Developing and Deploying Siebel Business Applications

Version 8.0

December 2006

ORACLE

Contents

Chapter 6: Getting Started in the User Interface Layer

Chapter 7: Configuring the House and Opportunity Entities

Chapter 10: Modifying the Look and Feel of the Web Client

Chapter 11: Testing the NREC Deployment

Chapter 12: Migrating to the Test Environment

Chapter 13: Using EIM to Load Data Into the Test Environment

Chapter 14: Required Application Administration Tasks

Chapter 15: Assignment Manager

Chapter 16: Siebel Business Process Designer

Chapter 17: Personalization

Chapter 18: Implementing Siebel Remote

Chapter 19: Deploying the Application

Index

1 What's New in This Release

What's New in Developing and Deploying Siebel Business Applications, Version 8.0

This guide has been updated to reflect product name changes. It was previously published as *Developing and Deploying Siebel eBusiness Applications*, Version 7.7.

2 Introducing the Business Case

This chapter introduces you to a fictitious company that is used as an example throughout this book. It summarizes a simple business case giving you a sense of the objectives, requirements, and the details of the company's solution design.

This chapter includes the following topics:

The NREC example is a simple example used to highlight particular tasks. It is not intended to present a solution design or suggest a comprehensive implementation methodology. Rather, it highlights particular aspects of NREC's solution design to present tasks in a real-world context.

Goals and Scope of Developing and Deploying Siebel Business Applications

This book presents an actual implementation of Oracle's Siebel Business Applications at a fictitious company. The goal is to learn by doing. If you follow the procedures in this book you will have a functioning implementation (without the data) that has some very typical customizations that are easy to complete. These customizations will give you a taste of what is possible; it does not delve into every conceivable customization you may wish to make. However, each customization topic provides cross-references to more information for those who wish to expand their knowledge.

This book also presents the basic strategy to take to plan, design, and implement Siebel applications for your specific requirements. By following how this strategy is applied to NREC, you can learn how to apply it to your company. Again, cross-references are supplied for each element of the strategy so you can get more detailed information where necessary.

Introducing NREC

The fictitious company used as an example in this book is National Real Estate Clearinghouse (NREC). NREC is a U.S.-based company that participates in the residential real estate industry. It acts as an agent for people who are selling houses. NREC maintains a database of houses for sale and makes this information available to a nationwide network of partner real estate agencies. These real estate agencies represent buyers. Real estate agents use NREC to find houses that match buyers needs. Through prenegotiated contracts with partner real estate agencies, NREC is able to close sales with fewer incurred costs than traditional real estate agencies.

NREC also solicits interest directly from buyers. Potential buyers call into a toll-free number to inquire about houses. NREC records the buyer's information and, rather than handling the leads itself, NREC passes them onto one of the agencies in its distribution network. For an illustration of these interactions, see Figure 1.

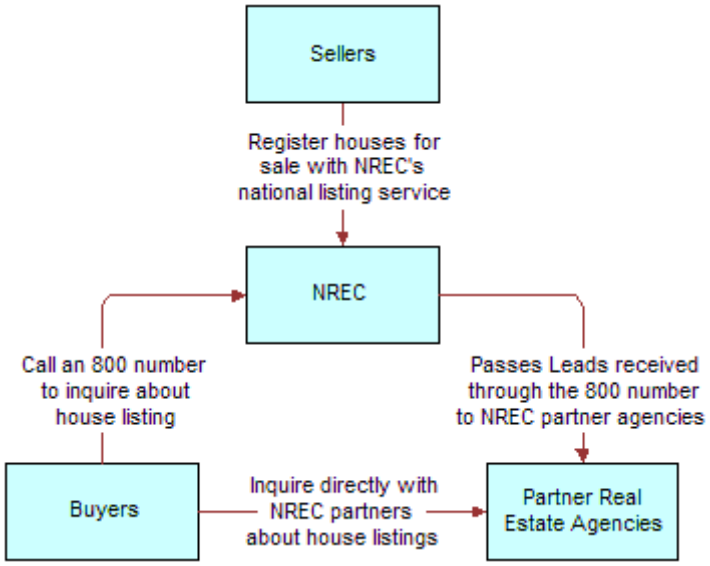


Figure 1. Key Business Processes

NREC's Business Objectives

Today NREC uses a homegrown solution to manage its pool of houses and buyer leads. It regularly faxes partner real estate agencies with information about available houses and dispatches leads in an ad-hoc manner. However the company realizes that this is an inadequate approach, which is costing it money and making it difficult to attract new independent real estate agencies to join its distribution network.

NREC has decided to address these problems by implementing a Siebel Business solution. This solution is both employee and partner facing, allowing participants access to the key information they need to do their job. In addition, business rules within the Siebel application streamline inefficiencies in the current process. For example, buyer leads can be automatically dispatched to partners based on agreed-upon assignment rules. In addition, agents will automatically be notified by email when an opportunity has been assigned to them.

The employee applications in NREC's implementation are based on Siebel Sales and Siebel Partner Manager, while the partner application are based on Siebel Partner Portal. These modules share a common back end, making sure that everyone always sees the most up-to-date information.

This book assumes that the configuration for the employee applications is proceeding successfully and focuses on the configuration of the partner application, Siebel Partner Portal. For all examples that show the user interface, use Siebel Partner Portal.

NREC's User Profiles

- **NREC Application Administrators.** NREC operates the Partner Portal Web site for its partner real estate agencies and their agents. It is responsible for adding houses as they become available and adding leads that are then automatically sent to a partner agency. They also perform administration tasks such as managing partner information.
- **Real Estate Agent.** Real estate agents of the partner agencies log into the NREC Partner Portal site to manage opportunities and contact information, browse houses for sale, and track activities related to each opportunity.
- **Real Estate Agent Manager.** Managers at partner real estate agencies log into the NREC Partner Portal site to track opportunities and activities for every agent in the agency. They also log in to see houses for sale, and to track their own opportunities, contact information, and activities.

Siebel Business Functionality

NREC is planning to implement the following functionality to support its business needs.

- Account Management
- Activity Management
- Assignment Manager
- Business Process Designer
- Contact Management
- Opportunity Management
- Partner Management
- Personalization
- Product Information

Access Requirements

In analyzing its requirements, NREC determined that it had the following data access needs.

- **Access through low-bandwidth Internet connections.** NREC's partner agents must be able to access data over low-bandwidth connections (such as dial-up connections) without having to install Siebel software on their personal computers.
- **Access through corporate intranet.** NREC Administrators and employees need to be able to administer the implementation (creating new users, assigning responsibilities, and so on) and maintain user data (houses, opportunities, contacts, and so on) through a graphical user interface connected to the corporate intranet.
- **Mobile access.** NREC employees need to be able to access and manipulate data on their laptops even without being connected to a corporate database. They must be able to synchronize with the central corporate database, when needed.

NREC plans to implement Siebel's Web client and Mobile Web client to meet the three requirements.

- The Web client—through which users connect to the Siebel server with their browser—meets the first and second requirements. The Siebel Web client is an HTML, Web interface providing access to Siebel applications through either the Internet or an intranet.
- The Mobile Web client meets the third requirement because agents can log into Siebel Business Applications and work offline in a local database that resides on the agent's computer. Users synchronize the local database with the server database when necessary. The mobile Web client provides the same HTML interface as the Web client, but the Web pages are served from the local Siebel application rather than a server.

For more information about deploying the Mobile Web client, read *Siebel Installation Guide for Microsoft Windows*.

NREC's Implementation Strategy

NREC has chosen to implement Siebel business applications using a phased approach. A phased approach helps NREC minimize risk and realize a return on its investment incrementally, thereby increasing the probability of project success. NREC's users and partners will experience the benefits of the project as early as possible. Additionally, the project team can take feedback gathered during the early phases and incorporate it into the end solution.

Siebel eRoadmap Methodology

NREC follows Siebel's eRoadmap Methodology through each one of its implementation phases. The Siebel eRoadmap Methodology provides a six-stage framework for implementing Siebel Business Applications. The stages include the following:

- **Discover.** The project team refines and documents functional and technical requirements that support the business goals.
- **Define.** The project team assembles, defines the project approach and scope, and implements project management controls.
- **Create Prototype.** The project team designs a hard-copy mock-up of the solution and uses the discovery stage requirements to develop application screen flows and design layouts.

- **Configure.** The project team configures the application, extensions, and external interfaces required to support the new implementation.
- **Validate.** The project team conducts a full-function test, including a user acceptance test of the application using production data.
- **Deploy.** The project team first conducts a Production Pilot that field-tests and revises every aspect of the new implementation, user training, technical infrastructure, the network, and the help desk. The team then focuses on a successful transition from the production pilot to a full rollout.
- **Sustain.** Evaluate to identify opportunities for improvement. The testing strategy and its objectives must be reviewed to identify any inadequacies in planning. Test cases must be updated to include testing scenarios that were discovered during testing and were not previously identified.

The tasks covered in this book are primarily in the configuration and deployment stages of the eRoadmap Methodology. The scenario for the book assumes that a design team has planned out the details and has provided NREC developers with design specifications used in configuration.

Introduction to Application Software Testing

Testing is a key component of any application deployment project. The testing process determines the readiness of the application. Therefore, it must be designed to adequately inform deployment decisions. Without well-planned testing, project teams may be forced to make under-informed decisions and expose the business to undue risk. Conversely, well-planned and executed testing can deliver significant benefit to a project, including:

- **Reduced Deployment Cost.** Identifying defects early in the project is a critical factor in reducing the total cost of ownership. Research shows that the cost of resolving a defect increases dramatically in later deployment phases. A defect discovered in the requirements definition phase as a requirement gap can be a hundred times less expensive to address than if it is discovered after the application has been deployed. When in production, a serious defect can result in lost business and affect the success of the project.
- **Higher User Acceptance.** User perception of quality is extremely important to the success of a deployment. Functional testing, usability testing, and performance testing can provide insights into deficiencies from the users' perspective early enough so that these deficiencies can be corrected before releasing the application to the larger user community.
- **Improved Deployment Quality.** Hardware and software components of the project must also meet a high level of quality. The ability of the application to perform reliably is critical in delivering consistent service to the users or customers. An outage caused by inadequate resources can result in lost business. Performance, reliability, and stress testing can provide an early assessment of the project to handle the production load and allow IT organizations to plan accordingly.

Inserting testing early and often is a key component to lowering the total cost of ownership. Software projects that attempt to save time and money by lowering their initial investment in testing find that the cost of *not* testing is much greater. Insufficient investment in testing may result in higher deployment costs, lower user adoption, and failure to achieve business returns.

Common Test Definitions

There are several common terms used to describe specific aspects of software testing. These testing classifications are used to break down the problem of testing into manageable pieces. Here are some of the common terms that are used throughout this book.

- **Integration Testing.** Validates that all programs and interfaces external to the Siebel application function correctly. Sometimes adding a new module, application, or interface may negatively affect the functionality of another module.
- **Performance Testing.** This test is usually performed using an automation tool to simulate user load while measuring resources used. Client and server response times are both measured. This must be conducted in an environment that has realistic test data.
- **Regression Testing.** Code additions or changes may unintentionally introduce unexpected errors or regressions that did not exist previously. Regression tests are executed when a new build or release is available to make sure existing and new features function correctly.
- **Stress Testing.** This test identifies the maximum load a given hardware configuration can handle. Test scenarios usually simulate expected peak loads.
- **Unit Testing.** Developers test their code against predefined design specifications. A unit test is an isolated test that is often the first feature test that developers perform in their own environment before checking changes into the configuration repository. Unit testing prevents introducing unstable components (or units) into the larger application.
- **Usability Testing.** User interaction with the graphical user interface (GUI) is tested to observe the effectiveness of the GUI when test users attempt to complete common tasks.
- **User Acceptance Test.** Users test the complete, end-to-end business processes. Functional and performance requirements are verified to make sure there are no user task failures and no prohibitive response times.

Siebel eRoadmap Implementation Methodology

The Siebel eRoadmap implementation methodology accelerates project implementations by focusing on the key strategic and tactical areas that must be addressed to maximize the customer's return on investment, while minimizing their business risk to promote a successful completion of a Siebel project. The Siebel implementation is composed of activities logically grouped into eight distinct eRoadmap stages to make sure proper project management and control techniques are used during the life cycle of a project. These stages (illustrated in Figure 2) are iterative in nature.

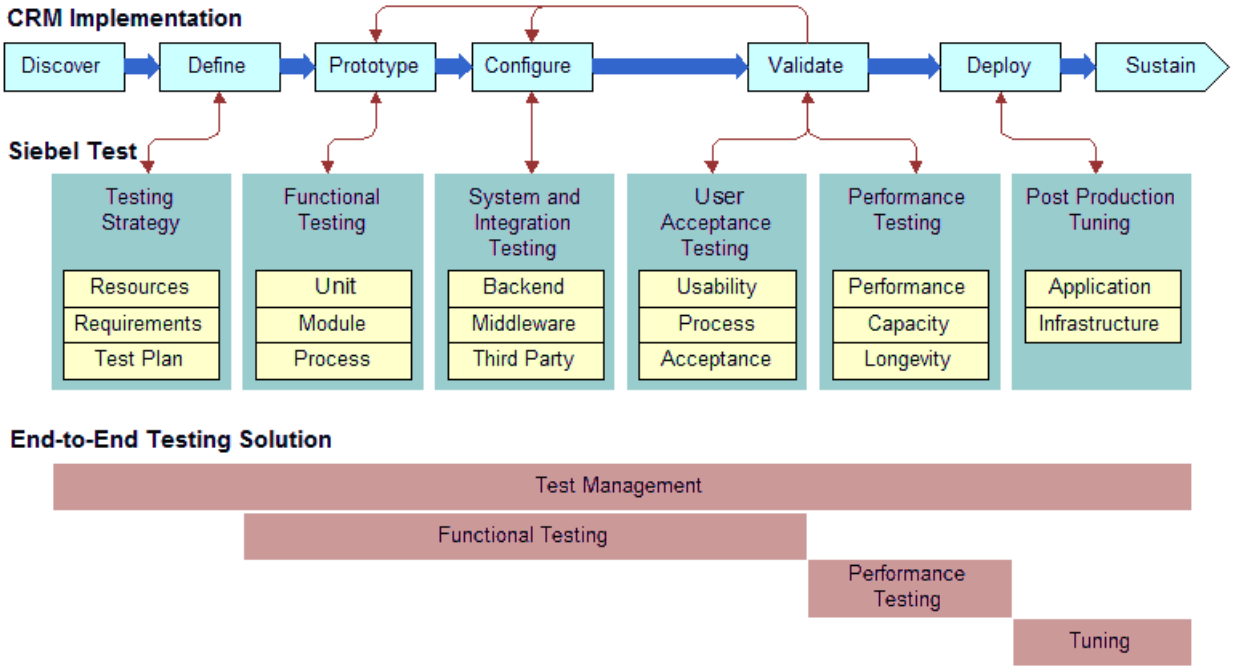


Figure 2. Siebel eRoadmap Implementation Methodology

Testing is an end-to-end process that begins when you start to configure the Siebel application. The first stage is the development of a testing strategy by the testing team to define environmental simulation requirements and testing approaches, establish priorities, and define and create proper functional and load test scripts. The output of this stage is a comprehensive Test Plan.

As mentioned earlier, functional testing begins when prototyping begins and continues throughout the configuration of the Siebel application as developers test each unit they configure. Tested units are then moved into the testing environment, where the appropriate units are combined to form a corresponding module. The test team then verifies whether or not the module functions correctly (for example, returns the correct value), has the correct layout (for example, drop-down menus and text fields), and has the correct interface. After validating a module, functional testing continues using business processes and scenarios to verify whether or not all modules work together as required.

The next stage of functional testing, System and Integration testing, is validation that the Siebel application operates with other applications and interfaces. Test the Siebel application in a test environment that allows the Siebel application to interoperate with the other required applications (such as CTI, load balancer, and middleware).

User Acceptance testing (UAT) consists of testing the Siebel application using the appropriate business owners and end users. When performing UAT, make sure that you have users who are familiar with the existing business processes.

Performance testing provides an assessment of whether or not an infrastructure performs and scales to your requirements. This phase requires an image of the full database and all interfaces with the Siebel application (such as CTI, middleware, and email). This must be conducted in an environment that has realistic test data. The first step is to establish a benchmark of performance through the completion of a performance test. Next, complete a capacity test by adding users until you reach the number of users expected to use the application over its life. Finally, execute the test over an extended period of time (longevity test) to determine durability of an application as well as capture any defects that become visible over time.

Overview of the Siebel Testing Process

Testing processes occur throughout the implementation lifecycle, and are closely linked to other configuration, deployment, and operations processes. Each of the seven testing processes described in this book are highlighted in bold in the diagram and are outlined briefly in the following topics:

- Plan Testing Strategy on page 20
- Develop Tests on page 21
- Execute Siebel Functional Tests on page 21
- Execute Integration Tests on page 21
- Execute User Acceptance Tests on page 21
- Execute Performance Tests on page 22
- Improve Testing on page 22

Plan Testing Strategy

The test planning process makes sure that the testing performed is able to inform the deployment decision process, minimize risk, and provide a structure for tracking progress. Without proper planning many customers may perform either too much or too little testing. The process is designed to identify key project objectives and develop plans based on those objectives.

Develop a testing strategy early and use effective communications to coordinate among all stakeholders of the project.

Develop Tests

In the test development process, the test cases identified during the planning process are developed. Developers and testers finalize the test cases based on approved technical designs. The written test cases can also serve as blueprints for developing automated test scripts. Test cases must be developed with strong participation from the business analyst to understand the details of usage and corner use cases.

Design evaluation is the first form of testing, and often the most effective. Unfortunately, this process is often neglected. In this process, business analysts and developers verify that the design meets the business unit requirements. Do not start development work in earnest until there is agreement that the designed solution meets requirements. The business analyst who defines the requirements must approve the design.

Preventing design defects or omissions at this stage is more cost effective than addressing them later in the project. If a design is flawed from the beginning, the cost to redesign after implementation can be high.

Execute Siebel Functional Tests

Functional testing is focused on validating the Siebel Business application components. Functional tests are performed progressively on components (units), modules, and business processes to verify that the Siebel application functions correctly. Test execution and defect resolution are the focus of this process. The development team is fully engaged in implementing features, and the defect-tracking process is used to manage quality.

Execute Integration Tests

Integration testing verifies that the Siebel application, validated earlier, integrates with other applications and infrastructure in your environment. Integration with various backend, middleware, and third-party applications must be verified. Integration testing occurs on the system as a whole to make sure that the Siebel application functions properly when connected to related applications and other infrastructure components.

Execute User Acceptance Tests

You perform user acceptance testing on the complete system and focus on validating support for business processes, as well as verifying acceptability to the user community from both the lines of business and the IT organization. This is typically a very busy time in the project, when people, process, and technology are all preparing for the rollout.

Execute Performance Tests

Performance testing validates that the application can meet specified service levels for performance, capacity, and reliability. In this process, tests are run on the complete system simulating expected loads and verifying performance. This must be conducted in an environment that has realistic test data.

Improve Testing

Testing is not complete when the application is rolled out. After the initial deployment, regular configuration changes are delivered in new releases. In addition, Oracle delivers regular maintenance and major software releases that may need to be applied. Both configuration changes and new software releases must be tested to verify that quality is sustained.

The testing process must be evaluated after deployment to identify opportunities for improvement. The testing strategy and its objectives must be reviewed to identify any inadequacies in planning. Test plans and test cases must be reviewed to determine their effectiveness. Test cases must be updated to include testing scenarios that were discovered during testing and were not previously identified.

Project Team

It is a common practice to have a dynamic project team wherein the number and specialty of people varies according to the stage of the implementation. For example, Table 1 shows the breakdown of resources needed at each stage in NREC's implementation project.

Table 1. NREC's Project Resources

Project Resources	Development Phases					
	Definition	Discovery	Design	Configuration	Validation	Deployment
Project Manager	X	X	X	X	X	X
Lead Business Analyst	X	X	X	X	X	X
Lead Configuration Specialist		X	X	X	X	X
Lead Architect			X	X	X	X
Configuration Specialist				X	X	X
Architect					X	X

Table 1. NREC's Project Resources

Project Resources	Development Phases					
	Definition	Discovery	Design	Configuration	Validation	Deployment
Training Specialist					X	X
Documentation Specialist					X	X

Sample Project Design Documentation

The types of documents listed in this section capture the business requirements and detailed design of NREC's Siebel solution. These documents are outputs of the design stage of the Siebel eRoadmap implementation methodology. They define the scope of the solution and document the detailed specifications needed for configuration.

Design documentation is a critical part of an implementation. Customers can submit these documents to Oracle's Siebel Expert Services for a design review. If documents are not submitted or the templates are not used to guide them through the documentation process, the customer may omit important information and reduce the value of the review.

The types of documents used to capture the design are:

- **Entity Relationship Diagram.** This type of diagram (see Figure 3 on page 28) shows the relationships between the major entities that are part of NREC's solution. Entities include object definitions such as business components and business objects.
- **Business Component Design.** The business component design template is used to define the properties of business components, including the user properties, joins, single-value fields, multi-value fields, and multi-value links.
- **Business Object Design.** The business object design template is used after business components are designed. The template groups business components and identifies links between business components. This indicates the parent and associated child records in the user interface.
- **Applet Design.** Applet designs are created after the implementation team decides what entities to track in their Siebel applications. Applet designs are reference documents created for each applet. They comprehensively document an applet's properties, the fields that are available on the applet, and the properties associated with each field. They must also include a mock-up or screen shot of the applet.
- **View Design.** View design documentation must show what applets are part of the view definition, identify any special visibility associated with a view, for example, a *My Team's* View or an *All* View. It must also document how the user navigates through the views using drilldowns, view tabs (third-level navigation), or the Site Map.
- **Screen Flow.** The screen design template lists the views available from each screen and indicates the default view for each screen.

- **Report Design.** Report templates track key information relating to reports. The report template includes the business component, Actuate file used, whether the report is dynamic or static, and any subreports available.
- **Workflow Policies.** Workflow Policy templates gather the key information needed to create workflow rules.
- **Assignment Rules.** Assignment Manager templates gather the key information needed to create assignment rules.
- **Testing Strategy Document.** The Testing Strategy Document identifies key project objectives and provides a structure for tracking progress. Without proper planning you may perform either too much or too little testing.
- **Test Cases Document.** Includes test cases based on approved technical designs. The written test cases can also serve as blueprints for developing automated test scripts. Test cases must be developed with strong participation from the business analyst to understand the details of usage and corner use cases.

NREC's Business Requirements

By progressing through the Discovery and Design stages of Siebel's eRoadmap methodology, the project team identified and documented the detailed requirements. These requirements provide the basis for the tasks covered in later chapters.

Some of these requirements are satisfied using standard Siebel functionality. Other requirements cannot be satisfied with standard functionality and are identified as gaps. Gaps require configuration work to modify the standard application to meet the specific requirements.

NOTE: As a general rule, a gap analysis results in less than 25% of the functionality requiring customization. If the gap is more than 25%, it may mean that standard functionality is not being employed. It is always a good practice to engage Oracle's Siebel Expert Services for a Configuration Design Review to verify gaps and required configuration changes.

Manage the List of Houses for Sale

NREC requires the ability to manage the list of houses that are currently for sale. This includes storing key characteristics, such as price, number of bedrooms, and square feet; keeping a history of appraisals; and pulling in any data about renovation history from an external file that is supplied by a third-party vendor.

Table 2 lists the detailed requirements for the House entity and lists where you can find the tasks in this book that describe how to implement the requirement.

Table 2. Requirements Summary for the House Entity

Requirement	Comment	Cross Reference to Task
Track houses for sale.	Create a new screen for Houses and add to the Siebel eChannel application.	Read "Creating the Houses Screen" on page 97.
Track the features of each house, such as square feet, number of bedrooms, and so on.	Add columns to S_PROD_IN T to store additional attributes and expose them in the user interface.	Read "Configuring the House Detail View" on page 87.
Entering and editing records for houses.	Modify Internal Product business component to allow updates and add necessary controls to the Product Form Applet.	Read "Configuring the Internal Product Business Component" on page 92 and "Modify Existing Product Applets to Display NREC Attributes" on page 93.
Provide users with predefined values to choose from when entering house features.	Configure static pick lists for fields.	Read "Configuring Pick Lists" on page 119.
Track and display previous appraisal information (date, amount, assessor) for each house.	Use standard 1:M table to store appraisal data and expose fields in the user interface.	Read "Configuring the House Detail - Appraisals View" on page 101.
Display renovation information for each house.	Create virtual business component to store and display data from flat file. Create new view to display the data.	Read "Creating a Virtual Business Component" on page 137.
Display houses to partner users based on ZIP Code.	Use Siebel Personalization to filter the list of houses based on the user's ZIP Code.	Read Chapter 17, "Personalization."

Manage Opportunities

NREC needs to give its internal employees as well as its partner agents ability to manage opportunities. Opportunities are recorded for potential buyers and include contact information and information about the type of house the buyer is interested in. Opportunities that come into NREC directly are passed on to a partner agency based on location. Table 3 summarizes the requirements for the opportunity entity.

Table 3. Requirements Summary for the Opportunities Entity

Requirement	Comment	Cross Reference to Task
For each opportunity, track the house features that the buyer is looking for and buyer characteristics.	Add columns to the base opportunity table S_OPTY to store the additional attributes.	Read "Configuring the House Detail View" on page 87.
Users may choose values for house attributes—square feet, price range, and so on—from a list rather than enter them.	Add pick lists for each of the fields that need pre-defined values.	Read "Static Pick Lists" on page 120.
Assign opportunities to partner agents based on ZIP Code.	Use Assignment Manager to assign opportunities to partner agencies.	Read Chapter 15, "Assignment Manager."
Send an email notification to partner agents when a new opportunity is assigned to them.	Use Business Process Designer to automatically send email notification.	Read Chapter 16, "Siebel Business Process Designer."

Manage Activities

NREC requires the ability to manage activities for each opportunity and contact. The standard Activity management functionality meets NREC requirements well, except for the gaps listed in Table 4.

Table 4. Requirements Summary for the Activities Entity

Requirement	Comment	Cross Reference to Task
Display the opportunity associated with each activity.	Add the opportunity to the Activity List and Form Applets.	"Exposing Fields in the User Interface" on page 82
When entering an activity, allow users to associate an opportunity to an activity.	Create a dynamic pick list that allows users to update the Opportunity field on the Account Form Applet. Constrain the values in the pick applet so it only shows the opportunities for the account associated with the activity.	"Dynamic Pick Lists" on page 122 "Constraining a Pick List" on page 124
Allow users to navigate from the Opportunity Detail view to one of the following views, depending on the sales stage: Account Detail - Activities (SCW) Account Detail - Contacts (SCW)	Configure dynamic drilldown that implements the conditions described in the business requirement.	"Creating a Dynamic Drilldown" on page 127

Manage Contacts

The standard Contact management functionality meets NREC requirements. There are no gaps identified.

Reviewing NREC's Design

The following sections describe NREC's solution design. An entity relationship diagram, such as the one shown in Figure 3, is particularly useful for describing the design. You may refer back to Figure 3 often when completing the tasks in subsequent chapters.

Data Layer

NREC will implement the entities shown in Figure 3. The diagram shows the relationships (1:1, 1:M, or M:M) between the major entities of NREC's solution. The diagram also shows the tables used to store the data for each entity. For example, house information will be stored in the table called S_PROD_INT. It also shows external data being handled by the virtual business component named House Renovations.

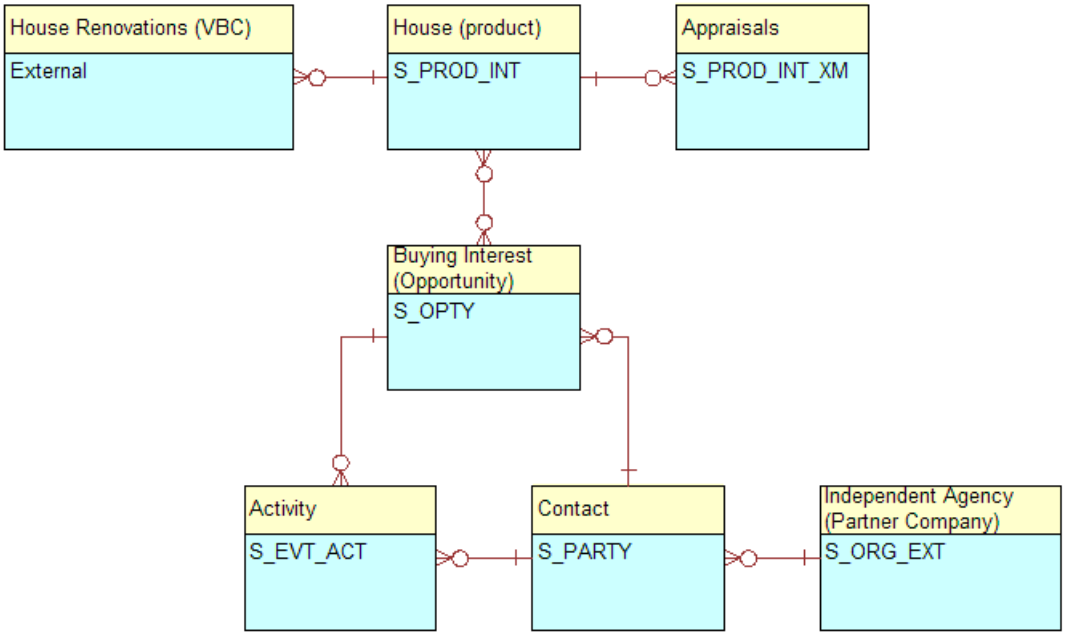


Figure 3. NREC Entity Relationship Diagram

Business Object Layer

This section defines the business objects and business components that represent the entities that make up NREC's solution.

Business Objects

The NREC implementation plans to use the following business objects.

- Accounts
- Activities
- Contacts
- Internal Products
- Opportunities

Business Components

The NREC implementation plans to use the following business components.

- Activity
- Appraisals (new business component)
- Contacts
- Internal Product
- Opportunity
- Partner Company
- Renovations VBC (new virtual business component)

User Interface Layer

At the user interface layer, NREC is planning to implement five screens for the Partner Portal. Each view contains one or more applets. The Opportunities, Contacts, and Activities screens are standard in the Partner Portal application. A new screen for Houses will be created for NREC. The views that make up each screen are either master-detail views or list-form views. The applets are either list applets or form applets.

NOTE: Figure 4 is a simple representation of part of NREC's user interface layer. There are other screens, such as Partner Management and Application Administration, that will be implemented for NREC.

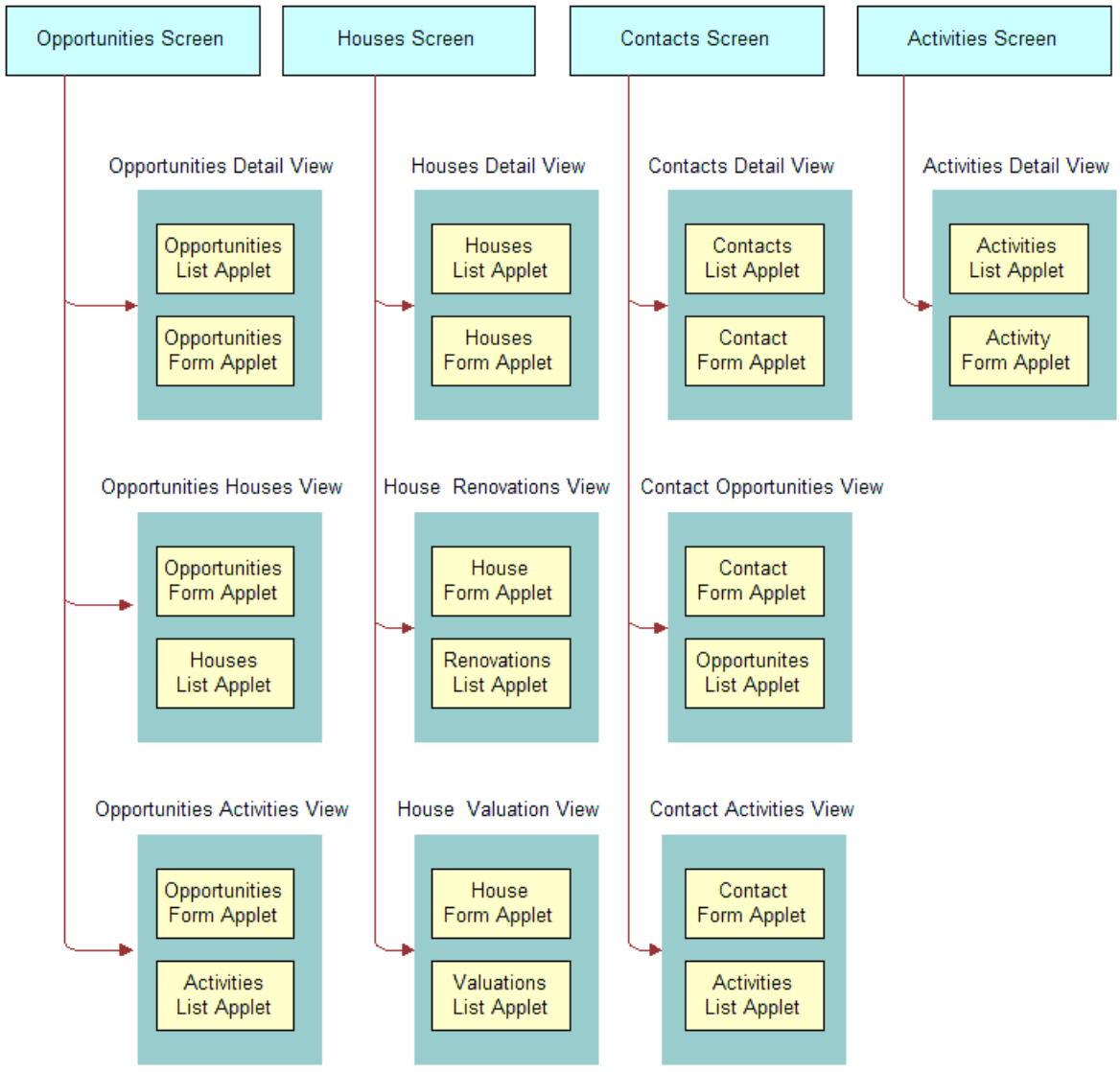


Figure 4. Screens, Views, Applets

3 Installing Siebel Applications

This chapter provides you with an overview of a typical installation, using NREC as an example. It tells you the different environments to set up to support the implementation process and then describes the hardware and software NREC installed in each environment. The example gives details, such as the machine names on which products were installed.

This chapter includes the following topics:

This chapter does not provide you with installation instructions. For each product mentioned in the following sections, there is a cross reference to where you can find detailed instructions.

Siebel Application Environments

Following the recommendation of Oracle, NREC chose to install Siebel products in three separate environments—one environment for each stage in the implementation process. The three environments are development, test, and production. Each environment contains hardware, software, and data. Using this structure prevents the activities in one environment from interfering with activities in another environment. For example, configuration work in the development environment does not interfere with testing activities in the test environment.

- **Development.** The environment is used for developing customized applications and configurations. It typically consists of a server machine, several client machines, development tools, and a small set of data (repository data, seed data, and sample transactional data). The server machine in the development environment is often called the *development server*.
- **Test.** The test environment is a separate machine (or several machines) with business data but no development tools. This environment is used to test the application with data that simulates the live production environment. A test environment is used to test customizations, patches, and version upgrades, before applying them to the production environment.
- **Production.** The production environment is the live Siebel operational environment. A production environment might be very similar to the test environment at the end of testing, but ultimately it consists of hundreds or even thousands of users and live business data.

Figure 5 shows the typical content of each environment.

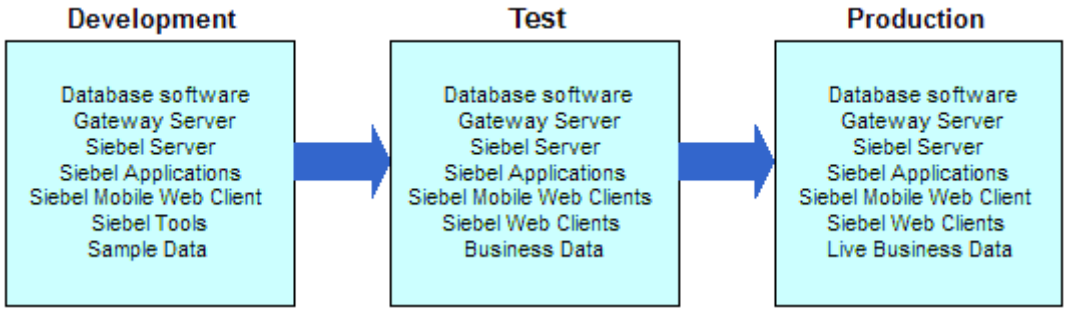


Figure 5. Software and Data in Each Environment

Each environment corresponds to a stage in the application rollout.

1. **Development.** Do your initial configuration in the development environment.
2. **Test.** Copy the configuration data from the development environment to the test environment and continue work there, using actual data for your organization. Here you configure business rules and set up the data to simulate the production environment. After the set up is complete, testing occurs.
3. **Production or Deployment.** After you have thoroughly tested your work in the test environment, you migrate the configuration and user data to the production environment and roll out the product to your users.

The subsequent parts of this guide correspond to these three environments and stages, each part telling you what NREC did at each stage.

The Development Environment

Table 5 lists and describes which products to install in the development environment, on which machine the NREC installed each product, and where you can find installation instructions.

NREC installed Siebel Tools on two machines, one for each developer who configure Siebel applications. You can have as many developers working simultaneously as you like, as long you set up each developer's machine according to the instructions in Chapter 5, "Setting Up a Developer's Local Database."

For detailed information on supported software, read *Siebel System Requirements and Supported Platforms* on Siebel SupportWeb.

NOTE: The NREC example assumes a Windows environment.

Table 5. Software Installed in the Development Environment

Product	Description	Machine Names	Where to Find Installation Instructions
Database software. For example, IBM DB2 Universal Database, Oracle, or SQL Server.	Third-party relational database software.	DEV_DB_server	Product documentation for the database application.
Siebel database schema	Installing and running a database script creates a database schema and populates some of the tables with seed data.	DEV_DB_server	*Siebel Installation Guide for Microsoft Windows*
Siebel File System	A shared directory tree used for storing files not managed by the database software.	DEV_server	*Siebel Installation Guide for Microsoft Windows*
Siebel Gateway Server	Software that handles connections between Siebel clients and servers.	DEV_server	*Siebel Installation Guide for Microsoft Windows*
Siebel Server	Application server software.	DEV_server	*Siebel Installation Guide for Microsoft Windows*
Siebel Dedicated Web Client	A client in which application layers except the user interface reside on the server.	DEV_server	*Siebel Installation Guide for Microsoft Windows*
Web server such as Microsoft Internet Information Server (IIS) or IBM HTTP Server.	Web server required for the Siebel Web architecture.	DEV_server	Product documentation for the Web server.
Siebel Tools	The primary software for configuring Siebel client applications.	DEV_tools_A DEV_tools_B	*Using Siebel Tools*

Table 5. Software Installed in the Development Environment

Product	Description	Machine Names	Where to Find Installation Instructions
Siebel Mobile Web Client	A client in which application layers reside on the user's personal computer.	DEV_tools_A DEV_tools_B	*Siebel Installation Guide for Microsoft Windows*
Sample Database	The sample database that you can use to test, evaluate, and configure.	DEV_tools_A DEV_tools_B	*Siebel Installation Guide for Microsoft Windows* and the release notes documentation for your Siebel application.
Microsoft Internet Explorer	The Web browser used to display the interface for the Siebel Web Client and the Mobile Web Client.	DEV_client_A DEV_tools_B	If this software is not pre-installed on the client computer, go to the Microsoft Web site for installation instructions.

The Test Environment

Table 6 lists which products to install in the test environment, on which machine NREC installed each product, and where you can find installation instructions.

For detailed information on supported software, read *Siebel System Requirements and Supported Platforms* on Siebel SupportWeb.

NOTE: The NREC example assumes a Windows environment.

Table 6. Software Installed in the Test Environment

Product Installed by NREC	Machine Name	Where to Find Installation Instructions
Database software. For example IBM DB2 Universal Database, Oracle, or SQL Server.	TEST_DB_server	Product documentation for the database platform.
Siebel database schema	TEST_DB_server	*Siebel Installation Guide for Microsoft Windows*
Siebel File System	TEST_server	*Siebel Installation Guide for Microsoft Windows*
Siebel Gateway Server	TEST_server	*Siebel Installation Guide for Microsoft Windows*

Table 6. Software Installed in the Test Environment

Product Installed by NREC	Machine Name	Where to Find Installation Instructions
Siebel Server	TEST_server	*Siebel Installation Guide for Microsoft Windows*
Siebel Mobile Web Client	TEST_client_A (and other client machines from which to test access)	*Siebel Installation Guide for Microsoft Windows*
Web server such as Microsoft Internet Information Server (IIS) or IBM HTTP Server.	TEST_server	Product documentation for the Web server.
Microsoft Internet Explorer	TEST_client_B (and other client machines from which to test access)	If this software is not preinstalled on the client computer, go to the Microsoft Web site for installation instructions.

The Production Environment

The software you install in your production environment is typically the same as the software installed in the test environment (shown in Table 6 on page 34). Eventually, during rollout to users, you copy the contents of one of the client machines to your mobile Web Client machines. Also, you distribute a URL to your application to your Web Client users.

NREC installed the Production software on machines with names starting with PROD. For example, it installed the server-side software on PROD_server.

4 Getting Started with Siebel Tools

This chapter describes key concepts and common tasks that get you started using Siebel Tools. You may refer back to these tasks as you work through the configuration tasks in subsequent chapters. For more information about Siebel Tools, read *Using Siebel Tools*.

This chapter includes the following topics:

Siebel Object Architecture

Siebel Business architecture includes a core set of object definitions that are grouped into different layers depending on the object's function and characteristics. Additionally, there is a core set of HTML templates and style sheets that control the appearance of the user interface (see Figure 6 on page 38). You can modify object definitions and templates, or create new ones, to tailor Siebel applications to meet your organization's business requirements.

Siebel Web templates occupy the top layer of the architecture. Siebel object definitions are grouped into the middle three layers. The physical RDBMS database occupies the bottom layer. You modify Web templates and style sheets using a text editor or a raw code HTML editor. You modify Siebel object definitions using Siebel Tools.

Objects depend on objects defined in the layers below, but are insulated from each other. Changes to objects in one layer require little or no changes to the layers below. For example, you can control how data is presented by modifying objects in the user interface layer, without having to modify objects in the business logic layer. Likewise, you can change the color and other style characteristics of the user interface by modifying Web templates and style sheets, without having to modify object definitions.

