saprfc.ini` file.

   For example, your entry might look something like this:

```
[SAPSubsys]
SAPRfcUserName          = auser
SAPRfcPassword          = apassword
SAPRfcConnectString     = DEST=DEV_Outbound CLIENT=555 LANG=E
SAPRfcDestEntry         = DEV_Inbound
```

7. If you intend to work with IDOCs in the Workflow Simulator, then add the following line to the end of the [SAPSubsys] section:

   `SAPSenderPrtnrNum = *Logical system name*`

   where the logical system name is the name of your external system as configured in SAP for ALE. Define this logical system name in SAP. For more information, read "Configuring ALE/IDOC Connectivity" on page 26.
8. Save the configuration file and close the editor.

Siebel-to-SAP integration uses the Siebel Server Request Broker to make requests to the Business Integration Manager. For this reason, you may need to add Server Request Broker information to your configuration file. For more information, read *Overview: Siebel eBusiness Application Integration Volume I*.

Configuring ALE/IDOC Connectivity

ALE communication occurs between entities known as logical systems. Logical system definitions must be created within SAP by an experienced SAP Basis Resource. You create a logical system definition for the external Siebel implementation and for the SAP client so that these two entities can communicate with one another through ALE. A logical system definition contains the list of IDOC Types and Message types allowed for communication with that logical system. It also specifies where the logical system exists on the network so that SAP can uniquely identify that system when it is connected to SAP.

Figure 2 shows the data associated with defining a logical system. A Partner Profile, with both inbound and outbound message types communicates with a Port, which consists of an RFC Destination. An RFC Destination contains tRFC Parameters and a Program ID. The Program ID is written to the saprfc.ini file.

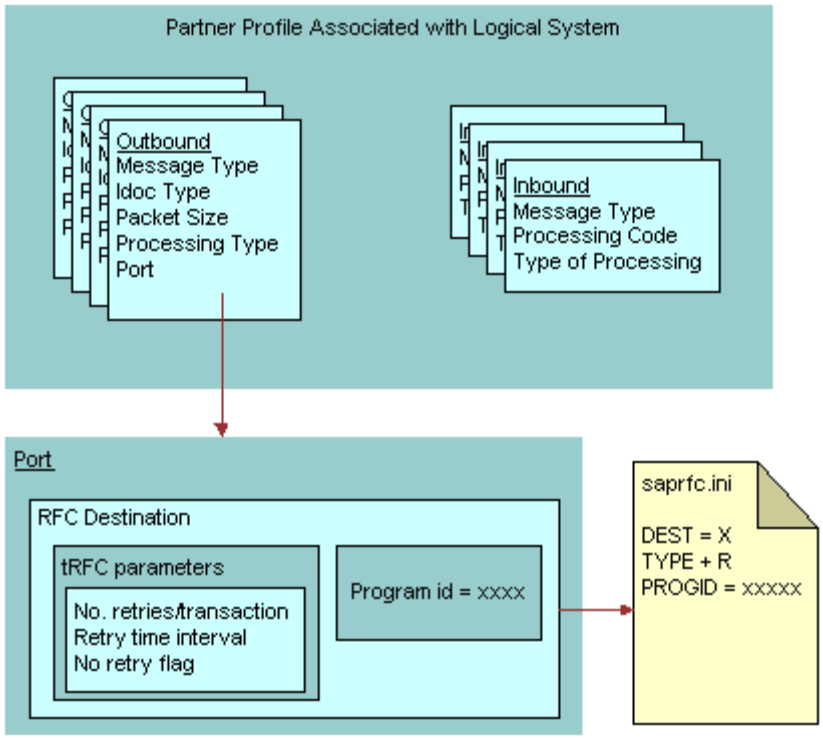


Figure 2. ALE Logical System Definition

You build logical systems from the bottom up. After you create a name for the logical system, then create an RFC Destination as shown in Figure 2. The most important part of the RFC Destination is the Program ID, as this value must exist in the TYPE=R destination entry of the `saprfc.ini` file. SAP uses the program ID to identify an external server and associate it with the logical system name.

After you have defined the RFC Destination, create a port definition and a Partner Profile definition. These are part of the logical system definition. These can be created manually or can be automatically generated by SAP if an SAP Distribution Model is created. The Distribution Model defines the list of Message Types that can be passed between any two logical systems. This information is used to create Partner Profiles in SAP. These Partner Profiles contain information for communication with an external system where data is transported outbound from SAP (Outbound Parameters) as well as information for communication with an external system where data is transported inbound to SAP (Inbound Parameters).

Table 9 describes the Outbound Parameters.

Table 9. Outbound Parameters

Parameter	Description
Message Type	For example: MATMAS (determines filtering and processing of IDOC)
Idoc Type	For example: MATMAS03 (determines structure of the data)
Packetsize	Number of IDOCs per transaction
Processing Type	Immediate or collect
Port	Defines where the IDOC is sent to

Table 10 describes the Inbound Parameters.

Table 10. Inbound Parameters

Parameter	Description
Message Type	For example: MATMAS (determines filtering and processing of IDOC)
Processing Code	Together with the Message Type is used to determine how the IDOC is processed.
Type of Processing	Process immediately or collect

To interface to SAP using ALE, you must create a logical system in SAP with all of its associated information. The process in this document can be used to create a logical system and to perform the minimum amount of configuration necessary to work with the Siebel ALE Interfaces. Depending upon your own configuration requirements you may need to modify these steps accordingly.

The SALE transaction handles all ALE configuration. This transaction takes you to the portion of the SAP Implementation Guide (IMG) that is relevant for ALE. The following procedures start with the SALE transaction. These procedures apply to SAP R/3 4.6C. Other versions of SAP differ slightly, but the SALE transaction still exists and the basic functions are available.

Creating Logical Systems Within SAP

You must define a logical system within SAP.

To create a logical system

1. Starting from the SALE transaction, select Sending and Receiving Systems > Logical Systems > Define Logical System.

   An SAP alert box warns you the table is client-independent.

2. Click the check mark button on the alert box.

   The Change View Logical Systems Overview window opens.

3. Click the New Entries button and complete the following fields.

   Log. System. Enter the name of a logical system to create.

   Description. Enter the description of the logical system.

4. Click Save.

   The Enter Change Request window opens.

5. Click Create Request.
6. Add a description in the pop-up box field.
7. Click Save.
8. Click the check mark to continue.
9. Click the left arrow button until you return to the Distribution ALE view.

NOTE: Another logical system must also be created and associated with the SAP Client. This may already exist in your implementation. Its name generally includes the client number. If it does not exist you need to create one using the preceding steps and then select Receiving Systems > Logical Systems > Assign Client to Logical System from SALE to assign it to the appropriate SAP Client. This logical system is referred to as the SAP Client Logical System, while the logical system you create to represent the Siebel applications is referred to as the External Logical System.

Creating RFC Destinations

You must create an RFC destination in SAP.

To create an RFC destination

1. From SALE, select Sending and Receiving Systems > Systems In Network > Define Target Systems for RFC Calls (SAP transaction SM59).
2. Click the folder icon for the TCP/IP connections RFC Destination to expand it.

3 Click Create.

The RFC Destination window opens.

4 Complete the following fields:

RFC Destination. Add an RFC destination name (suggestion: use the external system logical system name). If the names match, you can generate Partner Profiles for the logical system. If the names do not match you must manually create Ports and Outbound Parameters in your partner profile.

Connection Type. "T" for TCP/IP connection.

Description. Enter a text description.

5 Click Save.

The RFC Destination *CONNECTION_NAME* window opens.

6 Click Registration.

7 Type the Program ID for the name of the external program to which you want SAP to connect.

This is the same Program ID which you entered in the `saprfc.ini` file under the TYPE=R destination entry. Refer to the SAP Online Help for more information about Program ID.

8 If your external program is running and successfully registered to the SAP gateway, clicking the Test connection button returns ping times between the R/3 system and your program.

If your program has not connected successfully to the SAP gateway, then you get an error message.

NOTE: The program name is case sensitive, and if not defined correctly, IDOCs may not reach their destination. Also, after these parameters have been set, clicking other buttons may erase the settings. Use caution when in change mode in this window.

9 Click Save.

10 Click the check mark button to complete the task.

Distributing Logical Systems

NOTE: The following procedure must be done for each IDOC Message Type used. Message Types used in Standard Integrations are: DEBMAS, MATMAS, and ORDCHG. The distribution model entered here can later be used to generate Partner Profile Inbound and Outbound parameters.

To distribute a logical system

1 From SALE, select Modeling and Implementing Business Processes > Maintain Distribution Model and Distribute Views.

The Display Distribution Model window opens.

2 Click the Switch between Display and Edit Mode button to go into edit mode.

The Change Distribution Model window opens.

3 If you have not previously created a model that you can add to, click Create Model View. If you already have a model created that you wish to add to, skip to Step 6.

The Create Model View window opens.

4 Complete the following fields:

Short text. Enter a description in the Short Text field.

Technical Name. Enter a name for this model view, such as SIEBMODEL.

5 Click on the check mark button to continue.

6 Highlight the model view created and click Add Message Type.

The Add Message Type window opens.

7 Complete the following fields:

Field	Description		
Sender	Enter the logical system that is sending IDOCs. If the IDOC Message Type may be sent from the SAP system, this would be the SAP Client logical system name. If the IDOC Message Type may be sent from the Siebel application, this is the external logical system you created earlier.		
Receiver	Enter the logical system that is receiving IDOCs. If the IDOC Message Type may be received by SAP, this would be the SAP Client logical system name. If the IDOC Message Type may be received by the Siebel application, this is the external logical system you created earlier.		
Message Type	For the Standard Integrations you should add the following:		
	Message Type	Sender	Receiver
	DEBMAS	SAP Client LS	External LS
	DEBMAS	External LS	SAP Client LS
	MATMAS	SAP Client LS	External LS
	ORDCHG	SAP Client LS	External LS

8 Click the check mark button.

The Distribution Model Changed window opens with the logical system still highlighted.

9 Click Add Message Type again as necessary to set all message types used.

Repeat Step 6 through Step 8 for each message type and sender/receiver pair.

10 Click Save.

11 Click the left arrow button to exit.

Generating Partner Profiles

Partner profiles must be created for each logical system that SAP transfers IDOCs to or receives IDOCs from. The simplest way to create Partner Profiles is to use the Distribution Model created earlier to automatically generate them for you. You can then modify them as necessary.

To generate partner profiles

1. From SALE, select Modeling and Implementing Business Processes > Partner Profiles and Time of Processing > Generate Partner Profiles.
2. Enter Model Name in the Model View field and External Logical system name in the Partner System field.
3. Set the following default information.

   Postprocessing Type. Commonly US for user, depends upon the configuration.

   Postprocessing ID. If Type is US, this is a user name.

   Outbound Parameters Version. Use 3 for 4X ALE Interface.

   Outbound Parameters Packetsize. Set to 1 initially, this may be changed later for performance reasons.

   Outbound Parameters Output Mode. Set to Transfer immediately, this may be changed later for performance reasons.

   Inbound Parameters Processing. Set to trigger immediately, this may be changed later for performance reasons.
4. Click Execute to generate partner profiles.

   SAP displays a message indicating if you are successful.
5. Click the back arrow twice to return to SALE.
6. If you are using the Standard Integrations, after generating partner profiles check that the Outbound Parameters for the generated partner profiles use the correct IDOC Types.

   From SALE, choose Modeling and Implementing Business Processes > Partner Profiles and Time of Processing > Maintain Partner Profiles Manually.

   The Partner profiles window opens.
7. Select the LS category and find the logical system name for which you generated a partner profile.

   Select this logical system name.
8. Select each outbound parameter and edit it making sure that the following IDOC Types are used for each corresponding message type:

```
MATMAS  = MATMAS03
DEBMAS  = DEBMAS05
ORDCHG= ORDERS05
```

NOTE: If your SAP System version is not 4.6C, change the Seg. Release in IDoc Type field to the value 46C so that IDOCs sent from the SAP system match the integration objects created in Siebel.

Manually Creating Partner Profiles

You may also create Partner Profiles manually. If you have successfully generated Partner Profiles in "Generating Partner Profiles," you may skip to "Component Configuration" on page 34.

NOTE: Even if you manually create your Partner Profiles, you still need to add your Outbound Parameter message types to a Distribution Model. Otherwise, SAP does not generate communication IDOCs when you attempt to send an IDOC.

Before creating a partner profile manually, you must create a port for the RFC destination you created earlier. If you generate your partner profiles, ports are automatically generated for you and this step is not necessary.

To create ports

1. From SALE, select Sending and Receiving Systems > Systems in Network > Asynchronous Processing > Assigning Ports > Define Port.

   The Ports in IDOC processing view opens.

2. Select the Transactional RFC entry and click Create.
3. Click one of the following option buttons:

   Generate Port Number. SAP automatically creates a port number.

   Port number. The name of the port.

4. Click the check mark button through any informational pop-up message boxes.

   The Creating a tRFC port view opens.

5. Complete the following fields.

   Description. Brief description of this logical destination.

   Version. You may select either version. For the Siebel Connector for SAP R/3, V7.x, the default is 4x. The Siebel Connector for SAP R/3 V6.x adapter supports only the 3.x version.

   RFC Destination. The name of the logical destination (TCP/IP connection) created earlier (from SM59). Click the possible entries button to view the list of values for the logical system.

6. Click Save.

After you have created a port, you can create partner profiles.

To create partner profiles

1 From SALE, choose Modeling and Implementing Business Processes > Partner Profiles and Time of Processing > Maintain Partner Profiles.

The Partner profiles window opens.

2 Choose Partner Type LS and click Create.

3 Complete the following fields:

Partn.number. The receiving logical system name created earlier in the configuration.

Partn.type. The type of partner: LS = Logical System.

4 In the Post processing: permitted agent tab, complete the following fields:

Typ. Type of ID: U = User, O = Organizational chart.

Agent. UserID of person to be notified of IDOC transfer errors.

Lang. Language: EN = English.

5 In the Classification tab, complete the following fields:

Partner class. Enter a partner class if desired.

Partner status. A for Active.

6 Click Save.

7 Click Create Outbound Parameters.

The Partner Profiles: Outbound Parameters window opens.

8 In Partner Profiles: Outbound Parameters screen, complete the following fields:

Message type. Enter MATMAS for Material master records, DEBMAS for Customer master records, or ORDCHG for Order Update master records.

9 In the Outbound options tab, complete the following fields:

Receiver port. Enter the name of the previously created port for your logical system.

Packet size. Enter the number of IDOCs per package, for example 20.

NOTE: This field does not appear until after you have saved.

Output mode. Choose to either transfer IDOCs immediately or collect them and send them all at once.

NOTE: Entering a value into the Packet Size field is necessary only when all IDOCs are being sent at once. If you choose to transfer an IDOC immediately, only one IDOC will be sent, regardless of the value in the Packet Size field.

IDOC type/Basic type. Enter the IDOC that is associated with the following message types:

```
MATMAS = MATMAS03
DEBMAS = DEBMAS05
ORDCHG = ORDERS05
```

Seg. release in IDoc type. If your SAP system version is not 4.6C, then enter the value 46C here to insure that the IDOCs sent from SAP to Siebel match the integration objects in Siebel.

10 In the Post processing: permitted agent tab, complete the following fields:

Typ. Type of ID: US = User.

Agent. UserID of person to be notified of IDOC transfer errors.

Lang. Language: EN = English.

11 Click Save.

12 Click the left arrow to return to the Partner Profiles window.

13 Repeat Step 7 through Step 12 to add the DEBMAS and ORDCHG IDOC message types.

14 Click Create Inbound Parameters.

The Partner Profiles: Inbound Parameters window opens.

15 Partner Profiles: In the Inbound Parameters window, complete the following fields:

Message type. DEBMAS for Customer master records.

Process code. The SAP code behind the processing of this message type: DEBMAS = DEBM

Syntax check. If the IDOC should be syntax checked upon posting.

Processing by function module. Select a processing option.

16 In the Post processing: permitted agent tab, complete the following fields:

Typ. Type of ID: US = User, O = Organizational chart.

Agent. UserID of person to be notified of IDOC transfer errors.

Lang. Language: EN = English.

17 Click Save.

Component Configuration

The following objects must be configured:

- Server component groups
- Server components
- Server component parameters

Component groups are assigned and enabled by default. The components within the component group are also, by default, enabled and set to online. Check to make sure that the following component groups have been enabled:

- SAP Connector
- Enterprise Application Integration (EAI)

Also make sure that the following two components within the SAP Connector component group are enabled and online:

- SAP BAPI tRFC Receiver
- Business Integration ManagerServer

For information on enabling component groups, read *Siebel System Administration Guide*. You may enable component groups at server installation. Check the EAI and SAP component groups when prompted during Siebel server installation. For more information on enabling server component groups in the installation, refer to *Deployment Planning Guide*.

Setting Server Component Parameters

Specific component parameters must be set for the tRFC BAPI Receiver and Business Integration Manager server components.

Your SAP system administrator must create a CPIC user for your connection to SAP. This user name and password are required in the following steps.

To set the Business Integration Manager (BusIntMgr) component parameters

1. Set the SAPRfcUserName parameter for the Business Integration Manager component. You can use either of two methods:
   - Method 1. Within the Siebel client, navigate to the Component Parameters form and set the value of the parameter to the user name your administrator has created.
   - Method 2. Using the Server Manager line mode interface, use the `change param` command to set the value of the parameter to your SAP user name. For example: change param SAPRfcUserName="auser" for comp BusIntMgr.

   More information on how to set component parameters can be found in the *Siebel System Administration Guide*.
2. Set the SAPRfcPassword parameter for the Business Integration Manager component to the SAP password for the user created.
3. Set the SAPRfcConnectString parameter for the Business Integration Manager component.

   `DEST=`*`destination name`* `CLIENT=`*`SAP client`* `LANG=`*`SAP Login language`*

   where *destination name* is the destination name of a TYPE=A entry in the saprfc.ini file, *SAP client* is the three digit SAP client number and *SAP login language* is the login language character (E for English).

   For example your SAPRfcConnectString parameter value may look something like this:

   ```
   DEST=DEV_Outbound CLIENT=555 LANG=E
   ```

4 Set the SAPSenderPrtnrNum parameter for the Business Integration Manager component to the SAP logical system name that defines your external system to SAP.

This name was entered into SAP during ALE configuration, read "Configuring ALE/IDOC Connectivity" on page 26.

5 Restart the Siebel Server.

To set the SAP tRFC BAPI receiver (BAPIRcvr) component parameters

- Set the SAPRfcDestEntry component parameter for the SAP tRFC BAPI Receiver component value to the destination name of a TYPE=R entry in the `saprfc.ini` file.

  Within the Siebel Client navigate to the Component Parameters form and set the value of the parameter.

SAP Code Page Configuration

As of Siebel 7.5, all Siebel components use Unicode internally. Therefore, all SAP code page data interfaced to Siebel applications must be converted to and from Unicode by Siebel Connector for SAP R/3. This may require some additional configuration. The BAPI Adapter and BAPI Receiver do not need any special configuration, however, you must make configuration changes for Siebel Tools and for MQSeries.

Table 11 shows the SAP code page mappings delivered with Siebel 7.7. These are configured in the EAI Value Map SAP code page. The EAI Value Maps can be accessed by navigating to Integration Administration > EAI Value Maps and querying for SAP code page in the Type field. The list of possible Siebel code page values is in the Transcode Encoding picklist. The EAI Value Maps values can be modified if required.

Occasionally, the ISO code page will not match the Microsoft code page. The behavior of the SAP Connector in such cases is described in Appendix C, "SAP Code Page Mappings."

Table 11. SAP Codepages and Corresponding EAI Value Mappings

Siebel Value	SAP Code page	Description
CP1252	1100	SAP Latin-1 - ISO8859-1 - code page
ISO-8859-2	1402	SAP Latin-2 - ISO8859-2
ISO-8859-5	1500	SAP Cyrillic - ISO8859-5
CP1254	1610	SAP Turkish - ISO8859-9
CP1253	1700	SAP Greek - ISO8859-7 - Not a complete match
CP1255	1800	SAP Hebrew - ISO8859-8 - Not a complete match
CP932	8000	SAP Shift-JIS
CP950	8300	SAP Taiwanese
CP936	8400	SAP Chinese

Table 11. SAP Codepages and Corresponding EAI Value Mappings

Siebel Value	SAP Code page	Description
CP949	8500	SAP Korean
CP874	8600	SAP Thai

BAPI Adapter and BAPI Receiver

The BAPI Adapter and BAPI Receiver for Siebel 7.5 support SAP implementations using single codepages as well as Multi-Display Multi-Processing SAP implementations. Multi-Display Single Processing SAP implementations are not supported, as these use special SAP blended codepages. For the BAPI Adapter and BAPI Receiver, no additional configuration is required. The BAPI Adapter and BAPI Receiver retrieve SAP's code page from SAP and convert data based on this value. This is true even if the BAPI Adapter and Receiver are working with multiple SAP installations that are based on different codepages.

CAUTION: If you are interfacing to multiple SAP implementations with different codepages, you are responsible for partitioning the data within the Siebel application so the data can be correctly transported to SAP with the correct code page for that data. For example, if you are using Siebel applications with a Unicode database it would be possible to store data in Siebel data received from an SAP implementation using code page 1100 (Latin-1, European) and from an SAP implementation using code page 8000 (Shift-JIS, Japanese). However, you are then responsible for programmatically making sure that code page 8000 data is not returned to the code page 1100 SAP implementation, as Japanese characters cannot be converted to the European character set defined by code page 1100. Siebel Systems provides no predefined method for partitioning the data based upon the application it originated in.

When interfacing to an SAP implementation based on a single code page, all data that is transported to and from SAP is transported in a single code page. When data is sent from the Siebel database to SAP through the BAPI Adapter, the adapter retrieves the code page from SAP automatically after connecting. It then looks up the SAP code page value in the SAP Code Page EAI Value Map and uses the corresponding Siebel code page value to translate all data passing between the Siebel application and SAP.

When data is sent from SAP to the Siebel application through the BAPI Receiver, the receiver retrieves the code page value from SAP each time a packet of data is sent from SAP. It then looks up the SAP code page value in the SAP Code Page EAI Value Map and uses the corresponding Siebel code page value to translate all data it receives from SAP from the SAP code page to Unicode.

The interface to a Multi-Display Multi-Processing SAP implementation is similar. The difference is that the code page that is automatically retrieved is dependent upon the login language of the user in the SAP implementation. When using the BAPI adapter, the login language defined in the parameter SAPConnectString is used to determine the code page. When using the BAPI Receiver, the login language of the user responsible for the send of the data to the receiver is used. This means that the BAPI Receiver may receive packets of data in different codepages depending upon who sent the data.

Although not necessary, you can override the automatic retrieval of the SAP code page from SAP by setting the SAPCodepage parameter to a Siebel Value as defined in Table 11 on page 36. For the BAPI Adapter, set this parameter on the BusIntMgr component just as you would set the parameters SAPUserName, SAPConnectString, and SAPPassword. For the BAPI Receiver set this parameter on the BAPIRcvr component just as you would set the SAPRfcDestEntry parameter for this component.

Because the codepages are retrieved automatically, the BAPI Adapter can interface to multiple SAP implementations that use different codepages. Starting with Siebel v7.5, the BAPI Adapter can dynamically change its connection and its code page. To switch the connection from one to another, the parameters SAPUserName, SAPConnectString and SAPPassword must be set as input arguments to the BAPI Adapter business service call made from the workflow. The BAPI Adapter then uses these parameters instead of those set at the BusIntMgr component level. When a call is then made to the BAPI Adapter that is for a different SAP implementation (or different user or language in the same SAP implementation) than the previous call, the old connection to SAP ends and a new connection is made. When the new connection is made, the code page is retrieved from the new connection and data is processed using the new code page.

You can customize the handling of errors that occur during code page conversion by using the SAPIgnoreCharSetConvErrors parameter. When this parameter is set to True, errors in code page conversion are handled by substituting a question mark (?) for characters that cannot be converted. This parameter can be set on the BusIntMgr component and also on the BAPIRcvr for BAPI Adapter and BAPI Receiver usage respectively.

Siebel Tools

When using Siebel Tools the code page must be set in the tools.cfg file. This is because the EAI Value Map information is not available in the Siebel Tools environment. To set the value, follow the instructions for modifying the tools.cfg file in "Modifying SAP Configuration Files" on page 19.

MQSeries

For MQ Series, the code page must be explicitly set, therefore only single code page SAP implementations are supported. For transport of IDOCs from the Siebel application to SAP, set the SAPCodepage parameter on the BusIntMgr component to a Siebel code page value. For transport of IDOCs from SAP to the Siebel application, set the SAPCodepage parameter on the SAPIdocAMIMqRcvr component.

Checking Connectivity

When you have completed the installation and configuration, you can do some simple tests to see if your basic connection information is correct. These tests do not exercise all possible functionality, but they can tell you if your connection information for SAP is correctly defined. These tests are:

1. Checking Siebel Client connectivity.
2. Checking tRFC BAPI Receiver connectivity.
3. Checking Siebel Tools connectivity.

Checking Siebel Client Connectivity

You can check your SAP connection information by executing the TestSAPConnection workflow. This workflow contains a single call to the BAPI Adapter business service method MakeConnection. The flow can be executed through the Workflow Simulator. If no errors occur during its execution, the connection to SAP has been successful. If you receive errors, you may need to correct entries in the `saprfc.ini` file or the values of SAPRfcConnectString, SAPRfcUserName, or SAPRfcPassword.

To check client connectivity

1. Modify the TestSAPConnection workflow to add three new Input Arguments:
   - SAPRfcConnectString
   - SAPRfcUserName
   - SAPRfcPassword
2. Set their Type to Literal.
3. In the Value field, enter their values as they would have been configured in the Siebel Configuration file or as set for the BusIntMgr component.

   NOTE: In the Standard Integration workflows these parameters are set on the BusIntMgr server component, not within the workflow itself. They need to be set within the workflow because they are running within the workflow simulator. If you are using the Siebel Mobile Web Client the values can also be set in the Siebel Configuration file for the client.
4. Execute the workflow using the workflow simulator. For more information on running the workflow simulator, see *Siebel Business Process Designer Administration Guide*.

   If no errors are displayed, the connection information is correct.

For more information on troubleshooting connection problems, read Appendix D, "Troubleshooting Siebel Connector for SAP R/3."

Checking tRFC BAPI Receiver Connectivity

You should be able to start a tRFC BAPI Receiver background component and test its connection from the SAP environment. This topic guides you through these tests.

Starting the tRFC BAPI Receiver

Start a BAPIRcvr after the Server is running by using a Server Manager line mode command. In addition, you can set the value of parameters such as SAPRfcDestEntry within the line mode command. For example:

```
start server for comp BAPIRcvr with SAPRfcDestEntry=DEV_Inbound
```

This command starts a BAPIRcvr while the Server is running. In this way, multiple BAPIRcvrs can be started that connect to different SAP installations.

Testing BAPIRcvr Connection

After you have started BAPIRcvr, you can check the connection between SAP and the receiver from the SAP transaction SM59. From SM59, select your RFC Destination from the tree under the TCP/IP branch. Then select the Test Connection button. SAP displays an error message if it is unable to connect to the BAPIRcvr. It displays timing information if the connection is successful.

Checking Connectivity from Siebel Tools to SAP

You can test connectivity from Siebel Tools to SAP by executing the Integration Object Wizard.

To check Siebel Tools connectivity

1. Start Siebel Tools.
2. Make sure that you have a project checked out that you can add the new integration object into.
3. Click New or File > New Object.

   The New Object window opens.
4. Select the Integration Object icon and click OK.

   The Integration Object Builder window opens.
5. In the top drop-down list, choose the project into which you want to add the integration object.
6. In the second drop-down list, choose the name of the wizard used to create the type of integration object you want. In this case, choose EAI SAP IDOC Adapter Wizard.
7. Click Next.

   Siebel Tools displays a list of IDOCs in the drop-down list. If you receive an error message, some connection information is not correctly defined in your `saprfc.ini` or `tools.cfg` file. Appendix D, "Troubleshooting Siebel Connector for SAP R/3" has more information on troubleshooting connection problems.
8. Click Cancel if the test was successful, or click through the error message information to exit the wizard.

Configuring the SAP Connector for Use with IBM MQSeries

To transport IDOCS between SAP and the Siebel application with IBM MQSeries, use the SAP IDOC AMI Receiver for MQSeries to manage SAP-to-Siebel messages and use the EAI SAP IDOC MQ AMI Adapter to handle Siebel-to-SAP messages. You need to set configuration parameters for the Business Services you will invoke and the Receiver Server Components you will use.

Requirements to Run IBM MQSeries

If you choose to use the Siebel Connector for SAP R/3 with MQSeries, refer to the system requirements and supported platforms documentation for your Siebel application for more information.

Set Up MQSeries and AMI

The general setup requires that you install IBM MQSeries and AMI package and complete their configuration. Refer to the general setup and configuration information in *Transports and Interfaces: Siebel eBusiness Application Integration Volume III*.

Setting Up Connectivity to MQSeries Server and MQ Link

Set the connectivity parameters between your Siebel Server and the MQSeries AMI server using MQ Link. You also need to set the connectivity parameters between the SAP server and the MQSeries server. These instructions assume you are using the Siebel SAP IDOC AMI Receiver for MQSeries, and not the MQSeries AMI Receiver, as your server component.

To set the connectivity parameters

1 Edit the following file:

MQ Link Installation Directory\sample\out.ini

2 Set the following parameters:

Parameter	Description
Queue Manager =	*Queue Manager Name*
gatewayhost =	*SAP host name*
gatewayservice =	*SAP gateway service name*
programid =	SAP program ID associated with RFC destination and logical system

NOTE: A program ID is an identifier used by SAP to recognize a specific server program. This identifier is defined as part of the RFC Destination in SAP and is tied to your logical system name. The program ID you used to configure ALE for your SAP implementation needs to be entered here. For more information, read "Creating RFC Destinations" on page 28.

3 Save the out.ini file.

4 Edit the following file:

MQ Link Installation Directory\sample\smqdestconf

5 Set the following parameters:

Parameter	Description
receivingpartner =	Your receiver's logical system name as you have configured it in SAP
edi_mestype =	* (allows all types of IDOCs)
outboundqueuemanager =	*Queue Manager Name*
outboundqueue =	SMQ_OUTBOUND_QUEUE
HostName =	*SAP gateway host name*
UserID =	SAP RFC user ID
Password =	SAP password
Default =	"Yes"

6 Save the `smqdestconf` file.

To set the connectivity parameters for outbound (Siebel application to SAP)

1 Edit the following file:

`MQ Link Installation Directory``\sample\in.ini`

2 Set the following parameters:

Parameter	Description
client =	*client number*
user =	*user name*
language =	*language code*
password =	*password*
sysnbr =	*R/3 system number*
hostname =	*R/3 gateway host*
transactionqueue =	*Transaction Queue name*
queuemanager =	*Queue Manager name*
badmessagequeue =	*bad message queue name*
inboundqueue =	*queue name for inbound to SAP*

3 Save the `in.ini` file.

Running MQ Link to Connect SAP to the Siebel Receiver Server Component

To run MQLink

1. Change to the following directory:

   MQ Link Set Up Directory\sample

2. For inbound to the Siebel application, enter the following line after the command prompt:

   ```
   Smqso -iout.ini
   ```

   NOTE: For inbound to the Siebel application, the command links SAP to the Siebel Receiver Server Component, and passes data from SAP to the Siebel application.

3. Press ENTER to run the program.

4. For outbound from the Siebel application to SAP, enter the following line after the command prompt:

   ```
   Smqso -iin.ini
   ```

   NOTE: For outbound from the Siebel application, a sample workflow is provided. This workflow invokes the `EAI SAP IDOC MQ AMI Adapter` and from that adapter invokes EAI MQSeries AMI Transport Adapter to send data to MQ. The data then passes through MQ Link to SAP.

5. Press ENTER to run the program.

When you have established the MQSeries queues, policy, services and MQLink, you can use the Siebel SAP IDOC AMI Receiver for MQSeries for inbound integration, and use the sample workflow provided for outbound integration. The outbound sample workflow uses `EAI SAP IDOC MQ AMI Adapter` and EAI MQSeries AMI Transport to send data from the Siebel application to SAP.

Setting Server Subsystem to Start the SAP IDOC AMI Receiver

Because the EAI Receiver infrastructure and transport mechanisms have changed in Siebel 7, the previous parameter specification at the Business Service level and the server Component parameter level is no longer used for the EAI transport related receivers. The new receivers use named subsystems.

There are two kinds of subsystems to be specified to start the SAP IDOC AMI Receiver for MQSeries (sapidocamimqrcvr):

- ReceiverConnectionSubsystem
- ReceiverDataHandlingSubsystem

Because the ReceiverDataHandlingSubsystem is preconfigured for sapidocamimqrcvr, you only need to provide ReceiverConnectionSubsystem.

To set the ReceiverConnectionSubsystem

1. Log in to server manager line mode (you may also use server administration).
2. Type in this command:

   ```
   create named subsystem your subsystem name for subsystem MQSeriesAMISubsys with MqPolicyName=your MQ AMI Policy name, MqReceiverServiceName=your MQ AMI Receiver Service name
   ```

3. Use the command, list named subsystem, to make sure the subsystem is created.
4. Stop the Siebel server service and restart it.
5. Start the receiver with this command:

   ```
   start server for comp sapidocamimqrcvr with ReceiverConnectionSubsystem=your subsystem name
   ```

The SAP IDOC AMI Receiver can be used to trigger workflows upon the receipt of an IDOC just as the BAPIRcvr does. The receiver is configured by default to allow the invocation of the Standard Integration inbound workflows: Account - Send or Receive SAP 46C Customer (DEBMAS05), Product - Receive SAP 46C Material (MATMAS03), and SAP 46C Order Status (IDOC 4x).

Sample Outbound Workflow

A sample workflow is provided to update a customer in SAP from the Siebel application. This workflow can be used in place of the Standard Integration Workflow Account - Send SAP 46C Customer. The sample workflow is called Send46CIdoc_UpdateSAPCustomer_MQAMI. This workflow passes a customer IDOC from the Siebel application to SAP through MQ Series instead of using the IDOC and tRFC BAPI adapters.

The sequence of the flow is:

```
Siebel Adapter > Data Map > EAI SAP IDOC MQ AMI Adapter > EAI MQSeries AMI Transport.
```

To use the sample workflow in place of the standard integration workflow

1. Modify the standard Account Integration workflow SAP 4x Account - Submit SAP 46C Customer.
   a. Navigate to Workflow Process using Siebel Tools.
   b. Locate the Account - Submit SAP 46C Customer workflow, and then click Revise.
   c. Right-click the workflow, and select Edit workflow process.
   d. In the workflow steps applet for Update SAP Customer replace the value Account - Send SAP 46C Customer with Send46CIdoc_UpdateSAPCustomer_MQAMI.
   e. Save the changes.
   f. Repeat Step d and Step e for the Update Sales Area Info step.
   g. Return to Workflow Processes and deploy the workflow.
2. Modify the sample workflow Send4xIdoc_UpdateSAPCustomer_MQAMI.

a From Tools > Workflow Process locate the sample workflow Send46CIdoc_UpdateSAPCustomer_MQAMI and click Revise.

   A new version of the workflow appears with status In Progress.

b Right-click the workflow and select Edit workflow process.

c Select the business service step, 4. MQ AMI Transport.

d Right-click and select Show Input Arguments, change the value of the Connection Subsystem argument from SAPSubsys to a name of your own choice, for example MQSenderSubsys.

e Save the changes.

f Return to workflow processes and deploy the workflow.

3 Activate workflows.

4 Create the connection subsystem that you named in the Connection Subsystem argument.

a Log in to server manager line mode (you may also use server administration).

b Enter this command:

```
create named subsystem your subsystem name for subsystem
MQSeriesAMISubsys with MqPolicyName=your MQ AMI Policy name,
MqSenderServiceName=your MQ AMI Sender Service name
```

c Use the `list named subsystem` command to make sure the subsystem is created.

d Stop the Siebel server service and restart it.

Now when you send an account to SAP from the Siebel application, it invokes the sample workflow.

SAP Configuration for Standard Integrations

The mappings between the Siebel application and SAP objects are based on the Preconfigured Client Configuration (PCC) of SAP R/3 4.6C. The following topics describe some of these.

Enterprise Structure Setup for Sales and Distribution

To work with the preconfigured standard data flows for the Siebel-SAP Integration you must define, at minimum, a sales organization, distribution channel, division, plant, and a shipping point. In addition, you need to define the following assignments:

- Plant to company code
- Sales organization to company code
- Distribution channel to sales organization
- Division to sales organization
- Sales area

- Plant to sales organization and distribution channel
- Shipping point to plant

SAP Pricing Setup

In Quotes and Orders you can choose either the Siebel application or SAP as your master system for pricing.

