SIEBEL CONNECTOR FOR SAP R/3

VERSION 7.5, REV. B
SEPTEMBER 2003

Contents

Introduction

Chapter 1. Siebel Connector for SAP R/3 Overview

Chapter 2. Installing and Configuring the SAP Connector

Chapter 3. Standard Integrations

Chapter 4. Customizing Integrations

Testing the Interface

Chapter 5. BAPI Interfaces

Chapter 6. IDOC Interfaces

Chapter 7.   EAI Queue

Chapter 8.   Upgrading from v6.x to v7.x

Appendix A. Data Types Used in BAPIs and IDOCs

Appendix B. SAP R/3 v3.1H Mappings

Appendix C. SAP Field Mappings

Appendix D. Troubleshooting

Appendix E. Creating Integration Touch Points

Index

Introduction

This guide describes the Siebel Connector for SAP R/3, and provides installation and configuration steps for Siebel applications and SAP configuration steps. You need the assistance of qualified SAP experts to install, configure, and maintain SAP applications and interfaces used by Siebel Connector for SAP R/3.

This book is useful primarily to people whose title or job description matches one of the following:

Database Administrators	Persons who administer the database, including data loading; monitoring, backup, and recovery; space allocation and sizing; and user account management.
Installers	Persons responsible for setting up Siebel applications for initial use.
SAP Administrators	Persons responsible for planning, setting up, and maintaining SAP R/3 implementations.
Siebel Application Administrators	Persons responsible for planning, setting up, and maintaining Siebel applications.
Siebel Application Developers	Persons who plan, implement, and configure Siebel applications, possibly adding new functionality.
Siebel System Administrators	Persons responsible for the whole system, including installing, maintaining, and upgrading Siebel applications.

To get the most from this guide, you should have a good working knowledge of Siebel applications and SAP R/3. In addition, you should be aware of the usage of SAP BAPIs and IDOCs.

Product Modules and Options

This *Siebel Bookshelf* contains descriptions of modules that are optional and for which you may not have purchased a license. Siebel's Sample Database also includes data related to these optional modules. As a result, your software implementation may differ from descriptions in this Bookshelf. To find out more about the modules your organization has purchased, see your corporate purchasing agent or your Siebel sales representative.

How This Guide Is Organized

This guide provides information necessary to install, configure, and deploy the Siebel Connector for SAP R/3.

The chapters in this guide are organized so that specific readers can find the information they are looking for.

All readers should be interested in the overview in Chapter 1, "Siebel Connector for SAP R/3 Overview." Chapter 2, "Installing and Configuring the SAP Connector" should be read by Siebel Administrators, SAP Administrators and Application Developers. Chapter 3, "Standard Integrations" should be of interest to the users and Application Developers. The remaining chapters focus on modification and creation of integration scenarios between SAP and Siebel applications and should be of interest to Application Developers.

Additional Resources

Consult with your vendor for relevant books on SAP R/3 BAPIs and IDOCs.

Additional Documentation

The following Siebel documents are helpful in implementing Siebel Connector for SAP R/3.

■ *Siebel Business Process Designer Administration Guide*

■ *Overview: Siebel eBusiness Application Integration Volume I*

■ *Integration Platform Technologies: Siebel eBusiness Application Integration Volume II*

■ *Transports and Interfaces: Siebel eBusiness Application Integration Volume III*

Other Resources

If you use MQSeries, consult with IBM for the appropriate documentation.

Revision History

Siebel Connector for SAP R/3

Version 7.5, Rev. B

Table 1. Changes Made in Version 7.5, Rev. B

Topic	Revision
"Locating the SAP R/3 v3.1H Object Mapping Tables" on page 222	Added procedure for locating the SAP R/3 v3.1 H object mapping tables on the *Siebel Bookshelf*.
"Locating the SAP Field Mappings Tables" on page 225	Added procedure for locating the SAP field mapping tables on the *Siebel Bookshelf*.

Version 7.5, Rev. A

■ Modified descriptions of figures.

Siebel Connector for SAP R/3 Overview   1

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.

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*.

Integrations Used by the Siebel Connector for SAP R/3

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 Differences Defined Between Siebel Applications and SAP

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 2.) 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 2. 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 3 provides a list of many of these terms.

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

Table 4. Business Service Names

Name in Document	Business Service Name
BAPI Adapter	EAI SAP BAPI Adapter
BAPI Workflow Service	EAI SAP BAPI Workflow Service

Table 4. Business Service Names

Name in Document	Business Service Name
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 Prebuilt for the Siebel Connector for SAP R/3

The Siebel Connector for SAP R/3 includes prebuilt standard integrations, which may be used as-is, or be 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 5 lists the standard integrations.

Table 5. 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.

Standard Integrations Provided for SAP Master Data

Table 6 lists the standard integrations provided for SAP Master Data.

Table 6. 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.

Standard Integrations Provided for SAP Transaction Data

Table 7 lists the standard integrations provided for SAP Transaction Data. For more information, read Chapter 3, "Standard Integrations."

Table 7. 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 to create both Siebel and SAP sales orders from a single Siebel quote. A real-time synchronous BAPI interface provides an immediate response from SAP upon the submission of the quote. SAP returns the confirmation for the submitted sales order along with the SAP sales order number. All checks performed by the Siebel Sales Order to SAP Sales Order Standard Integration are also performed on Quote to Sales Order.

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 Administration views.

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.

Siebel EAI 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 complete 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.
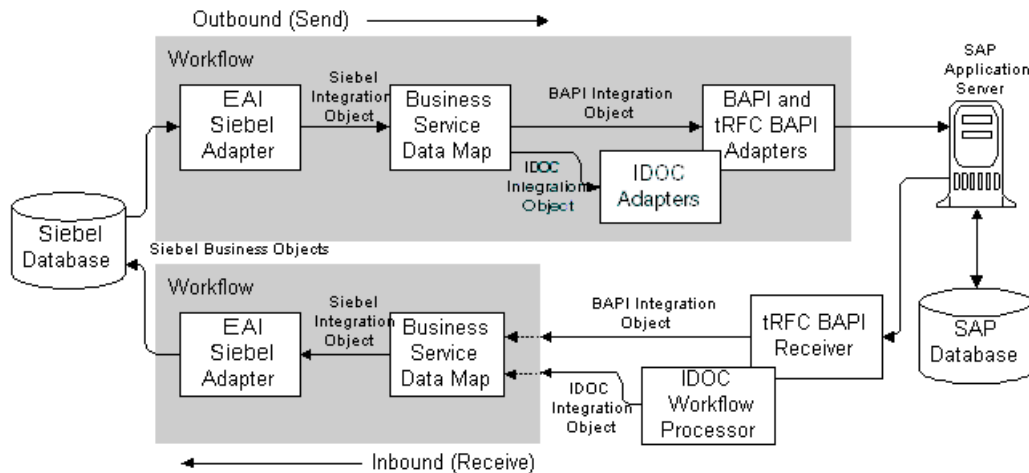
Figure 1. General Architecture of the Integration Process

As Figure 1 on page 20 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 7, "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 Server 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*.

Installing and Configuring the SAP Connector   2

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

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 4, "Customizing Integrations."

NOTE: Before using the Siebel Connector for SAP R/3 make sure you have installed Librfc32.dll version 4.6B, 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 26).

2 Modify Siebel configuration files such as siebel.cfg, uagent.cfg, and tools.cfg (read "Modifying Siebel Configuration Files" on page 31).

3 Configure ALE/IDOC (read "Configuring ALE/IDOC Connectivity" on page 34).

4 Configure components. Enable component groups and components, and set parameters (read "Configuring Server Component Objects" on page 46).

5 Check connectivity (read "Checking Connection Information for SAP" on page 52).

6 Configure for MQSeries (if you desire to use MQSeries) (read "Configuring the Connector for Use with IBM MQSeries" on page 56).

7 Configure SAP for standard integrations (read "SAP Configuration for Standard Integrations" on page 63).

8 Configure the Siebel application for standard integrations (read "Siebel Configuration for Standard Integrations" on page 65).

9 Activate workflows (read "Activating Workflows" on page 72).

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 the "DEST = " line 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 47.

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

5 Edit the "SYSNR = " line 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 in SAP" on page 38).

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 "SAP Connection Problems" on page 228.

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 RFC_INI environment variable

1 Choose Settings > Control Panel > System from Windows start.

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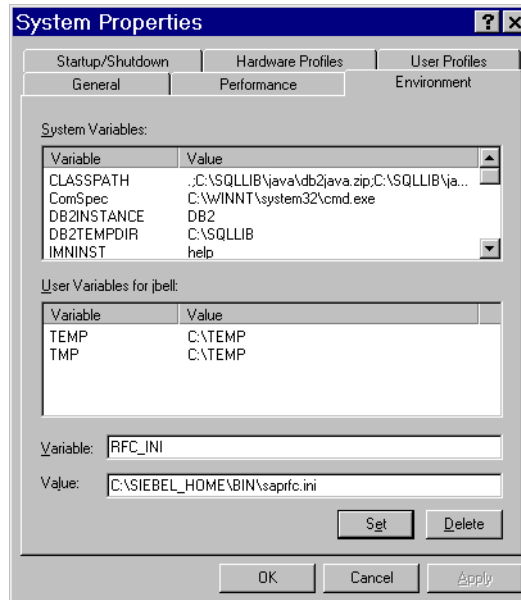
4 In the Value field, enter `%SIEBEL_HOME%\BIN\saprfc.ini`, where `%SIEBEL_HOME%` is the root directory of your Siebel installation.

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 Edit the "SAPRfcUserName = " line to replace CHANGE_ME with the CPIC user name.

4 Edit the "SAPRfcPassword = " line to replace CHANGE_ME with the password for the user name entered.

5 Edit the "SAPRfcConnectString = " line.

    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 Edit the SAPCodepage line to replace the CHANGE_ME with the Siebel Codepage value for your SAP implementation. You can find this value in Table 10 on page 49.

For example, your entry might look something like this now:

```
[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.

To modify the 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.

---

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 Edit the "SAPRfcUserName = " line to replace CHANGE_ME with the CPIC user name.

4 Edit the "SAPRfcPassword = " line to replace CHANGE_ME with the password for the user name entered.

5 Edit the "SAPRfcConnectString = " line.

   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 Edit the "SAPRfcDestEntry = " line to replace CHANGE_ME with the destination name of a TYPE = R entry in the saprfc.ini file.

For example your entry might look something like this now:

```
[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 34.

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.
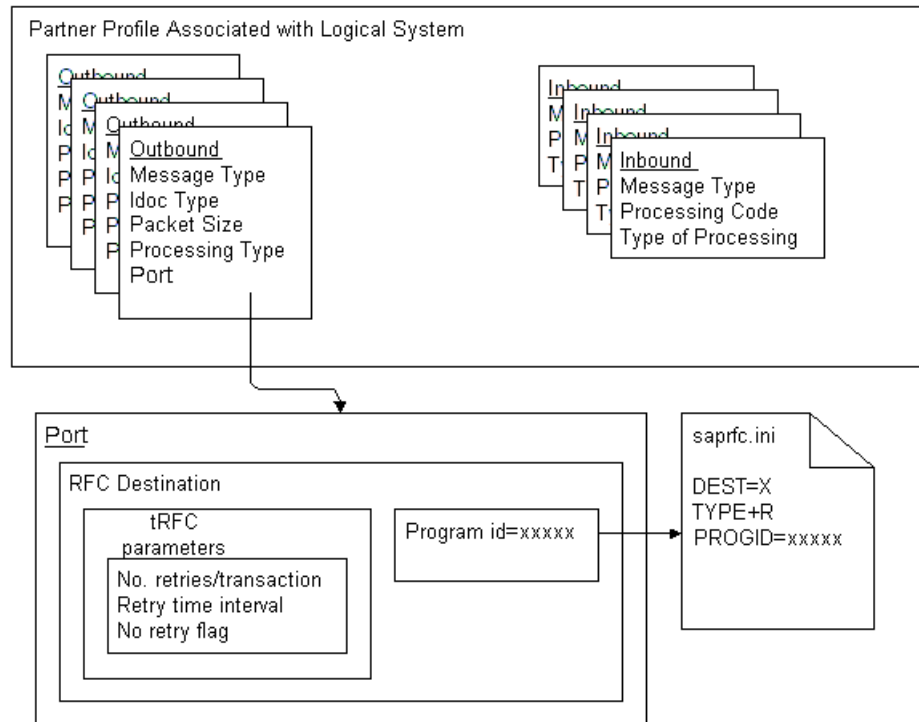
Figure 2. ALE Logical System Definition

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 all 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 8 describes the Outbound Parameters.

Table 8. 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 9 describes the Inbound Parameters.

Table 9. 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 4.6B. 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 steps above 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 in SAP

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, once 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 listed below are used for each corresponding message type:

```
MATMAS = MATMAS03
DEBMAS = DEBMAS02
ORDCHG= ORDERS05
```

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 "Configuring Server Component Objects" on page 46.

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.

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.

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

```
MATMAS = MATMAS03
DEBMAS = DEBMAS02
ORDCHG= ORDERS05
```

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.

Configuring Server Component Objects

The following objects must be configured:

- Server component groups

- Server components

- Server component parameters

Enabling Server Component Groups

You need to enable two component groups using Server Administration > Enterprise Configuration: SAP Connector and Enterprise Application Integration. (EAI). You may also use the Server Manager line mode commands to do this. For information on enabling component groups, read *Siebel Server Administration Guide*.

You may also 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 *Siebel Server Installation Guide for Microsoft Windows*.

Enabling Server Components

You need to enable the SAP BAPI tRFC Receiver and the Business Integration Manager server components. You can use the Siebel application or the Server Manager line mode commands to do this. *Siebel Server Administration Guide* describes how components are enabled.

To enable server components using the Siebel application

1 Navigate to Server Administration > Servers.

2 In the Servers list, select a server.

3 Click the Server Components view tab.

4 Select Business Integration Manager. If its Component State is Offline, click Online.

5 Select SAP BAPI tRFC Receiver. If its Component State is Offline, click Online.

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 Server 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 34).

5 Restart the Siebel Server.

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

1 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 Codepages and Corresponding EAI Value Mappings

As of Siebel 7.5, all Siebel components use Unicode internally. Therefore, all SAP codepage 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 10 shows the SAP codepage mappings delivered with Siebel 7.5. These are configured in the EAI Value Map "SAP Codepage". The EAI Value Maps can be accessed by navigating to Integration Administration > EAI Value Maps and querying for "SAP Codepage" in the Type field. The list of possible Siebel Codepage values is in the Transcode Encoding pick list. The EAI Value Maps values can be modified if required.