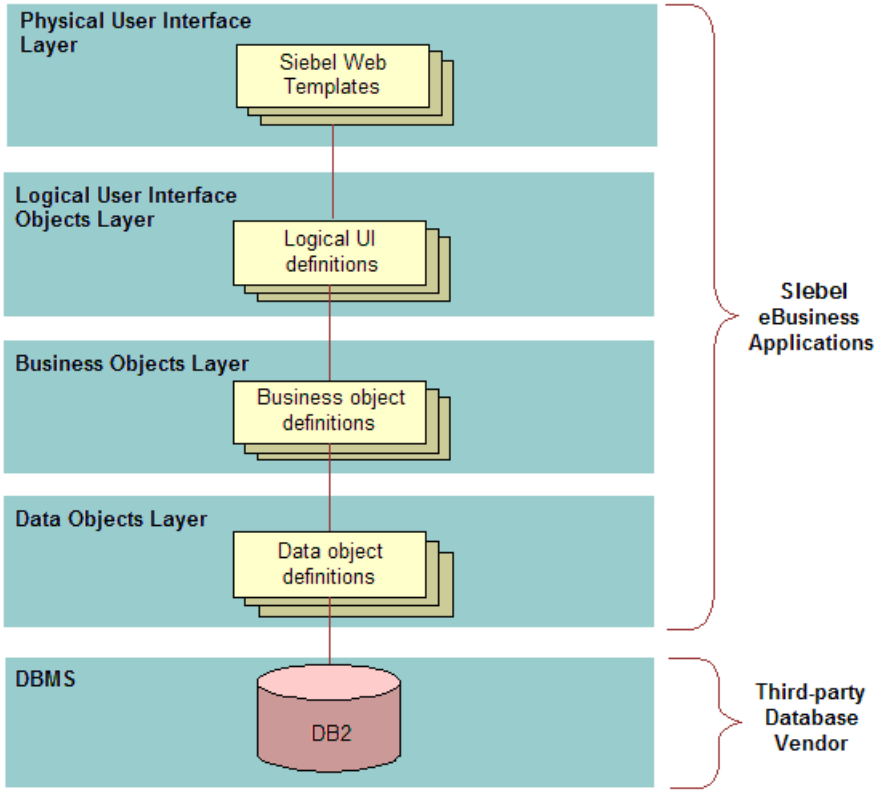


Figure 6. Siebel Object-Based, Layered Architecture

- **Physical User Interface Layer.** This layer contains Siebel Web template files that control the style and structure of the user interface. Web templates consist of HTML tags and proprietary Siebel tags. Siebel tags are embedded within the HTML of template files and serve as placeholders for user interface objects defined in the repository, such as controls and applets. At run time the Siebel Web Engine reads the tags, replaces them with interactive Web controls and values based on the UI object definitions, and renders the HTML that will be read by the user's browser.
- **Logical User Interface Layer.** Object definitions in this layer are the visual representation of objects in the Business Objects Layer. They define the interface presented to the user at run time, and allow users to manipulate data. Examples of user interface objects include applets, views, and controls, such as buttons and check boxes. User interface objects also define the information that associates objects in the repository with the Siebel Web templates.

- **Business Objects Layer.** Object definitions in this layer describe individual business entities (such as Accounts, Contacts, or Activities) and the logical groupings and relationships among these entities. Business objects are based on data object definitions.
- **Data Objects Layer.** Object definitions in this layer provide a logical representation of the underlying physical database. For example, object definitions such as table, column, and index describe the physical database. These object definitions are independent of the installed RDBMS.
- **DBMS.** The third-party database management system manages the Data Objects Layer. It is not a part of the Siebel Business Application.

Each layer of the Siebel object model contains several principal object types. Most of these object types contain child objects that further define the given object type.

For detailed information about the Siebel Object Architecture, read *Configuring Siebel Business Applications*.

Understanding the Object Definition Sequence

When configuring Siebel applications it is useful to think of configuration tasks in terms of the Siebel object model hierarchy shown in Figure 6 on page 38. Sometimes you work from the bottom up—data objects first, then business objects, and then user interface objects. Other times you work in one layer only, modifying objects as needed. Figure 7 shows the general sequence of tasks.

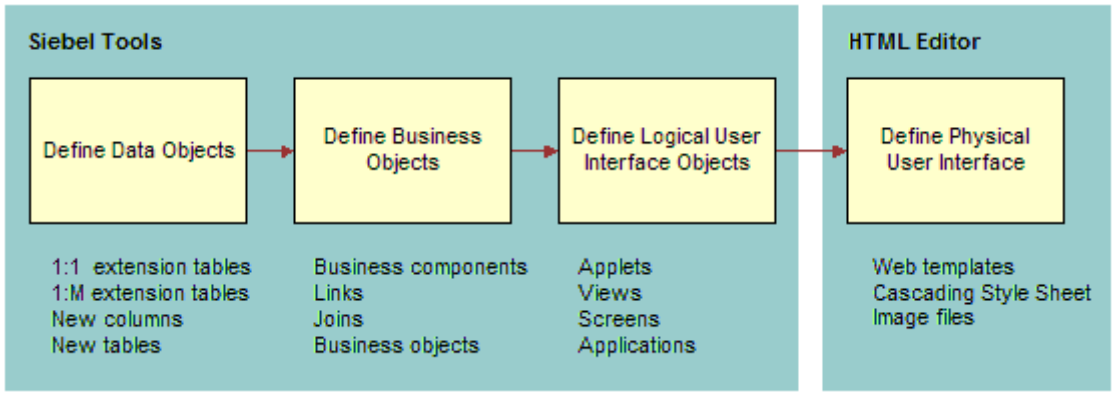


Figure 7. Development Sequence

Using Siebel Tools

Siebel Tools is a declarative software development environment used to create and modify object definitions in the Siebel repository. Using Siebel Tools you can configure a standard Siebel application without modifying source code or SQL code.

NOTE: You use an HTML code editor, not Siebel Tools, to modify Web templates. However, you can also view a color-coded rendition of Web templates using the Web Template Window in Siebel Tools. You can also specify a default HTML editor to open from within the Siebel Tools application. For more information, read "Setting Tools Options" on page 45.

Windows in Siebel Tools

Most of the work you do in Siebel Tools is through the Object Explorer Window, the Object List Editor, or the Object Properties Window. Each of these windows is described in the following sections.

Object Explorer

The Object Explorer is your starting point for working with object definitions. You use it to navigate through the Siebel object type hierarchy. It shows you the top level object types and lets you to expand them to reveal their child object types. You can also click the Flat tab to view object types listed alphabetically in a nonhierarchical structure. The Project drop-down list lets you display the object types for locked projects or a particular project only.

Not every top level object appears in the Object Explorer by default. You can display or hide object types by setting your Development Tools Options. Choose View > Options and then click the Object Explorer tab. Add or remove objects from the list of visible objects. Figure 8 shows the Object Explorer.

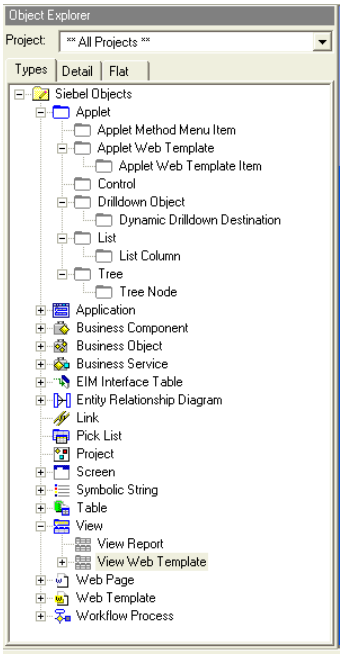


Figure 8. Object Explorer

Object List Editor

The Object List Editor appears to the right of the Object Explorer. It lists the object definitions for the object type selected in the Object Explorer. For example, in Figure 9 the Object List Editor displays the object definitions for the Applet object type. Use the Object List Editor to create, modify, or delete object definitions.

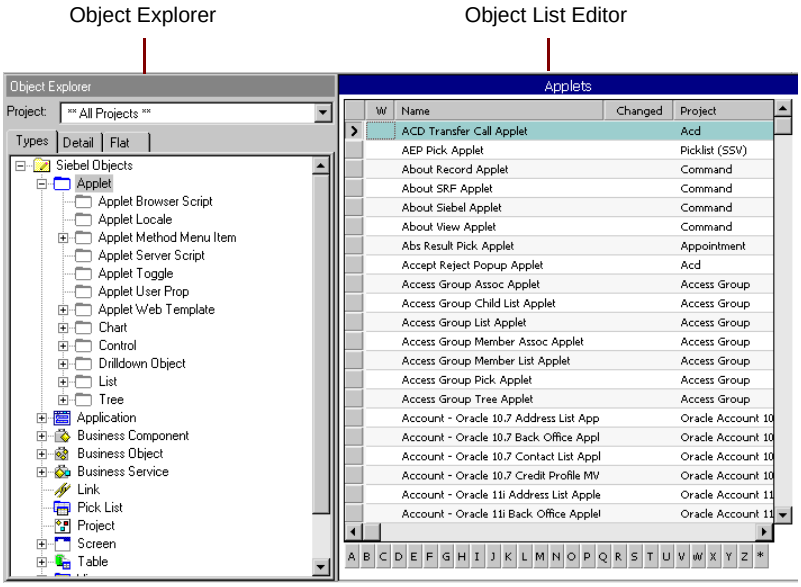


Figure 9. Object List Editor

Object Properties Window

The Properties Window displays the property values for the object definition selected in the Object List Editor. It provides you with property definitions at a glance. The property window is often more convenient than the Object List Editor when you view or edit objects that have many properties. You can show or hide the properties window by doing the following.

To show the Properties window

- Choose View > Windows > Properties window.

To hide the Properties window

- In the Properties window shown in Figure 10, right-click and then choose Hide.

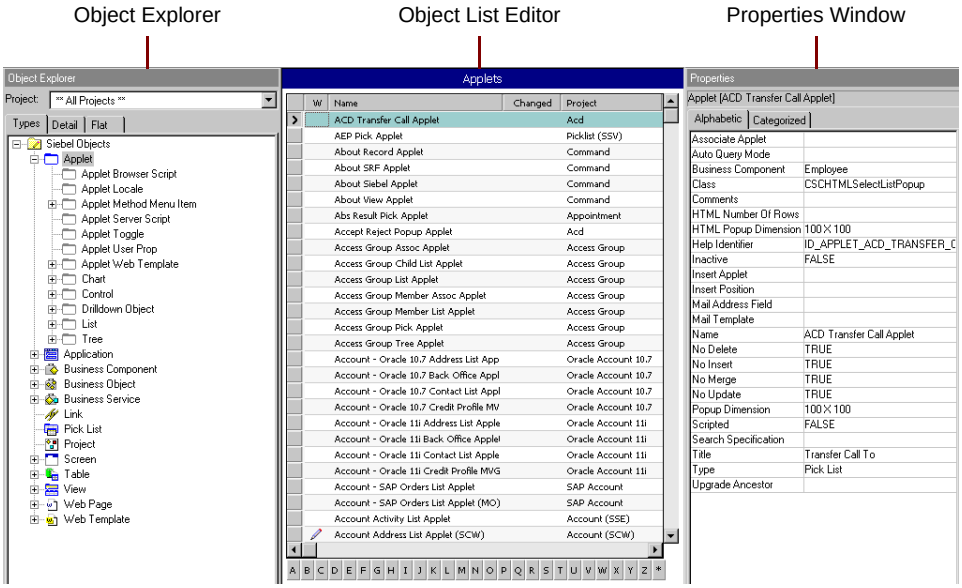


Figure 10. Properties Window

Wizards

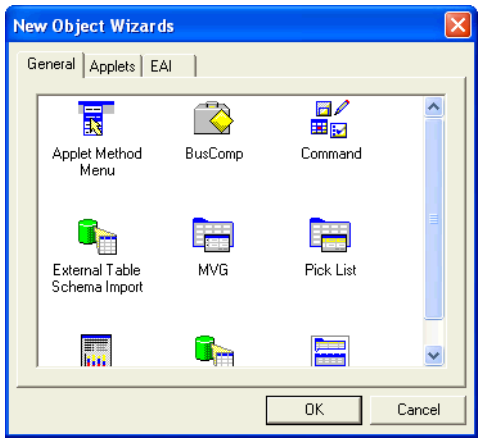
Siebel Tools includes many wizards that guide you through the process of creating new object definitions. Wizards assist you by prompting for key information necessary to define a particular object, such as an applet or a view. Then, based on the information entered, it creates the object definition and any related object definitions.

You do not have to use wizards to create objects. You can also create objects by adding records in the Object List Editor. But this requires thorough knowledge of the Siebel Object Model hierarchy. This guide uses wizards wherever possible.

To open a New Object Wizard

1 Choose File > New Object.

The New Object Wizards dialog box appears, as shown in the following illustration.



2 Choose an object to create.

The selected wizard opens.

Web Layout Editor

Use the Web Layout Editor to drag user interface objects such as applets and controls and drop them into placeholders in Siebel Web templates. You use the Web Layout Editor to design the layout of applets and views. You access the Web Layout Editor by selecting any of the following objects, right-clicking, and then choosing Edit Web Layout.

- Applets and Applet Web templates
- Views and View Web templates
- Web Templates

Figure 11 shows the Web Layout Editor.

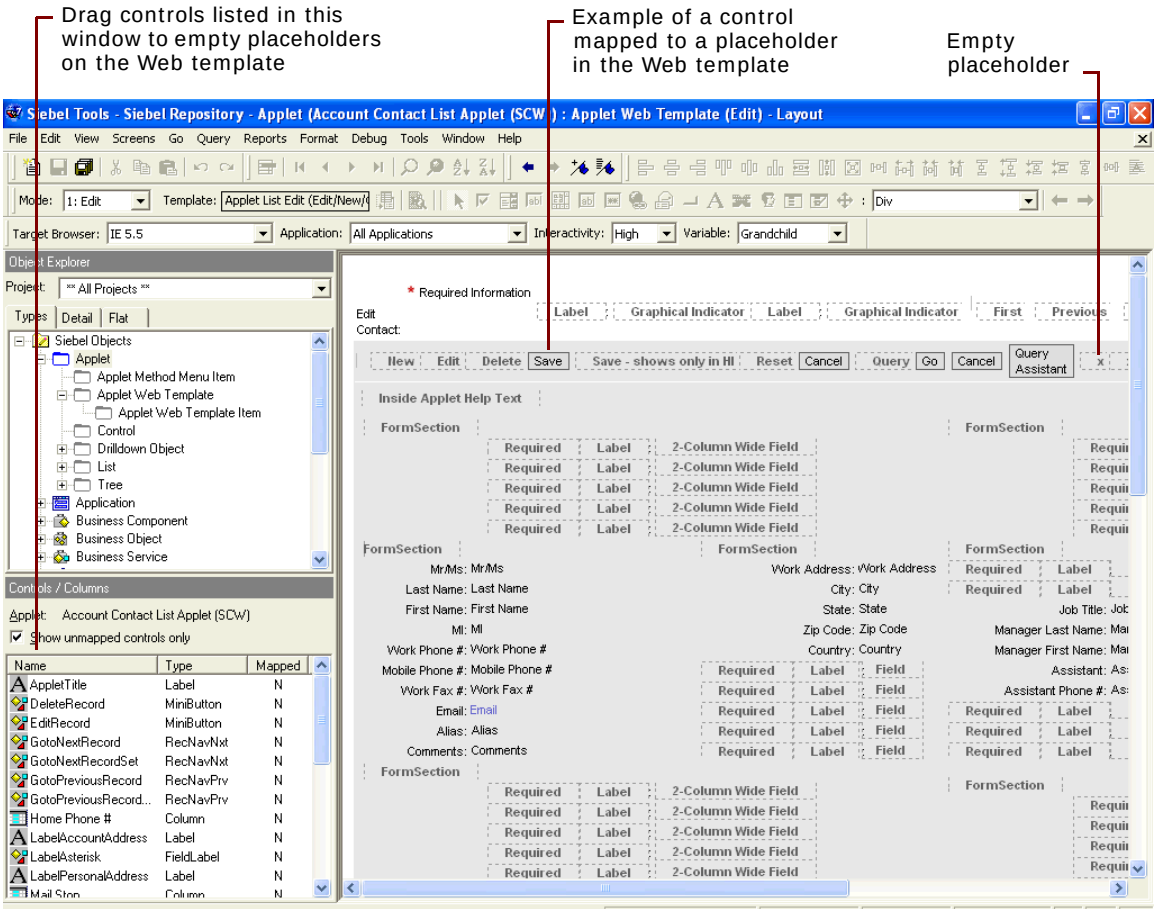


Figure 11. Applet Web Layout Editor

Setting Tools Options

There are several options in Siebel Tools that you can set to facilitate your configuration work. These options are available by choosing View > Options, and then choosing the tab for the options to configure. Some of the options are described below.

- **Language Settings tab.** These settings define your working language in Siebel Tools. Objects can have locale-specific attributes. For example, text strings such as an applet title can be defined in several languages.

  The Tools Language Mode determines the locale-specific data that is:

  - Displayed in the Tools interface
  - Available to edit
  - Compiled to the SRF file

- Transferred to and from the server during the check in and check out process

The current Tools Language Mode is displayed in the status bar located in the lower right corner of the Siebel Tools interface.

To edit locale-specific attributes, select the Enable Language Override check box. When you select this check box you can switch between the following two modes:

- **Base.** When working in this mode, changes made to locale-specific attributes are stored in the base table. The changes apply to every language.
- **Language Override.** When working in this mode, changes are stored in a child locale table. That is, the attribute is overwritten for the current language.

When the Enable Language Override check box is clear, you can work in base mode only.

For more information about language modes, read *Using Siebel Tools*.

- **Check In/Out tab.** This setting defines the development server database and your local client database to be used in the check in and check out process.
- **Web Template Editor tab.** Use this setting to define a default Web template editor. When working in the Web Layout Editor you can click the Edit Template button on the toolbar to open the Web template using a default editor.

The Configuration Process

The typical process for configuration applications using Siebel Tools can be broken down into the following basic steps.

- “Checking Out Projects from the Server”
- “Making Configuration Changes” on page 48
- “Compiling Projects” on page 48
- “Testing Changes” on page 50
- “Checking In Projects” on page 52

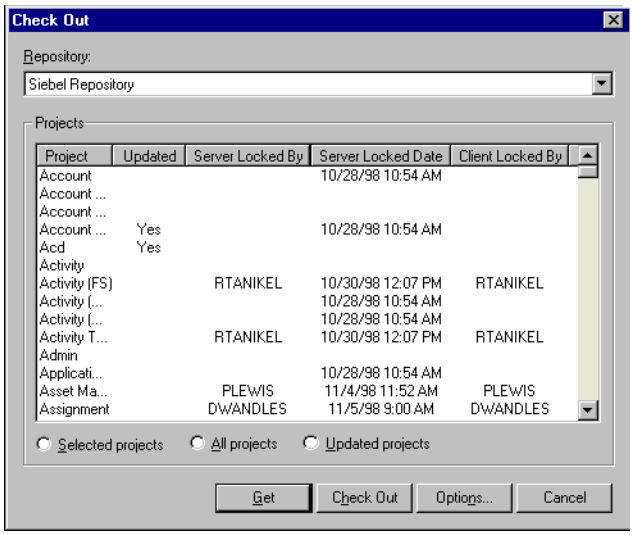
Checking Out Projects from the Server

You typically do your configuration work in a local database, checking out object definitions (projects) from the development server as needed. For an overview of the development environment, read “About the Local Development Environment” on page 58.

To check out a project

1 In Siebel Tools, choose Tools > Check Out.

The Check Out dialog box appears, as shown in the following illustration.



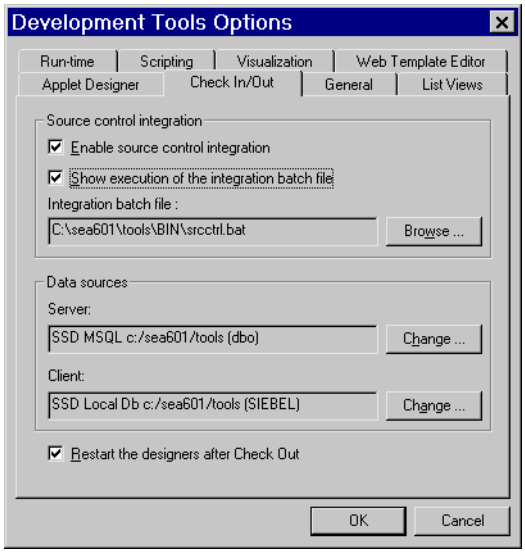
2 Make sure you have selected the correct repository.

The default value is Siebel Repository.

3 Select the project or projects to check out.

4 Click Options.

The Development Tools Options window appears, with the Check In/Out tab active, as shown in the following illustration.



5 In the Development Tools Options window, make sure the Server and Client data sources are specified correctly.

6 Close the Development Tools Options window.

The Check Out dialog box appears.

7 In the Check Out dialog box, click Check Out.

CAUTION: Password encryption must be disabled when checking projects in and out. You disable password encryption in the Tools configuration file (tools.cfg).

Making Configuration Changes

When you check out a project, the project remains locked in the server database and in your local database. This prevents other developers from checking out the project. During this time, you can make configuration changes using Siebel Tools, such as modifying user interface objects, extending business logic, and extending the database.

NOTE: You can also lock projects locally, without checking them out from a server database. Read "Locking Projects in Your Local Repository" on page 53. This is how the procedures in subsequent chapters are written.

Compiling Projects

After you have made your configuration changes, you must compile them into a Siebel repository file. Until you do so, your Mobile Web Client application that reads the repository file does not reflect the changes you have made.

There are various options for compiling the repository. You can compile at the project level—selected projects, all locked projects, all projects—or you can compile individual objects. Compiling individual objects is faster, but you must remember to do it for each object you modify.

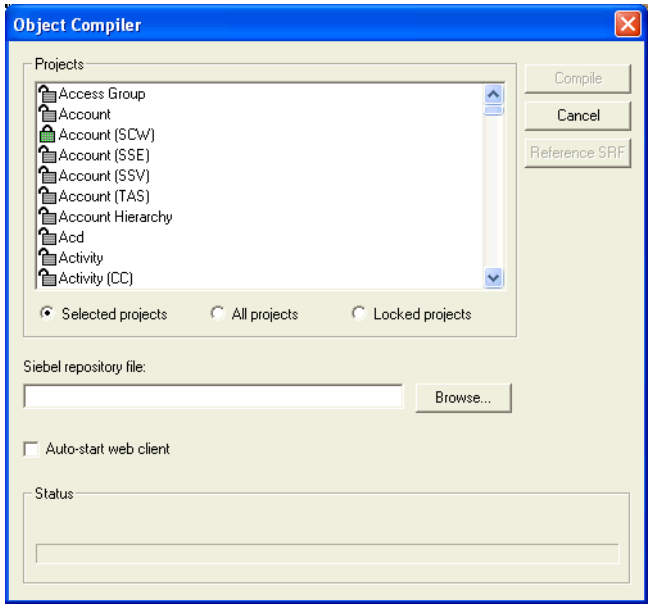
To compile projects

1 Exit Web Client applications that are running on the SRF file to compile.

While running, a client application maintains a lock on the SRF file.

2 In Siebel Tools, choose Tools > Compile Projects.

The Object Compiler window appears, as shown in the following illustration.



3 In the Object Compiler window, indicate whether to compile selected projects, locked projects, or all projects.

Compiling only the locked projects (those currently checked out) is faster than compiling all projects. Compiling locked projects is also often easier than selecting individual projects from the list.

NOTE: Be sure to make a backup copy of the repository file, siebel.srf.

4 In the Object Compiler window, click Browse to select the SRF file to compile.

The SRF file for your application is in the objects directory of your Siebel application client directory. For example, a typical path to an SRF file for a Mobile Web Client installed for testing on a developer's machine is C:\Program Files\Siebel\7.7\web client\OBJECTS\siebel.srf.

NOTE: Do not try to compile to the objects directory of Tools. The SRF file in this directory is locked because the Siebel Tools program itself reads from it constantly as it runs. If you attempt to compile to this filename and path, you will receive an error and be prevented from compiling.

5 In the Object Explorer window, click Compile.

After compilation is successful, the SRF file you specified contains the configuration changes you made.

To compile individual objects

1. In the Object Explorer, select the object type to compile.

   You can only select top level objects such as, applets, views, business components, or business objects. You cannot compile child-level objects.

2. In the Object List Editor, select the object or objects to compile.

   The objects can belong to different projects.

3. Right-click and choose Compile Selected Objects.

   The Object Compiler dialog box appears.

4. In the Object Compiler dialog box, select the repository file (SRF) to which to direct your changes.

5. Click Compile.

   After compilation is successful, the SRF file contains the configuration changes that you made.

Testing Changes

When you have compiled objects into a new repository file, you can see the results of your changes in the client application. Typically developers set up a Mobile Web Client on their machine to use for testing and point the Web Client to the repository file that has been recompiled with configuration changes.

NOTE: Make sure the correct SRF file—the one you compiled your changes to—is defined for the application to test. You can check this by verifying the value of the repository file parameter in the application's configuration file.

This type of testing is considered to be Unit Testing, which is one element of an overall testing strategy.

Registering New Views in the Application

When you add a new view to an application, you must make it visible in the client application before you can test it. You do this by registering the new view in the application using Administration - Application > Views and then associating it to a responsibility.

If you are working with existing views, these steps are not necessary. The views are already registered in the application.

To register new views in the application

1. From the application-level menu, choose Site Map > Administration - Application > Views.

   The View list appears.

2. Click the menu button and then choose New Record.

3 In the View Name field click the select button.

The View picklist appears.

4 Choose a view and then click OK.

5 Enter a description and select whether the view is to be available for local access in the Mobile Web Client.

6 Save the record.

To associate views with a responsibility

1 From the application-level menu, choose Site Map > Administration - Application > Responsibilities.

The Responsibilities list appears.

2 In the Responsibilities list, select the Responsibility to which to associate the view.

NOTE: You cannot edit the responsibilities that are part of the seed data that ships with the product. You must create a new responsibility to be able to associate new views to it. You can copy one of the sample responsibilities, such as Siebel Administrator, and then customize it for your purposes.

3 In the Views list, click the menu button and then choose New Record.

4 From the Add Views dialog box select a view.

The view is associated with the responsibility. When a user with the responsibility logs into the application, the view appears in the user interface.

Verifying Your Changes

After you have made any new views available in the application, you can log in to your Web Client application to verify the results of your configuration changes.

To verify your changes

1 Start your client application.

2 Navigate to the objects you modified.

For example, if you added a new applet to a view, go to the view to make sure the applet appears where it is supposed to.

3 Note any differences between what you find and the expected appearance or behavior.

4 If you find problems, repeat the following cycle:

a Return to Siebel Tools to make configuration changes necessary to fix the problem.

b Compile the project.

c Test the changes.

Checking In Projects

After you confirm completion of your configuration changes, you need to check the project back into the development server. Here are some guidelines for checking in projects.

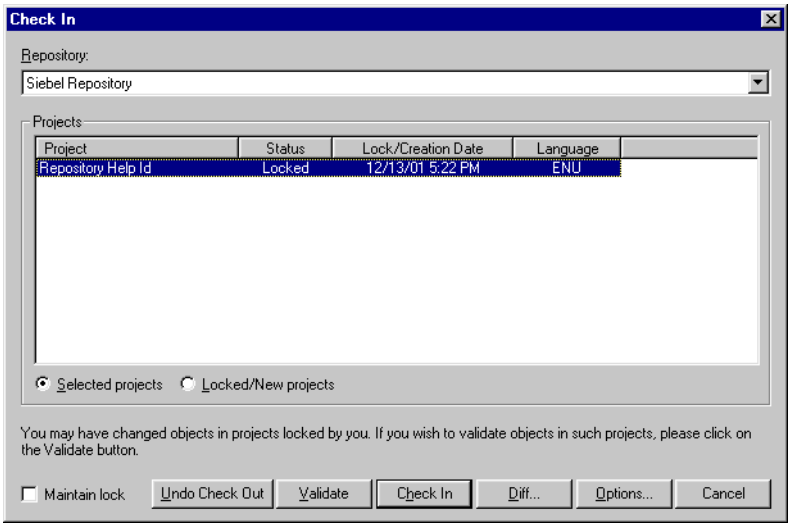
- Test objects before you check them in.
- Validate projects using the Validate button in the Check-In dialog box.
- Check in dependent projects at the same time to make sure the configuration on the server remains consistent.
- Keep in mind that the work of other developers may depend on the objects you are configuring. In some cases, this may require you to check in projects before your work is complete because other developers may be dependent on a feature you have added to your project.

For overview information about the development environment, read "About the Local Development Environment" on page 58.

To check in changes

1. Choose Tools > Check In.

   The Check In dialog box appears, as shown in the following illustration.



2. Click Options.

   The Development Tools Options dialog box appears.

3. As you did during check-out, make sure the server and client Data Sources are pointing to the correct databases.
4. Close the Development Tools Options dialog box.
5. Click the Locked/New Projects option button.
6. Click Check In.

Other Key Tasks for Managing Object Definitions

This section covers other tasks that are useful for working with object definitions.

Viewing Project Differences Before Check In

Before checking in changes, you can view the differences between projects on your local repository and the projects on the server repository. This is useful for finding errors or omissions before committing your changes to the server.

To view project differences

- Click the Diff button in the Check In dialog box.

  Siebel Tools displays the checked out projects that have been modified since the last check out.

Locking Projects in Your Local Repository

When you are prototyping ideas or are making changes to object definitions that you plan to discard, it is good practice to lock the project locally rather than checking it out from the server. This way the project remains unlocked on the server, allowing other developers to check out the project from the server, while still allowing you to modify and test changes in your local repository.

To lock a project locally

- In the Tools Object List Editor select the project or other object type to modify, then choose Tools > Lock Project.

  Tools locks the project in your local repository so you can make changes. It does not lock the project on the server database. You cannot check your changes back to the server, so be aware that any changes you make are not saved.

NOTE: Do not lock a project locally on the server repository.

Exporting Object Definitions

You can export object definitions making them available to import into another repository. This is useful when you are working in a multiple repository development environment and want to share object definitions.

To export object definitions

1. In the Object Explorer, select the object type to export.
2. In the Object List Editor, select the object or objects to export.

3 Choose Tools > Add to Archive.

The Export to Archive File dialog box appears.

4 If you want to include additional objects in the export file:

a Do not click Save.

b Return to the Object Explorer and select the object type to export.

c Repeat Step 2 and Step 3.

NOTE: You can remove objects from the Export to Archive dialog box by selecting the object and pressing Delete.

5 Enter the path to the directory where the archive file is to be to stored.

6 Click Save.

The object or objects are saved as files with an SIF extension.

NOTE: This file overwrites any existing SIF files with the same name. You cannot add objects to an existing export file.

Importing Object Definitions

You can import object definitions from an archive file into your local repository. To import object definitions you need to have access to the archive files (SIF) and the target repository needs to be the active repository on your local machine.

NOTE: A local or server database can contain more than one repository. However, Siebel Tools only shows information from only one repository at any time. This is the active repository. You can change the active repository by selecting the Repositories object type in the Object Explorer window or by choosing File > Open Repository.

To import archive files

1 Lock the projects that will contain the objects you are importing.

NOTE: If you do not lock the projects affected by the import, the Import Wizard prompts you to lock them in the second dialog box of the wizard.

2 Choose Tools > Import From Archive.

The Select Archive File to Import dialog box appears.

3 Find the object archive file (SIF) to import.

4 Click open.

The Import Wizard opens and the object archive file opens.

5 Select a conflict resolution option.

6 Click next.

The Import Wizard checks for differences between the object definitions in the current repository and the object definitions in the archive file.

7 If there are no differences, the object definition is imported.

If there are differences, the Import Wizard displays the following:

- **Conflicting Objects.** Shows the object types in which differences were found.
- **Object Differences.** Shows whether the object exists in the archive file only, in the current repository, or both, and shows the conflict resolution action that has been defined. You can change the action if necessary.
- **Attribute Differences.** Shows the object attributes that have different values. If the action for the object difference is merge, you can update the value in the Resolution column by selecting the attribute, right-clicking, and then choosing Repository or File.

5 Setting Up a Developer's Local Database

After you have installed software as described in Chapter 3, "Installing Siebel Applications," but before developers can do any configuration work, each developer must be set up with a local database. The process for setting up a local database includes tasks typically performed by an administrator and tasks performed by each developer. For example, an administrator creates database users and runs Siebel Remote server tasks to generate local databases, then each developer initializes their local database by downloading repository data from the server. For a list of tasks and where in the application they are performed, see Table 7.

NOTE: This chapter gives you an overview of the sequence of tasks that need to be performed to set up a developer's local database. It is not necessary to complete these tasks to be able to follow the configuration examples in Chapter 6, "Getting Started in the User Interface Layer" through Chapter 10, "Modifying the Look and Feel of the Web Client." In those chapters you can log in to the SAMPLE database using the SADMIN user name and password.

Table 7. Tasks for Setting Up a Developer's Local Database

Task	Where Performed
"Setting Up Database Users" on page 59	RDBMS Software
"Creating Positions" on page 59	Administration - User screen
"Associating Responsibilities" on page 60	Administration - User screen
"Setting Up Developers as Siebel Employees" on page 60	Administration - User screen
"Setting Up Developers as Mobile Web Clients" on page 61	Administration - Siebel Remote screen
"Generating a New Database Template" on page 62	Administration - Server Configuration screen
"Extracting the Local Database" on page 65	Administration - Server Configuration screen
"Initializing Each Developer's Local Database" on page 67	Siebel Tools
"Performing a Full Get" on page 67	Siebel Tools

About the Local Development Environment

Developers working with Siebel Tools work on a *local database*. A local database is a snapshot of the server database that is stored on a user's machine. The data in each local database is a subset of the server database, as determined by each user's visibility rules. The development environment is illustrated in Figure 12.

Developers check out object definitions from the server database, then after making and testing changes on their local machines, they check them back into the server database.

- **Check out.** Copies selected projects from the server database to the local database and locks the projects on the server. Locking the project on the server prevents other developers from checking it out.
- **Check in.** Copies selected projects (including any changes made by the developer) from the local database back to the server database and releases the lock on the server.

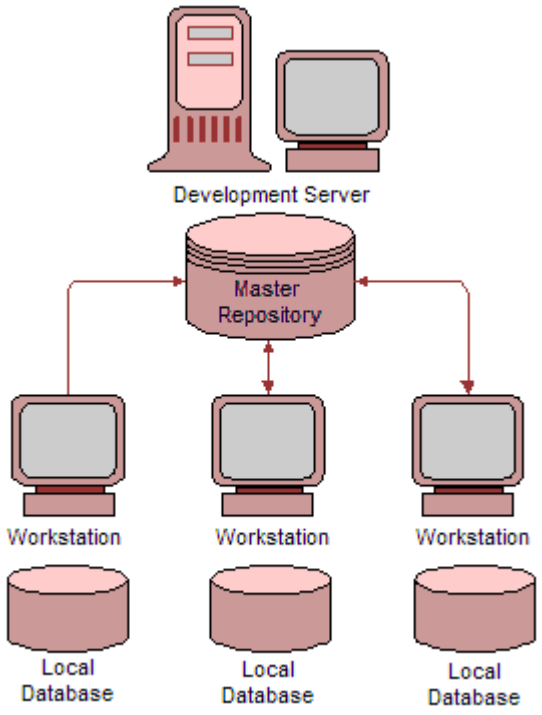


Figure 12. Development Environment

Working on a local database using check in and check out, rather than directly on the server database, provides the following benefits.

- Allows you to roll back unwanted changes without overwriting the work of other developers. Working on a local database gives you the option of not checking in changes to the server. Until you check in changes, the server database remains a clean backup.

- Allows several developers to concurrently use a single repository.
- Allows developers to prototype configuration changes by locking a project locally rather than checking out and locking the project on the server.
- Allows you to work remotely because you are not required to be connected to the development server.

NOTE: Siebel Web templates are not included in the check in and check out mechanism used to control object definitions. You must manage Siebel Web template files using a third-party source control application or a manual process. Read "Moving Modified Web Templates and Related Files" on page 177.

For information on how to perform check in and check out, read "Checking Out Projects from the Server" on page 46 and "Checking In Projects" on page 52.

Setting Up Database Users

Each user accessing a local database must be set up with a database user account. Work with your database administrator to add database accounts on the appropriate database and then add these accounts to the group SSE_ROLE. The exact steps for adding users and placing them in this role group depend on the database software you are using.

The user name used for the account is the name that appears in Siebel Tools when a developer locks a project. Make sure the user name clearly identifies the developer. Establish a convention in choosing user names, such as the first initial plus the last name.

For more information about setting up database users, read the product documentation for the specific database application your organization uses.

Creating Positions

Positions represent an actual job position in your organization. Positions determine which records are available to each user. A position can be any appropriate job title, such as the developer's actual job title or simply the word "Developer." For example, assume NREC set up positions for two developers, called Developer 1 and Developer 2.

There are several positions that are part of the seed data that ships with the Siebel product. However, none fit the role of the developer, so you need to create a new position for the users on your development team.

For more information about positions, read *Siebel Applications Administration Guide*.

NOTE: The position information is used by the Siebel Remote's routing rules and may affect the outcome of a database extraction. For more information, read *Siebel Remote and Replication Manager Administration Guide*.

Associating Responsibilities

Responsibilities determine which views users have access to. For developers, the responsibility must include access to the views necessary to perform testing and development tasks. The easiest choice is to assign developers the Siebel Administrator responsibility. It is already defined as seed data and it provides access to *every* view.

Because the Siebel Administrator responsibility is already defined as part of seed data, you do not need to create a new one. You can associate the users to this responsibility when you register the developer as an employee in "Setting Up Developers as Siebel Employees."

An alternative to using a predefined responsibility is to create your own.

For more information about responsibilities, read *Siebel Applications Administration Guide*.

Setting Up Developers as Siebel Employees

After developers have a database account, and the necessary positions and responsibilities are defined, you can set up each developer as an employee in your Siebel application.

To set up an employee user

1. Log in to the development server with a user name and password that have administrator privileges, such as the SADMIN user name and SADMIN password.
2. From the application-level menu, choose Site Map > Administration - User > Employees.

   The Employee list appears.

3 In the Employees list, add a new record and complete the fields in the following table.

Field	Example Value	Comments
First Name	Pat	The developer's first name.
Last Name	Adams	The developer's last name.
User ID	PADAMS	The user name to be used for logging on to Siebel Tools (and for identifying which user has checked out a project). The login name must match the user's database account name. Enter the login names in uppercase letters for compatibility across databases.
Password	NREC	The user's password to the application is the same as the user's password on the database account. This field is read-only.
Position	Developer_1	Positions determine which records are visible to users.
Responsibility	Siebel Administrator	Responsibilities determine which views a person has access to. Developers typically need access to every view for the purpose of testing and entering test data. For this reason, NREC chose to associate the Siebel Administrator responsibility to each developer. An alternative is to create a new responsibility that only allows developers access to a particular set of views.

NOTE: The employee information for each developer resides on the development server. When you create a local database for the developer, this information is also included in each local database.

Setting Up Developers as Mobile Web Clients

After setting up developers as Siebel Employees, you must register developers as mobile Web clients. This provides Siebel Remote with the information it needs to create a local database file for the developer and populate it with the appropriate data.

To register a developer as a new mobile Web client

1 Log in to your Siebel application using an account (for example, SADMIN) with access to the Administration - Siebel Remote screen.

NOTE: Although Siebel Remote is not used for the check-in and check-out of projects, you must set up developers as mobile Web clients because Siebel Remote uses this information to retrieve the local database during initialization.

2 From the application-level menu, choose Site Map > Administration - Siebel Remote > Mobile Clients.

The Mobile Client view appears with mobile Web clients for the Parent Server displayed.

3 In the Parent Server form, choose the appropriate parent node.

4 In the Mobile Clients list, click New.

5 Complete the fields in the following table.

Field	Example Value	Description
Mobile Client	PADAMS	Enter the mobile Web client name in uppercase letters, with a maximum of eight characters. The User ID makes a good mobile Web client name. The name cannot include spaces, periods, or other invalid characters for a DOS filename. It can contain only single-byte, alphanumeric characters, and the underscore (_) and hyphen (-) characters.
User ID	PADAMS	This value is used to log into the local database during initialization.
Routing Model	MOBILE CLIENT - EXTRACT ONLY	This value identifies the routing model to which the mobile Web client belongs. The MOBILE CLIENT - EXTRACT ONLY model does not allow synchronization. It is used simply for a snapshot of data, which is appropriate in the case of a developer.

For more detailed information about mobile Web clients, routing models, language preference, and Siebel Remote, read *Siebel Remote and Replication Manager Administration Guide*.

Generating a New Database Template

A database template is a cached representation of Siebel tables and indexes that are stored in a database file (DBF). You create a database template on the server, in this case the development server. Siebel Remote uses the template to create the local database for each developer. You create the database template by running the Generate New Database server component (GenNewDb).

NOTE: Before you can run server components, the component group to which it belongs must be enabled at the enterprise-level. If you enabled Siebel Remote during the server installation process, this is already done for the Remote component group. However, if you did not select Siebel Remote during the installation process, you need to enable it manually and then synchronize batch components. For instructions, read *Siebel System Administration Guide*.

For detailed information about Siebel Remote, read *Siebel Remote and Replication Manager Administration Guide*.

Running the Generate New Database Server Component

The following procedure explains how to generate a new database template.

To generate a new database template (GenNewDb)

1. From the application-level menu, choose Navigate > Site Map > Administration - Server Management > Jobs.
2. In the Jobs list, click New.
3. In the Component/Job field, select Generate New Database.
4. In the Requested Server field, enter the name of the server on which to run the GenNewDb job.

   After the job completes, the Execution Server field displays the name of the server that actually ran the job.
5. In the Job Parameters list, click New.
6. In the Name field, click the select button and use standard query techniques to select SQL Anywhere Database, and then click OK to return to the main window.

   The Value field typically defaults to sse_utf8.dbf and appears automatically. To use an encrypted template file, replace this value with sse_encr.dbf.
7. Modify the values of other parameters as necessary by repeating Step 5 and substituting different parameter names in Step 6.

   Table 8 lists the parameters and default values for the Generate New Database component.

   Set the UseDdlFile parameter to FALSE only when you run the Generate New Database component after a schema change.

8 In the Jobs list, with the Generate New Database job still selected, click Start.

A new database file generates. Typically, this takes a few minutes.

Table 8 shows the start-up parameters for generating new database components.

Table 8. Selected Parameters for the Generate New Database Component

Name	Alias	Required or Optional	Default Value and Usage Notes
SQL Anywhere Database	DbfFile	Required	SQL Anywhere database filename to initialize. The default value, sse_utf8.dbf, is a Unicode file that supports all languages. Alternatively, sse_encr.dbf provides standard Sybase encryption of the local database template.
DBA Password	DbaPwd	Optional	Password for the DBA account. Default for SQL Anywhere is SQL. Set the DbaPwd parameter to the password for the DBA user ID in the empty database template file.
New DBA Password	NewDbaPwd	Optional	The password assigned to the local database administrator account on the Mobile Web Clients. A parameter value specified at component run time is used only for the current run of the Generate New Database component. If no value is specified at run time, the value specified in an administrative view is used. If no value is specified at component run time or in an administrative view, the default value is the first 8 characters of the enterprise name. If the enterprise name has less than 8 characters, the name is padded with consecutive digits, for example 1234...
Table Space	TSpace	Optional	Space name in DB template to store Siebel tables. Do not specify the Table Space parameter unless you intend to build a custom empty database file using the specified table space.

