Going Live with Siebel Business Applications

Version 7.8, Rev. A
May 2005

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

Contents

1 What's New in This Release

What's New in Going Live with Siebel Business Applications, Version 7.8, Rev. A

Table 1 lists changes described in this version of the documentation to support release 7.8 of the software.

Table 1. What's New in Going Live with Siebel Business Applications, Version 7.8, Rev. A

Topic	Description
"About ADM Deployment Filters" on page 25	Added information on filtering child business components.
"Creating ADM Deployment Projects" on page 33	Modified note in procedure regarding viewing child data type records.

What's New in Going Live with Siebel Business Applications, Version 7.8

Table 2 lists changes described in this version of the documentation to support release 7.8 of the software.

Table 2. What's New in Going Live with Siebel Business Applications, Version 7.8

Topic	Description
"Preparing to Run the Repository Migration Configuration Utility" on page 10	Added this new section on preparing to run the repository migration configuration utility. This section contains information on new repository migration features.
"Running the Repository Migration Configuration Utility" on page 12	Updated the running the repository migration configuration utility procedures for Windows and UNIX.
"About Application Deployment Manager (ADM)" on page 23	Added new data types migrated by Application Deployment Manager (ADM).
Chapter 4, "Migrating Parameters Between Environments"	Added this new chapter, which describes migrating parameter values between environments using the cfgmerge utility. This utility is a new feature introduced in 7.8.
"Customizing Siebel.ini Files" on page 61	Added this section and subsections describing how to customize Siebel.ini files.

2 Migrating Repositories Between Databases

Migrating repositories between databases is a common requirement when going live to a new development, testing, or production environment. This chapter defines and describes the process for migrating repositories.

Part of the process of going live to a new environment may also include a migration of new run-time data customizations. If necessary, perform this task after migrating repository changes. See Chapter 3, "Migrating Customizations Between Applications" for information on this process.

This chapter includes the following topics:

About Migrating Repositories Between Databases

Migrate the repository and application customizations between databases so the database schema for the user data, the business objects, and the user interface remain synchronized. Populate the test database, and when sufficient testing has taken place, migrate the repository and update the target database schema. For information on migrating application customizations, see "Migrating Customizations Between Applications" on page 23.

For background information on Siebel repositories and schemas, see *Using Siebel Tools*.

For information about setting up your system and database environment, see the *Siebel Installation Guide* for the operating system you are using.

Note the following restrictions and recommendations when migrating repositories between databases:

- Isolate development and test environments from the production environment.
- Do not migrate repository customizations to your target environment until you have completed testing to verify that the customizations work correctly and meet your business requirements.
- Do not migrate repositories between different versions of Siebel applications, as this action leads to an inconsistent environment.

Use the configuration utility, as described in *Using Siebel Tools*, to replace the object definitions. Then distribute a new Siebel repository file (SRF) to the servers and any remote clients. For customizations involving schema changes, you need to use the configuration utility, which:

- Updates the data in the target Siebel Server to the new schema.
- Updates the repository object definitions.

All mobile users need to synchronize prior to the schema update and (if not using Siebel Anywhere) reextract following the upgrade.

If you are using Siebel Anywhere, mobile users need to synchronize the next time they log on to their local database after the migration has occurred. Synchronizing will download new schema changes from the Siebel Server to the mobile user's local database. If they do not synchronize, there will be a mismatch between the local database and the server database. For further information on Siebel Anywhere, see *Siebel Anywhere Administration Guide*.

NOTE: Importing a repository and then synchronizing the schema definition in a target system is equivalent to migrating the repository. However, after importing the repository and before synchronizing the schema definition, it is necessary to rename the older repository to a temporary name and give the imported repository the correct name. For further information on repository importing and exporting, see *Using Siebel Tools*.

After the repository migration, perform any post-repository migration tasks, if necessary. See "Post-Repository Migration Tasks" on page 20 for information on these tasks.

Process for Migrating Repositories

This topic lists the ordered tasks of the process for migrating repositories between databases. For background information on this process, see "About Migrating Repositories Between Databases" on page 7.

NOTE: A typical migration session requires the deployment of the new repository and the new non-repository customizations. To deploy (migrate) the repository, use the following process. To migrate non-repository customizations such as views, responsibilities, and lists of values, see "Post-Repository Migration Tasks" on page 20.

To migrate repositories between databases perform the following tasks:

1. Check in all projects—in both the source and target databases. For information on projects and checking in projects, see *Using Siebel Tools*.

   If you migrate a database schema with some projects still checked out, the migration will work but the project state will be not locked in the target database.

2. Prepare the target database for the new repository. See "Preparing the Target Database for the New Repository" on page 9 for information on this task.

3. Prepare to run the repository migration configuration utility. See "Preparing to Run the Repository Migration Configuration Utility" on page 10 for information on this task.

4. Run the repository migration configuration utility. See "Running the Repository Migration Configuration Utility" on page 12 for information on this task.

5. Upgrade mobile databases that are dependent on the target database. See "Upgrading Mobile Databases" on page 20 for information on this task.

Preparing the Target Database for the New Repository

This topic describes the task of preparing the target database for the new repository. This task is a part of the "Process for Migrating Repositories" on page 8. For background information on migrating repositories, see "About Migrating Repositories Between Databases" on page 7.

Complete the following actions before you migrate the repository to the target database:

- Make sure that all mobile users perform a full synchronization to avoid any unexpected issues in the target environment as a result of database schema changes made to the new repository.
- The target enterprise can be online for a portion of the repository migration procedure. (For more information on this feature, see "Preparing for Target Environment Status Selection" on page 11.) However, if you do not select this option, stop all Siebel Server tasks and disconnect all database access until migration has been successfully executed.

  NOTE: All connected users (including the database administrator) must disconnect at one point during the running of the repository migration configuration utility.
- Do a full backup of the target database after all mobile user transactions have been merged.
- Make sure that the target database configuration meets the database requirements outlined in the *Siebel Installation Guide* for the operating system you are using.
- Do not change ODBC parameters or settings for datasources created by the Siebel Server installation.
- Verify the names of all repositories in the target database.

  You will later choose a new name that the repository being migrated will have in the target database. It is recommended that you keep the name of your target repository constant. Accordingly, rename the existing target repository to show that it has been superseded. You will also later import a repository, to which you should give the standard name for your target repository. For further information on repository naming guidelines, see *Using Siebel Tools*.
- Update database statistics if warranted for your database.
- Delete older repositories from the target database. This procedure may run for some time. For further information on this task, see *Using Siebel Tools*.

NOTE: The repository migration process references two ODBC data sources. One is the source database and the other is the target database. When running the migration task from the source system, it is necessary to create an ODBC data source for the target database manually. This applies to all server and database platforms. For further information on verifying the ODBC data source, see the *Siebel Installation Guide* for the operating system you are using.

Preparing to Run the Repository Migration Configuration Utility

The repository migration process differs based on several selections made during the running of the repository migration configuration utility. Depending on your selections, some preparatory tasks may be required before running this utility.

This task is a part of the "Process for Migrating Repositories" on page 8. For background information on migrating repositories, see "About Migrating Repositories Between Databases" on page 7.

The three repository migration selections of note are:

- **Source Repository Selection.** For information on this selection, see "Preparing for Source Repository Selection" on page 10.
- **Target Environment Status.** For information on this selection, see "Preparing for Target Environment Status Selection" on page 11.
- **Schema Changes Selection.** For information on this selection, see "Preparing for Schema Changes Selection" on page 11.

See Figure 1 for a high-level view of the repository migration utility screen flow based on the different source repository selections.

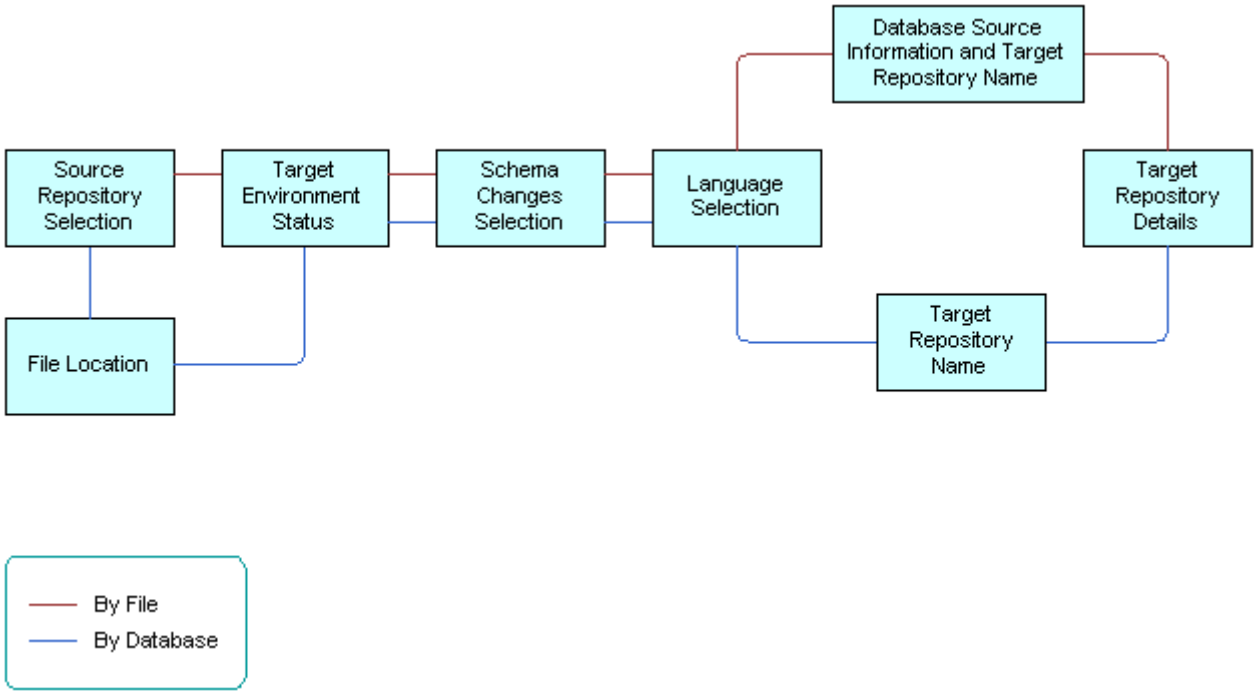


Figure 1. High-level View of Repository Migration by Source Repository Selection

Preparing for Source Repository Selection

The source repository selection allows the administrator to migrate the repository:

- By creating a source repository file (Recommended)

  If using this source repository migration selection, create a source repository migration file before running the repository migration utility. For information on this task, see the following section, "Creating a Source Repository Migration File" on page 12.

- Directly from the source database

  If using this source repository migration selection, no preparations are required.

Preparing for Target Environment Status Selection

The target environment status selection allows the administrator the option to import the repository while the target Siebel application is online, which lessens the downtime of the target environment.

To use this option, select the following at the Target Environment Status screen when running the repository migration configuration utility:

- The Target Enterprise will be online when the migration starts.

Selecting this option allows the administrator to run a repository migration process—a portion of which does not require the target environment to be offline—until it is absolutely necessary that the target environment be shutdown. At that point during the repository migration process, a dialog box appears informing the administrator to shut down the target environment for the remainder of the migration.

NOTE: If you are planning to run the repository migration unattended, for example, by script, do not select the online option.

Preparing for Schema Changes Selection

The schema changes selection allows the administrator the option not to run the schema migration portion of the repository migration process, which lessens the amount of time required for the migration process.

To use this option, select the following option at the Schema Changes screen when running the repository migration configuration utility:

- There are no new schema changes.

Selecting this option allows the administrator to skip the schema upgrade portion of the repository migration process if there are no changes to apply to the target's physical or logical schema. Examples of schema updates include the addition of an extension column to an existing table or the addition of a new extension table.

If in doubt, select the option;

- There are new schema changes to be applied.

Creating a Source Repository Migration File

Use the database configuration utility to create a source repository migration file by clicking the Import/Export Repository selection at the appropriate screen. For complete details on this task for Windows and UNIX, see *Using Siebel Tools*.

For further information on other source repository migration preparations, see the previous section,"Preparing to Run the Repository Migration Configuration Utility" on page 10.

Running the Repository Migration Configuration Utility

This topic describes the task of running the repository migration configuration utility. This task is a part of the "Process for Migrating Repositories" on page 8. For background information on migrating repositories, see "About Migrating Repositories Between Databases" on page 7.

Running the configuration utility achieves the following results:

- Exports the designated repository from the source database or reads the source repository from file.
- Imports the designated repository into the target database
- Exports the logical schema definition from the specified repository to a control file
- Synchronizes the physical schema of the target database with this logical schema definition

NOTE: The repository migration configuration utility skips the previous two schema-related steps if you select no schema changes while running the utility.