Table 10. SAP Codepages and Corresponding EAI Value Mappings

Siebel Value	SAP Codepage	Description
CP1252	1100	SAP Latin-1 - ISO8859-1 - codepage
Unsupported	1402	SAP Latin-2 - ISO8859-2 - unsupported by the Siebel application
Unsupported	1500	SAP Cyrillic - ISO8859-5 - unsupported by the Siebel application
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
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 codepage 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 codepage 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 codepage 1100 (Latin-1, European) and from an SAP implementation using codepage 8000 (Shift-JIS, Japanese). However, you are then responsible for programmatically making sure that codepage 8000 data is not returned to the codepage 1100 SAP implementation, as Japanese characters cannot be converted to the European character set defined by codepage 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 codepage, all data that is transported to and from SAP is transported in a single codepage. When data is sent from the Siebel Database to SAP through the BAPI Adapter, the adapter retrieves the codepage from SAP automatically after connecting. It then looks up the SAP codepage value in the SAP Codepage EAI Value Map and uses the corresponding Siebel codepage 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 codepage value from SAP each time a packet of data is sent from SAP. It then looks up the SAP codepage value in the SAP Codepage EAI Value Map and uses the corresponding Siebel codepage value to translate all data it receives from SAP from the SAP codepage to Unicode.

The interface to a Multi-Display Multi-Processing SAP implementation is similar. The difference is that the codepage 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 codepage. 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 codepage from SAP by setting the SAPCodepage parameter to a Siebel Value as defined in Table 10 on page 49. 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 codepage. 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 codepage is retrieved from the new connection and data is processed using the new codepage.

You can customize the handling of errors that occur during codepage conversion by using the SAPIgnoreCharSetConvErrors parameter. When this parameter is set to True, errors in codepage 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 codepage 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 26.

MQSeries

For MQ Series, the codepage must be explicitly set, therefore only single codepage 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 codepage value. For transport of IDOCs from SAP to the Siebel application, set the SAPCodepage parameter on the SAPIdocAMIMqRcvr component.

Checking Connection Information for SAP

When you have completed all 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 Navigate to Business Process Administration.

2 Click Workflow Processes.

3 Query for the workflow "TestSAPConnection".

4 Click Revise.

5 Click the Process Designer tab.

6 Double-click the Create Test Connection business service box in the diagram.

7 Add three new Input Arguments, one for each of the parameters SAPRfcConnectString, SAPRfcUserName, and SAPRfcPassword. Set their Type to Literal. 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.

8 Click Return To Designer.

9 Click the Process Simulator tab.

10 Click Start.

11 Click Continue.

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

For more information on troubleshooting connection problems, read Appendix D, "Troubleshooting."

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

This section covers starting the SAP tRFC BAPI receiver. You can use either of two methods available.

■ Starting with default tasks parameter

■ Starting receivers independently

Starting with Default Tasks Parameter

You set the default tasks parameter in the component parameters for the SAP BAPI tRFC Receiver.

To set the default tasks parameter

1 Navigate to Server Administration > Components.

2 In the Components list, select the SAP BAPI tRFC Receiver.

3 Click the Component Parameters view tab.

4 Select the parameter named Default Tasks. Set its Current Value field to 1.

5 To have the changes take effect, restart the server component.

To restart the SAP BAPI tRFC Receiver server component

1 Navigate to Server Administration > Servers.

2 In the Servers list, select a server.

3 Click the Server Components view tab.

4 Select SAP BAPI tRFC Receiver. If its Component State is Online, click Shutdown. Wait a few seconds, then click Startup.

If you have a high volume of IDOCS being sent from SAP, you should create one BAPIRcvr for each SAP Application Server dialog process.

In this event all parameters for the BAPIRcvr component must be set prior to the start of the server and all BAPIRcvr components which start use the same parameter values. If you want to start BAPIRcvr components that have different parameter values, you can create a new component definition for a BAPIRcvr with different parameter values or you can start the receivers independently, and pass in their parameter values.

Starting Receivers Independently

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 which connect to different SAP installations.

Testing Receiver Connection

Once 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 Siebel Tools Connectivity

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" 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 Connector for Use with IBM MQSeries

To use IBM MQSeries to transport IDOCS between SAP and the Siebel application, use the SAP IDOC AMI Receiver for MQSeries to manage SAP-to-Siebel messages, and use `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 in SAP" on page 38.

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 =	This is 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 >

Parameter	Description
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 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

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 all the Standard Integration inbound workflows: Account - Receive SAP Customer (4x IDOC Input), Product - Receive SAP 4x Material (4x IDOC Input) and Order - Changes SAP 4x Order Status.

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 "SAP 4x Account - Update SAP Customer (4x IDOC)". The sample workflow is called "Send4xIdoc_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 - Create or Update SAP Customer."

   a Navigate to Business Process Administration > Workflow Process.

   b Locate the "SAP 4x Account - Create or Update SAP Customer" workflow, then click Revise.

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

   c Click the Process Designer tab.

   d Double-click the Update SAP Customer sub-process palette.

   e In the Sub Process field, replace "SAP 4x Account - Update SAP Customer (4x IDOC)" with "Send4xIdoc_UpdateSAPCustomer_MQAMI".

   f Save the changes.

   g Click Return to Designer.

h Repeat Step d to Step f, for the Update Sales Area Info sub-process palette.

i Return to Workflow Processes and activate the workflow.

2 Modify the sample workflow Send4xIdoc_UpdateSAPCustomer_MQAMI.

a From Business Process Administration > Workflow Process locate the sample workflow Send4xIdoc_UpdateSAPCustomer_MQAMI and click Revise.

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

b Click the Process Designer tab.

c Double-click the business service step 4. MQ AMI Transport.

d In the Input Arguments tab, change the value of the argument Connection Subsystem from SAPSubsys to a name of your own choice, for example MQSenderSubsys.

e Save the changes.

f Click Return to Designer.

g Return to workflow processes and activate the workflow.

3 Create the connection subsystem that you have 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, the sample workflow is invoked.

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 4.6B. 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:

■ "Adding Siebel LOV Definitions"

■ "Adding Integration Administration Data" on page 68

Adding 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 11 must be defined within the List of Values (LOVs).

Table 11 and Table 12 contain an SAP Reference column that you may use to find the applicable SAP configuration data in your implementation. Table 12 contains definitions that are provided as part of your Siebel installation.

Table 11. 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)

Siebel Configuration for Standard Integrations

Table 11. List of Values for Customizable SAP Configuration Data

SAP Configuration Data	LOV Type	Display Value	Language Independent Value	SAP Reference
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
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 12. 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 12 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 4.6B. 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 Application Administration 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 and Translate fields.

List of Values							
Type	**Display Value**	**Language Independent Code**	**Language Name**	**Parent LIC**	**Order**	**Active**	**Translate**
ABS_COST_FCT_AREA	Professional Services	Professional Services	English-American		2	✔	✔
ABS_COST_FCT_AREA	Workforce Management	Workforce Management	English-American		3	✔	✔
ABS_COST_FCT_TYPE	Normal	Normal	English-American		1	✔	✔
ABS_COST_FCT_TYPE	Emergency	Emergency	English-American		2	✔	✔
ABS_COST_FCT_VARIABLE	Constraint Violation	Constraint Violation	English-American		1	✔	✔
ABS_COST_FCT_VARIABLE	FSE Overtime	FSE Overtime	English-American		2	✔	✔
ABS_COST_FCT_VARIABLE	Tardiness	Tardiness	English-American		3	✔	✔
ABS_COST_FCT_VARIABLE	Task Exclusion Penalty	Task Exclusion Penalty	English-American		4	✔	✔
ABS_COST_FCT_VARIABLE	Travel Distance	Travel Distance	English-American		5	✔	✔
ACCNT_EMP_SIZE	> 10000	> 10000	English-American		5	✔	✔
ACCNT_REVENUE_SIZE	< 100M	< 100M	English-American		1	✔	✔
ACCNT_REVENUE_SIZE	100M - 250M	100M - 250M	English-American		2	✔	✔
ACCNT_REVENUE_SIZE	250M - 500M	250M - 500M	English-American		3	✔	✔
ACCNT_REVENUE_SIZE	500M - 1B	500M - 1B	English-American		4	✔	✔
ACCNT_REVENUE_SIZE	> 1B	> 1B	English-American		5	✔	✔

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 13 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 4.6B. It may vary in other SAP versions.

Table 13. 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 Integration Administration.

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. Below it is the Organization SAP Sales Areas form showing the details for the selected record, such as Name, SAP Sales Organization, SAP Distribution Channel, SAP Division, Address, City, Zip, State, Country, Main Phone #, Main Fax #, Currency, Type, Parent Organization, and Siebel InterActive Layout fields.

4 In SAP, navigate to the SAP Reference listed for Sales Area in Table 13.

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

To assign SAP plants to Siebel organizations

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

1 In the Siebel application, navigate to Integration Administration > SAP Administration.

2 Select Organization SAP Plants. The following figure shows the Organizations list with a record selected. Below 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 13.

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.

---
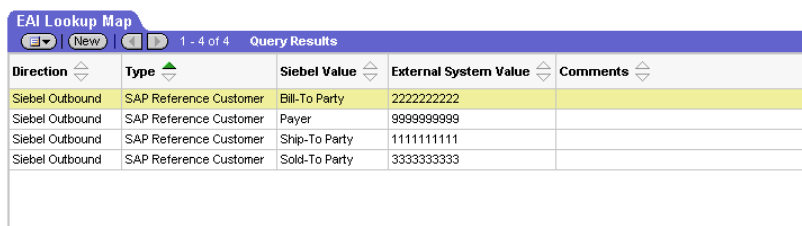
7 Repeat for all plants defined in SAP.

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 Integration Administration.

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.

Direction	Type	Siebel Value	External System Value	Comments
Siebel Outbound	SAP Reference Customer	Bill-To Party	2222222222	
Siebel Outbound	SAP Reference Customer	Payer	9999999999	
Siebel Outbound	SAP Reference Customer	Ship-To Party	1111111111	
Siebel Outbound	SAP Reference Customer	Sold-To Party	3333333333	

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 will appear 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 Business Process Administration.

2 Select Workflow Processes.

3 Query for all SAP Workflows by querying for those whose names fit the pattern "*SAP 4x*".

4 Select each workflow returned by the query individually, and activate a workflow version to use in your application.

NOTE: To activate the workflows previously provided in Siebel 6.x versions, activate all workflows listed in .

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

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

Installation

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 Gateway Server and Siebel Server on a UNIX server.

2 When prompted to specify a file system, use a path name on the UNIX server.

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

4 During Siebel Server installation on a Windows server, 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 server name and the Gateway 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 AIX and Solaris:

      ■ Business Integration Manager

      ■ Business Integration Batch Manager

      ■ SAP BAPI tRFC Receiver

      ■ SAP IDOC AMI Receiver for MQ Series

5 If you use mobile clients, then modify the `cfg` files so they direct the request server to a Windows-based Siebel server.

6 In the Siebel application, navigate to Server Administration > Enterprise Configuration > Batch Component Admin, and then click Synchronize.

7 Restart all Siebel servers.

Standard Integrations   3

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

Account/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

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 2, "Installing and Configuring the SAP Connector." 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 DEBMAS02 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 4.6B. The Business Service Data Map expects to receive a 4.6B DEBMAS02 IDOC. If you are using a version of SAP newer than 4.6B you can send a 4.6B IDOC to the Siebel application by setting the version to "46B" 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 4.6B IDOC to the Siebel application.

Executing Account to Customer

This describes how to transfer accounts and contacts information from the Siebel application applications 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 2, "Installing and Configuring the SAP Connector." 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 Navigate to the Account tab.

2 Pull down the Show menu and choose My Accounts.

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 Pull down the Show menu and choose My Accounts.

The Account list appears.

3 Select an account and click the Back Office (SAP) view tab.

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, Payer, and Status.

4 Fill out the account information including the mandatory Customer Account Group. Choose from Sold-To Party, Ship-To Party, Bill-To Party, or Payer.

5 Scroll down to the SAP Account Sales Area.

The SAP Account Sales Area list appears.

6 Fill out the list fields.

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

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

8 To add a new organization, click New, and repeat Step 7.

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. Once 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 will be 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 you have completed the installation steps as described in Chapter 2, "Installing and Configuring the SAP Connector." In particular, be sure that you have completed all ALE configuration for the message type MATMAS flowing outbound from SAP, 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 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 4.6B. The Business Service Data Map expects to receive a 4.6B 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 4.6B, you can send a 4.6B IDOC to the Siebel application by setting the version to 46B 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 4.6B 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"

- "Quote to Sales Order Standard Integration" on page 85

- "Sales Order Updates Standard Integration" on page 89

- "Account Order History Standard Integration" on page 89

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" on page 81

- "Siebel Sales Order to SAP Sales Order Limitations" on page 84

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 2, "Installing and Configuring the SAP Connector" 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 above 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, it is read-only. Click Get Status to update the Siebel Database with changes to the Sales Order in SAP. 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 Orders tab.

2 Pull down the Show menu and choose All Sales Orders. The following figure shows the Sales Order list and its fields: State, Order #, Type, Status, Priority, Order Date, Account, Last Name, and First Name.

State	Order #	Type	Status	Priority	Order Date	Account	Last Name	First Name
	1-1781505	Sales Order	Open	Medium	8/21/2001 10:33:25	Telenor Mobil As	Alley	Alan
	1-1781636	Sales Order	Open	Medium	8/21/2001 10:33:25	Telenor Mobil As	Alley	Alan
	1-1781681	Sales Order	In-transit	Medium	8/21/2001 10:33:25	Telenor Mobil As	Alley	Alan
	1-1781733	Sales Order	Open	Medium	8/21/2001 10:33:25	Telenor Mobil As	Alley	Alan
	1-1781255	Sales Order	Open	Medium	8/21/2001 10:19:45	Telenor Mobil As	Alley	Alan
	1-1772727	Web Order	Open	Medium	8/21/2001 3:26:25 P	Woollen Goodrich N		
	1-1742708	Web Order	Booked	High	8/19/2001 2:17:53 P	Atlanta Technical	Edwards	Kady

a Create a new Order Header or select an existing Sales Order.

b Enter your sales information in the form, review entries, and then save the record.

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

a Click the Payments view tab and scroll down to Payment Detail - Credit Card.

b Enter the Credit Card Information.

The following information is required for a successful credit card authorization in SAP or the Siebel application (depends on your setup):

- Card Name

- Card Number

- Card Holder

- Expiration date

4 Click the Line Items (SAP) view tab. 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, Ship to Account, Ship to Account Location, Bill to Account, Bill to Account Location, Billing Block, Delivery Block, Credit Status, Message (SAP), Sales Area, Pricing Date, Partial Ship, and Sub Total.

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.

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

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

7 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 are frozen after you submit the sales order to SAP. They are not updated even by pressing Get Status because of limitations of the Get Status BAPI. Also, Schedule Lines cannot be sent from the Siebel application to SAP. They only come 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 and Submit Data flows. If the sales order has been modified in SAP, Get Status does not update the tax and freight information sent from SAP. This is due to a limitation in BAPI_SALESORDER_GETSTATUS. In this event the tax and freight values are set to NULL.