Table 8. Selected Parameters for the Generate New Database Component

Name	Alias	Required or Optional	Default Value and Usage Notes
Index Space	ISpace	Optional	Space name in DB template to store Siebel indexes. Do not specify the Index Space parameter unless you intend to build a custom empty database file using the specified index space.
Use Transaction Log File	UseTxnLog	Optional	Use when creating a new template file. The default is TRUE.
Use DDL File	UseDdlFile	Optional	Use when creating a new template file. The default is FALSE—it means that the schema is read directly from the database. If the value is set to TRUE, it means the schema is read from the DDL file. When a schema change takes place in your environment, set the UseDDLFile parameter to FALSE. GenNewDb then reads the latest schema from the database rather than the DDL file.
Interface Tables	IFaceTbls	Optional	Create interface tables and indexes. The default is FALSE.
Warehouse Tables	WarehouseTbls	Optional	Create Warehouse tables and indexes. The default is FALSE.
Client Db Type	ClientDbType	Optional	Client database engine type. The default is SQL Anywhere.

Extracting the Local Database

The Database Extract server component extracts data from the server database for each mobile user and temporarily store it in a compressed file. The data in this file is used to populate the user's local database during the database initialization process.

NOTE: The mobile user must have a valid position in the organization's reporting hierarchy for the database extract to be successful. For more information about positions and organizations, read *Siebel Applications Administration Guide*.

For detailed information about Siebel Remote, read *Siebel Remote and Replication Manager Administration Guide*.

To run a database extract for a Mobile Web Client

1. From the application-level menu, choose Navigate > Site Map > Administration - Server Management > Jobs.
2. In the Jobs list, click New.
3. In the Component/Job field, select Database Extract from the picklist.
4. In the Requested Server field, enter the name of the server on which to run the Database Extract job.

   NOTE: After the job completes, the read-only Execution Server field displays the name of the server that ran the job. For a Database Extract Job, this is the same as the Requested Server.

5. In the Job Parameters list, which is located below the Jobs list and the Job Detail form, click New and add the necessary parameters.

   The required parameter for Database Extract is Client Name.

   The value for the Client Name parameter is the name of the Mobile Web Client.

6. In the Jobs list, with the Database Extract record still selected, click Start.

   The Mobile Web Client database is extracted. This may take a few minutes.

Sample Directory Tree After Running Database Extract

Each registered mobile Web client requires a separate directory on the Siebel Remote server. The Database Extract program creates the appropriate directories for each mobile Web client.

NOTE: The installation program also places a directory named txnproc in the docking directory within the Siebel server root directory. Do *not* modify the contents of this directory under any circumstances.

The following example shows a portion of the server directory tree after you run Database Extract for mobile Web clients (and developers) named PADAMS and SSCOTT.

```
siebel
   docking
      padams
         inbox
         outbox
      sscott
         inbox
         outbox
      txnproc
```

Running the Database Extract task creates a database snapshot for a given user, which consists of multiple files. These files contain the data required to initialize the user's local client database, and are placed in the directory server*siebel_srvr_root*\docking\user\outbox. Each mobile user downloads these files to create a local database and local file system (copies of literature files).

Initializing Each Developer's Local Database

After you extract each developer's local database, developers must now initialize their local databases. Initialization creates a local database file, called sse_data.dbf and stores it on the developer's local machine. Siebel Tools includes an initialization program that creates this file.

To initialize a developer's local database

1. Go to the machine on which Siebel Tools is installed.

   For the first developer, NREC used DEV_tools_A; for the second developer, NREC used DEV_tools_B. For information about the machines on which NREC installed different software, see Table 5 on page 33.

2. Log in and connect to the Local database.

   The following message appears.

   "The local Siebel database was not found. Would you like to connect to the Siebel Remote server to initialize the local database?"

3. Click Yes.

   Siebel Tools connects to the Siebel Remote server and initializes the developer's local database.

   If initialization is successful, the sse_data.dbf file appears in the *tools_root*\local directory (where *tools_root* is the directory in which Siebel Tools has been installed, for example c:\siebdev).

Performing a Full Get

After initializing a local database, you must populate it with a read-only copy of the projects and object definitions stored on the server database. This process is called a *full get*. A full get is equivalent to checking out every project from the server, however the projects are not locked and you cannot modify the object definitions until you check out a project from the server database or lock a project locally.

For more information about the check in and check out process, read "About the Local Development Environment" on page 58.

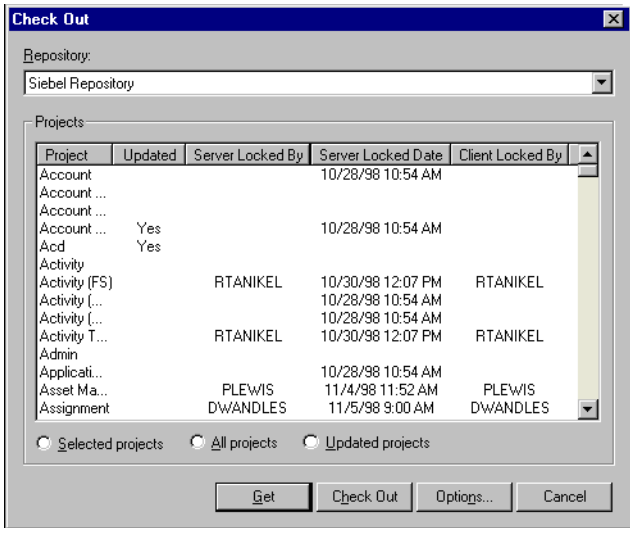
Having a read-only copy of the full repository on your local machine allows you to compile a Siebel repository file for your local database. This is required for testing changes locally.

To perform the initial full get of every project from the repository

1. Start Siebel Tools, choosing Local in the Connect field.

2 Choose Tools > Check Out.

The Check Out dialog box appears, as shown in the following illustration.



3 Choose the name of your development repository from the Repository pick list.

4 Click the All projects option button.

5 Click Options.

6 In the Development Tools Options window, make sure that your Server Data Source points to your server development database and your Client Data Source points to the local database you previously initialized and are currently running against.

These data sources must match those shown in the Control Panel, under ODBC Data Sources.

7 In the Check Out dialog box, click Get.

All objects from the server repository are copied to your local repository, meaning that your currently open local repository has the same contents as the server repository from which you did the full get.

For more information about working with projects, read *Using Siebel Tools*.

6 Getting Started in the User Interface Layer

This chapter gets you started defining and modifying object definitions using Siebel Tools. Following the NREC example, you complete configuration tasks focusing on the user interface level of the object hierarchy. Working through the tasks in this chapter helps you understand the hierarchy of the major user interface objects—applications, screens, views, and applets—and provides you with hands-on experience with basic configuration.

The NREC example helps place the tasks in context of making configuration changes to meet business needs. The chapter starts off by leading you through the process of narrowing down the preconfigured application to just those screens and views relevant to NREC. In the latter part of the chapter, you configure existing applets based on NREC requirements. You can follow these tasks like a tutorial, configuring against your sample database.

This chapter includes the following topics:

Creating an NREC Project

Before configuring object definitions, create a new project for NREC. You use this project to group the new and modified object definitions. This makes it easier for you to find your work and makes it easy to export and import the object definitions.

To create a new project

1. In the Object Explorer, select the Project object type.

   The Projects window opens in the Object List Editor.

2. In the Projects window, right-click and choose New Record.

3 Enter a new record with the following properties.

Property	Value
Name	NREC Configuration
Locked	TRUE

The project is now available for you to use. Moving the objects that you create or modify helps you keep track of your configuration changes.

Setting Your Target Browser

Use the Target Browser feature of Siebel Tools to configure applications conditionally for different browsers, for example, Internet Explorer 5, Netscape 4.7, Netscape 6. Although you do not configure for different browsers in this guide, Siebel Tools requires that you have a target browser set. Otherwise, you can not open the Web Layout Editor.

To set a target browser

1 Choose View > Toolbars > Configuration Context.

The Configuration Context toolbar appears.

2 From the drop-down list in the Target Browser field, choose Target Browser Config.

The Target Browser Configuration dialog box appears.

3 From the list of available browsers, select the target browsers and move them to the Selected pane—for example, select IE 5.0.

Creating Strings

In Siebel applications, the text that is displayed on the screens and views is stored in the repository and compiled to an SRF file. This includes the names of every screen, view, and field. For example, when you display information on accounts, the view has fields with names such as Account Name, Account Team, Site, Status, Territory, and so on. These names are stored in the SRF file. When you display a page, the server uses the SRF file to determine which fields to show, and what text (strings) are associated with each field.

This model of keeping the strings separate from the view definitions is called *symbolic strings*. It may seem awkward to not have text directly on a view definition, but there are some strong advantages to using symbolic strings.

- Reduces redundancy because many objects can refer to one symbolic string. For example, you only need to define the term “Account Team” one time. Every time you want to use “Account Team” on other views, you use the same link as you did on the first view.

- Results in a more consistent user interface. By reusing the same symbolic string, you avoid having slightly different phrasing used on similar views. For example “Acct. Team”, “Act. Team”, “Account Group”, “Client Team”, and so on.
- Simplifies maintenance because you only have to maintain one string for a given term. If you need to change the name from “Account Team” to “Client Group”, you update the symbolic string and that change then appears in all views that use that symbolic string. This is much more efficient and effective than having to change each view individually.
- Simplifies translations by eliminating duplicated translations of the same word. Because the symbolic strings are reused, the same term does not exist multiple times.
- Reduces localization costs, time, and inconsistencies. If the term “Account Team” is on five views, it would need to be translated five times. By using symbolic strings, the term is translated once. This saves on localization costs and reduces the overall time for the project. The use of symbolic strings also helps prevent inconsistent translations. Although the NREC example does not cover localization, many custom implementations of Siebel applications are localized.

How the Symbolic Strings Model is Implemented

Symbolic strings are implemented using a top-level object in the Siebel repository called Symbolic Strings and a child object called Symbolic String Locale. Each symbolic string record represents a word or phrase, for example Account or Contact, and is language-independent. All translations of that word or phrase, including English, are stored as child symbolic string locale records. User interface objects refer to symbolic string records for text strings. The literal display value is retrieved from one of the several translations stored as symbolic string locale records based on the current Tools language mode.

Strings Not Included in the Symbolic Strings Model

The symbolic strings model includes text strings stored in the repository and referenced by UI objects such as Applet Titles, Control Captions, and List Column Display Names. The symbolic string model does not include other types of strings typically supplied as seed data, such as LOVs, error messages, and predefined queries.

Creating Symbolic String References

You create new symbolic strings in Siebel Tools. Symbolic strings created by Oracle are included in the Symbolic Strings project. Create all custom symbolic strings in your custom projects, for example the NREC Configuration project.

NOTE: To be able to create symbolic strings, the EnableToolsConstrain parameter in the tools.cfg file must be set to FALSE.

The Symbolic String object is hidden by default. To work with symbolic strings you must make it visible.

To expose the Symbolic String object

1. In Siebel Tools, choose View > Options.
2. In the Development Tools Options dialog box, click the Object Explorer tab.
3. Click the check box for the Symbolic String object, and then click OK.

   The Object Explorer now shows the Symbolic String object.

To create a symbolic string

1. Check out the project in which to create the Symbolic String.
2. In the Object Explorer, select the Symbolic Strings object type.
3. In the Object List Editor, create a new record using the following table to complete the necessary fields.

Property	Description
Name	Unique name of the symbolic string. Siebel Tools enforces a predefined prefix for the symbolic string name, such as X_. This helps you distinguish custom symbolic strings from those created by the Siebel System (SBL_). The value used for the prefix is defined in the SymStrPrefix parameter in the tools.cfg file.
Current String Value	Calculated value based on the current Tools language mode and the String Value property of the corresponding child Symbolic String Locale object.
Definition	Description of the symbolic string.

Some of the customizations you make for the NREC project involve adding new fields or changing the names of existing fields. You make these changes by adding new symbolic strings or by editing the Symbolic String Locale of an existing string. To get started on this effort, add the new symbolic strings as shown in the following table.

Name	Current String Value	Project
X_Bathrooms	Bathrooms	NREC Configuration
X_Bedrooms	Bedrooms	NREC Configuration
X_House_ID	House ID	NREC Configuration
X_Number_of_Bathrooms	Number of Bathrooms	NREC Configuration
X_Number_of_Bedrooms	Number of Bedrooms	NREC Configuration
X_Square_Feet	Square Feet	NREC Configuration

Notice that as you create each symbolic string, a Symbolic String Locale record is automatically added that has ENU as the language. This is because the Siebel Tools language mode is set to ENU.

Inactivating Screens

NREC's implementation of the Siebel Partner Portal application does not include every screen that comes with the product, such as Service, Solutions, and Campaigns. These screens are not part of the design and NREC does not plan to use them in the future. Therefore, NREC has decided to remove the screens from the user interface.

NREC has chosen to remove the screens by inactivating objects in the repository. Screens appear in the user interface as first-level navigation tabs and as links on the Site Map, as shown in Figure 13 on page 73. These links are controlled by two child objects of the Application object, Page Tabs and Screen Menu Items. You can remove these links by making the object types inactive.

- Page Tabs control the first-level navigation tabs in the user interface.
- Screen Menu Items control the links that appear on the Site Map.

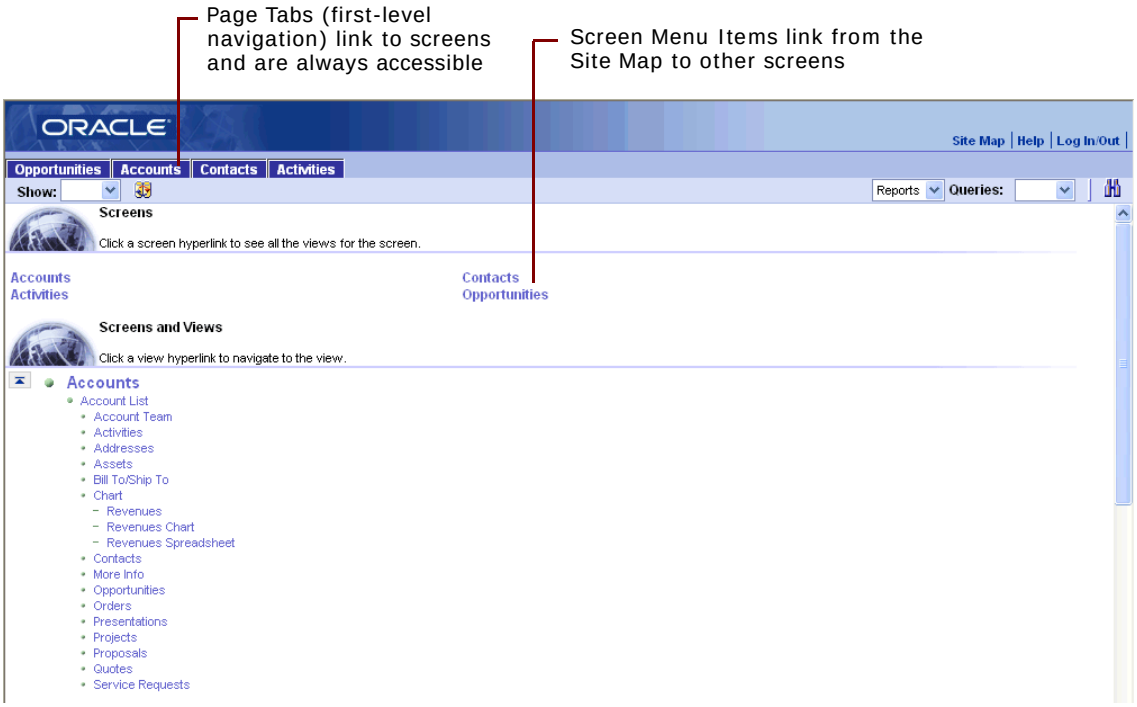


Figure 13. Page Tabs and Screen Menu Items

An alternative approach is to create a responsibility that does not include any of the views within the screen and assign users to that responsibility. This removes the screens from the user interface, but leaves the screen objects active in the repository. This approach allows you to add the screens back in the user interface without having to recompile the repository file. However, because these screens are not part of the design, assume NREC has chosen to inactivate the Screen Page Tab and Screen Menu Items in the repository.

To inactivate screen page tabs and screen menu items

1 In the Object Explorer, expand the Application object type.

2 Select the Siebel eChannel application.

NOTE: The application object named Siebel eChannel is the application called Partner Portal. The employee application is Siebel Partner Manager.

3 Lock the project locally by choosing Tools > Lock Project.

The project to which the application belongs (Siebel eChannel) is locked and a Pencil Icon appears in the W column indicating that you can modify the record.

4 Change the value for the Project property from Siebel eChannel to NREC Configuration.

This moves the application object to the NREC project making it easier for you to track changes.

5 In the Object Explorer, select the Page Tab object type (child of Application).

The Page Tabs window appears in the Object List Editor.

6 Inactivate unnecessary screens listed in the Page Tabs window by selecting the check box in the Inactive field for each record.

Inactivate every Page Tab record *except* the following.

- Accounts Screen (SCW)
- Activities Screen (SCW)
- Contacts Screen (SCW)
- Opportunities Screen (SCW)

NOTE: These are the screens that remain active in NREC's application. The others are not part of the design. SCW is a designation used for many of the Siebel Partner Portal screens, views, applets, and so on.

The inactive Page Tabs turn red in the repository and do not appear as first-level navigation tabs user interface at run time.

7 In the Object Explorer, select the Screen Menu Item object type (child of Application).

The Screen Menu Items window appears.

8 Inactivate the unnecessary screens listed under Step 6 on page 74 in the Screen Menu Item window by selecting the Inactive field for each record.

Inactive Screen Menu Items do not appear on the Site Map at run time.

9 Compile and Unit Test your changes.

a Choose Tools > Compile Projects.

b In the Object Compiler dialog box, select the Selected Projects or Locked Projects option button and then define the path to the Siebel repository file (SRF) to which you are compiling your changes.

Typically, this is the SRF file used by the Mobile Web client installed on your machine for testing. For example:

C:\Program Files\Siebel\7.7\web client\OBJECTS\ENU\siebel.srf

NOTE: Be sure to make a back up of the siebel.srf file before you compile any changes.

c In the projects list, select the NREC Configuration project.

d Click Compile.

Siebel Tools compiles the object definitions for the locked project to the repository file.

For more information about compiling, read "Compiling Projects" on page 48.

e Open the Partner Portal application.

The results of your changes appear when you start Siebel Partner Portal.

Inactivating Views

Each screen in the Siebel application has a set of related views. Views are associated to a screen using the Screen View object type, which is a child object of the Screen object type. These views appear in the user interface below the screen tabs as shown in Figure 14.

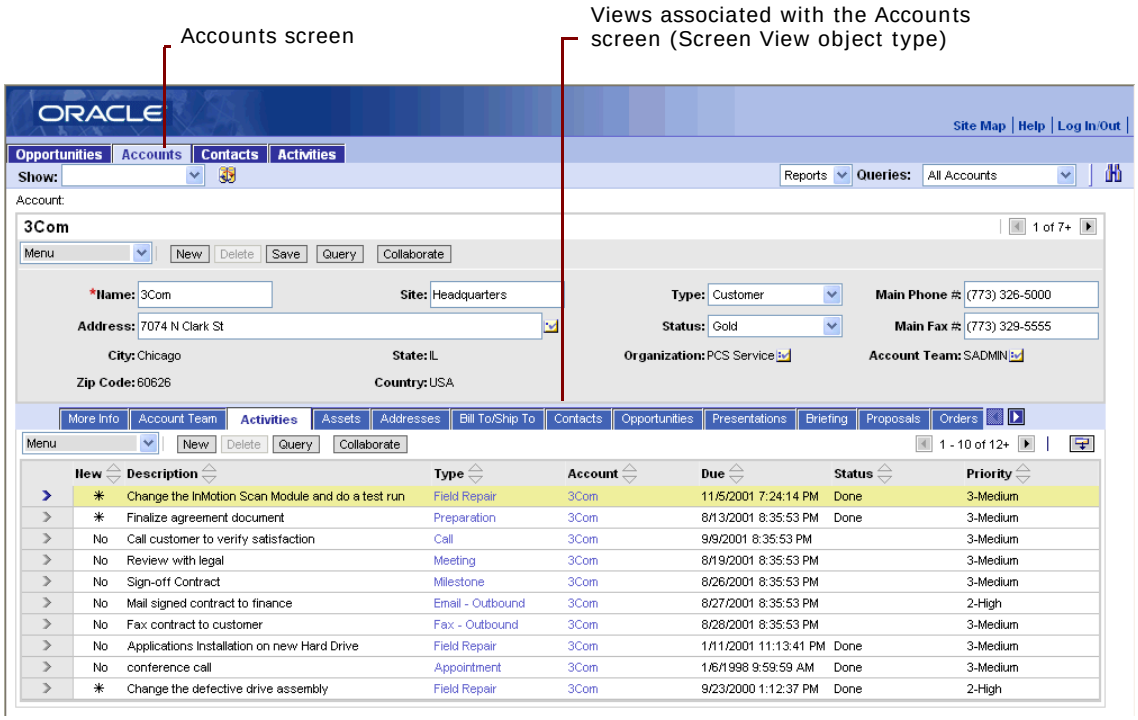


Figure 14. Screens and Views

There are four levels of navigation in Siebel applications. The first level is the row of page tabs across the top of the frame for navigating to other screens. The second level is the Show drop-down list to select different context views (for example My Accounts, All Accounts, My Team's Accounts). The third level is the second row of tabs to navigate to other views. The fourth level (not shown in Figure 14) is a drop-down list to select different grandchild views. For more information about configuring navigation levels, read *Configuring Siebel Business Applications*.

Many of the views associated with screens show data or contain functionality that is not part of NREC's design. For example, many of the views associated with the Opportunities Screen, such as Attachments, Presentations, Proposals, Quotes, and Sales Teams, are not part of NREC's Partner Portal solution.

NREC has chosen to inactivate the unnecessary views associated with each of the remaining, active screens. Table 9 summarizes the views that are to be inactive.

Table 9. Active Screens and Views in NREC's eChannel Application

For these screens...	Inactivate these views
Accounts Screen (SCW)	Account Detail - Account Team View (SCW)
	Account Detail - Assets View (SCW)
	Account Detail - Orders View (SCW)
	Account Detail - Projects View (SCW)
	Account Detail - Quotes View (SCW)
	Account Detail - Revenue Schedule Chart View (SCW)
	Account Detail - Revenue Schedule View (SCW)
	Account Detail - Revenue Schedule View DC (SCW)
	Account Detail - Service Requests View (SCW)
	Account Presentations View (SCW)
	Account Proposals View (SCW)
	SI Com Account Briefing View (eApps)
Activities Screen (SCW)	FS Activity Part Movements

Table 9. Active Screens and Views in NREC's eChannel Application

For these screens...	Inactivate these views
Contacts Screen (SCW)	Contact Detail - Service Request View (SCW)
Opportunities Screen (SCW)	Opportunity Detail - Attachments View (SCW)
	Opportunity Detail - Products View (SCW)
	Opportunity Detail - Quotes View (SCW)
	Opportunity Detail - Revenue Schedule Chart View (SCW)
	Opportunity Detail - Revenue Schedule View (SCW)
	Opportunity Detail - Revenue Schedule View DC (SCW)
	Opportunity Presentations View (SCW)
	Opportunity Proposals View (SCW)
	Oppty Chart View - Campaign Analysis
	Oppty Chart View - Campaign Pipeline Analysis
	Oppty Chart View - Current Opportunity Analysis
	Oppty Chart View - Pipeline Analysis
	Oppty Chart View - Quality Analysis by Rep

There is more than one approach to controlling the views that appear in the user interface. One option is to create a responsibility for NREC that includes only the relevant views for NREC's solution. Another option is to use Siebel Personalization to show or hide views based on user profile attributes. However, because these views are not part of NREC's design, NREC has chosen to inactivate unnecessary views in the repository.

For more information about responsibilities, read *Siebel Applications Administration Guide*. For more information about controlling views with Personalization, read *Siebel Personalization Administration Guide*.

The following procedure uses the Accounts Screen (SCW) as an example. To complete the NREC example, you need to inactivate the unnecessary views for the screens listed in Table 9 on page 76.

To inactivate screen views associated with a screen

1. In the Object Explorer, expand the Screen object type.

   The Screens window appears in the Object List Editor.

2. In the Screens window, select the Accounts Screen (SCW).
3. Lock the project locally by choosing Tools > Lock Project.
4. Change the Project property from Opportunity (SCW) to NREC Configuration.

5 In the Object List Editor, select the Screen View object type (child of Screen).

The Screen View window appears in the Object List Editor.

6 Select the Inactive field for the views that are not part of NREC's design.

To determine which views to inactivate, see Table 9 on page 76.

NOTE: To move the Inactive field next to the Name field, right-click in Screen View window, and then choose Columns Displayed.

7 Repeat steps 2 through 7 for each active screen in the NREC example.

8 Compile and Unit Test.

Read "Compiling Projects" on page 48, and "Testing Changes" on page 50.

Configuring NREC's Activity Applets

To configure NREC's applets you add and remove fields from the user interface, reposition controls, and modify text labels. After completing the procedures, you compile and Unit Test your changes.

About Applets and Applet Web Templates

Applets are rendered in the user interface by combining objects definitions stored in the Siebel repository with layout and formatting information contained in applet Web templates. Objects such as controls and list columns are mapped to placeholders in Web templates. At run time, the Siebel Web Engine creates these objects, places them in the appropriate spot in the Web template, retrieves the relevant data from the Siebel database, and uses the HTML contained in the Web template to display the applet in the user interface.

Modifying or creating applets is primarily done in Siebel Tools. Tasks include defining object properties in the Object List Editor and mapping controls to placeholders in Web templates using the Web Layout editor. However, you may also want to change style aspects of an applet, such as the color scheme, in which case you work with an external cascading style sheet (read Chapter 6, "Getting Started in the User Interface Layer"). Most of the time, you do not need to modify Web templates themselves.

For more information about Web templates, read *Configuring Siebel Business Applications*.

NREC's Business Requirements

Two applets appear on NREC's Activities view: Activity List Applet and the Activity Detail Applet. These applets need to be modified to meet NREC's business requirements. The business requirements and related tasks are listed below.

- To enhance usability NREC requires that unnecessary fields be removed from the user interface. Additionally they require that the fields appear in a particular order. You will remove fields such as Associated Cost, Campaign, and Fund Request, and reorder the remaining fields based on NREC's design. Read "Removing Fields (List Columns and Controls) from the User Interface" on page 80.
- Partner agents need to have the ability to track which opportunity is associated with each activity. This functionality is not available in the preconfigured application. Therefore, you must add the opportunity field to both applets. Read "Exposing Fields in the User Interface" on page 82.

These tasks are in the user interface layer of the Siebel Object Model. No work is required in the underlying business object layer or data object layer. The tasks in this section give you some hands-on practice configuring applets.

The design team prepared a view mock-up, shown in Figure 15, that shows what the view and applets will look like after they are complete.

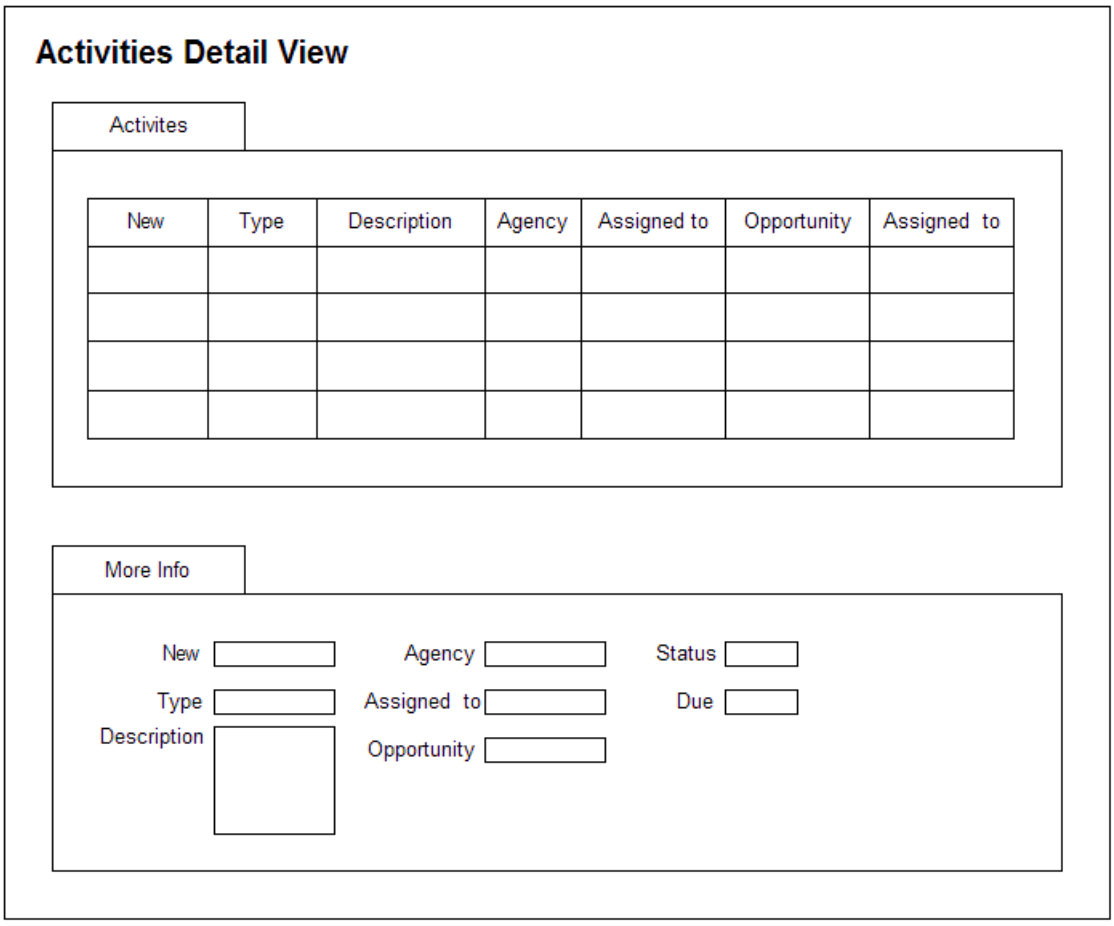


Figure 15. Activity Detail View Mock-up

Removing Fields (List Columns and Controls) from the User Interface

You can remove fields from the user interface using the Web Layout Editor. For each field (list columns or controls) that you do not want to be displayed, you delete it from the Web template. This unbinds the list column or control from the Web template. The objects remain intact as child objects of the applet, but they are not rendered in the user interface at run time.

To remove fields from the user interface

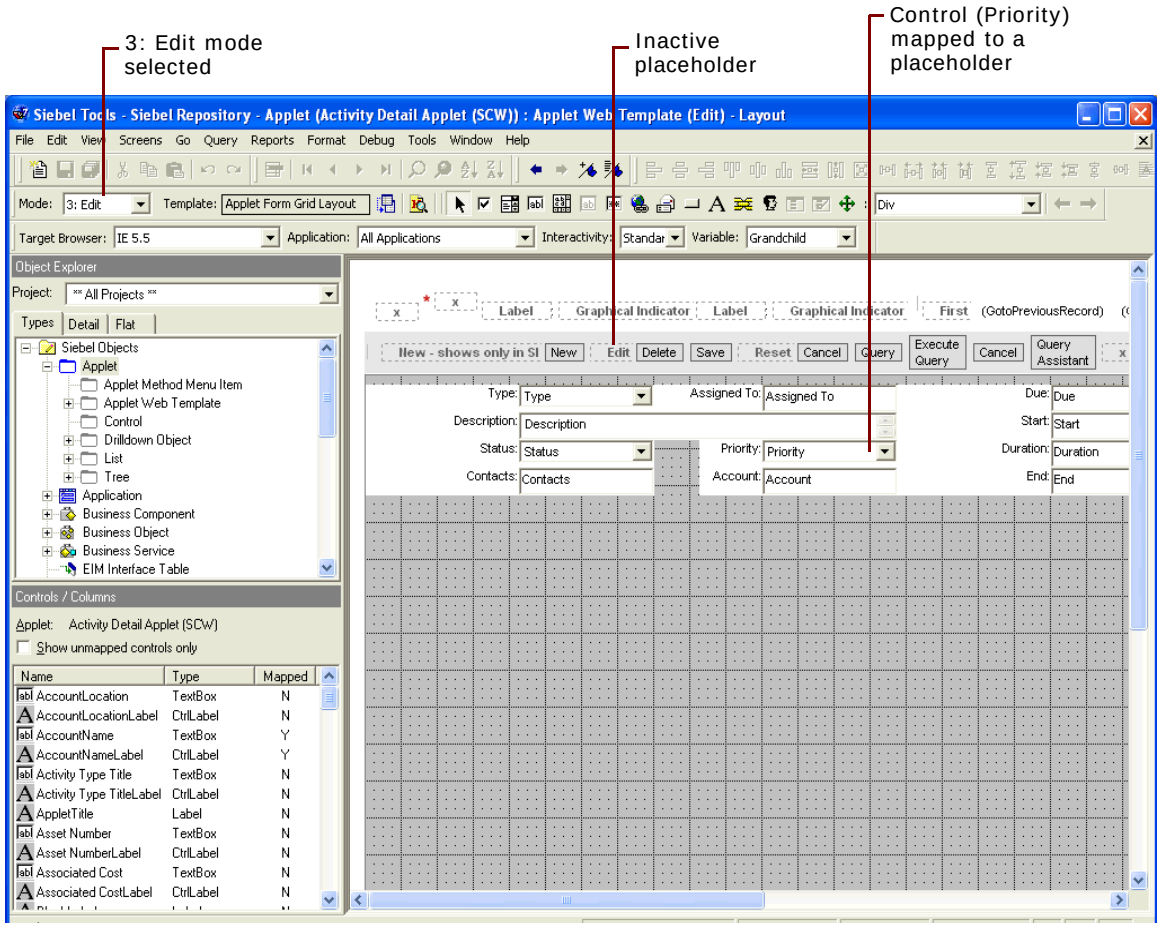
1. In the Object Explorer, select the Applet object type.
2. In the Object List Editor, select Activity Detail Applet (SCW).
3. Lock the project locally by choosing Tools > Lock Project.
4. Change the Project property from Activity (SCW) to NREC Configuration.

   NOTE: You need to have your target browser set before you can open the Web Layout Editor. Read "Setting Your Target Browser" on page 70. Make sure the Application drop-down menu is set to All Applications.

5. Right-click and then choose Edit Web Layout.

   The Web Layout Editor appears with the Base mode of the applet displayed.

6 Pull down the Mode menu (in the Toolbar) and choose 3: Edit as shown in the following illustration.



7 In the Web Layout Editor, delete the following fields: Account, Duration, and % Complete. Each field has two controls: a label (mapped to a symbolic string) and a field (mapped to a field in the database).

   a Select a control or use Control while clicking to select multiple controls.

   b Right-click and then choose Delete.

   c (Optional) Move the End field to the position formerly occupied by the Duration field. To determine where to place the field, see the mock-up in Figure 15 on page 79.

8 Choose File > Save.

9 To verify how the applet renders in the user interface, in the Web Layout Editor, right-click and then choose Preview. To turn off the preview, right-click and then choose Preview.

10 Choose File > Close.

Exposing Fields in the User Interface

NREC's partner agents must be able to associate an Opportunity record with an Activity record. However, the Activity List Applet and the Activity Detail Applet do not include the Opportunity field by default. Although the field does not appear by default, you can reconfigure the applets to display it. The Opportunity field is already defined on the underlying business component—the Action business component—so that what is required is to expose the field in the user interface.

The tasks for exposing the opportunity field in the user interface differ slightly for a list applet and form applet.

- List applets require that the field be defined as a list column object.
- Form applets require that the field be defined as a control object.

  NOTE: The Activity Detail Applet (SCW) is a form applet.

After list column or control objects are defined, the objects are mapped to placeholders in Web templates.

For detailed information about configuring applets, read *Configuring Siebel Business Applications*.

Objects that appear in the user interface, such as controls, fields, views, screens, and so on each have a display name or caption. The Caption field or Display Name field contains the text that appears in the user interface. You do not put text directly into these fields; instead you use the pick list in the appropriate String Reference field to select an existing symbolic string to use. Your selection in this pick list then populates the field.

The String Reference pick list can search for text. For example, you can search for the term "Opportunity" and get a listing of existing symbolic strings that start with that word. Such a search may produce many possible results, each one a slight variation on the base word. For example, you may get "Opportunity:", "OPPORTUNITY", "Opportunities", "Opportunity", and "Opportunity #".

Adding the Opportunity Field to the Activity List Applet (SCW)

On list applets, fields are defined as List Column object types. For the Activity List Applet (SCW) you need to define the Opportunity field as a List Column before mapping it to a Web template.

To define a list column object for a list applet

1. In the Object Explorer, expand the Applet object type.
2. In the Object List Editor, find and select the Activity List Applet (SCW).
3. If the project is not locked, do the following.
   a. Choose Tools > Lock Project.
   b. Change the Project property from Activity (SCW) to NREC Configuration.
4. In the Object Explorer, expand the List object type (a child object of Applet).
5. Expand the List object type, and then select the List Column object type (a child object of List).

   The List Columns window appears in the Object List Editor.

6 In the List Columns window, add a new record using the properties shown in the following table.

Property	Value	Description
Name	Opportunity	This is a name of the list column record.
Display Name - String Reference	SBL_OPPORTUNITY-1004224537-11Q	Use the pick list and search for an Opportunity symbolic string. There are several symbolic strings that start with "Opportunity". Select one that has an initial capital letter and no colon after it. Note that after you pick this symbolic string, the Display Name field shows the selected string.
Field	Opportunity	The values available from the drop-down list are the fields defined on the parent business component of the Applet (in this case, the Action business component).
Runtime	TRUE	When the property is set to TRUE, the application makes a run-time check to determine if a pick list, calculator, calendar, or MVG pop-up button must be provided.

Adding the Opportunity Field to a Activity Detail Applet (SCW)

For form applets, fields are defined as Control object types. For the Activity Detail Applet (SCW) you define the Opportunity field as a control before mapping it to a Web template.

To add a control object to a form applet

1 In the Object Explorer, expand the Applet object type.

2 In the Object List Editor, find and select the Activity Detail Applet (SCW).

3 In the Object Explorer, select the Control object type (child of Applet).

4 In the Controls window of the Object List Editor, add a new record with the properties shown in the following table.

Property	Value	Description
Name	Opportunity	Unique name of the control object.
Caption - String Reference	SBL_OPPORTUNITY-1004224537-11Q	Use the pick list and search for an Opportunity symbolic string. There are several symbolic strings that start with "Opportunity". Select one that has an initial capital letter and no colon after it. Select the same symbolic string you selected in Step 6 on page 83, so that the string on the form applet matches the one used on the list applet. Note that after you pick this symbolic string, the Caption field shows the selected string.
Field	Opportunity	Field defined on the parent business component. In this case, the parent business component is Action.
HTML Type	Field	HTML Control Type.

Mapping List Columns or Controls to Web Templates

After you have defined the list column and control object types for the applets, you are ready to map them to placeholders in a Web template. You do this using the Web Layout Editor. The process for mapping list columns or controls to Web templates is the same for both list applets and form applets.

NOTE: It is also possible to create controls while working in the Web Layout Editor. To do this you drag a control type from the toolbar, drop it onto the template, and then define the necessary properties. This method is an alternative to defining the controls in the Object List Editor as described in the previous section.

The following procedure uses the Opportunities list column on Activity List Applet (SCW) as an example. Note that NREC's design requires that you follow the same procedure for mapping the Opportunity control to the Web templates for the Activity Detail Applet (SCW) as well.

To map fields (controls and list columns) from an applet to a Web template

1 In the Object Explorer, select the Applet object type.

2 In the Object List Editor, select the Activity List Applet (SCW).

3 Right-click and choose Edit Web Layout.

The Web Layout Editor appears with the applet displayed in Base mode.

4 Change the mode to 3: Edit List.

5 Make room for the Opportunity field by rearranging the controls that are already mapped to the Web template.

You can move controls by dragging and dropping them or by using the Shift to Previous Placeholder and Shift to Next Placeholder buttons on the toolbar.

6 From the Controls/Columns window, drag the Opportunity list column and drop it into an empty placeholder in the Web template, as shown in the following illustration.

NOTE: If the Controls/Columns window is not open, choose View > Windows > Controls.

7 To verify how the applet renders in the user interface, right-click and then choose Preview.

NOTE: You can also export the preview to an HTML file by choosing File > Export, and then choosing a file name and location in the Save As dialog.

8 Make adjustments as necessary.

9 Choose File > Save.

10 Repeat the procedure for the Activity Detail Applet (SCW).

NOTE: On a form applet you need to add the label (OpportunityLabel) and the text box (Opportunity). You may need to resize the label to fit.

Reviewing the Results

In the previous sections you removed the unnecessary fields from applets, added a new field, and changed the display name and caption of a field. After you compile your changes and perform a Unit Test, the applets appear as shown in Figure 16.

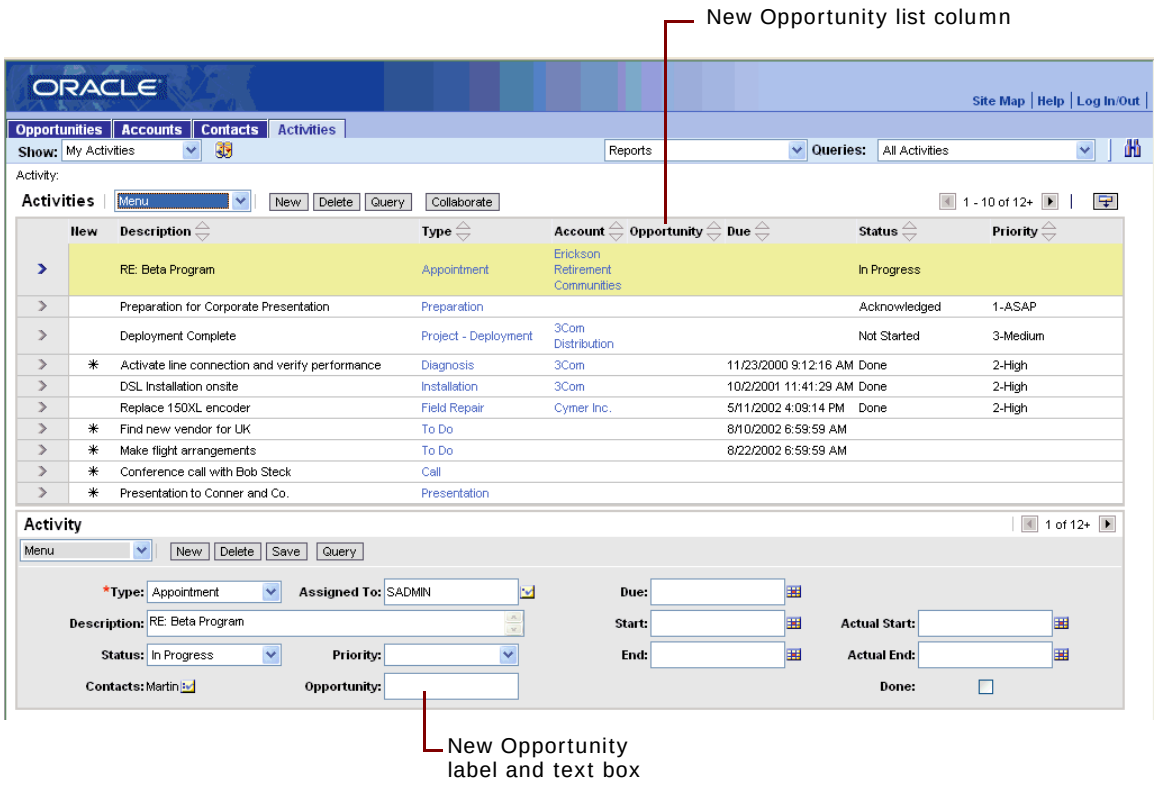


Figure 16. Activity Applets

7 Configuring the House and Opportunity Entities

In this chapter you will work in three layers of the object model—data objects layer, business objects, and user interface object—with the goal of constructing four views based on NREC's design specifications. This involves both modifying existing object definitions and creating new ones, and includes the following tasks.