Take note of the following items before running a repository migration:

- When migrating repositories over a wide area network (WAN) and running the repository migration utility from the target environment, only the process of exporting the source repository to a flat file takes place on the WAN. All other processing takes place on the local area network (LAN) of the target environment. Consider the export processing times when processing the export over a WAN. The most efficient option is to export the file on the source then transport the exported file to the target environment.
- For multilingual installations where the source respiratory contains additional language strings, all languages are copied with the repository to the target environment.
- Siebel Business Applications version 7.x do not support customized database triggers. If you have created customized triggers on your Siebel base tables, you must disable them before migrating the schema. You will then need to recreate the triggers after the migration is finished.

NOTE: If you have a custom table space defined on DB2 UDB for OS/390, the Database Server Configuration Utility used in the migration process is tablespace-aware.

To migrate a repository on Windows

1 Launch the Database Server Configuration Utility by choosing Start > Programs > Siebel Enterprise Server *version_number* > Configure DB Server.

The Gateway Server Address screen appears.

2 Specify your Siebel Gateway Name Server Address and Enterprise Server Name and click Next.

The Siebel Server Directory screen appears.

3 In the Siebel Server Directory screen, either accept the default value or choose the Browse button to select a directory, and then click Next.

The Siebel Database Server Directory screen appears.

4 In the Siebel Database Server Directory screen, either accept the default value or choose the Browse button to select a directory, and then click Next.

The RDBMS Platform screen appears.

5 In the RDBMS Platform screen, select the platform for your environment and then click Next.

The Siebel Database Operations screen appears.

6 In the Siebel Database Operations screen, select Migrate Repository from the list of operations, and then click Next.

The Source Repository Selection screen appears.

7 In the Source Repository Selection screen, select to read the source repository from the database or from a previously exported file.

- If you select to read the source repository from file, the Repository File Selection screen appears.

  Enter the path and filename of the source repository file and click Next. The Target Environment Status screen appears.

  For background information on creating a source repository file, see"Creating a Source Repository Migration File" on page 12.

- If you select to read the source repository directly from the database, the Target Environment Status screen appears.

8 In the Target Environment Status screen, select whether the target Siebel enterprise will be online or offline, and click Next. For further information on this selection, see "Preparing for Target Environment Status Selection" on page 11.

9 In the Schema Changes screen, select whether there are new schema changes to be applied during the repository migration, and click Next. For further information on this selection, see "Preparing for Schema Changes Selection" on page 11.

10 In the Language Selection screen, select the appropriate language, and then click Next.

NOTE: The repository migration process migrates all language strings within the repository. Select the base language installation that applies for this screen.

11 The next succession of screens are dependent on the repository source selection:

- If you chose to read the source repository from file, the Target Repository Name screen appears. Enter the target repository name, and click next.
- If you chose to read the source directly from database, the following succession of screens appears to define the source database.

Screen	Description
ODBC Data Source Name	If you chose to read the source repository from the database, enter the ODBC data source name.
Database User Name	If you chose to read the source repository from the database, enter the source database user name and source database password.
Database Table Owner	If you chose to read the source repository from the database, enter the source database table owner and source table owner password.
Source Database Repository Name	If you chose to read the source repository from the database, enter the database repository name and the target database repository name.

12 Progress by completing the information in the following screens, and then clicking Next.

Screen	Description
Target RDBMS Platform	Select the target RDBMS platform. IBM DB2 UDB v7.1 is the default.
Target Database Encoding	Select whether the target database is Unicode. Consult with your DBA on the setup of the target database.
Target Database ODBC Datasource	Enter the target database ODBC datasource.
Target Database User Name	Enter the target database user name and password.
Target Database Table Owner	Enter the target database table owner and table owner password.
Index Table Space Name (DB2-specific)	If you choose IBM DB2 for the target database platform, you get this screen. Enter the index table space name and 4-KB Table Space Name.

Screen	Description
16-KB Table Space Name (DB2-specific)	If you choose IBM DB2 for the target database platform, you get this screen. Enter the 16-KB table space name and 32-KB table space name.
Index Table Space Name (Oracle-specific)	If you choose Oracle as the target database platform, you have only two questions about index and table space. There are no 4-KB, 16-KB, and 32-KB tablespaces in Oracle. For Microsoft SQL server, there are no screens about tablespaces.

13 After the last screen in the previous step, a dialog box appears prompting you to run the repository migration now or at a later date. Click OK to begin the repository migration process, and CANCEL to run the migration process at a later date.

- If you select OK, a Configuration Parameter Review screen appears. Review this screen to check your configurations.
- If you select CANCEL, run the migration process at a later date by navigating to the bin subfolder of the Siebel Server root directory and entering the following command:

  ```
  Siebupg /m master_dev2prod.ucf
  ```

To migrate the repository on UNIX

1 Source environment variables from *$SIEBEL_ROOT* by typing:

```
source siebenv.sh
```

or

```
source siebenv.csh
```

2 Set the following environment variables:

- *SIEBEL_ROOT* should be the path of your Siebel Business Application installation directory.
- LANGUAGE should be set to the language in which the Configuration Wizard prompts appear; for example, ENU for U.S. English.

If either of these values is incorrect or empty, reset them using one of the following commands:

- setenv LANGUAGE *language_code* (where *language_code* represents your display language)
- setenv SIEBEL_ROOT *Installation_directory_path*

3 Navigate to *$SIEBEL_ROOT/bin* and enter:

```
dbsrvr_config.ksh
```

This launches the Database Server Configuration Wizard.

4 Review the values of the following environment variables and confirm whether or not the settings are correct by entering either Y or N.

- SIEBEL_ROOT
- LANGUAGE

NOTE: If either the SIEBEL_ROOT or LANGUAGE value is not set or is incorrect, you must correct them before proceeding.

5 Specify the path of your Siebel Server root directory, or accept the default by pressing ENTER.

6 Specify the path of your database server root directory, or accept the default by pressing ENTER.

7 Enter the number that corresponds to your database platform.

8 From the Select Repository Operations menu, choose Migrate Repository (4).

9 From the Source Repository Selection menu, select to read the source repository from the database or from a previously exported file.

- If you select to read the source repository from file, the Repository File Name menu appears.

  Enter the path and filename of the source repository file and click Next. The Target Environment Status menu appears.

  For background information on creating a source repository file, see"Creating a Source Repository Migration File" on page 12.

- If you select to read the source directly from the database, the Target Environment Status menu appears.
- In the Target Environment Status menu, select whether the target environment will be online or offline, and click Next. For further information on this selection, see "Preparing for Target Environment Status Selection" on page 11.

10 In the Target Environment Status screen, select whether the target environment will be online or offline, and click Next. For further information on this selection, see "Preparing for Target Environment Status Selection" on page 11.

11 In the Schema Changes menu, select whether there are new schema changes to be applied during the repository migration, and click Next. For further information on this selection, see "Preparing for Schema Changes Selection" on page 11.

12 In the Language Selection screen, select the appropriate language, and then click Next.

NOTE: The repository migration process migrates all language strings within the repository. Select the base language installation that applies for this screen.

13 The next succession of screens are dependent on the repository source selection:

- If you chose to read the source repository from file, the Target Repository Name screen appears. Enter the target repository name, and click next.

- If you chose to read the source directly from database, the following succession of screens appears to define the source database.

Screen	Description
ODBC Data Source Name	If you chose to read the source repository from the database, enter the ODBC data source name.
Database User Name	If you chose to read the source repository from the database, enter the source database user name and source database password.
Database Table Owner	If you chose to read the source repository from the database, enter the source database table owner and source table owner password.
Source Database Repository Name	If you chose to read the source repository from the database, enter the database repository name and the target database repository name.

14 Progress by completing the information in each screen.

Screen	Description
RDBMS Platform	Select an RDBMS platform. IBM DB2 UDB v7.1 is the default.
Target Database Encoding	Select whether or not the target database is Unicode.
Target Database Repository Name	Enter the target database repository name.
Target RDBMS Platform	Select the target RDBMS platform. IBM DB2 UDB v7.1 is the default.
Target Database ODBC Datasource	Enter the target database ODBC datasource.
Target Database User Name	Enter the target database user name and password used by the server components.
Target Database Table Owner	Enter the target database table owner and table owner password.
Index Table Space Name (DB2-specific)	If you choose IBM DB2 for the target database platform, this screen appears. Enter the index table space name and 4-KB table space name.

Screen	Description
16K Table Space Name (DB2-specific)	If you choose IBM DB2 for the target database platform, this screen appears. Enter the 16-KB table space name and 32-KB table space name.
Index Table Space Name (Oracle-specific)	If you choose Oracle as the target database platform, you have only two questions about index and table space. There are no 4-KB, 16-KB, and 32-KB tablespaces in Oracle. For Microsoft SQL server, there are no screens about tablespaces.

15 After the last screen in the previous step, a dialog box appears prompting you to run the repository migration now or at a later date. Click OK to begin the repository migration process, and CANCEL to run the migration process at a later date.

- If you select OK, a Configuration Parameter Review screen appears. Review this screen to check your configurations.
- If you select CANCEL, run the migration process at a later date by navigating to the bin subfolder of the Siebel Server root directory and entering the following command:

  ```
  Srvrupgwiz /m master_dev2prod.ucf
  ```

NOTE: Updating statistics is the job of the database administrator once any repository migration, upgrade, or installation finishes.

Importing and Exporting Repositories Using Repimexp

Import and export repositories using the repimexp.exe program only if your migration requires special parameters settings that are inaccessible through the batch files or if you need to perform a file dump. In all other cases, use the configuration utility.

For information on using the configuration utility, see "Running the Repository Migration Configuration Utility" on page 12. For further information on the process of migrating repositories, see "Process for Migrating Repositories" on page 8.

The repimexp.exe program imports, exports, or creates a file dump of a repository. It can also import an INTL table. INTL tables contain language-specific information and are a part of the repository.

To import a repository using repimexp

- In the command line, type the following:

```
repimexp /A I /G language_codes
```

where:

/A I = import switch.

language_codes = a list such as ENU, FRA, JPN. Use ALL for all languages.

NOTE: If you want to import your repository with locale objects, you must specify at least one language code. Otherwise, no locale objects will be imported, and the repository will not have any text in the user interface when you compile the imported repository.

To export a repository using repimexp

- In the command line, type the following:

```
repimexp /A E argument_list
```

Export uses the arguments listed in Table 3.

Table 3. Parameter Settings Passed as Export Arguments to repimexp.exe

Parameter	Required	Description
/A E	Yes	Export switch.
/U <userName>	Yes	Siebel administrator user name.
/P <password>	Yes	Siebel password.
/C <ODBC data source>	Yes	ODBC data source. The default is the value in the SIEBEL_DATA_SOURCE environment variable.
/D <table owner>	Yes	Siebel database table owner. The default is the value in the SIEBEL_TABLE_OWNER environment variable.
/R <repository>	Yes	Repository name. The default is Siebel Repository.
/F <dataFile>	Yes	Data file, including path, to which to export.
/T <Y|N>	No	Test only, do not export into database.
/V <Y|N>	No	Verify data. The default is N.
/N <0|1|2>	No	Change creation and update information: 0= no change

Upgrading Mobile Databases

This topic describes the task of upgrading mobile databases. This task is a part of the "Process for Migrating Repositories" on page 8. For background information on migrating repositories, see "About Migrating Repositories Between Databases" on page 7.

Follow these steps to upgrade mobile databases:

1. Stop and restart Siebel Remote server components.

   When you have restarted the processes, wait until the Transaction Processor (alias TxnProc) and the Transaction Router (alias TxnRoute) have processed all pending transactions before proceeding with the remaining steps.

2. Regenerate local database templates.

   Use the Siebel Server component Generate New Database (alias GenNewDb) to regenerate the local database template file to update its schema to the same version as the database server. For details on this task, see *Siebel Remote and Replication Manager Administration Guide*.

3. Reextract mobile users.

   If you are not using Siebel Anywhere to upgrade your mobile clients, reextract all mobile users, using the component Database Extract (alias DbXtract).

   If mobile databases are not reextracted, users will still be able to synchronize—no error message will be generated. This functionality allows Siebel Anywhere, which users might use to upgrade mobile databases, to continue working.

   If you are using Siebel Anywhere, refer to *Developing and Deploying Siebel Business Applications* for instructions on propagating schema extensions.

Post-Repository Migration Tasks

After a successful repository migration, there are other tasks required to migrate or re-create nonrepository configurations from one environment to another. Review the following list to update data and information to the new environment, if required:

- Enterprise customizations data, such as views, responsibilities, assignment rules, and so on. To migrate these customizations, see Chapter 3, "Migrating Customizations Between Applications."
- Web Templates and related files. To migrate these files, see "Migrating Web Templates and Related Files" on page 21.
- Custom configurations such as parameter values and component definitions. To re-create these configurations, see "Re-creating Custom Environment Configurations" on page 21. For specific details on re-creating parameter values, see Chapter 4, "Migrating Parameters Between Environments."