SAP as the Price Master

The Siebel SAP Price Integration for SAP as price master uses a PCC Standard Price Procedure. SAP R/3 computes the net price, tax, and freight information. The standard mappings use SAP pricing condition PB00. In the Siebel Standard Price Integration, tax and freight are sent back to the Siebel application. To support this function you need to add an additional subtotal line for freight amount in your SAP price procedure.

Siebel Application as the Price Master

The standard mappings use SAP pricing condition PN00 to transfer the Siebel Item Net Price to SAP, and the condition HB00 to transfer the freight charges. The Siebel SAP Price Integration for the Siebel application as price master uses a customized PCC Price Procedure. The sequence for the used price conditions should be set up as:

```
Step n:   PN00
Step n+X: HB00
```

If you customize your price procedure and conditions in SAP, make sure that you synchronize the names of the SAP condition types used for mapping purposes with the Business Service Data mappings.

The Siebel Connector for SAP R/3 uses the communication structure KOMP to transfer freight and tax.

- Freight. Carry over value to KOMP-KZWI4 (subtotal_4)
- Tax. Carry over value to KOMP-KZWI5 (subtotal_5)

Based on your business scenario for Pricing, you might want to enter price relevant data like Payment terms and Incoterms in your Siebel Quote /Order. Make sure that these data are maintained in the EAI value maps.

SAP Master Data

For customer data, you must:

- Define a company code, sales organization, distribution channel, and division.
- On the shipping screen, fill out the Delivering Plant field (KNVV-VWERK).

- Define the partner functions Sold-to party (SP), Bill-to party (BP), Payer (PY), and Ship-to party (SH). By default, the sales order mappings between Siebel applications and SAP R/3 assume a single Payer.
- Define a default partner for each partner function.

For materials data, you must define a plant, sales organization, and distribution channel.

Siebel Configuration for Standard Integrations

Before you can use any of the standard integrations, you must first configure the data elements that are common to all of the standard integrations. This configuration involves setting SAP configuration data within the Siebel environment. The standard integration workflows and maps can use this information during transformation of data between the Siebel application and SAP.

The steps necessary for adding the SAP data to the Siebel application are:

- "Changing Siebel LOV Definitions"
- "Adding Integration Administration Data" on page 50

Changing Siebel LOV Definitions

Specific List of Values (LOV) definitions are predefined to support the setup of the required SAP Configuration data in the Siebel application. The SAP configurations in Table 12 must be defined within the List of Values (LOVs).

Table 12 and Table 13 contain an SAP Reference column that you may use to find the applicable SAP configuration data in your implementation. Table 13 contains definitions that are provided as part of your Siebel installation.

Table 12. List of Values for Customizable SAP Configuration Data

SAP Configuration Data	LOV Type	Display Value	Language Independent Value	SAP Reference
Blocking Reason for Billing	SAP_SO_ HEADER_ BILL_BLOCK	SAP Description example: Calculation Missing	SAP Code example: 01	SAP transaction ovv3 IMG section: Sales and Distribution > Billing > Billing Documents > Define Blocking Reason for Billing, select Billing: Blocking Reasons
Customer Account Group	ACCOUNT_ GROUP_TYPE	SAP Description example: Sold to Party	SAP Code example: 0001	SAP transaction OBD2 IMG Section: Financial Accounting > Accounts Receivable and Accounts Payable > Customer Accounts > Master Records > Preparations for Creating Customer Master Records > Define Account Groups with Screen Layout (Customers)
Delivery Blocks	SAP_SO_ HEADER_ DELIV_BLOCK	SAP Description example: Credit Limit	SAP Code example: 01	SAP transaction ovlx IMG section: Logistics Execution > Shipping > Deliveries > Define Reasons for Blocking in Shipping, select Delivery Blocks
Sales Organization	SAP_SALES_ ORG	SAP Description example: Europe	SAP Code example: 0010	SAP transaction OVX3 IMG section: Enterprise Structure> Assignment > Sales and Distribution > Assign Sales Organization to Company Code
Distribution Channel	SAP_ DISTRIBUTION _CHANNEL	SAP Description example: Retail	SAP Code example: 10	SAP transaction OVXK IMG Section: Enterprise Structure > Assignment > Sales and Distribution > Assign Distribution Channel to Sales Organization

Table 12. List of Values for Customizable SAP Configuration Data

SAP Configuration Data	LOV Type	Display Value	Language Independent Value	SAP Reference
Division	SAP_DIVN_CD	SAP Description example: Software	SAP Code example: 10	SAP transaction OVXA IMG Section: Enterprise Structure > Assignment > Sales and Distribution > Assign Division to Sales Organization
Plant	SAP_PLANT	SAP Code example: 10	SAP Code example: 10	SAP transaction OVX6 IMG Section: Enterprise Structure > Assignment > Sales and Distribution > Assign Sales Organization - Distribution Channel-Plant

Table 13. List of Values for Seeded SAP Configuration Data

SAP Configuration Data	LOV Type	SAP Reference
Header Delivery Status	SAP_SO_HEADER_DELIV_STATUS	SAP Transaction SE11 Valid values for Domain STATV (Document status)
Item Delivery Status	SAP_SO_ITEM_DELIV_STATUS	Same as Header Delivery Status
Overall Status	SAP_SO_OVERALL_STATUS	Same as Header Delivery Status

These definitions must be made consistent with your SAP configuration data. Compare these definitions with the values in SAP and make changes as required. Table 13 contains an SAP Reference column that you may use to find the correct SAP configuration data in your implementation. This column contains an SAP transaction code in which you can view the configuration data as well as a reference section to the SAP Implementation Guide (IMG) where applicable. This information is correct for SAP R/3 4.6C. It may vary in other SAP versions.

To make sure the required SAP entities are copied into the Siebel database, you should synchronize the LOV types.

To synchronize LOV types

1 In the Siebel client, navigate to Administration - Data and select List of Values.

The following figure shows the List of Values list with its Type, Display Value, Language Independent Code, Language Name, Parent LIC, Order, Active, Translate, Multilingual and Replication Level fields.

List of Values | Menu | New | Delete | Query | Clear Cache | 1 - 1

Type	Display Value	Language-Indepe	Language Name	Parent LIC	Order	Active	Translate	Multilingual	Replication Level
ABS_COST_FCT_A	Field Service	Field Service	English-American		1	✓	✓		All
ABS_COST_FCT_A	Workforce Manageme	Workforce Manager	English-American		3	✓	✓		All
ABS_COST_FCT_A	Professional Services	Professional Servic	English-American		9	✓	✓		All
ABS_COST_FCT_T	Normal	Normal	English-American		1	✓	✓		All
ABS_COST_FCT_T	Emergency	Emergency	English-American		2	✓	✓		All
ABS_COST_FCT_V.	Constraint Violation	Constraint Violation	English-American		1	✓	✓		All
ABS_COST_FCT_V.	FSE Overtime	FSE Overtime	English-American		2	✓	✓		All
ABS_COST_FCT_V.	Tardiness	Tardiness	English-American		3	✓	✓		All
ABS_COST_FCT_V.	Task Exclusion Penalt	Task Exclusion Pena	English-American		4	✓	✓		All
ABS_COST_FCT_V.	Travel Distance	Travel Distance	English-American		5	✓	✓		All
ACCNT_CRDT_STA	Approved	1	English-American		1	✓	✓		All
ACCNT_CRDT_STA	Over Limit	2	English-American		2	✓	✓		All
ACCNT_CRDT_STA	Overdue	3	English-American		3	✓	✓		All
ACCNT_CRDT_STA	Block	4	English-American		4	✓	✓		All
ACCNT_CRDT_STA	Indeterminate	5	English-American		5	✓	✓		All

2 Query for each LOV Type in the List of Values list to see predefined values.

3 Navigate to the SAP Reference area using an SAPGui client application to see the values defined in SAP.

4 If differences exist between the definitions in the Siebel repository and SAP, correct or add entries in the Siebel repository as necessary.

5 Repeat Step 2 through Step 4 for each LOV type.

Adding Integration Administration Data

Integration administration data represents the SAP sales enterprise structure in the Siebel database. An SAP sales area and an SAP plant map to a Siebel organization. An SAP plant has a representation as a specific Inventory Location type in the Siebel application.

You need to add SAP configuration data to your Siebel application. Table 14 contains an SAP Reference column that you can use to find the correct SAP configuration data. This column contains an SAP transaction code where you can view the configuration data and a reference to the SAP Implementation Guide (IMG), where applicable. This information is correct for SAP R/3 4.6C. It may vary in other SAP versions.

Table 14. Integration Administration Data

SAP Configuration Data	Siebel Integration Application Data	SAP Reference
Plants	Organization SAP Plants	SAP transaction OVX6 IMG Section: Enterprise Structure > Assignment > Sales and Distribution > Assign Sales Organization - Distribution Channel-Plant
Sales Area	Organization SAP Sales Area	Sales Organization to Division Codes: SAP transaction OVXG IMG Section: Enterprise Structure> Assignment > Sales and Distribution > Setup Sales Areas

To complete the integration administration, do the following:

- Map SAP sales areas to Siebel organizations
- Change EAI value maps for reference customers

Mapping SAP Sales Areas

The Business Components used in the Siebel SAP standard integration are enabled for usage in a Multi Organization environment (such as BC Account, Internal Product, Inventory Location, and so on). By default an organization is assigned to those business components. Make sure that the default Siebel organization is synchronized with your SAP sales enterprise structure.

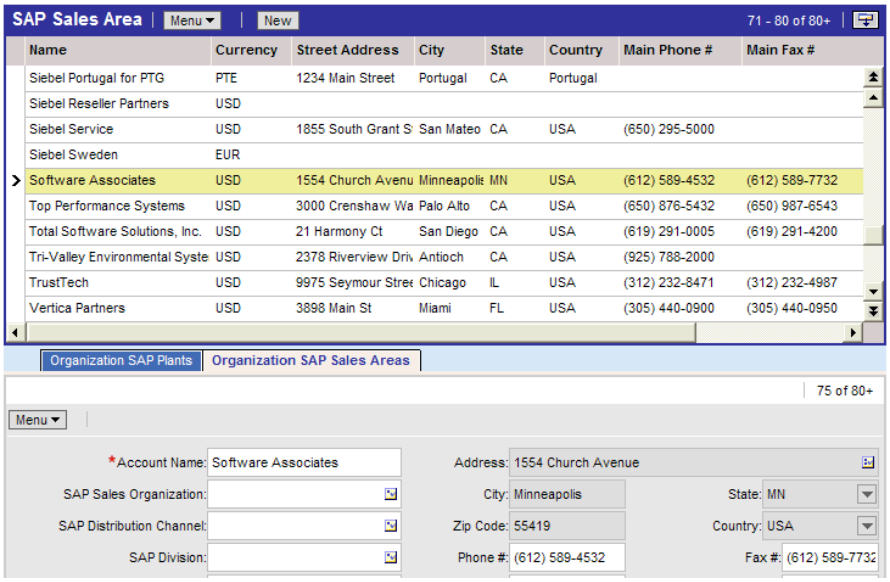
NOTE: This procedure requires the simultaneous use of the SAP and Siebel applications. The steps in this procedure indicate when to switch applications.

To map SAP Sales Areas to Siebel Organizations

1. In the Siebel application, navigate to Administration - Integration.
2. Select SAP Administration.

3 Select Organization SAP Sales Areas if it is not already selected.

The following figure shows the SAP Sales Area list with a record selected. Under it is the Organization SAP Sales Areas form showing the details for the selected record, such as Account Name, SAP Sales Organization, SAP Distribution Channel, SAP Division, Address, City, Zip Code, State, Country, Phone #, and Fax #.



4 In SAP, navigate to the SAP Reference listed for Sales Area in Table 14.

5 In the Siebel application, create one organization for every sales area in SAP.

6 Create a Siebel name for the organization.

TIP: Your Organization name should indicate the Sales Area in SAP by including the ID or text values for the sales organization, distribution channel, and division that are associated to this organization. For example, if you have a sales organization of 0010 (European), a distribution channel of 10 (Retail) and a division of 10 (Hardware), your organization name might be 0010:10:10 or European/Retail/Hardware.

7 Assign the SAP Sales Organization, SAP Distribution Channel, and SAP Division to the organization.

Name, SAP Sales Organization, SAP Distribution Channel, and SAP Division are required entries and should be unique.

Assigning SAP Plants

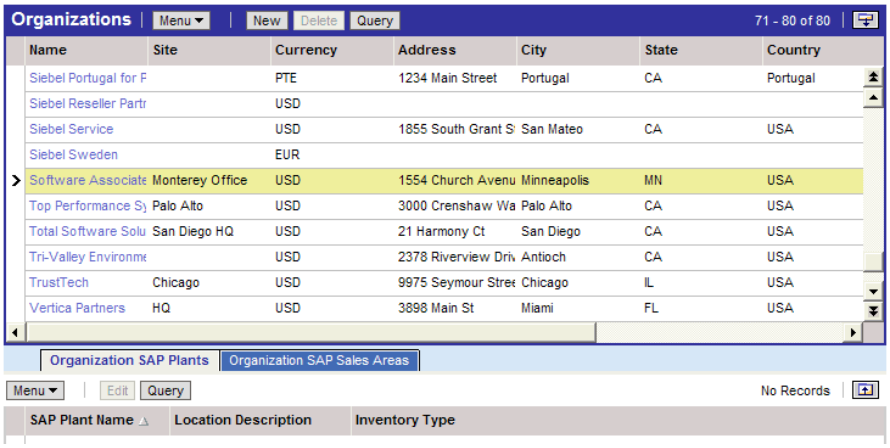
The following procedure requires the simultaneous use of both the SAP and Siebel applications. The steps in this procedure indicate when to switch applications.

To assign SAP plants to Siebel organizations

1 In the Siebel application, navigate to Administration - Integration.

2 Select Organization SAP Plants.

The following figure shows the Organizations list with a record selected. Under that is the Organization SAP Plants form and its fields: SAP Plant Name, Location Description, and Inventory Type.



3 In SAP, navigate to the SAP Reference listed for SAP Plants in Table 14.

4 In the Organization list, select a Siebel organization.

5 In the Organization SAP Plants list, add a new record for the plant to assign to this organization.

6 Enter the SAP Plant Name and Location Description fields.

The Inventory Type field fills in automatically.

NOTE: Use identical names for the plant in SAP and the Siebel application.

7 Repeat for all plants defined in SAP.

Maintaining SAP Reference Customers

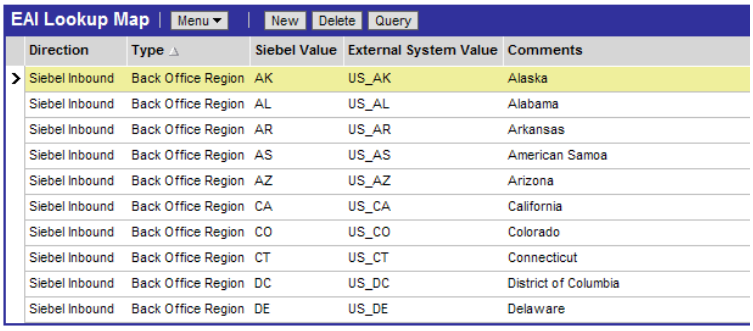
You need to update SAP reference customer information. In the Siebel application, a reference customer is required for each of the four predefined account groups Sold-To Party, Ship-To Party, Bill-To Party, and Payer. These customer to account group mappings are defined in the Siebel application as EAI Value Maps. Modify these EAI Value Maps to enter a customer ID from SAP to use as a customer reference. The customer reference ID is then used by the BAPI_CUSTOMER_CREATEFROMDATA1 BAPI in the Account to Customer standard integration as the reference customer for this account group.

To modify EAI Value Maps for reference customers

1 Navigate to Administration - Integration.

2 Select EAI Value Maps.

The following figure shows the EAI Lookup Maps list with its fields: Direction, Type, Siebel Value, External System Value, and Comments.



EAI Lookup Map | Menu | New Delete Query

Direction	Type	Siebel Value	External System Value	Comments
Siebel Inbound	Back Office Region	AK	US_AK	Alaska
Siebel Inbound	Back Office Region	AL	US_AL	Alabama
Siebel Inbound	Back Office Region	AR	US_AR	Arkansas
Siebel Inbound	Back Office Region	AS	US_AS	American Samoa
Siebel Inbound	Back Office Region	AZ	US_AZ	Arizona
Siebel Inbound	Back Office Region	CA	US_CA	California
Siebel Inbound	Back Office Region	CO	US_CO	Colorado
Siebel Inbound	Back Office Region	CT	US_CT	Connecticut
Siebel Inbound	Back Office Region	DC	US_DC	District of Columbia
Siebel Inbound	Back Office Region	DE	US_DE	Delaware

3 Edit all entries whose Type is SAP Reference Customer.

There are four entries included as part of your Siebel installation.

4 In each entry modify the External System Value field to contain the customer ID for your reference customer for this account group.

The account group name appears in the Siebel Values field.

The current standard integrations also use the EAI Value Map types. Although unlikely, it is possible to update these due to configuration changes in your SAP implementation. You can view these maps by navigating to Integration Administration, selecting EAI Value Maps, and then querying for SAP*.

Activating Workflows

The workflows provided in the standard integrations need to be activated before they can be executed.

To activate a workflow

1 In the Siebel client, navigate to Administration - Business Process.

2 Select Workflow Deployment.

3 Query for all SAP Workflows by querying for those whose names fit the pattern *SAP*46C*.

4 Select each workflow returned by the query, and then revise and activate it.

Installing Siebel Connector for SAP R/3 on Windows for a UNIX-Based Siebel Enterprise

If your enterprise installation uses a mix of UNIX-based and Windows-based servers, you can implement Siebel Connector for SAP R/3 and share the connection within the enterprise. The Siebel Connector for SAP R/3 must be installed and implemented on a Windows-based server.

Prerequisite for Installing Siebel Connector for SAP R/3 on a Mixed Operating System

Install Samba on the AIX or Solaris server. This makes the AIX and Solaris files accessible to the Windows servers.

Installing Siebel Connector for SAP R/3 on a Mixed Operating System

Use the following high-level steps to integrate the mixed environment.

To install Siebel Connector for SAP R/3 for mixed operating system enterprises

1. Install the Siebel Gateway Name Server and Siebel Server on a UNIX environment.

   For UNIX installation instructions, see *Siebel Installation Guide for UNIX: Servers, Mobile Web Clients, Tools*.

   NOTE: Complete all UNIX server installations before installing on any Windows-based servers.

2. During Siebel Server installation on a Windows environment, do the following:
   a. When prompted for Co-located Siebel Gateway Name Server, clear the check box for Yes.
   b. Set Gateway Name Server Hostname to the name of the Gateway Name Server name and the Gateway Name Server port name, using a colon (:) as a separator. Example: *GatewayServer1:GatewayPort*.
   c. Set Enterprise for this server to the name of the enterprise.
   d. Enable the SAP connector and EAI Components component groups.
   e. Test the database connection.
   f. Disable the following server components on UNIX:
      - Business Integration Manager
      - Business Integration Batch Manager
      - SAP BAPI tRFC Receiver
      - SAP IDOC AMI Receiver for MQ Series
3. If you use mobile clients, then modify the cfg files so they direct the request server to a Windows-based Siebel server.
4. In the Siebel application, choose Navigation > Site Map > Administration - Server Configuration.

a Click Enterprises from the link bar.

b Click the Synchronize view tab.

c Click the Synchronize button.

5 Restart all Siebel servers.

4 Standard Integrations

The Siebel Connector for SAP R/3 has the following standard integrations:

Siebel Account/SAP Customer Integration

Two Standard Integration scenarios exist for integrating Siebel Account and SAP Customer information. You can create customer/account data in either Siebel applications or in SAP R/3 and distribute this data to the other application. This section describes these two standard integrations.

Executing Customer to Account Data Flows

This section describes how to transfer customer master data from SAP R/3 to Siebel applications. To execute the Customer to Account Data Flow, first make sure you have gone through the installation steps as described in Chapter 3, "Installation and Configuration." In particular, be sure that you have completed all ALE configuration for the message type DEBMAS flowing from SAP-to-Siebel, you have checked your tRFC BAPI Receiver connectivity, and you have entered sales areas in the Siebel database.

The flow of IDOCs from SAP can be triggered in many ways. A simple way to send a Customer IDOC from SAP is to use transaction BD12. From this transaction enter the Customer ID, Message Type DEBMAS, and your external logical system, and then click Execute. A DEBMAS05 IDOC should be sent to the Siebel application. The new account appears in the Accounts list.

NOTE: The Siebel Connector for SAP R/3 includes standard integrations for use with SAP R/3 4.6C. The Business Service Data Map expects to receive a 4.6C DEBMAS05 IDOC. If you are using a version of SAP newer than 4.6C you can send a 4.6C IDOC to the Siebel application by setting the version to "46C" in the appropriate Outbound Parameter for the Partner Profile you have created for DEBMAS. You may need to modify this Standard Integration if you are unable to send a 4.6C IDOC to the Siebel application.

Executing Account to Customer Data Flows

This section describes how to transfer accounts and contacts information from the Siebel application to SAP R/3. Before executing the Account to Customer Data Flows, first make sure you have completed the installation steps as described in Chapter 3, "Installation and Configuration." In particular, be sure that you have completed all ALE configuration for the message type DEBMAS flowing inbound to SAP, you have checked your BAPI connectivity and you have entered sales areas in the Siebel database.

To create an SAP account in the Siebel application, use the standard Siebel account list.

To maintain general account information

1. Click the Accounts tab.
2. Click Accounts List.

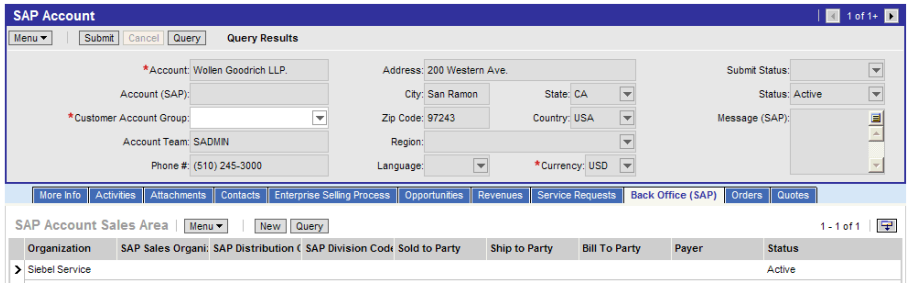
   The Account list appears.
3. To create accounts:
   a. Click New.
   b. Fill out the Account form.
4. To edit accounts:
   a. Select an account.
   b. Modify the Account form.
   c. Select a different record.

To maintain SAP customer-specific information (Sales Area data)

1. Navigate to the Accounts tab.
2. Click the Accounts List.
3. Select the account from the list of accounts.

4 Click the Back Office (SAP) View Tab. If the BackOffice (SAP) Tab is not visible, access it from the Site Map.

The SAP Account form appears. The following figure shows the SAP Account form. In the Back Office (SAP) view tab, the SAP Account Sales Area list displays details for the selected record such as Organization, SAP Sales Organization, SAP Distribution Channel, SAP Division Code, Sold To Party, Ship To Party, Bill To Party, Payer, and Status.



5 Fill out the account information including the mandatory Customer Account Group.

Choose from Sold-To Party, Ship-To Party, Bill-To Party, or Payer.

6 Scroll down to the SAP Account Sales Area.

The SAP Account Sales Area list appears.

7 Fill out the list fields.

In Siebel applications an account is assigned, by default, to an organization based on your position.

8 Record the organization you want to assign to this account. This populates the appropriate SAP Sales Area.

9 To add a new organization, click New, and repeat Step 8.

Account/Customer Integration Limitations

The Account/Customer standard integration supports the following SAP Standard Customer Account Groups: Sold-To Party, Ship-To Party, Bill-To Party, and Payer. When you submit a new account to SAP, SAP assigns it a customer number. After SAP assigns the customer number you cannot change the customer account group assignment in the Siebel application. If the account group for a customer in SAP changes, this information is updated in the Siebel application when the customer is sent to the Siebel application.

Phone numbers from SAP are imported with no formatting changes into the Siebel database. The Siebel application user interface attempts to display the phone number in the North American format, unless the number begins with a plus (+) sign. If the number begins with a plus sign, the user interface displays the number as an international number, beginning with a country code.

Product/Material Integration

One standard integration exists for integrating SAP Materials and Siebel products. The product information in Siebel applications is based on the SAP Sales Views for material master data.

To execute the Material to Product Data Flow, first make sure that you have completed the installation steps as described in Chapter 3, "Installation and Configuration." In particular, be sure that you have completed all ALE configuration for the message type MATMAS flowing outbound from SAP, checked your tRFC BAPI Receiver connectivity, and entered sales areas in the Siebel database.

The flow of IDOCs from SAP can be triggered in many ways. A simple way to send a Material IDOC from SAP is to use the transaction BD10.

If a product in SAP is assigned to a particular SAP Sales Organization/Distribution Channel (no division is specified) then this product is automatically released for existing sales areas in this combination of Sales Organization/Distribution Channel.

If a Division is assigned in the SAP Material Master, the product is released for the corresponding Organization (SAP Sales Area) in the Siebel application.

From this transaction enter the Material ID, Message Type "MATMAS" and your external logical system and then click Execute. This sends a MATMAS03 IDOC. You can view the new product created under the Products list. To see the product organization and plant assignments navigate to Organization and Inventory Location. Figure 3 shows the Products form with the Organizations view tab active. This displays Organization, Primary Plant, Unit of Measure, Minimum Order Units, Currency Code, Orderable, Status, SAP Sales Organization, and SAP Distribution Channel.

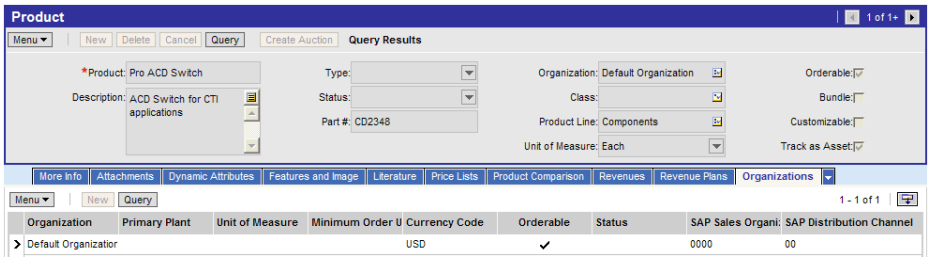


Figure 3. Product Form with the Organization Tab Active

NOTE: Standard Integrations have been created for use with SAP R/3 4.6C. The Business Service Data Map expects to receive a 4.6C MATMAS03 IDOC. If you are using a different version of SAP, this mapping may fail. If you are using a version of SAP newer than 4.6C, you can send a 4.6C IDOC to the Siebel application by setting the version to 46C in the appropriate Outbound Parameter for the Partner Profile you have created for MATMAS. You may need to modify this Standard Integration if you are unable to send a 4.6C IDOC to the Siebel application.

Sales Order Integration

The Siebel Connector for SAP R/3 has the following standard integrations for integrating Siebel and SAP sales orders.

Siebel Sales Order to SAP Sales Order Standard Integration

The following topics explain the Siebel Sales Order to SAP Sales Order standard integration.

Executing Siebel Sales Order to SAP Sales Order

This section describes how to transfer sales orders from Siebel applications to SAP R/3. It also describes how to validate the order and get the real-time status of sales orders from SAP. To execute the Siebel Sales Order to SAP Sales Order data flow, first make sure you have completed the installation steps as described in Chapter 3, "Installation and Configuration." and all data required for a successful Sales Order Creation (Master Data, Sales Enterprise structure) are in sync with SAP.

The Sales Order integration with SAP has two modes, Validate Order and Place Order.

Validate Order

This updates information on Credit Status Of Customer, Billing Block, Delivery Block, Pricing Information, and Delivery Proposal/Availability. No orders are created on the SAP side, but Siebel data is updated with the preceding information from SAP.

Place Order

When you are satisfied with the information in your Sales Order, you can submit sales orders to SAP by clicking Place Order (on the SAP Line Item view). This submits your Siebel sales order to SAP and updates Pricing Data, Delivery, Status information, and SAP Order Number. This may be different from what you got with Validate if there is a time difference between validating and submitting the sales order. After you submit a Siebel sales order to SAP you cannot resubmit the same Siebel sales order to SAP again. To avoid inconsistencies between SAP Sales Order and Siebel Sales Order, implement the state model to prevent any changes in the Siebel sales order after resubmission. For more information on the state model, see *Siebel Business Process Designer Administration Guide*. After submitting an order to SAP you can still make changes in SAP using the Get Status button (invoking Get Status Data flow). You can also automate the order status update process by configuring SAP to dispatch IDOCs.

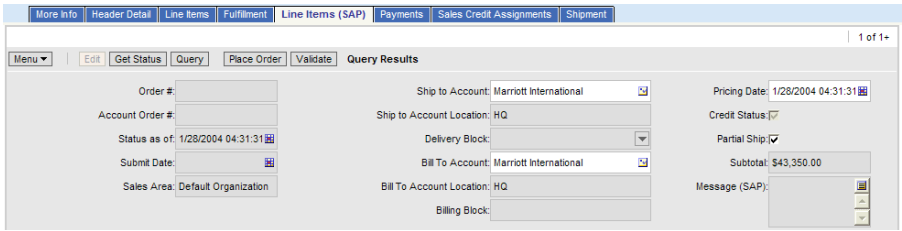
Only Sales Order (Type OR/TA)

In standard integration, the Siebel Connector for SAP R/3 supports the Sales order (Order Type OR in SAP). Order type "TA" is preset for SAP, which is recommended by SAP for order type OR irrespective of language. To extend the Siebel SAP Integration for further SAP document types (RMA, Contracts, and so on), read Appendix E, "Creating Integration Touch Points."

To execute the data flow in the Siebel application

1. Navigate to the Sales Order List View.
2. Create a new Order Header or select an existing Sales Order.
3. Enter your sales information in the form, review entries, and then save the record.
4. If your customer wants to pay with a credit card:
   - **a** Enter an order number. The Order Item view appears.
   - **b** Click the Payments view tab and scroll down to the Payment Detail - Credit Card applet.
   - **c** Enter the following information:
     - Payment Type (Visa, MasterCard, and so on)
     - Credit Card Number
     - Credit Card Holder
     - Expiration Month
     - Expiration Year
     - Payment Method (credit card)
5. Click the Line Items (SAP) view tab. If this tab is not visible in the current view, access it from the site map.

   The following figure shows the Line Items (SAP) view tab with the following fields in a form: Order #, Account Order #, Status As Of, Submit Date, Sales Area, Ship to Account, Ship to Account Location, Delivery Block, Bill to Account, Bill to Account Location, Billing Block, Pricing Date, Credit Status, Partial Ship, Sub Total, and Message (SAP).



   - **a** Add or modify the SAP-specific information in your Order Header.

     Make sure the correct SAP Sales Area is assigned.
   - **b** Create a new line item or select an existing one.
   - **c** Enter line item data, review the entries, then save the record.

6 To validate or create a Sales Order in SAP, click Validate or Place Order.

7 To update the sales order Status Information, click Get Status.

8 To get more detailed Delivery Information for a selected line item, navigate to the Schedule Line view tab.

Siebel Sales Order to SAP Sales Order Limitations

The following limitations apply to Siebel sales order to SAP sales order integration.

Schedule Lines (Delivery Proposal). Schedule Lines cannot be sent from the Siebel application to SAP. They come only from SAP. If you want to send different delivery proposals you must create different line items for the dates and quantity required.

Sales Area. The sales area is limited by the selected primary organization in the Organization field. In the Siebel application, organization is the equivalent of SAP's sales area. You can assign multiple organizations for Siebel quote and orders but SAP accepts only one sales area corresponding to primary organization of order/quote. If you select the wrong primary organization, you may get one of the following messages, depending on the SAP configuration:

- `Sold-to party ABCD not maintained for sales area XXX YY ZZ.`
- `Division XX in the header deviates from division YY in the item.`
- `No Pricing procedure could be determined for Sales Area.`

Check Accounts > Back Office (SAP) to see if the account / primary organization combination selected in the order is there.

Partner Types. The Siebel Connector for SAP R/3 supports Sold To Party, Ship to Party, Bill to Party, and Payer Partner types for sales orders and the data flows are preset for these fields. If you require additional partner types, make sure to modify the data flow maps accordingly.

Get Status. Get Status does not update the Billing Block and Credit Status fields, which are only updated by Validate Data flow.

Quote to Sales Order Standard Integration

The following topics explain the quote to sales order standard integration.

Executing Quote to Sales Order

This topic describes how to transfer a Siebel quote to an SAP sales order. It also describes how quotes can be validated. To execute the Quote to Sales Order data flow, first make sure you completed the installation steps as described in Chapter 3, "Installation and Configuration." and executed Account to Customer and Products to Material data flows successfully. You should have set up products and sales area accounts before you submit sales orders to SAP.

To validate quotes

1 Press Validate on the SAP Line Item view.

This is similar to Validate on Sales Order. Invoking the Validate data flow on quote returns Credit Status, Pricing information, and Delivery Proposal (Schedule Lines in Schedule Line form) only. Delivery block and billing block information is not provided, but can be added if required by modifying the applet (both the fields are in the underlying database and Bus Comp).

2 Press auto order to convert the Siebel quote to a Siebel order when you are satisfied with the Delivery Proposal and pricing.

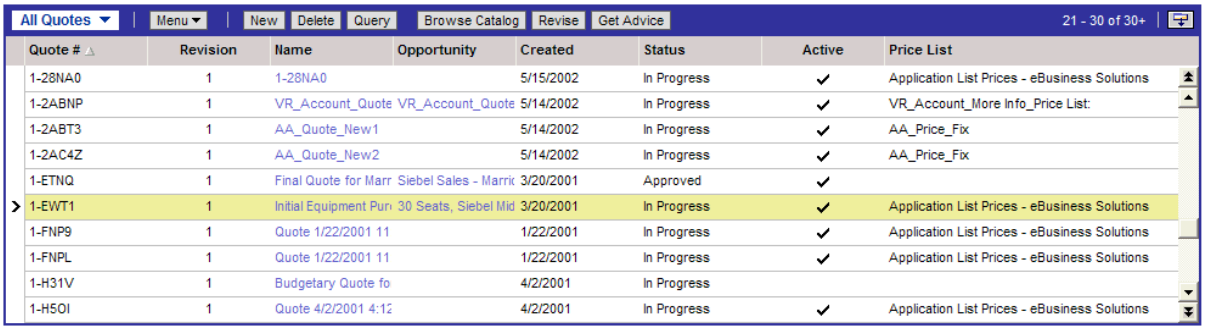
To execute the data flow in the Siebel application

1 Navigate to the Quotes tab.

2 To create a new Quote Header, click New or select an existing Quote.

3 Enter information in the list, review entries, and then click Save.

The following figure shows the Quotes list and its fields: Quote #, Revision, Name, Opportunity, Created, Status, Active, and Price List.



Quote #	Revision	Name	Opportunity	Created	Status	Active	Price List
1-28NA0	1	1-28NA0		5/15/2002	In Progress	✓	Application List Prices - eBusiness Solutions
1-2ABNP	1	VR_Account_Quote	VR_Account_Quote	5/14/2002	In Progress	✓	VR_Account_More Info_Price List:
1-2ABT3	1	AA_Quote_New1		5/14/2002	In Progress	✓	AA_Price_Fix
1-2AC4Z	1	AA_Quote_New2		5/14/2002	In Progress	✓	AA_Price_Fix
1-ETNQ	1	Final Quote for Marr	Siebel Sales - Marric	3/20/2001	Approved	✓	
1-EWT1	1	Initial Equipment Pur	30 Seats, Siebel Mid	3/20/2001	In Progress	✓	Application List Prices - eBusiness Solutions
1-FNP9	1	Quote 1/22/2001 11		1/22/2001	In Progress	✓	Application List Prices - eBusiness Solutions
1-FNPL	1	Quote 1/22/2001 11		1/22/2001	In Progress	✓	Application List Prices - eBusiness Solutions
1-H31V	1	Budgetary Quote fo		4/2/2001	In Progress		
1-H5OI	1	Quote 4/2/2001 4:12		4/2/2001	In Progress	✓	Application List Prices - eBusiness Solutions

4 If your customer wants to pay with a credit card:

- **a** Click the Payments view tab.
- **b** Create a new Payment Line record with credit card as the method of payment.
- **c** Scroll down to Payment Detail - Credit Card applet.
- **d** Enter the following information:
  - ❑ Card Name
  - ❑ Card Number
  - ❑ Card Holder
  - ❑ Expiration date

5 Click the Line Items (SAP) view tab.

- **a** Add or modify the SAP-specific Information in the Quote Header.