- Creating and modifying views
- Creating and modifying applets
- Creating and modifying business components
- Modifying business objects
- Adding columns to an existing base table
- Using 1:M extension tables

The tasks in each section start at the database layer then progress up to the user interface layer. Working through the material this way gives you a good understanding of the object definition sequence and give you plenty of practice with many of the common configuration tasks.

Tasks are covered in detail where they first appear. If a task also occurs later in the chapter, it is presented at a higher level with a cross reference to the detailed procedure.

You can follow the tasks in this chapter like a tutorial, configuring against the sample database.

Configuring the House Detail View

NREC's House Detail view needs to be able to display the current list of houses for sale and the details for each house, such as square feet and number of bedrooms. The product entity in the Siebel data model meets NREC's needs reasonably well. However, attributes such as square feet and number of bedrooms are not part of the Siebel data model.

You can extend the standard data model to meet your unique business requirements. For example, assume NREC's design team has decided to extend the internal products base table by adding additional columns. These columns will be used to store the following additional attributes for each house.

- Address
- City
- Number of Bathrooms
- Number of Bedrooms
- Price
- Square Feet

- State
- ZIP Code

NOTE: Although List Price exists as standard field on the Internal Product business component, assume that NREC's design team has decided to add a column to S_PROD_INT to keep the house attributes stored in one table.

For detailed information about extending the data model, read *Configuring Siebel Business Applications*.

In this section you will add columns to the internal products table, and then modify objects in the business object layer and the user interface layer to accommodate the columns changes. You will modify existing objects as well as create new objects where appropriate.

The tasks for accomplishing this are:

- "Extending the Database by Adding New Columns to the Base Table" on page 89
- "Configuring the Internal Product Business Component" on page 92
- "Modify Existing Product Applets to Display NREC Attributes" on page 93
- "Creating the House Detail View" on page 96
- "Creating the Houses Screen" on page 97
- "Compiling and Unit Testing" on page 100

Figure 17 shows a mock-up of NREC's House Detail View. This is a new list-form view using existing applets.

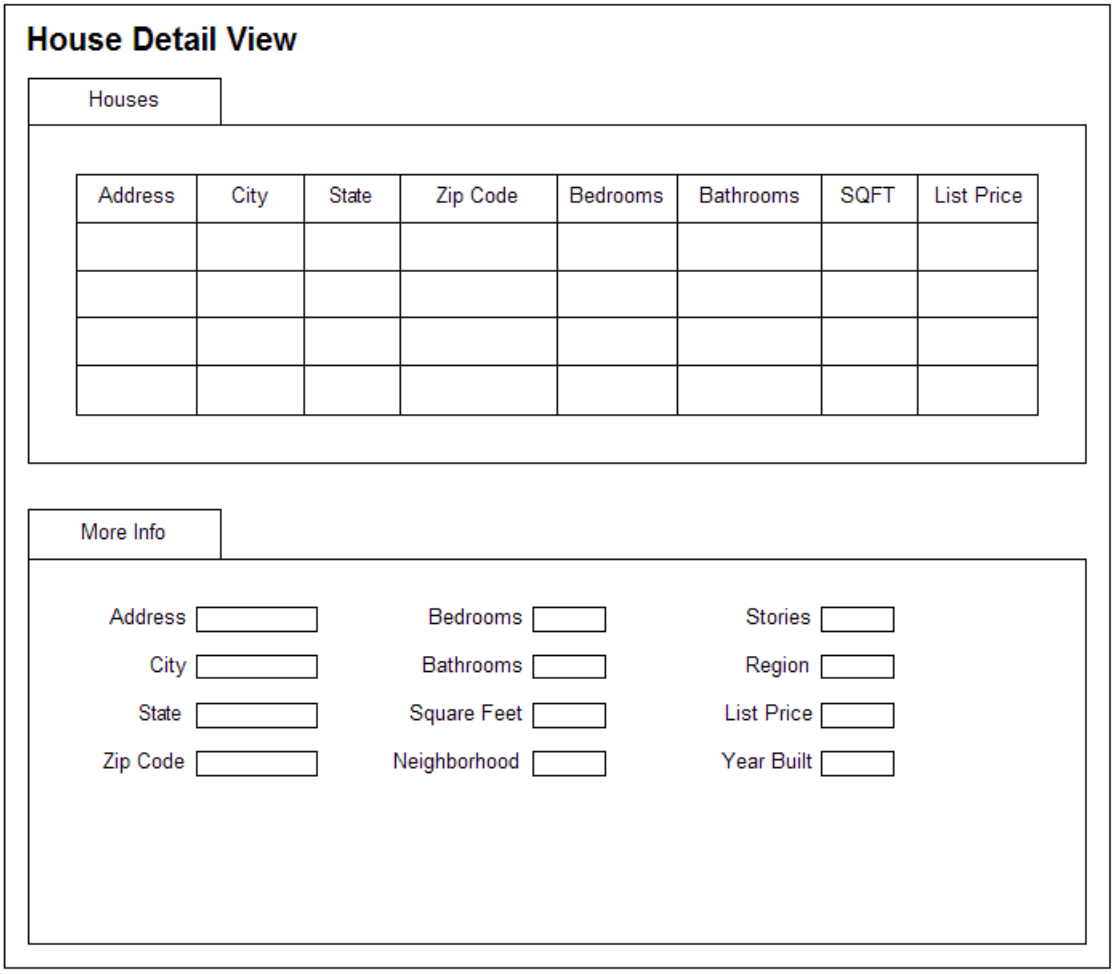


Figure 17. House Detail View Mock-Up

Extending the Database by Adding New Columns to the Base Table

You add columns to a base table using the Database Extension Designer. This involves adding columns to the base table using the Object List Editor in Siebel Tools and then applying the changes your local database schema.

Adding New Columns to the Products Table

First you must add the columns to the base table. You do this by creating additional Column object definitions for the base table object. For example, you are adding columns to the products table, S_PROD_INT.

To add a new column to a base table

1. In the Object Explorer, expand the Table object type.
2. In the Tables window, select S_PROD_INT.
3. Lock the project locally by choosing Tools > Lock Project.

   NOTE: The project field is read-only for tables. You cannot change the project from Newtable to NREC Configuration.

   Notice that the table is of type Data (Public). Only public tables can be extended.
4. In the Object Explorer, select Column object type (child of table).

   The names of the columns for the S_PROD_INT table appear in the Object List Editor.
5. In the Columns window, add the records shown in the following table.

Name	Physical Type	Length
X_ADDRESS	Varchar	30
X_BATHROOMS	Varchar	30
X_BEDROOMS	Varchar	30
X_CITY	Varchar	30
X_PRICE	Varchar	30
X_STATE	Varchar	30
X_SQFT	Varchar	30
X_ZIP_CODE	Varchar	30

   NOTE: Names of extension columns begin with X_ (for example, X_ADDRESS). The User Names of extension columns end with Ext (for example, X_ADDRESS Ext). Siebel Tools automatically enforces these conventions.

   The logical database schema is changed based on the information you entered, but you still need to physically apply the changes to your local database.

Applying Schema Changes to Your Local Database

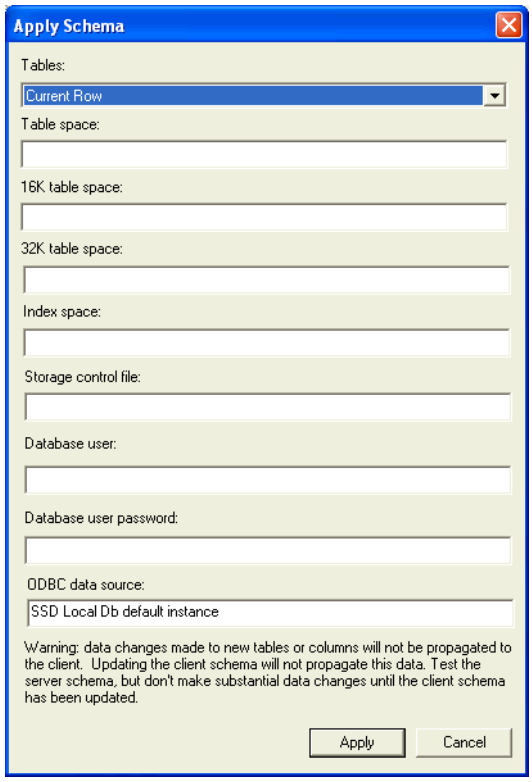
After you have added columns to the base table as described in the previous section, you have to physically apply the schema changes to your local database.

To apply schema changes to the local database

1. In the Object Explorer, select the Table object type.
2. In the Tables window, select S_PROD_INT.
3. Click Apply.

   A warning appears indicating that you are connected to a local database and asking if you want to continue.
4. Click OK.

   The Apply Schema dialog box appears, as shown in the following illustration.



5. In the Tables pick list, choose Current Row.

   This updates the database to reflect the schema change to the current selected row only (S_PROD_INT).

   Other options are:

   - **All.** Update the database to reflect the changes made to the dictionary. This option forces each database object to be compared with the data dictionary, and updated if required.
   - **Current Query option.** Update the database to reflect modifications made to the tables in the current query only.

6 Enter the Database user password.

NOTE: The default password is SIEBEL.

Do not specify a table space or index space.

7 Verify that the ODBC (Open DataBase Connectivity) connection specified in the ODBC Data Source text box is correct for your environment.

NOTE: You cannot apply schema changes to any database other than the one to which you are currently connected. If you are connected to the Sample database use “SIEBEL” as the Database user and the Database user password, and “SEAW Samp Db default instance” as the ODBC data source.

8 Click Apply to apply the new columns to the physical schema on your local database.

After this process has been completed, the columns you added to the logical schema as described in “Extending the Database by Adding New Columns to the Base Table” on page 89 now physically exist on your local database and are available to use in your configuration.

NOTE: Typically, after you have tested changes in the local database environment, you need to apply these changes to the server database. Checking in a project copies configuration changes to the server, but this process does not apply physical database extension changes. Therefore, you need to go through a separate process to apply database extension changes to the server database. For information on this process, read *Configuring Siebel Business Applications*.

Configuring the Internal Product Business Component

After you have applied the extension columns to your local database schema, you can add the fields to the business component that will use the columns to store data. In the NREC example, you add fields to the Internal Product business component. Additionally, you need to modify specific properties of the business component so users can update records.

The following procedure uses the square feet attribute as an example. To complete the NREC example, you need to add fields for the attributes listed at the beginning of this section as described in “Configuring the House Detail View” on page 87.

To add fields to the Internal Products business component

1 In the Object Explorer, expand the Business Component object type.

2 In the Business Components window, select the Internal Product business component.

3 Lock the project locally by choosing Tools > Lock Project.

4 Change the value of the Project property from Product to NREC Configuration.

5 In the Object Explorer, select the Field object type (child of Business Component).

6 In the Fields window, add records for the fields shown in the following table.

Name	Column
Address	X_ADDRESS
Bathrooms	X_BATHROOMS
Bedrooms	X_BEDROOMS
City	X_CITY
Price	X_PRICE
Square Feet	X_SQFT
State	X_STATE
Zip Code	X_ZIP_CODE

To change the default properties of the Internal Products business component

1 Select the Internal Product business component in the business component window.

2 Choose View > Windows > Properties Window.

3 Change the values of the following properties from the default value of TRUE to FALSE.

Field	Value
No Delete	FALSE
No Insert	FALSE
No Merge	FALSE
No Update	FALSE

Changing the values of these properties enables users to enter records using applets based on this business component.

Modify Existing Product Applets to Display NREC Attributes

After the fields are added to the business component, you can display them in the user interface. Assume that NREC's design team has determined that configuring the following two applets is easier than creating new ones.

- Product List Applet
- Product Form Applet

These applets are already based on the Internal Product business component and meet NREC's needs reasonably well.

The following procedures cover exposing fields in the user interface at a more abstract level than was covered in the previous chapter. For detailed procedures, read "Exposing Fields in the User Interface" on page 82.

The following procedure covers configuring the Product List Applet.

To expose fields in the Product List Applet

1 Navigate to the Product List Applet.

2 Lock the project locally by choosing Tools >Lock Project, and then change the value of the Project property from Product (SSE) to NREC Configuration.

3 Add List Column objects for each of the new fields shown in the following table. Remember that to set the Display Name field you must select a symbolic string using the pick list in the Display Name - String Reference field.

Note that the object hierarchy is Applet > List >List Column.

Name	Display Name	Field	Show In List
Address	Street Address	Address	FALSE
Bathrooms	Bathrooms	Bathrooms	TRUE
Bedrooms	Bedrooms	Bedrooms	TRUE
City	City	City	FALSE
Price	Price	Price	TRUE
Square Feet	Square Feet	Square Feet	TRUE
State	State	State	FALSE
ZIP Code	ZIP Code	ZIP Code	FALSE

NOTE: The Show In List property determines whether the field is displayed by default in the list applet. Fields with a Show In List property set to FALSE appear gray in the Web Template layout editor. These fields are hidden by default in the user interface. However, users can add them to the display by choosing Columns Displayed from the applet-level menu.

4 Change the display name of existing list columns.

- Change the display name of the Part # list column to House ID.
- Change the display name of the Name list column to Summary.

5 Select the Product List Applet record, right-click it, and choose Edit Web Layout.

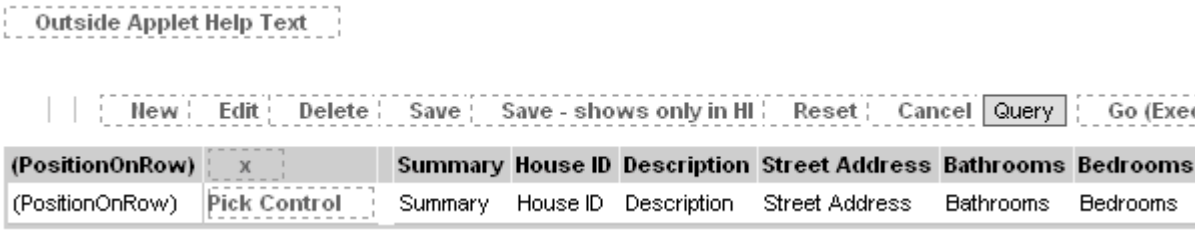
6 In the Web Layout Editor, delete the following unnecessary columns for the 1: Base mode of the applet.

- Class
- Compensable
- Customizable

- Description
- Effective End Date
- Effective Start Date
- Equivalent Product
- Lead Time
- Orderable
- Organization
- Product Line
- Revision
- Serialized
- Service Product
- Unit of Measure

7 Add the fields that you defined in Step 3 on page 94, including the fields to be hidden by default.

The result resembles the following illustration.



8 Choose File > Save.

9 Repeat Step 6 on page 94 through Step 8 on page 95 for the 2: Edit mode.

NOTE: You can switch applet modes using the Mode drop-down list on the Configuration Context Toolbar.

10 Switch to the 3: Query mode of the applet.

11 Delete the following unnecessary columns.

- Class
- Vendor

12 Add the fields that you defined in Step 3 on page 94, including the fields to be hidden by default. Drag the column names from the Controls/Columns window to a Field placeholder in the Web Layout Editor.

13 Choose File > Save.

14 Choose File > Close.

Configuring the Product Form Applet

The following procedure covers configuring the Product Form Applet.

To configure the Product Form Applet

1. Navigate to the Product Form Applet.
2. Lock the project locally by choosing Tools >Lock Project, and then change the value of the Project property from Product (SSE) to NREC Configuration.
3. Add Control objects (child of Applet) for each of the new fields as shown in the following table. Remember that to set the Caption field you must select a symbolic string using the pick list in the Caption - String Reference field.

Name	Field	Caption
Address	Address	Street Address
Bathrooms	Bathrooms	Bathrooms
Bedrooms	Bedrooms	Bedrooms
City	City	City
Price	Price	Price
Square Feet	Square Feet	Square Feet
State	State	State
ZIP Code	ZIP Code	ZIP Code

4. Modify the caption property of the existing controls.
   - Change the caption of the Part # control to House ID.
   - Change the caption of the Name control to Summary.
5. Select the Product Form Applet record, and right-click and then choose Edit Web Layout.
6. In the Web Layout Editor delete the unnecessary controls and add the new controls.

   Use Figure 17 on page 89 to determine what controls to add and delete.
7. In the Web Layout Editor, map the EditRecord control to the appropriate placeholder.
8. Choose File > Save, and then choose File > Close.

Creating the House Detail View

NREC wants to create a new view to display the Product List and Product Form Applets. They want to have the Products List Applet on top and the Products Form Applet below. This type of view is the standard list-form view. Use the View Wizard to create the view using the applets modified in the previous sections.

To create a new view

1 Choose File > New Object.

The New Object Wizards appears.

2 Under the General tab, select View and then click OK.

The New View wizard opens.

3 Complete the wizard using the information from the following table.

Field	Value
Project	NREC Configuration
Name	NREC House Detail View
Title	House Detail View
Business Object	Internal Product
Web Template	View Detail (Parent with Pointer)
Selected Applets	Product List Applet Product Form Applet

The wizard creates the view and related objects.

4 To review the results of the wizard, select the View object and select the NREC House Detail View. Right-click it and choose Edit Web Layout.

5 In the Web Layout view, right-click and choose Preview.

6 When done looking at the preview, choose File > Close.

NOTE: When creating views with special visibility properties, such as Organization, Manager, or Sales Rep, set the Visibility Applet Type property of the View object, not the View Web Template Item object. The only exception to this rule is the Home Page. Set the Home Page visibility using the View Web Template Item object.

Creating the Houses Screen

After creating the view, you must add it to a screen so it appears in the user interface. However, in this case, NREC decided to create a new screen for houses. This screen groups the views related to the Houses entity. After creating the screen, you need to define a Screen View child object to associate the screen with the House Detail View.

To create the Houses screen

1 Create a symbolic string for the term “Houses”. For more information, read “To create a symbolic string” on page 72.

2 In the Object Explorer, expand the Screen object type.

The Screens window opens in the Object List Editor.

3 In the Screen window, right-click and choose New Record.

4 Enter a new record with the properties shown in the following table.

Field	Value
Name	NREC House Screen
Project	NREC Configuration
Viewbar Text	Houses

5 In the Object Explorer, select the Screen View object type (child of Screen).

The Screen View window opens in the Object List Editor. Screen Views associate views to a screen.

6 Before adding the NREC House Detail View to the Houses screen, add a category to group this screen view and future screen views in the Site Map. Add a new record with values shown in the following table.

Field	Value	Description
Type	Aggregate Category	This defines a category that appears in the Site Map under the Houses heading (to be defined in Step 4 on page 99). This screen view and future screen views will be grouped together within this category.
Category Name	NREC	Provides a name for the category. When you define screen views, you will select this category name as the screen view's Parent Category.
Viewbar Text	More Info	Text that appears in the Show drop-down menu. Set this using Viewbar Text - String Reference.

7 Now add the NREC House Detail View. In the Screen Views window, add a new record using the values shown in the following table.

Field	Value	Description
View	NREC House Detail View	Name of associated view.
Type	Aggregate View	
Parent Category	NREC	This puts this screen view in the NREC category on the Site Map.

Field	Value	Description
Viewbar Text	More Info	Text that appears in the Show drop-down menu. Set this using Viewbar Text - String Reference.
Sequence	20	Determines the sequence in which view tabs are displayed.

8 Preferably, the NREC House Detail View is the default view for the NREC category. Select the NREC category record and set the Category Default View property to NREC House Detail View.

9 Also make the NREC House Detail View the default view for the House screen. In the Object List Editor, select the Screen object type again.

10 In the Screen window, select the NREC House Screen record and set the Default View property to NREC House Detail View.

Defining Page Tab and Screen Menu Items

For screens to appear in the user interface, you must define Page Tab and Screen Menu Items. These objects are child objects of the Application object type. Page tabs appear as the tabs across the top of the application and Screen Menu Items appear in the Site Map.

To define Page Tabs and Screen Menu Items

1 In the Object Explorer, expand the Application object type.

2 In the Application window, select the Siebel eChannel application.

3 In the Object Explorer, select the Page Tab (child of Application) object type.

4 In the Page Tab window, right-click and enter a record with the properties shown in the following table.

Field	Value	Description
Screen	NREC House Screen	Screen object associated with the page tab.
Sequence	125	Determines the sequence in which the page tabs appear.
Text	Houses	Text that appears in the tab.

5 In the Object Explorer, select the Screen Menu Item object type (child of Application).

6 In the Screen Menu Item window, add a new record with the properties shown in the following table.

Field	Value	Description
Screen	NREC House Screen	Screen associate with the Screen Menu Item.

Field	Value	Description
Sequence	15	Determines the sequence that the item appears.
Text	Houses	Text that appears as the link on the Site Map.

Compiling and Unit Testing

Now you are ready to compile and Unit Test your changes. After compiling the changes to a repository file, but before testing the changes, you must perform the following tasks using a Siebel employee application, such as Siebel Call Center or Siebel Sales. You cannot do these tasks in Siebel Partner Portal because it does not have the administrative views exposed.

- Register the House Detail View in the application.
- Associate the House Detail View to a responsibility.

  NOTE: You cannot edit the responsibilities that are part of the seed data that ships with the product. You must create a new responsibility to be able to associate new views to it. You can copy one of the sample responsibilities, such as Siebel Administrator, and then customize it for your purposes. You can create a new responsibility for testing, call it NREC Test, and then associate the SADMIN user to the responsibility. When you log in as SADMIN, you can access the new view.

You must complete these steps any time you create a new view. For detailed procedures, read "Testing Changes" on page 50.

Figure 18 shows what the applets and view look like after configuring them.

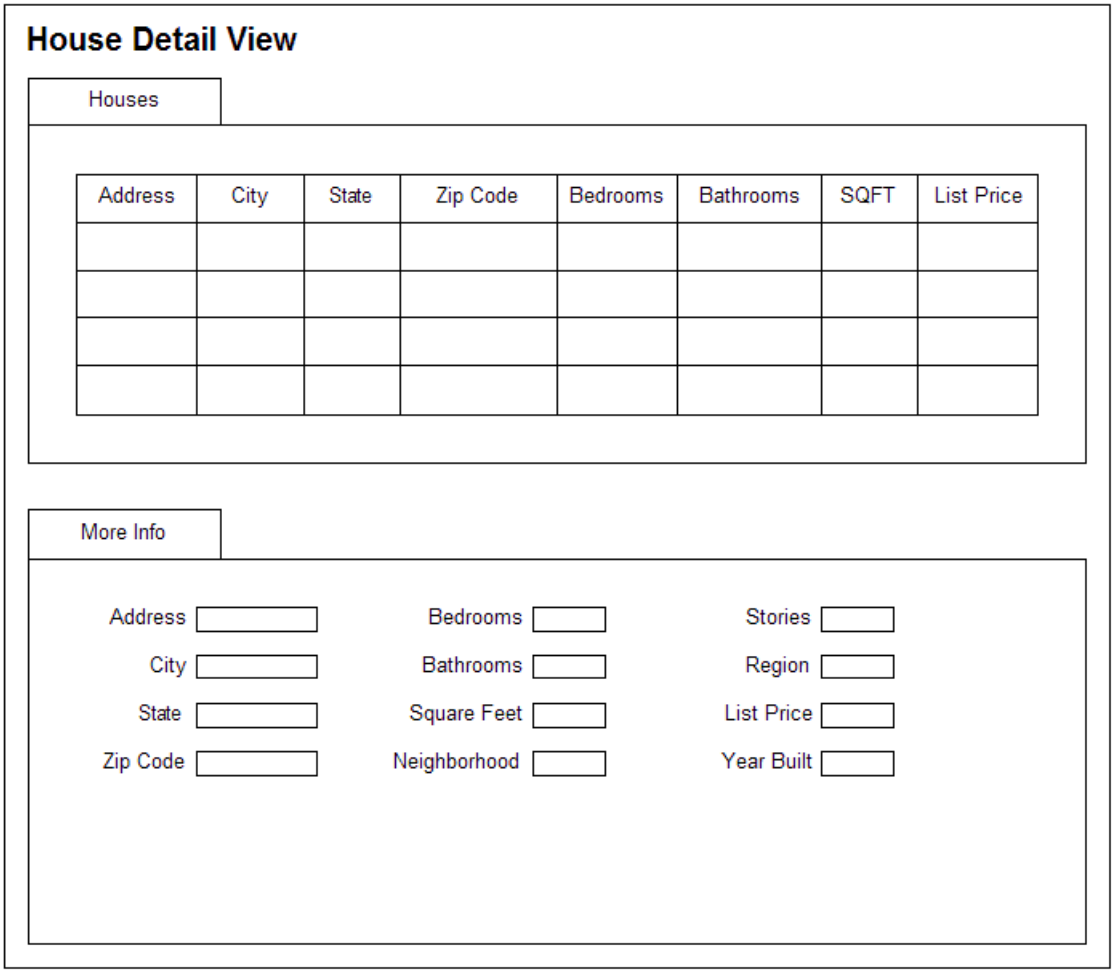


Figure 18. NREC House Detail

Configuring the House Detail - Appraisals View

The House Appraisals view is designed to provide a list of appraisal details (date, appraisal value, appraiser) for a particular house. This is a one-to-many relationship between the Products entity (Houses) and the Appraisals entity. Each house can be associated with many appraisal records. Review the ER diagram in "Data Layer" on page 28 for more information about NREC data entities and their relationships.

The standard Siebel data model does not include an appraisals entity. therefore, assume that the NREC design team has decided to create a new business component for appraisals, which will store its data in a standard one-to-many extension table. In the NREC example, you will use the 1:M extension table for S_PROD_INT, which is named S_PROD_INT_XM.

For more information about extension tables, read *Configuring Siebel Business Applications*.

The NREC design team has also determined that a master-detail view is required to display this data. The view displays a detailed house record in the top applet and a list of appraisals in the bottom applet. For the top applet, you can use the Product Form Applet that you configured as described in "Modify Existing Product Applets to Display NREC Attributes" on page 93. However, you need to create a new applet to display the list of Appraisals in the bottom applet. To see the end result of the configuration work, see Figure 19 on page 110.

Displaying data from a 1:M extension table requires the following steps.

About Standard 1:M Extension Tables

One-to-many extension tables are predefined tables that have one-to-many relationships with base tables. They have generic columns that you can use to store additional data. You can extend the data model for your purposes and track entities that are not part of the standard Siebel data model. Because the extension tables themselves are already part of the data model, you do not need to modify the database schema.

When using a one-to-many extension table to store data, you use the TYPE column to group records. You create a new business component for the entity to be tracked. The business component must have a Type field that defaults to a unique value and a search specification that finds only those records that contain this value. For details of how this is accomplished, read "Creating the Appraisals Business Component." With this practice you can use a single one-to-many extension table to store data for multiple business components. However, this practice must also be followed when the extension table is used by only one business component.

NOTE: There are more than 20 one-to-many extension tables in the standard data model. The names of one-to-many extension tables contain the suffix *_XM*.

Creating the Appraisals Business Component

To display data from a one-to-many extension table you must define a new business component with fields that map to the generic columns in the extension table (ATTRIB_01, ATTRIB_02, and so on), as well as three fields that provide the user key and map to the following columns.

- **PAR_ROW_ID.** This column maps to the foreign key field using in the one-to-many link.

- **NAME.** This column maps to the field being tracked in the business component. The value in the name needs to make the record unique for each parent record.
- **TYPE.** This column is used to group records in the extension table. Map this column to the Type field. You set a default value for the Type field and then configure the business component to search for those records in the extension table that contains this value.

NOTE: The combination of NAME, PAR_ROW_ID, and TYPE must be unique to satisfy the U1 index of the _XM table.

The following procedure guides you through creating the appraisals business component for the NREC example.

To create the Appraisals business component

1. Choose File > New Object.

   The New Object Wizards dialog box appears.

2. Under the General tab, select the BusComp icon and then click OK.

   The New Business Component wizard appears.

3. Enter the information from the following table, and then click Next.

Field	Value
Project	Product (SSE)
Name	Appraisals
Table	S_PROD_INT_XM

   The Single Value Fields dialog box appears.

4. Enter the columns and field names as shown in the following table, and then click Add for each one.

Column in Base Table	Field Name	Description
ATTRIB_01	Date	Date the appraisal is done.
ATTRIB_02	Appraisal Value	Appraised value of the property.
ATTRIB_03	Comments	Comments.
PAR_ROW_ID	Par Row Id	Parent Row ID.
NAME	Name	Used to store the name of the Appraiser.
TYPE	Type	Used to group records. Will be set to a unique predefault value.

5 When done adding all the fields, click Finish.

The Business Component Wizard creates the business component based on the information you entered.

Because the combination of NAME, PAR_ROW_ID, and TYPE must be unique to satisfy the U1 index of the _XM table, these fields must be marked as Required in the Appraisals business component.

To set the unique fields to be required

1 In the Object Explorer, expand the Business Component object type.

The Business Component window appears in the Object List Editor.

2 In the Business Component window, select the Appraisals business component.

3 In the Object Explorer, select the Field object type (child of Business Component).

The list of fields that you created in a previous procedure appear in the Fields window of the Object List Editor.

4 For each of the following fields, set the check mark on the Required property.

- Name
- Par Row Id
- Type

To set the search specification for the business component and the Predefault Value of the Type field

1 In the Object Explorer, expand the Business Component object type.

The Business Component window appears in the Object List Editor.

2 In the Business Component window, select the Appraisals business component.

3 In the Appraisals business component's Search Specification field, enter the following text:

```
[Type]= 'Appraisals'
```

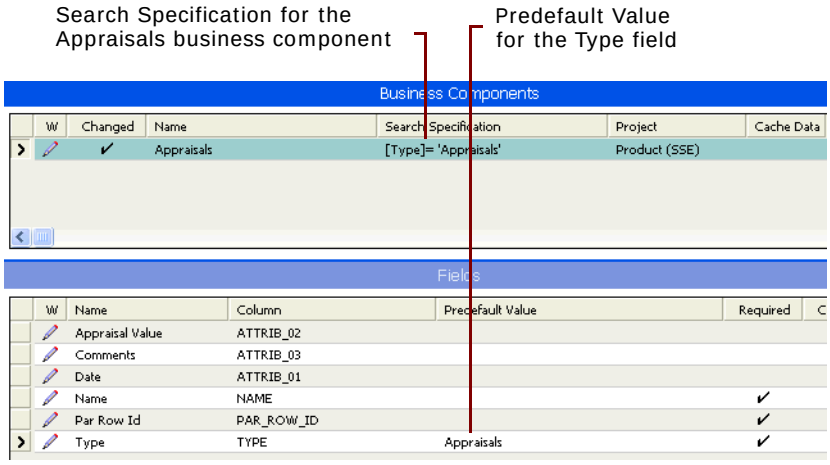
This sets the business component to retrieve only those records in which the Type field contains the value Appraisals.

4 In the Object Explorer, select the Field object type (child of Business Component).

The list of fields that you created in a previous procedure appear in the Fields window of the Object List Editor.

5 Select the Type field and then enter Appraisals as the Predefault Value.

The Type field always defaults to Appraisals, as shown in the following illustration.



Creating a Link Between Houses and Appraisals

The relationship between Houses and Appraisals is one-to-many—for each house record, there can be many Appraisal records. The link object establishes the one-to-many relationship between the parent business component (Internal Products) and the child business component (Appraisals). You must create a Link object to establish this relationship. When the relationship exists, you can create a master-detail view to display the data.

Before you can create the link, you must expose the Row_Id field in the Internal Products business component. This field is necessary to link with the Par Row Id field in the Appraisals business component.

To expose the Row_Id field in the Internal Product business component

1 In the Object Explorer, expand the Business Component object type.

The Business Component window appears in the Object List Editor.

2 In the Business Component window, select the Internal Product business component.

3 In the Object Explorer, select the Field object type (child of Business Component).

4 In the Fields window, right-click and choose New Record.

5 Set the Name property to Row_Id and in the Column property select ROW_ID.

To create a link between Internal Product and Appraisals

1 In the Object Explorer, select Link.

2 In the Links list applet, right-click and choose New Record.

3 Enter values for the properties as shown in the following table.

Field	Example Value	Comment
Name	Internal Product/ Appraisals	The convention is to use the name of the parent and child business component in the name of the link.
Project	NREC Configuration	Name of the project.
Parent Business Component	Internal Product	Name of the parent business component.
Child Business Component	Appraisals	Name of the child business component.
Source Field	Row_Id	Primary key value in the parent table.
Destination Field	Par Row ID	Foreign key value in the child table.

Adding the Appraisals Business Component to a Business Object

Business objects group related business components together. They also gather the Link objects that associate two business components with one another. Generally, business objects correspond to a screen in Siebel applications. For example, the Opportunity business object is analogous to the Opportunities screen.

Internal Products business object is the business object on which the Houses screen is based. You must associate the Appraisals business component that you create, as described in "Creating the Appraisals Business Component" on page 102, to the Internal Products business object.

To add the business component to a business object

1 In the Object Explorer, expand the Business Object object type.

2 In the Object List Editor, select the Internal Product business object.

3 In the Object Explorer, select the Business Object Component object type (child of business object).

The Business Object Component window appears in the Object List Editor.

4 In the Business Object Component window, enter a new record using the values shown in the following table.

Field	Value	Comments
BusComp	Appraisals	This is the business component you created as described in "Creating the Appraisals Business Component" on page 102.
Link	Internal Product/ Appraisals	The link defines the master-detail relationship between the parent and child business components.

Create a New Appraisals List Applet

Now that you have configured the business object layer, you are ready to expose the appraisal data in the user interface. You do this by creating a new applet. The type of applet called for in this case is a list applet. It displays a list of appraisal records for a single house record.

You can use the List Applet Wizard to create the applet. The wizard makes sure you define the correct properties and automatically maps the necessary list columns and controls to the Applet Web Templates.

To create the Appraisal list applet

1 Choose File > New Object.

The New Objects Wizard appears.

2 Under the Applets tab, select List Applet icon and then click OK.

3 In the General dialog box, enter the values shown in the following table, and then click Next.

Field	Value
Project	NREC Configuration
Applet Name	Appraisals List Applet
Display Title	Appraisals
Business Component	Appraisals

4 In the Web Layout - General dialog box, select the templates to use, as shown in the following table, and then click Next.

Field	Value
Template for Base read-only mode	Applet List (Base/EditList)
Template for Edit List mode	Applet List (Base/EditList)
Template for Edit mode	Applet List Edit (Edit/New/Query)

5 In the Web Layout - Fields dialog box, select the following fields and then click Next.

- Appraisal Value
- Date
- Name

6 In the second Web Layout - Fields dialog box, leave the controls selected by default and then click Next.

7 In the Finish dialog box, review the information entered and then click Finish.

The New Applet Wizard creates a new list applet based on the information you entered and displays the applet in Edit Web Layout Mode.

8 You can preview the applet by right-clicking in the Layout window and then choosing Preview.

9 Choose File > Close.

Creating the House Detail - Appraisals View

Next you need to create a new view to display the Product Form applet and the Appraisals List applet. Use the New View wizard to complete the steps.

To create the Appraisals view

1 Choose File > New Object.

The New Object Wizard appears.

2 Under the General tab, select the View icon and click OK.

The New View Wizard opens.

3 In the New View dialog box, enter the information shown in the following table, and then click Next.

Field	Value
Project	NREC Configuration
View Name	House Detail - Appraisals View
View Title	Appraisals
Business Object	Internal Product

4 In the View Web Layout - Select Template dialog box, select the View Detail (Parent with Pointer) Web template and then click Next.

5 In the Web Layout - Applets dialog box, select the following two applets and then click Next:

- Product Form Applet
- Appraisals List Applet

6 In the Finish dialog box, review the information you entered and then click Finish.

The New View wizard creates the applet and the necessary supporting objects and opens the applet in the Edit Web Layout mode.

7 You can preview the applet by right-clicking in the Layout window and then choosing Preview.

8 Choose File > Close.

Adding the House Detail - Appraisals View to the Houses Screen

After you have created the view, you need to associate it with a screen. In this case, NREC adds the House Detail - Appraisals View to the Houses Screen. For the detailed procedure, read "Creating the House Detail View" on page 96.

To associate the House Detail - Appraisals view to the Houses Screen

1 Create a symbolic string with the text "Appraisals". Use the procedure "To create a symbolic string" on page 72.

2 In the Object Explorer, expand the Screen object type.

The Screens window appears in the Object List Editor.

3 In the Screens window, select the NREC House Screen.

4 In the Object Explorer, select the Screen View object type (child of Screen).

5 In the Screen View window, add the record as shown in the following table.

Field	Value
View	House Detail - Appraisals View
Type	Detail View
Parent Category	NREC
Viewbar Text	Appraisals
Menu Text	Appraisals

Compiling and Unit Testing

After you have completed your configuration work, do the following.

- Compile your changes.
- Register the House Detail - Appraisals View in the application.
- Add House Detail - Appraisals View to a responsibility that you can use to test.
- Unit Test your work.

Figure 19 shows the House Detail - Appraisals View as it is rendered at run time.

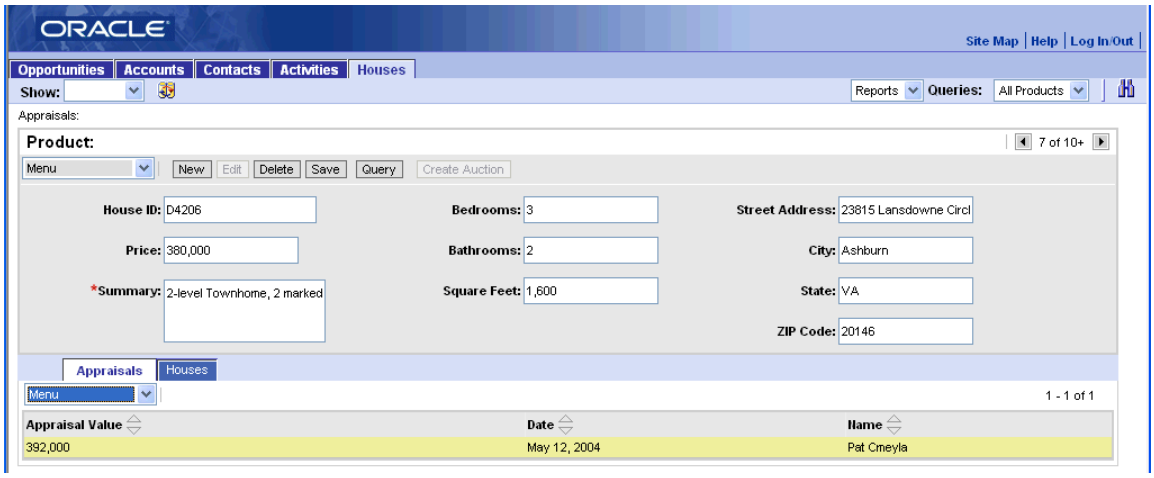


Figure 19. House Detail - Appraisals View

Configuring the Opportunity Details View

NREC uses Partner Portal's standard Opportunity Details view so partner agents can enter and display information about opportunities. However, many of the attributes that NREC needs to track for each opportunity are not part of the standard Siebel data model. For example, the standard data model does not include the number of bedrooms or bathrooms that a potential buyer may be interested in. The additional attributes are:

- Agent
- Number of Bathrooms
- Number of Bedrooms
- Price Range
- Square Feet

NREC follows an approach similar to what they did for the Houses entity—add columns to the base table and then modify the objects at the business logic layer and user interface layer accordingly.

For detailed information about extending the database, read *Configuring Siebel Business Applications*.

The high-level configuration tasks are:

- "Adding Additional Columns to the Opportunity Base Table" on page 111
- "Adding Fields to the Opportunity Business Component" on page 112
- "Modifying Applets to Display Additional Attributes" on page 113
- "Compiling and Unit Testing" on page 116

Figure 20 shows a mock-up design for the Opportunity Detail View. The Opportunity Detail view is the default view for the Opportunities screen. It is a list-form view. The Opportunity List Applet is the master applet. The Opportunity Form applet is the detail applet. The mock-up shows both the standard fields and the fields added for NREC.

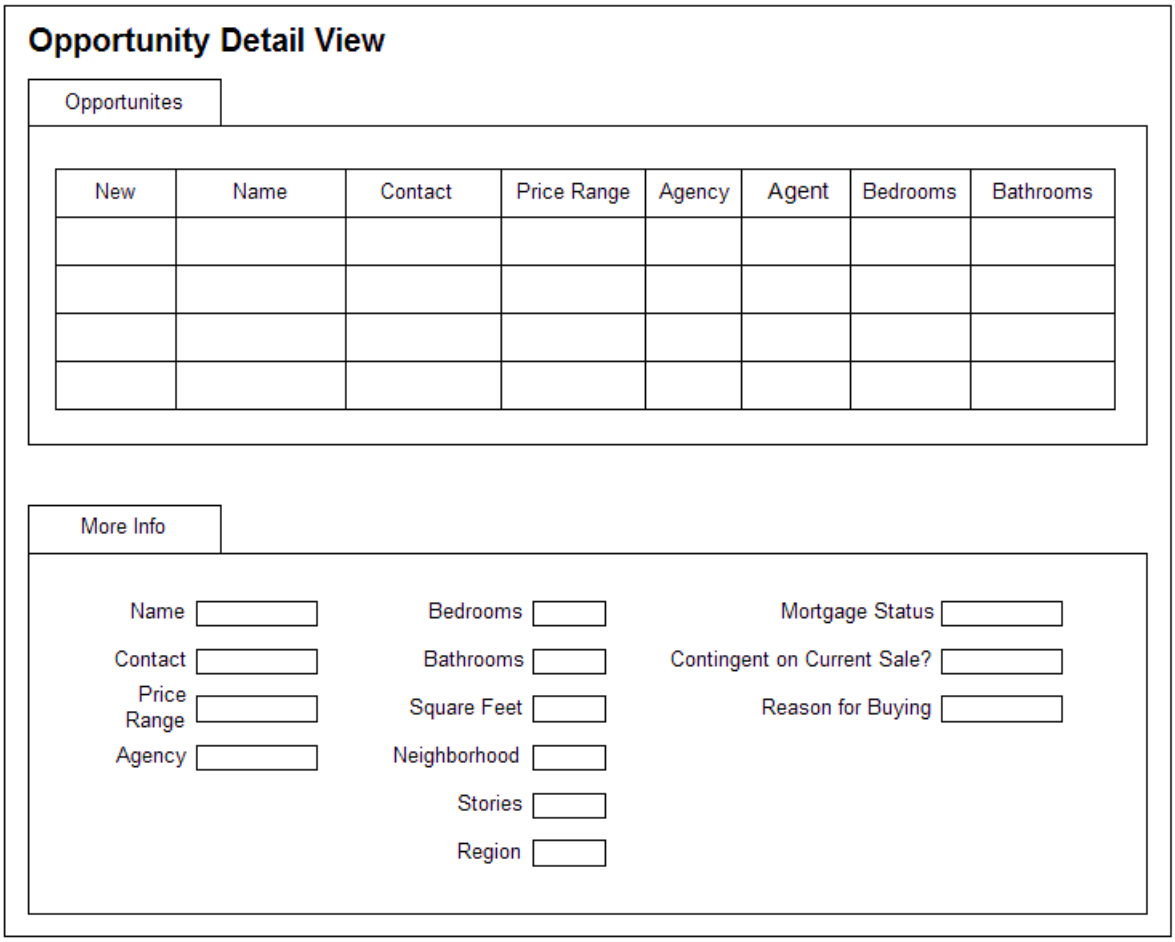


Figure 20. Opportunity Detail View Mock-Up

Adding Additional Columns to the Opportunity Base Table

The steps for adding columns to the base table are the same steps you followed to add additional columns to the internal products table. The procedure is covered here at a high-level. For detailed instructions, review the tasks covered in "Extending the Database by Adding New Columns to the Base Table" on page 89.