After these nonrepository items are migrated, roll out the updates to mobile clients. For more information on this task, see Chapter 5, "Rolling Out Updates to Mobile Clients."

Migrating Web Templates and Related Files

Every Siebel Server installation includes a set of files that define how the application appears in a Web browser. These include:

- Web templates (.swt files)
- Images (.gif and .jpg files)
- Cascading style sheets (.css files)

If you make changes to any of these files in your source environment, you must copy the modified files to the target environment.

To move Web templates and related files

1. Copy any new or modified files of the following types from the source Siebel Server to the target Siebel Server:

Files	File Location
Web Templates	siebsrvr_root\WEBTEMPL
Images	siebsrvr_root\webmaster\images\language_code
Cascading Style Sheets	siebsrvr_root\webmaster\files\language_code

Re-creating Custom Environment Configurations

When migrating from one environment to another, re-create any custom environment configurations from the source environment in the target environment, if necessary. Custom environment configurations include data such as custom component definitions. This information is stored in the Siebel Gateway siebns.dat file and is not migrated as part of a repository migration.

For specific details on automatically re-creating custom parameter values, see Chapter 4, "Migrating Parameters Between Environments."

Re-create custom environment configurations in one of two ways:

- Use the Server Manager GUI or command-line interface program (srvrmgr) to create these custom configurations manually in the new target environment. For information on creating custom component definitions, see *Siebel System Administration Guide*.
- Use the Server Manager command-line interface program (srvrmgr) with input scripts to automate this task in the new target environment. This is the preferred method because scripts can be saved and reapplied in future environment updates of the same version. For information on using scripts with srvrmgr, see *Siebel System Administration Guide*.

NOTE: If reapplying a srvrmgr script to re-create custom environment configurations, make sure to edit the script if the target environment is a new Siebel version (as in the case of a Siebel upgrade). New parameters or srvrmgr commands may be available with the new version.

3 Migrating Customizations Between Applications

A configured Siebel application generally has many run-time customizations that further define the deployment. Run-time customizations include data such as assignment rules, predefined queries, views, responsibilities, and so on. When going live to a new development, test, or production environment, a common requirement is moving run-time customization data from one environment to another. The Application Deployment Manager (ADM) feature automates the migration of the Siebel application run-time customization data.

For further information on ADM and the run-time customization data it migrates, see "About Application Deployment Manager (ADM)" on page 23.

Part of the process of going live to a new environment may also include a migration of a new repository. See Chapter 2, "Migrating Repositories Between Databases" for information on this process.

This chapter includes the following topics:

About Application Deployment Manager (ADM)

Application Deployment Manager (ADM) is a feature that automates the process of migrating enterprise customization data (views, responsibilities, assignment rules, and so on) from one Siebel application environment to another. For example, ADM can move customization data from a development environment to a testing environment.

ADM is designed to provide a single deployment tool that covers various areas within the Siebel application. The objective is to reduce the potential manual setup and deployment work and provide as much automation as possible to decrease the error rate.

NOTE: The term *migrating* is used in the ADM context as the moving of data from one environment to another. No changes to the data take place during migration.

The bulk of the administrative tasks to migrate data using ADM are performed at the Application Deployment Manager screen in the Siebel application GUI. These tasks are intended for those with Siebel administrator responsibility. The ADM set-up process in the ADM GUI creates a template in which one data type can be migrated on a regular basis, if required. The fundamental structure of this template is the deployment project. The deployment project consists of one or more data types that can be migrated.

The process for migrating ADM involves configuration, GUI set-up, and deployment. Configuration setup is necessary in Siebel Tools to enable ADM support for additional business objects within the Siebel application; GUI setup in the Siebel application is necessary to prepare for the ADM deployment execution; and the ADM deployment process of migrating customizations can be run from the application GUI or the Server Manager command-line. See "Process for Migrating Customizations Between Applications" on page 27 for further information on this process and links to their tasks.

The following data types are available for migration with ADM:

- Assignment Rule
- Assignment Groups
- Access Group
- List of Values (LOV)
- Public Predefined Query (PDQs)
- Expense Types
- Product Feature
- Product Line
- Responsibility
- View
- State Model
- User List
- Joint Workspace and additional objects used by Siebel Customer Order Management. For further information, see *Siebel Order Management Infrastructure Guide*.

These data types are preconfigured in the standard Siebel Business application.

As ADM runs within the Siebel application, it might have a dependency on the customized SRF or repository. Therefore, consider such an impact when planning the deployment. For example, if additional objects have been enabled for ADM support, then deploy the SRF and repository on the source and target environments for ADM to properly deploy the custom objects.

About ADM Data Type Relationships

Creating Application Deployment Manager (ADM) data type relationships is only necessary if certain data types must be deployed before other data types. Creating this relationship makes sure that related data types are deployed as a single transaction to the target system. If any of the related data types are not applied correctly, then all related data types are not migrated to the target system.

For descriptions on creating or removing data type relationships, see "Creating ADM Data Type Relationships" on page 31.

Example of ADM Data Type Relationship

An example of an ADM data type relationship is as follows: A List of Values (LOV) plus a State Model configured on that LOV. Therefore, the LOV must be governed by a State Model.

To set up this relationship in ADM add the LOV data type as a child record to the State Model.

With this relationship setup, the records of the LOV (LOV Type record and LOV value records) are inserted into the database first followed by the State Model records. If there is an error in moving either the State Model or the LOV records, then both are not allowed to exist separately on the target system. If the State Model encounters a database error, then the previously inserted LOV records are removed (rolled back) so that the errors are corrected and the session can be retried.

About ADM Deployment Filters

Application Deployment Manager (ADM) deployment filters allow the migration of specially selected records of a particular data type. Creating ADM deployment filters is a part of creating ADM deployment projects.

For information on creating deployment filters in deployment projects, see "Creating ADM Deployment Filters" on page 35.

ADM deployment filters can also be set or modified when creating deployment sessions. See "Creating Deployment Sessions Using ADM" on page 36 for further information on this task.

The deployment filter applies to the primary business component defined within an integration object, and can be set on any field defined on the primary business component and present in the integration object. Each data type is represented by a Siebel integration object. See "Creating Integration Objects for ADM" on page 28 for further information on Siebel integration objects. For detailed information on business components, see *Configuring Siebel Business Applications*.

Filtering is also possible on the child business components to exclude or include certain records. See the end of this topic for an example

The format for the deployment filter value entered into the Filter field of the data type record is as follows:

[*Field_Name*] *operator* '*Filter_Criterion*'

where:

Field_Name = The name of the field on which to filter records. For a list of available fields for each data type, click the select button in the Field List field. Make sure the field name is enclosed in brackets.

operator = A standard Siebel query operator that defines the filter. For example, `like` or =. For a list of Siebel query operators, see *Using Siebel Tools*.

Filter_Criterion = The criterion by which to filter the field name. The wild card character can be used as part of the criterion. For example, names beginning with `'A*'`. The filter criterion is case-sensitive and must be enclosed in quotes.

NOTE: Use multiple filters for one data type by using the logical AND or OR operators.

Some deployment filter examples follow:

- `[Name] like 'ABC*' or [Name] = 'My StateModel'`

  This example filters the data type State Models using the Name field. The primary business component for State Models is State Model, the field containing the state model name is Name.

- `[Activation Date]>'12/29/2001 14:58:29'`

  This example filters the data type Assignment Rules using the Activation Date field. All assignment rules with a start date greater than 12/29/2001 are filtered for migration.

- `[Value]='ACCOUNT_STATUS' AND [List Of Values Child (UDA).Language] ='ENU'`

  This example filters the data type LOV using the Value field and the child business component List of Values Child (UDA).Language. The filter only allows deployment of the English language LOVs for the ACCOUNT_STATUS LOV type.

Best Practices for Migrating Data Using ADM

Review the following information as recommendations of best practice when migrating data customizations using ADM.

- Do not set up a large number of data types. Keep the number of project items low to assist with error management. If the deployment scenario contains more than ten line items, break up the project into separate projects. This action helps improve usability.
- Maintain strong development guidelines, especially with naming conventions. Consistent development guidelines assist in the creation of deployment filters.
- Make sure that seed data, included as part of an installation, is in place for both the source and target environments. Do not use ADM to migrate this type of data. For example, do not migrate standard, unchanged LOV data types, which are available as part of a regular Siebel installation.
- The ADM is best suited for a high-frequency and low-volume deployment scenario. It is highly recommended to set up a deployment filter on all items. Review Siebel Enterprise Integration Manager (Siebel EIM) as a more appropriate tool for transferring large data volumes. For information on Siebel EIM, see *Siebel Enterprise Integration Manager Administration Guide*.
- Maintain a compatible database codepage between the source and target environments. If in doubt, contact Siebel Technical Support.
- When creating additional data areas, validate the integration object structure first before using it with the ADM framework. Use a simple workflow process in the process simulator to validate the integration object. For information on workflow processes and the process simulator, see *Siebel Business Process Designer Administration Guide*.
- When working with files for deployment, always use a network share name in the UNC format. On Windows, do not use drive letters. Remember that file name is specified for importing from the GUI but the directory name must be specified for exporting.
- Note that the deployment occurs in an asynchronous mode and the Server Request Broker (SRBroker) and Server Request Processor (SRProc) component parameters affect how fast the requests are processed. By default, no adjustments are necessary.

Process for Migrating Customizations Between Applications

The process for migrating run-time data customizations uses the Application Deployment Manager (ADM) feature and involves two stages: One-time setup and deployment.

For background information on migrating data using ADM, see "About Application Deployment Manager (ADM)" on page 23.

One-Time Set-Up Stage

To complete the one-time set-up stage for migrating customizations using ADM, perform the following tasks:

1. "Creating Integration Objects for ADM" on page 28
2. "Creating Content Objects for ADM" on page 28
3. "Configuring Batch Workflows for ADM Command-Line Interface Deployment" on page 29
4. "Setting Up the Application Environment for Data Migration" on page 30
5. "Creating ADM Data Types" on page 30
6. "Creating ADM Data Type Relationships" on page 31
7. "Creating ADM Deployment Projects" on page 33
8. "Creating ADM Deployment Filters" on page 35
9. "Enabling the ADM Deployment Project" on page 36

NOTE: Steps 1, 2, and 5 are only necessary if enabling ADM support for new objects. These steps are not necessary if using the preconfigured data types already available.

Deployment Stage

To complete the deployment stage for migrating customizations using ADM, perform the following tasks:

1. "Creating Deployment Sessions Using ADM" on page 36
2. Deploy the ADM Deployment Session. Choose one of the following three methods to deploy ADM sessions:
   - "Deploying Sessions Using the Application Deployment Manager GUI" on page 37
   - "Deploying Sessions Using Export Files" on page 38
   - "Deploying Sessions Using Command-Line Interface" on page 40

Creating Integration Objects for ADM

Creating an integration object for Application Deployment Manager (ADM) is a task in the "Process for Migrating Customizations Between Applications" on page 27. This task expands the data types that can be migrated using ADM.

NOTE: The preconfigured data types do not require any configuration in Siebel Tools.

The integration object specifies the format and structure of the set of data you want to migrate.

Use the EAI Object Wizard in Siebel Tools to create integration objects. For background information on integration objects and for details on using the EAI Object Wizard, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.

NOTE: Intermediate-level knowledge of the Siebel EAI architecture and integration is a prerequisite before performing these tasks. For further information on Siebel EAI, see *Overview: Siebel Enterprise Application Integration*.

To create an ADM Integration Object

1. Identify the business object that corresponds to the data type requiring migration using ADM. This business object must have only one primary business component.
2. In Siebel Tools, select File > New Object.
3. From the New Object screen, select Integration Object.

   The EAI Object Wizard guides you through the process of setting up an integration object.
4. Adjust the user keys of the integration object. For further information on user keys for integration objects, see *Integration Platform Technologies: Siebel Enterprise Application Integration*.
5. Validate the integration object by selecting the integration object and clicking Validate.
6. Review the report and modify your integration object as needed.

Creating Content Objects for ADM

Creating a content object for Application Deployment Manager (ADM) is a task in the "Process for Migrating Customizations Between Applications" on page 27. This task expands the data types that can be migrated using ADM.

NOTE: The preconfigured data types do not require any configuration in Siebel Tools.

Before creating a content object, make sure you have created an integration object for the data you want to migrate. See "Creating Integration Objects for ADM" on page 28.

The content object specifies the data that will be migrated from one Siebel application to another. Each content object is composed of a business object and an integration object.

Content objects are also used by Content Center. The following procedure references properties necessary for ADM functionality. For Content Center functionality, additional properties must be set for the Content object. For further information on Content Center, see *Applications Administration Guide*.

NOTE: One content object can be used by both Content Center and ADM at the same time.