Quote to Sales Order Standard Integration

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

■ "Executing Quote to Sales Order" on page 85

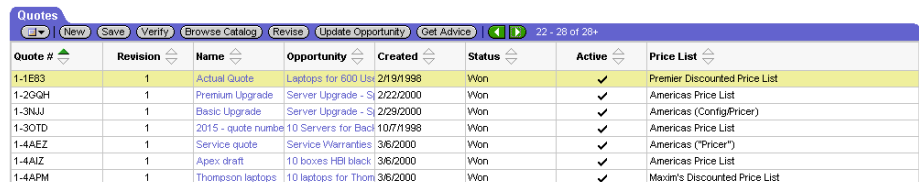■ "Siebel/SAP Pricing Integration for Quotes and Orders" on page 86

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 2, "Installing and Configuring the SAP Connector" 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.

Validate quotes by pressing 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). When you are satisfied with the Delivery Proposal and Pricing you can submit quotes by pressing Place Order. This does not submit quotes directly to SAP but first creates a Siebel sales order and then submits it.

To execute the data flow in the Siebel application

1 Navigate to the Quotes tab.

2 Click New to create a new Quote Header or select an existing Quote. 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
1-1E83	1	Actual Quote	Laptops for 600 Us	2/19/1998	Won	✔	Premier Discounted Price List
1-2GQH	1	Premium Upgrade	Server Upgrade - S	2/22/2000	Won	✔	Americas Price List
1-3NJJ	1	Basic Upgrade	Server Upgrade - S	2/29/2000	Won	✔	Americas (Config/Pricer)
1-3OTD	1	2015 - quote numbe	10 Servers for Back	10/7/1998	Won	✔	Americas Price List
1-4AEZ	1	Service quote	Service Warranties	3/6/2000	Won	✔	Americas ("Pricer")
1-4AIZ	1	Apex draft	10 boxes HBI black	3/6/2000	Won	✔	Americas Price List
1-4APM	1	Thompson laptops	10 laptops for Thom	3/6/2000	Won	✔	Maxim's Discounted Price List

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

   a Click the Payments view tab and scroll down to Payment Detail - Credit Card.

   b Enter the credit card information.

The following information is required for a successful credit card authorization in SAP or the Siebel application, depending on your setup:

- Card Name

- Card Number

- Card Holder

- Expiration date

4 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.

5 To validate a quote or create an order in SAP, click Validate or Place Order.

6 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 2, "Installing and Configuring the SAP Connector."

To add shipping and billing information in Quotes and Orders

1 Click the Fulfillment view tab.

2 Choose Shipping Terms from the predefined pick 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 pick list.

Siebel Application as Price Master

You maintain your complete pricing data in the Siebel application. For more information about Siebel ePricer, 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 14 and Table 15 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 14. Mapping Siebel Fields and Values to SAP BAPI Fields When the Siebel Application Is Price Master

Siebel Field/Value	SAP BAPI Field
SAP Condition Type HB00	ORDER_HEADER_IN-CD_TYPE1
Freight (Header)	ORDER_HEADER_IN-CD_VALUE1
SAP Condition Type PN00	ORDER_ITEMS_IN-CD_TYPE2
Item Price	ORDER_ITEMS_IN-CD_VALUE2

Table 15. 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 16 and Table 17 show the default field mappings used when SAP is the price master.

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

Siebel Field/Value	SAP BAPI Field
SAP Condition Type HB00	ORDER_HEADER_IN-CD_TYPE1
Freight (Header)	ORDER_HEADER_IN-CD_VALUE1
SAP Condition Type PB00	ORDER_ITEMS_IN-CD_TYPE2
Item Price	ORDER_ITEMS_IN-CD_VALUE2

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

SAP BAPI Field	Siebel Field/Value
ORDER_ITEMS_OUT-SUBTOTAL_4	Freight Amount
ORDER_ITEMS_OUT-SUBTOTAL_5	Tax Amount
ORDER_ITEMS_OUT-NET_VALUE	Adjusted List Price * Quantity Requested

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 2, "Installing and Configuring the SAP Connector." 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, all 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 Pull down the Show menu and choose My Accounts.

3 Select an account and click the Back Office Order History view tab. The following figure shows the SAP Account form.

The following is the Back Office Order History view tab with its fields in a list: Order #, Siebel Order #, Order Type, Order Date, Order Status, and Sales Area.

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.

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*.

Customizing Integrations   4

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.

Development Process

Your development process should follow the outline below. Information on each step in the outline is provided. An example of creating a new integration touch point is provided in Appendix E, "Creating Integration Touch Points."

1 Modifying the standard integrations (read "Modifying Standard Integration Interfaces" on page 94).

   a Understanding the standard integrations.

   b Modifying business service data maps.

2 Defining your business interface (read "Defining Your Business Interface" on page 107).

   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 customizations (read "Making Necessary Siebel Application and SAP R/3 Customizations" on page 116).

   a Extend the Siebel Database, business objects, and user interface if necessary.

    b Customize SAP as necessary.

4 Building the interface (read "Building the Interfaces" on page 118).

    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 (read "Testing the Interface" on page 145).

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.

- "Customer to Account Business Data Flow" on page 95

- "Account to Customer Business Data Flow" on page 95

- "Material to Product Business Data Flow" on page 98

- "Siebel Sales Order to SAP Sales Order Business Data Flows" on page 98

- "Quote to Sales Order Business Data Flow" on page 101

- "Sales Order Updates Business Data Flow" on page 104

- "Account Order History Business Data Flow" on page 105

Appendix C, "SAP Field Mappings" contains tables with all 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 SAP 4x Account-Update Siebel Account (4x IDOC) 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.

IDOC Type: DEBMAS02
Message Type: DEBMAS

Workflow: SAP 4x Account - Update Siebel Account (4x IDOC).

Business Service: SAP 4x Account - SAP Maps (MO)
eScript Function: ReceiveSAPCustomer_IDOCToSiebel_4x

SAP R/3   Siebel Adapter   Siebel Database

Busines
Object:
SAP 4x
Account

Integration Object: Account - Receive SAP Customer (4x IDOC Input) ————▶ Integration Object: SAP 4x Account
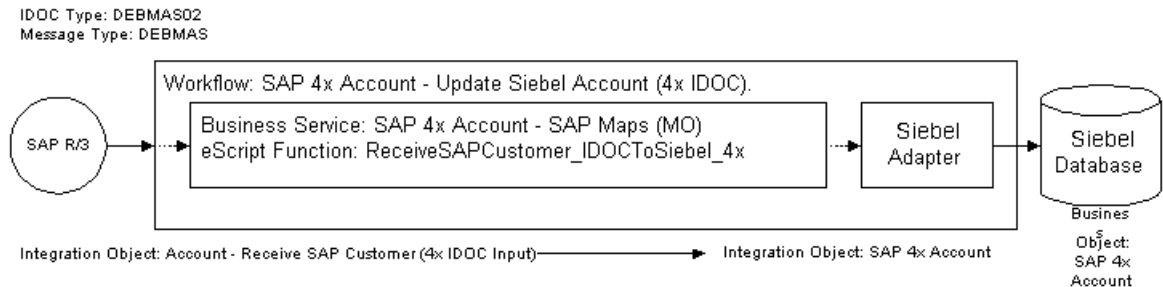
Figure 4. Business Data Flow for the Customer to Account Standard Integration

Account to Customer Business Data Flow

In this scenario Siebel 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 two business data flows at work here.

1 Create Initial SAP Customer from Siebel Account (SAP General Data)

2 Update SAP Customer from Siebel Account (SAP Sales Area Data)

These business data flows are called from the high-level workflow, "SAP 4x Account - Create or Update SAP Customer". This workflow branches based on whether or not a new customer is being created or a customer is being updated in SAP. If a new customer is being created, the first business data flow is invoked followed by the second data flow. If a customer is being updated, then only the second Business data flow is invoked.

The first business data flow (Figure 5) performs an initial creation of the customer in SAP. This is done through a synchronous BAPI call to SAP. This call provides an immediate response with the customer ID from SAP. Figure 5 shows the details of the SAP 4x Account-Create SAP Customer (BAPI) workflow. In this workflow, a record from the Siebel Database is extracted by the Siebel Adapter, passed to the SAP 4x Account-SAP Maps (MO) business service, and then passed to the BAPI adapter. The BAPI adapter communicates with SAP R/3 using the BAPI_CUSTOMER-CREATEFROMDATA BAPI, followed by the BAPI-TRANSACTIONCOMMIT BAPI. SAP R/3 returns the record to the BAPI Adapter, which forwards it to the SAP 4x Account - SAP Maps (MO) business service, which then forwards it the Siebel Adapter, which writes the record back to the Siebel Database.
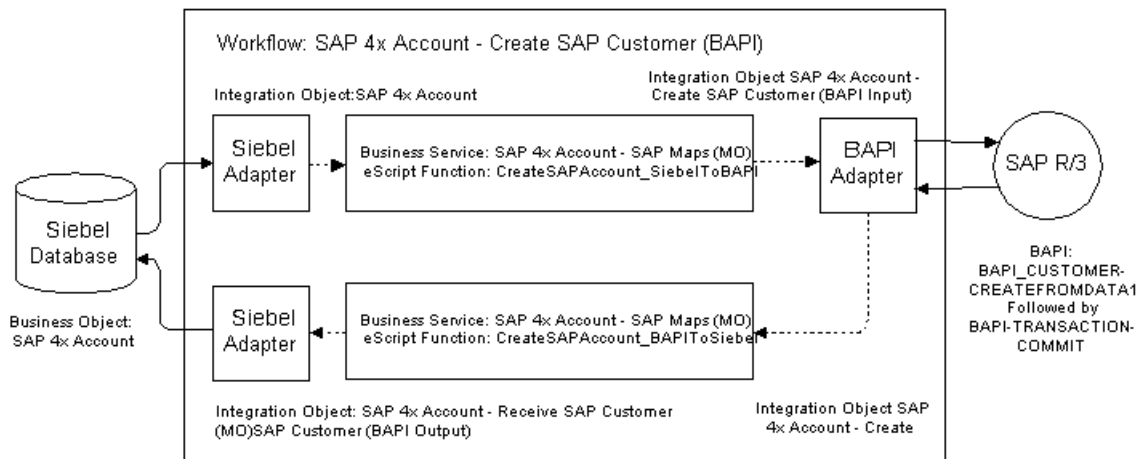
Figure 5. Business Data Flow Using BAPI Adapter

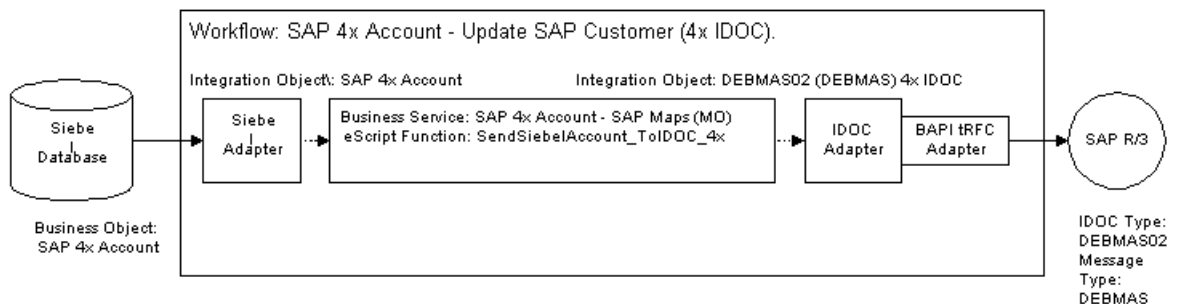In this business data flow, the Siebel Adapter extracts the account information. It passes an integration object instance containing this data to a business service data map, which converts it into a BAPI input integration object. Then it passes the integration object instance to the BAPI adapter and makes a call to SAP to create the customer. An additional call (not shown in Figure 6) commits the customer created in SAP. 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.

Figure 6 shows the second business data flow, which updates the customer through an IDOC. The IDOC can pass in more information for the customer than the BAPI call. In particular, an IDOC can pass sales area information for the customer but a BAPI cannot. The figure illustrates the SAP 4x Account-Update SAP Customer (4x IDOC) workflow, which consists of the Siebel Adapter passing a record to the SAP 4x Account - SAP Maps (MO) business service. The SAP 4x Account - SAP Maps (MO) business service passes the record to the IDOC Adapter, which uses the BAPI tRFC Adapter to pass the record to SAP R/3 using the DEBMAS02 IDOC type.
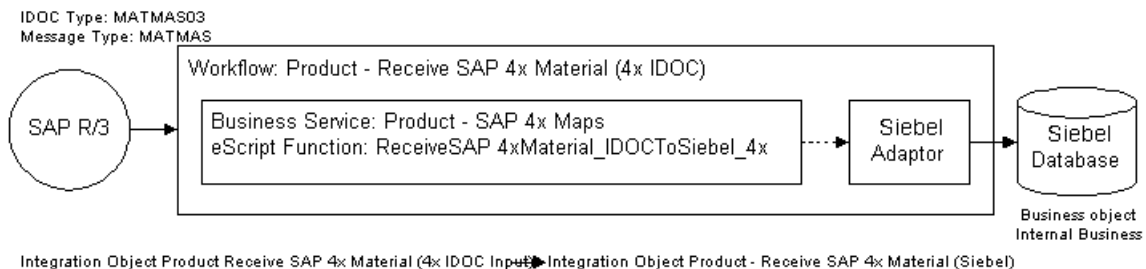

Figure 6. Business Data Flow Using IDOC Adapter

This business data flow extracts account information from the Siebel Database using the Siebel Adapter and passes this to a business service data 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. This business service is very similar to the BAPI adapter, but handles Transactional RFC calls. For more information on this adapter as well as the IDOC adapter, read Chapter 5, "BAPI Interfaces," and Chapter 6, "IDOC Interfaces."

Material to Product Business Data Flow

Figure 7 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 4x Material (4x IDOC) workflow. The workflow consists of the Product-SAP 4x 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.