  Make sure that the correct SAP Sales Area is assigned.

b Create a new line item or select an existing line item.

c Enter line item data, review the entries, then save the line item.

6 To validate a quote in SAP, click Validate.

7 To Create a Siebel sales order, click on the orders tab and then click on the Sales order.

All orders generated for the selected quote should be visible in this view. Drilling down on the latest order takes to you to the order screen. Here you can choose the line items (SAP) tab and validate and create SAP orders as described in the previous section.

8 To get more detailed delivery information for a selected line item, navigate to the Schedule Line tab.

Siebel/SAP Pricing Integration for Quotes and Orders

To execute pricing in Siebel quotes and orders two scenarios are possible:

- SAP as the price master
- Siebel application as the price master

To execute the Siebel/SAP price integration make sure you have completed the installation steps as described in Chapter 3, "Installation and Configuration."

To add shipping and billing instructions in Quotes and Orders

1 Click the Fulfillment view tab.

2 Choose Shipping Terms from the predefined drop-down list.

For Incoterms Part 1 use the Shipping Terms field, and for Incoterms Part 2 use the Shipping Info field.

3 Navigate to the Payment tab.

4 Choose Payment Terms from the predefined drop-down list.

Siebel Application as Price Master

You maintain your pricing data in the Siebel application. For more information about Siebel Pricer, refer to *Pricing Administration Guide*. Assign a price list in the Quote/Order Header or enter a manual price in the Discounted Price field. A flag is checked to let you know that the Siebel application is the pricing master. SAP then calculates with the Net Price sent from the Siebel application.

For standard integration, the Siebel application uses SAP to calculate the tax. To send freight charges to SAP, click the Total view tab, and enter the Freight Costs into the Shipping Charges field.

Condition Type. Table 15 and Table 16 show the default field mappings used when the Siebel application is the price master. You may need to change them depending on your SAP configuration settings.

Table 15. Mapping Siebel Fields and Values to SAP BAPI Fields When the Siebel Application Is Price Master

Siebel Field / Value	Data Flow	SAP BAPI Field
SAP Condition Type KF00	Submit	ORDER_CONDITIONS_IN - COND_TYPE
Freight Amount (if <> 0 or <> Null)	Submit	COND_VALUE
SAP Condition Type KF00	Submit	ORDER_CONDITIONS_IN - COND_TYPE
Item Price	Submit	COND_VALUE
SAP Condition Type KF00	Simulate	ORDER_ITEMS_IN-CD_TYPE3
Freight Amount (if <> 0 or <> Null)	Simulate	ORDER_ITEMS_IN-CD_VALUE3
SAP Condition Type KF00	Simulate	ORDER_ITEMS_IN-CD_TYPE2
Item Price	Simulate	ORDER_ITEMS_IN-CD_VALUE2

The default mappings in Table 16 are only valid in Simulate Data Flow. After you submit an order to SAP, SAP becomes the pricing master.

Table 16. Mapping SAP BAPI Fields to Siebel Fields and Values When the Siebel Application Is Price Master

SAP BAPI Field	Siebel Field/Value
ORDER_ITEMS_OUT-SUBTOTAL_5	Tax Amount
ORDER_ITEMS_OUT-NET_VALUE	Adjusted List Price * Quantity Requested

GETSTATUS. When you submit a sales order, SAP takes the ownership of the price data. Any changes you make to the pricing relevant data is updated in the Siebel application.

SAP as Price Master

In this scenario you maintain your pricing master data (price lists, discounts, and so on) in SAP. When you validate or submit a quote or an order to SAP, SAP calculates the Line Item price. This price is sent back to the Siebel application together with freight and tax information. Table 17 and Table 18 show the default field mappings used when SAP is the price master.

Table 17. Mapping Siebel Fields and Values to SAP BAPI Fields When SAP Is Price Master

Siebel Field/Value	Data Flow	SAP BAPI Field
SAP Condition Type PR00	Simulate	ORDER_ITEMS_IN-CD_TYPE2
SAP Condition Type PR00	Submit	ORDER_CONDITIONS_IN - COND_TYPE

Table 18. Mapping SAP BAPI Fields to Siebel Fields and Values When the SAP Is Price Master

SAP BAPI Field	Data Flow	Siebel Field/Value
ORDER_ITEMS_OUT-SUBTOTAL_4	Simulate	Freight Amount
ORDER_ITEMS_OUT-SUBTOTAL_5	Simulate	Tax Amount
ORDER_ITEMS_OUT-NET_VALUE	Simulate	Adjusted List Price * Quantity Requested
ORDER_CONDITIONS_OUT- COND_VALUE (where ORDER_CONDITIONS_OUT- COND_TYPE == PN00 or PR00 with first preference to PN00)	Get Status	Adjusted List Price
ORDER_CONDITIONS_OUT- COND_VALUE (where ORDER_CONDITIONS_OUT- COND_TYPE == KF00)	Get Status	Freight Amount
ORDER_CONDITIONS_OUT- COND_VALUE (where ORDER_CONDITIONS_OUT- COND_TYPE == UTXJ)	Get Status	Tax Amount

Sales Order Updates Standard Integration

This section describes how to update Siebel sales orders when the sales order status changes in SAP. To execute the Sales Order Status Change data flow, first make sure you have completed the installation steps as described in Chapter 3, "Installation and Configuration." In particular, be sure that you have completed all ALE configuration for the message type ORDCHG flowing outbound from SAP, you have checked your tRFC BAPI Receiver connectivity, and you have successfully created sales area account and SAP products in the Siebel application. To test the update, you should have successfully sent a Siebel sales order to SAP.

You need to perform some customization in SAP to send the IDOC, ORDERS05, to Siebel Connector for SAP R/3 whenever there is a change in an order, or new delivery is created or modified. You can also send a test ORDERS05 IDOC using SAP transaction we19. Siebel Connector for SAP R/3 receives the ORDERS05 IDOC and invokes the Get Status Dataflow for the order sent. As this process is based on the Get Status data flow for sales orders, the limitations of the Get Status data flow are applicable to this process as well.

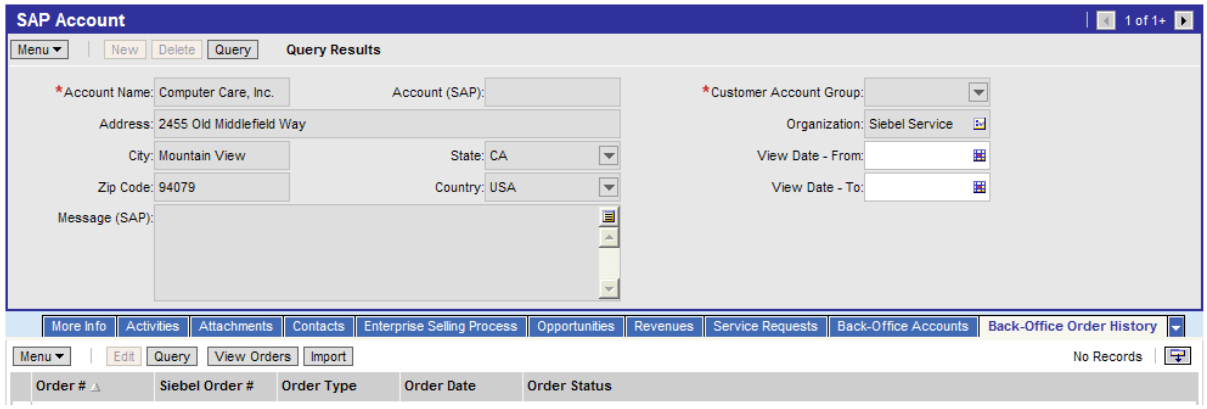
Account Order History Standard Integration

This describes how to view and import the SAP sales orders for a specified Siebel account.

To execute the Account Order History in the Siebel Application

1. Navigate to the Accounts tab.
2. Click My Accounts on the link bar.
3. Select an account and click the Back Office Order History view tab.

   The following figure shows the SAP Account form and the Back Office Order History view tab with its fields in a list: Order #, Siebel Order #, Order Type, Order Date, and Order Status.



4. Specify a date range using the View Data - From and View Data - To fields.
5. Specify the primary SAP Sales Area using the Organization field.
6. Click View Orders to retrieve SAP Orders (header information only).

   The selected orders appear.

The Import option obtains the order details of the selected order (from the View Orders option) and creates a Siebel order.

To import order details

1. In the Back Office Order History list, select an SAP order.

2 Click Import.

The Siebel order number is assigned to the SAP order.

3 To display Siebel order details, click the Siebel order number.

Remote Client Sales Order Synchronization

Incoming transactions from SAP to the Siebel application are routed to mobile users according to the visibility rules defined in the Siebel repository. In addition, Siebel mobile users' transactions are queued in their local databases. When these mobile users synchronize with the Siebel Server, their transactions are placed in the server queue and subsequently completed with SAP through the interfaces described in this guide.

To avoid multiple submissions of orders to SAP from remote clients, use wrapper workflow Order - Create SAP 46C Order (SAP). This workflow should be invoked in Workflow Manager and calls BIM after validation. This workflow prevents the duplication of orders that might occur with calling the create workflow directly (when called remotely). After making sure that no previous Order Submission request is pending, this workflow executes EAI Business Integration Manager (Server Request) (SAP) and calls Order - Create SAP 46C Order and Get Detailed List workflow. EAI Business Integration Manager (Server Request) (SAP) is a copy of the standard EAI Business Integration Manager (Server Request) Business Service with the user property mode modified from Sync to Auto. As its name suggests, Auto mode automatically detects whether it is a Server or Remote client and responds accordingly.

NOTE: For remote client Sales Order synchronization you must assign a category for the catalog of the products used in the order.

For further details on categories and catalogs, read *Applications Administration Guide*.

5 Customizing Integrations

To augment the standard integrations, you may want to modify them or create your own Custom Integrations between Siebel applications and SAP. This chapter describes the process of modifying standard integrations and creating custom integrations.

Plan your development process according to the following process flow. Information on each step in the process is provided. An example of creating a new integration touch point is provided in Appendix E, "Creating Integration Touch Points."

1. "Modifying Standard Integration Interfaces" on page 72.
   - a. Understanding the standard integrations.
   - b. Modifying business service data maps.
2. "Defining Your Business Interface" on page 79.
   - a. Determine the appropriate SAP interface and identify appropriate BAPIs, RFCs, or IDOCs to use.
   - b. Identify the appropriate Siebel business object.
   - c. Determine the data mapping between the Siebel business object and the SAP data object.
3. "Making Necessary Siebel Application and SAP R/3 Customizations" on page 87.
   - a. Extend the Siebel database, business objects, and user interface if necessary.
   - b. Customize SAP as necessary.
4. "Building the Interfaces" on page 88.
   - a. Create an integration object for the SAP data object.
   - b. Create a Siebel integration object for the Siebel business object.
   - c. Create a business service to perform the data mapping.
   - d. Create a workflow to contain the data mapping flow.
5. "Testing the Interface" on page 103.

Modifying Standard Integration Interfaces

This section provides the technical information you need to understand how the standard integrations are constructed. It also shows how to modify the business service data maps used within them.

Understanding the Standard Integration Interfaces

The following topics describe the technical implementation of each business data flow used by the standard integrations.

Appendix B, "SAP Field Mappings" contains tables with field mappings performed by the business service data maps.

Customer to Account Business Data Flow

Figure 4 shows the business data flow for the Customer to Account standard integration. In this scenario SAP is the System of Record (SOR) for Account/Customer master data. SAP initiates this flow in the form of a customer IDOC. The tRFC BAPI Receiver receives the IDOC and passes it to the Account - Receive SAP 46C Customer workflow. The workflow consists of a single business service data map that transforms the data contained in the IDOC to a Siebel integration object. This integration object instance is then passed to the Siebel adapter which updates the Siebel database.

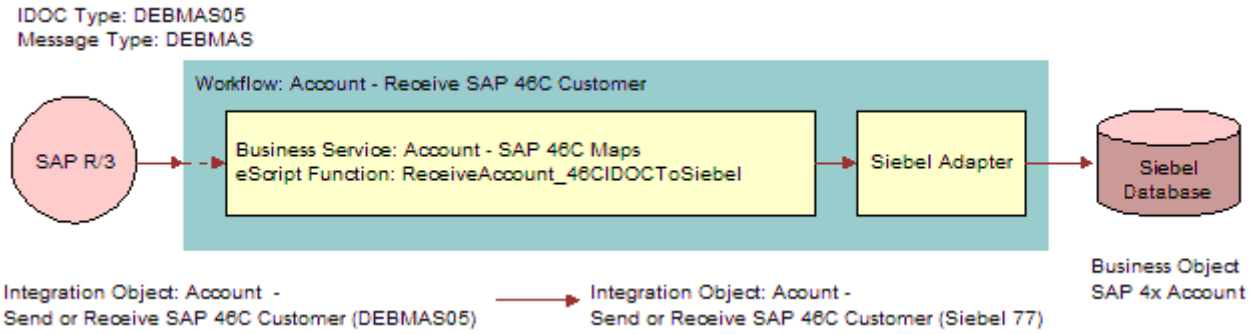


Figure 4. Business Data Flow for the Customer to Account Standard Integration

Account to Customer Data Flow

In this scenario, the Siebel application is the System of Record (SOR) for Account/Customer master data. The Account to Customer standard integration uses synchronous BAPI communication as well as SAP's ALE communication services to create and update SAP customer information from Siebel accounts.

There are four different data flows. The main workflow, Account - Submit SAP 46C Customer, orchestrates these data flows as shown in Figure 5.

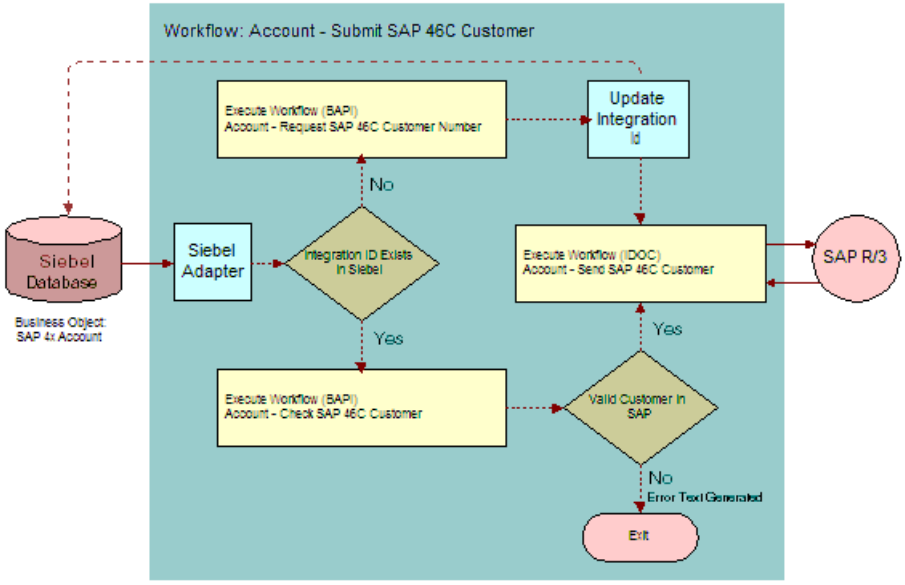


Figure 5. Business Data Flow for the Account to Customer Standard Integration

This flow does the following:

1. Retrieves the Siebel account from the Siebel database.
2. Branches on whether or not the SAP Integration ID (Customer Number) exists in the Siebel Account. This determines whether or not the account has been sent to SAP.
   - **a** If the Integration ID does not exist, then reserves a customer number for the account in SAP and sends an IDOC to SAP with the customer information.
   - **b** If the Integration ID exists, then checks for the existence of this customer in SAP (this action makes sure that any previously sent customer IDOC has been processed) and updates the customer in SAP by sending an IDOC. If the customer does not exist in SAP, then error text is generated.

Account - Request SAP 46C Customer Number Data Flow

This business data flow is called by the main workflow, Account - Submit SAP 46C Customer, when it determines that a new customer needs to be created. It uses a synchronous BAPI call to SAP to get the next available customer number. Note this BAPI does not create the customer in SAP but returns the next available customer number.

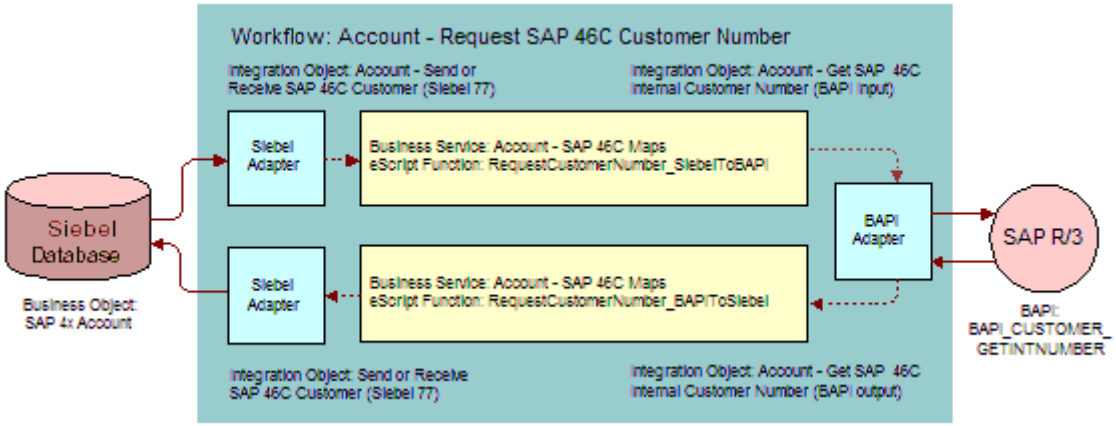


Figure 6. Business Data Flow Using BAPI Adapter

In this business data flow, the Siebel Adapter extracts the account information. It passes an integration object instance containing this data to a business service data map, which converts it into a BAPI input integration object. Then it passes the integration object instance to the BAPI adapter and makes a call to SAP to get the customer number. SAP returns the customer ID. The customer ID then resides in the BAPI Output integration object instance and this is passed to another business service data map. This business service data map creates a Siebel integration object instance with the name of the customer ID contained within it and this is used to update the Siebel database.

Account - Send SAP 46C Customer Data Flow

This business dataflow is used to create or update the customer in SAP. The account information is passed to SAP using IDOC DEBMAS05.

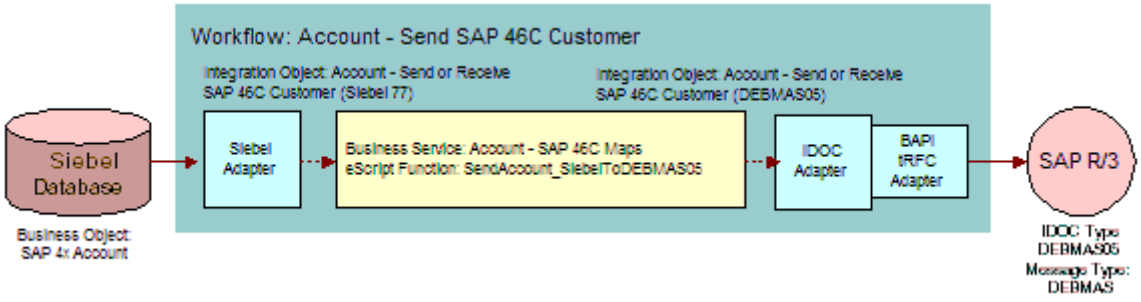


Figure 7. Business Data Flow Using IDOC Adapter

This business data flow extracts account information from the Siebel database using the Siebel Adapter and passes this to a Business Service Map in the form of an integration object. This map transforms the Siebel integration object into an IDOC integration object that is then passed to the IDOC adapter. The IDOC Adapter must be configured to call the BAPI tRFC adapter business service.

Account - Check SAP 46C Customer Data Flow

This business data flow is used to check the existence of a customer in the SAP system.

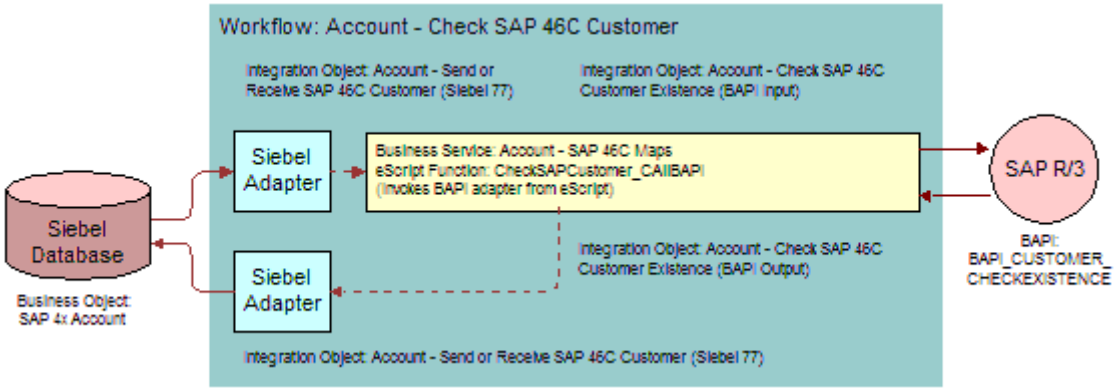


Figure 8. Business Data Flow for the Check SAP Customer Standard Integration

Material to Product Business Data Flow

Figure 9 shows the business data flow for the Material to Product standard integration. SAP initiates this flow in the form of a material IDOC of message type MATMAS03. The IDOC is received by the tRFC BAPI receiver and passed to the Product - Receive SAP 46C Material workflow. The workflow consists of the Product-SAP 46C Maps business service that transforms the data contained in the IDOC to a Siebel integration object. This integration object instance is then passed to the Siebel adapter and the Siebel database is updated.

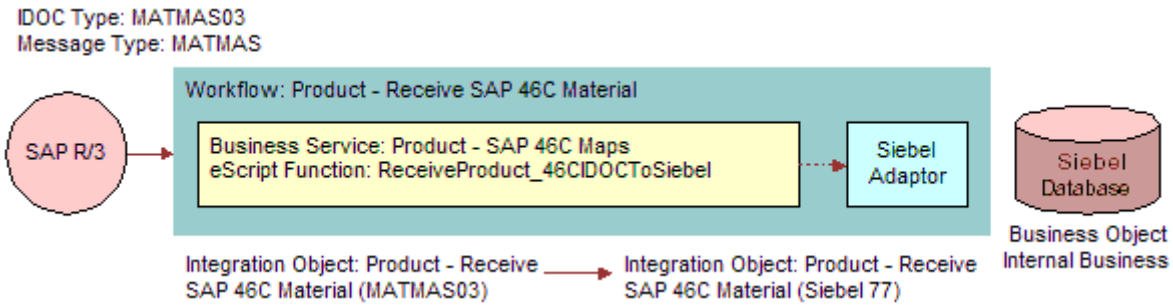


Figure 9. Business Data Flow for the Material to Product Standard Integration

Siebel Sales Order to SAP Sales Order Business Data Flows

The Siebel Sales Order to SAP Sales Order standard integration contains several business data flows. Using this interface you can perform the following actions:

- Validate the sales order.
- Submit the sales order.
- Update the sales order status.

Figure 10 shows the business data flow to validate a sales order.

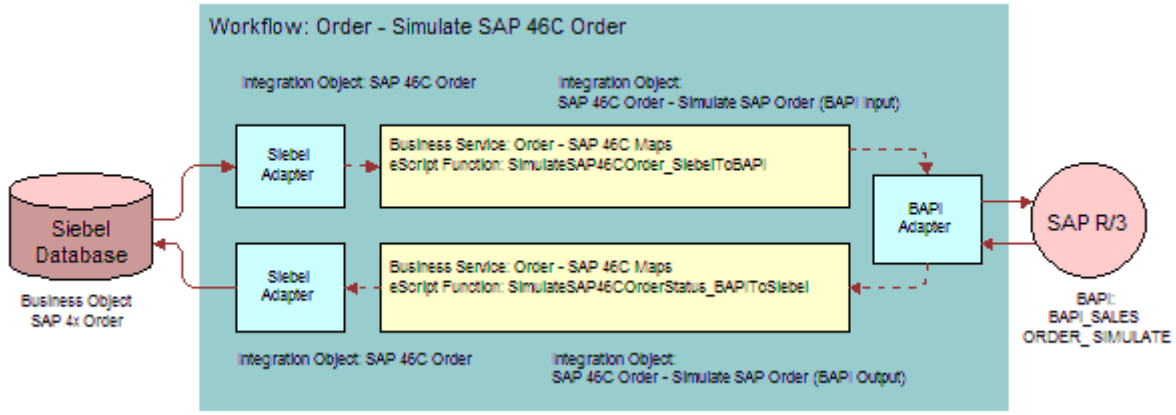


Figure 10. Business Data Flow to Validate a Sales Order

An Order is submitted from Siebel to SAP through the workflow Order - Create SAP 46C Order and Get Detailed List. This flow calls two lower level workflows: Order - Create SAP 46C Order and Order - Get Detailed List SAP 46C Order. These flows are shown in Figure 11 and Figure 12 on page 77. Figure 11 shows the business data flow for submitting the sales order to SAP.

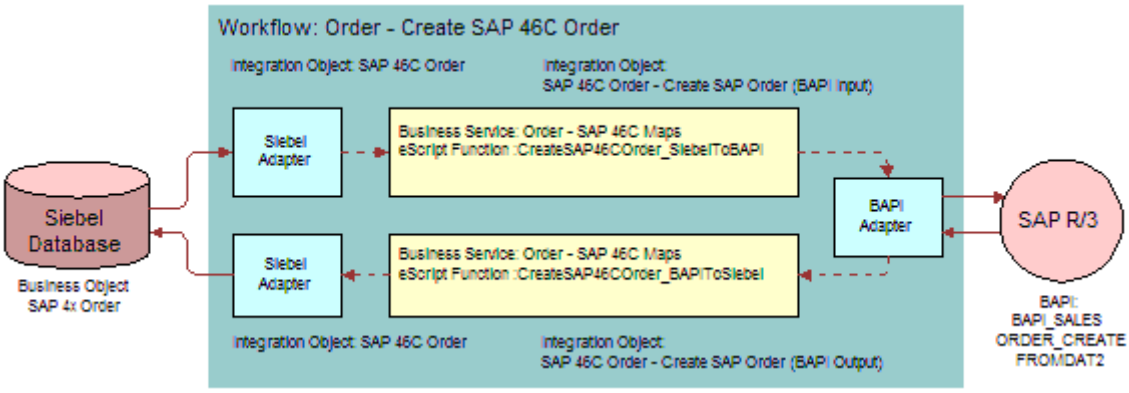


Figure 11. Business Data Flow for Submitting the Sales Order to SAP

Figure 12 shows the business data flow for retrieving sales order status. The GetSAP46CGetDetailedList_BAPIToSiebel map is dependent upon the LOV SAP_SO_PART_MOVEMENT_TYPE and uses LOV values defined to set Line Item Action Type in the Siebel application.

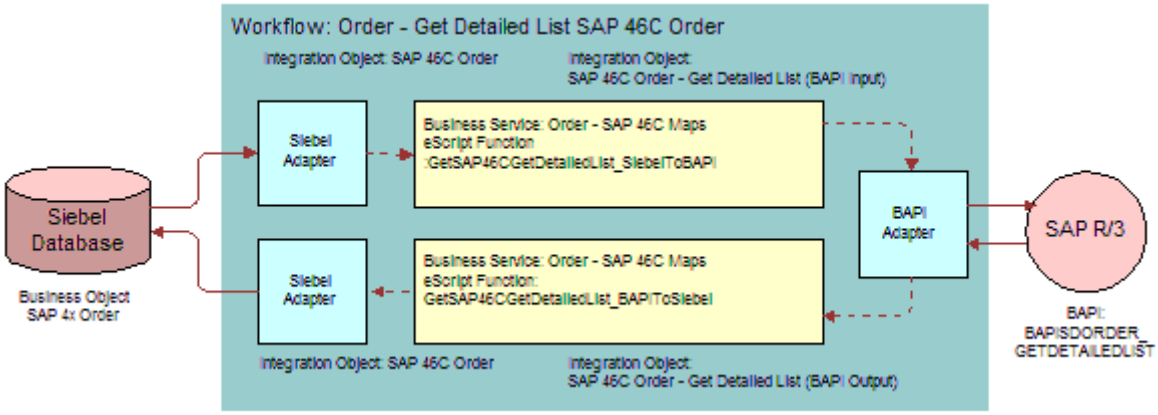


Figure 12. Business Data Flow for Retrieving Sales Order Status

The Order - Get Detailed List SAP 46C Order workflow is used both to update order status in the Siebel application and to get the detailed order information when creating an order.

Quote to Sales Order Business Data Flow

The Quote to Sales Order standard integration contains one business data flow to validate the quote. Figure 13 shows the business data flow for quote validation.

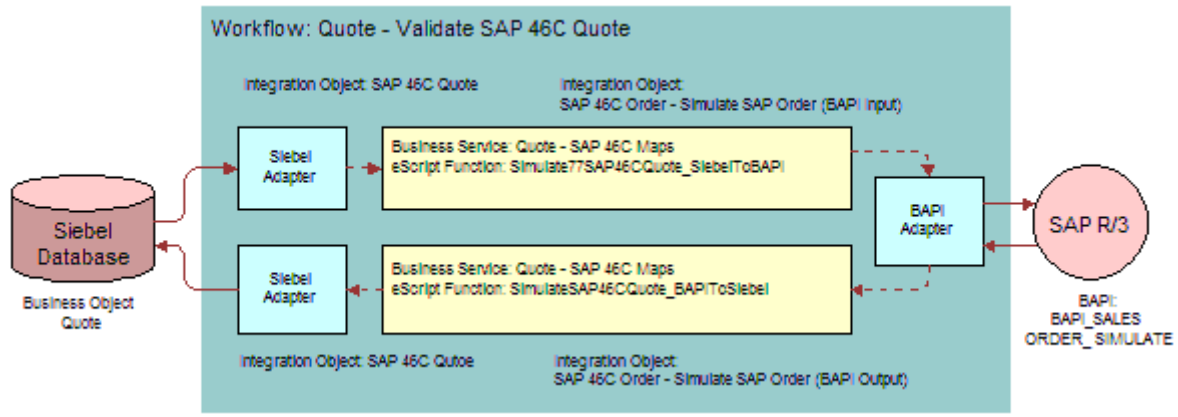


Figure 13. Business Data Flow for Quote Validation

Order Updates Business Data Flow

The Business Data Flow for the Order Updates Standard Integration is shown in Figure 14. It is triggered by the arrival of an ORDCHG message from SAP. SAP can be configured to send this message when an order in SAP has been changed. When the IDOC associated with the ORDCHG message is received by the Siebel application, the Order ID is retrieved from the message and the workflow, Order - Get Detailed List SAP 46C Order, is called to retrieve Order status from SAP and update the Siebel database. This workflow is discussed in "Siebel Sales Order to SAP Sales Order Standard Integration" on page 61.

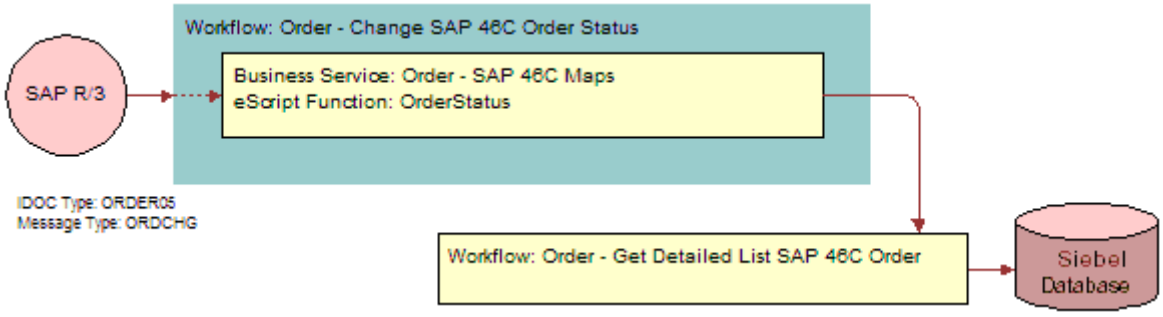


Figure 14. Business Data Flow for the Sales Order Updates Standard Integration

Account Order History Business Data Flow

Retrieval of Account Order History contains two business data flows called by the top-level workflow Account - Import SAP 46C Order and Get Detailed List SAP 46C Order. The first is shown in Figure 15, the second is shown in Figure 16 on page 79.

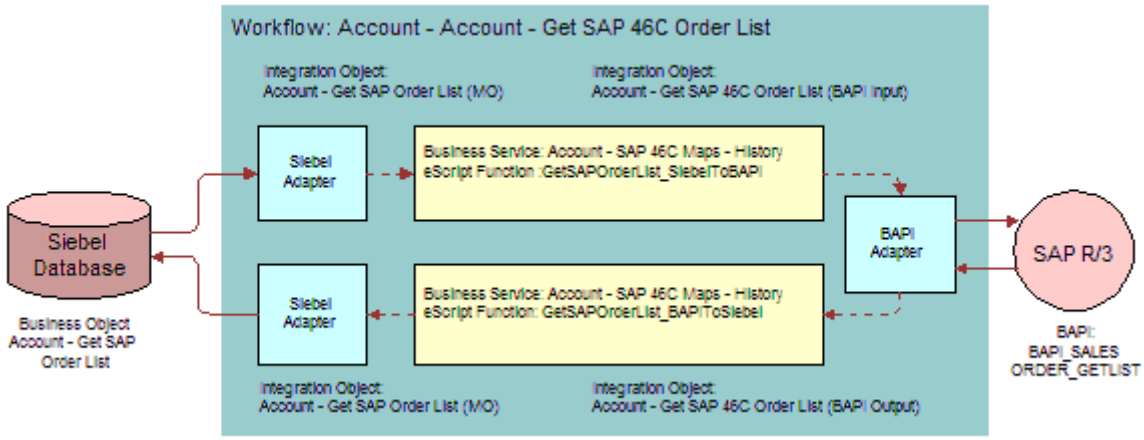


Figure 15. Business Data Flow for Get SAP Order List

This flow retrieves the list of orders from SAP when given a customer ID from an account in the Siebel database. When a Siebel user selects one of these orders, it can be imported from SAP into the Siebel database. This Business Data Flow is shown in Figure 16.

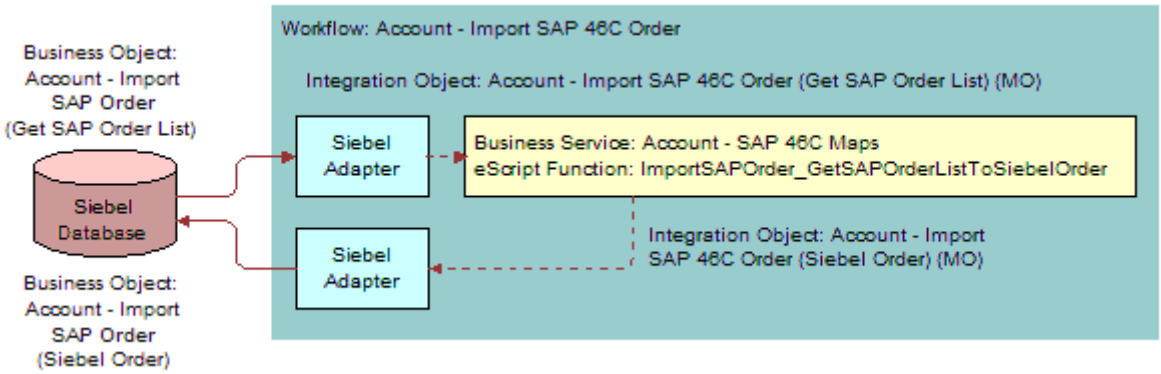


Figure 16. Business Data Flow for Importing the SAP Order

Modifying eScript Maps

You can modify the standard integrations to suit your business needs by editing the scripts for each of the business service data maps. Use Siebel Tools to edit them. When you are done with modifications, remember to compile the changes into the working .srf file.

To modify a map

1. Start Siebel Tools.
2. Click the Business Service folder in the Object Explorer.
3. Right-click a map service and choose Edit Server Scripts.