To create content object for ADM

1. In Siebel Tools, navigate to Content Objects type.
2. Create a new record.
3. Add the data type business object to the content object record.
4. Add the data type integration object to the content object record.
5. Create child records. On the child record, enter the name of a view that does not have any visibility setting (admin view) by selecting from the picklist.

   The child record determines which view will be used during the preview functionality in the ADM GUI. This view must correspond to the business object of the data area.

Configuring Batch Workflows for ADM Command-Line Interface Deployment

Configure Application Deployment Manager (ADM) batch workflows in Siebel Tools if deploying sessions by using export files or by using the Server Manager command-line interface program (srvrmgr). When using srvrmgr to execute the deployments, ADM exports all items in a deployment session in one step using multiple files. ADM also imports all items exported from that deployment session in one step.

This task is a one-time only requirement, and is an optional task in the "Process for Migrating Customizations Between Applications" on page 27. This task is not required if you plan to migrate deployment sessions using the ADM GUI.

To configure Siebel Tools for ADM command-line deployment

1. Launch Siebel Tools.
2. In the Object Explorer of Siebel Tools, select the Workflow Process object.
3. Locate the following workflow processes:
   - UDA Batch Deployment
   - UDA Batch Import
4. If not revised earlier, click Revise.
5. For the workflow process UDA Batch Deployment, navigate to the WF Process Props view; in the `Default Strings` column, configure the property `ExportFilePath` to a shared network path.

   ADM stores export files to this location.

6 For the workflow process UDA Batch Import, navigate to the WF Process Props view; in the `Default Strings` column, configure the property `Import File Directory` to a shared network path.

   ADM retrieves import files from this location.

7 Deploy the workflow processes from Siebel Tools.

   For further information on this procedure, see *Siebel Business Process Designer Administration Guide*.

8 Activate both workflow processes in the application GUI.

   For information on this procedure, see *Siebel Business Process Designer Administration Guide*.

NOTE: Activate the UDA Batch Deployment workflow on the source system and the UDA Batch Import workflow on the target system.

Setting Up the Application Environment for Data Migration

Before migrating configuration or administrative data from one Siebel application to another using Application Deployment Manager (ADM), make sure certain application environment functions are set up and running.

This set-up check is a task in the "Process for Migrating Customizations Between Applications" on page 27.

Make sure the following are set up and functioning in the application environment prior to running ADM:

- Enable the following server component groups or confirm availability: Workflow Management (Workflow), Enterprise Application Integration (EAI) and System Management (System). For information on enabling server components and server component groups, see *Siebel System Administration Guide*.
- Synchronize server components. For information on synchronizing server components see *Siebel System Administration Guide*.
- Activate necessary workflows that have a name starting with UDA. For information on activating workflows, see *Siebel Business Process Designer Administration Guide*.

Creating ADM Data Types

Creating Application Deployment Manager (ADM) data types is a task in the "Process for Migrating Customizations Between Applications" on page 27. Use the ADM GUI to complete this task. Creating an ADM data type establishes a reference to the content object, for example, an assignment rule or price list.

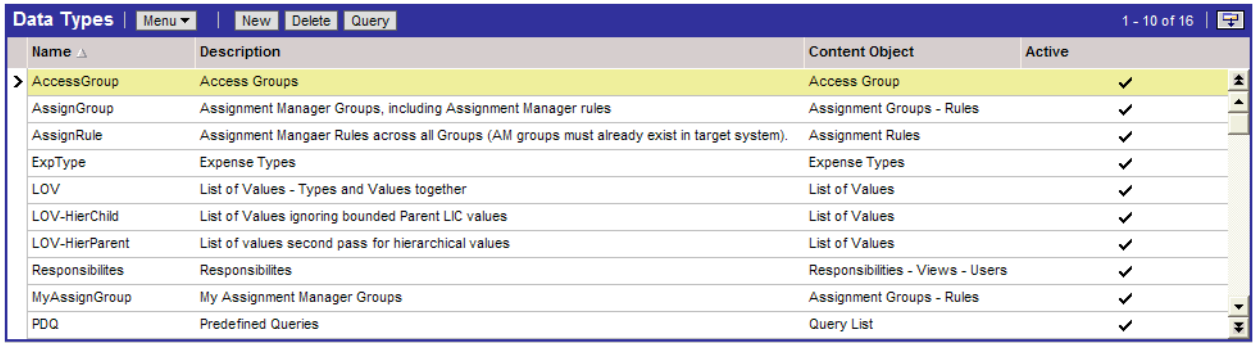
To create an ADM data type

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Data Type Details.
3. In the Data Types Details list, click the menu button and then New Record.
4. In the Data Type Object field, select the content object associated with the data type you want to migrate, for example, a product line or list of values.

   NOTE: If a content object is not available, make sure it is configured in Siebel Tools, and compiled into the SRF in use by the application. See "Creating Integration Objects for ADM" on page 28 for further information.
5. In the Name field, type in a descriptive name for the data type.
6. Click the menu button and then Save Record.

ADM sets the Data Type Active field check box by default. Clearing this field sets the data type to inactive. In the inactive state, the data type cannot be added to a deployment project, a subsequent task in "Process for Migrating Customizations Between Applications" on page 27.

Figure 2 shows an example of creating an ADM data type.



Name	Description	Content Object	Active
AccessGroup	Access Groups	Access Group	✓
AssignGroup	Assignment Manager Groups, including Assignment Manager rules	Assignment Groups - Rules	✓
AssignRule	Assignment Mangaer Rules across all Groups (AM groups must already exist in target system).	Assignment Rules	✓
ExpType	Expense Types	Expense Types	✓
LOV	List of Values - Types and Values together	List of Values	✓
LOV-HierChild	List of Values ignoring bounded Parent LIC values	List of Values	✓
LOV-HierParent	List of values second pass for hierarchical values	List of Values	✓
Responsibilites	Responsibilites	Responsibilities - Views - Users	✓
MyAssignGroup	My Assignment Manager Groups	Assignment Groups - Rules	✓
PDQ	Predefined Queries	Query List	✓

Figure 2. Creating an ADM Data Type

Creating ADM Data Type Relationships

Creating Application Deployment Manager (ADM) data type relationships is only necessary if certain data types must be deployed before other data types. This task is a step in "Process for Migrating Customizations Between Applications" on page 27

For background information on ADM data type relationships, see "About ADM Data Type Relationships" on page 24.

Set up parent-child relationships between data types using the Data Type Explorer view.

To add a child relationship to an ADM data type

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Data Type Explorer.
3. In the Data Types list or Data Type explorer list, select the data type of interest.
4. Click the Add button to access the data type list.
5. Select the child data type of interest from this list.

   The selected child data type appears under the parent data type in the Data Type Explorer view.

To remove a child relationship from an ADM data type

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Data Type Explorer.
3. In the Data Types list or Data Type explorer list, select the data type of interest.
4. In the child data type list, select the child data type to remove.
5. Click the Remove button.

Figure 3 shows an example of creating an ADM data type child relationship between StateModel and LOV data types.

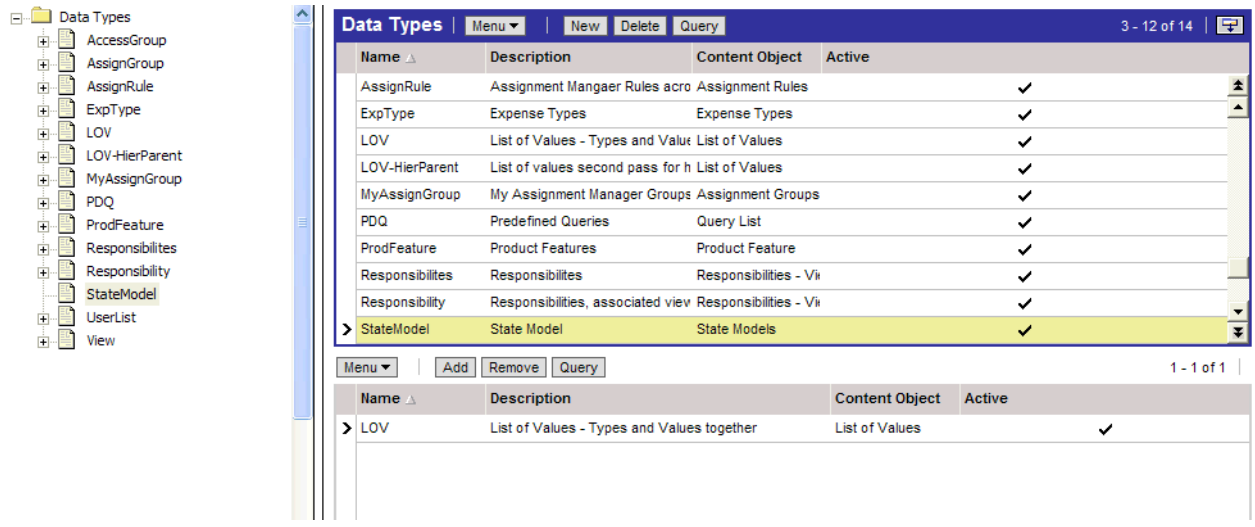


Figure 3. Creating a Child Data Type Relationship

Creating ADM Deployment Projects

Creating Application Deployment Manager (ADM) deployment projects is a step in "Process for Migrating Customizations Between Applications" on page 27. A project is the template that can be reused to migrate the same data types on a regular basis. A project can contain many data types.

To create an ADM deployment project

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Deployment Projects.
3. In the Deployment Projects list, click the menu button and then New Record.
4. Fill in the project record fields as necessary. See the following table for details on some of these fields.

Field	Description
Name	Name for the deployment project.
Target System	Target system for deploying the migration information. This should be the URL to the target EAI object manager.
Target User Name	User name required to access the target system.
Active Flag	Set by default, this flag signifies the project is available for deployment as a deployment session. Clear this field to make sure this project is not deployed.
Export to File	Check this field if migrating data by using export files. See "Deploying Sessions Using Export Files" on page 38 for information on this process.
Session Configurable	Check this field if further configurations are required when accessing this project as a deployment session. If this value is unchecked, the project field Export to File is read-only, as are the project data type fields, Active and Deployment Filter.

5. In the Deployment Project data type list, populate the new project with data types by clicking the menu button then New Record. For each record:
   a. Select a previously created data type from the Data Type Name drop-down list.

      NOTE: If the data type of interest is not available, it may be set to inactive. Check to make sure the Active field is set in the Data Type Details view.

b Fill in the other data type record fields as necessary. See the following table for details on some of these fields.

Field	Description
Active	Clear the active field to inactivate a data type within a project. Use this feature to migrate a deployment project without migrating the data from the inactivated data types.
Deployment Filter	See "Creating ADM Deployment Filters" on page 35 for details on this field.
Allow Child Delete	Check this flag to allow deletion of the target child records, which keeps the object's data structure identical between the source and target environments. This flag issues the EAI Adaptor command: Sync. For further information, see *Integration Platform Technologies: Siebel Enterprise Application Integration*

c Click the menu button and then Save Record.

6 Add other data types as required to the draft deployment project.

NOTE: If a relationship is set up between two data types, only the parent data type is visible in the picklist. However, after adding the data type to the project item list, the parent data type record can be expanded to reveal the child data types. Make sure to save the record if you cannot expand the parent data type.

7 From the Deployment Projects list, click the menu button and then Save Record.

Figure 4 shows an example of creating an ADM project.

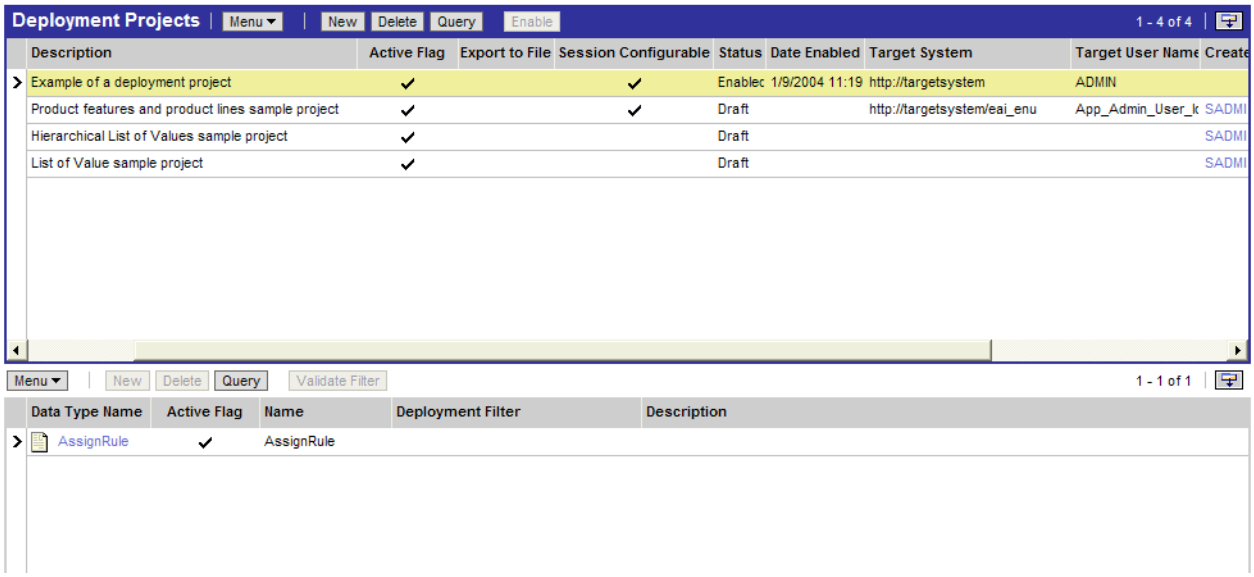


Figure 4. Creating an ADM Project

Creating ADM Deployment Filters

Creating Application Deployment Manager (ADM) data filters is an optional, but highly recommended, task of the ADM Server Manager GUI set-up process. This task is a step in the "Process for Migrating Customizations Between Applications" on page 27.

