Banking Application Build Process Guide

Version 2004.5, Rev. C

February 2005

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

Printed in the United States of America

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

Proprietary Information

Siebel Systems, Inc. considers information included in this documentation and in Siebel eBusiness Applications Online Help to be Confidential Information. Your access to and use of this Confidential Information are subject to the terms and conditions of: (1) the applicable Siebel Systems software license agreement, which has been executed and with which you agree to comply; and (2) the proprietary and restricted rights notices included in this documentation.

Contents

1 What’s New in this Release

The following changes have been introduced in the Banking Application Build Process Guide, Version 2004.5, Rev C:

Topic	Description
Common Software Resources and Build Pack Files, page 10.	The `\offline` and `\database` directories have been added to the Common Software Resources and Build Pack Files. The `\offline` directory contains the Offline source code and the ANT scripts for compiling and building the Offline Server. The `\database` directory contains the database dump file and the reset scripts for the database.

2 Banking Application Components

Banking Application Components

The Siebel Retail Finance Banking Application consists of the following main components:

- A single set of server-side business logic, shared between two products: Branch Teller and Internet Banking. This business logic is deployed within the EAR file that deploys the Banking Application as a set of EJB jar files.
- A Java Swing-based front end for the Branch Teller Client and the Branch Teller Administrator Client, deployed within the EAR file as Java WebStart-enabled applications.
- A JSP-based front end for the Internet Banking product. This is co-deployed in the same EAR as the server-side business logic.
- A Branch Offline Server.
- Web applications for Entitlements and MCA Services administration functionality.

About the Build Process

There is a single unified build process for building both Branch Teller and Internet Banking because they share a common server side; the front end is the only difference between the two products from a build perspective.

The build process for building the EAR is the same regardless of whether a license is held for Branch Teller, Internet Banking or both. For example, if just Branch Teller has been licensed the Internet Banking collateral will not be available and the build process will recognize this and skip this stage of the process.

Any build process steps which are specific to a particular product are noted in this guide.

3 Build Process Prerequisites

It is a prerequisite that the Siebel Retail Finance Banking Application for the appropriate target application server/database platform has been extracted from the appropriate Banking Application pack so that required directory structures are created. Please refer to the *Siebel Retail Finance Banking Application Installation Guide* for details on the Banking Application packs available for the supported platforms in the current release.

The build process is only currently supported on the Windows platform. The Banking Application pack will have been extracted to d:\siebel.

Install the Source Code

Prior to using the build process the build environment must be prepared. This involves extracting the source code, supports, and build files from the versions of the Common Software Resources and product Build Pack CDs, appropriate to the product or platform for which they are being built.

Resources for Building Branch Teller on WebLogic

The jar **SRFBankingApplicationCommonSoftwareResources.jar** from the Siebel Retail Finance Common Software Resources CD should be extracted to d:\. The Build Pack for Branch Teller appropriate to the WebLogic platform should be extracted to the same path: d:\ i.e. extract **SRFBranchTellerBuildPack.jar** from the Siebel Retail Finance Branch Teller Extension Pack (WebLogic) to d:\.

Resources for Building Branch Teller on WebSphere

The jar **SRFBankingApplicationCommonSoftwareResources.jar** from the Siebel Retail Finance Common Software Resources CD should be extracted to d:\. The Build Pack for Branch Teller appropriate to the WebSphere platform should be extracted to the same path: d:\ i.e. extract **SRFBranchTellerBuildPack.jar** from the Siebel Retail Finance Branch Teller Extension Pack (WebSphere) to d:\.

Common Software Resources and Build Pack Files

The files contained in the Common Software Resources and Build Packs will be extracted to the following Standard Directory structure for the Banking Application:

Directory	Description
`\siebel`	The root folder.
`\3rdParty`	Contains third-party libraries required for supporting the Siebel Retail Finance Banking

Directory	Description
	Application. Additional third-party jar files must be installed which are not included with this distribution—see below for further details.
\Branch	Contains all server-side implementation layer source code, deployment descriptors and resources for the Siebel Retail Finance Banking Application. It also contains ANT scripts to compile the Siebel Retail Finance Banking Application.
\branch-common	Contains common branch code.
\BranchAdministrationClient	Contains all the front-end code to run the Branch Teller administration application. Also contains an ANT script to compile the war file. [1]
\Branch Client	Contains all the front-end code to run the Branch Teller client. Also contains an ANT script to compile the war file. [2]
\Build	Contains the ANT scripts and other required resources for assembling and compiling the Siebel Retail Finance Banking Application and for generating a deployable EAR.
\Common	Contains the common jar files and resource files that are shared between Siebel Retail Finance Banking Application products. See the following section for further details.
\deploy	Contains the pre-built deployable Siebel Retail Finance Banking Application EAR file Siebel.ear for deployment and data to populate the application database.
\Entitlements	Contains the Entitlements source code, Swing code/JSPs and deployment descriptors. It also contains ANT scripts to build Entitlements.
\eBanking	Contains the Internet Banking source code, and JSPs. It also contains ANT scripts to build Internet Banking[3].
\Offline	Contains the Offline source code. It also contains ANT scripts to compile and build the Offline Server.
\database	Contains the database dump file as well as reset scripts for the database.