To add additional columns to the Opportunity base table

1 Navigate to the Opportunity base table S_OPTY.

2 Add the column records as shown in the following table.

Name	Type	Length
X_AGENT	Varchar	50
X_BATHROOMS	Varchar	20
X_BEDROOMS	Varchar	20
X_PRICE_RANGE	Varchar	20
X_SQUARE_FEET	Varchar	20

3 Apply the schema to the local database.

Adding Fields to the Opportunity Business Component

After extending the database to support NREC's additional attributes, you need to define the fields in the business object layer. You do this by adding the fields to the Opportunity business component. The fields map to the extension columns in the S_OPTY. The steps covered in this procedure are at a high-level. For detailed instructions, review the procedures covered in "Configuring the Internal Product Business Component" on page 92.

To add fields to the Opportunity business component

1 Navigate to the Opportunity business component.

2 Lock the project locally, and then change the value of the Project property to NREC Configuration.

3 Add the fields as shown on the following table.

Name	Column
Agent	X_AGENT
Number of Bathrooms	X_BATHROOMS
Number of Bedrooms	X_BEDROOMS
Price Range	X_PRICE_RANGE
Square Feet	X_SQUARE_FEET

These fields are now available to expose in the user interface.

Modifying Applets to Display Additional Attributes

Now that the fields are added to the business component, the next step is to expose them in the user interface. NREC's design is to modify the default applets that appear on the Opportunity Detail View to include the new fields. These two applets are:

- Opportunity List Applet (SCW)
- Opportunity Form Applet (SCW)

An alternative is to create new applets. However, modifying existing objects is typically a preferable method because it requires less configuration work and causes fewer problems with future upgrades.

The following procedures are the high-level steps required for modifying the existing Opportunity list and form applets to display new fields added to the Opportunity business component. As discussed in the previous chapter, the procedures differ slightly for form and list applets. Read "Exposing Fields in the User Interface" on page 82.

To expose the new fields on the Opportunity List Applet (SCW)

1. Create a symbolic string for the term "Price Range". Use the procedure "To create a symbolic string" on page 72.
2. Navigate to the Opportunity List Applet (SCW).
3. Add the List Columns as shown in the following table.

Name	Field	Display Name	Show in List
Agent	Agent	Agent	TRUE
Contact	Key Contact Id	Contact	TRUE
Number of Bathrooms	Number of Bathrooms	Bathrooms	TRUE
Number of Bedrooms	Number of Bedrooms	Bedrooms	TRUE
Price Range	Price Range	Price Range	TRUE
Square Feet	Square Feet	SQFT	TRUE

   The list columns are now available to map to applet Web templates. The Show in List property determines whether the field appears by default in the list applet. If the Show in List property is FALSE or null, users can add it to the list applet at run time by choosing Display Columns from the applet-level menu.

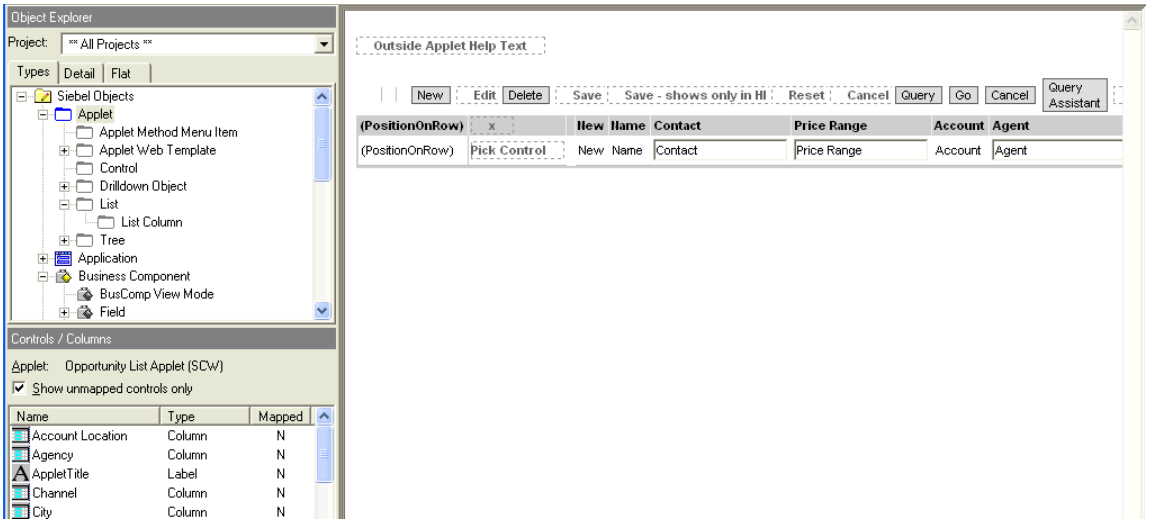
4. Reselect the Opportunity List Applet (SCW) applet.
5. Open the Web Layout Editor. Delete all unnecessary list columns that are not shown in Figure 20 on page 111. The columns to delete include:
   - Revenue
   - Close Date
   - Lead Quality

- Probability %
- Expected Value
- Status
- Reason
- Sales Method
- Created
- Description
- Site
- Address
- City
- State
- Zip Code
- Country
- Sales Team
- Organization
- Split Revenue

NOTE: To select multiple controls, press the Ctrl key while selecting.

6 Map the list columns defined in Step 3 to complete the layout as shown in Figure 20 on page 111.

When done, the Web layout resembles the following illustration.



7 Repeat Step 5 on page 113 and Step 6 for each applet mode.

To expose new fields on the Opportunity Form Applet (SCW)

1 Navigate to the Opportunity Form Applet (SCW).

2 Add the Controls as shown in the following table.

Name	Caption	Field
Agent	Agent	Agent
Contact	Contact	Key Contact Id
Number of Bathrooms	Bathrooms	Number of Bathrooms
Number of Bedrooms	Bedrooms	Number of Bedrooms
Price Range	Price Range	Price Range
Square Feet	SQFT	Square Feet

3 Reselect the Opportunity Form Applet (SCW) applet.

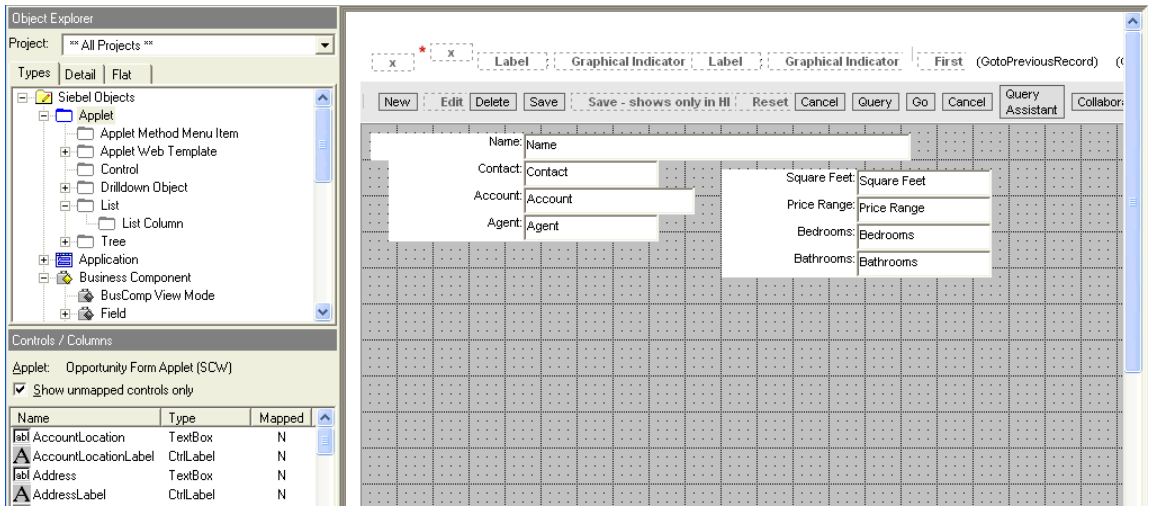
4 Open the Web Layout Editor and select an active mode. Delete all unnecessary controls that are not shown in Figure 20 on page 111. The controls to delete include:

- Description
- Site
- Probability %
- Close
- Organization
- Revenue
- Expected Value
- Committed
- Sales Team

NOTE: To select multiple controls, press the Ctrl key while selecting.

5 Map the controls you created in Step 2 on page 115 to complete the layout as shown in Figure 20 on page 111.

When done, the Web layout resembles the following illustration.



6 Repeat Step 4 on page 115 and Step 5 for each active applet mode.

Compiling and Unit Testing

After you have made your changes you are ready to compile and Unit Test your results. Figure 21 shows what the applets look like after being modified.

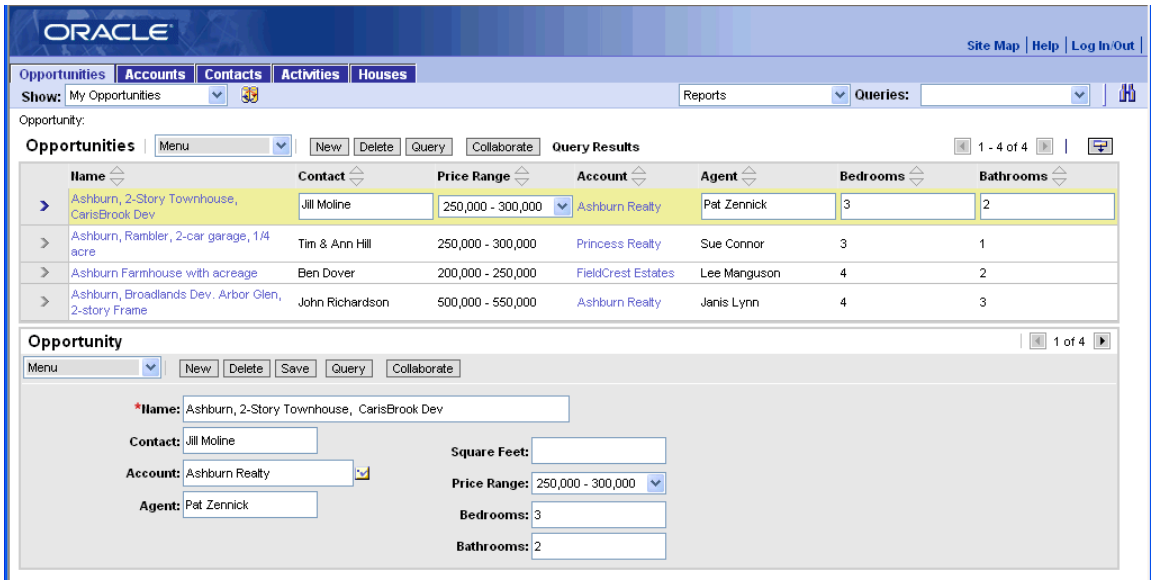


Figure 21. Modified Opportunity List and Form Applets

Testing the House and Opportunity Entities

NREC's testing plan calls for periodic integration and regression testing. These tests are necessary to make sure that the changes you have made up to this point do not prevent previously existing features from working, and to make sure that the application still can interact with other applications and products. NREC's test plan makes sure these checks happen at specific strategic intervals in the configuration process. In the event a test fails, NREC can quickly locate the faulty configuration change and make the appropriate modifications. If these tests are performed too infrequently, then troubleshooting errors may be a major effort.

8 Pick Lists, Drilldowns, and MVGs

This chapter takes a second pass at configuring. You will work with some new objects as well as some of the objects that you configured in previous chapters. You can complete the tasks in this chapter using your sample database.

Configuring Pick Lists

NREC wants to take advantage of pick lists to standardize data entry and minimize mistakes. Using pick lists you can populate a field by choosing values from a list rather than typing them in fields. There are two types of pick lists: static and dynamic. Static pick lists draw their data from the Siebel list of values table. The list of values table is maintained by an administrator. Dynamic pick lists draw their data from user-maintained tables.

Static Pick Lists

A static pick list is a list of predefined values that the user invokes from a field in an applet. When the user clicks the drop-down arrow to the right of the field, a single-column pick list appears. The user chooses a value from the list, and then clicks Save to enter the value for the field. The values in the pick list are predefined by an administrator or developer and stored in the list of values table. Figure 22 shows a pick list for the Price field.

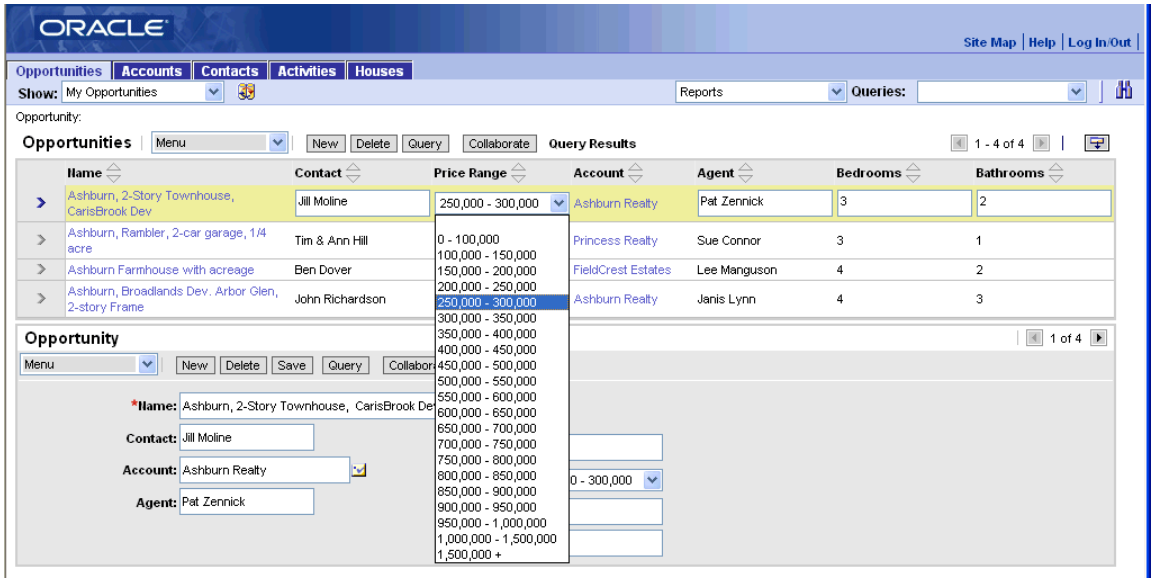


Figure 22. Static Pick List

A pick list can be bounded or unbounded. A bounded pick list restricts the user to choose values from the list only. With an unbounded pick list the user can choose a value from the list or type a entry directly into the field.

NREC's requirement is to have pick lists for the following fields on the Opportunity business component.

- Bathrooms
- Bedrooms
- Price Range
- Square Feet

The following procedure uses the Price Range field on the Opportunities business component as an example. To complete the NREC configuration, create pick lists for the remaining fields as well.

To create a new static pick list

1 Choose File > New Object.

The New Objects Wizard appears.

2 Under the General tab, select the Pick List icon, and then click OK.

The Pick List wizard opens.

3 In Pick List dialog box, enter the values shown in the following table, and then click Next.

Field	Value
Project	NREC Configuration
Business Component	Opportunity
Field	Price Range

The field is one you created in "Adding Additional Columns to the Opportunity Base Table" on page 111.

The Pick List Type dialog box appears, and it displays the controls that map to the field you selected in Step 3.

4 Click the Static option button and then click Next.

5 In the Pick List Definition dialog box, click the Create new Pick List option button and then click Next.

6 In the next Pick List Definition dialog box, enter Price Range as the name of the pick list, click the Create new List of Values option button, and then click Next.

7 In the List of Values dialog box, enter a name for the list and then, for each value to appear in the list, enter the value shown in the following table, and then click Enter.

Field	Value
Name	Price
Value	0 - 100,000
	100,000 - 150,000
	150,000 - 200,000
	200,000 - 250,000
	...and so on

8 In the next Pick List Definition dialog box, enter a comment in the Comment field and leave the Search Specification and Bounded Pick List check boxes blank.

A bounded pick list forces the user to enter a value that has been predefined on the list. Not selecting the Bounded check box allows the user to type in a value or choose one from the list.

9 In the Finish dialog box, review the information and then click Finish.

The wizard creates the Pick List object and related objects and adds the list of values to the database.

10 Compile and Unit Test.

The pick list appears as a drop-down list on the Price fields in the user interface. See Figure 22 on page 120. Notice that the pick list is available on the Price field in both the list and the form applet.

Dynamic Pick Lists

Like static pick lists, dynamic pick lists populate fields by presenting the user with a list of values. However, rather than drawing the values from the list of values table, a dynamic pick list draws its values from another user-maintained business component. Fields that use dynamic pick lists are typically joined fields displaying data from a table other than the business component's base table.

The NREC design includes a dynamic pick list on the Activity Detail Applet. The opportunity field that you exposed on the applet (read "Adding the Opportunity Field to a Activity Detail Applet (SCW)" on page 83) is a joined field. Creating a dynamic pick list on this field allows users to update the activity record by pulling in the opportunity name from the Opportunity table.

The Pick List Wizard helps you through the process of creating a dynamic pick list and related objects, which includes:

- **Pick List.** Object that defines the properties of the pick list, including the originating business component and the pick business component.

  NOTE: The originating business component is the one on which you are creating the dynamic pick list. In the current example, it is the Action business component. The pick business component is the one from which you are picking values to display to the user. In the current example, it is the Opportunity business component.

- **Pick Maps.** Child object of a business component field that map the source field in the pick business component with the target field in the originating business component.
- **Pick Applet.** Pop-up applet that displays the a list of records from which the user can choose.

Before creating a dynamic pick list you must lock the projects for both the pick business component and the originating business component. For the current example lock the Action business component locally and then change the value of the Project property to NREC Configuration. In previous chapters, you have already associated the Opportunity business component with the NREC Configuration project, which is locked.

To create a dynamic pick list

1 Select the Action business component and lock it. Change its Project property to NREC Configuration.

2 Choose File > New Object.

3 In the General Tab, select the Pick List icon, and then click OK.

4 In the Pick List dialog box, enter the values shown in the following table, and then click Next.

Field	Value	Description
Project	NREC Configuration	Project to which the pick list will belong.
Business Component	Action	Business component in the project that contains the field for which you are defining the pick list.
Field	Opportunity	Field for which you are defining the pick list.

5 In the Pick List Type dialog box, click the Dynamic option button and click Next.

6 In the Pick List Definition dialog box, click the Create New Pick List option button and click Next.

7 In the next Pick List Definition dialog box, enter the values shown in the following table, and then click Next.

Field	Value	Description
Pick Business Component (business component to base the pick list on)	Opportunity	This is the business component from which you are drawing values to display to the user.
Field to sort by	Created	Records are sorted by this field in the pick applet.
Name	Opportunity NREC	Name of the Pick List.
Search Specification	leave blank	This is optional.
Comment	Describe this pick list	Enter a comment that will help you identify this pick list in the future.

8 In the Pick List Specifications dialog box, accept the defaults; leave the check boxes clear.

9 In the Pick Map dialog box, use the drop-down lists to choose the fields shown in the following table, and then click Add.

Field in Originating Business Component	Field in Pick Business Component
Opportunity	Opportunity
	Oppty Id

This information is used to define the Pick Maps for the Pick List. They are the mappings between the source field and the target field.

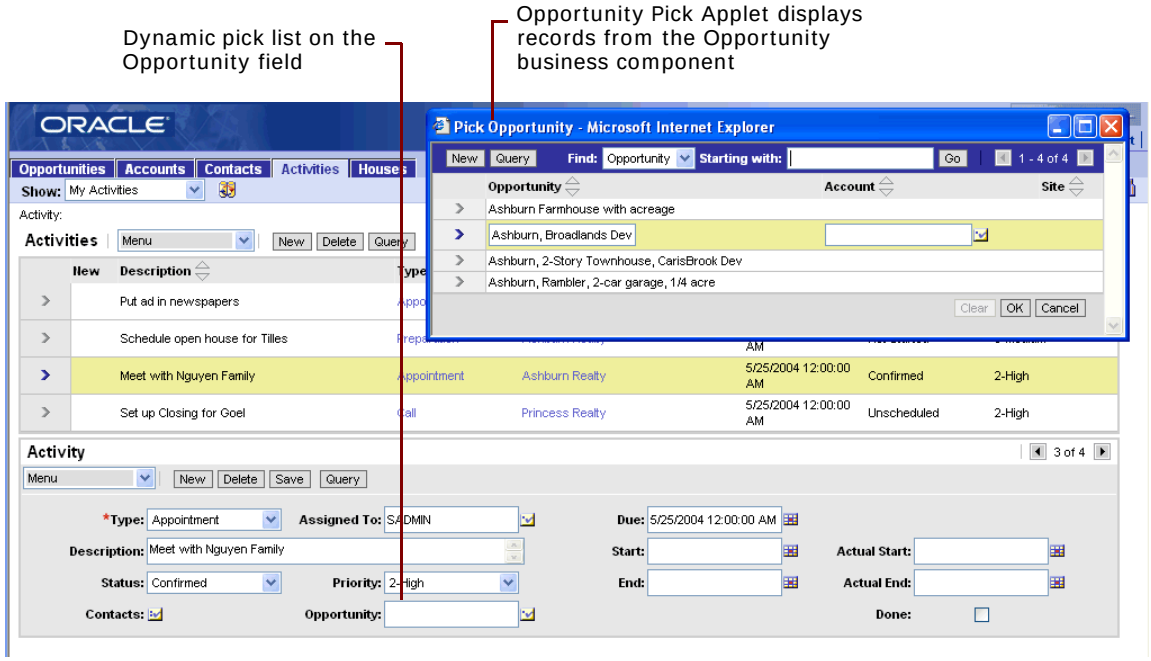
10 In the Finish dialog box, review the information you entered and then click Finish.

The Pick List wizard creates the pick list, pick maps, and determines if a pick applet already exists that can be used to display the records. For the NREC example, there is already an Opportunity pick applet defined, so the New Applet Wizard does not open. If an appropriate pick applet did not exist, then the New Applet Wizard opens.

11 Compile and Unit Test.

NOTE: Be sure to compile the Activity project.

The results resemble the following illustration.



Constraining a Pick List

You can filter a pick applet to display only records that have field values that match corresponding fields in the originating business component's records. For example, you can constrain the Opportunity pick applet, configured in the previous section, to display only the opportunities for the account associated with the activity.

To constrain a pick list you create another pick map with a Constrain property set to TRUE. This pick map defines fields on the originating business component and a field on the pick business component. These two fields must match for the record to be displayed in the pick applet. You can create more than one constrain pick map. Constrain pick maps do not copy the values of the field from the originating business component to the pick business component. They serve as a filter to determine which records appear in the pick applet.

Following the NREC example, you constrain the list of records displayed in the Opportunity Pick Applet based on the Account field.

To create a constrain pick map

1. In the Object Explorer, expand the Business Component object type.
2. In the Object List Editor, select the Action business component.

   The Action business component is the originating business component.

   NOTE: The Opportunity business component is the pick business component.
3. In the Object Explorer, expand the Field object type (child of Business Component).
4. In the Object List Editor, select the Opportunity field.

   The Opportunity Field is the field from which the pick applet is invoked.
5. In the Object Explorer, select the Pick Map object type (child of Field).
6. In the Object List Editor, enter the record as shown in the following table.

Field	Pick List Field	Constrain
Account Name	Account	TRUE

   The Account Name field on the originating business component must match the Account field on the pick list business component for the record to be displayed in the pick applet.
7. Compile and Unit Test.

   To test, you need to have some data entered for accounts, activities, and opportunities. Select an activity and then open the Opportunity pick list. The Pick Opportunity list shows only the opportunities that match the selected account.

Creating Drilldowns

Using drilldowns, users can click a field and display another view that has more information about the field. Fields with drilldowns appear as hyperlinks in the user interface. For example, when you click on the Name field in the Opportunity List Applet, you drill down to the Opportunity Detail - Contacts view as shown in Figure 23.

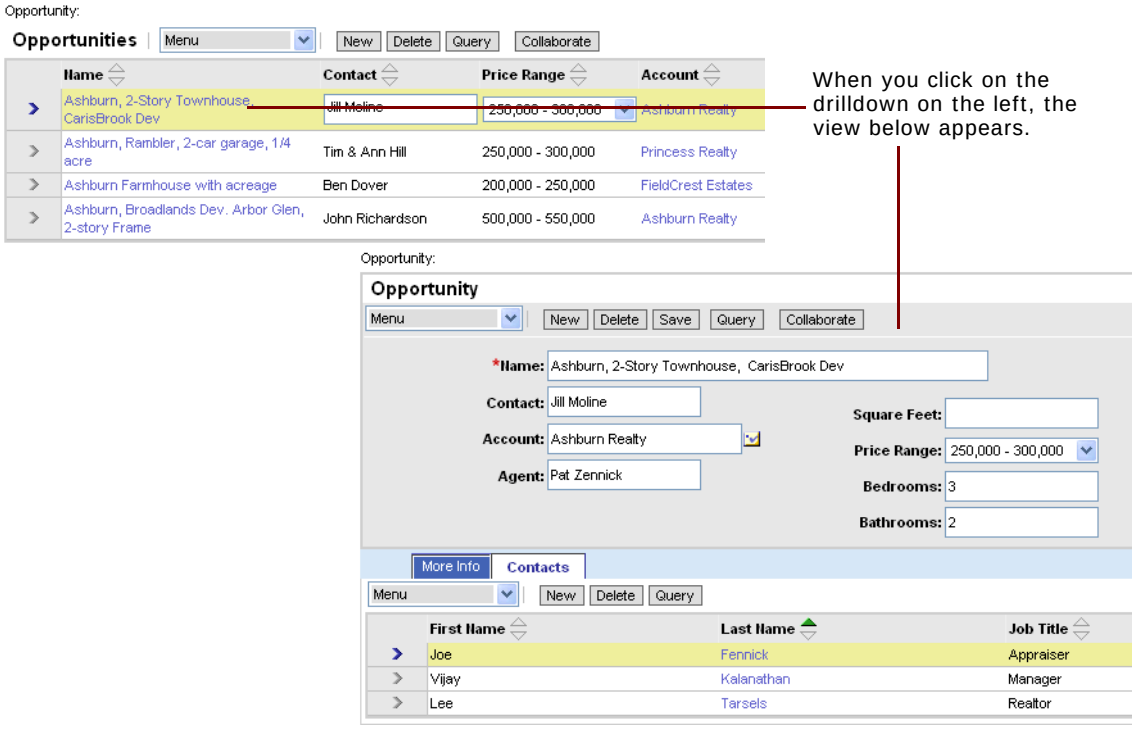


Figure 23. Drilldown Example

Drilldowns can be either static or dynamic. A static drilldown always takes the user to the same view. A dynamic drilldown can take the user different views depending on certain conditions, such as the value of a field.

You configure drilldowns in Siebel Tools. The Drilldown object type is a child object of the Applet object type. It defines the field on which the drilldown behavior is to be implemented and the view that appears when the user clicks the field. For example, the drilldown object definition for the Name field of the Opportunity List applet as shown in Figure 24.

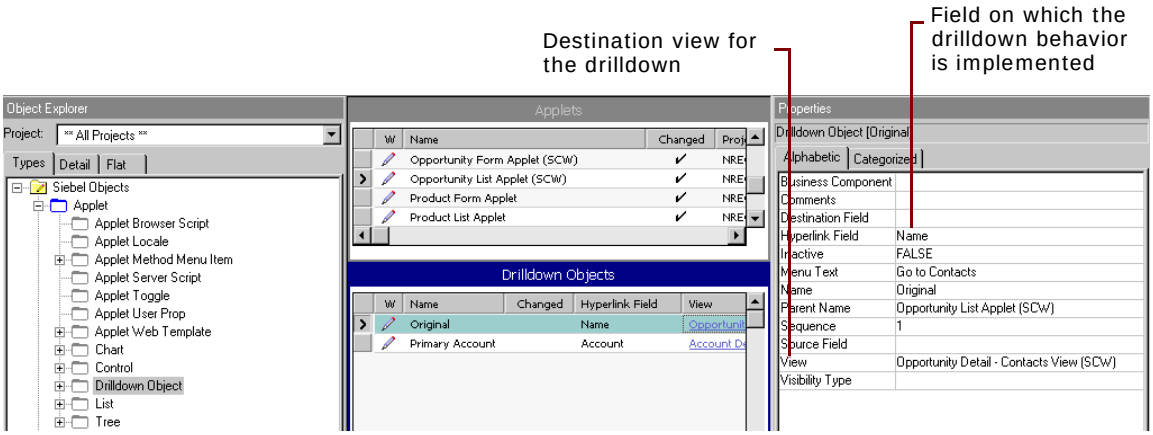


Figure 24. Static Drilldown

Creating a Dynamic Drilldown

A dynamic drilldown links to a different view depending upon certain conditions, such as the value of a field. For example, NREC wants to configure the drilldown shown in Figure 23 on page 126 so that it is dynamic. The requirement is the following:

- If the opportunity sales stage is Prospecting, Qualification, or Closing, go to the Opportunity Detail - Activities view.
- Otherwise, go to the Opportunity Detail - Contacts view. Note that this is the behavior of the existing drilldown object.

Dynamic drilldowns require one or more dynamic drilldown destination object types. Dynamic drilldown destination objects define the conditions that determine when to go to a particular drilldown object. They are child objects of a drilldown object.

For a dynamic drilldown, you define each candidate view by creating a drilldown object. You express the conditions under which each drilldown object is activated by defining one or more dynamic drilldown destinations. Dynamic drilldown destinations are defined as child objects of the default drilldown object. When the conditions expressed in the dynamic drilldown destinations are false, the parent drilldown acts as the default.

For the current example, you need to do the following.

- Define a drilldown object for the Opportunity Detail - Activities View (SCW).

- Define destination drilldown objects as children of the *Original* drilldown object.

  NOTE: Original is the existing drill drilldown object. It displays the Opportunity Detail - Contact view. This drilldown serves as the default when the conditions expressed in child drilldown destination objects are false.

To create a dynamic drilldown object

1. In the Object Explorer, expand the Applet object type.
2. In the Object List Editor, select the Opportunity List Applet (SCW).

   This is the applet from which the user can drill down.
3. In the Object Explorer, select the Drilldown Object object type.
4. In the Drilldown Object list, enter a new record using the values shown in the following table.

Field	Values for First Drilldown
Name	Activities
Hyperlink Field	Name
View	Opportunity Detail - Activities (SCW)

To create Drilldown Destination Objects

1. Select the drilldown object to serve as the parent (default) of your dynamic drilldown destinations.

   For the current example, select the drilldown object named *Original*.

   The target view defined in this drilldown object serves as the default for when the conditions in the Drilldown Destination objects are false.
2. In the Object Explorer, select Dynamic Drilldown Destination object type (a child object of Drilldown Object).
3. In the Dynamic Drilldown Destinations applet, enter the records as shown in the following table.

Name	Field	Value	Destination Drilldown Object	Sequence
Prospecting	Sales Stage	01 - Prospecting	Activities	1
Qualification	Sales Stage	02 - Qualification	Activities	2
Closing	Sales Stage	03 - Closing	Activities	3

4. Compile and Unit Test.

   Navigate to the Opportunity List Applet. When you drill down on the name column it takes you to Opportunity Detail - Contacts (SCW) view or Opportunity Detail - Activities (SCW) depending on the value of the Sales Stage field.

Configuring Multi-Value Groups

Multi-value groups (MVGs) incorporate child data into an applet. They provide you with a way to display multiple child records for a single parent record, yet they do not require a master-detail view. Instead, the user clicks the select button in a multi-value field, and an MVG applet pops up displaying a set of child records. For example, NREC's requirement is to create a MVG on the Opportunity Form Applet (SCW) to display the contacts associated with each opportunity. The results of the configuration work are shown in Figure 25.

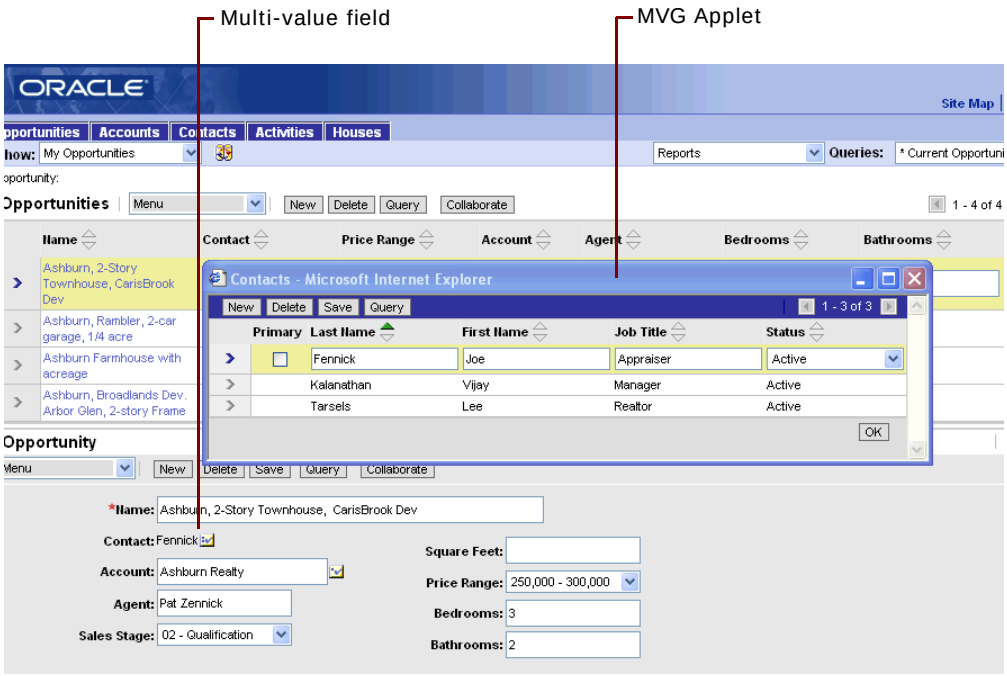


Figure 25. Multi-Value Field

Creating an MVG

You can use the MVG Wizard to create the objects necessary for MVGs. Based on the information you enter, the wizard creates the following objects that are necessary to implement the MVG.

- Multi-Value Link
- Multi-Value Field
- Multi-Value Group Applet, if necessary

To configure a multi-value field using the MVG Wizard

1 Choose File > New Object.

The New Object Wizards dialog box appears.

2 Under the General tab, click the MVG icon and then click OK.

The Multi Value Group wizard appears.

3 In the Multi Value Group dialog box, enter the values as shown in the following table, and then click Next.

Field	Value	Description
Project	NREC Configuration	Project to which the MVG will belong.
Master Business Component	Opportunity	The name of the master business component. For example, Opportunity is the parent business component in the Opportunity/Contact Link object.

4 In the second Multi Value Group dialog box, enter the values as shown in the following table, and then click Next.

Field	Value	Description
Detail Business Component	Contact	This is the name of the detail business component. For example, Contact is the child business component defined in the Opportunity/Contact link object.
Multi Value Link Name	NREC Contact	Unique name for the Multi Value Link object.

5 In the Direct Links dialog box, select the Opportunity/Contact link and then click Next.

This dialog box displays every link that describes a master-detail relationship between the business components defined in Step 3 and Step 4.

6 In the Primary ID Field dialog box, do not select any fields or check boxes. Click Next.

NOTE: Although this procedure does not cover implementing primaries, doing so can have performance benefits by reducing the number of SQL queries that are executed to populate the MVG for each record.

7 In the Multi Value Link dialog box, accept the default for Multi Value Link properties; leave every check box clear.

8 In the Multi Value Fields dialog box, use the drop-down list to choose a field from the detail business component to use to create the multi-value field in the master business component.

a From the drop-down list, choose the Last Name field.

b Use the default value of NREC Contact Last Name for the Multi Value Field Name.

c Click Add and then click Next.

9 In the Finish dialog box, review the information you entered and then click Finish.

The Multi Value Group Wizard uses the information you entered to create the necessary objects for a multi-value group. It also searches the repository for an existing multi-value group applet that can be used to display the data in the user interface. If a suitable applet is not found, the MVG Applet Wizard opens so you can define one.

In the current example, the Contacts MVG Applet already exists, so the MVG Applet wizard does not open.

Exposing the MVG in the User Interface

After you create the underlying objects necessary to display the MVG, you can expose the multi-value field in the user interface. The Opportunity Form (SCW) applet already has a Contact field. It needs to be redefined so it shows the Contact MVG.

To expose the MVG in the user interface

1 In the Object Explorer, select the Opportunity Form (SCW) applet.

2 Expand the Control object type (child of Applet).

3 Select the Contact control.

4 Change the fields as shown in the following table.

Field	Current Value	New Value
Field	Key Contact Id	NREC Contact Last Name
MVG Applet	<blank>	Contact MVG Applet
Runtime	FALSE	TRUE

NOTE: The Contacts MVG is suitable for NREC's purposes. You could also have created a new MVG applet using the MVG Applet Wizard.

5 Compile and Unit Test.

The multi-value field appears in the applet with a select button next to it. When the user clicks the select button, the MVG applet appears. See Figure 25 on page 129.

Testing the Configuration Changes

NREC's testing plan calls for periodic integration and regression testing. These tests are necessary to make sure that the changes you have made up to this point do not prevent previously existing features from working, and that the application still can interact with other applications and products. NREC's test plan makes sure these checks happen at specific strategic intervals in the configuration process. In the event a test fails, NREC can quickly locate the faulty configuration change and make the appropriate modifications. If these tests are performed too infrequently, then troubleshooting errors may be a major effort.

9 Creating a Virtual Business Component

This chapter describes creating a virtual business component. It follows the NREC example, building on work you did in Chapter 7, "Configuring the House and Opportunity Entities," for configuring the House entity.

Read this chapter for a general overview of virtual business components and the steps for configuring them. Keep in mind that this example is relatively simple. It uses a simple file as an external data source rather than an external database or external application.

For more information about how virtual business components can be used to integrate external data, read *Overview: Siebel Enterprise Application Integration* and *Integration Platform Technologies: Siebel Enterprise Application Integration*.

Understanding Virtual Business Components

Virtual business components (VBCs) bring data from external data sources into your Siebel application user interface. For example, you could create a VBC to represent data from data sources such as:

- Back office applications
- Legacy applications
- Transaction services
- Web sites

Virtual business components are based on business services. Business services determine the behavior of the VBC and define how the VBC manipulates data. You can base VBCs on predefined business services provided by Oracle, such as the XML Gateway, or you can create your own business service. You create both virtual business components and business services in Siebel Tools.

NREC's Virtual Business Component

Assume that NREC stores house renovation data in a comma-delimited text file that a third-party vendor provides to NREC. NREC's design includes a virtual business component used to retrieve the data from the file and display it in the Partner Portal application. Partner agents and NREC employees can use it to display a history of renovation information for each house stored in NREC's Siebel database. Figure 26 shows NREC's external file of renovation data. Figure 27 shows how the data from this file is displayed in the user interface.

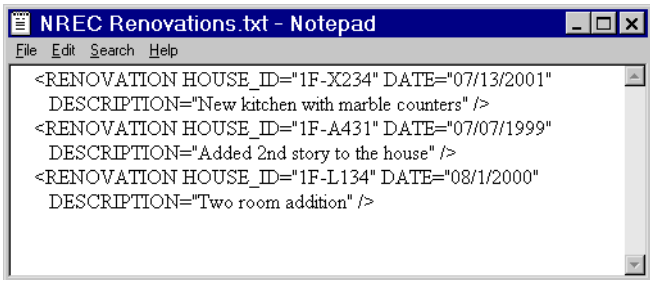


Figure 26. NREC's External File of Renovation Data

NREC's requirement is to display this data in a standard master-detail view showing a house record in the top applet and a list of renovations in the bottom applet.

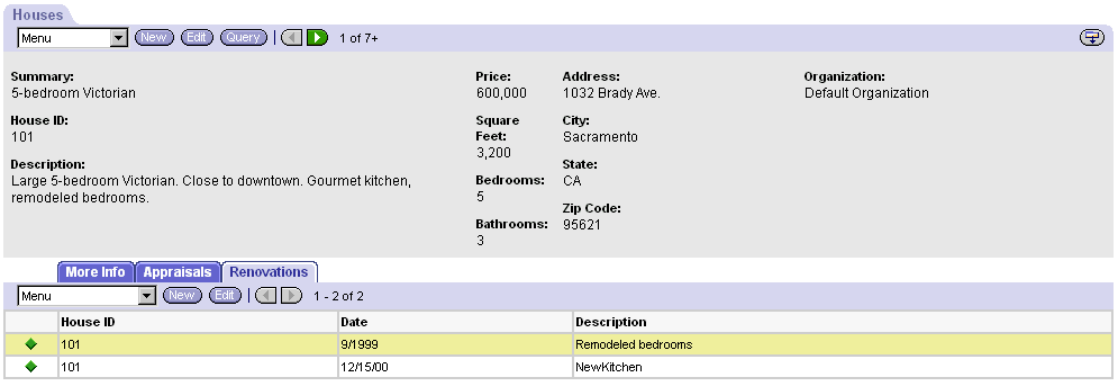


Figure 27. Applet Displaying Data from a VBC

The steps for configuring a virtual business component for the NREC's scenario are:

- "Creating a Business Service" on page 135
- "Creating a Virtual Business Component" on page 137
- "Creating Fields for the Virtual Business Component" on page 138
- "Defining User Properties for the Virtual Business Component" on page 138
- "Creating a Link" on page 139

Creating a Business Service

Business services are objects that define sets of functionality. For example, a business service can perform tasks such as moving data or converting data formats. Like business components, business services are objects stored in the Siebel repository. However, rather than interacting with tables and data as business components do, business services interact with other objects.