IDOC Type: MATMAS03
Message Type: MATMAS

Workflow: Product - Receive SAP 4x Material (4x IDOC)

Business Service: Product - SAP 4x Maps
eScript Function: ReceiveSAP 4xMaterial_IDOCToSiebel_4x

SAP R/3 → Siebel Adaptor → Siebel Database

Business object Internal Business

Integration Object Product Receive SAP 4x Material (4x IDOC Input) Integration Object Product - Receive SAP 4x Material (Siebel)

Figure 7. 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

Three business data flows support these actions. Figure 8 shows the business data flow to validate a sales order. This consists of the Order-Simulate SAP 4x Order workflow. This workflow extracts a record from the Siebel Database, using the Siebel Adapter, which passes the record to the Order-SAP 4x Maps business service. This business service sends the record to the BAPI Adapter. The BAPI Adapter communicates with SAP R/3 using the BAPI SALES ORDER SIMULATE BAPI. SAP R/3 returns the record through the BAPI Adapter, which passes it to the Order-SAP 4x Maps business service. The Order-SAP 4x Maps business service passes it to the Siebel Adapter, which writes it to the Siebel Database.
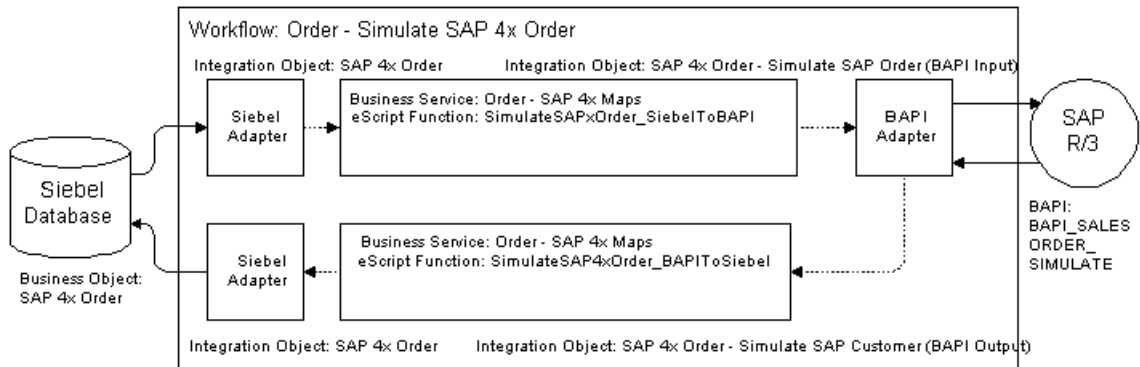
Figure 8. Business Data Flow to Validate a Sales Order

Modifying Standard Integration Interfaces

Figure 9 shows the business data flow for submitting the sales order to SAP. This consists of the Order-Create SAP 4x Order workflow. This workflow extracts a record from the Siebel Database using the Siebel Adapter, which sends the record to the Order-SAP 4x Maps business service. This business service passes the record to the BAPI Adapter. The BAPI Adapter communicates with SAP R/3 using the BAPI SALES ORDER_CREATEFROMDAT1 BAPI. SAP R/3 returns the record through the BAPI Adapter. The adapter sends it to the Order-SAP 4x Maps business service. This business service transmits the record to the Siebel Adapter, which writes it to the Siebel Database.
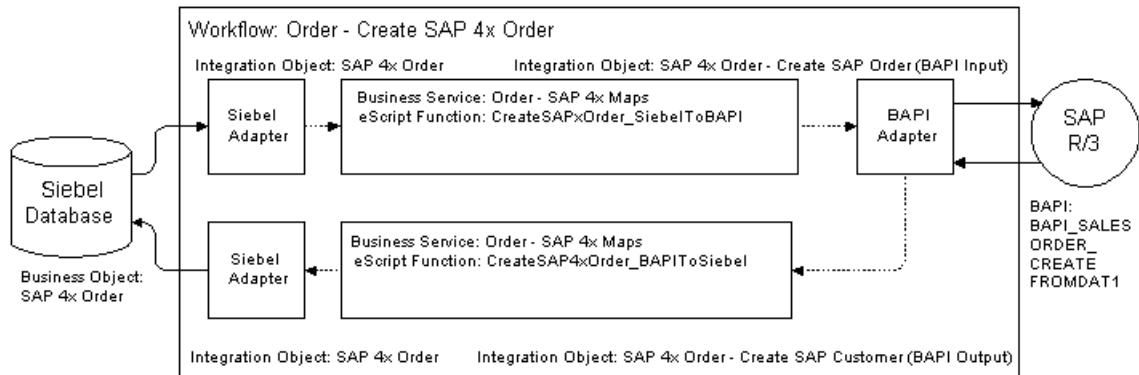
Figure 9. Business Data Flow for Submitting the Sales Order to SAP

Figure 10 shows the business data flow for retrieving sales order status. This consists of the Order-Create SAP 4x Order workflow. This workflow extracts a record from the Siebel Database, using the Siebel Adapter, which passes the record to the Order-SAP 4x Maps business service. This business service sends the record to the BAPI Adapter. The BAPI Adapter communicates with SAP R/3 using the BAPI SALES ORDER GETSTATUS BAPI. SAP R/3 returns the record through the BAPI Adapter, which passes it to the Order-SAP 4x Maps business service. This business service sends the record to the Siebel Adapter, which writes it to the Siebel Database.

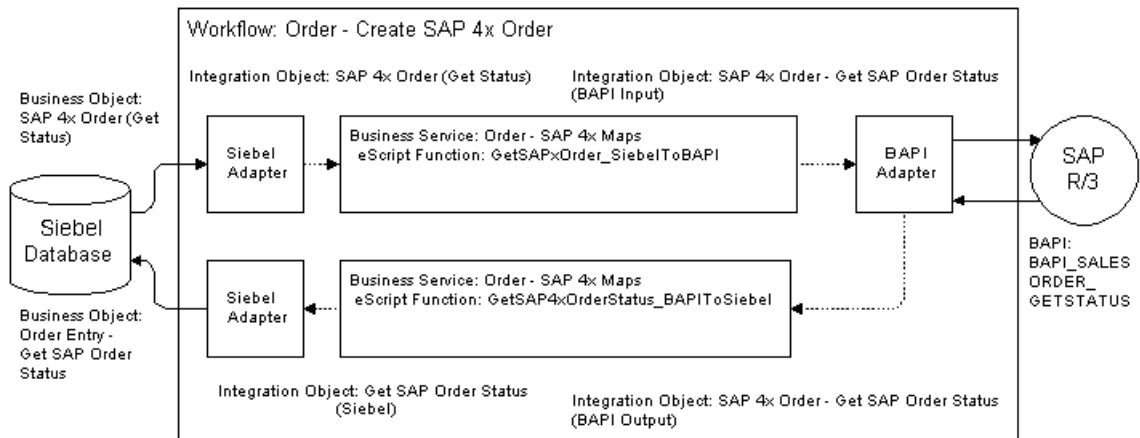**Figure 10. Business Data Flow for Retrieving Sales Order Status**

Quote to Sales Order Business Data Flow

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

■ Validate the Quote

■ Submit the Quote as a Sales Order

Figure 11 shows the business data flow for quote validation, which is similar to sales order validation, except for some additional business service data maps that convert a quote to a sales order and a sales order to a quote.

This consists of the Quote to Order SAP 4x Validate workflow. This workflow extracts a record from the Siebel Database, using the Siebel Adapter, which passes the record to the Order-SAP 4x Maps business service. This business service sends the record to the BAPI Adapter. The BAPI Adapter communicates with SAP R/3 using the BAPI_SALES ORDER_SIMULATE BAPI. SAP R/3 returns the record through the BAPI Adapter, which passes it to the Order-SAP 4x Maps business service. This business service sends the record to the Siebel Adapter, which writes it to the Siebel Database.

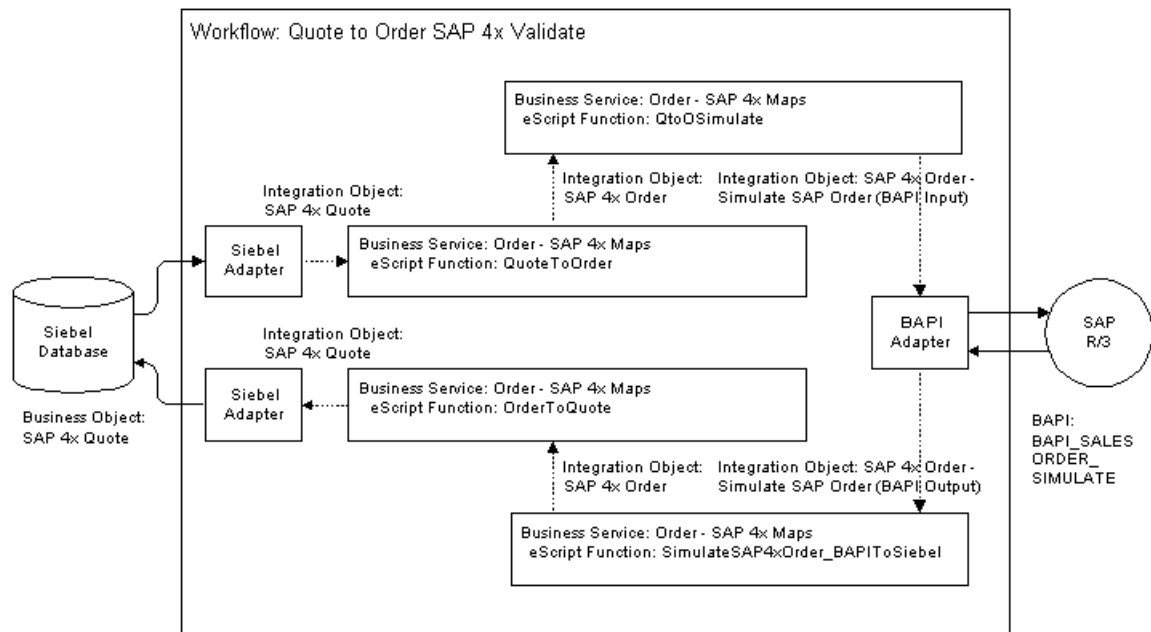Figure 11. Business Data Flow for Converting Quotes and Sales Orders

Submitting the quote as a sales order to SAP is similar to submitting a sales order. Figure 12 shows this business data flow. This consists of the Quote to Order SAP 4x Submit workflow. This workflow extracts a record from the Siebel Database, using the Siebel Adapter, which passes the record to the Order-SAP 4x Maps business service. This business service sends the record to the Siebel Adapter. The Siebel Adapter writes the record back the Siebel Database and calls the Order-Create SAP 4x Order workflow. The Order-Create SAP 4x Order workflow communicates with SAP R/3 using the BAPI_SALES ORDER_CREATEFROMDAT1 BAPI. SAP R/3 returns the record through the Order-Create SAP 4x Order workflow, which updates the Siebel Database. Note that the Siebel adapter is called to insert the sales order into the Siebel Database prior to the call to SAP.
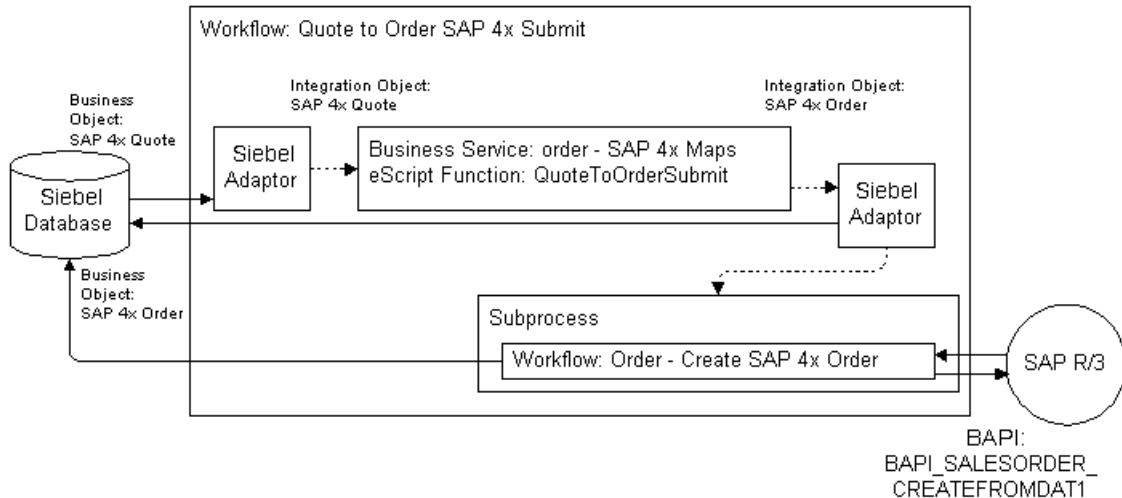
Figure 12. Business Data Flow for Quotes and Sales Orders

Sales Order Updates Business Data Flow

Figure 13 shows the business data flow for the Sales Order Updates standard integration. It is triggered by the arrival of an ORDCHG message from SAP. SAP can be configured to send this message when a sales order in SAP has been changed. When the Siebel application receives an IDOC associated with the ORDCHG message, it retrieves the Sales Order ID from the message and calls the workflow "Order - Change SAP 4x Order Status" to retrieve Sales Order status from SAP and update the Siebel Database. The Siebel Sales Order to SAP Sales Order standard integration covers this workflow.

IDOC Type: ORDER02
Message Type: ORDCHG

Workflow: Order - Changes SAP 4x Order Status

Business Service: Order - SAP 4x Maps
eScript Function: OrderStatus

SAP R/3

Integration Object: SAP 4x Order Status (IDOC 4x)

Workflow: Order - Change SAP 4x Order Status

Siebel Database
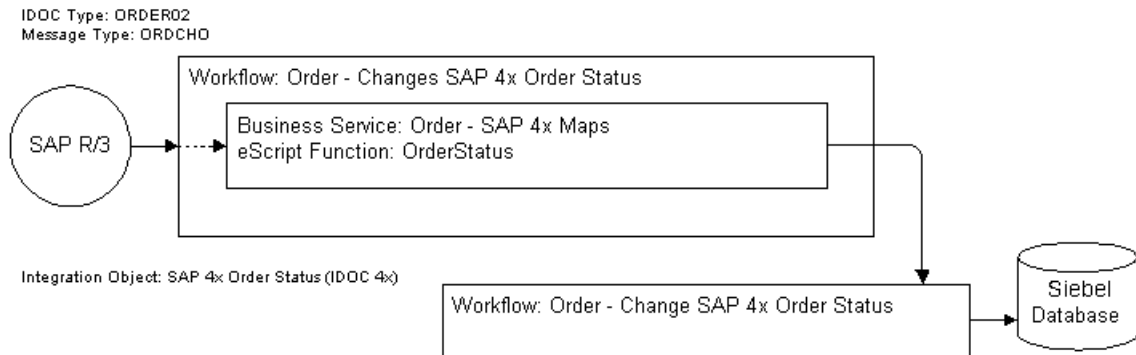
Figure 13. Business Data Flow for the Sales Order Updates Standard Integration