   The Scripts view opens.
4. Choose (general) from the Object list control.
5. Choose the procedure you want to modify from the Procedure list control.

   This displays one of the scripts that does the mappings.
6. Modify the script to meet your requirements.

Defining Your Business Interface

Depending upon your business requirements, you may need to develop new interfaces between the Siebel application and SAP. The first step in successfully building these new integrations is performing the necessary analysis. During your initial analysis, you should determine the following:

- The SAP interface that is appropriate for the job.

- The Siebel business object to use.
- The mapping between the chosen SAP interface and Siebel business object.

Selecting the Right SAP Interface for the Job

Siebel applications currently support two types of interfaces to SAP: BAPIs/RFCs and IDOCs through ALE.

BAPIs and RFCs

SAP R/3 provides a Remote Function Call (RFC) interface for use by external applications. This interface allows an external application to call ABAP functions defined in SAP. It also allows SAP to call functions defined in the external application. The function call interface can support the interchange of:

- Scalar parameters
- Tables
- Exception strings

Any ABAP function that is defined to be remotely callable is referred to as an RFC function and can be executed in SAP by an external application such as the Siebel application. Function calls can also be made from SAP to an external application such as the Siebel application. In that case an RFC interface that defines the function call must be created in SAP along with ABAP code to invoke the function in the external application from SAP. The function's implementation must then be in the external application.

The Siebel application takes full advantage of SAP's RFC interfaces that allow you to call functions within SAP as well as call workflows in the Siebel application from SAP through the RFC interface. Figure 17 shows the Siebel architecture surrounding SAP's RFC interface. The Siebel application to SAP data flow consists of a Siebel workflow using the Siebel BAPI adapter to communicate with the SAP R/3 Application Server. The SAP R/3 Application Server uses an ABAP function or other program to process the incoming information. The SAP to Siebel data flow consists of the SAP R/3 Application Server using an ABAP RFC function to send data to the Siebel tRFC BAPI Receiver, which uses a C++ RFC function. The data is then passed to a Siebel workflow to complete the processing.

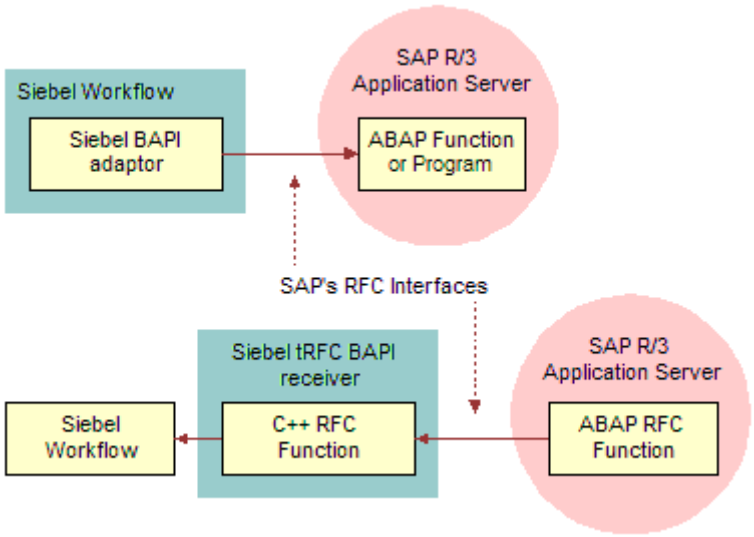


Figure 17. SAP's RFC Interface

The RFC interface is a standard part of SAP R/3. Upon the RFC foundation, SAP created the Business Application Programming Interface (BAPI). BAPIs are the methods of SAP Data Objects. While BAPIs can be viewed as part of a data object within SAP, their implementation is an ABAP function. Therefore, BAPIs, which are defined to be remotely callable, are used and invoked in the same manner as RFC functions. For this reason, the Siebel applications' interface to BAPIs and RFCs is the same.

What You Should Know About BAPIs and RFCs

When considering how best to interface to SAP keep the following information in mind.

- The Siebel application supports BAPI/RFC calls in either synchronous or transactional mode.
  - Synchronous calls contains return parameters or table information that is returned to the caller of the function.
  - Transactional calls do not contain return parameters or table information to be returned to the caller.
  - Synchronous calls are invoked immediately.
  - Transactional calls may not be invoked immediately.

- BAPI/RFC calls should be kept short in duration so that SAP is not overloaded.
- A large number of BAPI/RFC calls performed at the same time can have a very detrimental impact on SAP performance. The number of calls your system can support will be determined by your hardware and network configuration and the number of dialog processes in your application servers.
- Tables passed through a BAPI or RFC call are passed as a single block of memory, therefore passing large tables through the interface is not recommended.
- SAP's Transactional RFC (tRFC) interface is used to make transactional BAPI/RFC calls. This assures the safe and secure transit of data from the calling application to the called application.
- BAPI/RFC calls that are not made with tRFC can result in loss of data if a network problem occurs during the call. For example, if an external application makes a call to an RFC function in SAP R/3 and the network goes down before the return information is received by the caller, the external application has no way of knowing if the call completed in SAP or not.
- SAP has provided many standard BAPIs. Use a standard BAPI when possible as these interfaces are supported by SAP and change minimally from version to version. The level of support for a BAPI or RFC can be determined from the administration area of SAP transaction SE37 for the specific BAPI or RFC.
- Using SAP transaction SE37, you can create your own BAPI/RFC ABAP functions.

Finding the Right BAPI or RFC

It can be difficult to determine if SAP has a BAPI or RFC function call that is useful for your particular purpose. Here are some ways you can view information on BAPIs and RFC functions:

- BAPIs can be browsed through the SAP transactions SWO1, SWO2, and BAPI.
- SAP's web site provides information through the SAP Interface Repository http://ifr.sap.com.
- RFCs and BAPIs can be viewed through SAP transaction SE37.
- A test run of the BAPI or RFC with sample data can be done through the function debugger under SAP transaction SE37. In many cases, this is the only way of determining the exact functionality of an RFC or BAPI.

ALE and IDOCs

SAP supports the passage of Intermediate Documents (IDOCs) between external applications and SAP through Application Link Enabling (ALE). IDOCs are hierarchical structures that contain information for a data object in SAP. For example, IDOCs exist as containers for SAP customers, vendors, materials, purchase orders, quotes, and sales orders, to name just a few. The ALE technology used to pass IDOCs from one application to another is based on RFC. ALE consists of specific transactional RFC functions whose interfaces pass IDOC data containers. These functions are called using tRFC so that the safe transit of the data across the network can be assured.

What You Should Know About ALE and IDOCs

When considering how best to interface to SAP keep the following information in mind.

- ALE calls are usually not real-time
  - ALE calls out of SAP can be in immediate or batch mode.
    - If a call is made out of SAP in immediate mode, an attempt is made to send this to the external application at the time the IDOC is created using a dialog process. However, if the external application is busy, the call is placed in the tRFC queue and may be handled by background processes or by user configured batch jobs. For example, calls defined to be immediate may not actually be performed immediately. This is an important design consideration.
    - If a call is made out of SAP in batch mode, a batch job needs to be scheduled to flush the IDOCs from SAP to the external application.
  - ALE calls into SAP can use immediate or batch mode.
    - If a call is made into SAP in immediate mode, SAP adds the IDOC to the IDOC queue and application level functionality is immediately invoked to process the IDOC into SAP. This processing is often slow and can affect SAP's performance.
    - If a call is made into SAP in batch mode, the IDOC is added to the IDOC queue and application level functionality needs to be triggered with the scheduling of a background job.
- ALE works through the tRFC layer of SAP.
  - Before sending IDOCs from SAP, SAP bundles IDOCs together into a "transaction." The number of IDOCs in each transaction depends upon the packet size set in the Partner Profile.
    - Each transaction is passed to the external application as a single packet.
    - Transactions which error in transmission are kept in the tRFC layer and can be viewed with SAP transaction SM58.
    - SAP provides utilities for viewing and resending these transactions. This mechanism prevents data loss between SAP and the external application.
  - When an external application sends IDOCs into SAP, they are also sent as a transaction. If the send of this data fails, it is the external application's responsibility to resend the data as the same transaction to make sure that the data reaches SAP and is not processed twice. After SAP receives the IDOCs, they are visible in the IDOC layer.
- Because ALE works through RFC, one transaction packet is passed within a single RFC call. For this reason keep the packet size relatively small as the tables passed through the RFC call are passed as a single block of memory. This is an inherent limitation of SAP's RFC interface. A packet size of 2-3 megabytes works well. Each data record in an IDOC is a little over 1 kilobyte. For example, if you are sending IDOCs with an average of 10 data records in each IDOC, you can send 200-300 IDOCs per packet.
- SAP has provided many standard IDOCs. These can be extended in SAP using the IDOC Editor (WE30). Modification can be made to SAP's standard programs through user exits to populate IDOC information or modify the processing of the IDOC. You can also create your own custom IDOCs.

Finding the Right IDOC

SAP has developed many Standard IDOCs. In addition you can develop your own custom IDOCs. Here are some ways you can view information on IDOCs and ALE:

- IDOC Type structure and documentation can be viewed through SAP transaction WE60.
- Development of IDOCs is performed through SAP transaction WE30 and associated transactions under WEDI.
- ALE configuration is centralized under transaction SALE.
- ALE monitoring and processing can be found under transaction BALE.
- SAP's web site provides information through the SAP Interface Repository http://ifr.sap.com.

Finding the Appropriate Siebel Business Object

Siebel's database interface is based on business objects. A Siebel business object represents a data entity that may contain related data held in many tables. The business object is made up of business components that map to these tables. When you create an interface to the Siebel application, you create Siebel integration objects based on Siebel business objects. These integration objects have components created from the business components within the Siebel business object.

To create the appropriate Siebel integration object, you need to know the business object on which your integration object should be based. If the data you want to interface to in the Siebel database is visible through specific Siebel views, you can determine the business object associated with those views.

Elements of the Siebel user interface correspond to business objects and business components. Figure 18 shows this relationship between the Siebel UI Layer, Business Object Layer, and Data Object Layer. Table 19 defines these elements. The figure illustrates the three layers: UI Layer, Business Object Layer, and the Data Object Layer. The UI Layer consist of List Columns, Applets, Views, Screens, and Applications. The Business Object Layer consists of Fields, Business Components, and Business Objects. The Data Object Layer consists of Columns and Tables.

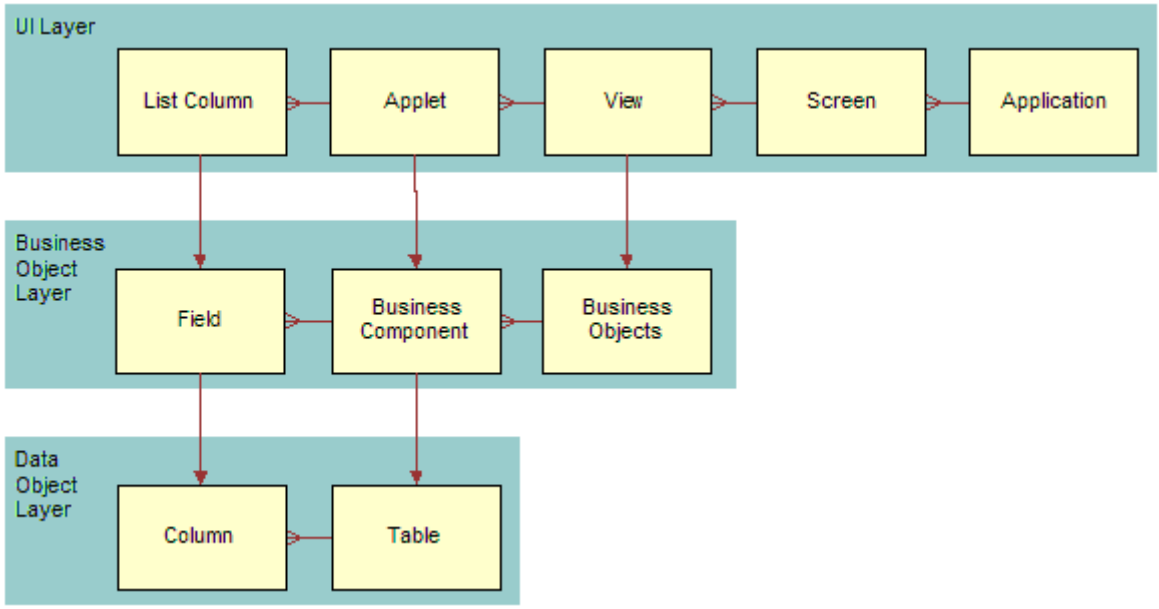


Figure 18. UI, Business Object, and Data Object Layer Relationships

Table 19. Siebel User Interface Elements Defined

Element	Definition
Application	A collection of screens.
Screen	A collection of related views. Usually all views in a screen map to the same business objects.
View	A collection of applets that appears on screen at the same time. A view maps to one business object.
Applet	Allows access to the data of one business component for viewing, editing and modifying fields in that business component. Consist of multiple list columns or text box controls that display data.
Business objects	Represent the fundamental business entities in the enterprise.
Business Component	Represents a logical grouping of data from one or more tables.

To determine the object definitions behind a view, click on the Help menu and choose About View.

The About View window provides the following information:

- View Title (for example: My SAP Accounts)
- View Name (for example: My Organization SAP 4x Account Sales Area View)
- View Applets (for example: SAP 4x Account Sales Area List Applet and SAP 4x Account Sales Area Entry Applet)
- Record Row ID of the current record displayed (for example: 1-OCS6)

Using Siebel Tools and with the help of the Major Object Definitions diagram, you can identify the business object and the business components associated with the view or applets.

To identify a business object

1. Start Siebel Tools.
2. Navigate to the View option and query for the desired View.

   The business object is one of the parameters in Properties.

To identify a business component

1. Start Siebel Tools.
2. Navigate to Applet View and query for the desired Applet.

   The business component is one of the parameters in Properties.

Mapping Business Objects

After you have identified the business objects on the Siebel applications and SAP sides, you need to consider the differences between these objects and how to physically map from one to the other. In particular you should consider:

- Is modification to SAP necessary to support data or functionality contained in Siebel applications?
- Is modification to Siebel applications necessary to support data or functionality provided by SAP?
- Do you have a complete understanding of the business usage of each field in each object?
- Can you create a field-by-field mapping document to determine if there are any specific business rules, format specifications, or changes in the definition of fields from one application to another that need to be changed within the Business Service Data Map?
- Did you scrutinize key fields that provide unique identifiers for each application in how they are to be mapped?
- Do specific rules need to be applied to each application to make sure the data is transferable? Should these rules be enforced procedurally or through specifically written code?
- Should one application or the other be designated as the Master for specific types of data objects? Is the creation of the object allowed in only one application with read-only access for the other?
- Does administrative information need to be created in either application to support the other?

Making Necessary Siebel Application and SAP R/3 Customizations

Depending upon the results of your analysis, you may determine that structural or configuration changes are necessary either in the Siebel application or in SAP. For example, if the Siebel business object and the chosen SAP interface do not match well enough to perform a simple map between them, you may need to make changes to the database or application prior to creating the map. This section covers some of the possible changes you may need to make.

Customization in the Siebel Application

Depending upon your analysis, some Siebel configuration may be necessary to build the desired interface. Customization may be necessary at several levels in the Siebel architecture as shown in Figure 19. In the Siebel User Interface Objects Layer you can customize screens, views, applets, controls, and navigation. In the Siebel Business Objects Layer you can customize business objects and business components. In the Siebel Data Objects Layer you can customize base tables and extension tables.

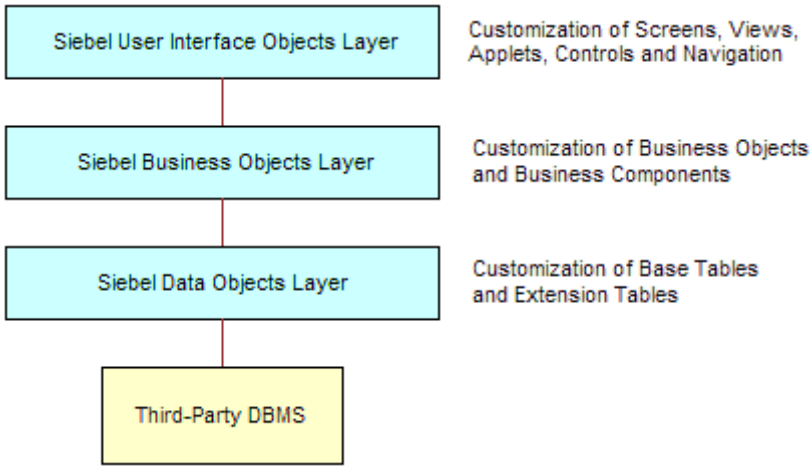


Figure 19. Siebel Architecture Diagram

The Siebel application allows customization in each of these layers. Major configuration tasks are performed through Siebel Tools. An overview of the changes you can make is provided here, but more detailed information is provided in *Using Siebel Tools*.

Although the Siebel database model is designed to accommodate most industry specific requirements, you may find that you need to make modifications. You can add both base and extension tables to the Siebel database as well as add new fields. Changes in the Siebel data model, such as the Siebel Data Objects Layer, then require modifications to the business objects and user interface objects layers.

Changes to the business objects layer may include the addition of columns to business components. These fields then need to be exposed to the user interface in the relevant code applets. Depending upon the changes, new views or screens may need to be added to support new functionality.

Because integration objects are based upon business objects and business components, changes in any existing business objects for which there are corresponding integration objects requires the update of those integration objects. Update integration objects through the synchronize function in Siebel Tools. Take care as the integration objects may have been modified after they were originally captured. You may need to reapply modifications manually after synchronizing. For comparison, you should save a copy of the original integration object prior to synchronization.

Standard integrations use LOV (List of Values) definitions for language independent values instead of hard-coded strings in eScript maps. Create and use LOVs wherever possible in scripts instead of hard-coded strings. Maintain LOVs in Application Administration > List Of Values.

Customization in SAP R/3

SAP customization may involve the addition of fields to tables, custom code in user exits, user interface changes, or the development of specific IDOCs or BAPIs. SAP basis, technical and business configuration personnel who are experienced with your own SAP implementation should perform this work if it is needed.

Building the Interfaces

When you have performed your analysis, design, and necessary changes to the Siebel application and SAP, you are ready to create the actual business data flow. This process consists of the following tasks.

Creating SAP Integration Objects

Depending upon the SAP interface you have decided to use, you need to capture either a BAPI/RFC or IDOC from SAP and create an Integration Object in the Siebel application that represents the external object.

NOTE: Integration Objects must be compiled into the working copy of the .srf file after they are created with the Integration Object Wizard.

To create IDOC integration objects

1. Start Siebel Tools.
2. Make sure that you have a project checked out into which you can add the new integration object.
3. Click New or File > New Object.

   The New Object Wizards window opens.
4. Click the EAI tab.
5. Select the Integration Object icon and click OK.

   The Integration Object Builder window opens.
6. In the top drop-down list, choose the project into which you want to add the integration object.
7. In the second drop-down list, choose the name of the wizard used to create the type of integration object you want.

   In this case, choose EAI SAP IDOC Adapter Wizard.
8. Click Next.

   The second page of the wizard appears.
9. From the drop-down list, choose an SAP Object.

   A default name for the integration object appears in the name field.
10. Edit the name to customize it for your use.
11. Click Next.

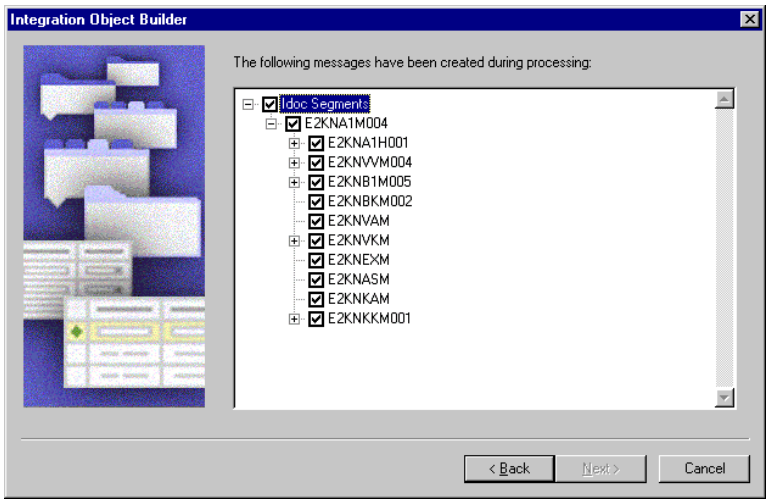
    Siebel Tools prompts you to continue.

12 Click OK.

Siebel Tools connects to SAP and retrieves the metadata for the SAP object requested.

NOTE: For this connection to take place, you must have added connection information to your `saprfc.ini` file and to your `tools.cfg` file as described in "Modifying SAP Configuration Files" on page 19 and "Modifying Siebel Configuration Files" on page 23.

The wizard displays a hierarchical display of the structure of the IDOC you have captured from SAP. The following figure shows a hierarchical list (tree list) starting with IDOC Segments and message names under it.



13 If you do not intend to use a component, clear its check box. Components are captured only if they are not disabled at this point. Capture all components and disable those that you do not use by setting the Inactive flag for the component in the integration object.

14 Click Next and click Finish to create the integration object in the Siebel repository.

15 To view the integration object, select the Integration Objects in the Navigation window in Siebel Tools.

NOTE: For more information on the structure of IDOC Integration Objects and configuration of the IDOC Integration Object Wizard, read Chapter 7, "IDOC Interfaces."

Creating BAPI/RFC Integration Objects

For a single BAPI/RFC interface, you must create two integration objects: one for the inbound call to the BAPI/RFC and one for the outbound return from the BAPI/RFC.

NOTE: For transactional RFCs, only one integration object is necessary, as this is always an inbound call only.

To capture inbound calls

1. Start Siebel Tools.
2. Make sure that you have a project checked out into which you can add the new integration object.
3. Click New or File > New Object.

   The New Object Wizards window opens.

   a. Click the EAI tab.

   b. Select the Integration Object icon and click OK.
4. From the top drop-down list, choose the project into which you want to add the integration object.
5. From the second drop-down list, choose the name of the wizard used to create the type of integration object you want.

   a. For inbound calls, choose EAI SAP BAPI Input Wizard.

   b. For outbound calls, choose EAI SAP BAPI Output Wizard.
6. Click Next.
7. From the drop-down list, choose an SAP Object.

   A default name for the integration object appears in the name field.
8. Edit the name to customize it for your use.

   The second page of the wizard appears.
9. Click Next.

   Siebel Tools prompts you to continue.
10. Click OK.

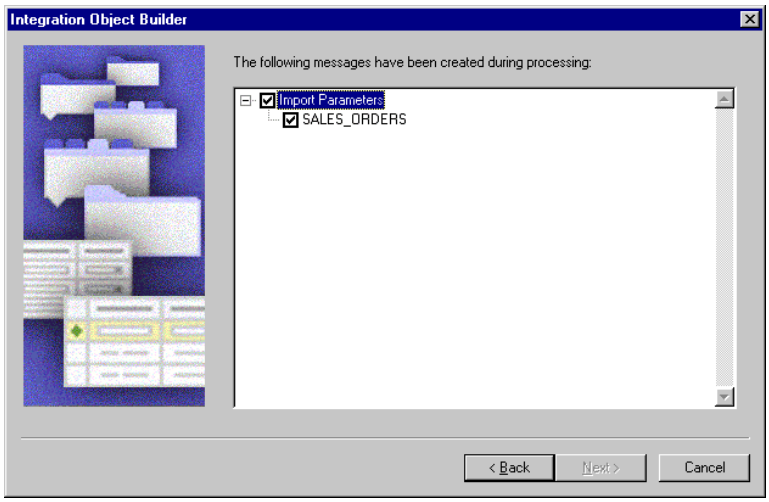
    Siebel Tools connects to SAP and retrieves the metadata for the SAP object requested.

    NOTE: For this connection to take place, you must have added connection information to your `saprfc.ini` file and to your `tools.cfg` file as described in "Modifying SAP Configuration Files" on page 19 and "Modifying Siebel Configuration Files" on page 23.

    The wizard displays a hierarchical display of the structure of the BAPI/RFC you have captured from SAP.

11 If you do not intend to use a component, clear its check box. Components are captured only if they are not disabled at this point. Capture all components and then disable those you do not currently use by setting the Inactive flag on the corresponding components.

The following figure shows a hierarchical list of components. The list shown has Import Parameters, and has SALES_ORDERS one level below.



12 Click Finish to create the integration object in the Siebel repository.

13 To view the integration object, select the Integration Objects in the Navigation window in Siebel Tools.

NOTE: For more information on the structure of BAPI/RFC Integration Objects, read Chapter 6, "BAPI Interfaces."

Modifying Integration Objects

In general you should not need to modify the BAPI integration object captured by the BAPI Integration Object Wizard. In some cases, however, you may want to modify it. This topic discusses the modifications necessary.

Using Normalization of Internal Tables

In some cases, SAP BAPI and RFC interfaces may denormalize data from several tables into a single internal table passed through the RFC interface. This can make it difficult to extract the data from the interface table into different components. For this reason the Siebel Connector for SAP R/3 provides a way to define multiple components for a single SAP internal table passed through the interface. This is currently implemented for extraction from an SAP internal table, and it applies only to BAPI Output integration objects.

The BAPI_SALESORDER_GETSTATUS BAPI is an example of this. The output integration object corresponding to BAPI_SALESORDER_GETSTATUS was modified after it was captured from SAP. This integration object is used in the Siebel Sales Order to SAP Sales Order standard integration.

Figure 20 on page 93 shows the modifications made to the integration object component and represents the STATUSINFO table returned by SAP. This SAP internal table contains information that can be separated into three normalized tables.

- Order Header information
- Order Line Item information
- Order Delivery Line Item information

Three new components can be created in the integration object to separate the Header, Line Item, and Delivery Item data. The BAPI adapter can then automatically create these new components from the original component. One component from SAP becomes three normalized components within the Siebel workflow after the invocation of the BAPI adapter.

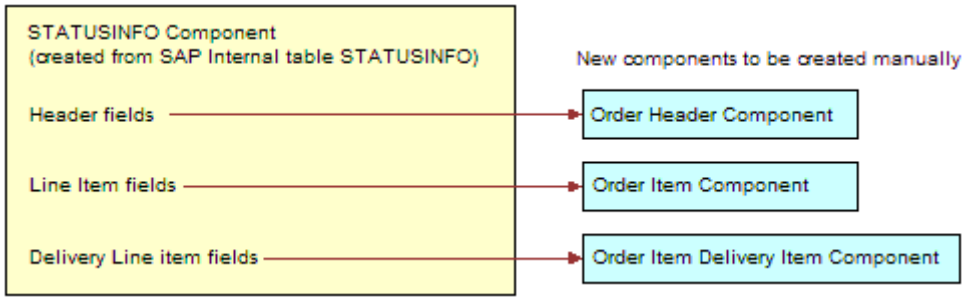


Figure 20. Normalization of Integration Object Components

To convert a single component to multiple components

1. From Siebel Tools, navigate to a BAPI integration object to modify.
2. Select the component applet under the integration object.
3. Select the component you want to divide into multiple components and make a copy of it for each new component.
4. Give each one a new name and new External Name Context.

   Make sure the External name retains the original SAP name for the table and the External Name Context is not the same as the External Name for any component.
5. Correct the Parent Integration Object field as necessary. In the case of BAPI_SALESORDER_GETSTATUS, the new components form a hierarchy:

   ```
   Export Parameters
      Order Header - parent Export Parameters
         Order Item - Parent Order Header
            Order Item Delivery Item - Parent Order Item
   ```

   The Parent Integration Object field has been changed to include the parent of each new component.
6. The External Sequence number field for components is currently not used by the BAPI Adapter. However, these fields do need to be filled in with values. If two or more components have the same parent, you can optionally give them unique External Sequence numbers.

7 Remove fields that are not needed for each component, leaving only the fields that you would like to have in each new component.

8 Create appropriate user keys for each component.

User keys provide a list of key fields used to determine uniqueness for a component. To create the user key:

a Select the component the user key needs to be created for.

b Select Integration Component Key from the Object Explorer.

c Add a new record to the Integration Component Keys list.

d Give the key a name, sequence number, and key type of "User Key".

e Select Integration Component Key Fields from the Object Explorer and add new records to the Integration Component Key Fields list, one for each field in the key.

Modifying External Scale Values

In certain cases, it may be necessary to change the External Scale value for numeric Integration Object fields. The External Scale value of a numeric field defines the number of decimals to appear in the number after the decimal point. This may be necessary in the following cases:

- SAP's BAPI interface passes a NUMC field with no decimal point, but the number is intended to be interpreted as a decimal number. For example: the field ORDER_ITEMS_IN-REQ_QTY is defined as a NUMC field but it is intended to be interpreted as a decimal number with three digits after the decimal point. A value of "0009000" then needs to be interpreted as 9.000 rather than 9000. SAP indicates this in the field's description, but no metadata is available in SAP which defines this. The BAPI Adapter has the capability to place the decimal point at the correct position given that the External Scale field is set. In this case the External Scale value would need to be set to 3 in the integration object.
- Incorrect currency values can be passed through the BAPI interface depending upon currency being used in the application. There are differences between the scale value passed to the BAPI wizard as metadata and the actual scale value for the data. If you find that the decimal point has moved after sending or receiving a currency value, you may need to adjust the External Scale value for the Integration Object field. The External Scale value of a numeric field defines the number of decimals to appear in the number after the decimal point.

Creating Siebel Integration Objects

To interface to the Siebel database, you need to create Siebel integration objects to represent the Siebel business objects you are working with. While it is possible to work directly with business objects, typically your workflow works with integration objects that represent business objects. To create a Siebel Integration object you need to execute the EAI Siebel Wizard in Siebel Tools. Modification of the Siebel integration object after it is created by the EAI Siebel Wizard is often needed. Information on the creation and modification of Siebel integration objects can be found in *Integration Platform Technologies: Siebel eBusiness Application Integration Volume II*.

Integration ID

Integration ID serves many purposes. It refers to the back office objects such as customer number, sales order number, and so on. Integration ID can also be used in the user keys. Integration ID is used for both BAPI and IDOC based data flows.

BAPI-Based Data Flows

Standard BAPI data flows are used to either create new objects in SAP or get updated information from SAP. For both these cases Integration ID is used to update Siebel objects with SAP document number such as order number and customer number created in SAP.

For some special business objects such order, Integration ID is used to enforce the business rules. After the integration ID is populated with the back office information, then further update to the order business object is not allowed. The Integration ID is also used as one of the import parameters for the BAPI when the BAPI is used to extract information from SAP.

IDOC-Based Data Flows

IDOC-based data flows are generally one-way data transfers and are used in an asynchronous mode. Here the Integration ID becomes even more important as it is used not only to hold information about SAP document number but also used in the User Key to synchronize IDOC objects.

For inbound IDOCs like Product and Account data flows (from SAP to the Siebel application), the Integration ID is used as first User key for data transactions like Insert, Update, and Delete. In this case, the Integration ID is unique, because it is used as the primary key in the connected external application.

Make sure that incoming documents, such as Customer/Material, have a unique document number when sent to the Siebel application.

Creating Business Service Data Maps

Business Service Data Maps can be created through the use of eScript or the Siebel Data Mapper. For more detailed information, read *Overview: Siebel eBusiness Application Integration Volume I*.

Working with eScript

Business Service Data Maps can be created using eScript. These maps can be created and edited within Siebel Tools. The maps must be compiled in the .srf file before they can be executed within the Siebel Client or Server applications.

To create a new business service

1. Open Siebel Tools.
2. Navigate to Tools > Object Explorer > Business Service.
3. Select Business Service and right click and choose New Record or Copy Record.

4 Enter Name, Project, Class "CSSEAIDTEScriptService", Display Name (as you want it to appear) and other Properties.

   For more information, read *Overview: Siebel eBusiness Application Integration Volume I*.

5 After you have created a business service, you can edit it by right-clicking on the business service record and choosing Edit Scripts.

For more information on eScript, read *Siebel eScript Language Reference*.

Working with Siebel Data Mapper

The Siebel Data Mapper can be used in place of an eScript business service to map fields from one integration object to another. The map is stored in the Siebel database and does not need to be compiled into the .srf.

You can visually map source integration object and target integration object structures using the map design and editor tools. These are available in the Siebel Client by navigating to Integration Administration > Data Maps, Data Map Editor, and Data Map Browser. Under the object level, there are maps at the component and field levels. Siebel Data Mapper uses standard Siebel Query Language as an expression evaluator.

Your map can be executed from workflow by executing the business service EAI Data Transformation Engine. For more information on how to create a map, refer to *Overview: Siebel eBusiness Application Integration Volume I*.

To create maps

1 Navigate to Administration - Integration > Data Maps.

2 Create a new record, using the following table as a guide.

Field	Description	Example
Name	Provide a unique name for this Integration Object map.	SAP Product map with Siebel Data Mapper
Source Object Name	Identify the source object for this map.	Product - Receive SAP 4x Material (4x IDOC Input) - DDTE
Target Object Name	Identify the target object for this map.	Product - Receive SAP 4x Material (Siebel)

3 Navigate to Data Maps Editor, and edit Integration Component Maps using the following table as a guide.

Field	Description
Name	Provide a name for this Integration Component map.
Parent Component Map Name	

Field	Description
Source Component Name	Choose an appropriate component name from the drop-down list.
Target Component Name	Choose an appropriate component name from the drop-down list.
Source Search Specification	Use to provide an optional component map condition. For information, consult *Overview: Siebel eBusiness Application Integration Volume I*.
Precondition	Use to provide optional error checking. For information, consult *Overview: Siebel eBusiness Application Integration Volume I*.
Postcondition	Use to provide optional error checking. For information, consult *Overview: Siebel eBusiness Application Integration Volume I*.

4 Edit Integration Field Maps using the following table as a guide.

Field	Description
Source Expression	Use Siebel Standard Query Language to compose an expression, provide the Source Field Name in the format of [Field Name], or enter a string that you want to assign to the Target Field.
Target Field Name	Choose an appropriate name from the drop-down list.

5 Create a workflow that calls this map.

The new business service is called EAI Data Transformation Engine, the method invoked is Execute, and there are three arguments: Map Name, Output Integration Object Name, and SiebelMessage. The Map Name is your new map name.

6 Create new maps between any two integration objects, and then build your new integration point based on your business requirement.

For Siebel Data Mapper limitations in this release, read *Overview: Siebel eBusiness Application Integration Volume I*.

Creating Workflows

After you have created the Business Service Data Map, you can create a workflow to execute it. For information about creating, deploying, and activating workflows, see *Siebel Business Process Designer Administration Guide*.

The workflow that you create will need to include and execute specific business services. The following topics describe the types of Business Services you are likely to use in building an SAP interface workflow.

Adding the Siebel Adapter

Many workflows between Siebel applications and SAP start or end with the Siebel Adapter. The Siebel Adapter interfaces to the Siebel database through a business object. You need to add a Siebel Adapter business service box to your workflow diagram each time you want to interface to Siebel applications.

To add a Siebel Adapter business service

1. Add a business service step to your workflow.

   For specific instructions about how to modify workflows, see *Siebel Business Process Designer Administration Guide*.

2. Use connector arrows to link the box into your flow.
3. Set the following information for your business service step:
   - Enter the name of the business service workflow step.
   - Select EAI Siebel Adapter as the business service for this step.
   - Select a method for the EAI Siebel Adapter.
4. Enter input and output arguments.
   a. If you are querying the Siebel database for an object, you output a Siebel integration object instance in the form of a SiebelMessage.
   b. If you are updating the Siebel database, you input a Siebel integration object instance in the form of a SiebelMessage.

For more information on using the Siebel Adapter, read *Transports and Interfaces: Siebel eBusiness Application Integration Volume III*. Refer to workflows provided for the SAP standard integrations for examples.

Adding the SAP Interfaces

You need to use different Siebel business services depending upon the type of interface you are using with SAP. The following topics discuss the business services that you need in your workflow.

Siebel-to-SAP Interfaces

If you are creating a workflow which must make a synchronous BAPI/RFC call you need to add the BAPI Adapter business service to your workflow.

To make synchronous BAPI calls

1. Add a business service step to your workflow.

   For specific instructions about how to modify workflows, see *Siebel Business Process Designer Administration Guide*.

2 Use connector arrows to link the box into your flow.

3 Set the following information for your business service step:

- Enter the name of the business service workflow step.
- Select EAI SAP BAPI Adapter as the business service for this step.
- Select the Execute method.