The deployment filter allows the migration of specially selected records of that particular data type. It is recommended to use a filter for each data type. For further information about deployment filters, see "About ADM Deployment Filters" on page 25.

To create an ADM deployment filter

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Deployment Projects.
3. In the Deployment Projects list, select the Project of interest.
4. In the Deployment Project data types list, select the Data Type of interest.
5. In the Deployment Filter field, create a search expression to filter out only those items of a data type matching the condition for migration.

   NOTE: For details on the format for deployment filters, see "About ADM Deployment Filters" on page 25.
6. To make sure the filter is accurate, click the Validate Filter button.

Enabling the ADM Deployment Project

Enabling the Application Deployment Manager (ADM) deployment project is a task in the "Process for Migrating Customizations Between Applications" on page 27.

Enabling the project:

- Validates data filters created on the data types.
- Enables the project for use in deployment sessions.
- Locks the project values, that is, nothing can change except the Active field. (If you inactivate the project by clearing the Active field, you cannot deploy the project.)

NOTE: After a project is enabled, it cannot be updated but can be set inactive to prevent further usage. To update an enabled project, copy the project and provide a different name.

To enable the ADM deployment project

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Deployment Projects.
3. In the Deployment Projects list, select the draft deployment project of interest. (The Status field of the draft deployment project record appears as Draft.)
4. Click the Enable button to activate the project.

   ADM populates the Status field with Enabled and the Publication Date/Time field with the data and time of the project activation.

Creating Deployment Sessions Using ADM

Creating deployment sessions using the Application Deployment Manager (ADM) is a task in the "Process for Migrating Customizations Between Applications" on page 27.

Before creating a deployment session, make sure that required configurations are complete and a deployment project exists.

To create a deployment session

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Deployment Sessions.
3. In the Deployment Sessions list, click the menu button and then New Record.

4 In the Project Name field drop-down list, select the deployment project of interest.

ADM populates several fields in the record with information from the deployment project.

NOTE: Only enabled deployment projects are available for addition to a deployment session. See "Enabling the ADM Deployment Project" on page 36 for further information on this task.

5 If necessary, clear the Deployment Lock field to change the Deploy button to read-only status. (Click the Refresh button to see this change.) When the Deployment Lock field is selected, the individual listed in the Locked By field is the only person who can deploy the deployment session. (In normal operations, the Deployment Lock flag is always checked.)

NOTE: The Deployment Lock flag is useful in complex environments where multiple individuals work together. For example, one person can create and unlock the session, which allows another person to login, lock, and deploy that session.

6 If necessary, change the values in the fields of the deployment project's data types. These fields are accessible only if the deployment project field Session Configurable was checked during the creation of the deployment project. For further information on these fields, see "Creating ADM Deployment Projects" on page 33.

7 To refresh the data in the Deployment Sessions view, click the Refresh button.

Figure 5 shows an example of creating an ADM deployment session.

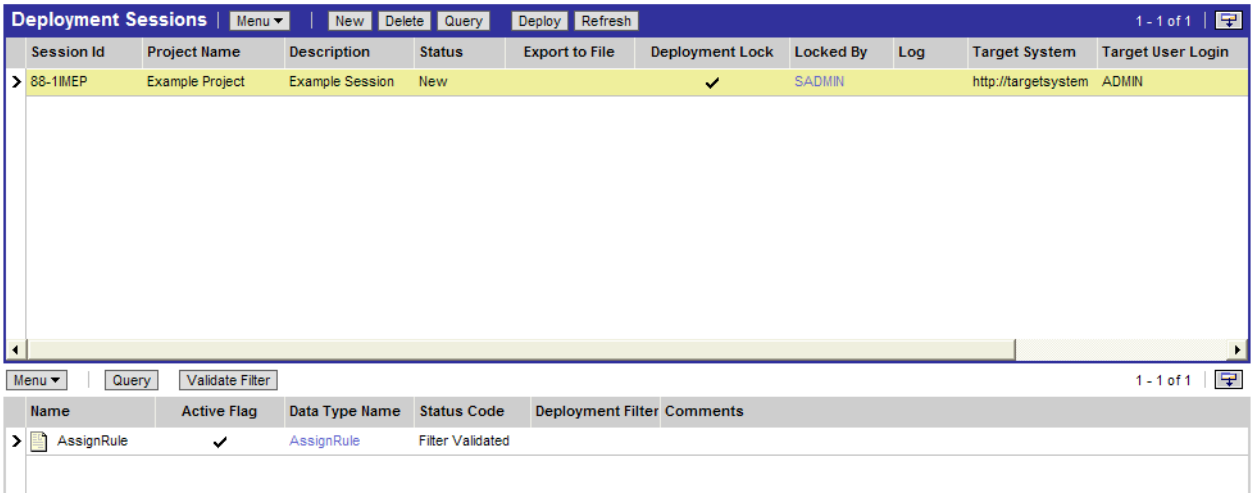


Figure 5. Creating an ADM Session

Deploying Sessions Using the Application Deployment Manager GUI

Deploying sessions by using the ADM GUI is one means to migrate data from one Siebel application environment to another, and is a task in the "Process for Migrating Customizations Between Applications" on page 27. For other deployment options, see the following sections:

- "Deploying Sessions Using Export Files" on page 38

■ "Deploying Sessions Using Command-Line Interface" on page 40

Before deploying a session, make sure that required configurations are complete and a deployment session exists. See "Creating Deployment Sessions Using ADM" on page 36 for information on creating deployment sessions and other required configurations.

To migrate a deployment session

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Deployment Sessions.
3. Select the deployment session of interest. Make sure the Deployment Lock field is checked.
4. Click the Deploy button.
5. Enter the password for the target system user and then click the Deploy button.

   The Status field of the deployment session record changes to Submitted. Click the Refresh button to further update the status of the deployment.

   NOTE: The volume of data and the number of items within a session contribute to the processing time.
6. Check the details of the deployment by reviewing ADM log files and the EAI queue. See "Reviewing the ADM Data Migration" on page 41 for further information on these tasks.

Deploying Sessions Using Export Files

Deploying sessions using export files is one means to migrate data from one Siebel application environment to another, and is a task in the "Process for Migrating Customizations Between Applications" on page 27. For other deployment options, see the following sections:

■ "Deploying Sessions Using the Application Deployment Manager GUI" on page 37

■ "Deploying Sessions Using Command-Line Interface" on page 40

Before deploying a session, make sure that required configurations are complete and a deployment session exists. See "Creating Deployment Sessions Using ADM" on page 36 for information on creating deployment sessions and other required configurations.

To migrate data using export files, perform the following tasks:

1. Export a deployment session from the source system to a deployment session XML file.
2. Import the deployment session XML file to the target system.

To export a deployment session file

1. Access the source system, that is, the system from which you want to migrate data.
2. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.

3 From the link bar, click Deployment Sessions.

4 Select the deployment session of interest. Make sure the Export to File and the Deployment Lock fields are checked.

5 Click the Deploy button.

6 Enter the network path for the export directory, then click the Export button. (Do not specify a file name. ADM automatically generates a file name based on the items within the session.)

   For example, on Windows, if the network administrator sets up a shared directory called stage_1 on the server OMEGA, then the network path supplied to ADM should be \\OMEGA\stage_1.

   Click the Refresh button to update the status of the deployment.

   NOTE: The network path is relative to the Siebel Server. It is recommended to set up a common area for file operations, which is accessible from all the Siebel Servers. Additional set-up tasks may be necessary depending on the Siebel Server operating system.

7 Navigate to the network path entered in the previous step to review and access the export file.

8 Check the details of the deployment by reviewing ADM log files and EAI queue. See "Reviewing the ADM Data Migration" on page 41 for further information on these tasks.

NOTE: The volume of data and the number of items within a session contribute to the processing time.

To import a deployment session file

1 Access the target system, that is, the system to which you want to migrate data.

2 Make sure the source file is accessible.

3 From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.

4 From the link bar, click Deployment Sessions.

5 In the Deployment Sessions list, click the menu button and then Deploy from File.

6 Enter the network path and file name for the import file, then click the Import button.

   NOTE: The import process requires the file name as input. If the export session generated multiple files, the import step needs to be executed for each file.

7 Check the details of the deployment by reviewing the target application object manager log files and EAI queue. See "Reviewing the ADM Data Migration" on page 41 for further information on these tasks.

Deploying Sessions Using Command-Line Interface

Deploying sessions using the Server Manager command-line interface (srvrmgr program) is one means to migrate data from one Siebel application environment to another, and is a task in the "Process for Migrating Customizations Between Applications" on page 27. For other deployment options, see the following sections:

- "Deploying Sessions Using the Application Deployment Manager GUI" on page 37
- "Deploying Sessions Using Export Files" on page 38

Before deploying sessions using the command-line interface, make sure that required configurations are complete and deployment sessions exist. See "Creating Deployment Sessions Using ADM" on page 36 for information on creating deployment sessions and other required configurations. Also make sure ADM batch workflows are configured in Siebel Tools. See "Configuring Batch Workflows for ADM Command-Line Interface Deployment" on page 29 for a description of this task.

To migrate deployment sessions using the Server Manager command-line interface, perform the following tasks:

1. From the source system Server Manager command-line interface, deploy sessions using export files to a shared network path.
2. From the target system Server Manager command-line interface, import the deployment session files to the target system.

To export a deployment session file using Server Manager command-line interface

1. In the source system, make sure deployment sessions intended for srvrmgr deployment have the field Export to File checked. See "Creating Deployment Sessions Using ADM" on page 36 for further information on deployment sessions.
2. In the source system, make sure the Deployment Lock value is checked and your Server Manager user ID is in the Locked By field for each session.

   NOTE: You can only migrate locked deployment sessions with your user ID.
3. Start Server Manager command-line interface (srvrmgr program) with your user ID.

   For details on this procedure, refer to *Siebel System Administration Guide*.
4. Verify the server component Workflow Process Manager (WfProcMgr) is online.

   If not, enable the Workflow component group and restart the Siebel Server. For details on these procedures, see *Siebel System Administration Guide*.
5. Enter the following command at the Server Manager command-line interface prompt:

start task for comp WfProcMgr server *Siebel_Server_Name* with ProcessName="UDA Batch Deployment", RowID="*Session_ID*"

where:

Siebel_Server_Name = Name of the Siebel Server.

UDA Batch Deployment = Name of the workflow process. This value is case-sensitive.

Session_ID = Session ID of deployment session created earlier through the ADM GUI.

6 Check the details of the deployment by reviewing the source system ADM log files and EAI queue. By default, an EAI queue entry is created in case of any deployment errors on the target system See "Reviewing the ADM Data Migration" on page 41 for further information on these tasks.

To import deployment session file using Server Manager command-line interface

1 Make sure the export deployment file exists in a shared network location.

NOTE: You do not need to create a deployment session in the target environment to import a deployment session file.

2 Start Server Manager command-line interface (srvrmgr program) with your user ID in the target system.

For details on this procedure, refer to *Siebel System Administration Guide*.

3 Verify the server component Workflow Process Manager (alias WfProcMgr) is online.

If not, enable the Workflow Management component group (alias Workflow) and restart the Siebel Server. For details on these procedures, see *Siebel System Administration Guide*.

4 Enter the following command at the Server Manager command-line interface prompt:

start task for comp WfProcMgr server *Siebel_Server_Name* with ProcessName="UDA Batch Import", RowID="*Session_ID*"

where:

Siebel_Server_Name = Name of the Siebel Server.

UDA Batch Import = Name of the workflow process. This value is case-sensitive.

Session_ID = Session ID of the deployment session used to generate the export files.

NOTE: This session ID does not exist anywhere in the target database. It is the session ID used to export files. This value maps to an ini file generated during the export.