Account Order History Business Data Flow

Retrieval of Account Order History contains two business data flows. Figure 14 shows the business data flow for Get SAP Order List. This consists of the Account-Get SAP Order List (MO) workflow. This workflow extracts a record from the Siebel Database, using the Siebel Adapter, which passes the record to the Order-SAP 1x Maps business service. This business service sends the record to the BAPI Adapter. The BAPI Adapter communicates with SAP R/3 using the BAPI SALES ORDER GETLIST BAPI. SAP R/3 returns the record through the BAPI Adapter, which passes it to the Order-SAP 4x Maps (MO) business service. This business service sends the record to the Siebel Adapter, which writes it to the Siebel Database.
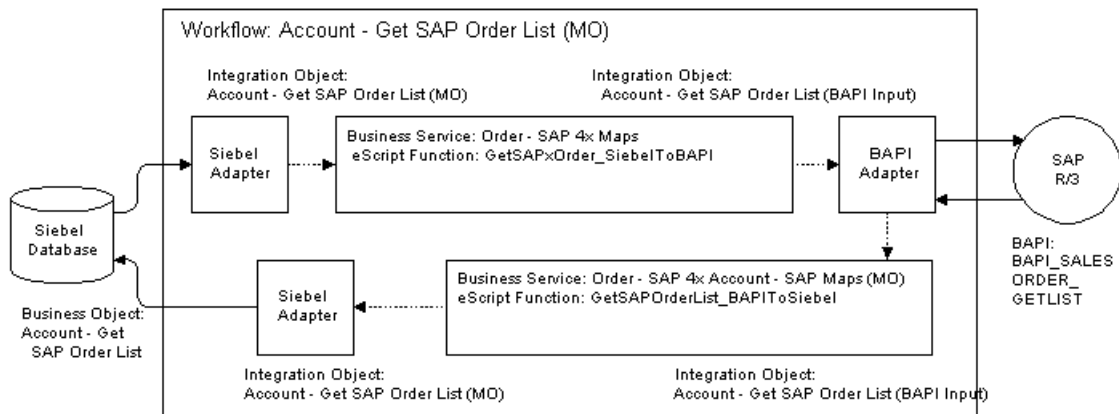
Figure 14. Business Data Flow for Get SAP Order List

This flow retrieves the list of orders from SAP when given a customer ID from an account in the Siebel application. When you select one of these orders, it can be imported from SAP into the Siebel application. Figure 15 shows the business data flow for importing the SAP order. This consists of the Account-Import SAP Order (MO) workflow. This workflow extracts a record from the Siebel Database using the Siebel Adapter, which passes the record to the SAP 1x Order (Get SAP Order List) (MO) business service. This business service creates the Account-Import SAP Order (Siebel Order List) (MO) integration object and sends this integration object to the Siebel Adapter, which writes it to the Siebel Database.
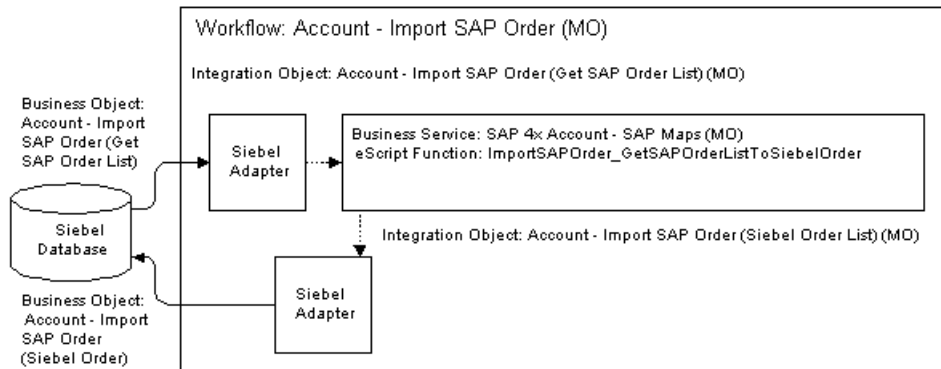
Figure 15. Business Data Flow for Importing the SAP Order

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 16 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.
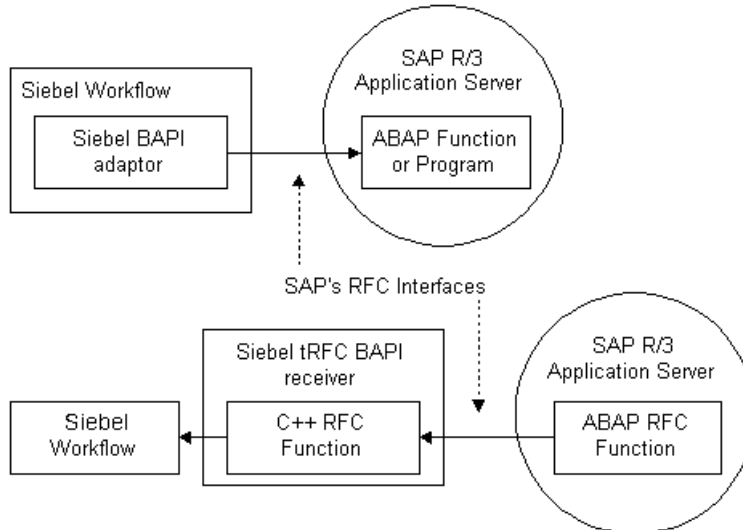
Figure 16. SAP's RFC Interface

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 complete 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. Once 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 17 shows this relationship between the Siebel UI Layer, Business Object Layer, and Data Object Layer. Table 18 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.
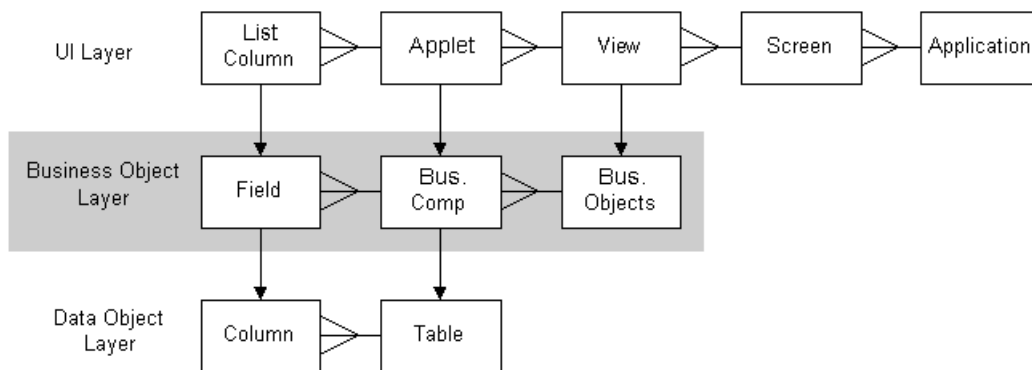
Figure 17. UI, Business Object, and Data Object Layer Relationships

Table 18. 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

Once 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 18. 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.
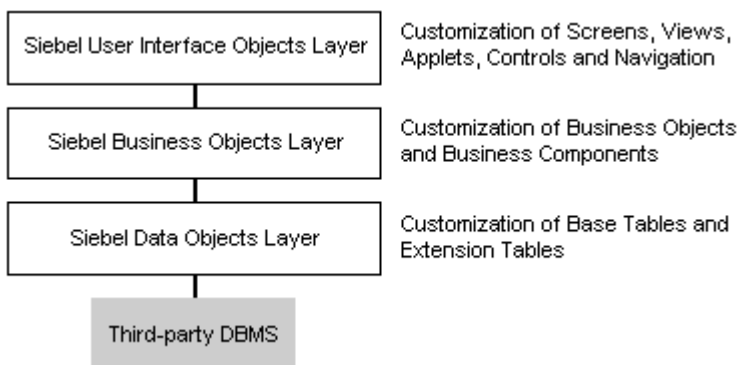
Figure 18. Siebel Architecture Diagram

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 *Siebel Tools Reference*.

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.

All 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"

- "Creating Siebel Integration Objects" on page 132

- "Creating Business Service Data Maps" on page 133

- "Creating Workflows" on page 137

- "Adding the Siebel Adapter" on page 138

- "Adding the SAP Interfaces" on page 139

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.
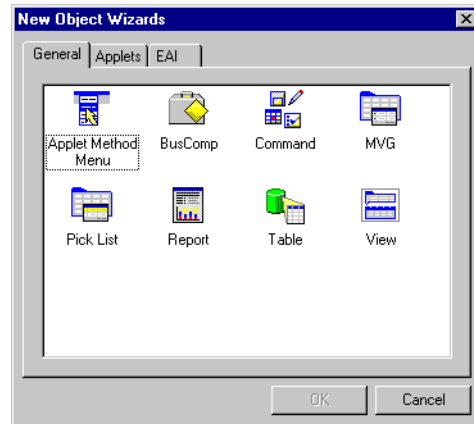
To create IDOC integration objects

1 Start Siebel Tools.

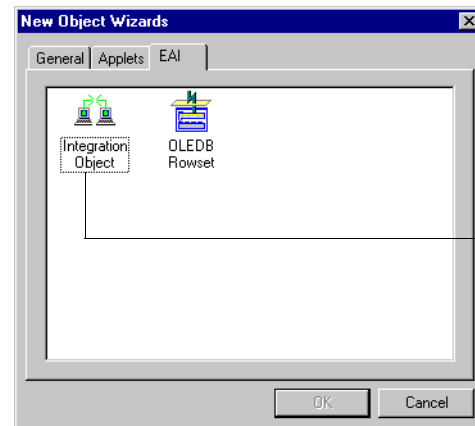**2** Make sure that you have a project checked out into which you can add the new integration object.

3 Click New or File > New Object.

The New Object Wizards window opens. The following figure shows the window's three tabs: General, Applets, and EAI.


4 Click the EAI tab.

The following figure shows the tab's two icons: Integration Object and OLEDB Rowset.

Select the Integration Object icon and click OK.

5 Select the Integration Object icon and click OK.

The Integration Object Builder window opens. The following figure shows the first page of this wizard, which has two drop-down lists. The first drop-down list is for the project the integration object will be part of, and the second list is for the source system on the new integration object.
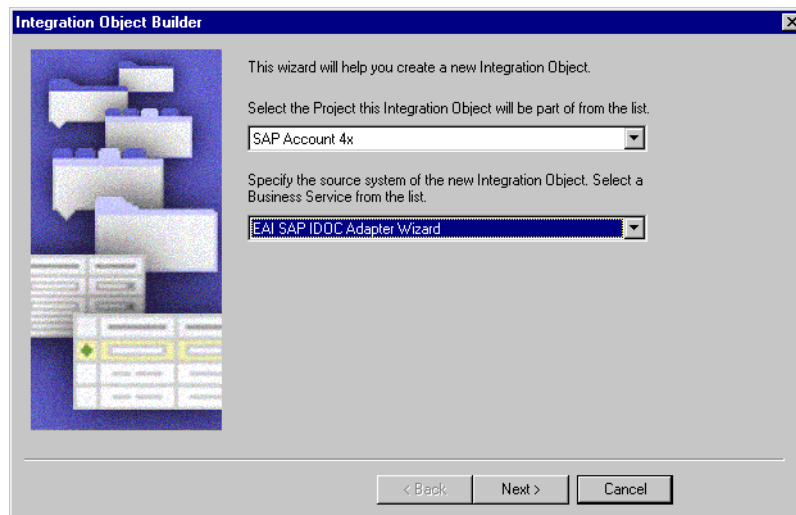
6 In the top drop-down list, choose the project into which you want to add the integration object.

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.

NOTE: There is a second IDOC Wizard visible in the drop-down list marked with the version number 6.2. This wizard supports the 6.x version of Siebel's IDOC Adapter/Receiver.

8 Click Next.

The second page of the wizard appears. The following figure shows this page with a drop-down list field to select a source object for the new integration object, and a text entry field to provide a unique name for the new integration object.
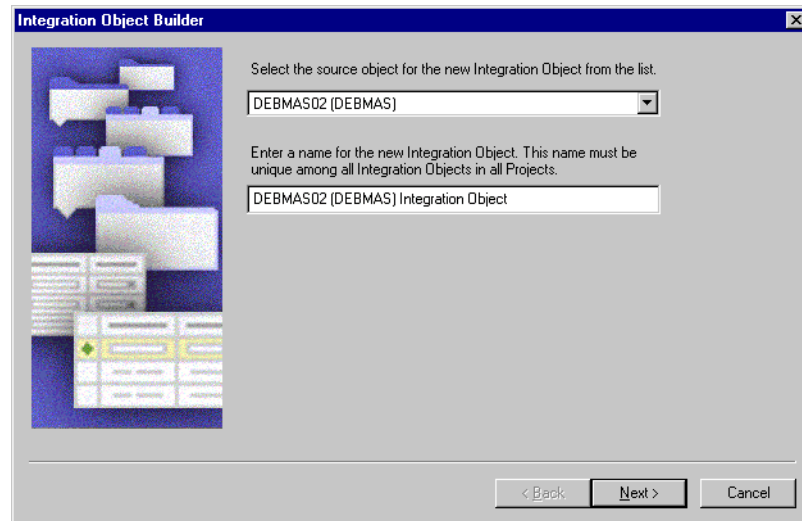
9 From the drop-down list, choose an SAP Object.

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 26 and "Modifying Siebel Configuration Files" on page 31.

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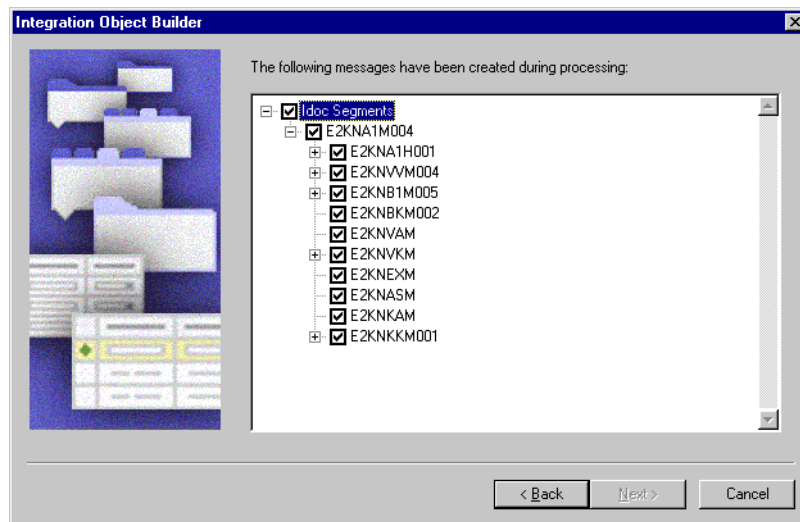
13 If you do not intend to use a component, clear its check box. Components are captured only if they are not disabled at this point. Capture all components and disable those that you do not use by setting the Inactive flag for the component in the integration object.

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 6, "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.

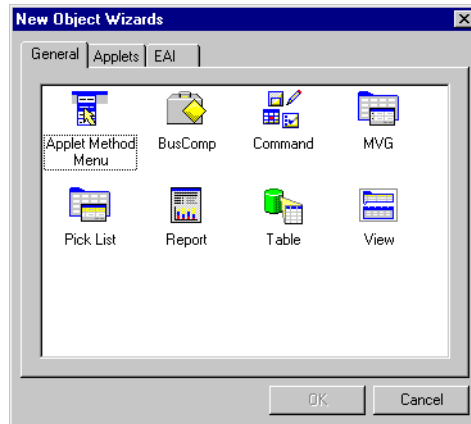### *To capture inbound calls*

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. The following figure shows the window's three tabs: General, Applets, and EAI.

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.