4 Enter appropriate input and output arguments.

At a minimum you must pass an EAI SiebelMessage into the BAPI adapter which contains a BAPI input integration object you have created with the Integration Object wizard and the name of the output integration object to be created. You then also need a SiebelMessage output argument.

For more information on using the BAPI Adapter read Chapter 6, "BAPI Interfaces." Refer to workflows provided for the SAP standard integrations for examples.

If you are creating a workflow which must make an transactional RFC call, you need to add the tRFC BAPI Adapter business service to your workflow.

To make Transactional RFC calls

1 Add a business service step to your workflow.

For specific instructions about how to modify workflows, see *Siebel Business Process Designer Administration Guide*.

2 Use connector arrows to link the box into your flow.

3 Set the following information for your business service step:

- Enter the name of the business service workflow step.
- Select EAI SAP BAPI Adapter (tRFC) as the business service for this step.
- Select the Execute method.

4 Enter input and output arguments.

At a minimum, you must pass an EAI SiebelMessage into the BAPI tRFC adapter that contains a BAPI input integration object you have created with the Integration Object wizard. No output integration object instance is created with a transactional RFC call.

The tRFC BAPI Adapter can be configured to use the EAI Queue to temporarily store data within the BAPI call for recovery in case of error during transit to SAP. For more information on using the tRFC BAPI Adapter read Chapter 6, "BAPI Interfaces."

If you are creating a workflow which must send an IDOC to SAP, you need to add the IDOC Adapter Business Service to your workflow.

To send IDOCs to SAP

1 Add a business service step to your workflow.

For specific instructions about how to modify workflows, see *Siebel Business Process Designer Administration Guide*.

2 Use connector arrows to link the box into your flow.

3 Double-click on the Business Service box to open the Business Service Arguments window.

4 Set the following information for your business service step:

- Enter the name of the business service workflow step.
- Select EAI SAP IDOC Adapter as the business service for this step.
- Select the Send method.

5 Enter input and output arguments.

At a minimum, you must pass an EAI SiebelMessage into the IDOC Adapter which contains an IDOC Integration Object you have created with the Integration Object wizard. Set the SAPSenderPtnrNum input argument as follows:

- If executing the map in the server, set the SAPSenderPrtnrNum component parameter for the Business Integration Manager.
- Within your map, set the sender partner field (SNDPRN) in the IDOC control record.
- Set the literal value of the SAPSenderPrtnrNum input argument in the workflow.

The IDOC Adapter converts the IDOC to an RFC call and then invokes the tRFC BAPI Adapter. For this reason, this call can be configured to use the EAI Queue to temporarily store data within the IDOC for recovery in case of error during transit to SAP. For more information on using the tRFC BAPI Adapter with EAI Queue read Chapter 6, "BAPI Interfaces." For more information on using the IDOC Adapter read Chapter 7, "IDOC Interfaces." The SAP standard integrations provide examples of sending an IDOC to SAP.

SAP-to-Siebel Interfaces

The tRFC BAPI Receiver processes data that SAP sends. This component invokes a workflow to process the data. The tRFC BAPI Receiver may receive an IDOC or a tRFC call from SAP. In each case, the first step in a workflow invoked by the receiver is a Business Service Data Map that accepts an EAI Property Set in the form of a SiebelMessage containing an IDOC or BAPI Input Integration Object instance.

Figure 21 illustrates IDOC Processing and tRFC Call Processing. In IDOC Processing, data from SAP R/3 goes to the Siebel tRFC BAPI Receiver, which passes it to the IDOC Workflow Processor, which passes it a specific workflow (data dependent). This workflow consists of a Business Service Data Map and the Siebel Adapter. The Siebel Adapter then writes the data to the Siebel database. In tRFC Call Processing, data from SAP R/3 goes to the Siebel tRFC BAPI Receiver, which passes it a specific workflow (data dependent). This workflow consists of a Business Service Data Map and the Siebel Adapter. The Siebel Adapter then writes the data to the Siebel database.

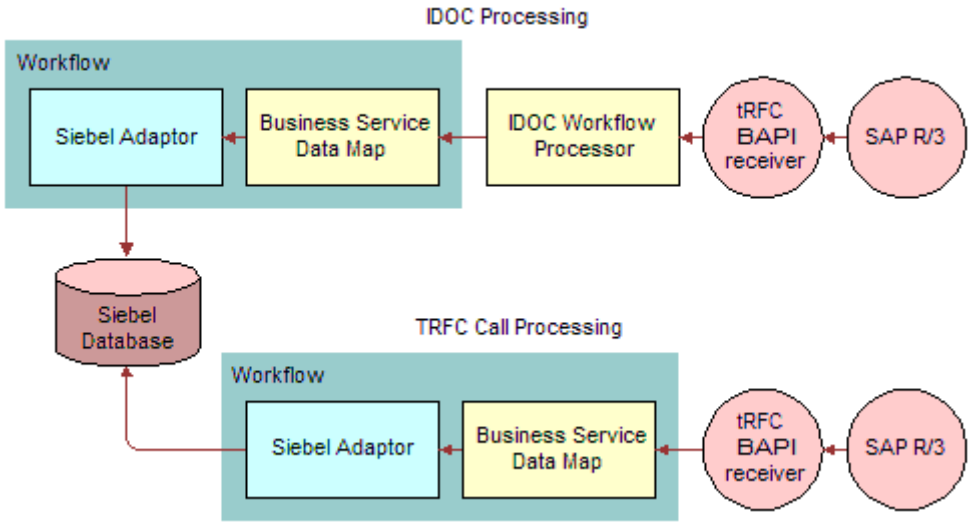


Figure 21. IDOC and tRFC Processing

In each case no specific adapter needs to be added to the workflow to interface to SAP as this interface occurs outside of the workflow.

When adding your business service data map to a Siebel workflow, you must create a process property in your workflow with the name SiebelMessage. This process property must then be an input to the initial business service data map. The services external to the workflow handles passing this SiebelMessage to your workflow and the process property of this name.

NOTE: The SiebelMessage Process Property must be named "SiebelMessage", exactly as shown here. Do not use any spaces in the name. If the name is not correct, an error message results indicating that there are no children of type SiebelMessage in the property set passed to your workflow.

Receiving IDOCs from SAP

When receiving IDOCs from SAP there are two important configuration items to define:

- The name of the workflow to be called upon receipt of the IDOC must be defined. This is usually done in the workflow user property of the IDOC integration object to be passed to workflow.
- The names of IDOC integration objects that may be accepted by the IDOC workflow processor must be defined in the SAPIdocAllowedObjectnn user properties of the IDOC workflow processor business service.

For more information on configuration of user properties, read Chapter 7, "IDOC Interfaces."

Receiving tRFC Calls from SAP

When receiving a tRFC call from SAP there are two important configuration items to define:

- The name of the Workflow to be called upon receipt of the tRFC call must be defined. This is usually done in the Workflow user property of the BAPI Input Integration Object to be passed to workflow.
- The names of BAPI Input Integration Objects that may be accepted by the tRFC BAPI receiver must be defined in the RFCFunctionIntObjnn user properties of the tRFC BAPI receiver business service.

For more information on configuration of user properties, read Chapter 6, "BAPI Interfaces."

Adding the Business Service Data Map

If you have created a business service data map, you need to add this to your workflow at the appropriate point in the flow.

To add a business service data map to a workflow

1. Add a business service step to your workflow.

   For specific instructions about how to modify workflows, see *Siebel Business Process Designer Administration Guide*.

2. Use connector arrows to link the box into your flow.
3. Set the following information for your business service step:
   - Enter the name of the business service workflow step.
   - Select the name of your business service for this step. If you are using a Siebel Data Mapper map, choose .EAI Data Transformation Engine.
   - Select the Execute method.
4. Enter input and output arguments.

   Typically the input arguments include:

Argument	Description
SiebelMessage	Contains the integration object instance.
MapName	Name of the function in eScript to be called if you use eScript map, or Siebel Data Mapper map name if you use Siebel Data Mapper map.
OutputIntegrationObjectName	Name of the integration object which is output.

The output arguments generally include an output SiebelMessage that contains a new integration object instance created during the map.

You can refer to workflows provided for the SAP standard integrations for examples. For more information, read *Siebel Business Process Designer Administration Guide* and *Overview: Siebel eBusiness Application Integration Volume I*.

Testing the Interface

The following tools aid in testing your workflow and business services:

- Business Service Simulator
- Workflow Simulator
- EAIRaiseError() function
- File Output
- Siebel Tools Debugger

Using the Business Service Simulator

The Business Service Simulator can be used to test specific business services. For SAP scenarios, the Business Service Simulator would normally be used to test your own business services and to test workflows that are invoked by the tRFC BAPI Receiver (this may be IDOC or BAPI as both are started with the tRFC BAPI Receiver).

The Business Service Simulator is executed within the Siebel client by navigating to Administration - Business Service > Simulator. To test your own business services or to find more information on the Business Service Simulator, consult *Overview: Siebel eBusiness Application Integration Volume I*. The following procedure explains how to execute a workflow that is triggered from the tRFC BAPI receiver upon the receipt of an IDOC or BAPI call from SAP.

To execute triggered workflows

1. Create a new record in the Service Methods list.
2. Enter the Service Name `EAI SAP BAPI Receiver (tRFC)`.
3. Enter the Method Name "ReceiveDispatch".
4. Select New in the Input Property Set view to create a new record.

   Then select the Property Name field and create a new property name value pair with the name SAPRfcDestEntry and its appropriate value.

   NOTE: If you are using the Siebel Mobile Client you may also set SAPRfcDestEntry in the client configuration file as described in "Modifying Siebel Configuration Files" on page 23.

5 In the Siebel application, click Run to start the receiver service.

   The service makes a connection to SAP and waits for input from SAP up to the number of seconds defined in the SAPSleepTime User Property on the `EAI SAP BAPI Receiver (tRFC)` service definition.

6 In SAP, trigger the send of an IDOC or BAPI call to the Receiver.

   You must do this within the time period SAPSleepTime (default value for this is 60 seconds).

7 Verify results of the workflow. This depends upon the action your workflow takes.

NOTE: When the tRFC BAPI receiver is executed from the Business Service Simulator it tests for input from SAP only for SAPSleepTime seconds. When the receiver is executed in the Siebel Server as the BAPIRcvr component, it continuously recalls itself, and looks each time for input from SAP.

Using the Workflow Simulator

The Workflow Simulator can be used to test workflow execution step-by-step. For information on how to execute the Workflow Simulator, read *Siebel Business Process Designer Administration Guide*. Here are some tips for testing SAP interfaces using the workflow simulator:

- If your workflow begins with the Siebel Adapter, you can set the Object ID field of the Siebel Message with the value of a ROW_ID that corresponds to an object you would like to extract from the Siebel application to begin the test. The ROW_ID of an object can be retrieved from the About Row Help when the row is selected.
- The XML Read from File and XML Write to File services can be added as intermediate steps between business services in your workflow to write the current Integration Object Instance to an XML file. This can be useful for creating test data and for verifying the content of the object.
- If you need to execute a workflow that is normally triggered by the tRFC BAPI receiver upon receipt of an IDOC or BAPI call in the workflow simulator, you can:
  - Add the EAI XML Write to File service as your first step in the workflow to write the input from SAP to an XML file.
  - Execute the tRFC BAPI receiver service with the Business Service Simulator as described previously. The first step of the workflow writes the input data to a file.
  - Modify the workflow to start by reading the data from the file. Replace the EAI XML Write to File service with the EAI XML Read from File service.
  - Execute the workflow in the Workflow Simulator.

- If you are making a BAPI or RFC call from your workflow into SAP and you would like to follow your data from the Siebel application to the ABAP debugger, set ABAP_DEBUG=1 in your TYPE=A Destination definition in the `saprfc.ini` file. When the BAPI adapter service is executed in your workflow, the SAP Client appears and displays the ABAP debugger at the start of the BAPI or RFC function call. The SAP Client must be installed on the same machine on which you are executing the Siebel Mobile Web Client.

NOTE: If your workflow contains a call to the BAPI adapter business service, you must remember that you are invoking this business service from the Siebel Client. This requires that connection parameters used for this service, which are normally set as component parameters on the Business Integration Manager will need to be set in the Siebel Client configuration file, if working with the Siebel Mobile Web Client, or as input arguments to the BAPI adapter business service in your workflow.

The EAIRaiseError() Function

You can also use the EAIRaiseError() function as a way to debug. This function terminates the execution of the script (which is what you might want at times). For example, to see if there is a value in the City field (and you do not want to go any further if there is none), you can use code similar to the following:

```
if  (!iAddrComp.GetFieldValue("City") )
EAIRaiseError ("Value in the City field is empty");
```

For more information on EAIRaiseError() function, refer to data mapping using scripts in *Business Processes and Rules: Siebel eBusiness Application Integration Volume IV*.

File Output

You can use the system created log files to further assist you in your debugging. You can also write some identifying strings or variable values to a file yourself using the file manipulation capabilities in eScript. For example:

```
var fp = Clib.fopen("SAPIntAcc.log", "w");
Clib.fputs("Integration Id: " + IntId,fp);
//Add more fputs() lines according to your needs.
Clib.fclose(fp);
```

For more information on system log files, read *Configuring Siebel eBusiness Applications*. For more information on the Clib object, read *Siebel Tools Online Help*.

The Siebel Tools Debugger

Siebel Tools provides capabilities to debug the eScript code. First, identify which script you want to debug. For example, to debug exiting workflows, you can find out which map function executes when the workflow process runs. This is available from the process properties of the Workflow. Put breakpoints in that script and run the Siebel client from Siebel Tools. The execution will break at one of those breakpoints when you click the appropriate button in Siebel client. From there on, you can step through the code and look at the values of variables.

For more information about running the Siebel client from Siebel Tools, read *Using Siebel Tools*.

6 BAPI Interfaces

This chapter provides reference information for BAPI and RFC interfaces. It contains information on BAPI Integration Objects as well as information on configuration of the BAPI Integration Object Wizard, BAPI Adapter, tRFC BAPI Adapter, and tRFC BAPI Receiver.

Create SAP Integration Objects with the BAPI Wizard

This section describes BAPI Integration Objects in detail and describes configuration options for the BAPI Integration Object Wizard.

BAPI Integration Objects

BAPI Integration objects represent the interface to a BAPI or RFC function call in SAP. There are two types of objects: BAPI Input Integration Objects and BAPI Output Integration Objects. The Input Object represents the initial call to the function and includes import parameters and tables. The Output Object represents the return information from the function call and includes export parameters and tables. Figure 22 shows the structure of an input integration object.

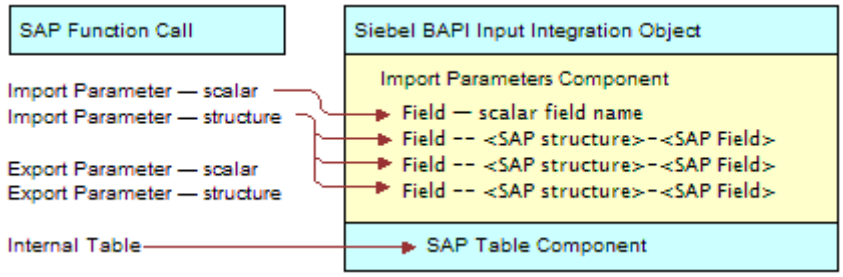


Figure 22. Input Integration Object Structure

For example, Figure 23 shows BAPI BAPI_SALESORDER_CREATEFROMDAT1 having the following structure and corresponding Input Integration Object.

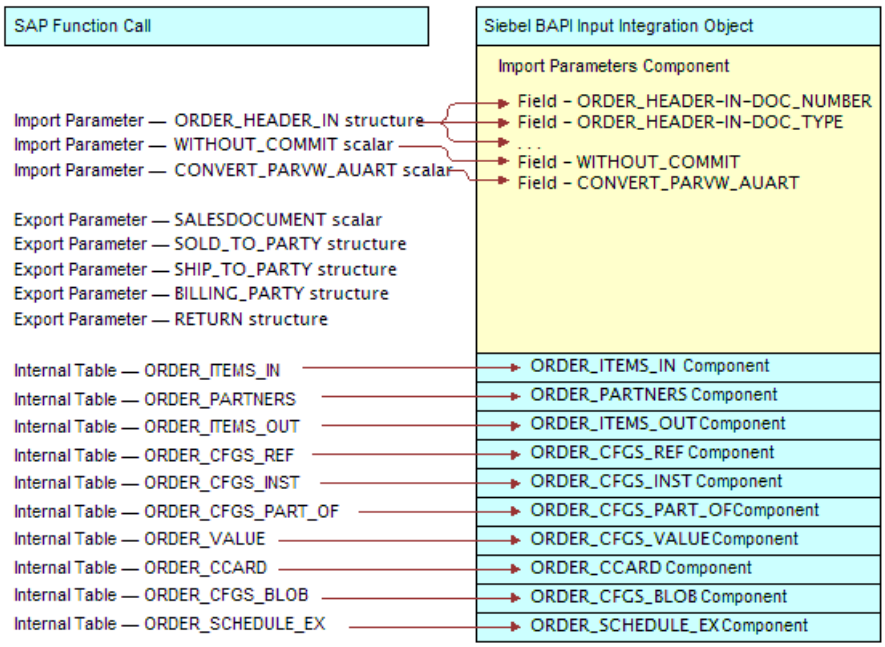


Figure 23. Input Integration Object Structure Example

Figure 24 shows the structure of an output integration object.

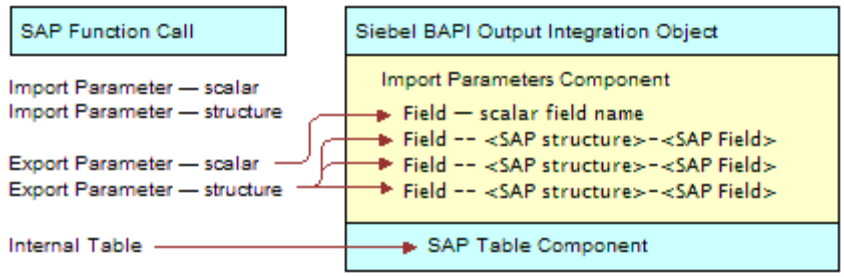


Figure 24. Output Integration Object Structure

For example, Figure 25 shows BAPI BAPI_SALESORDER_CREATEFROMDAT1 having the following structure and corresponding Output Integration Object.

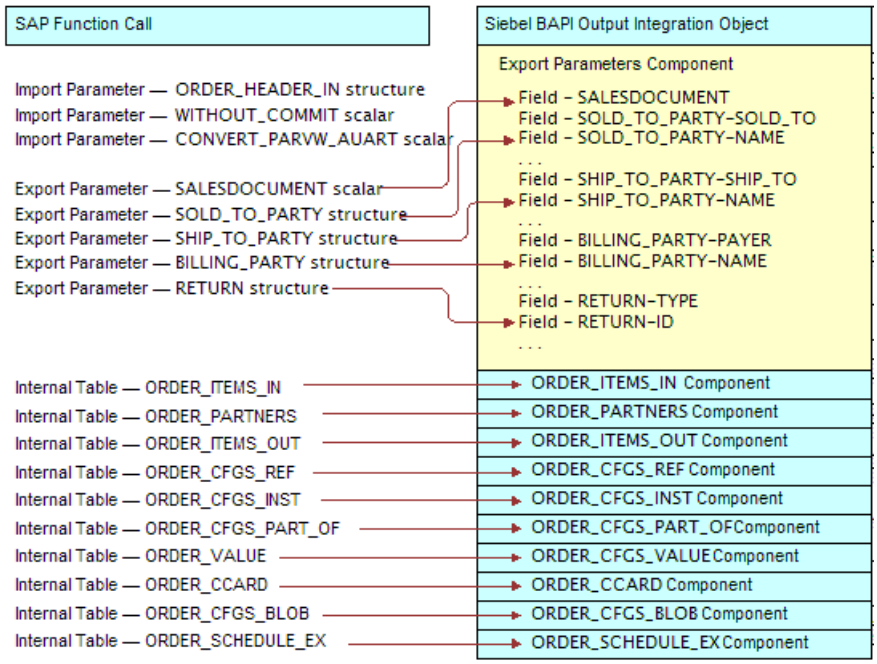


Figure 25. Output Integration Object Structure Example

Each component field contains both the Siebel application and SAP data types. These data types are described in detail in Appendix A, "Data Types."

In addition, BAPI Input Integration objects contain a user property called Workflow. This user property is only used for the tRFC BAPI Receiver. For information on this user property, read "tRFC BAPI Receiver Configuration" on page 120.

Creating and Viewing the Integration Object

BAPI Input and Output Integration Objects are created with the EAI SAP BAPI Input Object Wizard and EAI SAP BAPI Output Object Wizard respectively. "Creating BAPI/RFC Integration Objects" on page 90 provides instructions for executing these wizards in Siebel Tools.

You can view integration objects in Siebel Tools by selecting Integration Objects from the Navigation window. Sample integration objects have been created as part of the Standard Integrations. For information on these integration object names, read "Understanding the Standard Integration Interfaces" on page 72.

BAPI Integration

The Siebel application supports the following interface options through BAPI and RFC:

- Making BAPI and RFC Calls to SAP
  - Synchronous BAPI calls using the BAPI Adapter
  - Transactional BAPI calls using the tRFC BAPI Adapter
    - EAI Queuing options for data recoverability
- Receiving BAPI and RFC Calls from SAP
  - Transactional RFC calls using the tRFC BAPI Receiver
  - EAI Queuing options for data recoverability

This functionality is performed through a series of business services provided by Siebel applications. Other sections of this document refer to these business services by the names in Table 20.

Table 20. Business Service Names

Reference Name	Business Service Name
tRFC BAPI Adapter	`EAI SAP BAPI Adapter (tRFC)`
tRFC BAPI Receiver	`EAI SAP BAPI Receiver (tRFC)`
BAPI Adapter	`EAI SAP BAPI Adapter`
BAPI Workflow Service	`EAI SAP BAPI Workflow Service`
Send Transaction Service	`EAI SAP Send Transaction Service`
Process Transaction Service	`EAI SAP Process Transaction Service`

Configuration requirements for each of these Business Services are discussed in the following topics. The behavior of Business Services can be controlled using:

- User Properties
- Component Parameters
- Method Arguments

Set the values of User Properties in Siebel Tools. Select Business Service > Business Service User Prop from the Object Explorer and edit the Value field for the specific User Property. Compile changes to User Property values into the .srf file.

The values of Component Parameters may be set by default or by you. You can change parameter values through the Server Manager. For more information on setting Component Parameters, read *Siebel System Administration Guide*. If a Component Parameter has the same name as a User Property, its value when it is set overrides any previously defined value of the User Property.

The values of Method Arguments are set when a method in a Business Service is invoked. Values for Method Arguments on services that are invoked from a workflow are defined within workflow. For this reason, they can have fixed or variable values. If a Method Argument and a Business Service User Property have the same name, the value of the Method Argument overrides the value of the User Property. If a Method Argument and a Component Parameter have the same name, the value of the Method Argument overrides the value of the Component Parameter.

To summarize, the value of User Properties can be overridden with Component Parameter or Method Argument values and the value of a Component Parameter can be overridden with a Method Argument value.

Make Synchronous BAPI Calls to SAP

A synchronous BAPI or RFC call to SAP has both an inbound and outbound interface to SAP. The `EAI SAP BAPI adapter` service is used for this type of communication with SAP. This section provides information on the configuration and use of the BAPI adapter.

BAPI Adapter Configuration

The `EAI SAP BAPI adapter` business service can be invoked from a workflow. Table 21 and Table 22 summarize the User Properties, Method Arguments, and Component Parameters that may be set to control its behavior. The `EAI SAP BAPI Adapter` is a cached business service. Variables that are defined to be User Properties in the Business Service are generally used only once for initialization purposes at the time of the first call to the Business Service. Variables that are defined to be Method Arguments are used on each and every call and may vary from call to call. Those variables that may be set as component parameters must be set for the Business Integration Manager component.

Table 21 contains User Properties, Component Parameters, and Input Method Arguments for the Execute Method.

Table 21. BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
OutputIntObjectFormat			X		
OutputIntObjectName			X	Must be a BAPI Output Integration Object name	Integration object name for the BAPI Output integration object containing export parameters and tables returned by SAP from the call.

Table 21. BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPRfcConnectString		X	X		SAP connection string. This string defines the destination (used in `saprfc.ini`), client, and language for the SAP connection.
SAPRfcUserName		X	X		SAP user name
SAPRfcPassword		X	X		SAP password
SAPRfcTrace		X	X	"true" or "false"	SAP trace file usage on or off. Same as TRACE=0 or 1 in `saprfc.ini` file.
SiebelMessage			X		EAI Siebel Message containing BAPI Input integration object instance.
DisconnectAlways	X			"true" or "false"	Determines connection behavior of BAPI adapter. When "true", the adapter makes a new connection each time the Execute method is called. If "false", the adapter keeps its connection open. (See also SAPRfcMaxConnectTime.)

Table 21. BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPRfcMaxConnectTime	X			Integer value in seconds	Used only if Disconnect Always is "false". Number of seconds a connection stays open measured from the time of the last BAPI call. For example, if SAPRfcMaxConnectTime is set to 600 seconds (10 minutes), then all BAPI calls made within the same workflow and within 10 minutes of one another are guaranteed to be made within the same connection. When a BAPI call is made more than 10 minutes from the time of the last BAPI call, a new connection is made with SAP. This allows BAPI calls to be made together that must occur within the same connection yet also reduces the possibility of stale or invalid connections.
SAPAutoError	X		X	"True" or "False"	The default is "True". When the value of this flag is True, and the value of the SAPErrorTypeField is E or A, the BAPI adapter raises an error, causing workflow to terminate at the BAPI adapter. When the value of this flag is False, the BAPI adapter logs error information but does not raise an error and terminate the workflow. The remaining steps in the workflow are responsible for handling the error.

Table 21. BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPCodepage	X	X	X	Value from Transcode Encoding drop-down list	Siebel code page name for SAP code page.
SAPIgnoreCharSetConv Errors	X	X	X	True or False	Default is False. If True, code page conversion errors are handled by substitution of the question mark (?) character in place of an unrecognized character.

SAPRfcConnectString uses SAP's internal connect string format. This must include the following elements:

- DEST=*Destination name from saprfc.ini file*
- CLIENT=*3-digit SAP client number*
- LANG=*single character language indicator*

These elements must be separated in the string by one or more spaces, for example, "DEST=DEV_Outbound CLIENT=555 LANG=E". Table 22 contains Output Method Arguments for the Execute Method.

Table 22. Execute Method Output Arguments

Name	Usage
RfcExceptionName	Contains the RFC exception string returned by the SAP function. If connection to SAP fails, this string contains the text: "__SIEBEL__SAP_CONNECT_FAILED__"
SiebelMessage	EAI Siebel Message containing BAPI Output integration object instance.

The Execute method is likely to be all that you need to use with the BAPI Adapter. There are two more utility methods defined. The first is MakeConnection. You can use this to test your connection information to SAP. For information on how to test your connection, read "Checking Siebel Client Connectivity" on page 39. The second is EndConnection. Use this method to force a disconnect from SAP.

NOTE: To control the disconnection with SAP by using the EndConnection method, you must use this method within the same workflow where a previous call to the adapter opened a connection. You cannot make a connection within one workflow and disconnect within a different workflow.

EndConnection uses no method arguments. It disconnects the BAPI adapter from SAP. MakeConnection uses the SAP connection variables described in Table 21 on page 111: SAPRfcConnectString, SAPRfcUserName, SAPRfcPassword, and SAPRfcTrace.

SAP BAPIs often use a standard return structure to send error messages back to the calling application. The BAPI Adapter has the capability of handling this error information in a flexible manner. The BAPI Adapter's flexibility is based on the values of two user properties or method arguments: `SAPAutoError` and `SAPErrorTypeField`. These parameters can be used together to direct the BAPI adapter to automatically raise errors or let these errors pass on to other portions of workflow for handling. The `SAPAutoError` flag turns on or off the automatic error generation. The `SAPErrorTypeField` specifies the field in the BAPI interface that contains the SAP error type (E, A, I, S, or W). Table 23 summarizes the functionality.

Table 23. BAPI Adapter Error Handling Summary

SAPAuto Error	SAPErrorTypeField	Behavior
True	Siebel name of a scalar export parameter in the BAPI interface.	BAPI Adapter raises an error if the SAPErrorTypeField contains the values E or A. Workflow stops at the BAPI adapter step.
True	Siebel name of a field in a structure export parameter in the BAPI interface.	BAPI Adapter raises an error if the SAPErrorTypeField contains the values of E or A. The values of non-blank fields in the structure containing the SAPErrorTypeField are written within the error message. Workflow stops at the BAPI adapter step.
True	Siebel name of a field in an internal table in the BAPI interface.	BAPI Adapter raises an error if the SAPErrorTypeField contains the values of E or A in any record of the internal table. The values of non-blank fields in the table containing the SAPErrorTypeField are written within the error message. One error message is created for each record in the table. Workflow stops at the BAPI adapter step.
False	Siebel name of a scalar export parameter in the BAPI interface.	BAPI Adapter does not raise an error, but the error information appears in the Siebel log. Workflow processing continues, and error handling must be implemented in the steps following the BAPI adapter.
False	Siebel name of a field in a structure export parameter in the BAPI interface.	BAPI Adapter does not raise an error, but the error information appears in the Siebel log and contains the values of non-blank fields in the export structure. Workflow processing continues, error handling must be implemented in the steps following the BAPI adapter.
False	Siebel name of a field in an internal table in the BAPI interface.	BAPI Adapter does not raise an error, but the error information appears in the Siebel log and contains the values of non-blank fields in the table. Workflow processing continues, and error handling must be implemented in the steps following the BAPI adapter.

These parameters can be set in the workflow as input method arguments to the BAPI adapter business service call. The `SAPAutoError` parameter defaults to True when not set and the `SAPErrorTypeField` parameter defaults to RETURN-TYPE when not set.

Make Transactional RFC Calls to SAP

A transactional RFC call to SAP uses only an inbound interface to SAP. The `EAI SAP BAPI Adapter (tRFC)` service is used for this type of communication with SAP. For instructions on adding the `EAI SAP BAPI Adapter (tRFC)` service to a workflow, read "Creating Workflows" on page 97. This topic provides additional information on the configuration and usage of the tRFC BAPI adapter. The tRFC BAPI adapter can be used with the EAI Queue. For more information read Chapter 8, "EAI Queue."

tRFC BAPI Adapter Configuration

The `EAI SAP BAPI adapter (tRFC)` business service can be invoked from a workflow. Read Chapter 5, "Customizing Integrations" for more information. Table 24 and Table 25 summarize the User Properties, Method Arguments, and Component Parameters that may be set to control its behavior. The `EAI SAP BAPI adapter (tRFC)` is a cached business service. Variables defined as user properties in the business service are used only once for initialization purposes at the time of the first call to the business service. Variables defined to be method arguments are used on each call and may vary from call to call. Those variables that may be set as component parameters must be set for the Business Integration Manager component.

Table 24 contains User Properties, Component Parameters, and Input Method Arguments for the Execute Method.

Table 24. tRFC BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPRfcConnectString		X	X		Refer to Table 21
SAPRfcUserName		X	X		Refer to Table 21
SAPRfcPassword		X	X		Refer to Table 21
SAPRfcTrace		X	X	"true" or "false"	Refer to Table 21
SiebelMessage			X		Refer to Table 21
DisconnectAlways	X			"true" or "false"	Refer to Table 21
SAPRfcMaxConnectTime	X			Integer value in seconds	Refer to Table 21
tRFCMode	X			"tRFC"	If value is "tRFC", indicates that this adapter is working with tRFC.

Table 24. tRFC BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPWriteXML	X	X		"WriteNone", "WriteAlways", "WriteBeforeErr", "WriteOnErr"	Indicates usage of EAI Queue. For more information, read Chapter 8, "EAI Queue." If value is "WriteNone", the queue is not being used.
SAPXMLQueueCleanup	X			"true" or "false"	If true, entries are deleted from the EAI Queue upon completion.
SAPXMLQueueService	X			XML queuing service name	Provides the name of the Business Service that handles the interface to the EAI Queue.
SAPXMLQueueName	X			EAI Queue name	Provides the name of the EAI queue being used to store queue entries created by this service.
SAPCodepage	X	X	X	Value from Transcode Encoding drop-down list	Siebel code page name for SAP code page.
SAPIgnoreCharSetConvErrors	X	X	X	True or False	Default is False. If True, code page conversion errors are handled by substitution of the question mark (?) character in place of an unrecognized character.

Table 25 contains Output Method Arguments for the Execute Method.

Table 25. Execute Method Output Arguments

Name	Usage
RfcExceptionName	Contains the RFC exception string returned by the SAP function. If connection to SAP fails, this string contains the text: "__SIEBEL__SAP_CONNECT_FAILED__"
SiebelMessage	EAI Siebel Message containing output properties

As with the BAPI adapter, you may also use the methods MakeConnection and EndConnection. For more information, read "BAPI Adapter Configuration" on page 111.

Receive tRFC Calls from SAP

A transactional RFC call from SAP uses only an inbound interface to the Siebel application. The `EAI SAP BAPI Receiver (tRFC)` service is used for this type of communication with SAP. To use the `EAI SAP BAPI Receiver (tRFC)` service with a workflow read "Creating Workflows" on page 97. This section provides additional information on the configuration and usage of the tRFC BAPI Receiver. The tRFC BAPI Receiver can be used with the EAI Queue for data recoverability. For more information, read Chapter 8, "EAI Queue."

tRFC BAPI Receiver Usage

The tRFC BAPI Receiver is used for two purposes:

- Receiving and dispatching transactional RFC calls from SAP
- Receiving and dispatching IDOCs from SAP

Receiving tRFC Calls

When the tRFC Receiver receives an RFC call from SAP, it dispatches the information it receives in the form of a BAPI Input Integration Object to a workflow. The workflow can be defined in two ways:

1. The Workflow User Property on the BAPI Input Integration Object definition in Siebel Tools can be set to the workflow name. When this type of BAPI Integration Object is received, it is sent to this workflow. The workflow name must be set using Siebel Tools and the changed integration object must be compiled into the Siebel .srf file.
2. The workflow name may be set in the ProcessName user property of the `EAI SAP BAPI Workflow Service` business service. If the workflow name is set here, BAPIs received from SAP are sent to this workflow. The workflow name in the integration objects is ignored.

The list of BAPI Input Integration Objects that the receiver can create must be defined in the User Properties: RfcFunctionIntObjnn where nn represents numbers from 01 to 50. For example if the BAPI receiver needs to be configured to receive two function calls from SAP, you would need to add two User Properties, RfcFunctionIntObj01 and RfcFunctionIntObj02 to the `EAI SAP BAPI Receiver (tRFC)` business service. The values of these user properties would be the BAPI Input Integration Object names for the integration objects representing the RFC calls to the Siebel application.

To implement the tRFC call from SAP to a Siebel workflow

1. Define the function call interface in SAP under transaction SE37.

   The function may or may not have ABAP source code, because the function implementation is within a Siebel workflow, not in SAP.
2. Activate the function.

3 Using the BAPI Input Integration Object Wizard in Siebel Tools, capture the function interface as a BAPI Input Integration Object.

4 Add the name of the workflow you wish to invoke when the tRFC call is made to the Siebel application to:

- The workflow user property of the new integration object

  or

- The ProcessName user property of the `EAI SAP BAPI workflow service` if all tRFC calls are routed to the same workflow.

5 Add the integration object name to the Business Service definition for the `EAI SAP BAPI Receiver (tRFC)` by adding a user property of the form, RfcFunctionIntObjnn, as previously mentioned.

6 Compile these changes into your .srf file.

7 Using the Siebel client, create the workflow you named in Step 4.

8 Create the ABAP code in SAP to invoke the function in the Siebel application.

Call the function IN BACKGROUND TASK with your external logical system name as the DESTINATION. Follow the call with a COMMIT WORK statement.

```
CALL FUNCTION 'ZRFC_TEST_TRFC'
   IN BACKGROUND TASK
   DESTINATION Logical System name
   .  .  .
   COMMIT WORK.
```

NOTE: SAP allows you to make multiple calls prior to the COMMIT WORK statement. This causes SAP to place multiple tRFC calls in the same transaction. The tRFC BAPI Receiver does not currently support this. If you need to make multiple calls prior to the COMMIT WORK, you must use the modifier AS SEPARATE UNIT to cause each BAPI call to be made within a separate transaction.

9 Start the Siebel Server and the BAPIRcvr component using your new .srf file.

Make sure that you followed all setup procedures described in Chapter 2 and tested the connectivity of the tRFC BAPI receiver.