5 Check the details of the batch deployment by reviewing the source system ADM log files and EAI queue. See "Reviewing the ADM Data Migration" on page 41 for further information on these tasks.

NOTE: One import step reads in all previously generated files by the export session executed on the source system.

Reviewing the ADM Data Migration

Review an Application Deployment Manager (ADM) migration by:

- Monitoring the status field of the deployment session record
- Accessing the log file for the deployment session
- Reviewing the EAI queue

For information on deployment sessions and the process for migrating data, see "Process for Migrating Customizations Between Applications" on page 27.

For background information on ADM, see "About Application Deployment Manager (ADM)" on page 23.

Reviewing Deployment Session Status

The deployment session status field can have the values described in Table 4.

Table 4. Deployment Session Status Field Values

Status Value	Description
New	State of the deployment session after creation but prior to deployment.
Submitted	State of the deployment session while in process of deployment.
Deployment Completed	State of a complete and successful deployment. See "Accessing ADM Migration Log Files" on page 42 log file for further details on the deployment.
Deployment Failed	State of a failed deployment. See "Accessing ADM Migration Log Files" on page 42 log file for details on why the deployment was unsuccessful.
Incomplete Deployment	State of an incomplete deployment. Incomplete deployment occurs when there are multiple items within the session. One of these items encountered an error while others completed successfully. Users need to copy the session and retry. Successful items could be inactivated.
Validation Failed	State of a deployment session that cannot access the target system (either password or target path is incorrect). A session with this status can be deployed again once the error is corrected.

NOTE: With the exception of Validation Failed and New, all sessions must be copied or re-created to be deployed again.

Accessing ADM Migration Log Files

Access ADM migration log files to review or troubleshoot the migration of a deployment session. After a deployment session completes or fails, ADM creates a log file in the deployment session record. The log files provide details on any failures, as well as source and target environment information.

To access other migration logging in the EAI queue, see "Accessing ADM Migration Logging from EAI Queue" on page 43.

To access ADM migration log files

1. From the application-level menu, choose Navigation > Site Map > Application Deployment Manager screen.
2. From the link bar, click Deployment Sessions.
3. In the Deployment Sessions list, click the deployment session record of interest.
4. In the Log field, click on the log file number.
5. Review log file details in the Log view.

Accessing ADM Migration Logging from EAI Queue

Migration logging data is also logged by default to the EAI queue in case of errors. This data is accessible from the Administration - Integration screen of the target system. For further information on the EAI queue, see *Siebel Connector for SAP R/3*.

NOTE: A user property setting on the Session business component determines EAI Queue logging behavior: Never, Always, and On Error. File imports are always logged unless this property setting value is Never.

To access other ADM logging files, see "Accessing ADM Migration Log Files" on page 42.

To access ADM migration logging from EAI queue

1. In the target system application-level menu, choose Navigation > Site Map > Administration - Integration screen.
2. From the link bar, select EAI Queue.
3. Search on the Session ID number or the prefix *File_Import* to locate EAI queue logging information for data deployed from a file.

   The EAI queue item contains the actual message in the form of an XML attachment. Each data type has a separate queue item created.

Troubleshooting ADM Migration

This section provides guidelines for resolving Application Deployment Manager (ADM) problems.

To resolve the problem, look for it in the list of Symptoms/Error messages in Table 5.

Table 5. Resolving Application Deployment Manager Migration Problems

Symptom/Error Message	Diagnostic Steps/Cause	Solution
Log file error: `No change update is in effect.`	■ Siebel Server components are not synchronized. ■ Workflow Management (Workflow) component group disabled or off-line.	■ Synchronize Siebel Server components. ■ Enable or online the Workflow Management (Workflow) component group. For a description of these tasks, see *Siebel System Administration Guide*.
On some objects, errors are seen during deployment. Error depends on the object and the operation at that time (insert or update).	Users used in deployment must hold a position.	Add users to a position. For a description of this task, see *Security Guide for Siebel Business Applications*.
The value for the field Last Updated By does not match the user who is used during deployment	If deploying in asynchronous mode (default), the system fields on the target system (Created By, Last Updated By) will point to the user running the Server Request Broker (SRBroker) component and NOT the User ID that was entered on the source system.	
Cannot connect to target.	■ Errors in the EAI configuration file. ■ No UDADeploy subsystem.	■ Confirm entry in `eai.cfg`: `UDADeploy = UDADeploy` (under `[HTTP Services]`) ■ Confirm UDADeploy Subsystem under Enterprise Profile. For information on the Enterprise Profile, see *Siebel System Administration Guide*.

Table 5. Resolving Application Deployment Manager Migration Problems

Symptom / Error Message	Diagnostic Steps / Cause	Solution
Cannot deploy sessions using Server Manager command-line interface (srvrmgr program).	■ Command-line syntax errors. ■ Different User IDs used to create sessions and run srvrmgr program.	■ Make sure command-line syntax is correct. For information on using the srvrmgr program, See *Siebel System Administration Guide*. User creating session must be the same user running srvrmgr program session.
GUI error message: `The characters: '<?>' that are being inserted into column '<?>', are not compatible with the Server Database Charset. (SBL-DBC-00110).`	Codepage different between the source system and target system.	Change the codepage on the source or target database. The source must be compatible with the target.

4 Migrating Parameters Between Environments

Migrating Siebel Enterprise or Siebel Server parameter values from one Siebel application environment to another is a common requirement when going live to a new development, test, or production environment. The configuration upgrade utility, cfgmerge, facilitates the automatic migration of these parameters between applications.

This chapter includes the following topics:

About Migrating Parameters Between Environments

Migrating configured Siebel Enterprise and Siebel Server parameter values is a necessary task when updating from one Siebel environment to a newer or alternate Siebel environment. If undertaken manually, the process can incur data entry errors and extend downtime of the target Siebel environment. The configuration upgrade utility, cfgmerge, allows for the automation of the parameter migration process by comparing the source environment with the target environment and creating a parameter migration script, which includes documentation on the parameter differences between the two applications. The migration script documents the differences between environments, including noting parameters that cannot be updated automatically and identifying obsolete or modified parameters in the new environment. After a review, the migration script can then be applied on the target environment to update the parameters.

Use the automated migration process to:

- Update environments of the same software version (for example, updating a test environment from the development environment).
- Update environments with different software versions (for example, you can migrate parameter values between a 7.7 and a 7.8 environment).

NOTE: In all cases, the target environment must be 7.8 or later.

Migrating parameters between environments can be run in either:

- **Enterprise mode.** This mode migrates enterprise parameters, component definition parameters, and named subsystem parameters between environments.

- **Siebel Server mode.** This mode migrates Siebel Server parameters and Siebel Server component parameters between Siebel Server environments.

For procedures on migrating parameters between environments, see "Process for Migrating Parameters Between Environments" on page 48.

The cfgmerge utility runs on both Windows and UNIX. For further detail on this utility, see "About Cfgmerge Utility" on page 48.

The cfgmerge utility only migrates parameter values between environments. Other environment configurations, such as custom components, are not migrated to the target environment. For example, if you have a custom component in the source environment, you need to create a component definition of the same type, with the same name, in the target environment to migrate the parameter settings for that component. For information on creating component definitions, see *Siebel System Administration Guide*.

About Cfgmerge Utility

The cfgmerge program is a command-line utility that creates a parameter migration script after comparing two different Siebel environments. You run the cfgmerge utility as part of the task "Running Environment Comparison" on page 49. This task is a part of the overall "Process for Migrating Parameters Between Environments" on page 48.

The cfgmerge utility resides in the bin subdirectory of the Siebel Server root directory as the executable program cfgmerge.exe on Microsoft Windows or cfgmerge on UNIX.

The parameter migration script created after a cfgmerge utility execution contains parameter listings, analysis, and recommendations for migrating the parameters. For background information on the parameter migration script, see "About Parameter Migration Scripts" on page 51

Process for Migrating Parameters Between Environments

This topic lists the ordered tasks of the process for migrating parameters between environments. For background information on this process, see "About Migrating Parameters Between Environments" on page 47.

Perform the following tasks to migrate parameters between applications:

1. Run a comparison analysis between the two applications of interest using the cfgmerge utility. For information on this task, see "Running Environment Comparison" on page 49.
2. Review and edit the migration script, which results from running the application comparison. For information on this task, see "Reviewing and Editing a Parameter Migration Script" on page 50.
3. Run the parameter migration script on the target application to migrate the parameters. For information on this task, see "Running a Parameter Migration Script" on page 52.

Running Environment Comparison

This task uses the cfgmerge command-line utility to run a comparison between two applications and creates a parameter migration script, which documents the parameter differences between the two applications. In all cases, run the cfgmerge utility in the target environment, which must be version 7.8 or later.

Running an application comparison is a task in the "Process for Migrating Parameters Between Environments" on page 48.

For background information on the parameter migration process, see "About Migrating Parameters Between Environments" on page 47. For background information on the cfgmerge utility, see "About Cfgmerge Utility" on page 48.

To run the environment comparison

1. Before running the cfgmerge utility, make sure all component groups of interest are enabled on both the source and target application.

   For example, if you want to migrate component parameters for a server component in the Siebel Remote component group (alias Remote), make sure this component group is enabled on both the source and target environment.

2. Make a backup copy of the target application's `siebns.dat` file. For information on this GUI or command-line interface procedure, see *Siebel System Administration Guide*. Rename the backup copy to a unique value, for example, `target_siebns.dat`.

   The `siebns.dat` file is available in the Administration folder of the Siebel Gateway Name Server root directory.

3. Make a backup copy of the source application's siebns.dat file. Rename the backup copy to a unique value, for example, source_siebns.dat.

4. Move the `source_siebns.dat` file and the `target_siebns.dat` file copies to the bin subdirectory of the Siebel Server root directory, which contains the cfgmerge utility.

   NOTE: The cfgmerge utility does not require the Siebel application to be up or down when running.

5. Run the cfgmerge utility using the following commands to execute the application comparison. The cfgmerge utility can run a comparison in Enterprise mode or Siebel Server mode:

   - Running Enterprise-mode comparison:

     `cfgmerge -l` *`language_code`* `-i` *`source_siebns.dat`*`,`*`target_siebns.dat`* `-e` *`source_enterprise_name`*`,`*`target_enterprise_name`* `-o` *`output_file`*`.cmd`

   - Running Siebel Server-mode comparison:

     `cfgmerge -l` *`language_code`* `-i` *`source_siebns.dat`*`,`*`target_siebns.dat`* `-e` *`source_enterprise_name`*`,`*`target_enterprise_name`* `-s` *`source_server_name`*`,`*`target_server_name`* `-o` *`output_file`*`.cmd`

   NOTE: Do not include a space between the target and source parameter pairs.

   For details on the cfgmerge utility's flags and arguments, see Table 6

A successful execution results in the creation of a parameter migration script, which is saved by specifying an output file during command execution. Review this file to note the difference between applications. For further information on this task, see "Reviewing and Editing a Parameter Migration Script" on page 50.

Table 6. Siebel Cfgmerge Utility Flags

Flag	Arguments	Description
-e	*source_enterprise_name*, *target_enterprise_name*	Use this flag and two arguments to specify the source and target Siebel Enterprise Server names used for the application comparison. Make sure to include the comma after the first argument, but do not insert a space after the comma.
-i	*source_siebns.dat*, *target_siebns.dat*	Use this flag and two arguments to specify the source and target siebns.dat files. The siebens.dat file, stored in the Siebel Gateway Name Server, defines the configurations of an individual application. Make sure to include the comma after the first argument, but do not insert a space after the comma.
-l	*language_code*	Use this flag to set the language in the script file. The default language code is ENU. Make sure the appropriate language pack is installed before using this flag.
-o	*output_file*.cmd	Use this flag to specify a file name and path for the migration script, which is created as a result of execution of the cfgmerge utility and documents the application differences.
-s	*source_server_name*, *target_server_name*	Use this flag and two arguments to specify the source and target Siebel Server names used for the application comparison. Make sure to include the comma after the first argument, but do not insert a space after the comma.

Reviewing and Editing a Parameter Migration Script

This task provides information on reviewing and editing a parameter migration script, which is created as a result of an execution of the cfgmerge utility. This utility compares the differences between two environments.

Reviewing and editing a migration script is a task in the "Process for Migrating Parameters Between Environments" on page 48.

For background information on:

- Parameter migration process, see "About Migrating Parameters Between Environments" on page 47.
- Parameter migration scripts, see"About Parameter Migration Scripts" on page 51.

To review and edit a parameter migration script

1. Locate the parameter migration script specified by a cfgmerge utility execution.

   The migration script has an extension of CMD, and the default location of the script—that is, if no folder path is specified—is the same folder as the cfgmerge utility.

2. Open the parameter migration script with a text editor.
3. Review the results of the comparison analysis, and make edits to the migration script as appropriate by deleting or adding preceding semi-colons, which activate and deactivate commands respectively.