The following figure shows the first page of this wizard, which has two drop-down lists. The first drop-down list is for the project the integration object will be part of, and the second list is for the source system on the new integration object.
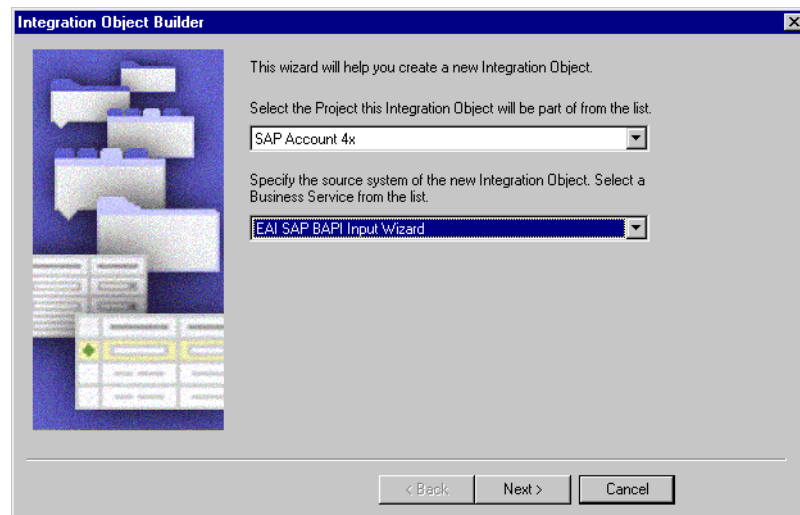
6 Click Next.

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. The following figure shows this page with a drop-down list field to select a source object for the new integration object, and a text entry field to provide a unique name for the new integration object.
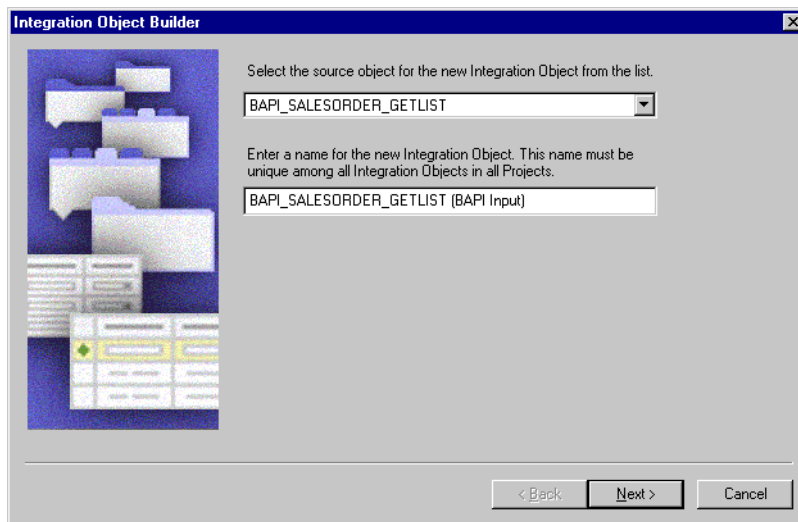
9 Click Next.

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 26 and "Modifying Siebel Configuration Files" on page 31.

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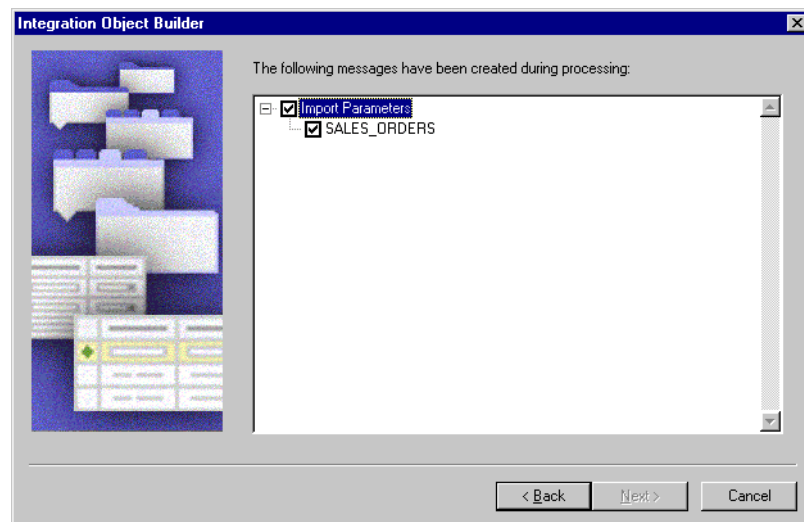
12 Click Finish to create the integration object in the Siebel repository.

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 5, "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 19 on page 130 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.
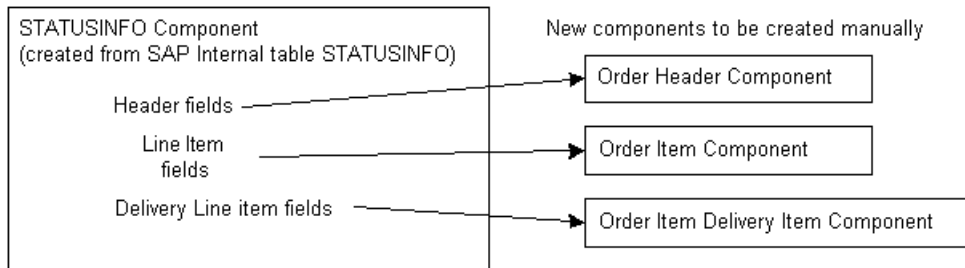
Figure 19. Normalization of Integration Object Components

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

All 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. Once 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 Once 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 and Business Service Data Maps, read *Siebel Tools Online Help* and *Siebel Tools 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 Integration Administration > 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	
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.

The first figure shows the Integration Field Maps form (and its fields: Name, Group, Persistence Frequency, Created By, Business Object, Activation Date/Time, Persistence Level, Created, Status, Expiration Date/Time, Error Process Name, and Version). The second figure is the Process Designer with the following steps defined: Start Step, Map IDOC to Siebel DDTE, Sync data to the Siebel application, and End Step.
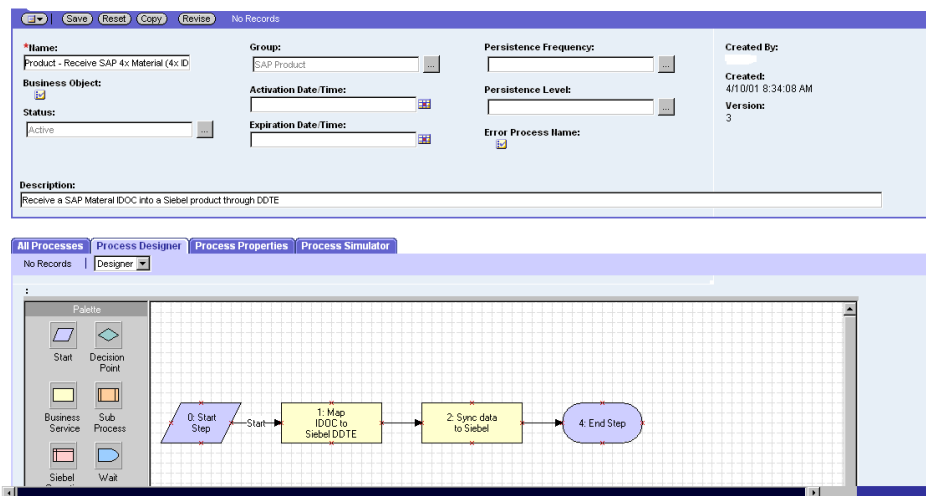
6 Create new maps between any two integration objects, and then build your new integration point based on your business requirement. For Siebel Data Mapper limitations in this release, read *Overview: Siebel eBusiness Application Integration Volume I*.

NOTE: Siebel Connector for SAP R/3 includes a sample map (ReceiveSAP4xMaterial_IDOCToSiebel_DDTE) and a sample workflow (Product - Receive SAP 4x Material (4x IDOC) - DDTE). The sample differs slightly in functionality from the eScript map that performs this function. The "Description" field mapping requires aggregation and sorting features of Siebel Data Mapper, which are not supported by the current release. This may be available in a later release. For now this field is now mapped with empty string.


Creating Workflows

Once you have created the Business Service Data Map, you can create a workflow to execute it. *Siebel Business Process Designer Administration Guide* provides more information.

To create a workflow

1 Navigate to Business Process Administration > Workflow Processes view.

2 Add a new record to the Workflow Processes list.

3 Enter a name for the workflow.

4 Click the Process Designer view tab.

5 Select the Start step from the palette and drag it to the diagram.

6 Add business services to your flow.

The following topics describe the types of Business Services you are likely to use in building an SAP interface workflow.

The following figure shows the Process Designer in its initial appearance: a toolbox of functions on the left edge and a Start Step in the workspace. The toolbox functions are Start, Decision Point, Business Service, Sub Process, Siebel Operation, Wait. User Interact, Stop, End, Connector, and Exception.
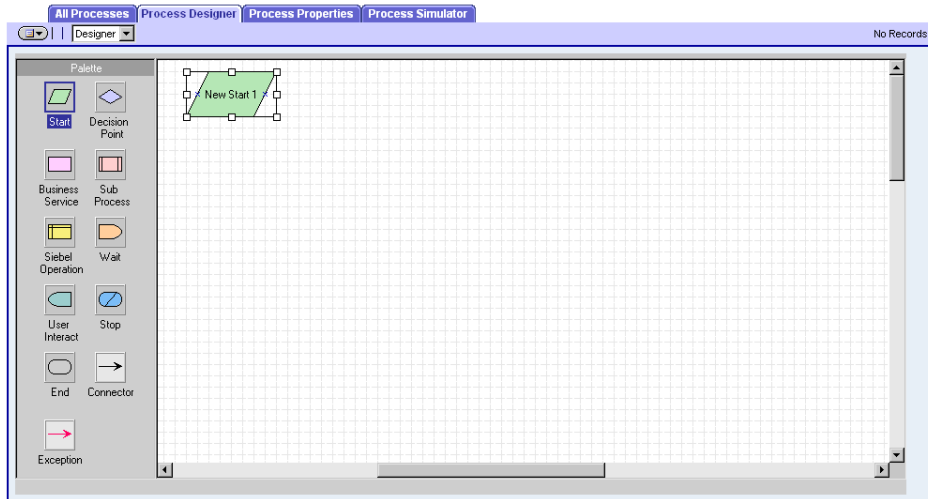
Adding the Siebel Adapter

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 From the Workflow Diagram Editor, drag a Business Service box to your diagram.

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 Edit the Name field to add a name for your workflow step.

5 From the drop-down list for Business Services, choose the EAI Siebel Adapter.

6 From the drop-down list for Business Service Methods, choose a method.

7 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 From the Workflow Diagram Editor, drag a Business Service box to your diagram.

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 Edit the Name field to add an appropriate name for your step.

5 From the drop-down list for Business Services, choose the `EAI SAP BAPI Adapter`.

6 From the drop-down list for Business Service Methods, choose the Execute method.

7 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 5, "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 From the Workflow Diagram Editor, drag a Business Service box to your diagram.

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 Edit the Name field to add an appropriate name for your step.

5 From the drop-down list for Business Services, choose the `EAI SAP BAPI Adapter (tRFC)`.

6 From the drop-down list for Business Service Methods, choose the Execute method.

7 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 5, "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 From the Workflow Diagram Editor, drag a Business Service box to your diagram.

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 Edit the Name field.

5 From the drop-down list for Business Services, choose the EAI SAP IDOC Adapter.

6 From the drop-down list for Business Service Methods, choose the Send method.

7 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 5, "BAPI Interfaces." For more information on using the IDOC Adapter read Chapter 6, "IDOC Interfaces." The SAP standard integrations provide examples of sending an IDOC to SAP.

SAP-to-Siebel Interfaces

The tRFC BAPI Receiver processes all 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 will be 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 20 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.
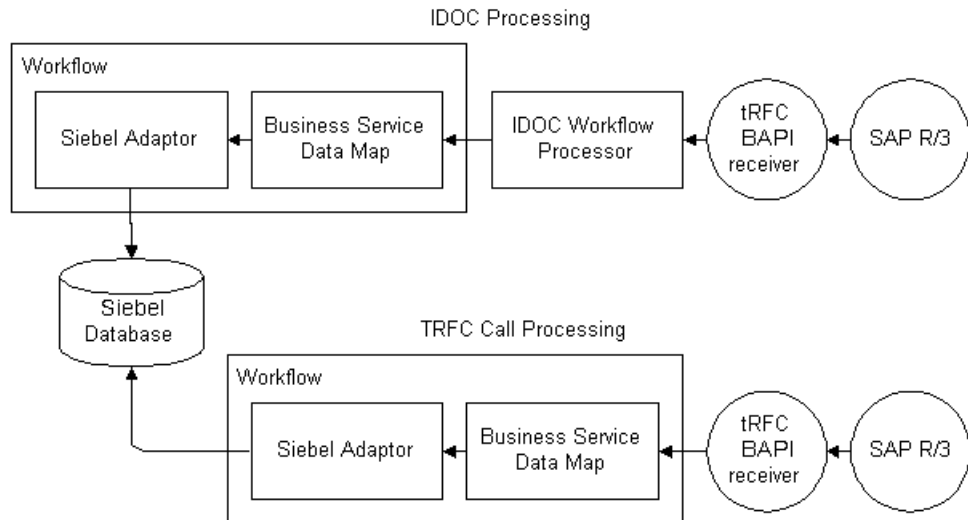
Figure 20. IDOC and tRFC Processing

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 6, "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 5, "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 From the Workflow Diagram Editor, drag a Business Service box to your diagram.

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 Edit the Name field.

5 From the drop-down list for Business Services choose the name of your new Business Service. If you are using a Siebel Data Mapper map, choose "EAI Data Transformation Engine".

6 From the drop-down list for Business Service Methods choose the Execute method for the map.

7 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 Business Service 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 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.

5 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).

> **NOTE:** When the tRFC BAPI receiver is running with the Siebel client, the component parameters must be set as follows:
> - In the client configuration file, if working with a Siebel Mobile client
> or
> - As input arguments on the input property set in the Business Service Simulator.

6 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 above. 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 *Siebel Tools Reference*. 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 *Siebel Tools Reference*.

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. Table 19 shows the structure of an input integration object.

Table 19. Input Integration Object Structure

SAP Function Call	Siebel BAPI Input Integration Object	
Import Parameter - scalar	Import Parameters Component	Field - scalar field name
Import Parameter - structure		Field - < SAP structure > - < SAP Field1 >
		Field - < SAP structure > - < SAP Field2 >
		Field - < SAP structure > - < SAP Fieldn >
Internal Table		SAP Table Component

For example, Table 20 shows BAPI BAPI_SALESORDER_CREATEFROMDAT1 having the following structure and corresponding Input Integration Object.