[1] Only present if Branch Teller is licensed.

[2] Only present if Branch Teller is licensed.

[3] Internet Banking front-end code will only be present if the Internet Banking product has been licensed.

Two third-party libraries that are not in the release distribution, comm.jar and win32com.dll, are required to compile the Siebel Retail Finance Application and enable some peripheral devices. See the *Siebel Retail Finance Banking Application Installation Guide* for instructions on how to obtain these libraries. Once downloaded these files must be copied to the folder \siebel\3rdParty\lib\comm\Win32. You must create this folder if it is not already present.

All the source code and supports required for a Banking Application build should now be in place.

Classes jar files in the Standard Directory Structure

The Siebel Retail Finance Common Software Resources pack contains several important classes jar files contained in the *\siebel\Common\lib* folder.

The jar files in this folder (organized by category) include the following:

General support classes jar files

The following jar files contain compiled classes only—no source code. For ease of future support these jar files must be placed without alteration in the classpath of the extended Siebel Retail Finance Banking Application when it is being extended and deployed.

Support Classes JAR Files	Description
mca.jar	This is the jar of MCA Services classes
core-bos.jar	This jar file contains the core and sector layer classes for all Siebel Retail Finance Banking Application entity beans
bfa-utils.jar	This jar file contains utility classes
statemachine-ext.jar	This jar file contains extension classes for the StateSoft Statemachine framework supporting the Screen Orchestrator tool in the Financial Transactions WorkBench
entitlements-war-support.jar	This jar file contains controllers for the Statemachine in Entitlements
mca-eabmq.jar	This jar file contains EAB and MQ Series connector helpers

Solutionset layer classes JAR files

The following jar files contain compiled classes only—no source code. For ease of future support these jar files must be placed without alteration in the classpath of the extended product when it is being deployed.

Solutionset Layer Classes JAR Files	Description
branch.jar	This jar file contains all server-side classes, including EJBs, Parameter Objects, Constants and utility classes, for the

Solutionset Layer Classes JAR Files	Description
	solutionset layer of the Siebel Retail Finance Banking Application codebase
entitlements.jar	This jar file contains solutionset level classes from the Entitlements sub-module
branch-common-fe.jar	This jar file contains all common FE classes used between the Branch Teller client and the Branch Administration Client
branch-common.jar	This jar file contains all common classes used between the Branch Teller client and the Branch Server side
offline.jar	This jar file contains solutionset level classes from the Offline sub-module

Domain layer classes jar files

There are only compiled classes in these jar files (no source code). As these are domain level jar files the classes in these jar files will be altered when extensions are made.

Domain Layer Classes JAR Files	Description
branch-impl.jar	This jar file contains the domain layer classes for all server-side Siebel Retail Finance Banking Application classes: Entity, Session, Parameter/Value/Factory objects, and non-BFA utility classes. Some or all of the classes in this jar file may be required for deployment of the generic or extended Siebel Retail Finance Banking Application
entitlements-impl.jar	This jar file contains domain-level classes from the Entitlements sub-module
branch-common-impl.jar	This jar file contains all common domain layer classes, shared between the Branch Teller client and the Branch Server side

Signed jar files in the release

A number of jar files supplied with this release have been signed. Signing has been carried out for jar files that contain compiled code pertaining to the Core, Sector, and Solutionset layers of the four-tier structure (the layers that should not be altered on a customer installation). This is intended to aid the verification of jar versions. Signing a jar file prevents any changes being made to it. This means that Siebel Systems can be sure that the version of a jar file being used on a customer site corresponds exactly to the version originally supplied.