10 Execute your new ABAP code in SAP to make the function call to the new Siebel Workflow.

Receiving IDOC calls

When the tRFC Receiver receives an IDOC from SAP, it passes the information it receives through the `EAI SAP IDOC Workflow Processor` business service. This business service transforms the raw data into an IDOC Integration object and invokes a workflow. The name of the IDOC Integration Object you are using must be defined in a user property on the `EAI SAP IDOC Adapter` business service. The workflow you want to invoke may be defined in two ways. For information on how to define your workflow and IDOC Integration Object, read "IDOC Workflow Processor Configuration" on page 134.

tRFC BAPI Receiver Configuration

The `EAI SAP BAPI Receiver (tRFC)` business service is called by the background Component BAPIRcvr repeatedly to check for tRFC calls from SAP. When a call is received, this business service dispatches the received call to a service which invokes a workflow. IDOCs can be received from SAP in this manner also. For more information on IDOCs received through the `EAI SAP BAPI Receiver (tRFC)` service read "Receiving IDOCs from SAP" on page 133.

Table 26 summarizes the User Properties and Component Parameters that may be set to control its behavior. Those variables that may be set as component parameters must be set for the BAPIRcvr component.

Table 26. tRFC BAPI Receiver Configuration Options

Name	User Prop	Comp Param	Valid Values	Usage
SAPRfcDestEntry		X		SAP Destination - references a TYPE=R entry in the `saprfc.ini` file.
SAPBAPIDispatchService	X	X		Business Service to dispatch RFC calls received. This must be a service based on CSSWfEngine.
SAPBAPIDispatchMethod	X	X		Business service method to call in SAPBAPIDispatchService.
SAPIDOCDispatchService	X	X		Business service to dispatch IDOCs received.
SAPIDOCDispatchMethod	X	X		Business Service method to call in SAPIDOCDispatchService.
SAPSleepTime	X	X	Integer value in seconds	The business service repeats this cycle: Wait for input from SAP to return to the Siebel application to check for shutdown. The SAPSleepTime is the time spent waiting for input from SAP. Suggested time is 60 seconds. Shorter time periods can cause performance problems.
SAPWriteXML	X	X	"WriteNone", "WriteAlways", "WriteBeforeErr", "WriteOnErr"	Indicates usage of EAI Queue. If value is "WriteNone", queue is not being used. Read Chapter 8, "EAI Queue."
SAPXMLQueueCleanup	X		"true" or "false"	If true, entries are deleted from the EAI Queue upon completion. Read Chapter 8, "EAI Queue."
SAPXMLQueueService	X		XML queuing service name	Provides the name of the Business Service that handles the interface to the EAI Queue. Read Chapter 8, "EAI Queue."

Table 26. tRFC BAPI Receiver Configuration Options

Name	User Prop	Comp Param	Valid Values	Usage
SAPXMLQueueName	X		EAI Queue name	Provides the name of the EAI queue being used to store queue entries created by this service. Read Chapter 8, "EAI Queue."
RfcFunctionIntObjnn	X			Name of BAPI Input Integration Object that can be accepted by the tRFC Receiver. Read the following note.
SAPCodepage	X	X	Value from Transcode Encoding drop-down list	Siebel code page name for SAP code page.
SAPIgnoreCharSetConv Errors	X	X	True or False	Default is False. If True, code page conversion errors are handled by substitution of the question mark(?) character in place of an unrecognized character.

NOTE: There may be up to 50 user properties defined with a name of the form, RfcFunctionIntObjnn, such as RfcFunctionIntObj01, RfcFunctionIntObj02, RfcFunctionIntObj03, and so on. Each of these user properties would contain the name of a BAPI Input Integration Object that can be accepted by the tRFC BAPI Receiver.

7 IDOC Interfaces

This chapter provides reference information for ALE/IDOC interfaces. It contains information on configuration of the IDOC Integration Object Wizard, IDOC Adapter, IDOC Workflow Processor, and IDOC Workflow Service as well as information on IDOC Integration Objects.

Before reading this chapter, you should be familiar with the tRFC BAPI Adapter and tRFC BAPI Receiver described in Chapter 6, "BAPI Interfaces." These Business Services provide the SAP connectivity for transporting IDOCs to and from SAP R/3.

Creating SAP Integration Objects with the IDOC Wizard

This section describes IDOC integration objects in detail and describes configuration options for the IDOC Integration Object Wizard.

IDOC Integration Objects

IDOC integration objects represent the structure of an SAP IDOC. An IDOC received from SAP has a very simple structure. It consists of a single control record followed by multiple data records. The control record contains fields that specify the IDOC type, message type, extension type, sender logical system, receiver logical system, IDOC number, and so on. The data record contains the physical data being transported in the IDOC, as well as some information about the data record, for example, the name of the IDOC segment contained in the data record.

An IDOC integration object in the Siebel application corresponds very closely to this structure. The integration object contains components for the control record and for each data record. Table 27 shows part of the DEBMAS02 IDOC Type and how this would be represented by a Siebel IDOC integration object.

Table 27. DEBMAS02 IDOC and Siebel IDOC

DEBMAS02 IDOC	Siebel IDOC Integration Object
Control Record	IdocSegments Component
E2KNA1M Segment	E2KNA1M Component
E2KNA1H Segment	E2KNA1H Component
E2KNA1L Segment	E2KNA1L Component
E2KNVVM Segment	E2KNVVM Component
E2KNVPM Segment	E2KNVPM Component

The IdocSegments component contains the control record fields. These fields are named with a prefix of "CONTROL-". How these fields are used depends upon whether the IDOC is being sent to SAP or received from SAP. When an IDOC is sent from the Siebel Connector for SAP R/3 to SAP, you may set some control fields in a business service data map. Other fields are populated by the IDOC adapter business service or are populated by SAP. When an IDOC is received from SAP by the Siebel Connector for SAP R/3, you can access the information in the fields from a business service data map.

The fields in the control record may differ depending upon which version of SAP ALE is being used. If you are using SAP R/3 version 3.1, you are using ALE version 3X. If you are using SAP R/3 version 4.0 you should use ALE version 4X, but you can still use 3X if other constraints on your implementation dictate that it must be used. The IDOC Integration Object has a property ALEVersion that can be set to either 3X or 4X. If it is not defined, it is assumed that it is a 3X version. Its initial value is set by the IDOC wizard. Its default value is 3X for SAP R/3 3.1 and is 4X for SAP R/3 4.0 and newer. The default value for this property can be changed by setting the ALEVersion property on the EAI SAP IDOC Adapter Wizard. If you need to convert an existing IDOC Integration Object from 3X to 4X or the reverse, the ALEVersion property must be changed accordingly and the external name for the IDOC Integration Object must be consistent with the 3X or 4X usage as described later in this section.

Table 28 lists the control record fields and indicates if they are used in a 3X or 4X control record. The User Can Populate? column indicates whether you can set the value of the field from a Business Service Data Map for input to SAP. The Mandatory column indicates when the field must be filled in for SAP to process the IDOC. The IDOC Adapter Action column indicates what the IDOC Adapter does to the field prior to sending the IDOC to SAP.

Table 28. Control Record Fields

4X Field	3X Field	User Can Populate?	Mandatory	IDOC Adapter Action
TABNAM	TABNAM	No	Yes	"EDI_DC" or "EDI_DC40"
MANDT	MANDT	No		Filled by SAP.
DOCNUM	DOCNUM	No	Yes	Populate incrementally within a transaction.
DOCREL	DOCREL	Yes		
STATUS	STATUS	Yes		
DIRECT	DIRECT	No	Yes	Set to "2" for inbound direction.
OUTMOD	OUTMOD	No		Must remain blank.
EXPRSS	EXPRSS	Yes		Set to "X" to override batch processing and process IDOC immediately. If set to anything other than blank, an "X" is sent.
TEST	TEST	Yes		Test messages can be indicated with an "X". If set to anything other than blank, an "X" is sent.
IDOCTYP	IDOCTYP	No	Yes	The basic IDOC type. For more information, read "IDOCTYP Information" on page 126.

Table 28. Control Record Fields

4X Field	3X Field	User Can Populate?	Mandatory	IDOC Adapter Action
CIMTYP	CIMTYP	No	Yes if extension is used	Is set to extension type parsed from External Name.
MESTYP	MESTYP	Yes	Yes	If not set, is set to message type user property in integration object.
MESCOD	MESCOD	Yes		
MESFCT	MESFCT	Yes		
STD	STD	Yes		
STDVRS	STDVRS	Yes		
STDMES	STDMES	Yes		
SNDPOR	SNDPOR	Yes		
SNDPRT	SNDPRT	Yes	Yes	User value is used if given. Otherwise defaults to business service method argument or component property (“LS”).
SNDPFC	SNDPFC	Yes		
SNDPRN	SNDPRN	Yes	Yes	User value is used if given. Otherwise defaults to business service method argument or component property.
SNDSAD	SNDSAD	Yes		Field is not currently used by SAP.
SNDLAD	SNDLAD	Yes		
RCVPOR	RCVPOR	Yes		
RCVPRT	RCVPRT	Yes	Yes	User value is used if given. Otherwise defaults to business service method argument or component property (“LS”).
RCVPFC	RCVPFC	Yes		
RCVPRN	RCVPRN	Yes	Yes	User value is used if given. Otherwise defaults to business service method argument or component property.
RCVSAD	RCVSAD	Yes		Field is not currently used by SAP.
RCVLAD	RCVLAD	Yes		
CREDAT	CREDAT	No		Date set by SAP. Left blank.
CRETIM	CRETIM	No		Date set by SAP. Left blank.
REFINT	REFINT	Yes		
REFGRP	REFGRP	Yes		

Table 28. Control Record Fields

4X Field	3X Field	User Can Populate?	Mandatory	IDOC Adapter Action
REFMES	REFMES	Yes		
ARCKEY	ARCKEY	Yes		
SERIAL	SERIAL	Yes		
None	DOCTYP	No	Yes/3X	Populated with External Name field when using the 3X interface.

IDOCTYP Information

DOCTYP, IDOCTYP and CIMTYP are related as in the following examples:

- 3X control record - using SAP defined type or user created IDOC type:
  - DOCTYP - DEBMAS02 - contains same name as IDOCTYP
  - IDOCTYP - DEBMAS02 - basic IDOC type
  - CIMTYP - blank
- 3X control record - using sap extended IDOC
  - DOCTYP - ZDBMAS02 - unique name for extension/basic IDOC type pair
  - IDOCTYP - DEBMAS02 - basic IDOC type
  - CIMTYP - DEBMASEX - extension name
- 4X control record - using sap defined type or user created IDOC type
  - IDOCTYP - DEBMAS02
  - CIMTYP - blank
- 4X control record - using sap extended IDOC
  - IDOCTYP - DEBMAS02 - basic IDOC type
  - CIMTYP - DEBMASEX - extension name

DOCTYP is no longer required as a unique name for the IDOCTYP/CIMTYP pair in SAP R/3 version 4.0 or newer.

NOTE: DOCTYP can be populated in a 3X control record by a 4X implementation by maintaining the conversion table in the IDOC type editor (we30). Navigate to Environment > Conversion > IDOC Type and enter a unique DOCTYP name for the basic type/extension type pair being used.

The integration object has an external name as well as a Siebel name. The External Name field is specially encoded when an IDOC Extension is being used. The external name for the integration object is defined as follows:

1. If you are using ALE 3X and the IDOC is not an extension:

   The external name holds the 3X Idoc type name (value as entered in DOCTYP of the 3X control record).

2. If you are using ALE 3X version and the IDOC is an extension:

   The external name holds the 3X Idoc type name (value as entered in DOCTYP of the 3X control record). Although the IDOCTYP and CIMTYP could be filled in the control record, they are not required and are filled in by SAP.

   NOTE: If the SAP version is 4.0 or above and if you must use a 3X ALE interface and an extension IDOC type, first enter the DOCTYP name to be used in SAP and then enter this name in the external name field of the integration object. The wizard does not capture this name automatically.

3. If you using an ALE 4X version and the IDOC is not an extension:

   The external name holds the 4X basic IDOC type name (value as entered in IDOCTYP of the 4X control record).

4. If you using an ALE 4X version and the IDOC is an extension:

   The external name holds both the 4X basic IDOC type name (values as entered in IDOCTYP of the 4X control record) and the extension type (CIMTYP) in the form: IDOCTYP/CIMTYP.

The IDOC Integration Object has a user property Workflow that can be set to the name of a workflow to process upon receipt of the integration object from SAP. This is used only when an IDOC is received from SAP. It is not used when an IDOC is sent to SAP. For more information on the usage of this user property, read "Receiving IDOCs from SAP" on page 133.

Table 29 summarizes the IDOC integration object user properties.

Table 29. IDOC Integration Object User Properties

Name	Valid Values	Usage
Logical Message Type	SAP IDOC Message Type	Filled in by the IDOC integration object wizard. You do not need to modify this.
Workflow	Siebel Workflow name	You need to populate this if the IDOC is to be sent from SAP to the Siebel application.
ALEVersion	"3X" or "4X"	Set by the IDOC Integration object wizard. You do not need to modify this.

In addition, the user property Offset appears for each field in the IDOC. This field is set by the IDOC integration object wizard and you must not change it.

When selecting IDOC objects to capture with the wizard, the list contains IDOC Type, Extension Type, and Message Type.

- If the IDOC does not represent an Extension Type you see:
  IDOC Type (*Message Type*)
  For example, MATMAS03 (MATMAS)

- If the IDOC is an extension type, you see:
  IDOC Type/Extension Type (*Message Type*)
  For example, MATMAS03/ZMATMAS (ZMATMS)

IDOC Wizard Configuration

Some changes can be made to the IDOC Wizard business service to change the behavior of the wizard. User properties may be set as defined in Table 30.

Table 30. IDOC Wizard Configuration Values

Name	Valid Values	Usage
SAPIdocVersion	Any SAP version less than or equal to the SAP Version	Indicates the version of the IDOC to be retrieved from SAP. This is used only if the SAP version is 4.5A or newer. It causes the wizard to select a version of an IDOC other than the most recent version. For example, if you are working with SAP R/3 4.6C, you can retrieve a 3.1H IDOC by setting the SAPIdocVersion to "31H". If this was not set, a 4.6C IDOC would be retrieved.
ALEVersion	3X, 4X	Defines the default value for the IDOC Integration Object User Property ALEVersion (corresponds to SAP versions 2 or 3). If not set defaults to 3X for SAP R/3 3.1 and 4X for SAP R/3 4.0 or newer.

When you make any changes to user properties in the IDOC wizard business service, you must compile these changes into the .srf file.

To change user properties

1. Make a copy of the .srf file used by your Siebel Tools application.
2. Start Siebel Tools.
3. Lock the SAP Business Services project.
4. Add or change the desired user properties on the EAI SAP IDOC Adapter Wizard business service as in Table 30.
5. Compile the SAP Business Services project into your .srf file copy.
6. Exit Siebel Tools.
7. Copy the modified .srf file to the name of your old .srf file.
8. Restart Siebel Tools.

When you invoke the wizard it now uses the new user property definitions.

IDOC Integration

The Siebel Connector supports both sending and receiving IDOCs to and from SAP. In each case, the EAI Queue may be used for secure data transport. This functionality is performed through a series of business services provided by the Siebel application. Other sections of this document refer to these business services by the names in the following list.

Reference Name	Business Service Name
IDOC Adapter	`EAI SAP IDOC Adapter`
IDOC Workflow Processor	`EAI SAP IDOC Workflow Processor`
IDOC Workflow Service	`EAI SAP IDOC Workflow Service`
Send Transaction Service	`EAI SAP Send Transaction Service`
Process Transaction Service	`EAI SAP Process Transaction Service`

Configuration requirements for Business Services for each of these options are discussed in the following sections. The behavior of Business Services can be controlled in three ways.

- User Properties
- Component Parameters
- Method Arguments

The values of User Properties are set within Siebel Tools by selecting Business Service > Business Service User Prop from the Object Explorer and editing the Value field for the specific User Property. Changes to User Property values must be compiled into the .srf file.

The values of Component Parameters may be set by default or by you. You can change parameter values through the Server Manager. For more information on setting Component Parameters see *Siebel System Administration Guide*. If a Component Parameter has the same name as a User Property, its value overrides any previously defined value of the User Property.

The values of method arguments are set when a method in a business service is invoked. Values for method arguments on services that are invoked from a workflow are defined within a workflow. For this reason they can have fixed or variable values. If a method argument and a business service user property have the same name, the value of the method argument overrides the value of the user property. If a method argument and a component parameter have the same name, the value of the method argument overrides the value of the component parameter.

To summarize, the value of user properties can be overridden with component parameter or method argument values and the value of a component parameter can be overridden with a method argument value.

Each section in this chapter provides information on user properties, component parameters and method arguments that may be used for each business service. An explanation of how to incorporate these business services into a workflow is provided in "Creating Workflows" on page 97.

Sending IDOCs to SAP

As explained in Chapter 5, "Customizing Integrations," the IDOC adapter is responsible for converting an IDOC integration object into the raw data required by the tRFC BAPI adapter and invoking the tRFC BAPI Adapter to send the IDOC to SAP. The `EAI SAP IDOC adapter` service is the business service that represents the IDOC adapter. This topic provides additional information on the configuration and use of this service.

Table 31 and Table 32 summarize the user properties, method arguments, and component parameters that may be set to control the behavior of the `EAI SAP IDOC adapter`. The `EAI SAP IDOC adapter` is a cached business service. Variables that are defined to be user properties in the business service are generally used only once for initialization purposes at the time of the first call to the business service. Variables that are defined to be method arguments are used on each call and may vary from call to call. Those variables that may be set as component parameters must be set for the Business Integration Manager component.

Table 31 contains User Properties, Component Parameters and Input Method Arguments for the Send Method.

Table 31. IDOC Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPRfcConnectString		X	X		SAP connection string - read "Modifying Siebel Configuration Files" on page 23 for format. This string defines the destination (used in `saprfc.ini`), client and language for the SAP connection.
SAPRfcUserName		X	X		SAP user name.
SAPRfcPassword		X	X		SAP password.
SAPRfcTrace		X	X	true or false	SAP trace file usage on or off. Same as TRACE=0 or 1 in `saprfc.ini` file.
SiebelMessage			X		EAI Siebel Message containing IDOC integration object instance.
SAPReceiverPrtnrNum		X	X	Logical system name	Logical system for the SAP client you are sending the IDOC to.
SAPReceiverPrtnrType		X	X	Partner type	Defaults to LS if not set for logical system.

Table 31. IDOC Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPSenderPrtnrNum		X	X	Logical system name	Logical system name for your external system. This name must be set up in SAP. Read "Creating Logical Systems Within SAP" on page 28.
SAPSenderPrtnrType		X	X	Partner type	Defaults to LS if not set for logical system.
DispatchFormat	X			"ALE"	Currently set to ALE for the IDOC adapter. This should not be changed.
DispatchService	X			Business service name	Currently set to the `EAI SAP BAPI adapter (tRFC)` service. This should not be changed.
DispatchMethod	X			Method name	Currently set to the Execute method of the `EAI SAP BAPI adapter (tRFC)` service. This should not be changed.
SAPWriteXML		X		"WriteOnly", "WriteNone", "WriteOnErr", "WriteBefore Err"	If set, overrides the user property setting for this in the `EAI SAP BAPI adapter (tRFC)` service.
SAPXMLQueueCleanup		X		"true" or "false"	If set, overrides the user property setting for this in the `EAI SAP BAPI adapter (tRFC)` service.
SAPXMLQueueService		X		XML queuing service name	If set, overrides the user property setting for this in the `EAI SAP BAPI adapter (tRFC)` service.
SAPXMLQueueName		X		EAI Queue name	If set, overrides the user property setting for this in the `EAI SAP BAPI adapter (tRFC)` service.

Table 31. IDOC Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPCodepage	X	X	X	Value from Transcode Encoding drop-down list	Siebel code page name for SAP code page.
SAPIgnoreCharSetConvErrors	X	X	X	True or False	Default is False. If True, code page conversion errors are handled by substitution of the question mark (?) character in place of an unrecognized character.

SAPRfcConnectString uses SAP's internal connect string format. This must include the following elements:

- DEST=*Destination name from saprfc.ini file*
- CLIENT=*3-digit SAP client number*
- LANG=*single character language indicator*

These elements must separated in the string by one or more spaces, for example, "DEST=DEV_Outbound CLIENT=555 LANG=E". Table 32 contains output method arguments for the Send Method.

Table 32. Send Method Arguments for Output

Name	Usage
DispatchErrorStatus	Contains the RFC exception string returned by SAP. If connection to SAP fails, this string begins with the text: "RFC Exception: __SIEBEL__SAP_CONNECT_FAILED__"

If the value of SAPWriteXML is set to anything other than "WriteNone", the EAI Queue is used. For more information on the EAI Queue, read Chapter 8, "EAI Queue." When sending IDOCs to SAP the queue is accessed by the `EAI SAP BAPI adapter (tRFC)` service.

Receiving IDOCs from SAP

The EAI SAP BAPI receiver (tRFC) service may receive IDOCs from SAP as well as BAPI calls. When an IDOC is received, the EAI SAP IDOC Workflow Processor service is invoked to convert the raw data received from SAP into an IDOC integration object. This service then invokes the EAI SAP IDOC Workflow Service to invoke a workflow in the Siebel application. This topic provides additional information on the configuration and usage of the EAI SAP IDOC Workflow Processor service and the EAI SAP IDOC Workflow Service.

IDOC Workflow Processor Configuration

The business service `EAI SAP IDOC Workflow Processor` is called by the tRFC BAPI receiver when it receives an IDOC. Table 33 summarizes the user properties and component parameters that may be set to control its behavior. Those variables that may be set as component parameters must be set for the BAPIRcvr component.

Table 33. IDOC Workflow Processor Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SiebelMessage			X		Integration Object with raw data from the tRFC BAPI receiver.
DispatchFormat	X			"ALE"	Currently set to "ALE". This should not be changed.
DispatchService	X				Business Service to send IDOC integration object to workflow. This must be a service based on CSSWfEngine. Currently set to `EAI SAP IDOC Workflow Service`.
DispatchMethod	X				Business service method to call in DispatchService.
SAPIdocAllowedObjectnn	X				Name of IDOC Integration Object that can be accepted by the workflow processor. Read and the following note and caution.
SAPCodepage	X	X	X	Value from Transcode Encoding drop-down list	Siebel code page name for SAP code page.
SAPIgnoreCharSetConvErrors	X	X	X	True or False	Default is False. If True, code page conversion errors are handled by substitution of the question mark (?) character in place of an unrecognized character.

NOTE: There may be up to 99 user properties defined with a name of the form, SAPIdocAllowedObjectnn, such as SAPIdocAllowedObject01, SAPIdocAllowedObject02, SAPIdocAllowedObject03, and so on. Each of these would contain the name of an IDOC Integration

Object that can be accepted by the IDOC Workflow Processor.

CAUTION: When an IDOC is received from SAP, the External Name is extracted from information in the control record of the IDOC. This name is then compared with the External Names for each integration object listed in the user properties SAPIdocAllowedObjectnn. When a match on the External Name is found, the corresponding IDOC integration object is used. For this reason, each IDOC integration object defined by an SAPIdocAllowedObjectnn user property must have a distinct external name. For example, two IDOC integration objects that both have the external name of DEBMAS02 cannot be used at the same time.

Invoking Workflows

After an IDOC is received and passed to the Workflow Processor, the Workflow Processor must pass this IDOC integration object on to a specific workflow. The workflow name can be defined in two ways:

1. The Workflow User Property on the IDOC Integration Object definition in Siebel Tools can be set to the workflow name. When this type of IDOC Integration Object is received, it is sent to this workflow name. The workflow name must be set using Siebel Tools and the changed integration object must be compiled into the Siebel .srf file.
2. The workflow name may be set in the ProcessName user property of the `EAI SAP IDOC Workflow Service` business service. If the workflow name is set here, *all* IDOCs received from SAP are sent to this workflow. The workflow name in the integration objects is ignored.

Standard integrations have been set up to use the first method. All workflow names are set in the Workflow User Property of the IDOC Integration Objects.

Sending IDOCs with MQSeries

You can use MQ Series to transport IDOCs from the Siebel application to SAP. The following sections discuss how to configure your implementation to use the `EAI SAP IDOC MQ AMI Adapter` business service to send IDOCs through MQ Series.

Configuring the MQSeries Adapter

The `EAI SAP IDOC MQ AMI Adapter` business service can be invoked from workflow. See the following section for more information on how to do this. The following tables summarize the User Properties, Method Arguments, and Component Parameters that may be set to control its behavior. The `EAI SAP IDOC MQ AMI Adapter` is a cached business service. Variables that are defined to be User Properties in the Business Service are generally used only once for initialization purposes at the time of the first call to the Business Service. Variables that are defined to be Method Arguments are used on each call and may vary from call to call.

Table 34 contains User Properties, Component Parameters, and Input Method Arguments for the Send Method.

Table 34. IDOC MQ AMI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SiebelMessage			X		EAI Siebel Message containing IDOC integration object instance.
SAPReceiverPrtnrNum		X	X	Logical system name	Logical system for the SAP client you are sending the IDOC to.
SAPReceiverPrtnrType		X	X	Partner type	Defaults to LS if not set for logical system.
SAPSenderPrtnrNum		X	X	Logical system name	Logical system name for your external system. This name must be set up in SAP. Read "Creating Logical Systems Within SAP" on page 28.
SAPSenderPrtnrType		X	X	Partner type	Defaults to LS if not set for logical system.
DispatchFormat	X			"MQ"	Currently set to MQ for the AMI Adapter. This should not be changed.
SAPCodepage	X	X	X	Value from Transcode Encoding drop-down list	Siebel code page name for SAP code page.
SAPIgnoreCharSetConvErrors	X	X	X	True or False	Default is False. If True, code page conversion errors are handled by substitution of the question mark (?) character in place of an unrecognized character.

In addition, the default format on the AMI service point needs to be set to MQSTR or MQHSAP depending upon whether or not a user exit is used. For more information, consult your MQSeries documentation.

Invoking the MQSeries Adapter from Workflows

If you are sending an IDOC to SAP and want to use MQSeries as the transport, use the sample workflow "Send46CIdoc_UpdateSAPCustomer_MQAMI" or create your own workflow. To create a new workflow, add the EAI SAP IDOC MQ AMI Adapter business service and the EAI MQSeries AMI Transport as a minimum, and you need to add *Value* as a process property.

To add Value as a process property

1. Using Process Properties of Workflow Process editor, add a new record.
2. For the Name field enter *Value*.
3. For the Data Type field, specify String.

To add the EAI SAP IDOC MQ AMI Adapter business service

1. Using the Workflow Diagram Editor, drag a Business Service box to your diagram.
2. Use a connector arrow to link the box into your flow.
3. Double-click on the Business Service box to open the business service arguments screen.
4. Edit the Name field to add a name for your step.
5. From the drop-down list for Business Services, choose the `EAI SAP IDOC MQ AMI Adapter`.
6. From the drop-down list for Business Service Methods, choose the Send method.
7. Enter appropriate input arguments.

   At a minimum, you must pass an EAI SiebelMessage into the IDOC adapter which contains an IDOC Input integration object you have created with the Integration Object wizard. The SAPSenderPtnrNum input argument must either be set or your map must set the sender partner field in the IDOC control record.
8. Enter appropriate output argument. You must set an output argument as *Value* which is defined from Process Property.

To add EAI MQSeries AMI Transport business service

1. Using the Workflow Diagram Editor, drag a Business Service box to your diagram.
2. Use a connector arrow to link the box into your flow.
3. Double-click on the Business Service box to open the business service arguments screen.
4. Edit the Name field to add a name for your step.
5. From the drop-down list for Business Services, choose the EAI MQSeries AMI Transport.
6. From the drop-down list for Business Service Methods, choose the Send method.

7 Enter appropriate input arguments.

You must pass *Value* into the transport which is the output from `EAI SAP IDOC MQ AMI Adapter`. To allow binary data to be exchanged, set the input argument IsReceivingTextData to false. In addition, set the Connection Subsystem argument to a name of your choice.

8 Create a named subsystem on the server with the same name as that defined in the Connection Subsystem argument.

On this subsystem, define the values you are using for MqPolicyName and MqSenderServiceName. For example, using server manager command line mode the command to create the subsystem would look like this:

```
create named subsystem your connection subsystem name for subsystem
MQSeriesAMISubsys with MqPolicyName=your MQ AMI Policy name,
MqSenderServiceName=your MQ AMI Sender Service name
```

The IDOC MQ AMI Adapter converts the IDOC to a buffer which MQSeries can understand, and passes this to the AMI Transport which sends the IDOC to the MQSeries Queue. For more information on MQSeries, read Chapter 2, "EAI MQSeries Transport" in *Transports and Interfaces: Siebel eBusiness Application Integration Volume III*.

Receiving IDOCs with MQSeries

The EAI MQSeries AMI Transport Service is used for this type of communication with MQSeries. This service is called repeatedly from the SAP IDOC AMI Receiver for MQ Series (alias SAPIdocAMIMqRcvr) component. When an IDOC is placed in the MQSeries queue, the EAI MQSeries AMI Transport Service retrieves the IDOC from the queue and invokes the `EAI SAP IDOC MQ AMI Workflow Processor`. This service extracts the IDOC from an MQSeries buffer and creates an IDOC Integration Object. The IDOC Integration Object is passed to the `EAI SAP IDOC Workflow Service` which invokes the appropriate workflow.

Configuring the MQSeries Workflow Processor

Table 35 summarizes the User Properties and Component Parameters that may be set to control the behavior of the `EAI SAP IDOC MQ AMI Workflow Processor` service method ReceiveDispatch. Those variables that may be set as component parameters must be set for the SAPIdocAMIMqRcvr component.

Table 35. IDOC MQ AMI Workflow Processor Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
Value			X		Raw data buffer from MQSeries.
DispatchFormat	X			"MQ"	Currently set to "MQ". This should not be changed.
DispatchService	X				Business service to send IDOC integration object to workflow. This must be a service based on CSSWfEngine, and is currently set to `EAI SAP IDOC Workflow Service`.
DispatchMethod	X				Business service method to call in DispatchService.
SAPIdocAllowedObjectnn	X				Name of IDOC integration object that can be accepted by the workflow processor. Read the following note and caution.
SAPCodepage	X	X		Value from Transcode Encoding drop-down list	Siebel code page name for SAP code page.
SAPIgnoreCharSetConvErrors	X	X		True or False	Default is False. If True, code page conversion errors are handled by substitution of the question mark (?) character in place of an unrecognized character.

NOTE: There may be up to 99 user properties defined with a name of the form, SAPIdocAllowedObjectnn, such as SAPIdocAllowedObject01, SAPIdocAllowedObject02, SAPIdocAllowedObject03, and so on. Each of these would contain the name of an IDOC Integration Object that can be accepted by the IDOC MQ AMI Workflow Processor.

CAUTION: When an IDOC is received from MQSeries, the External Name is extracted from information in the control record of the IDOC. This name is then compared with the External Names

for each integration object listed in the user properties SAPIdocAllowedObjectnn. When a match on the External Name is found, the corresponding IDOC Integration Object is used. For this reason, each IDOC Integration Object defined by an SAPIdocAllowedObjectnn user property must have a distinct external name. For example, two IDOC Integration Objects that both have the external name of DEBMAS02 cannot be used at the same time.

Invoking Workflows

After an IDOC is received and passed to the Workflow Processor, the Workflow Processor must pass this IDOC Integration Object on to a specific workflow. The workflow name to which this IDOC Integration Object is passed must be defined. For information on how to define the workflow name, read "Receiving IDOCs from SAP" on page 133.

Starting the MQSeries Server for AMI

Starting the SAP IDOC AMI Receiver for MQ Series (alias SAPIdocAMIMqRcvr) component as a background process is very similar to starting the tRFC BAPI Receiver component. For more information, read "Starting the tRFC BAPI Receiver" on page 39. The following are required component parameters for starting this receiver:

- **ReceiverConnectionSubsystem**. Named subsystem that represents the MQ AMI connection.
- **ReceiverDataHandlingSubsystem.** Named subsystem that represents the MQ AMI data handling.

Read Chapter 2, "EAI MQSeries Transport" in *Transports and Interfaces: Siebel eBusiness Application Integration Volume III* for more information on the EAI MQSeries AMI Transport service on which this component is based.

8 EAI Queue

This chapter describes the EAI Queue, which is used by the Siebel Connector for SAP R/3. This information helps you understand the processes that the Siebel Connector for SAP R/3 uses, which may be valuable when creating your integrations.

About the EAI Queue

The EAI Queue provides temporary storage for data in transit between Siebel eBusiness applications and external applications and a means by which to monitor the data exchange. This temporary storage of data can facilitate error recovery in the event that the flow of data to or from the Siebel Connector for SAP R/3 is interrupted.

Each entry in the EAI Queue may contain the following information:

- XML file containing the data object in transit
- Processing status of data object
- Reference ID for data object in external application
- Additional fields to be used for error information and other external application specific information.

Siebel administrators can view the data in the queue and its current processing status. If errors occur while the data is in transit, the external system reference ID and error information can be used to determine the problem, correct it and restart the data flow between the Siebel application and the external application.

To view entries in the EAI Queue, navigate to Administration - Integration > EAI Queue. For SAP there are two queues, tRFC Inbound from SAP, and tRFC Outbound to SAP. Select a queue in the upper list. Queue entries then appear in the lower form. You can edit and delete queue entries, however, be aware that if the Siebel Server is running, queue entries may be in progress. You can safely edit and delete queue entries with the following statuses: Confirmed, Error in Processing, and Error in Sending. Entries having any other status value may be in progress while the server is running. Do not edit or delete entries in this case.

A business service API is also provided to the EAI Queue. This business service contains methods to update information held in the queue and allows other components in the Siebel application, as well as customers, to develop software to use the queue. The business service contains the following methods:

- **AddMessage.** Adds an XML data object to the queue.
- **GetMessage.** Retrieves an XML data object from the queue.
- **GetStatus.** Retrieves processing status of the data object.
- **UpdateStatus.** Updates the processing status of the data object.

- **DeleteMessage.** Removes an XML data object from the queue.

This chapter explains the usage of the EAI Queue within the Siebel Connector for SAP R/3 and defines the EAI Queue business service methods and configuration options.

EAI Queue Usage with SAP R/3 tRFC

The EAI Queue is used in Siebel 7 in conjunction with SAP's Transactional Remote Function Call (tRFC) interface. Applications that are external to SAP can use the tRFC interface to communicate with SAP in an asynchronous manner.

Outbound from the Siebel Application

Figure 26 shows the current implementation of the EAI Queue when making tRFC calls from the Siebel application to SAP R/3.

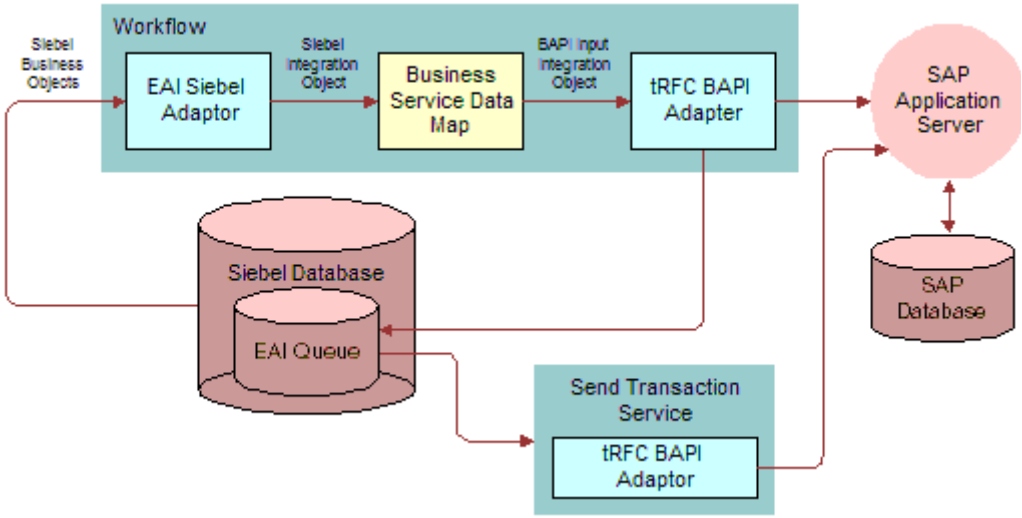


Figure 26. Outbound from the Siebel Application Using tRFC

Figure 26 shows the flow of a business object from the Siebel database to the SAP database. From an EAI workflow, the EAI Siebel Adapter is invoked to extract data from the Siebel database corresponding to a Siebel business object definition. This data is used by the Siebel Adapter to create a Siebel integration object instance. This is passed through a transformation map service to create a BAPI input integration object instance. The BAPI input integration definition contains the structure of the data object to be passed to SAP in the tRFC call.