Table 20. Input Integration Object Structure Example

SAP Function Call	Siebel BAPI Input Integration Object	
Import Parameter - ORDER_HEADER_IN structure	Import Parameters Component	Field - ORDER_HEADER_IN-DOC_NUMBER Field - ORDER_HEADER_IN-DOC_TYPE Field - ORDER_HEADER_IN-n
Import Parameter - WITHOUT_COMMIT scalar		Field - WITHOUT_COMMIT
Import Parameter - CONVERT_PARVW_AUART scalar		Field - CONVERT_PARVW_AUART
Internal Table - ORDER_ITEMS_IN		ORDER_ITEMS_IN Component
Internal Table - ORDER_PARTNERS		ORDER_PARTNERS Component
Internal Table - ORDER_ITEMS_OUT		ORDER_ITEMS_OUT Component
Internal Table - ORDER_CFGS_REF		ORDER_CFGS_REF Component
Internal Table - ORDER_CFGS_INST		ORDER_CFGS_INST Component
Internal Table - ORDER_CFGS_PART_OF		ORDER_CFGS_PART_OF Component
Internal Table - ORDER_CFGS_VALUE		ORDER_CFGS_VALUE Component
Internal Table - ORDER_CCARD		ORDER_CCARD Component
Internal Table - ORDER_CFGS_BLOB		ORDER_CFGS_BLOB Component
Internal Table - ORDER_SCHEDULE_EX		ORDER_SCHEDULE_EX Component

Table 21 shows the structure of an output integration object.

Table 21. Output Integration Object Structure

SAP Function Call	Siebel BAPI Output Integration Object	
Export Parameter - scalar	Export Parameters Component	Field - scalar field name
Export Parameter - structure		Field - < SAP structure > - < SAP Field1 >
		Field - < SAP structure > - < SAP Field2
		Field - < SAP structure > - < SAP Fieldn >
Internal Table		SAP Table Component

For example, Table 22 shows BAPI BAPI_SALESORDER_CREATEFROMDAT1 having the following structure and corresponding Output Integration Object.

Table 22. Output Integration Object Structure Example

SAP Function Call	Siebel BAPI Output Integration Object	
Export Parameter - SALESDOCUMENT scalar	Export Parameters Component	Field - SALESDOCUMENT
Export Parameter - SOLD_TO_PARTY structure		Field - SOLD_TO_PARTY-SHIP_TO Field - SOLD_TO_PARTY-NAME Field - SOLD_TO_PARTY-n
Export Parameter - SHIP_TO_PARTY structure		Field - SHIP_TO_PARTY-SOLD_TO Field - SHIP_TO_PARTY-NAME Field - SHIP_TO_PARTY-n
Export Parameter - BILLING_PARTY structure		Field - BILLING_PARTY-PAYER Field - BILLING_PARTY-NAME Field - BILLING_PARTY-n
Export Parameter - RETURN structure		Field - RETURN-TYPE Field - RETURN-ID Field - RETURN-n
Internal Table - ORDER_ITEMS_IN		ORDER_ITEMS_IN Component
Internal Table - ORDER_PARTNERS		ORDER_PARTNERS Component

Table 22. Output Integration Object Structure Example

SAP Function Call	Siebel BAPI Output Integration Object	
Internal Table - ORDER_ITEMS_OUT		ORDER_ITEMS_OUT Component
Internal Table - ORDER_CFGS_REF		ORDER_CFGS_REF Component
Internal Table - ORDER_CFGS_INST		ORDER_CFGS_INST Component
Internal Table - ORDER_CFGS_PART_OF		ORDER_CFGS_PART_OF Component
Internal Table - ORDER_CFGS_VALUE		ORDER_CFGS_VALUE Component
Internal Table - ORDER_CCARD		ORDER_CCARD Component
Internal Table - ORDER_CFGS_BLOB		ORDER_CFGS_BLOB Component
Internal Table - ORDER_SCHEDULE_EX		ORDER_SCHEDULE_EX Component

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

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 164.

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 123 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 94.

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 23.

Table 23. Business Service Names

Reference Name	Business Service Name
tRFC BAPI Adapter	`EAI SAP BAPI Adapter (tRFC)`
tRFC BAPI Receiver	`EAI SAP BAPI Receiver (tRFC)`

Table 23. Business Service Names

Reference Name	Business Service Name
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 Server 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 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 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 24 contains User Properties, Component Parameters, and Input Method Arguments for the Execute Method.

Table 24. 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.
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

Table 24. BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
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)
SAPRfcMaxConnectTime	X			Integer value in seconds	Used only if DisconnectAlways 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 24. BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPCodepage	X	X	X	Value from Transcode Encoding picklist	Siebel Codepage name for SAP codepage.
SAPIgnoreCharSetConvE rrors	X	X	X	True or False	Default is False. If True, codepage 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 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 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 52. 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 24 on page 155: 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 26 summarizes the functionality.

Table 26. BAPI Adapter Error Handling Summary

SAPAutoError	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 all non-blank fields in the structure containing the SAPErrorTypeField are written within the error message. Workflow stops at the BAPI adapter step.

Table 26. BAPI Adapter Error Handling Summary

SAPAutoError	SAPErrorTypeField	Behavior
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 all 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 contain the values of all 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 contain the values of all 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 137. 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 7, "EAI Queue."

tRFC BAPI Adapter Configuration

The `EAI SAP BAPI adapter (tRFC)` business service can be invoked from a workflow. Read Chapter 4, "Customizing Integrations" for more information. Table 27 and Table 28 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 27 contains User Properties, Component Parameters, and Input Method Arguments for the Execute Method.

Table 27. tRFC BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPRfcConnectString		X	X		Refer to Table 24
SAPRfcUserName		X	X		Refer to Table 24
SAPRfcPassword		X	X		Refer to Table 24
SAPRfcTrace		X	X	"true" or "false"	Refer to Table 24
SiebelMessage			X		Refer to Table 24
DisconnectAlways	X			"true" or "false"	Refer to Table 24
SAPRfcMaxConnectTime	X			Integer value in seconds	Refer to Table 24
tRFCMode	X			"tRFC"	If value is "tRFC", indicates that this adapter is working with tRFC.
SAPWriteXML	X	X		"WriteNone", "WriteAlways", "WriteBeforeErr", "WriteOnErr"	Indicates usage of EAI Queue. For more information, read Chapter 7, "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.

Table 27. tRFC BAPI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
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 picklist	Siebel Codepage name for SAP codepage.
SAPIgnoreCharSetConvErrors	X	X	X	True or False	Default is False. If True, codepage conversion errors are handled by substitution of the question mark (?) character in place of an unrecognized character.

Table 28 contains Output Method Arguments for the Execute Method.

Table 28. 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 155.

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 137. 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 7, "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, ALL 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 alternatively to 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 mentioned above.

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, by calling 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 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 180.

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 179.

Table 29 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 29. 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.

Table 29. tRFC BAPI Receiver Configuration Options

Name	User Prop	Comp Param	Valid Values	Usage
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 7, "EAI Queue."
SAPXMLQueueCleanup	X		"true" or "false"	If true, entries are deleted from the EAI Queue upon completion. Read Chapter 7, "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 7, "EAI Queue."
SAPXMLQueueName	X		EAI Queue name	Provides the name of the EAI queue being used to store queue entries created by this service. Read Chapter 7, "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 picklist	Siebel Codepage name for SAP codepage.
SAPIgnoreCharSetConvErrors	X	X	True or False	Default is False. If True, codepage 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.

IDOC Interfaces 6

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 5, "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 30 shows part of the DEBMAS02 IDOC Type and how this would be represented by a Siebel IDOC integration object.

Table 30. 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 31 lists all the control record fields and indicates if they are used in a 3X or 4X control record. The "User can populate?" column indicates if you can set the value of the field from a Business Service Data Map for input to SAP. The Mandatory column indicates if 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 31. 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.

Table 31. Control Record Fields

4X Field	3X Field	User Can Populate?	Mandatory	IDOC Adapter Action
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 171.
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.

Creating SAP Integration Objects with the IDOC Wizard

Table 31. Control Record Fields

4X Field	3X Field	User Can Populate?	Mandatory	IDOC Adapter Action
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		
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 179.

Table 32 summarizes the IDOC integration object user properties.

Table 32. 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 33.

Table 33. 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 4.6B, you can retrieve a 3.1H IDOC by setting the SAPIdocVersion to "31H". If this was not set, a 4.6B 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 33.

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 Server 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 137.

Sending IDOCs to SAP

As explained in Chapter 4, "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 34 and Table 35 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 34 contains User Properties, Component Parameters and Input Method Arguments for the Send Method.

Table 34. 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 31 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.
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 37.
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.

Table 34. IDOC Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
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", "WriteBeforeErr"	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.
SAPCodepage	X	X	X	Value from Transcode Encoding picklist	Siebel Codepage name for SAP codepage.
SAPIgnoreCharSetConvErrors	X	X	X	True or False	Default is False. If True, codepage 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 35 contains output method arguments for the Send Method.

Table 35. 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 7, "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 36 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 36. 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 picklist	Siebel Codepage name for SAP codepage.
SAPIgnoreCharSetConvErrors	X	X	X	True or False	Default is False. If True, codepage 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

Once 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 37 contains User Properties, Component Parameters, and Input Method Arguments for the Send Method.

Table 37. 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 37.
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.

Table 37. IDOC MQ AMI Adapter Configuration Options

Name	User Prop	Comp Param	Method Arg	Valid Values	Usage
SAPCodepage	X	X	X	Value from Transcode Encoding picklist	Siebel Codepage name for SAP codepage.
SAPIgnoreCharSetConvErrors	X	X	X	True or False	Default is False. If True, codepage 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 "Send4xIdoc_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`. Set IsReceivingTextData to false to allow binary data passing. 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 38 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 38. 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 picklist	Siebel Codepage name for SAP codepage.
SAPIgnoreCharSetConvErrors	X	X		True or False	Default is False. If True, codepage 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

Once 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 will be passed must be defined. For information on how to define the workflow name, read "Receiving IDOCs from SAP" on page 179.

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 54. 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.

EAI Queue 7

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.

EAI Queue Overview

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 Integration Administration > 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 21 shows the current implementation of the EAI Queue when making tRFC calls from the Siebel application to SAP R/3.
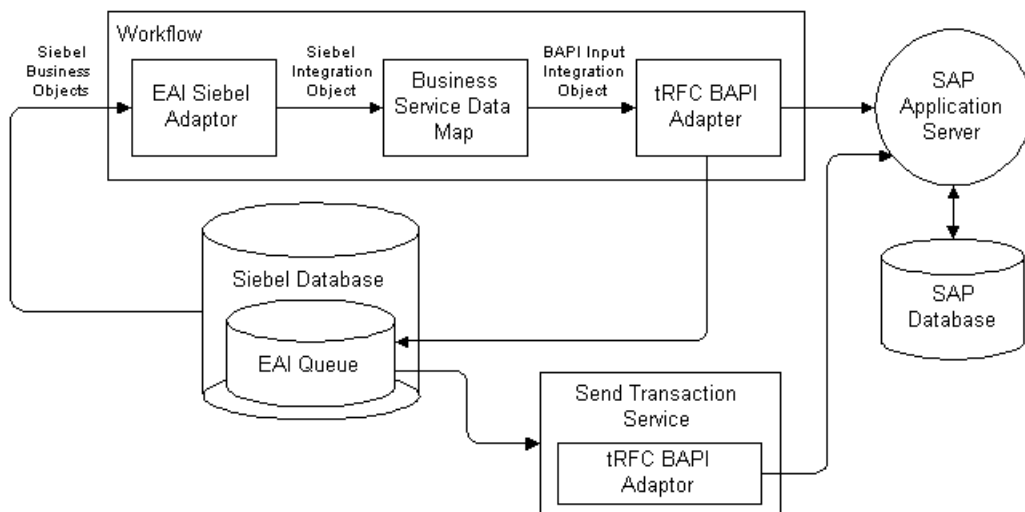
Figure 21. Outbound from Siebel Using tRFC

Figure 21 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 39 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 above modes.

Table 39. 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		

Table 39. 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)
WriteBeforeErr	Success	Yes (Initial)	Yes	Yes	Yes (Initial or Sent)
				No	Yes (Confirmed)
	Fail	No	No		

Inbound to the Siebel Application

Figure 22 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.
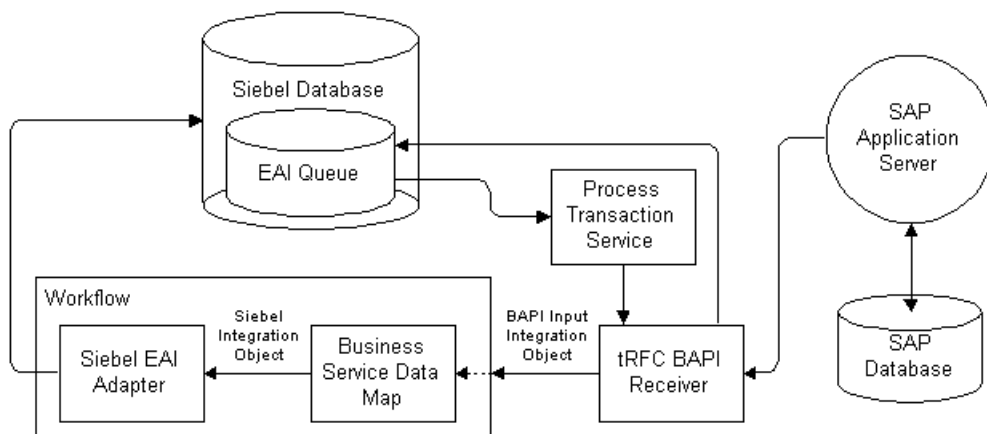
Figure 22. Inbound to the Siebel Application Using tRFC

Figure 22 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 40 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 above modes.

Table 40. 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

Table 40. 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
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 23 shows the current implementation of the EAI Queue when making tRFC calls from Siebel 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.
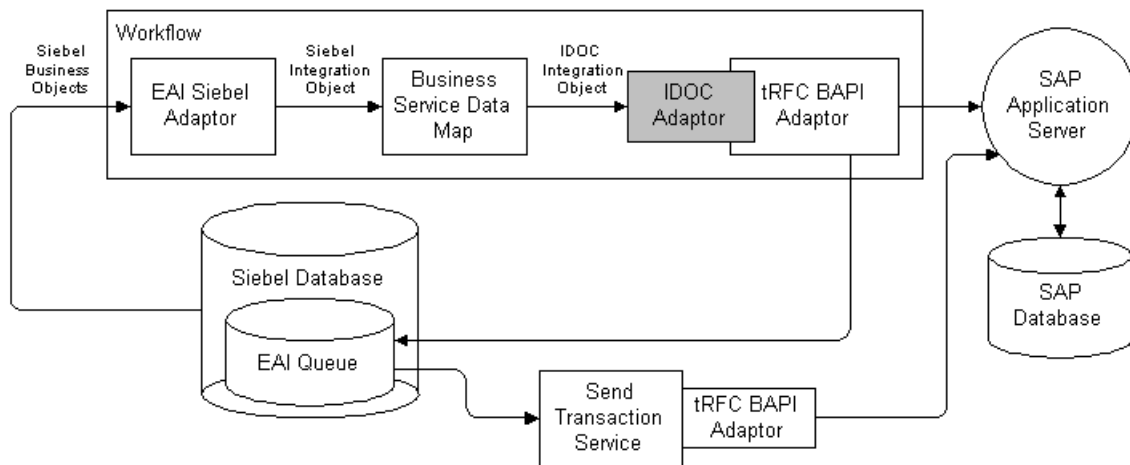
Figure 23. Outbound from the Siebel Application Through ALE

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 190.