The following jar files have been signed as they should not be modified in customer implementations:

```
/siebel/Common/lib/bfa-utils.jar
```

```
/siebel/Common/lib/branch.jar
```

```
/siebel/Common/lib/core-bos.jar

/siebel/Common/lib/entitlements.jar

/siebel/Common/lib/mca.jar

/siebel/Common/lib/mca-eabmq.jar

/siebel/Common/lib/statemachine-ext.jar

/siebel/Common/lib/entitlements-war-support.jar

/siebel/Common/lib/offline.jar

/siebel/Common/lib/branch-common-fe.jar

/siebel/Common/lib/branch-common.jar
```

The following jar files have not been signed as they may be modified in customer implementations:

```
/siebel/Common/lib/branch-impl.jar

/siebel/Common/lib/entitlements-impl.jar

/siebel/Common/lib/branch-common-impl.jar
```

Server-side source code in the release

Server-side source code is provided for domain layer classes only.

The source code for the domain layer is provided as Java files in the various source folders in the Standard Directory Structure. The locations used are the locations expected by the build files that are part of this release.

Teller, Internet Banking and Entitlements state chart and supports

The Branch Teller client application is implemented as a Swing application using the Screen Orchestrator tool. This tool uses a product state chart that can be extended onsite in client customizations where the Screen Orchestrator tool is being used. The statechart XML files and supports for the Branch Teller Client are located in the `\siebel\BranchClient\statechart` subfolder in the Standard Directory Structure. The statechart XML files and supports for the Branch Teller Administration Client are located in the `\siebel\BranchAdministrationClient\statechart` subfolder in the Standard Directory Structure.

The statechart XML files and supports for Internet Banking are located in the *\siebel\eBanking\JSPs* subfolders in the Standard Directory Structure when an Internet Banking Extension Pack has been extracted.

The statechart XML files and supports for Entitlements are located in the *\siebel\Entitlements\Source\JSPs* subfolders in the Standard Directory Structure.

The statechart XML files and supports for Internet Banking are located in the `\siebel\eBanking\Source\JSPs` subfolders in the Standard Directory Structure.

4 WebLogic 8.1 Build Process

To build the Retail Finance EAR you must build the projects in the order they are listed below:

`Entitlements`

`Branch` (includes `internetBanking` and `branch-common` projects)

`BranchAdministratorClient`

`BranchClient`

`Build`

This is automatically catered for in the build scripts supplied in the product extension packs.

Prerequisites for WebLogic Build Process

In order to run the WebLogic build process it is necessary to have WebLogic 8.1.0 installed on the build machine.

In order to build an EAR for WebLogic, the WebLogic variable in `\siebel\Common\build.properties` must be uncommented (remove the # symbol) and pointed at the user's WebLogic Home. The webSphere.home variable must be commented out. As the WebLogic build process as supplied is database independent, the target database variable must be commented out. Below is an example configuration for building a WebLogic EAR for Oracle:

```
weblogic.home=/bea/weblogic81/server
#websphere.home=C:/Progra~1/WebSphere/AppServer
#target.database.db2=true
target.database.oracle=true
```

The following settings also must be set to non-empty textual values in the properties file in order to sign the jars and war files (signing is needed for the front-end war files that are deployed using WebStart):

```
alias=your keystore alias
storepass=your keystore password
```

```
companyName=your section
operatingUnit=your unit
organisation=your company
country=your country.
```

Entering valid values (for instance, some values are required to be at least six characters in length) for these settings will sign the jars/war files with the credentials supplied.

Automated WebLogic Build Process

The \siebel\Build folder contains a Windows script called JavaPrompt.vbs that creates a command prompt that has the classpath and path variables configured to invoke the Retail Finance Banking Application Build Process. There is a build script within this folder called build-all.xml. Running this script will build all the projects in the correct order and create the EAR file, which can be found in \siebel\Build\release.

Double-click on the JavaPrompt.vbs file in \siebel\Build to open a command prompt.

Enter the following at the command prompt:

```
D:\siebel>cd Build

D:\siebel\Build>ant –f build-all.xml
```

This will build the projects in the order above and finally create the EAR. This EAR is located in \siebel\Build\release\siebel.ear.

Building Projects Individually

If however the user wishes to build each project individually, then the following guide can be used to build each project.

NOTE: The MCA project is delivered precompiled and will not be rebuilt during the build. The precompiled EJB files and BankFrameMCA.war are found in the subfolders at siebel\Build\resource\mca. This project does not get rebuilt during the build process.

Building Entitlements

Double-click on the JavaPrompt.vbs file in \siebel\Build to open a command prompt.

Enter the following at the command prompt:

```
D:\siebel>cd Entitlements

D:\siebel\Entitlements>ant
```

This will start the ANT build process, which compiles all the source code and then creates EJB jar files.

The compiled code is stored in D:\siebel\Common\lib\entitlements-impl.jar.
The compiled EJBs are stored in D:\siebel\Build\staging\ejb.
The Web application is stored in D:\siebel\Build\staging\war.