When the tRFC BAPI Adapter receives the integration object instance it behaves in one of the following ways depending upon the setting of the SAPWriteXML component parameter:

- **WriteNone.** EAI Queue is not used, the data is sent directly to SAP.

- **WriteOnErr.** Should the network or SAP be unavailable or an error is returned from SAP upon the tRFC call, the data in transit is saved in the EAI Queue. If data cannot be saved in the queue an error is returned to the caller (usually workflow). It is then up to workflow to handle the error.
- **WriteBeforeErr.** Data in transit is stored in the EAI Queue prior to the call to SAP. If the network or SAP is unavailable or an error is returned from SAP upon the tRFC call, the data has already been saved in the EAI Queue and can be resent to SAP by the Send Transaction service. The caller receives an error only if the data cannot be saved in the queue initially. WriteBeforeErr always creates a queue entry regardless of error status.
- **WriteOnly.** Data in transit is placed in the EAI Queue and no immediate call is made to SAP to send the data. The Send Transaction service then sends the data at a later time. The caller receives an error only if the data cannot be saved in the queue.

Table 36 summarizes the possible status values for a data object stored in the queue during processing from the Siebel application to SAP in each of the preceding modes.

Table 36. Possible Status Values for EAI Queue Entries (Siebel to SAP)

SAPWriteXML Value	Result of Workflow	Entry in EAI Queue (Status)	Data Passed to SAP	SAP Network Error While Data Passed	Entry in EAI Queue (Status)
WriteOnly	Success	Yes (Initial)	No		
	Fail	No	No		
WriteNone	Success	No	Yes	Yes	No
				No	No
	Fail	No	No		
WriteOnErr	Success	No	Yes	Yes	Yes (Initial or Sent)
				No	No
	Fail	No	No		
WriteBeforeErr	Success	Yes (Initial)	Yes	Yes	Yes (Initial or Sent)
				No	Yes (Confirmed)
	Fail	No	No		

Inbound to the Siebel Application

Figure 27 shows the current implementation of the EAI Queue when making tRFC calls from SAP R/3 to the Siebel application. The SAP Application Server extracts a record from the SAP database, and sends it to Siebel's tRFC BAPI Receiver which forwards it to the EAI Queue. The EAI Queue then forwards the record to a Process Transaction Service, which sends it to the tRFC BAPI Receiver which creates a BAPI Input Integration Object. This object goes to a workflow, which consists of a business service data map and the Siebel EAI Adapter. The adapter writes the record to the Siebel database.

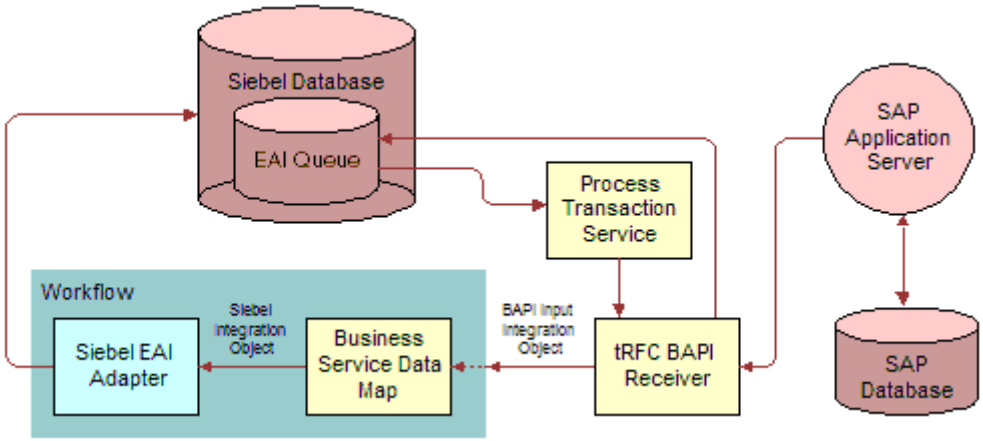


Figure 27. Inbound to the Siebel Application Using tRFC

Figure 27 shows the flow of a business object from SAP to the Siebel database. When SAP makes a tRFC call to the Siebel Connector for SAP R/3, it is received by the tRFC BAPI Receiver. This receiver is responsible for invoking the correct workflow for this particular tRFC call.

The data object transferred to the Siebel application in the form of the tRFC call may be saved in the EAI Queue depending upon the success or failure of workflow processing and the value of the SAPWriteXML component parameter. The possibilities are as follows:

- **WriteNone.** EAI Queue is not used, and a workflow is invoked. If an error occurs in workflow the error shows up in the tRFC layer in SAP.
- **WriteOnErr.** When a call is received from SAP and an error occurs in the workflow invoked in the Siebel application, the data in transit is saved to the EAI Queue. If the data cannot be saved in the queue, an error is returned to the tRFC layer in SAP. If the data is saved in the queue successfully, no error is returned to SAP and error handling occurs in the Siebel application.
- **WriteBeforeErr.** When a call is received from SAP, the data is immediately saved in the EAI Queue. The workflow is then invoked in the Siebel application, and if it fails the data is reprocessed by a Process Transaction Service at a later time. An error is returned to the tRFC layer in SAP only if the data cannot be saved in the queue initially. WriteBeforeErr always creates a queue entry regardless of error status.

- **WriteOnly.** When a call is received from SAP, the data is immediately saved in the EAI Queue and no immediate attempt is made to call workflow. The workflow is invoked at a later time by a Process Transaction Service. An error is returned to the tRFC layer in SAP only if the data cannot be saved in the queue.

Table 37 summarizes the possible status values for a data object stored in the queue during processing from SAP to the Siebel application in each of the preceding modes.

Table 37. Possible Status Values for EAI Queue Entries (SAP to the Siebel Application)

Inbound from SAP	Result of Workflow	SAP Network Error	Entry in EAI Queue (Status)	Transaction Errors in tRFC Layer
WriteOnly	N/A	N/A	Yes (Initial)	No
WriteNone	Success	N/A	No	No
	Failed	N/A	No	Yes
WriteOnErr	Success	N/A	No	No
	Failed	N/A	Yes (Initial)	No
WriteBeforeErr	Success	No	Yes (Confirmed)	No
		Yes	Yes (Initial or Processed)	No
	Failed		Yes (Initial)	No

EAI Queue Usage with SAP R/3 ALE

The EAI Queue can be used when exchanging IDOCs with SAP using ALE. The SAP ALE interface is actually a series of tRFC calls to and from SAP. IDOCs are passed to and from SAP through these tRFC calls. For this reason the processing of IDOCs with the EAI Queue is very similar to the processing of tRFC calls.

Outbound from the Siebel Application

Figure 28 shows the current implementation of the EAI Queue when making tRFC calls from the Siebel application to SAP R/3. From an EAI workflow, the EAI Siebel Adapter is invoked to extract data from the Siebel database corresponding to a Siebel business object definition. This data is used by the Siebel Adapter to create a Siebel integration object instance. This is passed through a transformation map service to create a BAPI input integration object instance. The BAPI input integration definition contains the structure of the data object to be passed through an IDOC Adapter to SAP in the tRFC call.

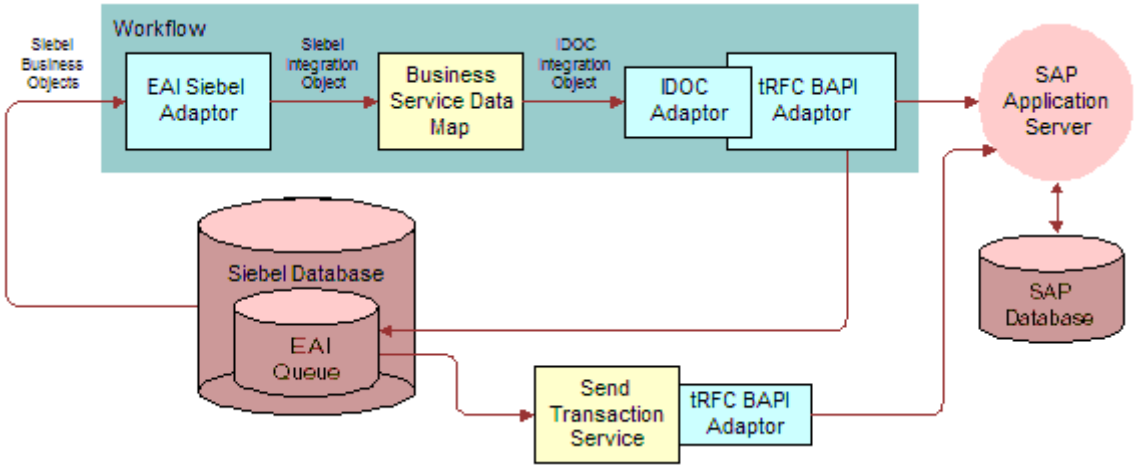


Figure 28. Outbound from the Siebel Application Through ALE

Processing is the same as with tRFC calls except for the use of the IDOC Adapter. This adapter converts the IDOC Integration object to the raw tRFC call format required for the actual call to SAP. Consequently, when an IDOC is viewed in the queue, and it is in the form of a tRFC call to SAP, it is not in the form of the IDOC integration object.

Possible processing modes are as defined in "EAI Queue Usage with SAP R/3 tRFC" on page 142.

Inbound to the Siebel Application

Figure 29 shows the current implementation of the EAI Queue when making tRFC calls from SAP R/3 to the Siebel application. The SAP Application Server extracts a record from the SAP database, and sends it to Siebel's tRFC BAPI Receiver which forwards it to the EAI Queue. The EAI Queue then forwards the record to a Process Transaction Service, which sends it to the IDOC Workflow Processor which creates a BAPI Input Integration Object. This object goes to a workflow, which consists of a business service data map and the Siebel EAI Adapter. The adapter writes the record to the Siebel database.

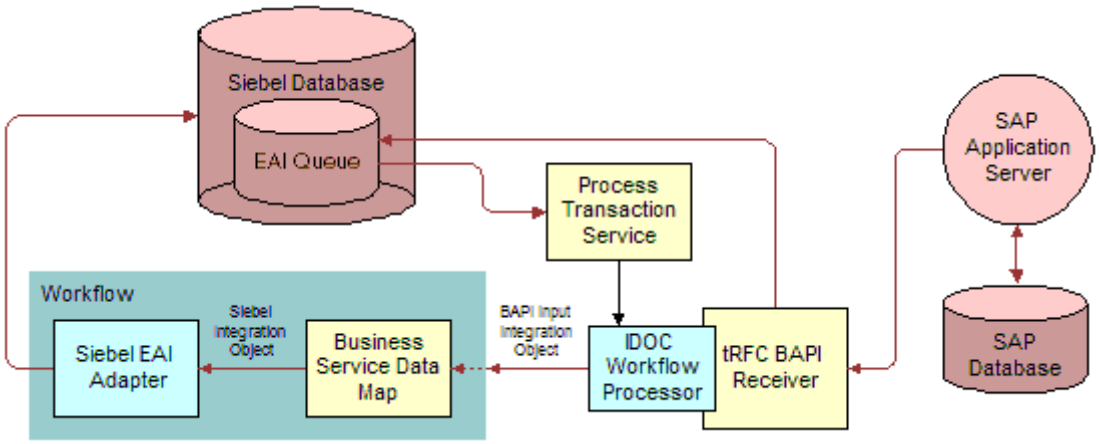


Figure 29. Inbound to the Siebel Application Using ALE

Processing is the same as with tRFC calls except for the use of the IDOC Workflow Processor. This adapter converts the raw tRFC call format required for the actual call received from SAP to the IDOC Integration object. Consequently, when an IDOC is viewed in the queue, and it is in the form of a tRFC call to SAP, it is not in the form of the IDOC integration object.

Possible processing modes are as defined in "EAI Queue Usage with SAP R/3 tRFC" on page 142.

EAI Queue Usage

The use of the EAI Queue is enabled through changes to the SAPWriteXML component parameter on the BAPIRcvr and BusIntMgr components. The default value for this component parameter is "WriteOnErr".

To enable the Siebel Application to SAP interfaces

1. Set the SAPWriteXML component parameter on the BusIntMgr component to one of the following choices:
   - WriteBeforeErr
   - WriteOnly

- WriteOnErr
- WriteNone

2 Restart the Siebel server.

3 Start a Send Transaction Service background task to pick up erred transactions from the queue and resend them to SAP.

For more information, read "The Send Transaction Service" on page 150.

4 Test the queue by executing the Account to Customer Standard Integration.

The data flow that sends an IDOC to SAP to create sales area information for the customer creates an entry in the queue, depending upon the mode.

5 Choose Integration Administration EAI Queue from the Siebel application.

6 Select the "tRFC Outbound to SAP" queue from the EAI Queue applet.

Entries in the queue appear in the EAI Queue Items applet. Each entry contains:

a The Transaction ID used to identify the transaction to SAP in the Reference ID field.

b The data in transit as an XML file attachment.

c The Status field indicating the current status.

Status values for Siebel application to SAP Send Transaction are Initial, Sent, or Confirmed. If the process errors, status remains in one of these three states until a Send Transaction Process executes. If the Send Transaction Process errs, the status appears as "Error in Sending." At this point, manual intervention is needed to determine the cause of the error.

7 If a transaction has the status "Error in Sending" in the EAI Queue, make a note of the Transaction ID in the Reference ID field.

This value uniquely identifies the transaction in the Siebel log file. Using this ID, you can find error information in the log file to help you determine the cause of the problem. When you determine the cause you may do the following:

a Edit the XML data held in the EAI queue for that transaction to correct the information and change the status from "Error in Sending" to "Initial". A Send Transaction service picks up the data and resends it to SAP.

b Delete the entry in the EAI Queue for that transaction and recreate it from the Siebel application using whatever correction may have been made.

To enable the SAP to Siebel interfaces

1 Set the component parameter SAPWriteXML on the component BAPIRcvr so that it contains:

- WriteBeforeErr
- WriteOnly
- WriteNone
- WriteOnErr

2 Start a BAPIRcvr component.

For more information, read "Starting the tRFC BAPI Receiver" on page 39.

3 Start a Process Transaction Service background task to pick up erred transactions from the queue and reprocess them into the Siebel application.

For more information, read "The Process Transaction Service" on page 151.

4 Execute the Customer to Account standard integration to test the queue.

The data flow that sends an IDOC to the Siebel application to create an account creates an entry in the queue, depending upon the mode chosen.

5 Choose Integration Administration EAI Queue to display the queue.

6 Select the "tRFC Inbound from SAP" queue from the EAI Queue form.

Entries in the queue appear in the EAI Queue Items list. Each entry contains:

a The Transaction ID used to identify the transaction to SAP in the Reference ID field.

b The data in transit as an XML file attachment.

c Status field indicating the current status.

Status values for SAP to Siebel applications Send Transaction are Initial, Processed and Confirmed. If the process errors, status remains in one of these three states until a Process Transaction Service executes. If the Process Transaction Service errors, status appears as "Error in Processing". At this point, manual intervention is needed to determine the cause of the error.

7 If a transaction has the status "Error in Processing" in the EAI Queue make a note of the Transaction ID in the Reference ID field.

This value uniquely identifies the transaction in the Siebel log file and also in the SAP tRFC layer. Using this ID you can find error information in the log file to help you determine the cause of the problem. You can also retrieve information from SAP by SAP transaction SM58. When you determine the cause you may do the following:

a Edit the XML data held in the EAI queue for that transaction to correct the information and change the status from "Error in Processing" to "Initial." A Process Transaction Service picks up the data and reprocesses it into the Siebel application.

NOTE: You should not modify queue entries with the status Initial, Send, or Processed while the server is running. These entries may be in use by the background processes in the server. You may edit entries with Error in Processing or Error in Sending status values while the server is running.

b Delete the entry in the EAI Queue for that transaction and recreate it from SAP using whatever correction may have been made.

NOTE: All components which may have services accessing the queue must execute on the same server machine. These components use a locking mechanism around the EAI Queue that operates correctly only when all processes execute on the same machine. These components are: BusIntMgr, SAPSendTrans, BAPIRcvr, and SAPProcessTrans.

The Send Transaction Service

The Send Transaction service is called repeatedly from the SAPSendTrans component running as a background task to check the EAI Queue for data that needs to be sent to SAP. This topic covers configuration options for this service and describes how to start the SAPSendTrans component.

Configuring the Send Transaction Service

The business service EAI SAP Send Transaction is called by the background component SAPSendTrans repeatedly to check erred transactions in the EAI Queue. When a transaction is found, this business service calls the tRFC BAPI adapter service that created the queue entry to again try to send the data to SAP. IDOCs can be resent to SAP in this manner also. For more information on IDOCs sent through the `EAI SAP BAPI adapter (tRFC)` service, read "IDOC Integration" on page 129.

Table 38 summarizes the User Properties that may be set to control its behavior. Those variables that may be set as component parameters must be set for the SAPSendTrans component.

Table 38. Send Transaction Service Configuration Options

Name	User Prop	Comp Param	Valid Values	Usage
SAPWaitTime	X		Integer value in seconds	Number of seconds to pause between sending transactions to SAP. This may be used to slow down the flow of information to SAP. Too many RFC calls to SAP or IDOCs that are set to process immediately can overwhelm SAP.
SAPWakeupCount	X		Integer value	Number of times to try to reconnect to SAP before erring the transaction, if SAP is unreachable.
SAPWakeupTime	X		Integer value in minutes	Number of minutes to wait between tries to connect to SAP, if SAP is unreachable. This is used in conjunction with SAPWakeupCount. If SAPWakeupCount is 10 and SAPWakeupTime is 5, then the Resend service tries every 5 minutes to reach SAP for up to 10 attempts before marking the transaction as erred, for a total of 50 minutes.
SAPtRFCService	X	X		Business service that originally created the entry in the queue. It is important to use the same service because the user properties in this service are used by the Send service.

Table 38. Send Transaction Service Configuration Options

Name	User Prop	Comp Param	Valid Values	Usage
SiebelWaitTime	X		Integer value in seconds	Number of seconds to wait before rechecking the Siebel EAI Queue when the queue is found empty. If an entry was last found in the queue, then no wait occurs when looking for the next entry in the queue.
SAPRfcConnectString		X		Refer to Table 21 on page 111.
SAPRfcUserName		X		Refer to Table 21 on page 111.
SAPRfcPassword		X		Refer to Table 21 on page 111.

Starting the Send Transaction Service

The Send Transaction service can be started in the same manner as the tRFC BAPI receiver component. Read "Starting the tRFC BAPI Receiver" on page 39. You must set the values of the component parameters SAPRfcConnectString, SAPRfcUserName, and SAPRfcPassword the same as the original tRFC BAPI adapter. These parameters were set on the Business Integration Manager component.

The Process Transaction Service

The Process Transaction Service is called repeatedly from the SAPProcessTrans component, running as a background task to check the EAI Queue for data that needs to be processed into the Siebel application. This topic covers configuration options for this service and describes how to start the SAPProcessTrans component.

Configuring the Process Transaction Service

The business service EAI SAP Process Transaction is called by the background Component SAPProcessTrans repeatedly to check erred transactions in the EAI Queue. When a transaction is found, this business service invokes the workflow needed to try again to process the data into the Siebel application. IDOCs can also be reprocessed into the Siebel application in this manner also. For more information on IDOCs sent through the `EAI SAP BAPI receiver (tRFC)` service, read "IDOC Integration" on page 129.

Table 39 summarizes the user properties and component parameters that may be set to control its behavior. Those variables that may be set as component parameters must be set for the SAPProcessTrans component.

Table 39. Process Transaction Service Configuration Options

Name	User Prop	Comp Param	Valid Values	Usage
SAPBAPIDispatch Service	X	X		Should be set to the name of the same BAPI Dispatch Service used by the tRFC BAPI Receiver that created the queue entry.
SAPIDOCDispatch Service	X	X		Should be set to the name of the same IDOC Dispatch Service used by the tRFC BAPI Receiver that created the queue entry.
SAPtRFCService	X	X		Business Service which originally created the entry in the queue. It is important to use the same service as the user properties in this service are used by the Reprocess Service.
SiebelWaitTime	X		Integer value in seconds	Number of seconds to wait before rechecking the Siebel EAI Queue when the queue is found empty. If an entry was last found in the queue, then no wait occurs when looking for the next entry in the queue.

Starting the Process Transaction Service

The Process Transaction Service can be started in the same manner as the tRFC BAPI receiver component. Read "Starting the tRFC BAPI Receiver" on page 39. There are no required component parameters that must be set for this component.

EAI Queue Business Service

The EAI Queue business service can be used to access the EAI Queue. This section describes the user properties and methods for this service. The business service is called "EAI XML Queuing Service" in Siebel Tools. User properties for the EAI XML Queuing service are:

- **TempDirectory.** Directory for temporary storage, default of c:\temp. Depending upon your configuration you may need to change this value. In Windows 2000 a temp directory is under the WINNT folder. TempDirectory is not used within the Siebel server environment. This variable is used only when running within the Siebel Mobile Web Client.
- **FileExtension.** The file extension to be used for files created by the EAI XML Queuing service. The default value is `txt`.

The business service methods are:

- "AddMessage"

AddMessage

The AddMessage method creates an XML file with the given Siebel Message and inserts a status row into the status table. Table 40 summarizes the AddMessage arguments.

Table 40. AddMessage Arguments

Argument Name	Input/ Output	Required ?	Argument Type	Description
SiebelMessage	Input	No	Hierarchy	Input transaction data in form of Siebel Message. This is written to an XML file.
ReferenceID	Input	Yes	String	Reference ID for status table.
QueueName	Input	Yes	String	Name of queue to insert the status row into.
Status	Input	Yes	String	Status value for status table.
Comments	Input	No	String	Comments for status table.
RowID	Output		String	Row ID of new entry in queue items table.
ReferenceValue2	Input	No	String	Additional information.
ReferenceValue3	Input	No	String	Additional information.
SequenceID	Output		String	Sequence ID of new entry in queue items table.

DeleteMessage

The DeleteMessage method either deletes the status record or updates its status as Confirmed. Table 41 summarizes the DeleteMessage arguments.

Table 41. DeleteMessage Arguments

Argument Name	Input/ Output	Required ?	Argument Type	Description
RowID	Input	No (see following comment)	String	Row ID for Queue Items Table.
DeleteStatusRecord	Input	No	String	Value of True or False. False is default.
QueueName	Input	Yes	String	Type for status table.
Status	Input	No	String	Status value for status table - used if not deleting status record.
Comments	Input	No	String	Comments for status table - used if not deleting status record.
SearchSpec	Input	No (see following comment)	String	Search criteria.
SortSpec	Input	No	String	Sort criteria.
NumberRowsAffected	Output		String	Number of rows deleted/updated.

Comment. Either RowID or SearchSpec is required.

- If Row ID is provided, it is the only search criteria used and a single row is deleted.
- If a search spec is provided, all rows matching the search criteria are deleted.
- If a sort spec is provided, the first row retrieved using the sort spec is deleted, and other rows are not.
- SearchSpec is modified to be "[Queue Name] = '*Queue name given*' AND (*SearchSpec given*)".

GetMessage

GetMessage retrieves the XML file data and outputs a Siebel Message. It also retrieves the current status information. Table 42 summarizes the GetMessage arguments.

Table 42. GetMessage Arguments

Argument Name	Input/ Output	Required ?	Argument Type	Description
RowID	Input/ output	No (see following comment)	String	Row ID for Queue Items Table.
QueueName	Input	Yes	String	Queue name.
SearchSpec	Input	No (see following comment)	String	Search criteria.
SortSpec	Input	No	String	Sort criteria.
SiebelMessage	Output		Hierarchy	Transaction data from XML file in the form of a Siebel Message.
ReferenceID	Output		String	Reference ID.
SequenceID	Output		String	Sequence number.
Status	Output		String	Status value for status table.
ReferenceValue2	Output		String	Additional information.
ReferenceValue3	Output		String	Additional information.
Comments	Output		String	Comments for status table.

Comment. Either RowID or SearchSpec is required.

- If Row ID is provided, it is the only search criteria used and that row is retrieved.
- If both a search spec and sort spec are provided, the first record selected using this information is returned.
- If a search spec is provided and no sort spec is provided, then the records are sorted based on Sequence number and the oldest entry in the queue that matches the search spec is returned.
- SearchSpec is modified to be "[Queue Name] = '*Queue name given*' AND (*SearchSpec given*)".

GetStatus

GetStatus retrieves the current status record from a given Row ID or Search Spec. Table 43 summarizes the GetStatus arguments.

Table 43. GetStatus Arguments

Argument Name	Input/ Output	Required?	Argument Type	Description
RowID	Input/ output	No (see following comment)	String	Row ID for Queue Items Table.
QueueName	Input	Yes	String	Type for status table.
SearchSpec	Input	No (see following comment)	String	Search criteria.
SortSpec	Input	No	String	Sort criteria.
ReferenceID	Output		String	Reference ID.
SequenceID	Output		String	Sequence number.
Status	Output		String	Status value for status table.
ReferenceValue2	Output		String	Additional information.
ReferenceValue3	Output		String	Additional information.
Comments	Output		String	Comments for status table.

Comment. One of either RowID or SearchSpec is required.

- If Row ID is provided, it is the only search criteria used and that row is retrieved.
- If both a search spec and sort spec are provided, the first record selected using this information is returned.
- If a search spec is provided and no sort spec is provided, then the records are sorted based on Sequence number and the oldest entry in the queue that matches the search spec is returned.
- SearchSpec is modified to be "[Queue Name] = '*Queue name given*' AND (*SearchSpec given*)".

UpdateStatus

UpdateStatus updates the fields in the status record for the given Row ID or SearchSpec. Table 44 summarizes the UpdateStatus arguments.

Table 44. UpdateStatus Arguments

Argument Name	Input/ Output	Required?	Argument Type	Description
RowID	Input/ output	No (see following comment)	String	Row ID of record to update.
QueueName	Input	Yes	String	Name of queue to insert into.
Status	Input	Yes	String	Status value for status table.
Comments	Input	No	String	Comments for status table.
SearchSpec	Input	No (see following comment)	String	Search criteria.
SortSpec	Input	No	String	Sort criteria.
ReferenceID	Output		String	Reference ID.
SequenceID	Output		String	Sequence number.
ReferenceValue2	Output		String	Additional information.
ReferenceValue3	Output		String	Additional information.
NumberRowsAffected	Output		String	Number of rows updated.

Comment. Either RowID or SearchSpec is required.

- If Row ID is provided, it is the only search criteria used and a single row is updated.
- If a search spec is provided, all rows matching the search criteria are updated.
- If a sort spec is provided, the first row retrieved using the sort spec is updated, other rows are not.
- SearchSpec is modified to be "[Queue Name] = '*Queue name given*' AND (*SearchSpec given*)".

Reference ID and Sequence ID are output only if it is guaranteed that only one row is returned, for example, if the Row ID is input, or both the Sort and the Search specs are provided.

9 Upgrading from v6.x to v7.x

Siebel Connector for SAP R/3 supports two sets of data flows, v6.x and v7.x. The 6.x data flows were originally included with 6.x versions of the Siebel Connector. These data flows have been modified slightly so that they can be used within Siebel 7.

About Upgrading the Siebel Connector for SAP R/3

If you have been using version 6.x of the Siebel Connector for SAP R/3 and have data flows that may be based upon the 6.x data flows, you may need to make the same modifications to use your flows in Siebel 7. The following modifications have been made to 6.x data flows in Siebel 7.

- The AllLangIndependentVals = "Y" user property has been added to the Order - Create SAP Order (Siebel) integration object. This is now required for the Siebel adapter to pick LIVs (Language Independent Values) when the database is non-ENU.
- The Order Item XA business object component has been added to the Order Entry - Get SAP Order Status business object. This is because in Siebel 7 the calculation of the quantity of items has been moved to a separate business component and this business component must be included in the business object on which the integration object is based.
- Picklists and Pickmaps were added to the Delivery Block, Delivery Status, and Pricing Status fields in the Order Entry - Get SAP Order Status Header business component and the Delivery Status field in the Order Entry - Get SAP Order Status Item business component. The PICKMAP user property was set to Y for the four corresponding fields in Order - Get SAP Order Status (Siebel) integration object.

You can also upgrade your 6.x business data flows to work with the new Siebel 7.x infrastructure. The following sections cover the changes you need to make. While this was not required for 7.0, these changes are required for version 7.5.

Integration Objects

The structure of IDOC integration objects are different in Siebel 6.x and Siebel 7.x. The following section explains how to upgrade a Siebel 6.x IDOC integration object.

To upgrade IDOC integration objects

1 Lock the project containing your IDOC integration object.

2 Temporarily rename your current IDOC integration object.

   For example, if you have an IDOC integration object based on DEBMAS02 with the name Customer IDOC Integration Object, change the name of your integration object to something else, such as Customer IDOC Integration Object (old).

3 Using the EAI SAP IDOC Adapter Wizard, create a new IDOC integration object with the original name of your old IDOC integration object.

   Continuing with the example in Step 2, capture a DEBMAS02 IDOC from SAP using the new wizard and name this Customer IDOC Integration Object. This new integration object will contain control and data record fields as well as the user properties ALEVersion and Workflow that do not exist in 6.x integration objects.

4 If you are sending IDOC Integration Objects to SAP, make sure that the setting of the ALEVersion user property in your new integration object is consistent with the ALE interface you want to use in SAP.

   If you are using SAP Version 3.1, ALEVersion must be set to 3X. If you are using SAP Version 4.0 or higher you may use 3X or 4X, although 4X is preferred.

5 If you normally receive this IDOC from SAP, enter in the Workflow user property of the IDOC integration object the name of the workflow process that you would like to be invoked when this IDOC is received by the Siebel application.

6 If you intend to receive the IDOC from SAP, add the name of the integration object to a user property of the type SAPIdocAllowedObjectnn in the business service definition of the EAI SAP IDOC Workflow Processor.

   This registers your integration object in the Siebel application.

   a In Siebel Tools, lock the SAP Business Services project.

   b Navigate to the Business Service definition for the EAI SAP IDOC Workflow Processor.

   c Select User Properties for this business service.

   d Add a new User Property record with the name SAPIdocAllowedObjectnn, where nn is a number sequentially one more than the highest existing number (for example, if the highest number is 05, then add SAPIdocAllowedObjects06).

   e Set the value of the user property to the name of the integration object you are using.

7 Recompile the new IDOC integration object and the SAP Business Services project into your .srf file.

8 Copy the .srf file as needed for use by the server and client applications.

Workflows

If you are sending IDOCs to SAP, the service that you invoke to send the IDOC to SAP is different. The service for 6.x was EAI SAP IDOC RFC Adapter; the service for Siebel 7 is `EAI SAP IDOC Adapter`.

If you are receiving IDOCs from SAP, you do not need to modify your workflow.

To switch to the new service safely

1. Revise your workflow using Siebel Tools.

   For more information on modifying workflows in Tools, see *Siebel Business Process Designer Administration Guide*.

   NOTE: You may want to make a backup copy of your workflow before making the changes.

2. Determine which business service box on your workflow corresponds to the EAI SAP IDOC RFC Adapter (the 6.x business service).

   You will be creating a new business service to replace this one.

3. View the input arguments of your existing business service.
4. Make note of how these are used because you will need to add them to the new business service.
5. Create a new business service box on your diagram in the vicinity of the old business service.
6. View the input arguments for this business service.
7. Define the input and output method arguments in the new service as they were defined in your old service. These have not changed.
8. Modify the flow of the workflow diagram to use the new service you have created.
9. Test your workflow using the workflow simulator. Follow the suggestions for testing in "Testing the Interface" on page 103.

Business Service Data Maps

If your business service data maps currently use the MessageBox function they need to be modified. The MessageBox function is not available in Siebel 7.

10 Modifying Siebel Interfaces After an SAP Upgrade

Siebel interfaces to SAP R/3 can be affected by an SAP upgrade (or even an SAP patch installation). For this reason it is important to thoroughly review your Siebel interfaces in the event of an upgrade and to retest your interfaces after an SAP patch has been applied. This chapter describes some of the changes that might invalidate your Siebel implementation when the R/3 version is changed.

SAP IDOC Interfaces

SAP IDOCs are version dependent. For instance, a single IDOC type will change structure from one R/3 version to another: New segments may be added, fields may be added to the segments, and application processing of the IDOC within SAP may change.

Generally, you can use an older IDOC type structure within a new SAP R/3 version. The R/3 version of an IDOC to be sent from SAP to the Siebel application is configurable within the partner profile for the logical system being used.

To set the version of the IDOC to be sent from SAP

1. Navigate within SAP to transaction we20.
2. Select the logical system/partner profile in use.
3. Select the outbound parameters display for this logical system.
4. Set the Seg.Release in IDOC type field to your previous R/3 IDOC version.

For example, if you are using the DEBMAS02 IDOC type, running SAP R/3 version 4.5B, and upgrading to R/3 version 4.6B; set the Seg.Release in IDOC type field to 45B. SAP then sends the IDOC type structure of version 4.5B as opposed to 4.6B. If the field is not filled in, the IDOC type structure corresponding to the current version is sent.

NOTE: This version is only applicable to SAP developed IDOC type structures. It is not applicable to custom IDOCs if no version change has been made.

When sending IDOCs from the Siebel application to SAP, no version setting is necessary. SAP determines the version of the IDOC when it is sent.

IDOC type structures that change from version to version are changed in minimal ways. Fields and segments may be added to the structure; no fields are deleted nor does the order of preexisting fields change. However, SAP also provides application software that populates the IDOC fields on output from SAP and extracts field values on input to SAP. This application software may change during an upgrade of SAP. For this reason, test all IDOC interfaces during the upgrade process.

SAP BAPI/RFC Interfaces

BAPI/RFC interfaces may also change from one SAP version to another. In particular, BAPI or RFC functions that are marked as "Not released" are the most subject to changes by SAP from version to version. Functions that are "Not released" may not function properly when moving from one version of R/3 to another. Therefore, any functions in use that are marked "Not released" should receive special scrutiny when performing an SAP upgrade. These functions may be untested in the newer SAP version and should be used only if absolutely necessary.

To determine the status of a BAPI or RFC

1. Navigate to SAP transaction SE37.
2. Enter the name of the RFC or BAPI call in use.
3. Select the Display button.
4. Select the Attributes tab.

   The status appears at the bottom of the General Data area. If Released, a release date also appears.

BAPIs and RFCs that have been released may also change slightly during an SAP upgrade. However, because of their release status, they are tested and supported by SAP and should remain functional from one version to another, unless SAP indicates otherwise.

To check for more information on the status of released BAPIs

1. Navigate to the SAP transaction `bapi`.
2. Select the BAPI you are interested in from the left panel.
3. Check the release information in the Status area of the Detail tab.

   This may supply more specific information on release status. For example, BAPI_SALESORDER_CREATEFROMDAT1 shows a released status, but also shows the warning, "Caution: Method is obsolete as of Release 46C!"

Check the following for all BAPIs and RFCs during an SAP upgrade:

- Check for any OSS notes that might indicate a change in functionality or defects.
- Recheck the release status as mentioned in the preceding procedure.
- Check any existing function documentation, accessed through the Function Module Documentation button in SAP transaction SE37.
- Extract the new version of the BAPI into new integration objects (input and output) in Siebel Tools and use the Compare Objects function to detect any differences in structure between the new and old integration objects.

  If the structure is different, the new structure should always be used. If the structure is significantly different, it may be necessary to review and modify eScript maps that reference fields in the structures.

- Always retest, even when there are no structure changes, as the function itself may have a slightly different behavior than in the previous R/3 system.

Be aware that patch installations in SAP may also affect the functionality of BAPIs and RFCs. Patches should be reviewed for content and some testing may be required after the application of a patch.

SAP DLL Library

Both BAPI/RFC and IDOC interfaces make use of the SAP DLL, librfc32.dll. The Siebel application currently tests with version 46C of this library. SAP guarantees that all newer versions of the SAP library will contain all functionality provided by version 46C and hence should be compatible with Siebel Connector for SAP R/3. However, should an issue arise that may be related to the DLL version, Siebel Technical Support may request that you use the 46C DLL temporarily to eliminate any possible problems that may arise in future versions of the DLL.

For this reason, when upgrading from one SAP version to another, it is not necessary to upgrade the librfc32.dll used by the Siebel Connector for SAP R/3 components (Siebel Tools, Siebel Dedicated Client, Siebel Server Components) to a version higher than 46C. To do so may subject the installation to additional issues that may exist in newer versions of SAP's library. Any issues that are shown to be related to a newer version of this library will need to be referred to SAP support.