Virtual business components use business services to provide the functionality to manipulate external data. Business services define the methods, properties, and states that determine the behavior of virtual business components.

There are many predefined business services provided for specialized needs, such as using XML to display data in a Siebel application. You can also create your own business services. In the NREC example, you use Siebel Tools to create a business service that reads and writes data to a comma-delimited text file.

For more information about business services, read *Siebel Object Interfaces Reference* and *Integration Platform Technologies: Siebel Enterprise Application Integration*.

In the NREC example, the steps for creating the business service are:

Defining the Business Service

You create business service objects using Siebel Tools.

To define a business service

1. Create a symbolic string for the term “NREC Text File Handler”.
2. In the Object Explorer, expand the Business Service object type.
3. In the Business Services window, enter a new record as shown in the following table.

Field	Value
Name	NREC Text File Handler
Project	NREC Configuration
Class	CSSService
Display Name	NREC Text File Handler

Defining Business Service Scripts

After defining the Business Service, you can define the business service scripts that actually do the processing associated with the NREC Text File Handler business service object. Sample code for this example is shown in "Code Samples" on page 143.

Business Service methods to implement are listed in Table 10.

Table 10. Business Service Methods

Method	Custom Method Name in Code Sample	Description
Init	Pal_Init	Creates the initial link between the columns in the VBC and the columns in the external data source.
Insert	Pal_Insert	Inserts a record into the external data source.
PreInsert	Pal_PreInsert	Provides any default values for new rows. In the current example, there are no default values.
Query	Pal_Query	Queries the external data source and returns rows that match a given value. In the current example, the method queries the text file and returns rows that have a matching value for the current House_ID field.
Service_PreInvokeMethod	Service_PreInvokeMethod	Handles requests that come into the business service and calls other functions as appropriate.
Update	Not used in sample code	Updates an existing row in an external data source.
Delete	Not used in sample code	Deletes a row in the external data source.

For more information about business service methods, read *Siebel Object Interfaces Reference*.

To define and write the business service script for NREC Text File Handler

1. Select the NREC Text File Handler business service.
2. With the business service selected, right-click, and then choose Edit Server Scripts.
3. Select Visual Basic as the scripting language and then click OK.

   The script editor appears with Service_PreInvokedMethod selected as the service.
4. Enter the custom functions by doing the following.

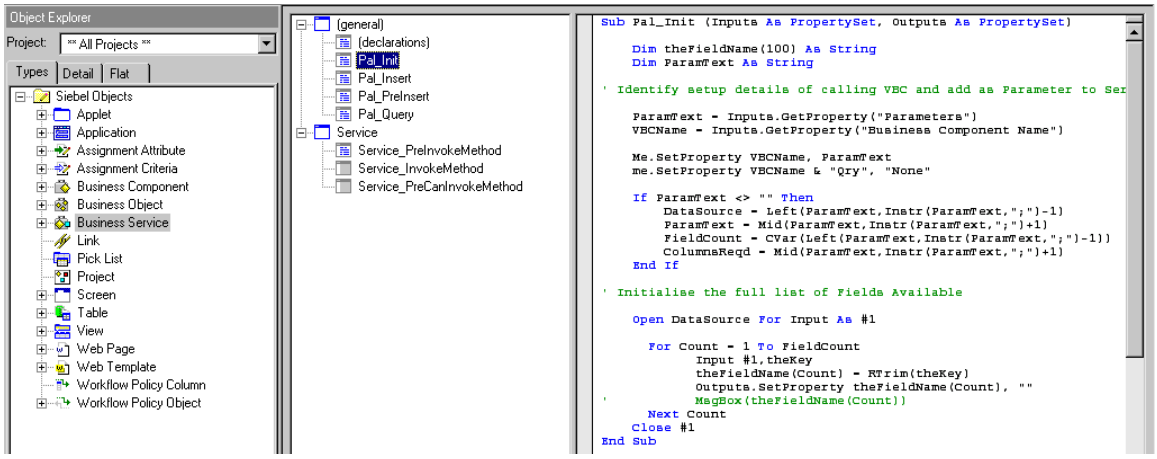
a Expand the (general) icon in the script editor and then select the (declarations) icon.

b In the text entry window, enter the code for the custom function.

c Choose File > Save.

An icon representing the function appears in the left pane as shown in the following illustration.

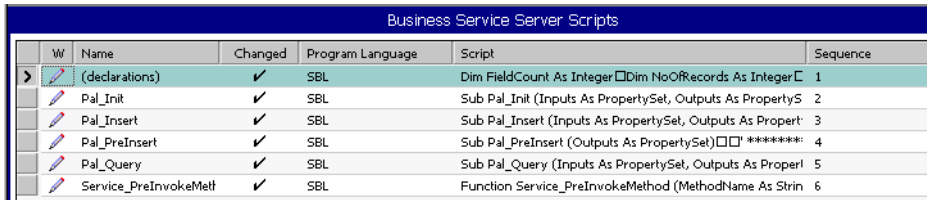
d Repeat Step a through Step c for each custom function.



5 Select the Service_PreInvokeMethod icon. Copy the sample code into the text entry window, replacing the existing stub text.

6 Choose File > Save.

After saving your work, Business Service Server Script objects are available, as shown in the following illustration.



Creating a Virtual Business Component

When creating a virtual business component you do not use the Business Component Wizard because it forces you to select a table on which the business component is to be based and does not set the class to CSSBCVExtern.

To create a virtual business component

1 In the Object Explorer, select the Business Component object type.

2 Right-click in the Object List Editor and choose New Record.

3 Enter Values for the following properties as shown in the following table.

Field	Value	Comments
Name	NREC Renovations VBC	Unique name for the virtual business component
Project	NREC Configuration	Choose a locked project
Class	CSSBCVExtern	Provides the virtual business component functionality

Creating Fields for the Virtual Business Component

After creating the virtual business component you need to create the fields to display in the Siebel application. These fields display data retrieved from the House Renovations file shown in Figure 26 on page 134.

NREC's requirement is to define the following fields for the business component:

- Date
- Description
- House_ID

To add fields to the virtual business component

1 In the Object Explorer, expand the Business Component Object type.

2 In the Object List Editor, select NREC Renovations VBC.

3 In the Object Explorer, select Single Value Field object type (child of Business Component).

4 Enter each of the records as shown in the following table.

Name	Type	Comments
Date	DTYPE_TEXT	Date the renovation was recorded
Description	DTYPE_TEXT	Description of the renovation
House_ID	DTYPE_TEXT	House Identification Number

Defining User Properties for the Virtual Business Component

The user properties required by the NREC Renovations VBC are Service Name and Service Parameters. The Service Name user property defines which business service the VBC invokes. The Service Parameters define the parameters that are passed into the business service upon initialization. In this example, the parameters are defined in the script attached to the business service. Read the Pal_Init function in the "Code Samples" on page 143.

To define virtual business component user properties

1 Select the NREC Renovations VBC business component.

NOTE: If the Business Component User Prop is not visible in the Object Explorer, then you need to make it visible by choosing View > Options to display the Development Tools Options dialog box, then clicking the Object Explorer tab. The check box for Business Component User Prop is in the Business Component hierarchy.

2 In the Object Explorer, select the Business Component User Prop object type (child of business component).

3 Add the user properties records as shown in the following table.

User Property Name	Value	Comments
Service Name	NREC Text File Handler	Name of the business service.
Service Parameters	D:\NREC\nrec.txt;3;House_ID, Date,Description	The script for the current example specifies the service parameters as the full path to the file name, number of columns, and column names. The parameters must be separated by semicolons, and there can be no spaces in the comma-delimited strings. Service parameters vary depending on the business service being used.

Creating a Link

The relationship between houses and house renovations is one to many (1:M); there can be many renovations recorded for one house. Therefore, you need a link to create the parent-child relationship between the Internal Product business component and the NREC Renovations VBC business component. Internal Product is the business component that NREC is using to store houses. See NREC's entity relationship diagram shown in Figure 3 on page 28.

To create a link between parent and child business components

1 Select the Link object type.

2 In the Links window, add a new record as shown in the following table.

Field	Value	Description
Name	Internal Product/NREC Renovations VBC	Unique name for the link.
Project	NREC Configuration	Project to which the object belongs.
Parent Business Component	Internal Product	Parent business component.

Field	Value	Description
Child Business Component	NREC Renovations VBC	Child Business component.
Source Field	Part #	Unique ID on the parent record. Often this field is Row Id. In this case, the unique field is a manually entered value.
Destination Field	House_ID	Column in the child record that stores the parent ID.

Updating the Business Object

Like any business component, a virtual business component must be grouped together with objects of similar focus using a business object. Now that you have a link defined for the Internal Product/ NREC Renovations VBC relationship, you need to add the Renovation VBC business component to the Internal Product business object.

To add the Renovations business component to the Internal Product business object

1. Expand the Business Object object type.
2. Select the Internal Product business object.
3. Select the Business Object Component object type (child of business object).
4. In the Business Object Components window, add a new record as shown in the following table.

Field	Value
Business Component	NREC Renovations VBC
Link	Internal Product/NREC Renovations VBC

The NREC Renovations VBC business component is added to the Internal Product business object.

Exposing the Fields in the User Interface

When the business object layer has been configured, you can expose the fields defined in the NREC Renovations VBC in the user interface. The steps to do this include:

- “Creating a New List Applet” on page 141
- “Creating a New View” on page 141
- “Adding the View to a Screen” on page 142

Creating a New List Applet

To display the data from the NREC Renovations VBC you must create a new applet. The NREC design specifies a standard list applet. To create it you use the List Applet New Object wizard. In the next section, you create a master-detail view to contain the applet.

To create a new list applet to display renovation data

1 Choose File > New Object.

The New Object dialog box appears.

2 Click the Applets tab and then select List Applet.

3 Complete the wizard using the values as shown in the following table.

Field	Value
Project	NREC Configuration
Applet Name	Renovations List Applet
Display Title	Renovations
Business Components	NREC Renovations VBC
Web Templates	Base: Applet List (Base/List Edit) Edit List: Applet List Edit (Edit/New/Query) Edit: Applet List Edit (Edit/New/Query)
Fields	House_ID, Date, Description
Controls	Accept default controls selected for the applet.

Creating a New View

To display the Renovations list applet you need to add it to a view. NREC requires a standard master-detail view, showing the House Form applet on top and the Renovations list applet on the bottom. This displays the renovation records for each house record.

To create a new view

1 Choose File > New Object.

2 In the New Objects Wizard dialog box, select View.

The New View Wizard opens.

3 Complete the wizard using the values as shown in the following table.

Field	Value
Project	NREC Configuration
Name	House Details - Renovations View
Title	Renovations
Business Object	Internal Product
Web Template	View Detail (Parent with Pointer)
Applets	Product Form Applet Renovations List Applet

Adding the View to a Screen

The next step is to add the new view to a screen. For the current example, you add the House Details - Renovations View to the NREC House screen.

For information about the Houses Screen, read "Creating the Houses Screen" on page 97.

To add a view to a screen

1 Add a symbolic string for the term "Renovations".

2 Expand the Screen Object type.

3 Select the NREC House Screen.

4 Select the Screen View object type (child of Screen).

5 In the Screen View window, add a new record using the properties as shown in the following table.

Field	Value
View	House Details - Renovations View
Type	Detail View
Parent Category	NREC
Viewbar Text	Renovations
Sequence	30
Menu Text	Renovations

Compiling and Testing

After you have completed your configuration work, perform the following tasks.

- Compile your changes. Be sure to compile changes for the projects that contain objects you modified. For example, the Internal Product business object belongs to the Product project by default.
- Register the House Details - Renovations View in the application.
- Add House Renovations to a responsibility that you can test.
- Unit Test your work.

  To see what the results look like, review Figure 26 on page 134.

Code Samples

The following is the code breakdown for the NREC's example business service. Read "Defining Business Service Scripts" on page 136 for the steps for defining the code and associating it with the business service. There are two sets of code to be copied:

- "Code to Copy into General:Declarations"
- "Code to Copy into Service:Service_PreInvokeMethod" on page 149

Code to Copy into General:Declarations

For each of the following code samples, copy the code into the General:Declarations text entry window, then choose File > Save.

- "(declaration)"
- "Pal_Init"
- "Pal_Insert" on page 144
- "Pal_PreInsert" on page 145
- "Pal_Query" on page 146

(declaration)

```
Dim FieldCount As Integer
Dim NoOfRecords As Integer
Dim DataSource As String
Dim ColumnsReqd As String
Dim VBCName as String
```

Pal_Init

```
Sub Pal_Init (Inputs as PropertySet, Outputs as PropertySet)

   Dim theFieldName(100) As String
   Dim ParamText as String

' Identify setup details of calling VBC and add as Parameter to Service
```

```
   ParamText = Inputs.GetProperty("Parameters")
   VBCName = Inputs.GetProperty("Business Component Name")

   Me.SetProperty VBCName, ParamText
   me.SetProperty VBCName & "Qry", "None"

   If ParamText <> "" Then
      DataSource = Left(ParamText,Instr(ParamText,";")-1)
      ParamText = Mid(ParamText,Instr(ParamText,";")+1)
      FieldCount = CVar(Left(ParamText,Instr(ParamText,";")-1))
      ColumnsReqd = Mid(ParamText,Instr(ParamText,";")+1)
   End If

   ' Initialize the full list of Fields Available

   Open DataSource For Input As #1
      For Count = 1 To FieldCount
         Input #1,theKey
         theFieldName(Count) = RTrim(theKey)
         Outputs.SetProperty theFieldName(Count), ""
      Next Count
   Close #1

   End Sub
```

Pal_Insert

```
   Sub Pal_Insert (Inputs As PropertySet, Outputs As PropertySet)

   Dim InputFields(100) As String
   Dim InputValues(100) As String
   Dim NoOfValues As Integer
   Dim InputPS As PropertySet
   Dim OutputStr As String
   Dim PadReqd As Integer

   NoOfValues = 0

' ****************************************************************************
' * Establish the Value Pairs to Be Used for Insert from the Inputs Property set *
' ****************************************************************************

   If Inputs.GetFirstProperty <> "" Then
      Set InputPS=TheApplication.NewPropertySet()
      Set InputPS = Inputs.GetChild(0)
      InputFields(1) = InputPS.GetFirstProperty
      InputValues(1) = InputPS.GetProperty(InputPS.GetFirstProperty)
      NoOfValues = NoOfValues + 1
      For i = 2 to FieldCount
         InputFields(i) = InputPS.GetNextProperty
         If InputFields(i) <> "" Then
            InputValues(i) = InputPS.GetProperty(InputFields(i))
         NoOfValues = NoOfValues + 1
```

```
         End If
      Next i
   End If

' ************************************************************************
' * Create the Output String for Writing to the File in the right order *
' ************************************************************************

   Open DataSource For Input As #1

   For Count = 1 To FieldCount
      PadReqd = 1 ' Identify by default we need a Pad string in the file
      Input #1,theKey
      For i = 1 to NoOfValues
         If InputFields(i) = theKey Then
            PadReqd = 0
            If OutputStr <> "" Then
               OutputStr = OutputStr & ","
            End If
            OutputStr = OutputStr & InputValues(i)
         End If
      Next i
      If PadReqd = 1 Then
         If OutputStr <> "" Then
            OutputStr = OutputStr & ","
         End If
         OutputStr = OutputStr & " "
      End If
   Next Count

      Close #1

' ************************************************
' * Create the Output String In the DataSource File *
' ************************************************

      Open DataSource For Append As #1

      Print #1,OutputStr

      Close #1

   End Sub
```

Pal_PreInsert

```
   Sub Pal_PreInsert (Outputs As PropertySet)

' ********************************************************************************
' * A Pre Insert Method is Required before any insert is applied, Its use is     *
' * to provide Default Values, This service does not provide any defaults, But   *
' * must return a populated property set, so this points to the first field by default. *
' ********************************************************************************
```

```
      dim newRow As PropertySet
      set newRow = TheApplication.NewPropertySet()
      newRow.SetProperty Left(ColumnsReqd,Instr(ColumnsReqd,",")-1),""
      Outputs.AddChild newRow

   End Sub
```

Pal_Query

```
   Sub Pal_Query (Inputs as PropertySet, Outputs as PropertySet)

' ************************************************************************
' * The adding of properties in the me.SetProperty statements in this    *
' * code is used To store key information and the relevant data against  *
' * parameters identified by the VBC name. This technique is used to create *
' * a persistantstore and negate yhe need for repeat calls to the external  *
' * source for the same data.                                            *
' ************************************************************************

   Dim Row(999) As PropertySet

   Dim theFieldName(100) As String
   Dim QueryValues(100) As String
   Dim QueryFields(100) As String
   Dim QueryList As PropertySet
   Dim ColList, ColData, QryData As String
   Dim OutColCount, OutRowCount, NoOfQs, RowMatches, DoQuery As Integer

   DoQuery = 1
   NoOfQs = 0
   OutColCount = 0
   OutRowCount = 0

' ************************************************************************
' * Establish the list of Columns Which define the Query Values from the VBC *
' ************************************************************************

   Set QueryList = TheApplication.NewPropertySet()
   Set QueryList = Inputs.GetChild(0)

   ' Inputs.GetChild(0) retrieves user inputs ,like ("First Name","Angela") , etc...

   If QueryList.GetFirstProperty <> "" Then

      QueryFields(1) = QueryList.GetFirstProperty
      QueryValues(1) = QueryList.GetProperty(QueryList.GetFirstProperty)
      NoOfQs = NoOfQs + 1
      QryData = QueryFields(1) & "=" & QueryValues(1)
      For i = 2 to FieldCount
         QueryFields(i) = QueryList.GetNextProperty
         If QueryFields(i) <> "" Then
            QueryValues(i) = QueryList.GetProperty(QueryFields(i))
            NoOfQs = NoOfQs + 1
            QryData = QryData & ";" & QueryFields(i) & "=" & QueryValues(i)
```

```
         End If
      Next i
   End If

' ********************************************************************************
' * Establish If Query is same as Last Time for this VBC and If So Set Query Flag *
' * to use Persistant Cache
' ********************************************************************************

     If me.GetProperty(VBCName & "Qry") <> QryData Then
        me.SetProperty VBCName & "Qry", QryData
     Else
        DoQuery = 0
     End If



' ********************************************************************************
' * Establish Full List of Columns Available and Which ones are needed by the VBC *
' * Where not required, Identify this with Column Value Not Required CVNR         *
' ********************************************************************************

   If DoQuery = 1 Then

   Open DataSource For Input As #1
      For Count = 1 To FieldCount
         Input #1,theKey
         If Instr(ColumnsReqd,theKey) <> "0" Then
            theFieldName(Count) = RTrim(theKey)
            OutColCount = OutColCount + 1
         Else
            theFieldName(Count) = "CVNR"
         End If
   Next Count

   me.SetProperty VBCName & "ColCnt", CStr(OutColCount)

' ********************************************************************************
' * Retreveal of Data from Text File, Including ignoring Columns Not Required and *
' * Ignoring rows which do not match the selection criterion                      *
' ********************************************************************************

   For r = 1 to 999

      On Error Goto EndLoop
      rowMatches = 1  'set to true
   Input #1, theKey
   theValue = RTrim(theKey)
      If theValue <> "" Then
         Set Row(r) = TheApplication.NewPropertySet()
         If theFieldName(1) <> "CVNR" Then
            Row(r).SetProperty theFieldName(1), theValue
            ColList = theFieldName(1)
            ColData = theValue
         For x = 1 to NoOfQs
            If QueryFields(x) = theFieldName(1) and QueryValues(x) <> theValue Then
```

```
               rowMatches = 0
            End If
         Next x
      End If
   For Count = 2 To FieldCount
      Input #1, theKey
      theValue = RTrim(theKey)
         If theFieldName(Count) <> "CVNR" Then
            Row(r).SetProperty theFieldName(Count), theValue
            ColList = ColList & ";" & theFieldName(Count)
            ColData = ColData & ";" & theValue
         For x = 1 to NoOfQs
            If QueryFields(x) = theFieldName(Count) and QueryValues(x) <> theValue Then
               rowMatches = 0
            End If
         Next x
      End If
   Next Count
      If rowMatches = 1 Then
         Outputs.AddChild Row(r)
         OutRowCount = OutRowCount + 1
         me.SetProperty VBCName & CStr(OutRowCount), ColData
      End If
   Else

EndLoop:

         Exit For
      End If
   Next r
         Close #1
         me.SetProperty VBCName & "RowCnt", CStr(OutRowCount)
         me.SetProperty VBCName & "Col", ColList
       Else

' ****************************************************************************
' * Re Create Output property set from the Persistant Cache held in the Business *
' * Service Property Set
' ****************************************************************************

      OutColCount = CVar(GetProperty(VBCName & "ColCnt"))
      OutRowCount = CVar(GetProperty(VBCName & "RowCnt"))
      For i = 1 to OutRowCount
         ColList = GetProperty(VBCName & "Col")
         ColData = GetProperty(VBCName & CStr(i))
         Set Row(i) = TheApplication.NewPropertySet()
         For x = 1 to OutColCount-1
            theField = Left$(ColList,Instr(ColList,";")-1)
            theValue = Left$(ColData,Instr(ColData,";")-1)
            ColList = Mid$(ColList,Instr(ColList,";")+1)
            ColData = Mid$(ColData,Instr(ColData,";")+1)
            Row(i).SetProperty theField, theValue
         Next x
         Row(i).SetProperty ColList, ColData
```

```
         Outputs.AddChild Row(i)
      Next i
   End If

End Sub
```

Code to Copy into Service:Service_PreInvokeMethod

```
Function Service_PreInvokeMethod (MethodName As String, Inputs As PropertySet, Outputs
As PropertySet) As Integer

   Dim ParamText as String

'******************************************************************************
'* SYSTEM DEFINITION SECTION
'*
'* The Calling V.BusComp must include the following parameters
'*
'* Service Name = Pal Mk2 (Unless Renamed)
'* Service Parameters = <Full Path & DataFile Name>;
'*       <No Of Columns in File>;
'*       <Columns in Calling VBC comma separated>
'*
'* e.g Service Parameters = c:\Siebel\Data.VBC;5;KeyId,FName,LName,Age,DOB
'*
'******************************************************************************

' Identify the Context the Service is Called from and setup parameters

   VBCName = Inputs.GetProperty("Business Component Name")
   ParamText = me.GetProperty(VBCName)
   If ParamText <> "" Then
      DataSource = Left(ParamText,Instr(ParamText,";")-1)
      ParamText = Mid(ParamText,Instr(ParamText,";")+1)
      FieldCount = CVar(Left(ParamText,Instr(ParamText,";")-1))
      ColumnsReqd = Mid(ParamText,Instr(ParamText,";")+1)
   End If

' Handle the Method

   If MethodName = "Init" Then
      Pal_Init Inputs, Outputs
      Service_PreInvokeMethod = CancelOperation
   ElseIf MethodName = "Query" Then
      Pal_Query Inputs, Outputs
      Service_PreInvokeMethod = CancelOperation
   ElseIf MethodName = "Update" Then
      'Pal_Update Inputs, Outputs         - Not Yet Completed
      Service_PreInvokeMethod = CancelOperation
   ElseIf MethodName = "PreInsert" Then
      Pal_PreInsert Outputs
      Service_PreInvokeMethod = CancelOperation
   ElseIf MethodName = "Insert" Then
```

```
      Pal_Insert Inputs, Outputs
      Service_PreInvokeMethod = CancelOperation
   ElseIf MethodName = "Delete" Then
      'Pal_Delete Inputs        - Not Yet Completed
      Service_PreInvokeMethod = CancelOperation
   Else
      Service_PreInvokeMethod = ContinueOperation
   End If

End Function
```

Testing the Virtual Business Component

NREC's testing plan calls for periodic integration and regression testing. These tests are necessary to make sure that the changes you have made up to this point do not prevent previously existing features from working, and that the application still can interact with other applications and products. NREC's test plan makes sure these checks happen at specific strategic intervals in the configuration process. In the event a test fails, NREC can quickly locate the faulty configuration change and make the appropriate modifications. If these tests are performed too infrequently, then troubleshooting errors may be a major effort.

Documenting Your Changes in an ERD

NREC has made changes to data objects (Chapter 7, "Configuring the House and Opportunity Entities") and has added a virtual business component (Chapter 9, "Creating a Virtual Business Component"). Document these changes in an Entity-Relationship Diagram (ERD).

Figure 28 shows the NREC Entity Relationship Diagram. This is a sketch of the relationships between data objects. In Siebel Tools you can create a true ERD that is stored in the database.

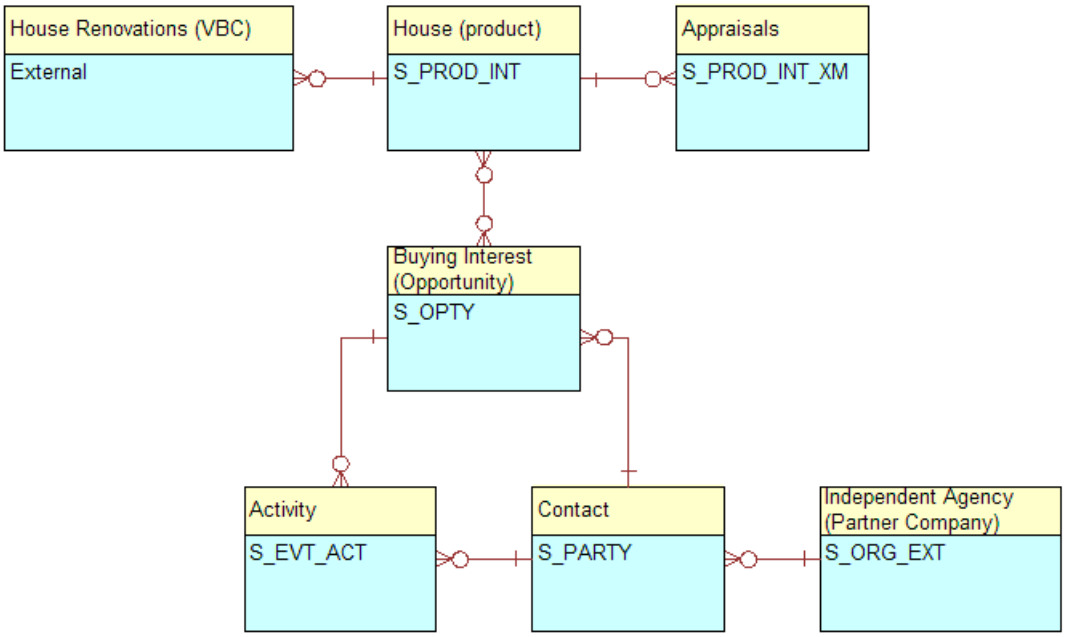


Figure 28. NREC Entity Relationship Diagram

To create an entity relationship diagram

1. In the Object Explorer, select the Entity Relationship Diagram object type.
2. Right-click and then choose New Record. Enter the values as shown in the following table.

Field	Value
Name	NREC House ERD
Project	NREC Configuration
Status	Scope

3. Right-click on the record and then choose Edit Entity Relationship Diagram.

   The Entity Relationship Diagram Canvas appears.

4. Drag and drop entity shapes from the palette to the canvas. Use Figure 28 as a guide. You need seven entity shapes.
5. Click on the canvas background to deselect any shapes.

   In the Entities list, all seven entities (from Entity0 to Entity6) are listed.

6 Edit the Entity information to name each one and bind it to a business component as shown in the following table.

Entity	Name	Business Component
Entity0	House Renovations (VBC)	NREC Renovations VBC
Entity1	House (Product)	Internal Product
Entity2	Appraisals	Appraisals
Entity3	Buying Interest	Opportunity
Entity4	Activity Plan	Activity Plan
Entity5	Contact	Contact
Entity6	Independent Agency (Partner)	Account Home Search

7 Add relationship lines (1:1, 1:M, or M:M) to all entities, using Figure 28 on page 151 as a guide.

Attach the left side of the line to an entity, and wait for the line to redisplay to show its endpoints. Then you can connect the right side of the line to the next entity. The lines start off as straight lines, directly connecting from one entity to another. They do not reshape themselves to avoid other lines or entities. In a later step you can adjust the shapes of the lines to improve the legibility of the diagram.

8 For each relationship line, bind its relationship.

a Right-click a relationship line and choose Bind Entity Relation.

The Bind Relationships window opens.

b Select a relationship and then click OK. If there are multiple relationships possible, look carefully at the business components and tables listed and select the most appropriate pairing. If no relationships are listed, then there is no link or join between the business components.

When the relationship binds, the line thickness increases.

9 (Optional) Improve legibility by reshaping the relationship lines.

a Select a relationship line.

b Add a reshaping point by pressing Ctrl+A.

c Move the reshaping point as needed to adjust the line.

d Repeat Step b and Step c as needed.

10 Edit relationship line names.

a In the Object Explorer, select Entity Relation (child of Entity Relationship Diagram).

b Adjust the column display to show the Name, Entity 1, and Entity 2 columns.

c Edit the name of each relationship line to indicate the names of both entities.

d Keep the names short to improve legibility. Use abbreviations or a single word to indicate each entity. For example, with the Independent Agency (Partner) link to Real Estate Agent, use the name Partner - Agent.

11 Redisplay the ERD.

a In the Object Explorer select the Entity Relationship Diagram object.

b Select the NREC House ERD record.

c Right-click on the record and then choose Edit Entity Relationship Diagram.

The Entity Relationship Diagram Canvas appears.

When done, the entity relationship diagram resembles Figure 29.

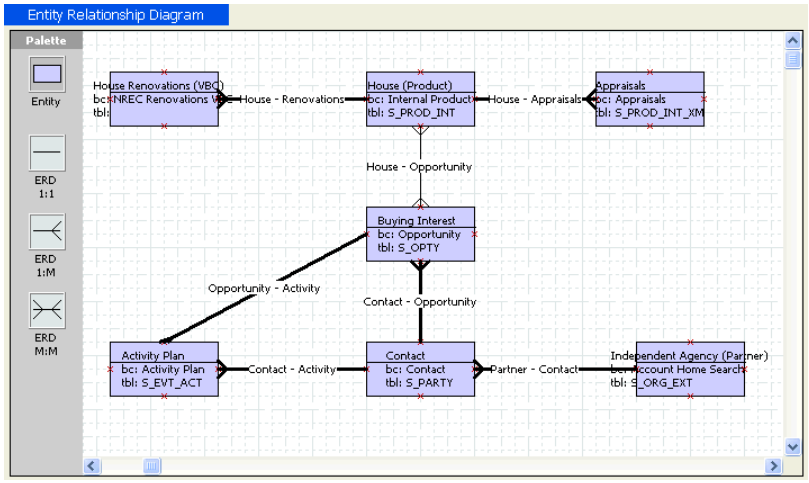


Figure 29. NREC Entity Relationship Diagram in Siebel Tools

For more information on entity relationship diagrams, read *Configuring Siebel Business Applications*.

10 Modifying the Look and Feel of the Web Client

This chapter follows the NREC example to highlight key tasks and information about configuring the look and feel of the user interface. These tasks lead you through the process of modifying the principal user interface elements based on NREC's requirements. Tasks include modifying graphics, Web templates, and cascading style sheets.

NREC's requirement is to modify the look and feel of the user interface to match their corporate Web site. This includes modifying the shape of tabs and buttons and changing the color scheme.

For the before-and-after look of the user interface, see Figure 30 and Figure 31.

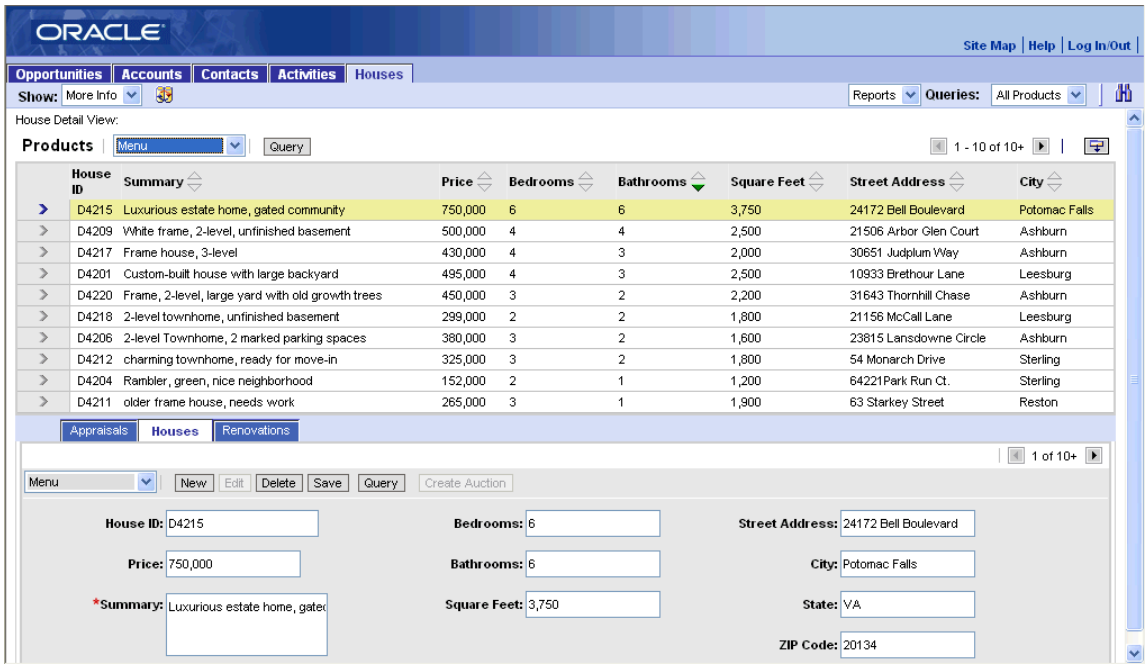


Figure 30. Standard Look and Feel

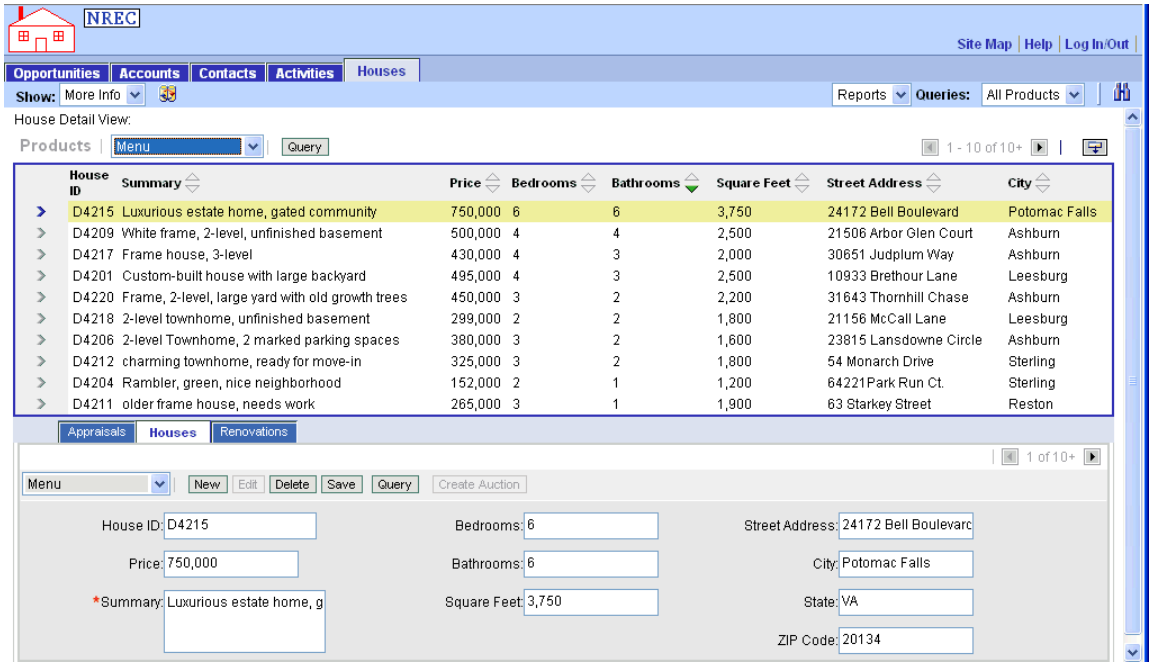


Figure 31. NREC's Modified Look and Feel

The tasks that are required to make the changes are covered in the rest of this chapter.

User Interface Elements

The user interface elements discussed in the this chapter are shown in Figure 32 and Figure 33. For reference information about Web templates and other user interface topics, read *Configuring Siebel Business Applications*.

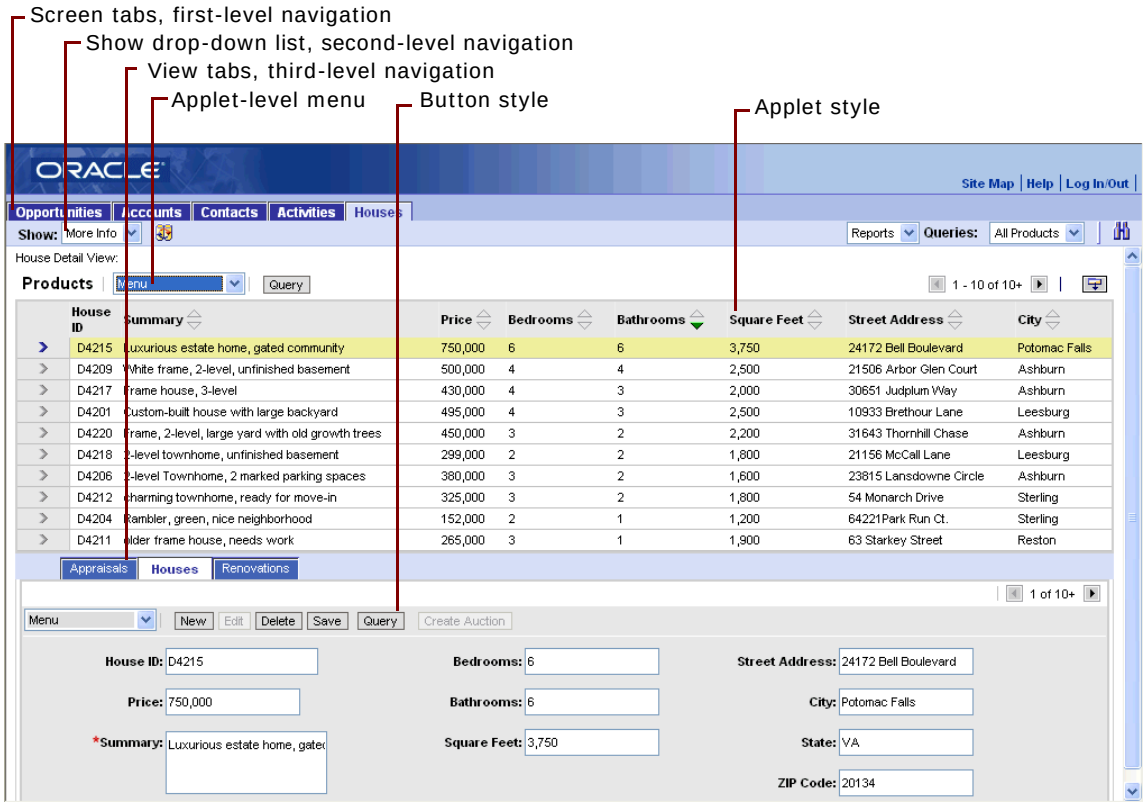


Figure 32. UI Elements

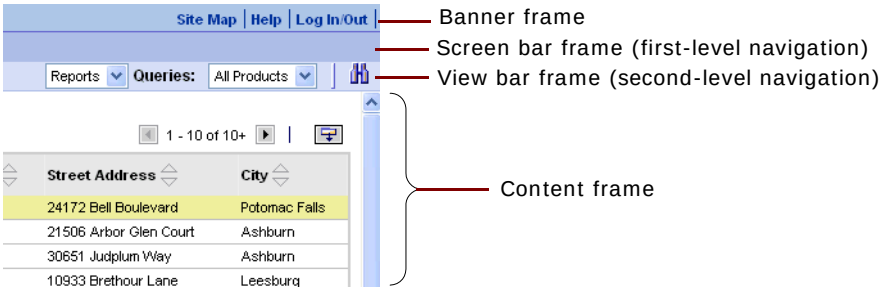


Figure 33. Frames

Source Control for Web Template and Related Files

Siebel Web templates and supporting files such as graphics and cascading style sheets are not stored in the Siebel repository. They are stored under the root directory of your Siebel installation. Because these files are outside a Siebel repository they are not controlled by the Siebel Tools Check In and Check Out Process. If multiple developers are modifying files outside the repository, use a third-party source control application to make sure developers do not overwrite each other's work.

For the locations of Web templates and supporting files, read "Location of Siebel Web Templates and Related Files."

Location of Siebel Web Templates and Related Files

Siebel Web templates and supporting files are installed with the Siebel Server, Siebel Tools, and the Mobile and Dedicated Web Clients. The locations of the files are listed below.

NOTE: In the directory paths listed below, *language_code* is the three letter code of the language pack for the installed software. For example, ENU is the code for English and FRA is the code for French.

- **Siebel Tools.** The files in the Siebel Tools installation directory are used when objects such as views and applets are rendered using the Preview option in the Edit Web Layout window. The locations of the files in the Siebel Tools installation directory are shown in the following table.

Files	Directory Path
Image files	tools_root\PUBLIC*language_code*\IMAGES
Style Sheets	tools_root\PUBLIC*language_code*\FILES
Web templates	tools_root\WEBTEMPL

- **Siebel Server.** The files in the Siebel Server installation directory are used by the Siebel Web engine when rendering the user interface in the Web Client. The locations of the files for a Siebel Server installation are shown in the following table.

Files	Directory Path
Image files	siebsrvr_root\webmaster\images*language_code*
Style Sheets	siebsrvr_root\webmaster\files*language_code*
Web templates	siebsrvr_root\WEBTEMPL

- **Siebel Mobile or Dedicated Web Client.** The files in the Mobile or Dedicated Web Client installation directories are used when rendering the user interface for these clients. The locations of these files are shown in the following table.

Files	Directory Path
Image files	client_root\PUBLIC*language_code*\IMAGES
Style Sheets	client_root\PUBLIC*language_code*\FILES
Web templates	client_root\WEBTEMPL

The Web Template Development Process

When modifying Siebel Web templates you typically work with the files stored on your local machine. After you have tested the changes locally, you copy them to the appropriate directory on the server to make them available to the Siebel Web Engine.

The typical development process for Siebel Web templates varies depending on whether the changes are simple or complex.

- **Simple Web template modifications.** For simple changes, such as moving the position of a control placeholder, it is easiest to make the changes to the Web templates in the Siebel Tools directory. Siebel Tools parses the file when you save it and identifies any syntax errors that may occur. Additionally, the Preview functionality displays the Web template and the objects that are mapped to it as they would be rendered in the user interface. For minor changes, it is easier to debug the files this way, rather than having to run the Mobile Web client and navigate to the applet where the changes would appear. After the change is verified using Siebel Tools, developers can manually copy the changed template into the Web template directory of their Siebel Mobile Web Client, launch the application, and do final testing.