Inbound to the Siebel Application

Figure 24 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.
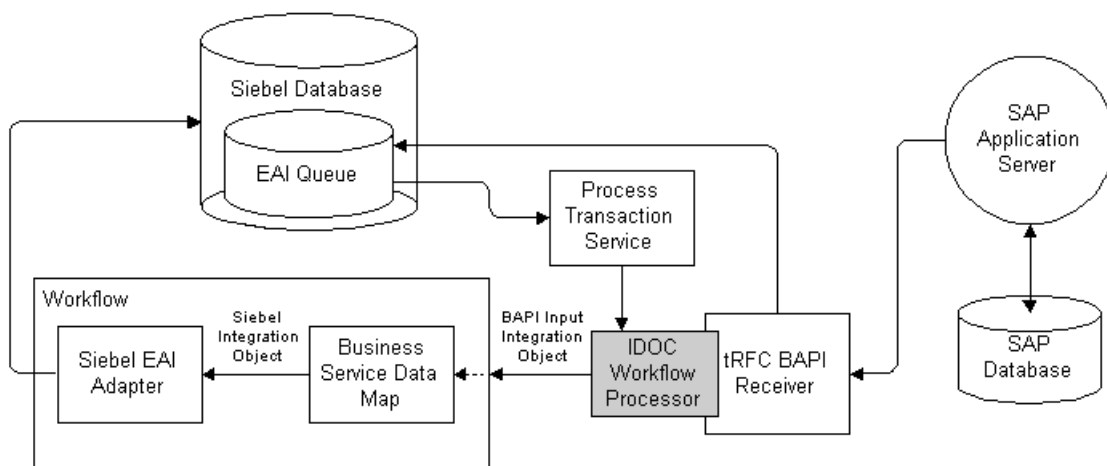
Figure 24. Inbound to the Siebel Application Using ALE

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 190.

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, or 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 .

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, or WriteOnErr.

2 Start a BAPIRcvr component. For more information, read "Starting the tRFC BAPI Receiver" on page 54.

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 202.

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 175.

Table 41 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 SAPSendTrans component.

Table 41. Send Transaction Service Configuration Options

Name	User Prop	Comp Param	Valid Values	Usage
SAPWaitTime	X	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	X	Integer value	Number of times to try to reconnect to SAP before erring the transaction, if SAP is unreachable.
SAPWakeupTime	X	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.

Table 41. Send Transaction Service Configuration Options

Name	User Prop	Comp Param	Valid Values	Usage
SAPtRFCService	X	X		Business service that 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 Send service.
SiebelWaitTime	X	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 24 on page 155.
SAPRfcUserName		X		Refer to Table 24 on page 155.
SAPRfcPassword		X		Refer to Table 24 on page 155.

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 54. 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 175.

Table 42 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 42. 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	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 54. 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" on page 204

- "DeleteMessage" on page 205

- "GetMessage" on page 206

- "GetStatus" on page 208

- "UpdateStatus" on page 209

AddMessage

The AddMessage method creates an XML file with the given Siebel Message and inserts a status row into the status table. Table 43 summarizes the AddMessage arguments.

Table 43. 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

Table 43. AddMessage Arguments

Argument Name	Input/Output	Required?	Argument Type	Description
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 44 summarizes the DeleteMessage arguments.

Table 44. DeleteMessage Arguments

Argument Name	Input/Output	Required?	Argument Type	Description
RowID	Input	No (see comment below)	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 comment below)	String	Search criteria.

Table 44. DeleteMessage Arguments

Argument Name	Input/Output	Required?	Argument Type	Description
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 45 summarizes the GetMessage arguments.

Table 45. GetMessage Arguments

Argument Name	Input/Output	Required?	Argument Type	Description
RowID	Input/output	No (see comment below)	String	Row ID for Queue Items Table.
QueueName	Input	Yes	String	Queue name.
SearchSpec	Input	No (see comment below)	String	Search criteria.
SortSpec	Input	No	String	Sort criteria.

Table 45. GetMessage Arguments

Argument Name	Input/Output	Required?	Argument Type	Description
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 46 summarizes the GetStatus arguments.

Table 46. GetStatus Arguments

Argument Name	Input/Output	Required?	Argument Type	Description
RowID	Input/output	No (see comment below)	String	Row ID for Queue Items Table.
QueueName	Input	Yes	String	Type for status table.
SearchSpec	Input	No (see comment below)	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 47 summarizes the UpdateStatus arguments.

Table 47. UpdateStatus Arguments

Argument Name	Input/Output	Required	Argument Type	Description
RowID	Input/output	No (see comment below)	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 comment below)	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.

Upgrading from v6.x to v7.x 8

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.

6.x Data Flows Modifications in Siebel 7

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.

Upgrading IDOC Integration Objects

You do not need to change existing BAPI integration objects.

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 Create a new IDOC integration object with the original name of your old IDOC integration object using the EAI SAP IDOC Adapter Wizard. 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.

NOTE: The Product - Receive SAP Material (IDOC Input) and Account - Receive SAP Customer (IDOC Input) integration objects have already been updated for you in version 7.5. They can now be used by the BAPI Receiver instead of the 6.x IDOC Receiver.

Switching the Service for Sending IDOCs to SAP for Workflows

You do not need to change workflows that invoke the BAPI adapter.

If you are receiving IDOCs from SAP, you do not need to modify your workflow.

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.

To switch to the new service safely

1 Revise your workflow using the workflow diagram editor.

> **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). Create a new business service to replace this one.

3 Double click on the newly created business service.

The editor displays the input and output arguments for this business service.

4 Make note of how these are used, because you need to add them to the new business service.

5 Navigate back to the diagram editor.

6 Drag a new business service box to your diagram in the vicinity of the old business service.

7 Double click on this new business service box and select the name of the service, EAI SAP IDOC Adapter, and the method Send.

8 Define the input and output method arguments in the new service as they were defined in your old service. These have not changed.

9 Navigate back to the diagram editor.

10 Modify the flow of the workflow diagram to use the new service you have created.

11 Test your workflow using the workflow simulator. Follow the suggestions for testing in .

Business Service Data Maps Using the MessageBox Function

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.

Business Service Data Maps Using the MessageBox Function

Data Types Used in BAPIs and IDOCs   A

This appendix covers data types used in BAPIs and IDOCs.

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 will 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 will 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.

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.

SAP R/3 v3.1H Mappings   B

The object mappings and ALE configurations described in this appendix are based on SAP R/3 version 3.1H and Siebel Connector for SAP R/3 version 6.x with settings in English on the Oracle/Windows NT platform. If you are running a different version of SAP, or you are running R/3 in a different language or platform, refer to your Systems Integrator for assistance in configuring your implementation.

Siebel Connector v6.x Workflow Process Mappings

The maps listed in this appendix are implemented as and supported by workflow processes. These standard workflow processes are visible in your Siebel application in the All Workflow Processes view. Table 48 shows the workflows that implement particular integration objects. These workflow processes are for SAP R/3 version 3.1H only.

Table 48. Workflows and Integration Objects

Workflow	Integration Object
Account - Get SAP Order List	Account - Get SAP Order List (Siebel)
	Account - Get SAP Order List (BAPI Input)
	Account - Get SAP Order List (BAPI Output)
Account - Import SAP Order	Account - Import SAP Order (Get SAP Order List)
	Account - Import SAP Order (Siebel Order)
Account - Receive SAP Customer	Account - Receive SAP Customer (IDOC Input)
	Account - Receive SAP Customer (Siebel)

Table 48. Workflows and Integration Objects

Workflow	Integration Object
Order - Create SAP Order	Order - Create SAP Order (Siebel)
	Order - Create SAP Order (BAPI Input)
	Order - Create SAP Order (BAPI Output)
Order - Get SAP Order Status	Order - Get SAP Order Status (Siebel)
	Order - Get SAP Order Status (BAPI Input)
	Order - Get SAP Order Status (BAPI Output)
Product - Receive SAP Material	Product - Receive SAP Material (IDOC Input)
	Product - Receive SAP Material (Siebel)

Some compound workflows call other workflows to use the functionality already provided. Table 49 lists the workflows and the primary workflows that they call.

Table 49. Compound Workflows

Compound Workflow	Calls
Account - Import SAP Order and Get SAP Order Status	Account - Import SAP Order
	Order - Get SAP Order Status
Order - Create SAP Order and Get SAP Order Status	Order - Create SAP Order
	Order - Get SAP Order Status
SAP IDOC Processor	Product - Receive SAP Material
	Account - Receive SAP Customer

Locating the SAP R/3 v3.1H Object Mapping Tables

Due to the size of the object mapping tables, the SAP R/3 v3.1H object mappings are located on the Bookshelf for Siebel 7.5.x eBusiness Applications.

To locate the SAP R/3 v.3.1H Object 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 *Connector for SAP R/3* in the Application Integration category.

4 To access the SAP R/3 v3.1H object mappings, click SAP R/3-Maps-V31H.

SAP Field Mappings   C

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 94 in Chapter 4, "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 Bookshelf for Siebel 7.5.x eBusiness Applications.

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, click SAP R/3-Maps-Account.

- For sales order mappings, select one of the following maps:

  ❏ SAP R/3-Maps-SalesOrder (Order Create Mappings)

  ❏ SAP R/3-Maps-SalesOrder (Order Get Status Mappings)

  ❏ SAP R/3-Maps-SalesOrder (Order Validate Mappings)

  ❏ SAP R/3-Maps-SalesOrder (Quote to Order [Submit] Mappings)

  ❏ SAP R/3-Maps-SalesOrder (Quote to Order [Validate] Mappings)

- For product-material mappings, click SAP R/3-Maps-Product.

Troubleshooting D

The troubleshooting information is organized in the following categories:

■ Debugging Tips for Workflows

■ SAP Connection Problems

■ SAP Debugging Options

■ SAP Configuration Problems

Debugging Tips for Workflows

The following tips apply to troubleshooting workflows.

Problem: 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.

Solution: 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.

Problem: When transferring data to or from SAP through a BAPI interface, the decimal position in a numeric field is shifted.

Solution: You may need to change External Scale for the field in the integration object. Refer to "Modifying External Scale Values" on page 131.

SAP Connection Problems

Use the following tips to troubleshoot SAP connection problems.

Problem: Error message received: 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'

Solution: User name, password, or client number is incorrect. Check values in SAPRfcUserName, SAPRfcPassword, and SAPRfcConnectString.

Problem: Error message received: 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'

Solution: 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.

Problem: Either of the following error messages appears.

■ "The DLL 'SSCAEIRF.DLL' could not be loaded."

■ "The DLL 'SSCAEIIR.DLL' could not be loaded."

Solution: Librfc32.dll version 4.6B, 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 4.6B. 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 46B. To check the Product Version, select the file in Windows Explorer, right-click, and choose Properties. Then select the Version tab and the Product Version item in the list box.

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.

SAP Configuration Problems

Use the following tips to troubleshoot SAP configuration problems.

Problem: When sending IDOCs from SAP-to-Siebel, no communication IDOCs are sent. You may see the message in SAP "No communication IDOCs have been selected".

Solution: Verify that you have added your IDOC message type to the Distribution Model under SAP transaction SALE as defined in "Distributing Logical Systems" on page 39.

Problem: When sending IDOCs from SAP to the Siebel application, no IDOCs are being received by the Siebel tRFC BAPI Receiver.

Solution: Check the following:

1 Verify that your tRFC BAPI Receiver is connected with SAP Transaction SM59. Select the TCP/IP RFC destination you created in Chapter 2, "Installing and Configuring the SAP Connector" 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. In particular, verify that the program ID in the saprfc.ini file matches the program ID name in the RFC destination.

2 Once you have verified that your receiver is connected with SM59, go to SAP Transaction WE02 or WE05 and verify that the IDOC has been created and there are no errors in its transmission. You may find one of the following conditions:

a Errors in the status record - specific application errors may have occurred. These would need to be corrected within SAP. The IDOC would need to be deleted or resent. BD87 can be used for manual resend.

b 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.

c IDOC with status 29: Error in ALE service - Your Partner Profile may be missing an outbound parameter record for your message type.

d IDOC with status 03: Data passed to port OK - It appears to be OK, but 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 (see below).

3 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. Entries can exist here for the following reasons:

a A network problem occurred or the receiver was not connected at the time the IDOC was sent. These transactions can be resent.

b 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.

c Depending upon the load, some transactions may time out. These can be resent.

d 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.

There are three ways to resend transactions from the tRFC queue:

e Manually using SM58 - a specific transaction may be selected and resent with Edit- > Execute LUW

f Through a scheduled background job. This is the preferred method. You can schedule RSARFCCP to resend all 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.

g 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 tie up all 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:

i   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.

ii   Connection attempts up to task
This specifies the total number of attempts SAP makes to resend this transaction.

iii   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 (ii) is 5 and the value for (iii) 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 (ii) and (iii) 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.

Problem: IDOCs you are sending into SAP from the Siebel application are not being processed.

Solution: There may be many reasons. Start by looking for the IDOC in SAP using transaction WE02 or WE05.

1 If the IDOC is not there, the problem may be that you have not set the Sender Partner Number for the IDOC. This is set with the SAPSenderPtnrNum component parameter or method argument.

2 If the IDOC is there, check the status record for error information.

a 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.

b 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.

Creating Integration Touch Points   E

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:

■ "Creating New Integration Touch Points" on page 235

■ "Sample Business Services" on page 238

Creating New Integration Touch Points

Use the following process to create a new integration touch point.

1 "Identify the SAP Object (IDOC/BAPI)"

2 "Identify the Siebel Business Object" on page 236

3 "Create BAPI Integration Objects" on page 237

4 "Create an Integration Object for the Siebel Business Object" on page 237

5 "Create Business Services for Mapping" on page 237

6 "Create the Workflow" on page 237

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 4, "Customizing Integrations," documents all 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 238. *Overview: Siebel eBusiness Application Integration Volume I* covers the data transformation business services. All 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 50.

Table 50. Workflow Process Properties

Process Property	Value
Input: BAPI Int Object Name	BAPI_MATERIAL_AVAILABILITY (Input)
Input: Map Name	GetSAPMaterialATP_SiebelToBAPI

Table 50. Workflow Process Properties

Process Property	Value
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