A Data Types

This appendix covers data types used in BAPIs and IDOCs.

Data Fields

Each data field in a BAPI or IDOC Integration Object contains information on its external data type and its Siebel data type. The following information is kept for each field:

- Data Type - This is the Siebel Data type. It can have the following values:
  - DTYPE_TEXT - text string
  - DTYPE_DATETIME - date and time string
  - DTYPE_NUMBER - numeric value
- Length - Siebel Data type length of a DTYPE_TEXT field
- Precision and Scale - Siebel data type precision and scale values
- External Data Type - SAP data type
- External Length - SAP length in bytes
- External Scale - number of decimal places

The SAP data types CHAR, LCHR, LANG, CUKY, UNIT, CLNT and VARC have a Siebel type of DTYPE_TEXT. The Length and External Length columns are populated with the length of the field in SAP. These field types are used both in BAPI and IDOC Integration Objects. When a field of these types is being sent to SAP, the data in the field is left-justified and blank filled. When a field of these types is received from SAP, trailing blanks on the field are truncated, except in the case of the LCHR field. The LCHR field is passed exactly as it is received from SAP.

The SAP data types DATS and TIMS have a Siebel type of DTYPE_DATETIME. The Siebel type is a 19 character string that includes both date and time, in MM/DD/YYYY HH:MM:SS format. DATS is the 8 character SAP date in YYYYMMDD format. TIMS is the 6 character SAP time in HHMMSS format.

When a DATS field is sent to SAP, the SAP adapters extract the date portion of the Siebel DTYPE_DATETIME field and construct an 8 character DATS field in the correct format. When a TIMS field is sent to SAP, the SAP adapters extract the time portion of the Siebel DTYPE_DATETIME field and construct a 6 character TIMS field in the correct format.

When a DATS field is received from SAP, a Siebel DTYPE_DATETIME field is constructed with a time of 00:00:00. When a TIMS field is received from SAP, a Siebel DTYPE_DATETIME field is constructed with a date of 01/01/1980. Example, if a DATS field for November 7, 2000 is received from SAP, it looks like this: 20001107. When converted to a Siebel DTYPE_DATETIME field, it looks like this: 11/07/2000 00:00:00.

The SAP data types DEC, CURR, and QUAN are packed decimal fields in SAP. These have a Siebel type of DTYPE_NUMBER. Both precision and scale values are kept for these fields. Precision is the character length of the number and scale is the number of decimal places after the decimal point. In some cases, this information can be incorrect in the SAP data dictionary and may need to be corrected in the integration object after it is captured from SAP. These types are supported for BAPI and RFC calls only. All data contained within IDOCs is in character form when passed to or from SAP.

The SAP data types INT1, INT2, INT4, and PREC are binary integer fields in SAP. These have a Siebel type of DTYPE_NUMBER. These types are supported for BAPI and RFC calls only. All data contained within IDOCs is in character form when passed to or from SAP.

The SAP data types ACCP and NUMC are numeric character strings in SAP. These have a Siebel data type of DTYPE_NUMBER. While a NUMC field does not normally have a scale value, as it generally represents an integer, a scale value can be entered for the field in the Siebel Integration Object. In this event, a decimal point is created at the appropriate place. This feature supports BAPI interfaces that send integer values that are intended to have fixed decimal places. For example, if the number "00054321" is sent as a NUMC field and the comment for the field indicates that the number should be interpreted as 543.21, the Siebel SAP adapters have the capability of inserting the decimal point at the correct position, if the external scale value for the field is set to 2. When the number is sent to SAP, the decimal point is removed. NUMC and ACCP fields are also padded on the left with zeros prior to sending to SAP. Leading zeros are removed from NUMC and ACCP fields when the value is received from SAP. These data types are used for both BAPI and IDOC interfaces. However, only BAPI interfaces use the External Scale field to shift the decimal point.

The SAP data type FLTP represents an 8 byte floating point number. This has a Siebel data type of DTYPE_NUMBER. The 8 byte binary floating point number passed from SAP is converted to a character string DTYPE_NUMBER field. The DTYPE_NUMBER field is converted to an 8 byte binary floating point number and passed to SAP. Exponential notation is not supported. The number must be representable as a 16 character field without exponential notation.

IDOC Integration Object fields that represent decimal numbers are treated specially by the adapters. All IDOC data is passed as character strings. The format for a decimal value passed as a character string in an IDOC field is to left-justify the number and place any "-" sign after the number. For example, the value "–123.45" would be placed into the IDOC field as "123.45– ". To handle this situation some special external data types are used in the IDOC Integration Object. These are IDOC-DEC, IDOC-CURR, IDOC-QUAN, and IDOC-FLTP. When an IDOC is sent or received, to or from SAP, the number is converted to or from a DTYPE_NUMBER field. The "-" sign is moved to the front of the number in the DTYPE_NUMBER field.

Similarly IDOC Integration Object fields that represent integer numbers are treated specially by the adapters. The types of IDOC-INT1, IDOC-INT2, IDOC-INT4 and IDOC-PREC are treated as if they were NUMC fields with the exception that no scale information is used.

When an IDOC Integration Object field does not require a value, you may enter the SAP no-data character, usually a forward slash (/). This indicates to SAP that no data is being passed for this field. The interpretation of this is up to the SAP application processing the data, but usually it is interpreted as an initial value for data that does not exist or no change to previously existing data.

Siebel Applications display SAP material numbers, customer numbers, and sales order numbers with leading zeros. These numbers are stored in this manner in the SAP database, although the leading zeroes are not displayed in the SAP user interface.

B SAP Field Mappings

This appendix contains instructions on assessing the field mapping tables used by the business service data maps. For more information, see "Modifying Standard Integration Interfaces" on page 72 in Chapter 5, "Customizing Integrations."

Locating the SAP Field Mappings Tables

Due to the size of the field mapping tables, the SAP field mapping tables are located on the *Siebel Bookshelf*.

To locate the SAP Field Mappings Tables

1. From the *Siebel Bookshelf*, in the Documentation by Product Line section, click eBusiness Applications.
2. Click the Application Integration category.
3. Navigate to the *Connector for SAP R/3 Guide* in the Application Integration category.
4. To access the SAP field mappings table that you are interested in, choose from the following listings:
   - For account-customer mappings, select one of the following maps:
     - SAP R/3-Maps-Account (Siebel to SAP)
     - SAP R/3-Maps-Account (SAP to Siebel)
   - For order mappings, click SAP R/3-Maps-Order
   - For product mappings, click SAP R/3-Maps-Product.
   - For quote validation mappings, click SAP R/3-Maps-QuoteValidate.

C SAP Code Page Mappings

This appendix provides specific information on code page mappings used by the Siebel Connector for SAP R/3. The connector directly maps SAP's ISO codepages to Microsoft codepages; however it is possible for the ISO code page to differ from the Microsoft code page. The ISO code page can differ from the Microsoft code page if:

- There are characters in the Microsoft code page that are not defined in the ISO code page, see "Mapping Microsoft Characters Not Available in ISO."
- There are characters in the ISO code page that are not defined in the Microsoft code page, see "Mapping ISO Characters Not Available in Microsoft.".
- There are characters that are defined in both but are not the same, see "Mapping SAP Characters that Differ Between Codepages.".

The behavior of the SAP connector in each of these cases is described in the following sections. The last section includes tables of commonly used codepages and their differences.

Mapping Microsoft Characters Not Available in ISO

In certain mappings, characters exist in the Microsoft code page that do not exist in its corresponding ISO code page. In the event that a codepoint of this type exists in the Siebel database and is passed to SAP through the SAP connector, the value of the codepoint is passed to the SAP database unchanged. For instance, in the mapping of Microsoft 1252 to ISO 8859-1, the codepoint 80 (the Euro character) is unchanged and is stored in the SAP database. However, SAP may not display the character. The SAP interface may display a # for the character as the codepoint 80 is not defined in the ISO code page used by SAP.

In the event that a codepoint of this type exists in the SAP database and is passed to the Siebel application through the SAP connector, the value of the codepoint remains unchanged and is correctly displayed by the Siebel application. In the preceding example, if the 80 codepoint in the SAP database is passed back to the Siebel application, the application receives it and displays it as the Euro character.

If these characters originate in the Siebel application and are not modified in SAP, they transfer to and from SAP without error and correctly display in the Siebel application.

Table 45 contains a list of codepoints in the Microsoft codepages that do not exist in the ISO codepages for all mappings in Table 11 on page 36.

Table 45. ISO 8859-8 and Microsoft 1255 (Hebrew) Codepoints

Codepoint in Microsoft 1255 (Hex)	Unicode value (Hex)	Character representation
BF	00BF	Inverted question mark
C0	05B0	Hebrew Point Sheva
C1	05B1	Hebrew Point Hataf Segol
C2	05B2	Hebrew Point Hataf Patah
C3	05B3	Hebrew Point Hataf Qamats
C4	05B4	Hebrew Point Hiriq
C5	05B5	Hebrew Point Tsere
C6	05B6	Hebrew Point Segol
C7	05B7	Hebrew Point Patah
C8	05B8	Hebrew Point Qamats
C9	05B9	Hebrew Point Holam
CB	05BB	Hebrew Point Qubuts
CC	05BC	Hebrew Point Dagesh or Mapiq
CD	05BD	Hebrew Point Meteg
CE	05BE	Hebrew Punctuation Maqaf
CF	05BF	Hebrew Point Rafe
D0	05C0	Hebrew Punctuation Paseq
D1	05C1	Hebrew Point Shin Dot
D2	05C2	Hebrew Point Sin Dot
D3	05C3	Hebrew Punctuation Sof Pasuq
D4	05F0	Hebrew Ligature Yiddish Double Vav
D5	05F1	Hebrew Ligature Yiddish Vav Yod
D6	05F2	Hebrew Ligature Yiddish Double Yod
D7	05F3	Hebrew Punctuation Geresh
D8	05F4	Hebrew Punctuation Gershayim
FD	200E	Left-to-right mark
FE	200F	Right-to-left mark

Table 45. ISO 8859-8 and Microsoft 1255 (Hebrew) Codepoints

Codepoint in Microsoft 1255 (Hex)	Unicode value (Hex)	Character representation
80	20AC	Euro sign
82	201A	Single low-9 quotation mark
83	0192	Latin small letter F with hook
84	201E	Double low-9 quotation mark
85	2026	Horizontal ellipsis
86	2020	Dagger
87	2021	Double dagger
88	02C6	Modifier letter circumflex accent
89	2030	Per mille sign
8B	2039	Single left-pointing angle quotation mark
91	2018	Left single quotation mark
92	2019	Right single quotation mark
93	201C	Left double quotation mark
94	201D	Right double quotation mark
95	2022	Bullet
96	2013	En dash
97	2014	Em dash
98	02DC	Small tilde
99	2122	Trademark sign
9B	203A	Single right-pointing angle quotation mark

Table 46. ISO 8859-7 and Microsoft 1253 (Greek) Codepoints

Codepoint in Microsoft 1253 (hex)	Unicode value (hex)	Character representation
80	20AC	Euro sign
82	201A	Single low-9 quotation mark
83	0192	Latin small letter F with hook
84	201E	Double low-9 quotation mark
85	2026	Horizontal ellipsis
86	2020	Dagger
87	2021	Double dagger
89	2030	Per mille sign
8B	2039	Single left-pointing angle quotation mark
91	2018	Left single quotation mark
92	2019	Right single quotation mark
93	201C	Left double quotation mark
94	201D	Right double quotation mark
95	2022	Bullet
96	2013	En dash
97	2014	Em dash
99	2122	Trademark sign
9B	203A	Single right-pointing angle quotation mark
A4	00A4	Currency sign
A5	00A5	Yen sign
AE	00AE	Registered sign

Table 47. ISO 8859-9 and Microsoft 1254 (Latin-2) Codepoints

Codepoint in Microsoft 1254 (Hex)	Unicode value (Hex)	Character representation
80	20AC	Euro sign
82	201A	Single low-9 quotation mark
83	0192	Latin small letter F with hook

Table 47. ISO 8859-9 and Microsoft 1254 (Latin-2) Codepoints

Codepoint in Microsoft 1254 (Hex)	Unicode value (Hex)	Character representation
84	201E	Double low-9 quotation mark
85	2026	Horizontal ellipsis
86	2020	Dagger
87	2021	Double dagger
88	02C6	Modifier letter circumflex accent
89	2030	Per mille sign
8A	0160	Latin capital letter S with caron
8B	2039	Single left-pointing angle quotation mark
8C	0152	Latin capital ligature oe
91	2018	Left single quotation mark
92	2019	Right single quotation mark
93	201C	Left double quotation mark
94	201D	Right double quotation mark
95	2022	Bullet
96	2013	En dash
97	2014	Em dash
98	02DC	Small tilde
99	2122	Trademark sign
9A	0161	Latin small letter s with caron
9B	203A	Single right-pointing angle quotation mark
9C	0153	Latin small ligature oe
9F	0178	Latin capital letter Y with diaeresis

Table 48. ISO 8859-1 and Microsoft 1252 (Latin-1) Codepoints

Codepoint in Microsoft 1252 (Hex)	Unicode value (Hex)	Character representation
80	20AC	Euro sign
82	201A	Single low-9 quotation mark

Table 48. ISO 8859-1 and Microsoft 1252 (Latin-1) Codepoints

Codepoint in Microsoft 1252 (Hex)	Unicode value (Hex)	Character representation
83	0192	Latin small letter F with hook
84	201E	Double low-9 quotation mark
85	2026	Horizontal ellipsis
86	2020	Dagger
87	2021	Double dagger
88	02C6	Modifier letter circumflex accent
89	2030	Per mille sign
8A	0160	Latin capital letter S with caron
8B	2039	Single left-pointing angle quotation mark
8C	0152	Latin capital ligature OE
8E	017D	Latin capital letter Z with caron
91	2018	Left single quotation mark
92	2019	Right single quotation mark
93	201C	Left double quotation mark
94	201D	Right double quotation mark
95	2022	Bullet
96	2013	En dash
97	2014	Em dash
98	02DC	Small tilde
99	2122	Trademark sign
9A	0161	Latin small letter s with caron
9B	203A	Single right-pointing angle quotation mark
9C	0153	Latin small ligature oe
9E	017E	Latin small letter z with caron
9F	0178	Latin capital letter Y with diaeresis

Mapping ISO Characters Not Available in Microsoft

A case where characters exist in the ISO code page, but not the Microsoft code page, is extremely rare. The only instance of this in currently mapped codepages (Table 11 on page 36), is the mapping between Microsoft 1255 and ISO 8859-8 (Hebrew). In this case, ISO 8859-8 contains the codepoint DF (Double low line). The character corresponds to Unicode 2017. This character does not exist in the Microsoft 1255 code page.

When this character is stored in the SAP database and is passed through the SAP connector from SAP to the Siebel application, an error is generated because the DF codepoint value is not defined in the Microsoft 1255 code page.

When this character (Unicode 2017) exists in the Siebel database and is passed to SAP through the SAP connector, an error is also generated. This is because no value exists in the Microsoft 1255 code page for the Unicode character.

Mapping SAP Characters that Differ Between Codepages

In the following mappings, codepoints with the same binary values in both Microsoft and ISO codepages have different character representations.

SAP code page 1700 - ISO 8859-7 maps to Microsoft 1253 (Greek).

SAP code page 1800 - ISO 8859-8 maps to Microsoft 1255 (Hebrew).

In these codepages, a single binary codepoint value can represent different characters in each code page. A specific list of these codepoints is provided later in this section in Table 49 and Table 50 on page 178. These codepages have been listed in Table 11 on page 36 as "not a complete match." In this event data may be corrupted if it is modified in both the SAP and Siebel applications.

When a codepoint of this type exists in the SAP database and is transferred to the Siebel application, the binary value of the codepoint is converted to Unicode based on the Microsoft code page mapping. For example, the codepoint B6 in the ISO 8859-7 code page is a Greek capital letter alpha with tonos. The codepoint B6 in the Microsoft 1253 code page is a pilcrow sign. If the B6 value exists in the SAP database, SAP displays it as the Greek alpha character. When this value is passed to the Siebel application, the application interprets it with Microsoft code page 1253 and assumes that it is a pilcrow sign. The Siebel application then converts the value to a Unicode codepoint for the pilcrow sign and displays the character as a pilcrow sign.

If the value remains unchanged in the Siebel application and is then sent back to SAP, SAP converts the character from Unicode to the binary B6 value. SAP interprets the B6 using the ISO code page and correctly displays the Greek alpha character once again.

A similar scenario holds true if the original value is entered into the Siebel application. If the character is originally entered in the Siebel application as a pilcrow, this value is converted, transported to SAP, and appears on an SAP display as an alpha character. If the value is unchanged and is sent back to the Siebel application at a later time, the character still appears as a pilcrow character.

If either the SAP or the Siebel application is defined as the system of origin for the data, then no data corruption occurs; however, the data displays differently in each system (correct in the system of origin and incorrect in the other system). Data corruption is possible if data is changed in both the SAP and the Siebel applications because displayed values are different in each system for these characters.

Table 49 and Table 50 list codepoints that differ for SAP codepages 1700 and 1800.

Table 49. Mapping SAP code page 1700 - ISO 8859-7 to Microsoft 1253 (Greek)

Codepoint value (Hex)	Unicode value for ISO	Character representation for ISO	Unicode value for Microsoft	Character representation for Microsoft
A1	02BD	Modifier letter reversed comma	0385	Greek dialytika tonos
A2	02BC	Modifier letter apostrophe	0386	Greek capital letter alpha with tonos
B5	0385	Greek dialytika tonos	00B5	Micro sign
B6	0386	Greek capital letter alpha with tonos	00B6	Pilcrow sign

Table 50. Mapping SAP code page 1800 - ISO 8859-8 to Microsoft 1255 (Hebrew)

Codepoint value (Hex)	Unicode value in ISO (Hex)	Character representation in ISO	Unicode value in Microsoft (Hex)	Character representation in Microsoft
A4	00A4	currency sign	20AA	New Sheqel sign
AF	203E	overline	00AF	Macron

D Troubleshooting Siebel Connector for SAP R/3

The troubleshooting information is organized in the following categories:

Setting SAP Debugging Options

You can set debugging options in the saprfc.ini file to aid in tracing the data from the Siebel application to SAP. In the TYPE=A or TYPE=R entries you may add RFC_TRACE=1. This causes SAP's underlying RFC library to write debug files with names of the form "rfc*.trc" to be written to the bin directory. These files contain the raw data sent between the Siebel application and SAP.

Also, for a TYPE=A entry you may add ABAP_DEBUG=1. This causes the ABAP debugger to appear when the Siebel application makes an RFC call to SAP. You can then examine the data passed to SAP, step through the ABAP code and examine the data passed back to the Siebel application. You need to have the SAPGui installed on the same computer you are executing the RFC from.

Troubleshooting Workflows

This section provides guidelines for resolving workflow issues. To resolve the problem, look for it in the list of Symptoms/Error Messages in Table 51.

Table 51. Resolving Workflow Issues

Symptom/Error Message	Diagnostic Steps/Cause	Solution
When an IDOC is sent from SAP to the Siebel application, you do not see any data for a given component, even though this data is in the IDOC in SAP.	The component may be marked as inactive in the IDOC Integration Object or you may not have captured this component into the integration object. It is also possible that the IDOC structure you are sending does not match the IDOC Integration Object you have in the Siebel implementation.	Check that the version of the IDOC Type you are expecting and the version of the IDOC Type you are sending match. For example, if you are sending a newer version of DEBMAS02, you may be sending a segment with name E2KNA1M004, but if your IDOC Integration object is based on an older version of the IDOC, you may be expecting E2KNA1M. In this case the segment name would not match the integration object component name and it would appear that no data was created for your E2KNA1M component.
When transferring data to or from SAP through a BAPI interface, the decimal position in a numeric field is shifted.	You may need to change External Scale for the field in the integration object.	Refer to "Modifying External Scale Values" on page 94.

Troubleshooting SAP Connection Problems

This section provides guidelines for resolving SAP connection problems. To resolve the problem, look for it in the list of Error Messages in Table 52.

Table 52. Resolving SAP Connection Problems

Error Message	Diagnostic Steps/Cause	Solution
SAP error calling 'RfcOpenEx' for Rfc method ''. Message: 'You are not authorized to logon to the target system (error code 1).', group: 'h', key: 'CALL_FUNCTION_SIGNON_REJECTED'	User name, password, or client number is incorrect.	Check values in SAPRfcUserName, SAPRfcPassword, and SAPRfcConnectString.
SAP error calling 'RfcOpenEx' for Rfc method ''. Message: 'See RFC trace file or SAP system log for more details', group: 'h', key: 'RFC_ERROR_SYSTEM_FAILURE'	Error in destination, in connect string, or in saprfc.ini definition for destination.	Look for the file `dev_rfc.trc` in your bin folder for more information.
The DLL 'SSCAEIRF.DLL' could not be loaded. or The DLL 'SSCAEIIR.DLL' could not be loaded.	Librfc32.dll version 4.6C, which is part of the SAP RFC Software Development Kit (SDK), must be installed on all the machines on which Siebel Tools and Siebel Server are installed. If you receive this error message, it may be due to a missing librfc32.dll file or a file that is an earlier version than 4.6C. This may happen if you have not installed the librfc32.dll from SAP. This can be installed during installation of the SAPGui by selecting the Development Tools check box.	The Siebel applications look for the librfc32.dll file in the SYSTEM32 folder of the Windows NT or Windows 2000 installation. The product version of this file should be 46C. To check the Product Version: **1** Select the file in Windows Explorer. **2** Right-click, and choose Properties. **3** Select the Version tab and the Product Version item in the list box.

Troubleshooting SAP Configuration Problems

This section provides guidelines for resolving SAP configuration problems. To resolve the problem, look for it in the list of Symptoms/Error Messages in Table 53.

Table 53. Resolving SAP Configuration Problems

Symptom/Error Message	Diagnostic Steps/ Cause	Solution
When sending IDOCs from SAP to the Siebel application, no communication IDOCs are sent. You may see the message in SAP "No communication IDOCs have been selected".	Verify that you have added your IDOC message type to the Distribution Model.	Add your IDOC message type to the Distribution Model under SAP transaction SALE as defined in "Distributing Logical Systems" on page 29.

Table 53. Resolving SAP Configuration Problems

Symptom/Error Message	Diagnostic Steps/ Cause	Solution
When sending IDOCs from SAP to the Siebel application, no IDOCs are being received by the Siebel tRFC BAPI Receiver.	Verify that your tRFC BAPI Receiver is connected with SAP Transaction SM59.	Select the TCP/IP RFC destination you created in Chapter 3, "Installation and Configuration" and select the Test Connection button. If this test connection errors: ■ Check your `saprfc.ini` file "TYPE=R" entry against information for your TCP/IP RFC destination. ■ Verify that the program ID in the `saprfc.ini` file matches the program ID name in the RFC destination.
	Go to SAP Transaction WE02 or WE05 and verify that the IDOC has been created and there are no errors in its transmission.	■ **Errors in the status record.** Specific application errors may have occurred. These must be corrected within SAP and the IDOC deleted or resent. BD87 can be used for manual resend. ■ **IDOC with status 30: Idoc ready for dispatch.** Partner profile has been defined to collect IDOCs. A background job (RSEOUT00) needs to be scheduled to send IDOCs or they can be sent manually by transaction WE14 or BD87. If the IDOC is supposed to be sent immediately, the partner profile must be changed to indicate this processing. ■ **IDOC with status 29: Error in ALE service.** Your Partner Profile may be missing an outbound parameter record for your message type. ■ **IDOC with status 03: Data passed to port OK.** There may still be a problem. Execute program RBDMOIND (BD75). This program updates the status of IDOCs that were successful to 12: Dispatch OK. If there was a problem the status remains 03. If status remains 03, check the tRFC queue for entries and reprocess from there.

Table 53. Resolving SAP Configuration Problems

Symptom/Error Message	Diagnostic Steps/ Cause	Solution
When sending IDOCs from SAP to the Siebel application, no IDOCs are being received by the Siebel tRFC BAPI Receiver.	If you do not see any errors at the IDOC layer, check the tRFC layer. Go to SAP transaction SM58 and look for entries here.	■ A network problem occurred or the receiver was not connected at the time the IDOC was sent. Resend transaction, see "Resending Transactions from the tRFC Queue." ■ The transaction may be waiting to be processed or may be currently in process. These entries are removed from the tRFC queue when their processing is complete. ■ Depending upon the load, some transactions may time out. Resend transaction, see "Resending Transactions from the tRFC Queue." ■ A termination in processing occurred. This can happen when the data cannot be saved in the Siebel application. If your receiver is executing without using the EAI Queue, this can indicate that a problem in workflow occurred. If you are using the EAI Queue these errors occur only if the insert of the data to the EAI Queue failed. Check with your database administrator to determine if a problem has arisen. These transactions can be resent, see "Resending Transactions from the tRFC Queue."
IDOCs you are sending into SAP from the Siebel application are not being processed.	There may be many reasons. Start by looking for the IDOC in SAP using transaction WE02 or WE05.	■ **The IDOC is not there.** You may not have set the Sender Partner Number for the IDOC. This is set with the SAPSenderPtnrNum component parameter or method argument. ■ **The IDOC is there.** Check the status record for error information, see "Checking the Status Record for Error Information" on page 186.
Siebel Tools BAPI wizard issues error messages when working with an SAP R/3 4.0 system.	If you are working with an SAP 4.0 version, an additional configuration change is required to use the BAPI input and output wizards.	To make the configuration change, see "Working with SAP R/3 Version 4.0" on page 187.

Resending Transactions from the tRFC Queue

There are three ways to resend transactions from the tRFC Queue.

1. **Manually using SM58.** Select a specific transaction and resend with Edit > Execute LUW.
2. **Through a scheduled background job.** This is the preferred method. You can schedule RSARFCCP to resend failed tRFC calls for a given logical system or RSARFCEX to selectively resend failed tRFC calls. To do this you need to turn off the automatic resend of transactions, as described in the next step.
3. **Automatically through background processes.** Initially this is the default setting for RFC Destinations. This is not recommended if the IDOC load is expected to be heavy. The resend is done with background processes and, depending upon your configuration, can limit your background processes and cause a very slow resend of IDOCs to the Siebel application.

   The automatic resend is configured under SM59 by making changes to your RFC destination. The RFC Destination can be assigned tRFC parameters consisting of three values. (This is done through the tRFC Options selection in the Destination menu under sm59.) These options are:

   a. **Suppress background job if connection error**

      If this flag is set to X and a connection error occurs there is no automatic resubmittal of the transaction to the external application. If it is not set, jobs are automatically created according to the following two parameters.

      NOTE: This flag should be set and resubmission of failed tRFC transactions should be handled through scheduled background jobs.

   b. **Connection attempts up to task**

      This specifies the total number of attempts SAP makes to resend this transaction.

c **Time between 2 tries [mins]**

This specifies the number of minutes between each attempt to resend the transaction.

For example, if a transaction were sent to an external system that did not exist or was not available at the time the transaction was sent, a batch job would be scheduled to resubmit the transaction at a later time. If the value for (b) is 5 and the value for (c) is 10, then the job would be scheduled to run every 10 minutes up to 5 times. At the end of this time the transaction would have to be manually restarted from the tRFC queue (or a batch job to execute it would have to be set up manually).

If these values are not set for the specific RFC Destination, the default values for (b) and (c) are typically 30 and 15 respectively. This means that the default behavior for a failed job (due to connection or time out problems) is to reschedule itself to run every 15 minutes for up to 30 attempts.

You can see the time the batch job is rescheduled to run in sm58. This is the time column on each transaction line. This time is first set to the original submission time of the transaction. If the transaction is scheduled as a background job, this time is updated to be the start time for that job. To see more information about the scheduled job, select a transaction line and choose Information > Display Planned Jobs from the menu.

Remember that the timing values for retry of a transaction determine only when the batch jobs are scheduled to run. A scheduled batch job may not run at the scheduled time, but may run later than this, depending upon the load in SAP and parameters set on the type of job submitted. This can create a very slow transfer of information out of SAP, causing IDOCs to appear in the Siebel application, possibly many hours after they were first created. It is more efficient to schedule your own jobs for the resend of failed transactions.

Checking the Status Record for Error Information

When IDOCs you are sending into SAP from the Siebel application are not being processed and the IDOC is there, check the following status records for error information:

- **Status of 64.** IDOC ready to be passed to application. If you expected the IDOC to be processed immediately, check your Partner Profile to see that you have selected this type of processing. If you do not want to process immediately, you can create a background job to process the IDOC into the application layer. You can schedule RBDAPP01 to process the IDOC. Transaction BD87 can be used to manually process these.

  NOTE: If you expect a large quantity of incoming IDOCs, consider setting the processing type in the partner profile to collect IDOCs, rather than process them immediately. Immediate processing can tie up dialog processes and slow down your performance. Alternatively, you can also use the EAI Queue to control the flow of IDOCs to SAP. Set SAPWriteXML to "WriteOnly" and the `EAI SAP Send Transaction service` user property SAPWaitTime to slow the flow of IDOCs to SAP so that SAP is not flooded.

- **Status of 51.** Application Document not posted. This means that errors occurred when the IDOC was being processed in the application layer. This is normally a data problem. Examine the error message for information. Data can be corrected.

NOTE: The SAP workflow should be configured to send erred IDOCs to an SAP Inbox for processing. This can greatly aid in the debugging process when data errors occur.

Working with SAP R/3 Version 4.0

If you are working with an SAP 4.0 version, an additional configuration change is required to use the BAPI input and output wizards.

To make the configuration change

1. Make a copy of the Siebel Tools .srf file.
2. Start Siebel Tools and lock or checkout the SAP Business Services project.
3. Select Business Service from the Object Explorer.
4. Select the EAI SAP BAPI Input Wizard business service.
5. Select Business Service User Prop from the Object Explorer (under Business Service).
6. Create a new record in the Business Service User Props window.
7. Enter `SAPSysVersion` in the name field and your SAP version in the Value field (for instance, 40B).
8. Repeat steps Step 4 through Step 7 for the EAI SAP BAPI Output Wizard business service.
9. Compile the changes into your copy of the .srf file.
10. Exit Siebel Tools and use the new .srf file for starting Siebel Tools and executing the wizards.

E Creating Integration Touch Points

This appendix provides an example of how to create a new integration touch point using the Siebel Connector for SAP R/3. It consists of the following sections:

Process of Creating New Integration Touch Points

Use the following process to create a new integration touch point.

Identify the SAP Object (IDOC/BAPI)

Start by identifying which business process needs to be automated. For example, you want to check the availability of the material in a given plant. The business requirement is that this needs to be done in real time, which dictates that you must use a BAPI instead of an IDOC.

There are different ways in which you can find out the availability of BAPIs for your SAP version. One way is to use the BAPI browser in SAP. The other way is to use the SE37 transaction in SAP.

The BAPI for the example is BAPI_MATERIAL_AVAILABILITY. Next, you need to study the import and export parameters for the BAPI. Pay special attention to the mandatory parameters for the BAPI. For this BAPI, the mandatory parameters are PLANT, MATERIAL and UNIT. So, you need to specify these parameters before the BAPI is invoked.

Identify the Siebel Business Object

Next, identify the Siebel business object. The example uses products, so the appropriate Siebel business object is Internal Product. Study the business object to see if it contains all the data required for your interface or if it requires a data model extension.

For the BAPI to work, you need to specify MATERIAL, UNIT, and PLANT. The Internal Product business object has an integration ID field. This can be used to store the SAP material number. Unit of measure is also there. However, Plant is missing.

For simplicity of the example, assume that Plant has a fixed value. The alternative would be to use Siebel Tools to extend the data model, but that is beyond the scope of this example.

When you have identified the Siebel and SAP objects, you need to determine the mappings. At minimum, you need to specify the three input parameters (MATERIAL, UNIT, and PLANT). Use the integration ID of the internal product business component, as it contains the SAP material number. The internal product business component has unit of measure information also. For PLANT, set the value to constant value, although this could also be obtained from the extended business component.

You need to store the stock value returned by the BAPI. The field "Stock_Level" can be used for this purpose. If there are no corresponding fields to store the BAPI return values, then you need to extend the Siebel business component that you are working with.

Create BAPI Integration Objects

Create BAPI integration objects next. You need two objects, one that corresponds to the import parameters of the BAPI and one that corresponds to the export parameters. Name these integration objects BAPI_MATERIAL_AVAILABILITY (Input) and BAPI_MATERIAL_AVAILABILITY (Output).

Create an Integration Object for the Siebel Business Object

There is an integration object (Product - Receive SAP Material (Siebel)) for the Internal product. You can copy it. For example, name it Product - Get SAP Material (Siebel).

Create Business Services for Mapping

You need a business service to transform the data between Siebel and SAP objects. Start by creating a business service. Call it ATP - SAP Maps. Then create an Execute method for this business service and create two scripts: one for the inbound and one for the outbound mappings. Look at the existing business services as an example. Chapter 5, "Customizing Integrations," documents business services provided with Siebel Connector for SAP R/3. For the actual code for the business services for this example, see "Sample Business Services" on page 191. *Overview: Siebel eBusiness Application Integration Volume I* covers the data transformation business services. The helper functions used in the sample code are explained in this bookshelf.

Create the Workflow

You need a workflow. Copy an existing workflow and change the process properties. You can use any workflow that uses BAPI integration objects. In this example, copy Orders - Create SAP Order and call it Product - ATP SAP Material. Change the process properties for this workflow as shown in Table 54.

Table 54. Workflow Process Properties

Process Property	Value
Input: BAPI Int Object Name	BAPI_MATERIAL_AVAILABILITY (Input)
Input: Map Name	GetSAPMaterialATP_SiebelToBAPI
Input: Siebel Int Object Name	Product - Get SAP Material (Siebel)
Output: BAPI Int Object Name	BAPI_MATERIAL_AVAILABILITY (Output)
Output: Siebel Int Object Name	Product - Receive SAP Material (Siebel)

Using the workflow process designer, modify the existing business service for the mapping steps. Remember to use the display name. You can find the name in the drop-down list.

Use the workflow simulator to test the workflow. Use the row-ID of the material. Specify this row-ID as the object ID of the process parameter for this workflow. Step through the simulator and debug the workflow. When it is working properly, activate the workflow.

Sample Business Services

```
//////////////////////////////////////////////////////////////////////////////
//
// Copyright (C) 1999, Siebel Systems, Inc., All rights reserved.
//
//  $Revision: 4 $
//      $Date: 12/10/99 10:29a $
//    $Author: Achaudhr $ of last update
//
// CREATOR:    P. Lim
//
// DESCRIPTION
//    Javascript maps from Siebel Order to SAP entities
//
//////////////////////////////////////////////////////////////////////////////

#include "eaisiebel.js"
/* This function shared by "Simulate Order" and "Create Order" */

function GetSAPMaterialATP_BAPIToSiebel (inputMsg, outputMsg)
{
    /*
    * Set up EAI Input Message objects
    */
    var iGSProductObj;   // BAPI Product instance
    var iGSExportComp; // BAPI Export Parameters

    // Store inputId to be denormalized to all RFC table rows
    var inputId = inputMsg.GetArgument ("Product Id");
```

```
    /*
    * Set up EAI Output Message objects
    */
    var oProductObj;     // Siebel Productinstance
    var oProductComp;    // Product

    var oGSOutputComp; // Create Output

    /*
    * Find and create top-level integration object
    */
    iGSProductObj = inputMsg.GetIntObj("BAPI_MATERIAL_AVAILABILITY (Output)");
    oProductObj = outputMsg.CreateIntObj ("Product - Get SAP Material (Siebel)");

    /*
    * Read int object instances from EAI message
    */

    while (iGSProductObj.NextInstance ())
    {
    /*
    * Create "Product" object in output message
    */
    oProductObj.NewInstance ();

    /*
    * Create "Product" component
    */
    oProductComp = oProductObj.CreatePrimaryIntComp ("Product");

    /*
    * Read "Export Parameters" component
    */
    iGSExportComp = iGSProductObj.GetPrimaryIntComp ("Export Parameters");

    /*
    * Create the output component.
    *
    */
    if (iGSExportComp.NextRecord())
        {
        /*
        * Write "Create Output" component
        */
        oProductComp.NewRecord ();
        oProductComp.SetCopySource  (iGSExportComp);
        oProductComp.SetFieldValue  ("Id",     inputId);
        oProductComp.CopyFieldValue ("Stock Level",       "AV_QTY_PLT");

        }

    }

}
```

Index

C

D

E

L

M

O

P

Q

R

S

T

U

W

Z