  NOTE: The modifications presented in this chapter are relatively simple. The tasks are written assuming that you are using this development process.

- **Complex Web template development.** For complex template development, such as creating new Web templates or significantly modifying the layout of existing templates, the process must include creating a prototype in HTML, translating the various parts of the template into SWE syntax, and then debugging as described in the previous paragraph.

  For more information about SWE tags and syntax, read *Siebel Developer's Reference*.

Adding a New Logo to the Banner

The banner area is the top frame in the application frame set. It contains the application banner and the Powered by Siebel logo.

The banner frame has its own Siebel Web template file that you can modify. For example, you add your company's logo to the left side of the banner, as shown in Figure 34.

Figure 34. Banner Frame with Your Logo

In this example, assume that the NREC requirement is to add their corporate logo to the banner frame and change the color scheme of the frame to match their corporate Web site. To do this, you need to complete the following two tasks:

- "Modifying the Banner Frame Web Template" on page 161
- "Modifying the Banner Color Scheme" on page 162

Modifying the Banner Frame Web Template

The first step for adding a new logo to the Frame Banner is to add the image file to the Images directory and then modify the Web template to include the new image.

To add a logo to the banner frame

1. Add the image file you will use as your logo to the images directory of your Siebel Tools installation and to the Images directory of your test application.

   For example:

   - `tools_root\PUBLIC\`*`language_code`*`\IMAGES`
   - `client_root\PUBLIC\`*`language_code`*`\IMAGES`

2. In Siebel Tools choose View > Windows > Web Templates Window.

   The Web Template Explorer appears.

3. In the Web Template Explorer find dCCFrameBanner.

   The content of the template is displayed.

   dCCFrameBanner is the banner frame template for customer and partner applications.

   NOTE: Make a backup copy of the dCCFrameBanner.swt file.

4. Right-click in the window that displays the HTML, and then choose Edit Template.

   Your default HTML editor opens the dCCFrameBanner.swt file.

   NOTE: If you have not set your default HTML Editor, choose View > Options, click the Web Template Editor tab, and then define the path to the editor to use.

5. Find the following line of HTML:

```
<td class="applicationMenu"><span valign="top"><swe:image name="BANNER_GLOBE_LOGO"
category="HTML Control Icons"/></span></td>
```

6 Replace the file name “BANNER_GLOBE_LOGO” with the name of your logo file.

7 Find the following line:

```
<td width="100%" class="bannerFade"> </td>
```

8 Remove the class=”bannerfade” property as shown:

```
<td width="100%"> </td>
```

9 Save the file.

10 Copy the file dCCFrameBanner.swt from the Web Templates directory in your Tools installation directory to the Web Templates directory in your client installation.

11 Clear your browser’s cache directory and then open the Siebel application to be tested.

For the current example, open the Partner Portal application.

Changes appear similar to Figure 34 on page 161.

Modifying the Banner Color Scheme

The color schemes and other styles in Siebel Web templates are controlled using a cascading style sheet. To change the color scheme of the banner frame, you modify the rules defined in the cascading style sheet for your application.

In this example you work with the banner selector. The banner selector defines display rules for the banner frame. To modify the background color, you change the value of the Background-color property. For a list of selectors used in cascading style sheets, read *Configuring Siebel Business Applications*.

To modify the banner background color

1 Navigate to the client_root\public*language_code*\files directory and then open the file dCCmain.css.

The file, dCCmain.css, is the cascading style sheet for Siebel Customer and Partner Applications.

NOTE: The cascading style sheet for Employee applications is main.css.

2 Find the following selector and properties in the Banner Definitions section of the file:

```
.banner,
TD.banner TD A,
TD.banner TD A:visited,
TD.banner TD A:hover { background-color: #afc2ea; color: #003399; text-
decoration:none; font-weight:bold;}
```

3 Change the value of the background-color property and the color property as shown below:

```
.banner,
TD.banner TD A,
TD.banner TD A:visited,
TD.banner TD A:hover { background-color: #BDD3FF; color: #333399; text-
decoration:none; font-weight:bold;}
```

4 Find the following line:

```
.applicationMenu        {  background:#123783;  }
```

5 Change the value of the background-color property as shown below:

```
.applicationMenu        {  background:#BDD3FF;  }
```

6 Save the file.

7 Clear your browser's cache directory and then open the Siebel application to be tested.

For the current example, open the Partner Portal application.

The banner frame appears with the new background color, as shown in Figure 35.

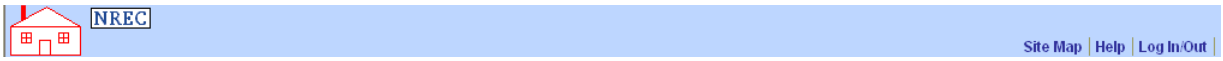


Figure 35. Banner Frame with Modified Color Scheme

Modifying the Screen Bar and View Bar Colors

The screenbar and viewbar are the frames that hold the screen tabs and the show drop-down list. For an example, see Figure 33 on page 158. These frames contain color elements, which are controlled using the cascading style sheet. Additionally, in the case of customer and partner applications, the background color of the screen bar frame is controlled using a graphic image.

The NREC requirement is to modify the background color for the screenbar and viewbar to match the background of the other user interface elements that were defined for the banner frame.

The image that controls the screenbar background is:

dccscrnbar_back.gif. Controls the background color of the screen bar frame (first-level navigation tabs.) This image is a tiling background used to make sure that the backgrounds display well in both Internet Explorer and Netscape browsers. It is only used for customer and partner applications. Open this image and modify color accordingly.

The selectors that control the viewbar background are:

- **Tier2On.** Controls the background color of the view bar frame (second-level navigation).
- **Tier2Rule.** Controls the background color of the view bar frame (second-level navigation).

To modify screenbar and viewbar style

1 Navigate to the client_root\PUBLIC*language_code*\FILES directory and then open the file dCCmain.css.

The dCCmain.css file is the cascading style sheet for Siebel Employee Applications.

2 Locate the tier2 selectors and then modify the value for the background color as shown below:

```
.tier2Back   {background-color:#BDD3FF; }

TD.tier2Rule   {background-color:#003366; }

TR.tier2On, TD.tier2On,
TD.tier2OnNormal
   font-size: 9pt;
   color:#000000;
   text-decoration:none;
   background-color: #BDD3FF;
   font-weight:bold;}
```

3 Save the dCCmain.css file, clear the browser cache, and then open the application.

The background color on the screenbar and viewbar reflects your changes, as shown in the following illustration.

Modifying Applet Colors

There are three major applet styles in your Siebel application: Parent, Child, and Grandchild. The applet styles define the color of the tab that appears at the top of the applet and the color of the control bar where applet-level menus and buttons appear. In the cascading style sheet, the selectors for each applet style are:

- **AppletStyle1.** The selector for parent applet style.
- **AppletStyle3.** The selector for child applet style.
- **AppletStyle7.** The selector for grandchild applet style.

Each applet style is defined by sub-elements To modify the appearance of each applet style, you modify the definitions for the following subelements.

- **AppletButtons.** The bar where applet buttons appear.
- **AppletBorder.** The border around the content of the applet.
- **AppletBack.** The background color of the applet content.
- **AppletTitle.** The area where the applet title appears.

To modify the applet color scheme

1 Open dCCmain.css.

2 Locate the following selectors for parent style applets and modify the attributes as shown in the following table.

Selector	CSS Attribute	Modified Value
AppletStyle1.AppletButtons	Background-color	#3366CC
AppletStyle1.AppletBorder	Background-color	#3366CC
AppletStyle1 .AppletTitle, AppletStyle1 TD.AppletTitle A, AppletStyle1 TD.AppletTitle A:visited, AppletStyle1 TD.AppletTitle A:hover	Background-color	#3366CC
	Color	#FFFFFF

3 Locate the following child applet selectors and modify the attributes as shown in the following table.

Selector	CSS Attribute	Modified Value
AppletStyle3 .AppletButtons	Background-color	#BDD3FF
AppletStyle3 .AppletBorder	Background-color	#BDD3FF
AppletStyle3 .AppletTitle, AppletStyle3 TD.AppletTitle A, AppletStyle3 TD.AppletTitle A:visited, AppletStyle3 TD.AppletTitle A:hover	Background-color	#BDD3FF

4 Locate the following grandchild applet selectors and modify the attributes as shown in the following table.

Selector	CSS Attribute	Modified Value
AppletStyle7 AppletButtons	Background-color	#BDD3FF
AppletStyle7 AppletBorder	Background-color	#BDD3FF
AppletStyle7 .AppletTitle, AppletStyle7 TD.AppletTitle A, AppletStyle7 TD.AppletTitle A:visited, AppletStyle7 TD.AppletTitle A:hover	Background-color	#BDD3FF
	Color	#003399

5 Save your changes and move the file to your test machine.

6 Clear your browser's cache directory and then open the Siebel application to be tested.

The background color scheme on parent and child applets reflects the changes.

Figure 31 on page 156 shows the modified applet color scheme.

Testing the Web Client Changes

NREC's testing plan calls for periodic integration and regression testing. These tests are necessary to make sure that the changes you have made up to this point do not prevent previously existing features from working, and that the application still can interact with other applications and products. NREC's test plan makes sure these checks happen at specific strategic intervals in the configuration process. In the event a test fails, NREC can quickly locate the faulty configuration change and make the appropriate modifications. If these tests are performed too infrequently, then troubleshooting errors may be a major effort.

11 Testing the NREC Deployment

NREC's test plan includes testing their entire development environment prior to porting to their production environment. Some of these tests need to be repeated in the production environment as well. The following topics describe these tests in more detail, as well as explain some of the automated testing options available.

Execute Integration Tests

Completion of the Functional Testing process verifies that the Siebel application functions correctly as a unit. In Integration Testing you verify that this unit functions correctly when inserted into the complete, larger environment. In this process, define your test cases to test the integration points between the Siebel application and other applications or components. Typical components include back-office applications, integration middleware, network infrastructure components, and security infrastructure. Tests in this process must focus on exercising integration logic, and validating end-to-end business processes that span multiple systems. Figure 36 illustrates this process.

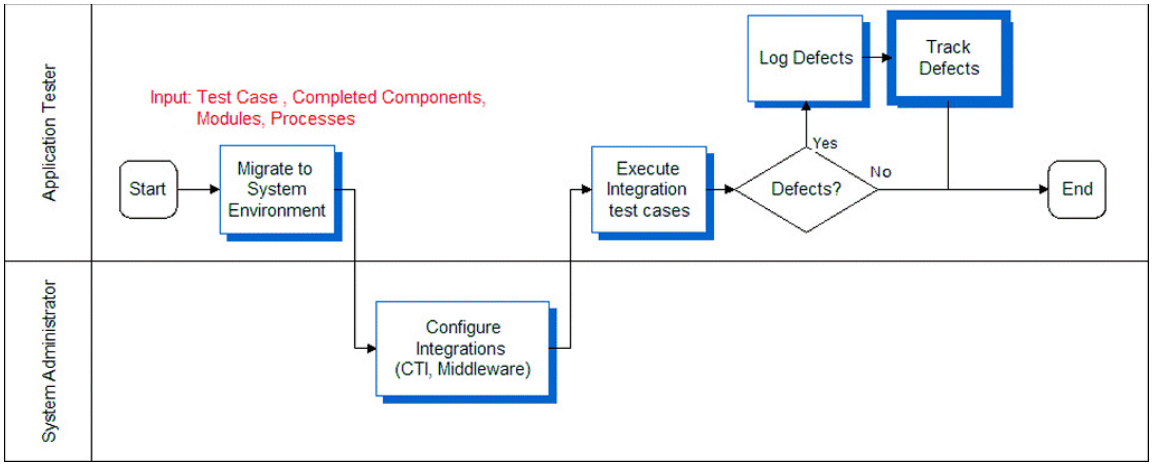


Figure 36. Execute Integration Tests Process

Execute User Acceptance Tests

When the application as a whole has been validated, you need to make sure that the functionality provided is acceptable to the business users. Hopefully, the business user has been engaged all along, approving at each phase of the project to make sure that there are no surprises. In the User Acceptance testing process, open the deployment up to a larger community of trained users and ask them to simulate running their business. Design User Acceptance testing to simulate live business as closely as possible. Complete this process by having the user community representative (business user) approve the acceptance test results. Figure 37 illustrates this process.

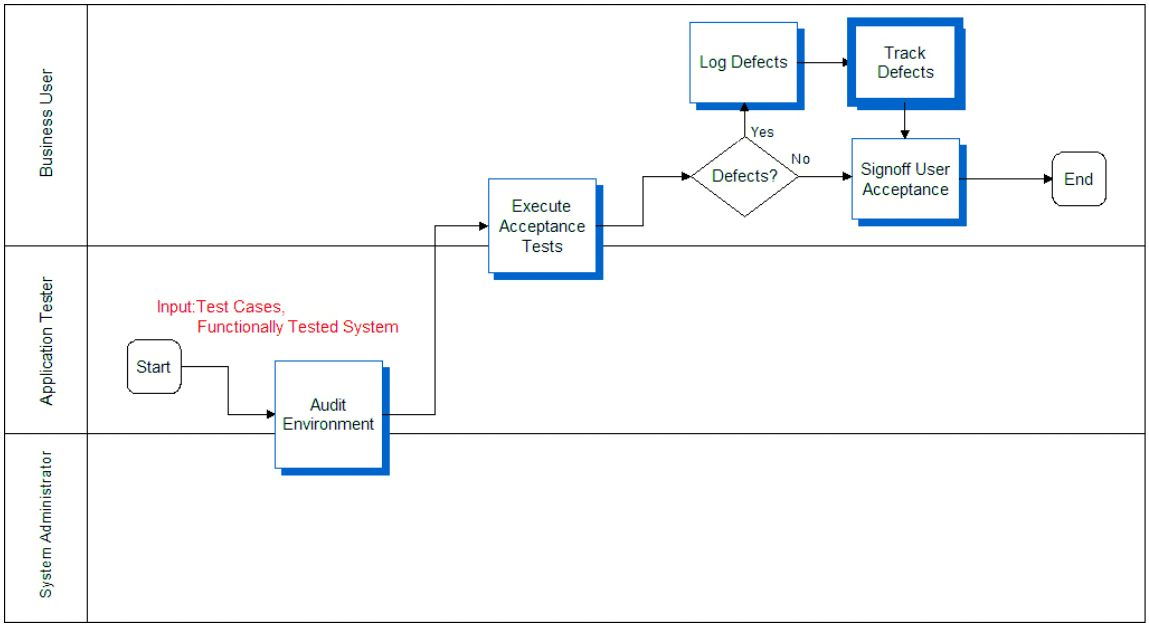


Figure 37. Diagram of the Execute Acceptance Tests Process

Executing Performance Tests

There are three types of performance test cases that are typically executed: response time, stress, and reliability testing. Each is intended to measure different key performance indicators. Performance tests are typically managed by specialized members of the testing and administration organizations, who have ownership of the architecture and infrastructure.

Figure 38 illustrates the process for performance test execution. The first step involves validating recorded user-type scripts in the system test environment.

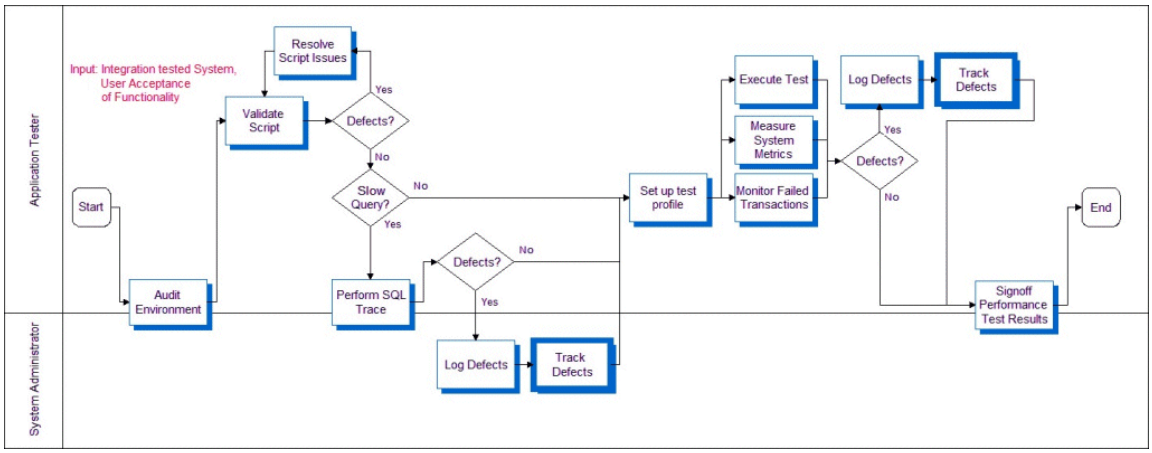


Figure 38. Diagram of the Execute Performance Tests Process

Execute each script for a single user to validate the health of the environment. Obtain a low user-load baseline before attempting the target user load. Use a baseline to measure scalability by comparing results between the baseline and target loads.

Users need to be started at a controlled rate to prevent excessive resource utilization due to large numbers of simultaneous logins. This rate depends on the total configured capacity of the deployment. For every 1000 users of configured capacity, add one user every three seconds. For example, if the product is configured for 5000 users, you add five users every three seconds.

Excessive login rate causes the application server tier to consume 100% CPU, and logins begin to fail. Randomize think times during load testing to prevent inaccuracies due to simulated users executing transactions simultaneously. Set randomization ranges based on determining the relative think times of expert and new users when compared to the average think times in the script.

Improve Testing

After the initial deployment, regular configuration changes are delivered in new releases. In addition, Oracle delivers regular maintenance and major software releases. Configuration changes and new software releases must be tested to verify that the quality of the system is sustained. This is a continuous effort using a phased deployment approach.

This ongoing life cycle of the application is an opportunity for continuous improvement in testing, as shown in Figure 39. First, a strategy for testing functionality across the life of the application is built by identifying a regression test suite. This test suite provides an abbreviated set of test cases that can be run with each delivery to identify any regression defects that may be introduced. The use of automation is particularly helpful for executing regression tests. By streamlining the regression test process, organizations are able to incorporate change into their applications at a much lower cost.

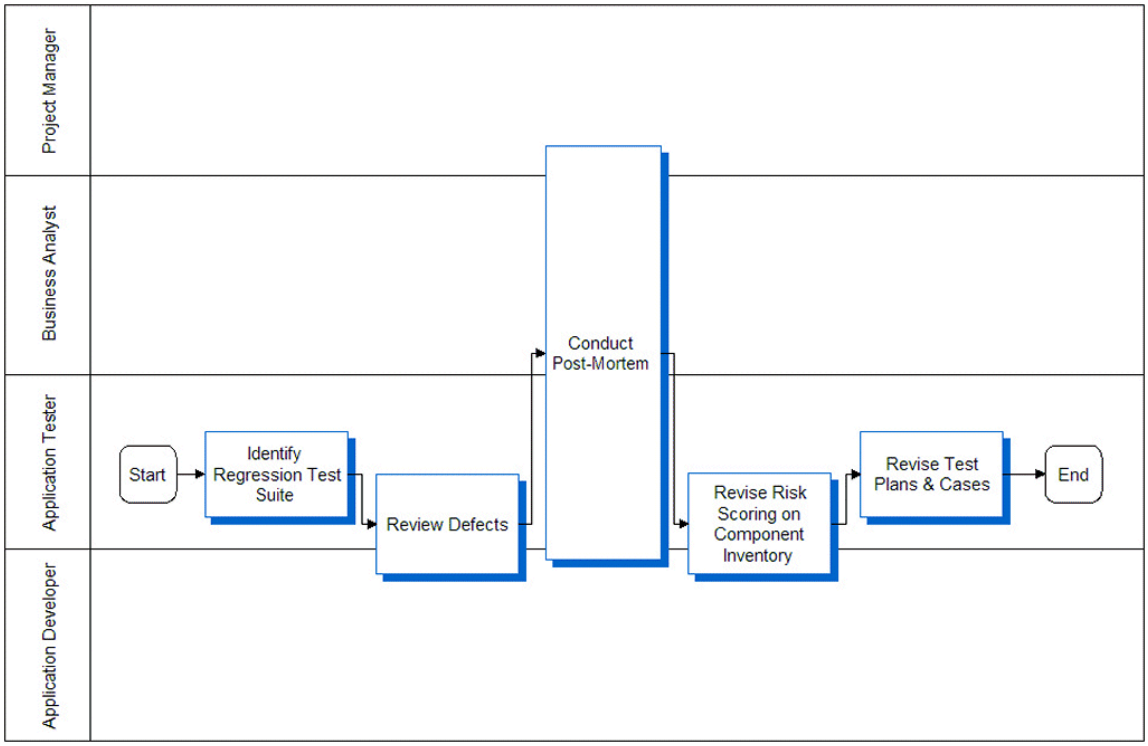


Figure 39. Diagram of the Improve Testing Process

Review the testing strategy and its objectives to identify any inadequacies in planning. A full review of the logged defects (both open and closed) can help calibrate the risk assessment performed earlier. This provides an opportunity to measure the observed risk of specific components (for example, which component introduced the largest number of defects). A project-level final review meeting (also called a post-mortem) provides an opportunity to have a discussion about what went well and what could have gone better with respect to testing. Review test plans and test cases to determine their effectiveness. Update test cases to include testing scenarios exposed during testing that were not previously identified.

Benefits of Functional Test Automation

Functional testing provides validation of the functional processes of your Siebel application. Without test automation, this can be accomplished by having users log on and manually perform the tasks in a business process. Although this can be effective, it can become quite costly.

By using a test automation tool, you can prepare a test script that exercises the functionality of a particular module or performs the tasks in a comprehensive business process. Then you can share and use this script repeatedly. This approach leads to better application configurations and increased user acceptance because it allows you to perform additional software testing with little additional cost. Test automation eliminates the need for multiple human test passes and reduces the risk of human errors in the testing process.

Key Features of Functional Test Tools

Tools for functional test automation, such as QuickTest Pro from Mercury Interactive, provide features (like those listed below) for creating and running functional test scripts:

- **Test script recording.** Rather than composing a script in a text editor, you start a recorder and perform a series of steps in the application. The test tool records the session in a script, which you can replay later to execute the test.
- **Automated test execution.** Rather than manually performing the tasks in the test script each time, you start the test using a test tool and let it run on its own.
- **Data value parameterization.** You can use variables in your test script that pull in data values from an external table during test execution. This avoids the need to hard-code data values into a test script.
- **Reusable tests.** Tests can be shared among multiple testers, and across multiple cycles of application testing.
- **Result tracking.** The test tool records important statistics for identifying functional errors.

Not all functional test automation tools have all of these features, and some have more. For more information on available features, refer to the documentation for your testing tool.

Using Siebel Test Automation for Functional Testing

Siebel Test Automation objects support the automation of functional tests on the following types of Siebel applications:

- Standard Web Client
- Mobile Web Client
- Dedicated Web Client

You can use a third-party test automation tool to write, record, and run test scripts on these types of applications. When using a third-party tool, refer to the documentation that accompanies the test tool for information about the specific features and functionality that it provides.

NOTE: The Siebel Test Automation framework does not provide generalized automation services; it is only supported for test automation.

Benefits of Load Test Automation

Load testing measures key performance indicators (KPIs), such as response time and reliability, of your Siebel application while it is under a *load* of multiple users. Without load test automation, this can be accomplished by having many users log on and run through a business process during a specified period of time. Although this can be effective, it can become quite costly.

By using a test automation tool, you can simulate the load and achieve more comprehensive and precise results without the cost associated with multiple human testers. In addition, a load testing automation tool reduces the risk associated with user errors.

Key Features of Load Test Tools

Tools for load test automation, such as LoadRunner from Mercury Interactive, provide features (like those listed below) for simulating a multiple-user environment:

- **Test script recording.** Rather than composing a script in a text editor, you start a recorder and perform a series of steps in the application. The test tool records the session, which you can use to perform the test.
- **Virtual users.** A single test script can be executed simultaneously by many virtual users from a single machine.
- **Controlled test execution.** Use the test tool to specify parameters that indicate how the load test is run, such as how quickly to add users, how many users to add, and which scripts to run.
- **Result tracking.** The test tool records important statistics, such as memory usage and response times, for tracking KPIs and isolating bottlenecks.

Not all load testing tools have all of these features, and some have more. For more information, refer to the documentation for your load testing tool.

Architectural Overview of Load Testing

The Siebel Correlation Library allows you to use a third-party tool to automate load testing on Siebel applications. This is a dynamically linked library (DLL) that provides services necessary to help the testing tool generate and execute test scripts against your Siebel application.

When the Correlation Library is installed and your test environment is properly set up, the Correlation Library helps the test tool translate the recorded test session into a script that can be executed to test your Siebel application. The Correlation Library essentially acts as an interpreter for the test tool. The test tool records the HTTP traffic from the Siebel application. (This HTTP traffic equates to a Web page.) Then the test tool sends the Web page to the Correlation Library. The Correlation Library parses the Web page and returns to the test tool the appropriate correlation information for the Web page. The correlation information provides for parameterization of data values.

For more information on load testing, such as how to use the Siebel Correlation Library, best practices, and troubleshooting, read the chapter on Automating Load Tests in *Testing Siebel Business Applications*.

In addition to the application data entities that are parameterized by the Correlation Library, you must also parameterize the transaction data entities. Application data entities are those that are associated with the function of the application, such as Row ID, SWE Timestamp, and SWE Count. Transaction data entities are those that contain record data entered into the application by users, such as Contact Name and Account Name. Transaction data entities must be parameterized manually, because they are not handled by the Correlation Library.

Setting Up Your Load Testing Environment

To set up your environment for automated load testing using Siebel Test Automation, perform the following steps.

- Install your load testing tool.
- Copy the Correlation Library (ssdtcorr.dll) to a location that is accessible to the load testing tool (for example, the BIN directory of the load testing tool). You can find the ssdtcorr.dll file in your Siebel Server installation (siebsrvr\bin on Windows, siebsrvr/lib on UNIX).

NOTE: The Correlation Library is licensed for use only by licensed users of Siebel Test Automation. If you have not purchased a license for Siebel Test Automation, you are not permitted to write to or otherwise access the Correlation Library.

In addition, set up your test tool so that it communicates properly with the Correlation Library. For more information, refer to the documentation from your test tool vendor.

12 Migrating to the Test Environment

This chapter tells you how to migrate your configuration from the development environment to the test environment. The data to be migrated includes:

- Repository data, such as new or modified applets, views, and business components. For more information, read "Migrating Repository Data from Development to Test."
- Modified Web templates and files such as cascading style sheets. For more information, read "Moving Modified Web Templates and Related Files" on page 177.
- Nonrepository data, such as assignment rules and LOVs. For more information, read "Migrating Nonrepository Data from Development to Test" on page 178.

Repository data and database schema changes are migrated using the Repository Migration Configuration Utility. Web templates and cascading style sheets are migrated by manually copying the files from the development server to the test server. Nonrepository data is migrated using the Application Deployment Manager.

Migrating Repository Data from Development to Test

This section explains how you migrate repository data from a source environment to a target environment. Following the NREC example, the source is the development environment and the target is the test environment.

You use the Repository Migration Configuration Utility to migrate data when there has been a configuration or a database schema change in the source environment, such as adding columns to a base table (read Chapter 7, "Configuring the House and Opportunity Entities"). The Repository Migration Configuration Utility exports the repository data from the source environment, imports the object definitions, and then updates the physical database schema of the target database. The result is that the database schema and object definitions of target environment matches source environment.

NOTE: If you only need to migrate user interface and business object definitions, you can use the Export and Import Repository Utilities.

For detailed information about managing repositories, read *Going Live with Siebel Business Applications*.

Preparing for the Repository Migration

Migration occurs between one server and another. Therefore, make sure the developers check in their projects from their local databases back to the server before the migration. After migration begins, developers must not make any more changes—such changes will be lost.

Before migrating the repository you also need to stop all server tasks on the target environment and disconnect database access until the migration is complete.

NOTE: If testing in the test environment reveals problems, these problems can be corrected either in the development environment (and the changes migrated again) or in the test environment.

Migrating the Repository

The Repository Migration Configuration Utility is a wizard that prompts you for the required information as you proceed through a succession of dialog boxes. You launch the Repository Migration Configuration Utility from the Siebel Database Configuration Utility. When the configuration session completes, the migration can be executed immediately or at a later time using the command line.

To migrate a repository

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 following succession of screens appears.

7. Progress by completing the information as shown in the following table, and then clicking Next.

Screen	Description
ODBC Data Source Name	Enter the ODBC data source name.
Database User Name	Enter the source database user name and source database password.

Screen	Description
Database Table Owner	Enter the source database table owner and source table owner password.
Source Database Repository Name	Enter the database repository name and the target database repository name.
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.
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)	Applicable only if you indicated that the target database is IBM DB2. Enter the index table space name and 4-KB Table Space Name.
16-KB Table Space Name (DB2-specific)	Applicable only if you indicated that the target database is IBM DB2. Enter the 16-KB table space name and 32-KB table space name.
Index Table Space Name (Oracle-specific)	Applicable only if you indicated that the target database is Oracle. There are two questions about index and table space. There are no 4-KB, 16-KB, and 32-KB table spaces in Oracle. For Microsoft SQL server, there are no screens about table spaces.
Configuration Parameter Review	Summary screen gives a list of your choices. You can accept this configuration by clicking finish.

Moving Modified Web Templates and Related Files

Every Siebel Server installation includes a set of Web templates (SWT files) and other related files (such as image files and cascading style sheets). If you make changes to any of these files in your source environment, you must copy the modified files to the target environment. For example, NREC needs to migrate the files modified as described in this chapter from the development environment to the test environment.

To move Web templates and related files from Development to Test

- Copy any new or modified files of the following types from the development server to the test server.

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

Migrating Nonrepository Data from Development to Test

Migrating application run-time customization data from one Siebel application environment to another is a common requirement when going live to a new development, test, or production environment. The Application Deployment Manager (ADM) feature automates the migration of the Siebel application run-time customization data.

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 eliminates the former process of transferring this data manually.

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 following data types are available for migration with ADM:

- Access Groups
- Assignment Rule
- Expense Types
- List of Value (LOV)
- Product Feature
- Product Line
- Public Predefined Query (PDQs)
- Responsibility
- State Model

- User List
- View

These data types are preconfigured in the standard Siebel Business application. Using Siebel Tools you can add more data types for use with ADM.

For more information on migrating nonrepository data and using ADM, read *Siebel Application Deployment Manager Guide*.

13 Using EIM to Load Data Into the Test Environment

You have already completed many of the tasks to prepare your test environment. For example, you have moved the application (including your configuration changes) from development to test and have performed the setup tasks described in Chapter 14, "Required Application Administration Tasks."

To do further development and testing work, however, you need to have actual corporate data in your Siebel database. Up until now, you have been working only with the sample data available with your Siebel installation.

This chapter tells you how to load data from your legacy applications into a Siebel database, using the Siebel Enterprise Integration Manager (EIM).

Generally, EIM is used to exchange data between Siebel database tables and other data sources. This exchange of data can include bulk imports, exports, merges, and deletes. Because the current goal is to load data into a Siebel database, this chapter deals only with bulk imports. For detailed information about EIM, read *Siebel Enterprise Integration Manager Administration Guide*.

This chapter focuses on one specific import example: loading account data for the NREC. The section "Import Example" on page 182, provides more detailed information about the specific import example presented in this chapter.

Basic EIM Concepts

To begin working with EIM, you need to understand some fundamental concepts, which are described in this section.

Interface Tables

As illustrated in Figure 40, *interface tables* are intermediate database tables between the Siebel database and another database.

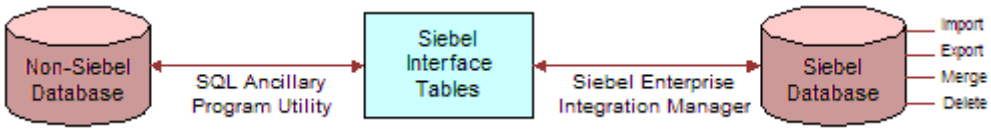


Figure 40. Process Flows Between Siebel Database and Other Databases

Rather than loading legacy data directly into Siebel base tables, you or your database administrator first load it into Siebel interface tables using your database-software-specific utility. Then, you use EIM to load data from the interface tables into the Siebel base tables.

Process Overview

For any EIM process (whether it be import, export, merge, or delete), you need to complete the following steps.

1. **Load the interface tables.** For more information, read "Using the Interface Tables" on page 182.
2. **Edit the EIM configuration file.** For more information, read "Editing the Configuration File" on page 185.
3. **Run EIM.** For more information, read "Running EIM" on page 187.
4. **Check results.** For more information, read "Checking the Results of the Data Import" on page 189.

This chapter follows these steps to import data.

Import Example

This chapter focuses on how NREC used EIM to import its account data into the Siebel database. To do this, the NREC administrator needed to load the applicable interface table (EIM_ACCOUNT), prepare the NRECimp.ifb configuration file, and run the EIM import process. This chapter provides the information for this example in the following order.

- Account data to be loaded into the EIM_ACCOUNT interface table

  For information about how to load an interface table, including finding out which columns are mandatory, read "Using the Interface Tables" on page 182.

- Configuration file

  To view the contents of the IFB file that used to customize the behavior of the EIM process, read "A Sample Configuration File" on page 185. There is also a brief explanation of the parameters that appear in the configuration file.

- EIM process

  For the step-by-step instructions on running the EIM process, read "To run an EIM process using the GUI" on page 187. The process pulls the accounts data from the EIM_ACCOUNT interface table into the specified base tables in the database.

Using the Interface Tables

Read *Siebel Enterprise Integration Manager Administration Guide* for information on using these tables, including:

- Specific data and file attachments that EIM can process
- The names of the interface tables
- The target base tables mapped to the interface tables
- Any secondary tables associated with the target tables

Determining Which Columns Are Required

To some extent, you can decide which columns from your source table to load into an interface table. However, for some columns interface tables are required: they cannot be left NULL.

You can find out which columns are required either through Siebel Tools or by referring to *Siebel Enterprise Integration Manager Administration Guide*.

To use Siebel Tools to determine which columns are required

1. In the Object Explorer, expand the EIM Interface Table object.

   The EIM Interface Tables window appears in the Object List Editor.

   NOTE: If the EIM Interface Tables object type is not visible, choose View > Options, and then under the Object Explorer tab, activate the EIM Interface Tables object types.

2. In the EIM Tables window, find the interface table you plan to load.

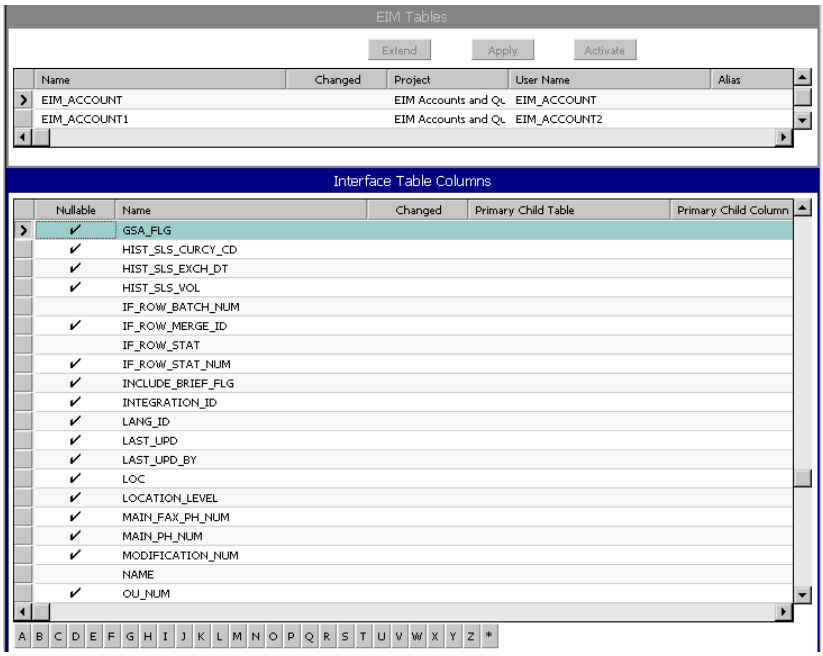
   For the current example, find EIM_ACCOUNT.

3. In the Object Explorer, select EIM Interface Table Column (a child object of EIM Interface Table).

   The EIM Interface Table Columns window appears in which the columns of the current table are displayed.

4 In the Interface Table Columns window, look at which columns have the Nullable property checked.

The columns marked Nullable are *not* required. For example, as shown in the following figure, the IF_ROW_BATCH_NUM, IF_ROW_STAT, and NAME columns are required because they are not marked Nullable.



NOTE: For your convenience, right-click and choose Columns Displayed to change the column order on this view so that the Nullable property appears next to the Name property, as shown in the preceding figure.

Loading the Interface Tables

To load data into an interface table, use your database-software-specific load utility, such as one of the load utilities available for DB2. For information, read the documentation for your database software.

Make sure you load the required columns.

Some columns in the interface tables are part of a *user key sequence*. The user key sequence is the set of columns (called user key columns) whose values uniquely identify a row. As EIM loads each row, it checks whether the interface table already contains that row's user key sequence, thus preventing the import of duplicate rows.

Editing the Configuration File

EIM reads a special configuration file that specifies the EIM process to perform and the appropriate parameters. This example features the import process.

The EIM configuration file is an ASCII text file of extension type IFB that resides in the admin directory under the Siebel Server directory. Before initiating an EIM process, you must edit the contents of the EIM configuration file to define the process to be performed.

NOTE: In a Unicode environment, the EIM configuration file must be saved as a Unicode text file. For more information, read *Siebel Enterprise Integration Manager Administration Guide*.

In this example, EIM sets the process locale as specified in the configuration file at start up. The default configuration file, default.ifb, provides the data used in the process if no other configuration file is specified. This example uses the configuration file NRECimp.ifb.

The configuration file begins with a header section used to specify global parameters that apply to the process sections defined later in the file. Following the header section, there must be at least one process section with its associated parameters. Some process section parameters are generic to all EIM processes. Other process section parameters are specific to a particular EIM process. The following parameter sections describe the required header section and generic process section parameters.

Read the *Siebel Enterprise Integration Manager Administration Guide* for the following topics.

- Other process and header parameters
- Alternative sources from which EIM accepts parameter values
- The process for defining multiple processes in the configuration file

A Sample Configuration File

The sample configuration file in this section shows the entries that are necessary only for the operations in the particular import EIM example for this chapter. For more information about the EIM configuration file, read *Siebel Enterprise Integration Manager Administration Guide*.

The text of the sample configuration file, NRECimp.ifb, is shown here.

```
/* These are header parameters. */
[Siebel Interface Manager]
   LOGIN USER = "SADMIN"
   LOGIN PASSWORD = "SADMIN"
   RUN PROCESS = Import Accounts

/* These are generic process parameters. */
[Import Accounts]
   TYPE = IMPORT
   TABLE = EIM_ACCOUNT
   ONLY BASE TABLES  = S_ORG_EXT, S_ADDR_ORG, S_PARTY

   ONLY BASE COLUMNS = S_ORG_EXT.NAME, S_ORG_EXT.LOC, S_ADDR_ORG.ADDR, S_ADDR_ORG.OU_ID,
S_ADDR_ORG.CITY, S_ADDR_ORG.STATE, S_ADDR_ORG.ZIPCODE, S_PARTY.PARTY_UID,
```

```
S_PARTY.PARTY_TYPE_CD

/* There are no process-specific parameters required in this example. */
```

Header Parameter Section

This section describes the header parameters that you need to specify in the configuration file to properly configure EIM for an import process. For the list of the values assigned to each of the header parameters, read "A Sample Configuration File" on page 185. Table 11 shows only those header parameters featured in the NRECimp.ifb.

Table 11. Required Header Section Parameters

Command	Description
USERNAME	Database employee login (SADMIN for NREC)
PASSWORD	Database password (SADMIN for NREC)
PROCESS	Initial or main process section to run (Import Accounts for NREC)
[Siebel Interface Manager]	Header section must use this reserved name

Process Parameter Section

A parameter in the process section only applies to the particular process specified and overrides any corresponding value in the header section for the specific process. There are both generic and specific process parameters.

Table 12 describes the parameters that NREC used in the process section of NREC's configuration file, NRECimp.ifb. The process section is generic to all EIM processes. For parameters specific to the import process, read the *Siebel Enterprise Integration Manager Administration Guide*.

Table 12. Process Section Parameters Generic to All EIM Processes

Command	Description
ONLY BASE COLUMNS	Only process these columns. S_ORG_EXT.NAME, S_ORG_EXT.LOC, S_ADDR_ORG.ADDR, S_ADDR_ORG.OU_ID, S_ADDR_ORG.CITY, S_ADDR_ORG.STATE, S_ADDR_ORG.ZIPCODE for NREC.
ONLY BASE TABLES	Only process these tables. S_ORG_EXT, S_ADDR_ORG, S_PARTY for NREC.
TABLE	Interface (IF) table for process. EIM_ACCOUNT for NREC.
TYPE	IMPORT,EXPORT,DELETE,MERGE,SHELL. IMPORT for NREC's import example.

Disabling Logging Before Initial Loads

Before performing an initial load through EIM, it is a good idea to disable transaction logging to improve EIM performance. Disabling transaction logging consists of disabling the Docking: Transaction Logging system preference.

CAUTION: Do not disable Docking Transaction Logging if you have active mobile clients. Otherwise, the server database and mobile client databases will not be synchronized after the import. To synchronize them again, you need to again extract the database for each mobile client, and run database initialization on each client. For more information on synchronization, read Chapter 18, "Implementing Siebel Remote," and *Siebel Remote and Replication Manager Administration Guide*.

To disable transaction logging

1. Navigate to the System Preferences screen.
2. Select Docking: Transaction Logging.
3. In the System Preference Value field, type FALSE.

   Do not change this value to TRUE until after you import the initial data.
4. Click outside the row to save changes.

You can also change the transaction logging preference by changing the LOG TRANSACTIONS parameter in the EIM configuration file.

Running EIM

This section describes how to start an import process from the Siebel Client. For information about starting the import process from the command line, read *Siebel Enterprise Integration Manager Administration Guide*.

To run an EIM process using the GUI

1. Navigate to Administration - Server Configuration > Enterprises > Parameters.
2. Click the Component Requests view tab.
3. In the Component Requests form, add a new record.
4. In the Component/Job field, click the select button, and in the Component/Jobs window, select the Enterprise Integration Mgr component.

   To use a component job based on EIM for your component request, first define the component job. For information on defining component jobs, read *Siebel System Administration Guide*.
5. Complete the remaining fields in the Component Requests form and save the record.
6. In the Component Request Parameters list, add or change any component parameters for the EIM process and save the record.

Figure 41 on page 189 shows an example of setting parameters in the Component Request Parameters list.

a In the Component Request Parameters list, add a new record.

b In the Name field, click the Select button.

c In the Job Parameters window, select an item from the list and click OK.

d In the Value field, type the appropriate value for that list selection and click Save.

e Continue to make other selections in the Job Parameters window by repeating Step a through Step d for the required and optional selections listed in the following table.

f You need to identify at least a batch number, process name, and configuration file for the task. The possible selections, values required, and default values appear in the following table.