Building the SRF Banking Application (Branch)

To build the Branch project:

Double-click on the JavaPrompt.vbs file in \siebel\Build to open a command prompt.

Enter the following at the command prompt:

```
D:\siebel>cd Branch

D:\siebel\Branch>ant
```

This will start the ANT build process, which compiles all the source code and then creates EJB jar files.

If the customer has also licensed the Internet Banking product, the Internet Banking Web applications will also be built at this point, since the files necessary to build the Web applications will be present.

The compiled code is stored in D:\siebel\Common\lib\branch-impl.jar.

The compiled EJBs are stored in D:\siebel\Build\staging\ejb.

The Web applications (only if you have licensed Internet Banking) are stored in D:\siebel\Build\staging\war.

Building BranchAdministratorClient

Double-click on the JavaPrompt.vbs file in \siebel\Build to open a command prompt.

Enter the following at the command prompt:

```
D:\siebel>cd BranchAdministratorClient

D:\siebel\BranchAdministratorClient>ant
```

This will start the ANT build process, which creates a branchadmin.war war file.
This web application is stored in D:\siebel\Build\staging\war.

Building BranchClient

Double-click on the JavaPrompt.vbs file in \siebel\Build to open a command prompt.

Enter the following at the command prompt:

```
D:\siebel>cd BranchClient

D:\siebel\BranchClient>ant
```

This will start the ANT build process, which creates a branchteller.war war file.
This web application is stored in D:\siebel\Build\staging\war.

Building the EAR

Double-click on the `JavaPrompt.vbs` file in `\siebel\Build` to open a command prompt.

Enter the following at the command prompt:

```
D:\siebel>cd Build

D:\siebel\Build>ant
```

This builds an EAR containing all the server-side code and front-end applications (including the Internet Banking war, if you have licensed Internet Banking).

This EAR is located in `\siebel\Build\release\siebel.ear`.

Deploying the EAR

Now that you have rebuilt the EAR, you can redeploy on your application server. Please refer to the *Siebel Retail Finance Banking Application Installation Guide* for instructions.

5 WebSphere 5.1 Build Process

This section contains information about how to import and build the Siebel Retail Finance Banking Application for WebSphere 5.1 using the automated mechanism run by the file EJBDeploy.bat that is located in WebSphere 5.1 at {WebSphereHome}\bin\ejbdeploy.bat.

Prerequisites for the WebSphere Build Process

NOTE: This guide is provided to give an example build process; it is not an exhaustive guide to building the Siebel Retail Finance Banking application for WebSphere. In order to use this guide you must have WebSphere 5.1 on your build machine

In order to build an EAR for WebSphere, the websphere.home variable in \siebel\Common\build.properties must be uncommented (remove the # symbol) and pointed at the user's WebSphere home. The weblogic.home variable must be commented out. The variable relating to the target database must also be uncommented. Below is an example configuration for building a WebSphere—DB2 ear:

```
#weblogic.home=/bea/weblogic81/server
websphere.home=C:/Progra~1/WebSphere/AppServer
target.database.db2=true
#target.database.oracle=true
```

The following settings also must be set to non-empty textual values in the properties file in order to sign the jars and war files (signing is needed for the front-end war files that are deployed using WebStart):

```
alias=your keystore alias
storepass=your keystore password

companyName=your section
operatingUnit=your unit
organisation=your company
country=your country.
```

Entering valid values (for instance, some values are required to be at least six characters in length) for these settings will sign the jars/war files with the credentials supplied.

Automated WebSphere Build Process

The \siebel\Build folder contains a Windows script called JavaPrompt.vbs that creates a command prompt that has the classpath and path variables configured to invoke the Retail Finance Banking Application Build Process. There is a build script within this folder called build-all.xml. Running this script will build all the projects in the correct order and create the EAR file, which can be found in \siebel\Build\release.

Double-click on the JavaPrompt.vbs file in \siebel\Build to open a command prompt.

Enter the following at the command prompt:

```
D:\siebel>cd Build

D:\siebel\Build>ant –f build-all.xml
```

This will build the projects in the correct order and finally create the EAR. This EAR is located in \siebel\Build\release\siebel.ear.

NOTE: The MCA project is delivered precompiled and will not be rebuilt during the build. The precompiled EJB files and BankFrameMCA.war are found in the subfolders at siebel\Build\resource\mca. This project does not get rebuilt during the build process.

NOTE: A large number of reflection errors get thrown during the WebSphere build process; these errors should be ignored. A full build can run for a considerable length of time—typically 30-40 minutes on a 2GHz or faster PC with 512MB RAM.