About Parameter Migration Scripts

A parameter migration script results after an execution of the cfgmerge utility, which compares parameter differences between applications. Migration scripts have the extension CMD and are named as part of the command to run the cfgmerge utility, see "Running Environment Comparison" on page 49 for further details on this task.

The resulting parameter migration script is composed of the following information:

- List of the source and target parameter values if they are different.
- Messages and recommend actions.
- Server Manager command-line interface (srvrmgr) commands to change the target environment's parameter values to match the source environment's value.
- Commented out srvrmgr commands (with a preceding semi-colon) if the utility recommends not to synchronize the values with the source environment.

You must open, review, and edit the migration script prior to running the script. For information on this task, see "Reviewing and Editing a Parameter Migration Script" on page 50.

Parameter migration scripts can act as a documentation record for an environment's configurations. The migration script files can be reused or reviewed for historical comparisons at a later date.

An example portion of a parameter migration script follows:

```
;  Component definition SCCObjMgr_enu
;
;  Parameter Actuate Server Report Server Host (ActuateReportServerHost)
;  Value on source system: acttest_winxp
;  Value on target system: actprod_hpux
;  Recommended action: retain target value
;  To apply value from source configuration, enable the next line
;  change param ActuateReportServerHost=actprod_winxp for compdef SCCObjMgr_enu
;
```

```
;  Parameter DB Multiplex - Min Number of Shared DB Connections (MinSharedDbConns)
;  Value on source system: 10
;  No value set on target system:
;  Recommended action: apply value from source
;  To keep the target configuration unchanged, comment out the next line
change param MinSharedDbConns=10 for compdef SCCObjMgr_enu
;
```

Running a Parameter Migration Script

This task provides information on running a parameter migration script created as a result of an execution of the cfgmerge utility. Running a migration script is a task in the "Process for Migrating Parameters Between Environments" on page 48. For background information on this process, see "About Migrating Parameters Between Environments" on page 47.

Run the parameter migration script using the following procedure and the Server Manager command-line interface program. For background information on the Server Manager command-line interface, see *Siebel System Administration Guide*.

Make sure to review and edit the migration script before running. For information on this task, see "Reviewing and Editing a Parameter Migration Script" on page 50.

NOTE: As a best practice, run the parameter migration script at times of low usage of the Siebel application.

Running the parameter migration script

1. Copy and save the reviewed and edited migration script in an accessible location for the Server Manager command-line interface program (srvrmgr) accessing the target application; that is, the application receiving the parameter update.
2. Log in to the srvrmgr program and set the program at either the enterprise or Siebel Server level depending on whether the migration script updates enterprise or Siebel Server parameters. For details on these srvrmgr commands, see *Siebel System Administration Guide*.

   NOTE: Make sure server components planned for parameter updates are enabled on the appropriate Siebel Server.
3. Run the migration script using the read command at the srvrmgr command prompt, which inputs commands from the script to the srvrmgr program.

   For example:

   ```
   srvrmgr>read Migration_Parameter_Script.cmd
   ```

   (You can also run the migration script when logging into the srvrmgr program by specifying the `i` switch and file name with the other login parameters.)
4. Check to make sure the parameters have successfully updated in the target application.

5 Rolling Out Updates to Mobile Clients

Rolling out application configuration changes or updates to mobile clients is the final step in going live to a new application environment. Use the Siebel Packager utility for this purpose.

This chapter includes the following topics:

About Rolling Out Updates to Mobile Clients

The Siebel Packager utility allows the Siebel administrator to roll out Siebel application configurations and customizations to Siebel Mobile Web Clients. Siebel Packager assembles the Siebel client executable program and other installed files, including your custom configuration, into a customized installation package.

Siebel administrators can use these installation packages when installing Siebel Business Applications for the first time or when upgrading from previous versions. Optionally, you can package the Siebel client installation as a single, self-extracting archive file.

CAUTION: Do not use Packager to distribute Siebel patch releases. Packager is designed for use with full releases. It does not include the necessary functionality to install a Siebel release that depends on the existence of a previous release.

For details on the process of rolling out customizations using the Siebel Packager utility, see "Process of Rolling Out Updates to Mobile Clients" on page 54.

Distributing Siebel Client Packages

After the Siebel client installation has been packaged, the package can be distributed to your users in several ways. For details, see "Making Your Customized Installer Available to End Users" on page 60.

- **Siebel Anywhere.** Distribute and execute the package automatically as a Siebel Anywhere kit. For further information on Siebel Anywhere, see *Siebel Anywhere Administration Guide*.
- **CD-ROM or DVD.** Distribute the package to end users on CD-ROMs or DVDs.

- **LAN, WAN, VPN, or modem.** Distribute the package across a LAN, WAN, VPN, or modem. Do this directly or using third-party software.
- **Other methods.** Distribute the package by email or FTP to end users.

General Process of Creating a Package

The Packager utility creates a standard installation package in two steps:

1. Gathers the Siebel components and files you specify, copies the standard InstallShield components into the client installer directory, and creates a packing list used by InstallShield.
2. (Optional.) Packages the Siebel client installer (prepared in the previous step) into a self-extracting archive, which, when executed, automatically decompresses and starts the Siebel client installer.

Process of Rolling Out Updates to Mobile Clients

This topic lists the ordered tasks of the process for rolling out customizations to mobile clients using Siebel Packager. For background information on this process, see "About Rolling Out Updates to Mobile Clients" on page 53.

To roll out customizations using Siebel Packager:

1. Make preparations before running the Siebel Packager utility. See "Preparing to Use the Siebel Packager Utility" on page 54 for information on this task.
2. Run the Siebel Packager utility. See "Running the Siebel Packager Utility" on page 55 for information on this task.
3. Distribute the update to mobile clients. See "Making Your Customized Installer Available to End Users" on page 60 for information on this task.

Preparing to Use the Siebel Packager Utility

Before using the Siebel Packager utility, follow the steps in this topic. This task is a part of the "Process of Rolling Out Updates to Mobile Clients" on page 54.

For background information on Siebel Packager and rolling out customizations to mobile clients, see "About Rolling Out Updates to Mobile Clients" on page 53.

For instructions on installing Siebel clients, see the *Siebel Installation Guide* for the operating system you are using.

NOTE: This chapter refers to the Siebel client root directory, such as `C:\program files\Siebel\7.7\Web client`, as *SIEBEL_CLIENT_ROOT*.

To prepare to use the Packager utility

1 Perform a Siebel client installation on the computer on which you will run the Packager utility.

The Packager uses the files from this model client installation (or another client installation, as specified when running the Packager) in creating the installation package.

During Siebel client installation, select *Typical*, or select *Custom* and make sure to select the Packager Utility option.

NOTE: If two Siebel clients are installed on the same computer, where one is used as a Mobile Web Client and the other is a master installation for Packager, make sure to exit the SQL Anywhere engine before running Packager. An installation that has an initialized local database should never be used as a master installation for creating packages.

2 Customize the model Siebel client installation so that it is identical to how you intend to package it. When you create the custom installer, the Packager utility reproduces this model installation.

If you have custom Siebel repository file (SRF) or configuration files (CFG), report files, Web templates, or other changes or additions, copy them to the appropriate subdirectories under *SIEBEL_CLIENT_ROOT*, or under the root directory of another installation that you will use to create the custom installation package. Make sure your custom files are the latest version so they pass version check. For more information please see FAQ 1361 on Siebel SupportWeb.

NOTE: The Packager utility can package only those files that reside in the *SIEBEL_CLIENT_ROOT* directory.

3 Make sure that you have sufficient free disk space on the computer on which you are installing the Packager utility and will create packages.

During the packaging process, the Packager utility temporarily requires at least:

- Three times the amount of disk space required by the Siebel Mobile Web Client software you are packaging, and
- Twice the disk space required by the third-party software (provided with the Siebel Business Application) that you are packaging.

Running the Siebel Packager Utility

This topic describes how to run the Siebel Packager utility. The Siebel Packager utility wizard guides you in creating the custom Siebel client installer. This task is a part of the "Process of Rolling Out Updates to Mobile Clients" on page 54.

For background information on Siebel Packager and rolling out customizations to mobile clients, see "About Rolling Out Updates to Mobile Clients" on page 53.

NOTE: You must run the Packager once for the base Siebel client modules (BASE option), and once for each language pack you want to include. To include elements in the same package, you must specify the same package name each time.

To run the Siebel Packager utility

1 From the Windows Start menu, select Programs > Siebel Web Client 7.*x* > Siebel Packager.

The Siebel Client Packager wizard launches and the Choose Setup Language window appears.

2 Choose the language in which to conduct the rest of the Siebel Packager procedure and click OK. The Directory Definition window appears.

3 Specify the following values:

- **Package.** The name of the package. This is used as the name of the self-extracting archive file (if you create one) and as the name of the subdirectory under *SIEBEL_CLIENT_ROOT*\packager\temp in which the custom installer is created.
- **Siebel Client.** The root-level directory of the Siebel client installation that is included in the custom installation. Accept the default, represented in this book as *SIEBEL_CLIENT_ROOT*.
- **Language Packs.** Specify BASE or specify an installed language pack (for example, ENU for U.S. English). If you want to include language packs in the customized installer, select a language pack.

  NOTE: Siebel Packager does not work with custom languages.

4 At the bottom-right of the window, choose Full Install or Patch Install, based on your desired goal:

- **Full Install.** Intended for full installations of Siebel Business Applications. This performs an entire installation, using the parameters in the siebel.ini file. For more information on the siebel.ini file, see Siebel SupportWeb.
- **Patch Install.** Copies only the packaged files, preserving the same directory structure as the source. Typically, this is used with an existing installation not requiring further customizing. When you run a patch installer, it prompts only for the existing installation directory.

5 Click Next and the Module Definition window appears.

6 In the Module Definition window:

a Choose the Siebel modules to be included in the custom installation package.

A list of possible modules appears in the Modules list on the left. For explanations of some of these modules, see Table 7 on page 58.

b If you want to include or exclude a template, select an item in the Modules list. Notice that *.* appears in the Include Templates box on the right.

The Include Templates and Exclude Templates boxes allow you to set the filters used to include or exclude files for each selected component. The default Include filter is *.*, which includes all files.

Include Templates also has an Include Subdirectories check box to indicate whether files in subdirectories for these components are included.

NOTE: You do not need to modify Include Templates and Exclude Templates for a typical installer.

- If you want to prevent a directory from being created for a particular module, select the module and click Remove.
- If you want to add modules that are located in the *SIEBEL_CLIENT_ROOT* directory but do not appear in the Modules list, click Add and specify the path of the module.

To create a required directory without any files, select that module from the Modules list and, under Include Templates, click Remove.

NOTE: When preparing a full installation, do not remove any module or components unless you know they will not be needed. See module descriptions in Table 7 on page 58.

c Click Next and the Packaging screen appears.

7 To create the custom installer, click Start.

The Packager utility displays progress information while the Packager executes and creates the package.

8 (Optional.) After this process is finished, you can further customize the behavior of the packaged installer by editing the siebel.ini file. To do so, click the button labeled Edit siebel.ini. For more information on this procedure, see "Customizing Siebel.ini Files" on page 61 and Siebel SupportWeb.

9 (Optional.) In order to bypass the installer screen labeled Choose Setup Language, edit the `setup.ini` file, and change the value of the parameter EnableLangDlg from Y to N.

10 (Optional.) If you want to package the custom installer into a self-extracting archive, click Next.

NOTE: If you do not want to perform this step at this time, you can do it later by running the `selfex.bat` file in the directory *SIEBEL_CLIENT_ROOT*\Packager\Temp*package_name*.

11 (Optional.) In the Self-extracting Archive window, if you are producing an installer for BASE, click Start to package the self-extracting archive.

The Packager creates the archive as a single executable file called setupex.exe in the directory *SIEBEL_CLIENT_ROOT*\Packager\Temp*package_name*\selfex. This step may take some time to complete, depending on the processing speed of the computer you are using. Verify the location of the executable file after the process has completed.

About Siebel Modules for Packaging

The Siebel Mobile Web Client consists of the installable modules described in Table 7. These modules correspond to the subdirectories under the *SIEBEL_CLIENT_ROOT* directory. Additional subdirectories, other than those shown here, may apply for your client installations.

NOTE: When you create a package, you can decide which modules or files to include or exclude in the package for distribution to end users. Do not remove any modules or component subdirectories or files unless you know they will not be needed. Verify that your Siebel directory structure contains the files you require, including any modified files or other configuration changes.

Table 7. Siebel Business Applications Modules for Packaging

Component	Description
ACTUATE	Actuate-related files for Reports, located in the *SIEBEL_CLIENT_ROOT*\actuate directory.
BIN	Siebel executable files (binaries) located in the *SIEBEL_CLIENT_ROOT*\bin directory, including the required DLL files, configuration files, and executable files such as siebel.exe. If you have customized the configuration files, replace the default configuration files in this directory with your own before you start the Packager utility. **NOTE:** When you create a package, include this module and all components in the bin directory, except for the user preferences file, *user_ID&SiebelAppname*.spf, and the session file, siebel.ses.
CHARTS	ChartWorks components for generating charts, located in the *SIEBEL_CLIENT_ROOT*\charts directory.
FONTS	Barcode font files, located in the *SIEBEL_CLIENT_ROOT*\fonts directory.
ISSTEMPL	Template files for the SIS (Siebel Interactive Selling) CDA application, located in the *SIEBEL_CLIENT_ROOT*\isstempl directory.
LOCAL	Location of the local database, local Siebel File System, and docking files for Siebel Remote, located in the *SIEBEL_CLIENT_ROOT*\local directory. **NOTE:** Include this module in order to create the local directory when you create a package for a full installation. However, you should not initialize a local database before you create the package. Each local database is unique to an individual user and should not be packaged.
LOCALE	Language-specific files, located in the *SIEBEL_CLIENT_ROOT*\locale directory. **NOTE:** Include this module and all components in the locale directory when you create a package for a full installation.

Table 7. Siebel Business Applications Modules for Packaging

Component	Description
LOG	Log files from Siebel client operations (such as synchronization), located in the *SIEBEL_CLIENT_ROOT*\log directory. **NOTE:** Include this module in order to create the log directory when you create a package for a full installation.
MSGTEMPL	Message template files used by the Siebel client, located in the *SIEBEL_CLIENT_ROOT*\msgtempl directory. **NOTE:** Include this module and all components in the msgtempl directory when you create a package for a full installation.
OBJECTS	Object configuration template files (configured objects), located in the *SIEBEL_CLIENT_ROOT*\objects directory—the precompiled SRF file to distribute to end users. The objects directory must contain at least one SRF file before you start the Packager utility. **NOTE:** Include this module and all components in the objects directory when you create a package for a full installation.
PACKAGER	**NOTE:** Do not include the Packager utility itself when you create a package for distribution to end users.
PATCH_BACK	Removes modules that were applied as part of a patch. **NOTE:** Do *not* include this module when you create a package for distribution to end users.
PUBLIC	HTML, help, JavaScript, image, and other files for the Siebel client, located in the *SIEBEL_CLIENT_ROOT*\public directory. **NOTE:** Include this module and all components in the public directory when you create a package for a full installation.
REPORTS	Report template files located in the *SIEBEL_CLIENT_ROOT*\reports directory. If you have created your own reports, replace the standard report templates in this directory with your own, or add your own, before you start the Packager utility. **NOTE:** Include this module and all components in the reports directory when you create a package for a full installation.

Table 7. Siebel Business Applications Modules for Packaging

Component	Description
SAMPLE	Location of the Sample Database and sample Siebel File System, located in the *SIEBEL_CLIENT_ROOT*\sample directory—if you have installed the Sample Database. **NOTE:** Generally, you would not distribute the Sample Database to your end users, who will access a local database (Mobile Web Client). You may instead decide to distribute a separate client installer package that includes the Sample Database. If you do this, include the Sample Database, sample configuration files that refer to the Sample Database (located in *SIEBEL_CLIENT_ROOT*\bin*LANGUAGE*), and sample SRF files (located in *SIEBEL_CLIENT_ROOT*\objects).
SQLTEMPL	SQL template files, located in the *SIEBEL_CLIENT_ROOT*\sqltempl directory.
TEMP	Working report files, located in the *SIEBEL_CLIENT_ROOT*\temp directory. **NOTE:** Include this module in order to create the temp directory when you create a package for a full installation.
UPGRADE	Siebel Anywhere upgrade files retrieved by the user, located in the *SIEBEL_CLIENT_ROOT*\upgrade directory. **NOTE:** This module is applicable only to upgrades, not to new installations.
WEBTEMPL	Siebel Web templates, located in the *SIEBEL_CLIENT_ROOT*\webtempl directory. **NOTE:** Include this module and all components in the webtempl directory when you create a package for a full installation.

Making Your Customized Installer Available to End Users

After you have tested your customizations and are satisfied with the client installer you have created, make your customized installer available to end users. This task is a part of the "Process of Rolling Out Updates to Mobile Clients" on page 54.

For background information on Siebel Packager and rolling out customizations to mobile clients, see "About Rolling Out Updates to Mobile Clients" on page 53.

You can distribute your customized installer to end users using one of the methods described in the following topics:

- "Deploying the Installer Using Siebel Anywhere" on page 61
- "Deploying the Customized Siebel Client Installer" on page 61

The suitability of each method will depend on many factors, including available network bandwidth.

Deploying the Installer Using Siebel Anywhere

You can distribute and execute the customized Siebel client installer automatically as a Siebel Anywhere kit.

User access to a Siebel Anywhere kit requires an existing installation of the Siebel client and administrative rights on the user's machine. Therefore, you can use Siebel Anywhere for upgrades, but not for an initial deployment or new installations.

For more information on using Siebel Anywhere, see *Siebel Anywhere Administration Guide*.

Deploying the Customized Siebel Client Installer

For new installations or for upgrades, you can distribute the customized Siebel client installer to end users on CD-ROM or local area network (LAN). You can also distribute the customized Siebel client installer across a WAN or VPN, using a modem, by email, or by FTP.

The customized Siebel client installer may optionally be in the form of a self-extracting archive file. Packaging the installer as a self-extracting archive is generally best for distribution methods such as using a modem, email, or FTP.

To distribute a customized Siebel client installer on a CD-ROM or network

1. Perform the following step, as applicable:
   - *CD-ROM:* Place the self-extracting archive file (setupex.exe) or installer package directory onto a CD-ROM, then distribute the CD-ROM to your users.
   - *Network:* Place the self-extracting archive file (setupex.exe) or installer package directory in a network-accessible directory, such as on a LAN. Make sure that all users have access to this directory.
2. Notify your users how to access the installer package and initiate the installation process. Optionally, users may need to copy files to their local machine, such as if you packaged a self-extracting archive file.

   The procedure users will follow varies according to how you distributed the package, how you created the package, and whether the package is a self-extracting archive file. Users can install a self-extracting archive package by running the archive file.

Customizing Siebel.ini Files

The siebel.ini file controls the behavior of the Siebel client installation. Separate versions of this file are created for packages for a base installation and those for language packs. The siebel.ini file is located in the following directories, where *package_name* is the name of your package:

- *SIEBEL_CLIENT_ROOT*\packager\temp*package_name* (for BASE)

- *SIEBEL_CLIENT_ROOT*\packager\temp*package_name**LANGUAGE* (where LANGUAGE is the subdirectory for the applicable language pack, such as ENU for U.S. English)

NOTE: The value of the FolderName parameter, described in "Key Parameters in Siebel.ini File" on page 62, must be the same in all siebel.ini files for the same package.

Review all applicable siebel.ini files and customize the files as necessary to make sure the client installation uses the correct settings for your specific environment.

NOTE: If you customize the siebel.ini file, do so at Step 8 in "Running the Siebel Packager Utility" on page 55. This step occurs after you create the package itself, and before you (optionally) create a self-extracting archive file. If you customize the siebel.ini file before you create the package, those changes will not become part of the customized installer.

The siebel.ini file determines all of the parameters used by the client installer, including the following:

- Third-party software programs and versions that are required on the client computer
- System settings that improve performance
- Configuration of data sources
- Which installation screens end users see when they run the Siebel client installer
- Which shortcuts (icons) are created upon installation

Instructions for modifying the siebel.ini file can be found in the file itself. Use a standard text editor to review and edit siebel.ini.

Key Parameters in Siebel.ini File

The following parameters, located in different sections in the siebel.ini file, are generally already set appropriately, based on performing the master installation. Review these settings and modify those that require it.

- Set the `FolderName` parameter in the `[Defaults]` section for each language pack to the same value as the `FolderName` parameter in the `[Defaults]` section for the base installation. If you do so, the relevant shortcuts (icons) will be delivered to the same location under the Windows Start menu.
- Set the `RootDirectory` parameter in the `[Defaults]` section to the installation location on the target system—for a full, language pack, or patch (maintenance release) installation. By default, the parameter value is populated by the Packager utility to correspond to the staging location where you built the package—for example, `C:\sea752\client`. You can leave this value as is, or set it to another location on the target system.
- Add and set the parameter `NoFolder` to `No` in the `[Startup]` section to suppress the creation of the startup folder at install. Set this parameter in both the base and language siebel.ini file.
- Do not enclose the `DockConnString` parameter value in double quotes.
- Do not modify the parameter `AppServer` in the `[Startup]` section.

- If the SystemDSN parameter is set to no, the installer creates single-user data sources. This type of data source is visible only to the user who installed it.

  If SystemDSN parameter is set to yes, the installer creates system data sources, which are shared by all users who log into that particular machine. You must have administrator privileges to create a system installation.

- You can use the addLanguages parameter to enable users to add language packs to an existing Siebel client installation.

  If addLanguages (in the [Dialog] section) is defined and set to yes, when the user installs, a single dialog box appears, listing all installed instances of the Siebel client. The user can choose a Siebel client instance to add a language pack to, or click Next to install a new Siebel client.

  If addLanguages is set to no, then an installation directory must be specified using the RootDirectory parameter.

  For a package installing a patch, do not use the addLanguages parameter.

NOTE: Do not enable any dialog boxes in a packaged installation, except the one described above for adding languages. Enabling any other dialog boxes in the Packager is redundant, because the necessary user input has already been captured by the initial installation. (If other dialog boxes are enabled, user input is ignored.)

About Major Sections of the Siebel.ini File

The major sections of the siebel.ini file are as follows:

[Startup]—This section defines values needed for setup initialization. Examples include version number, patch install, application name, and so forth.

[AppCollision]—This section defines the file which defines a product. This file is then used to validate that no installations try to overwrite one another.

[StartupFiles]—This section defines third-party startup (.ini) files that may need to be updated or expanded during installation.

[DeleteFiles]—Defines any files that need to be deleted prior to file delivery.

[Module]—Defines component descriptions for use in the user interface.

[Module.Configuration]—This section defines what components or features to assign to a given setup type.

[Module.Destination]—This section defines where a component or feature should be installed on the end user's system.

[Regsvr32]—This section defines the files to register on Windows when using the regsvr32 utility.

[Prerequisites]—This section defines prerequisites for installation to proceed.

[Dialog]—This section defines what dialog boxes should be enabled or disabled in the installer at run time.

[Defaults]—This section defines default values to be used in the user interface of the installer, or the prompt to use if a dialog box is turned off.

[Behavior]—This section defines general installer behavior, such as whether to abort or continue on a failed condition.

[RunAfter]—This section defines the programs or functions to run or call after the installation is complete.

[CustomUninstall]—This section defines programs or what functions should be run or called during uninstallation.

[Icons]—This section defines what shortcuts (icons) to create on the end user's system.

About Siebel.ini File Hierarchy and Organization

This section describes the hierarchy and organization of the siebel.ini file.

Sections Containing Child Sections

In sections like the following, a child subsection provides additional information about the parent section. Parent and child sections take the following format:

```
[Parent_Section_Name]
Child_Section_Name = Value
```

where *Child_Section_Name* is a key in the parent section.

```
[Child_Section_Name]
Key = Value
Key = Value
```

The key in the parent section tells the installer whether that element is enabled. If it is enabled, the installer looks to the child section whose name is the key from the parent.

In the following example, the [AppCollision] section is traversed. The installer finds a key (GtwySrvr) and determines if that check should be enabled. In this case, the check would be enabled if a Siebel Gateway was selected during installation. If so, the installer looks for the definition. The previous key (GtwySrvr) is redefined as a section, which then defines the behavior.

```
[AppCollision]
GtwySrvr = $(Gateway Selected)=yes

[GtwySrvr]
Description = Siebel Gateway
File = gtwysrvr\bin\namesrvr.exe
```

Sections Without Child Sections

In sections like the following, all necessary information for the key is contained in the value:

```
[Section_Name]
Key = Value
```

In the following example, the installer displays a welcome dialog box. All necessary information for the key is contained in the value; no child section is required.

```
[Dialog]
Welcome = yes
```

Testing an Installer After Customizing the Siebel.ini File

After you customize the siebel.ini file, you should test the installer.

To test an installer

1. Finish modifying the siebel.ini file.
2. Run install.exe from the *SIEBEL_CLIENT_ROOT*\packager\temp*package_name* directory on the network installation server.
3. Repeat Step 1 and Step 2 until the siebel.ini file is configured to achieve the desired installation.
4. Test any self-extracting archive file you create.

NOTE: If you test a self-extracting archive file on the same machine on which you generated the package, the installer will overwrite the Siebel client directory. To install to a different location, specify an installation directory using the `RootDirectory` parameter in the siebel.ini file. Or, run the archive file on a different machine.

Index

W