Selection	Required or Optional	Value	Default	Comments
Configuration file	Required	Name of the EIM configuration file	default.ifb	You can use the Uniform Naming Convention (UNC) filename when specifying the EIM configuration file if you have read permission to the path.
Batch number	Required	Batch number	0	If the batch number component parameter is set to 0, the batch number in the EIM configuration file (if any) is used.
Process	Required	Process name of the EIM process you want to run	not applicable	The initial process to be run, defined in the EIM configuration file.
Error Flags	Optional	1	0	
SQL Trace Flags	Optional	8	0	
Trace Flags	Optional	1	not applicable	

7 In the Component Requests form, click the menu button, and then click Submit Request.

CAUTION: EIM is a multistep process. When the EIM process is running, do not stop or pause the task. Otherwise, some steps may not roll back correctly.

Figure 41 shows an example of running an EIM process as described.

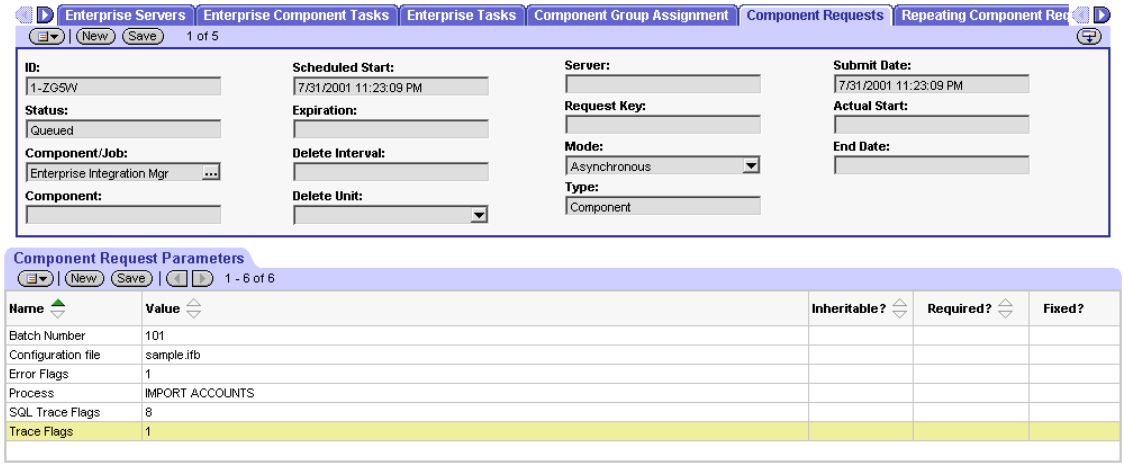


Figure 41. Running an EIM Process

To reactivate Docking Transaction Logging

1. Navigate to the System Preferences screen.
2. Select Docking: Transaction Logging.
3. In the System Preference Value field, type TRUE.
4. Click outside the row to save changes.

Checking the Results of the Data Import

Your EIM log file contains comprehensive status and diagnostic information about the import process. NREC's EIM log files are located in the *siebel_srvr_root*\log\server directory, the default.

To review your import process

- View the EIM log file for the corresponding task number.

  The log file name is Eim_*task#*.trc, where *task#* is the Task Number for the EIM process that you ran.

In addition, you can check the status of each row EIM tried to import by using the IF_ROW_STAT column. For more information, read *Siebel Enterprise Integration Manager Administration Guide*.

14 Required Application Administration Tasks

This chapter describes how NREC completed the general setup tasks in the Test environment. This chapter assumes a successful installation of Siebel products in the Test environment. For an overview of NREC's installation process, read Chapter 3, "Installing Siebel Applications."

The tasks in this chapter are performed in the administration views of the Siebel Web client or dedicated Web client. Assume NREC accessed the views by logging on to the Partner Manger application. The same views are accessible through other employee applications, such as Call Center. Although the tasks in this chapter are written from the perspective of an NREC employee setting up the application in the test environment, you can follow along using an application logged on to the Sample database.

This chapter includes the following topics:

NOTE: Generally, always perform administration tasks against the server database. Although you can perform these tasks against your local database and synchronize, there is the potential for error, including data conflicts, performance problems resulting from a large local database, and large numbers of transactions to route.

The tasks in this chapter do not cover Access Groups and Catalogs, which control access to master data. Because NREC did not use Access Groups or Catalogs in their implementation, the tasks for setting them up are not covered in this chapter. For more information about setting up Access Groups and Catalogs, read *Siebel Applications Administration Guide* and *Siebel Security Guide*.

Logging On as the Siebel Administrator

To perform the tasks in this chapter, you must log in to your Siebel application using the *administrator* user name and password assigned by your database administrator. The default values are SADMIN and SADMIN. You can use SADMIN and SADMIN to log in to the Siebel Sample database to complete the tasks in this chapter.

To log in as an administrator to a server data source

1. Open your Web browser.
2. Go to the URL for your application.

   The login screen appears.

3 Enter SADMIN in the User ID and Password fields.

4 Click OK.

Logging in as SADMIN provides access to all screens and views mentioned in this guide.

To log in as an administrator to the sample data source

1 Open the Mobile Web Client on your local machine.

For the NREC example, log in to Siebel Partner Manager.

2 Enter SADMIN and SADMIN as the user name and password.

3 Select Sample as the data source.

Siebel Partner Manager opens, displaying data from the Sample database.

NOTE: If you modified tabs as described in Chapter 10, "Modifying the Look and Feel of the Web Client," the images for the off state may appear in your employee application. To return the user interface to the original look and feel, you need to replace the original images. For more information about tab images, read "Modifying the Screen Bar and View Bar Colors" on page 163.

Defining Views

Often developers add views during the configuration stage of the implementation process. For the new views to appear in the user interface you need to register them in the application. You did this during the development cycle, to unit test the new views (read "Testing Changes" on page 50). However, you registered the views in your local development environment; now you need to register them in the test environment.

Use Administration - Application > Views to register new views. You must do this before you can associate views with responsibilities.

NOTE: Turn off any of the forecast views that you do not intend to use based on your organization's forecasting configuration decision.

To register new views

1 From the application-level menu, choose Site Map > Administration - Application > Views.

2 Add records for the new views.

For example, new views for the NREC example, are:

- House Detail View
- House Detail - Appraisals
- House Detail - Renovations

NOTE: You generally do not need to modify or delete records for registered views. A view record requires modification only if its name is changed. The only time a view may need to be deleted is if it no longer existed or if you did not want anyone to have access to it. Even then, a preferable approach is to not assign it to any responsibilities.

Defining Company Structure

The first step in setting up your Siebel application is to define your company's structure. This includes defining organizations, divisions, positions, and responsibilities.

NOTE: Changing your company structure—such as positions and divisions—can cause routers to reevaluate visibility for every object related to the objects that have changed. This can affect performance. For more information, read *Siebel Remote and Replication Manager Administration Guide.*

Understanding Company Structure

Defining your company's structure involves setting up the following entities:

- **Organizations and organization skills.** Organizations represent the broad divisions of your company. They can represent internal or external groups of users—for example, internal divisions or partners. Organizations are used to control access to data.
- **Divisions.** Divisions belong to organizations but do not affect visibility. Divisions are used to group positions, record addresses, and maintain default currencies.
- **Positions and position skills.** Positions represent specific job slots. Positions determine which record users can access.
- **Responsibilities.** Responsibilities control the screens and views a user can access.

For more information about setting up company structures, read *Siebel Applications Administration Guide*.

Understanding Access to Data

How you define your company's structure in your Siebel application affects the records and views to which users have visibility, or access.

There are several relationships between organizations, responsibilities, positions, and people.

There is a many-to-many relationship between a person and that person's positions. For example, an individual could hold one or more positions—sales representative for the X territory definition and sales manager for the XYZ region. Conversely, more than one person may be assigned to a single position, such as when two people share a single job.

In addition to having one or more positions, each user is then assigned one or more responsibilities. Responsibilities determine which views users can access; positions determine which records users can access.

In a situation in which a user is associated with one responsibility and multiple positions, that user has the same set of views regardless of which position that user logged on with. However, if the user changed position, that user would access different data. A user who is logged on as a sales representative can access only data (accounts, opportunities, contacts, and so on) that pertains to this position. Even though the user still has the My Team's Accounts view, the position the user is logged on as does not have any other positions reporting to it so the user does not get other data.

NOTE: A user can change positions while logged in by choosing View > User Preferences > Change Position and choosing a different position in the list.

If the user changes to the sales manager position, the My Team's Accounts view now shows the user information for the positions reporting to that user if the user is the primary on the account. Again, the views have not changed, but the data in the views has.

Having organizations adds another level of visibility. Some data records are unavailable depending on what organization a user's position belongs to. (Users can also be given more visibility within their organization by receiving access to certain views.)

For more information on organizations and access to data, read *Siebel Applications Administration Guide* and *Siebel Security Guide*.

Setting Up Organizations and Organization Skills

Setting up organizations is an optional step in an implementation. If you do not set up additional organizations, you automatically have everything assigned to a default organization. Having one organization is like having no organization. There is no effect on visibility and data access.

Because NREC is using Partner Portal, it sets up one parent organization called NREC and then a child organization for each of its partner real estate agencies. This structure lets NREC to logically group these different user groups and partition data accordingly.

Initially NREC creates the NREC organization. Then after the rest of the company structure is set up—positions, responsibilities, and so on—NREC uses Siebel Partner Manager to set up partners and promote them to organizations with NREC as the parent organization. Read "Registering Partners and Creating Partner Organizations" on page 202.

For more information on organizations and company structure, read *Siebel Applications Administration Guide*.

To set up an organization

1 From the application-level menu, choose Site Map > Administration - Group > Organizations.

The Organizations view appears.

2 Add a new record.

Some fields are listed in the following table.

Field	Value	Description
Name	NREC Organization	Unique name for the organization. Using the word *organization* in the name provides a visual clue for users.
Partner Manager Position	SADMIN	Used for Partner Portal to represent the person in the organization who manages the relationship with partners. For more information, read *Siebel Partner Relationship Management Administration Guide*.
Partner Flag	FALSE	Indicates whether the organization represents an external enterprise.

NOTE: Organizations cannot be deleted.

Setting Up Divisions

Divisions belong to organizations and are used to record addresses and to maintain default currencies. User reporting structures are defined by their parent position, but their country of operation and currency are defined by their division. To implement Siebel Business Applications, you must have at least one division set up.

A division is automatically created when you create an organization. For example, the NREC division was created when you created the NREC Organization.

To review the division created for your organization

1 From the application-level menu, choose Site Map > Administration - Group > Internal Divisions.

The Internal Divisions list appears.

2 In the Internal Divisions list, query for the organization you created in "Setting Up Organizations and Organization Skills" on page 194.

For example, query for NREC.

3 Review the values defined for the division.

Setting Up Positions and Position Skills

To log in to the Siebel application an employee has to have a position. Positions determine which records users with a particular position can access. Positions represent a job slot in your organization. As you enter your company's positions, refer to your company's organization chart to determine reporting relationships (positions and parent positions). There is always one position that does not have a parent position. For example, the CEO position probably does not report to a higher level. Define positions in each level of your company's division hierarchy.

Because you choose parent positions as you create new positions, start at the top of the organization chart and work your way down.

To create positions

1 From the application-level menu, choose Site Map > Administration - Group > Positions.

The Positions view appears.

2 In the Positions list, add a new record for each of the positions to define.

Some fields are described in the following table.

Most fields in the Position list are filled in automatically from the Employee record of the active employee. If you have not set up employees, you can associate them with positions later.

Field	Comments
Billing Product	Used for Professional Services. For more information, read *Siebel Applications Administration Guide*.
Compensable	Used for incentive compensation.
Organization	Select an organization for the position. A position can only have one organization. If you want a user to have visibility to some organizations, but not all organizations, then you must create a position for each organization and assign the employee to each position. The employee can then get one organization's data at a time, by choosing View > User Preferences > Change Position.
Position	The name of the position. Required.
Territory	Allows a position to be associated to a territory for use by the Assignment Manager module. For more information, read *Siebel Assignment Manager Administration Guide*.
Position Type	Informational field that indicates the type of position. It has no effect on visibility.

The following table lists sample positions for the NREC example. Later, as described in the section "Setting Up an Employee in Your Siebel Application" on page 200, you add the employees listed in the last column.

Position	Parent Position	Employee
Administrator	Information Technology Manager	Sandy Bolivar
CEO	not applicable	Pat Bosch
Information Technology Manager	CEO	Chris Brown
Marketing Associate	Sales Manager	Terry Picardo
Partner Manager	Sales Manager	Dale Abizaid
Real Estate Agent 1	Sales Manager	Dana Castro
Real Estate Agent 2	Sales Manager	Alex Grey
Sales Manager	CEO	Lee Smith

Most fields in the Position applet get filled in automatically from the Employee record of the active employee. For example, the Start Date field shows the start date of the employee marked Active for the position.

To set up position skills

1. From the application-level menu, choose Site Map > Administration - Group > Positions.

   The Positions view appears.

2. In the Positions list, click the position to which to add skills.
3. Click the Assignment Skills tab.
4. In the Assignment Skills list, add a new record.
5. Select a skill in the Item field.

   Skills must exist before you can add them. Skills are added using Siebel Tools.

6. Add comments, if desired.
7. Save the record.

Defining Responsibilities

Responsibilities determine which views users can access. For example, the Siebel Administrator responsibility allows access to all views. Defining responsibilities lets you limit user access to views and, therefore, to your Siebel application's information and functions.

Define responsibilities that correspond to the major job functions in your organization. For example, you might create responsibilities for NREC CEO, sales manager, partner manager, field sales representatives, and so on.

- In the NREC example The Real Estate Agent responsibility might have a select set of views in Siebel Sales, such as Opportunities, Contacts, Accounts, Activities, and so on.
- The sales manager responsibility might have access to the same screens as the sales representative, plus additional views.
- The CEO responsibility might have access to all views except applications administration.

To define a responsibility, you must specify which views are available to that responsibility. You may prefer to copy the sample responsibilities that ship with your Siebel application and then customize them for your purposes.

NOTE: You cannot edit sample responsibilities.

Grant access to the System Preferences view to only a select group of administrators. System preferences control server logic and processing. Therefore, do not give end users access to the System Preferences view.

Do not add Administration - Applications screens to responsibilities associated with end users. Likewise, limit access to the Employees, Master Forecasts, Mobile Clients, Responsibilities, Views, and Territories screens. The work performed using these screens has far-reaching implications for the entire application.

You may want to hide the license key view to discourage unauthorized users from attempting to change license keys. You can do this using the License Keys view in the Administration - Applications screen. To hide the license key view, disable the Siebel License Key view in the user's responsibility.

NREC created new views during the development cycle and then registered the views in the application. Now NREC must add these views to the responsibilities that need to have access to them.

NOTE: Add all new views to the administrator responsibility, because the people with this responsibility typically help troubleshoot the Siebel application.

To copy an existing responsibility

1. From the application-level menu, choose Site Map > Administration - Application > Responsibilities.

   The Responsibility view appears.

2. Select a responsibility to copy.

   NOTE: You cannot edit the sample responsibilities. You must copy them before you can modify them.

3. Click the menu button and then choose Copy Record.

   The views for the existing responsibility are copied, but the users are not.

4. Add any other desired views to the responsibility.

   For the NREC example, add the following views.

   - House Detail View
   - House Detail - Appraisals
   - House Detail - Renovations

5. Enter a new name for the responsibility, such as NREC Partner Real Estate Agent.
6. Enter a new description for the responsibility so that you can later identify its function.
7. Select an organization for the responsibility.

   By default, the Responsibility view shows all responsibilities, regardless of organization. However, you may want to configure new views in Siebel Tools that restrict the visibility to responsibilities as a way to delegate administration of responsibilities. For more information on configuring views, read *Configuring Siebel Business Applications*.

Now you must set up your users, so they can be assigned responsibilities. Without responsibilities, a user cannot use the Siebel application.

Setting Up Users

Before users can access Siebel data, you must define users in your database server, your Siebel application, and your authentication environment. Your database administrator creates database accounts and optionally encrypts their passwords. Depending on your user authentication environment, each user may have an individual database account, or accounts may be shared by multiple users. If you implement an external authentication application, you must configure user authentication including associating users with database accounts.

For the current example, assume NREC is setting up each user with an individual database account and is not using an external authentication application. For more information about setting up user authentication, read *Siebel Security Guide*. For more information about setting up users, read *Siebel Applications Administration Guide*.

Setting Up Database Users

Before you can set up someone as a Siebel application user, that person must have a database account user name.

Add the database accounts on the appropriate database and add these accounts to the group SSE_ROLE.

The exact steps for adding users and placing them in this role group depend on the database software your organization is using. Work with your database administrator to set up developers as database users in the SSE_ROLE group.

Types of Users

Siebel Business Applications recognize several types of users:

- **Employees.** Individual internal users. Employees typically have login privileges.
- **Partners.** Individual users in partner companies to whom your company may grant access to some of its data. To access views a partner needs a position. For more information, read "Registering Partners and Creating Partner Organizations" on page 202.
- **Users.** Internal and external users with login privileges. Customers for customer applications to whom your company may grant some data access. A user does not need a position to access views.
- **Persons.** Includes employees, partners, users, and contacts. Persons may or may not have login privileges.

Setting Up an Employee in Your Siebel Application

After users have a database account, you can set up these database users as Siebel application users.

As the Siebel application administrator, you are able to:

- Add or delete a user
- Add multiple positions, responsibilities, organizations, and territories to a user
- Remove all positions connected to a user

This task can be done by an administrator in your organization or an administrator at a partner organization whose responsibility allows them access to the view.

Initially, you want to set up employee records. However, you also need to set up partners, users, and persons. The procedures for setting up other types of users are similar to the procedures for setting up employees.

To add an employee

1 From the application-level menu, choose Site Map > Administration - User > Employees.

The Employees view appears.

2 Add a new record.

3 Select at least one responsibility. You can also add organizations and positions.

4 Complete the additional fields.

Some fields are described in the following table.

Field	Comments
User ID	Enter the user ID for this employee in uppercase letters for compatibility across databases.
Password	Applicable only when using LDAP or Microsoft Active Directory for login authentication. Otherwise, it is not editable. For more information about parameter setups to make the field editable when using LDAP or Microsoft Active Directory, read *Siebel Security Guide*.
Position	The user's position. Positions and their employees are defined for NREC in the section "Setting Up Positions and Position Skills" on page 196.
Responsibility	The user's responsibilities. Associating responsibilities with users is described in the section "Associating Responsibilities with a User."

5 Repeat these steps for each employee you need to add.

Associating Responsibilities with a User

When you are first implementing your Siebel application, you define responsibilities, as described in "Defining Responsibilities" on page 198. After you have defined responsibilities, you assign them to users as described in "Setting Up Users" on page 199. Using this process you can assign user responsibilities at the same time as you enter other user information.

After your initial implementation, you may work directly from the Responsibility view, or you may continue to work from the Employee view to change or add associations between users and their responsibilities.

To associate a user with a responsibility

1. From the application-level menu, choose Site Map > Administration - Application > Responsibilities.

   The Responsibility view appears.

2. Select a responsibility to assign to a user.
3. Select the User list applet.
4. Click the menu button and then choose Edit > Add New Record.
5. Select one or more employees from the Add Employees dialog box, and then click Add.

   The added employees now have access to the views listed in the Views applet.

Registering Partners and Creating Partner Organizations

This section covers how NREC sets up partners in their test environment to resemble its business model. NREC will create an organization for each of its partner real estate agencies. Additionally, NREC will set up the company structure for each partner, including positions, responsibilities, and employees. This section uses one partner, San Francisco Real Estate, as an example.

For detailed information on managing partners, read *Siebel Partner Relationship Management Administration Guide*.

Registering Partners and Promoting Them to Organizations

NREC registers each partner and then converts it to an organization within the NREC company structure.

To register a partner and then promote it to an organization

1. From the application-level menu, choose Site Map > Administration - Partner > Approved Partners.

   The Approved Partners view appears.

2. In the Approved Partners list, click New to enter a new record representing the partner. For example, enter San Francisco Real Estate.

   A Partner form appears.

3. Save the record.

   The record is displayed in the Partners list.

4 Click the Register button.

A form appears where you can select the Organization flag, which automatically promotes the partner to an organization.

5 Select the Organization flag and enter NREC as the parent organization.

The partner is moved to the Registered partners list and an organization is created for the partner as a child of the parent organization—NREC.

Creating Positions, Responsibilities, and User Assignments

Now that you have created a registered partner, you can add the positions, responsibilities, and user assignments for the partner.

To create positions for a registered partner

1 From the Show drop-down list, choose Registered Partners.

2 In the Partners form, query for the partner you created in the previous procedure.

For example, San Francisco Real Estate.

3 Enter new records using the following views:

- **Positions.** Enter positions for the partner organization.
- **Responsibilities.** Enter responsibilities for the partner organization.
- **User Assignments.** Enter employee records for the partner organization.

15 Assignment Manager

In previous chapters, you used Siebel Tools to configure the business logic of NREC's Partner Portal application. In addition to Siebel Tools, there are other mechanisms t to configure business logic as well. These include modules such as Siebel Business Process Designer, Siebel Personalization, and Assignment Manager. These modules are administered at run time using views in the Siebel Web Client. You can modify business logic without having to recompile the Siebel SRF file. This chapter describes how NREC used Assignment Manager to implement business rules. Subsequent chapters cover Business Process Designer and Personalization.

Use Assignment Manager to create business rules that automatically assign entities such as opportunities, service requests, or activities to the appropriate individuals. For example, NREC wants to automatically assign opportunities to employees at partner real estate agencies. When an opportunity is entered into the application, it contains the ZIP Code where the customer is interested in buying property. Assignment Manager uses this attribute to automatically assign the opportunity to a real estate agent located within that ZIP Code. After assignment rules are defined and made active, they are automatically triggered whenever a new opportunity is created.

This chapter includes the following topics:

Configuring Territory Assignment Components

NREC's assignment rules use the ZIP Code attribute on the opportunity to determine how to assign the opportunity. This is a custom attribute that you added to the opportunity business component in "Adding Fields to the Opportunity Business Component" on page 112.

Because this is a custom attribute, NREC needs to add a new assignment criteria using Siebel Tools before being able to create assignment rules. This section summarizes the high-level tasks necessary to create the assignment criteria. For detailed information about configuring assignment attributes, read *Siebel Assignment Manager Administration Guide*.

The tasks to create the Opportunity ZIP Code assignment criteria are:

- Create a new Workflow Policy Column

- Define a new Workflow Policy Component Column as a child record of the existing Opportunity Workflow Policy Component
- Create a new Assignment Attribute and Assignment Attribute Column
- Create a new Assignment Criteria and Assignment Criteria Attributes

NOTE: Defining assignment criteria is only necessary when the predefined assignment criteria do not meet your organization's business needs. The tasks are summarized here to give you an idea of the underlying configuration that supports the subsequent sections of this chapter.

Creating Assignment Rules Based on Territories

The first step is to create assignment rules based on territories. For example, three of NREC's territories are San Francisco, Peninsula, and East Bay. Each of these territories is served by a partner real estate agency. Each real estate agency is associated with a set of ZIP Codes, which you will define as criteria values in the next procedure.

- San Francisco: 94101, 94102, 94103
- Peninsula: 94301, 94401, 94010
- East Bay: 94701, 94602, 94580

For the NREC example, you need to create an assignment rule for each territory listed above. The following procedures use San Francisco as an example.

To create assignment rules based on territories

1. From the application-level menu, choose Site Map > Administration - Assignment > Assignment Rules List.
2. In the Assignment Rules list, click New.
3. Enter a new record using the values as shown in the following table.

Field	Example Value	Description
Name	San Francisco	Unique name for the rule
Object	Opportunity	The object to be assigned
Rule Group	Default Rule Group	
Minimum Score	0	Although NREC is not using scoring, you must enter a value here.
Assignee Filter	All Above Minimum	Use all assignees with an assignment score greater than or equal to the assignment rule's minimum score. For a description of other methods read *Siebel Assignment Manager Administration Guide*.

Specifying Assignment Criteria and Values for Rules

After defining assignment rules for each territory, you need to define the assignment criteria and criteria values.

- **Assignment criteria.** Criteria used to determine which candidates qualify to receive the assignment. In the NREC example, the criteria compares the attributes of an opportunity and the attributes of a partner employee. Partner employees that have appropriate attributes qualify for the assignment.
- **Criteria values.** Specific values or range of values for a given criterion. Criteria values are used for comparison. In the NREC example, the values for the assignment criteria are the ZIP Codes that fall within a particular territory.

To specify territory assignment criteria and values

1. Navigate to the Assignment Rule View and select the assignment rule for San Francisco that you created in the previous section.
2. Drill down on the Assignment Rule name (San Francisco).

   The Assignment Rule appears in the top form applet and Criteria list is displayed below it.
3. Add a new record in the Criteria list, as shown in the following table.

Field	Value for NREC Example	Description
Rule Criterion	Prospect ZIP Code	Defines the type of criteria to be used. In this case, the NREC developer needed to create the assignment criteria, because the attribute being used is a custom attribute defined by NREC. For information about defining assignment criteria, read *Siebel Assignment Manager Administration Guide*.
Comparison Method	Compare to object	Method used to match objects to candidates. For example, Compare to object compares criteria values to object attributes.
Inclusion	Include	Method used to match criteria values and candidates.
Required	Always	Determines whether the criteria is required to qualify.
Score	0	Candidates that satisfy this criteria have this score added to their total score.

4 In the Values List applet (located below the Criteria List applet), add new records for each of the ZIP Codes included in this assignment rule.

For example, for the San Francisco territory assignment rule, you enter records for:

- 94101
- 94102
- 94103
- And so on

Adding Positions for Each Assignment Rule

Just as each rule has criteria that specify when the rule takes effect, each rule also has a list of positions, employees, or organizations that specifies the candidates for the assignment. In the NREC example, the assignment is based on positions. Each assignment rule is associated with the positions at the respective partner real estate agency. For example, the positions at the San Francisco Real Estate Agency are associated to the assignment rule for the San Francisco territory.

To add positions to an assignment rule

1 Navigate to the Assignment Rule view, and then select the assignment rule for San Francisco.

2 Click the Positions view tab.

3 Click the menu button and then choose New Record.

The Add Positions dialog box appears.

4 Select the Positions for the partner real estate agency to associate with this rule.

In the current example, these are the positions for the San Francisco Real Estate Agency:

- SF Real Estate Sales Manager
- SF Real Estate Agent 1
- SF Real Estate Agent 2

Releasing the Assignment Rules

After you have created and defined assignment rules, you must release them to instruct Assignment Manager to use these rules.

NOTE: This procedure releases all assignment rules. Do not release assignment rules while associated server tasks are running.

To release assignment rules

1. Navigate to the Assignment Rules view, and then select the rule for San Francisco.
2. Click Release.

   Assignment rules are released to Assignment Manager for use.

Activating the Rules

To activate Assignment Manager rules, you must start several Siebel Server components.

However, it is best to start the components just once—after you have created your rules in both Assignment Manager and Business Process Designer. For this reason, instructions on starting the components appear only at the end of the Business Process Designer chapter, in "Activating the Rules" on page 218.

For more information about Assignment Manager, read the *Siebel Assignment Manager Administration Guide*.

16 Siebel Business Process Designer

There are many ways to customize a Siebel application. In the previous chapter, you used one way—writing assignment rules using Assignment Manager. This chapter introduces another way to customize your application—using Business Process Designer to automate business processes.

For example, NREC plans to use Siebel Business Process Designer to send an email notification to partner real estate agents when an opportunity has been assigned to them. The email is based on a simple template, and informs them that they have a new opportunity in their queue. A *workflow process* defines the steps necessary to send the email. A *workflow policy* triggers the process when it detects that an opportunity has been assigned. This chapter describes the tasks required to set up this business rule, including:

The work in this chapter is done using the administration screens in NREC's test environment.

Configuring Siebel Communications Server

To send email using Siebel Business Process Designer, you must successfully configure your email communications application and configure Siebel Communications Server. This includes tasks such as:

- Setting up your Siebel Server machine to interface to different messaging applications that Siebel Communications Server supports
- Configuring and administer the Siebel Communications Server components you need, including the Communications Outbound Manager server component
- Configuring Siebel Communications Server communications drivers and profiles
- Configuring additional recipient groups for your communications, based on Siebel business objects and business components

For detailed instructions on performing these configuration tasks, read *Siebel Communications Server Administration Guide*.

Creating an Email Template

Before you can set up a workflow process to send an email message, you must first configure an email template. An email message sent from the Siebel application uses an email template, similar to how a form letter uses a template. Some template elements are replaced in the final message text. For example, Siebel field names are substituted with field values from database records. Use the Administration - Communications screen to create email templates.

There are two kinds of email templates: simple and advanced. Simple templates are used for commands such as Send Email, Send Fax, Send Wireless Message, and Send Page, and for replies in Siebel eMail Response. Advanced templates are used for outbound communication requests, such as the request triggered by a workflow process. The template discussed in this section is an advanced template.

To create an email template

1. From the application-level menu, choose Site Map > Administration - Communications > All Templates.

   The All Templates view appears.

2. In the Templates list, enter a new record using the values as shown in the following table.

Field	Value
Name	NREC Email template
Channel Type	Email

3. Click the Advanced view tab.

4 In the Advanced form, enter a new record using the values as shown in the following table.

Field	Value	Comments
Delivery Profile	Default SMTP Profile	Created for the Internet SMTP/POP3 Servers communications driver, for an email template. The profile must exist before you create the template. In this profile, define the communications sender using the From Address parameter override.
Recipient Group	Opportunity Sales Team Members	The recipient group you specify here only determines which substitution fields are available for the template text. The actual recipients are determined in the context in which the template is used.
Subject	New Opportunity in Queue	Default subject line for the email. The subject line can also include substitution fields.
Public	TRUE	Determines whether the template is available to all users.
Template Text	Dear [Full Name], A new opportunity has been assigned to you. Thank you, NREC	Text to appear in the body of the template. The term in brackets is a substitution.

For detailed information about email templates and fields, and about communications drivers and profiles, read *Siebel Communications Server Administration Guide*.

The email template is now available to use in the workflow process.

Creating a Workflow Process

A workflow process defines the series of actions you want to occur in the workflow. After the workflow process is triggered, it performs the specified actions. In general, a workflow process consists of one or more process steps, which can be Start steps, decisions, invocations of business services methods, subprocesses, or other types of steps. In the current example, the steps are simple.

- **Start.** Each workflow process must begin with a start step.
- **Send Email business service.** Predefined business services are available to use for performing actions such as sending email.
- **End.** Each workflow process must end with an End step.

To create the workflow process you use the Workflow Process features in Siebel Tools.

Using Workflow Process Features in Siebel Tools

Figure 42 shows the Workflow Process Designer with the steps for the current example.

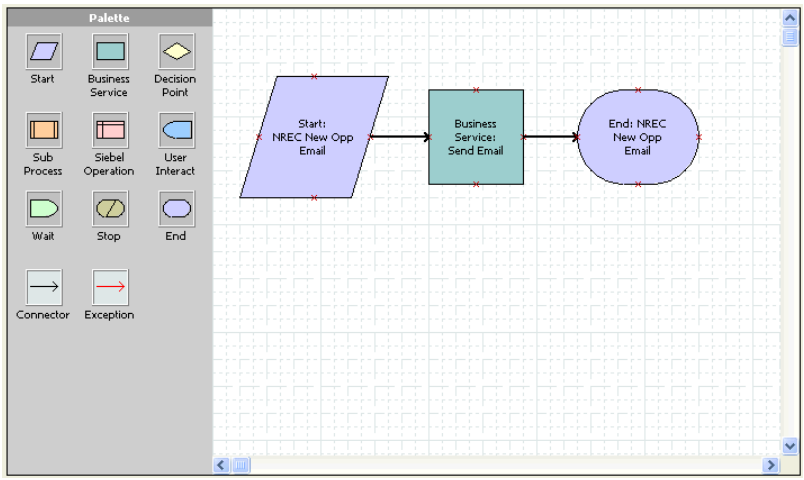


Figure 42. Workflow Process Designer View with Flowchart of Current Example

To create a workflow process in Siebel Tools

1. Set Siebel Tools to display all objects within the Workflow Process hierarchy.
   a. Choose View > Options.
   b. Click the Object Explorer tab.
   c. For the Workflow Process object, select every object in its hierarchy and then click OK.
2. In the Object Explorer, click Workflow Process.
3. Enter a new record with the values as shown in the following table.

Field	Value	Comments
Name	NREC New Opportunity Notification	Unique name that identifies the process.
Business object	Opportunity	Business object is Opportunity because the goal is to trigger an action based on changes to the Opportunity record. The pick list contains only those business objects that have a primary business component defined.
Project	NREC Configuration	

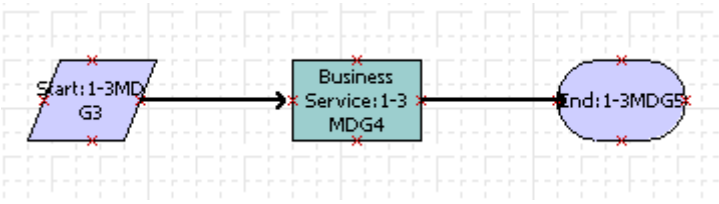
4. Right-click on the record and choose Edit Workflow Process.

   A Workflow Process palette and workspace appear.

5 Drag appropriate icons from the palette to the flowcharting workspace.

Notice that different shapes and colors are used to represent different types of steps. In the current example, a pink rectangle represents a business service. For information about the different types of steps, read *Siebel Business Process Framework: Workflow Guide*.

6 Drag connectors from the palette to the flowcharting workspace to connect the steps of the process, as shown in the following illustration.



7 In the WF Steps window, edit the name for each step as shown in the following table.

Step	Name
Start	Start: NREC New Opp Email
Business Service	Business Service: Send Email
End	End: NREC New Opp Email

8 For the Business Service step, make these additional settings as shown in the following table.

Field	Value
Business Service	Outbound Communications Manager
Business Service Method	CreateRequest

9 In the workspace, select the Business Service step. Right-click and choose Show Input Arguments.

The Input Arguments window appears below the workspace.

10 In the Input Arguments window, add the records as shown in the following table. When picking an Input Argument using the Business Service Method Arguments window, search on the Name field.

Table 13. Field Values for Input Arguments

Input Argument	Type	Value	Property Name
PackageNameList	Literal	NREC Email Message	
RecipientGroup	Literal	Opportunity Sales Team	
RequestName	Literal	NREC Request	
SourceIdList	Process Property		Object Id

For a detailed description of the arguments required for predefined business services, read *Siebel Business Process Framework: Workflow Guide*.

11 In the Object Explorer, click Workflow Process to redisplay the list of workflows.

12 Select the NREC New Opportunity Notification workflow process.

13 Click Deploy (the button is located between the window title “Workflow Processes” and the field headings).

The status changes from “In Progress” to “Completed.”

After the workflow has a status of Completed, the next task is to deploy it in the Siebel application.

To deploy a workflow process

1 Log in to the client software as an administrator.

2 Navigate to Administration - Business Process > Workflow Deployment and query for the workflow you just deployed.

3 With the workflow process selected, click Activate.

This checks the syntax for validity, registers run-time events if used, and changes the status of the process to Active. It also changes the status of the previous active version to Outdated.

4 Check the Active Workflow Processes list (below the Administration - Business Process list) to make sure the NREC New Opportunity Notification workflow process is listed and has a status of Active.

Now you must create a policy that calls this workflow process. A Workflow policy consists of a condition and an action. The action in this example is to invoke the Account Status Change process. For instructions on defining the action and conditions, read the following section, “Creating a Workflow Policy.”

Creating a Workflow Policy

Workflow policies define the conditions under which a workflow process is invoked. In this example, the condition is when an opportunity is assigned by Assignment Manager. When this occurs the workflow policy invokes the workflow process defined in "Creating a Workflow Process" on page 213.

Creating a Policy Action

Workflow policy actions define the events that you want to occur when the conditions of your Workflow policy are met. In the NREC example, the action is to run a workflow process.

To create a policy action

1. Navigate to Administration - Business Process > Actions.
2. In the Actions list, add a new record to specify what actions the policy triggers.

   In the current example, the required fields and their values are shown in the following table.

Field	Value	Comment
Name	NREC Opportunity Notification	The name of the action can be any name meaningful to you.
Program	Run Workflow Process	This value specifies that the policy triggers a workflow process.
Workflow Object	Opportunity	Name of the object associated with the workflow policy.

3. In the Arguments list, add records for the necessary arguments.

   In the current example, the required fields and their values are shown in the following table.

Argument	Value	Comment
Argument	ProcessName	Because you previously specified Run Workflow Process, ProcessName is the only item in the pick list.
Value	NREC New Opportunity Notification	This value is the name of the process you created as described in "Creating a Workflow Process" on page 213. The workflow process must be active for it to appear in the drop-down list for the Value field.

Creating a Policy Condition

Now that you have defined the event that you want to occur as a Policy Action, you can define the conditions under which the action occurs. You do this by defining a Policy Condition.

To create a policy condition

1 Navigate to Administration - Business Process > Policies.

2 In the Policies list, enter a new record.

For the current example, complete the fields shown in the following table.

Field	Value	Comment
Name	NREC Opportunity Notification	This name can be any name that is meaningful to you.
Workflow Object	Opportunity	Choose this value because the action is based on changes to an Account field.
Group	NREC	Because this group does not currently exist, you must create it. 1 Click the select button in the Group field. 2 In the Workflow Groups dialog box, click New. 3 Specify NREC in the Name field. 4 Click Save. 5 Click OK.

3 In the Conditions list, enter the condition as shown in the following table.

Condition Field	Operation
Position ID	IS ADDED

4 In the Actions list, enter the record as shown in the following table.

Field	Value	Comments
Action	NREC Opportunity Notification	This is the name of the Workflow Policy Action you created in "Creating a Policy Action" on page 217.
Sequence	10	Sequence of the Actions.

Activating the Rules

To activate rules—whether they are Assignment Manager rules or Workflow rules—you must run the following Siebel Server components.

- Generate Triggers
- Workflow Monitor

Running Generate Triggers

Use the Generate Trigger (GenTrig) component on the Siebel Server to create database triggers. The Workflow Policies module of Siebel Business Process Designer uses these database triggers to identify which records may match policy conditions. The Generate Triggers component needs to be rerun whenever new policies are created or deleted.

You can run the Generate Triggers component from either the Server Manager graphical user interface, or command line mode. The following procedure is for the GUI. For information about running Server Manager using the command line interface, read *Siebel System Administration Guide*.

To generate triggers using the GUI

1. Navigate to Administration - Server Management > Enterprises.
2. In the Component Requests view, click New.
3. Select Generate Triggers from the Component Job list.

   This creates a new line entry but does not start the task.
4. Click New in the Component Job Parameter list and define the parameters as shown in the following table.

Name	Value	Comment
Trigger File Name	TRIGGER.SQL	This filename identifies the SQL script that needs to be run. The file is created in the *siebel_srvr_root* directory during installation, and its contents get created as a result of generating triggers.
EXEC	TRUE	Setting this parameter to TRUE causes the SQL script to be run automatically.
Table Owner Password	Your Password	Password for the table owner. This is required for Oracle and IBM DB2.

5. Enter your Privileged User name and password.
6. Click the Component Request form applet menu, then select Submit Request.
7. To view changes to the state, refresh the screen by clicking Run Query from the applet menu.

   Upon completion, the Status field contains either Success or Error.
8. View log details by doing the following:
   a. Click the Screen Enterprise Operation.
   b. In the Show field, select Tasks.
   c. Click the Task Info Log view tab.

   For more information about administering server component parameters, read *Siebel System Administration Guide*.

Starting Workflow Monitor Agent

You must start two Workflow Monitor Agent tasks: one for Assignment Manager and the other for Workflow policies.

To create a Workflow Monitor Agent component definition

1. Navigate to Administration - Server Configuration > Enterprises.

   The Enterprise Configuration view appears.

2. Click the Component Definitions tab.

   Two Component Definitions lists appear.

3. From the upper Component Definitions list menu, choose New Record.

   A new record appears.

4. Complete the fields as shown in the following table.

Field	Value	Description
Name	NREC's WorkMon	Name of the component.
Component Type	WorkMon	Worfkow Monitor Agent component type.
Component Group	NREC Workflow Component Group	Select an existing component group.
Alias	NRECWorkMon	Alias for the component. The alias can not contain blank spaces.

5. From the upper Component Definitions list menu, choose Save Record.

   The component definition is saved. To view the definition, you must perform a query.

To set parameters and activate a Workflow Monitor Agent component definition

1. In the upper Component Definitions list, perform a query for the component definition.
2. In the lower Component Definitions list, choose the Group Name parameter. Enter the name of the Workflow Policy Group for the requests the component will process.
3. In the lower Component Definitions list, choose the Default Tasks parameter. Set the Value to 1.

   This sets the component to start when the Siebel server starts, and stop when the Siebel server shuts down.

4. Optional. You may make additional changes to the component parameters. For a description of Workflow Monitor Agent parameters, read *Siebel Business Process Framework: Workflow Guide*.
5. From the upper Component Definitions list menu, choose Enable Component Definition.

   The definition state changes from *Creating* to *Active*.

6 Restart the Siebel server.

Your changes take effect.

17 Personalization

This chapter describes the steps for using Siebel Personalization to create a simple business rule that filters NREC's list of houses. The filter is based on the ZIP Code value of the house record and the user's Postal Code profile attribute. When a partner real estate agent logs in to NREC's application, the agent can access only the houses in the House Detail view that are in the same ZIP Code as is defined on the user's profile.

NOTE: ZIP Code is a field that you added to the Internal Products business component in "Configuring the House Detail View" on page 87.

The tasks in this chapter are performed using administration views available in the Siebel Web Client. Like Siebel Workflow and Siebel Assignment Manager, Siebel Personalization provides you with another way to customize the business logic of your application without having to work in Siebel Tools and recompile an SRF file.

This chapter includes the following topics:

Creating Rule Sets

Rule sets are collections of the business rules that define how content is displayed. You create one rule set to contain the business rules for the NREC example.

To create a rule set

1. In a Siebel employee application, navigate to Administration - Personalization > Rule Sets.

   The Rule Sets view appears.

2. Add a new record to the Rule Sets list using the values as shown in the following table.

Field	Value
Name	NREC House Filter
Active	TRUE

Associating Rule Sets with Applets

Rule sets are associated with one or more applets. These are the applets to which the business rules for the rule set apply. NREC is using Personalization to filter the list of houses for sale based on a user's ZIP Code. When users view the list of houses for sale, they can access only the houses for sale in their area. In this case, the applets to which the Personalization rule applies are the Product Form Applet and the Product List Applet.

To associate a rule set with an applet

1 In a Siebel employee application, navigate to Administration - Personalization > Applets.

The Applets view appears.

2 In the Applets list, enter a new record for the Product List Applet.

3 In the Rule Sets list, add a new record as shown in the following table.

Field	Value	Description
Name	NREC House Filter	This is the rule set you created in "Creating Rule Sets" on page 223.
Sequence	1	Choose the order in which the rule sets are evaluated.
Start Date	Leave blank	The date when the conditional expression starts to be evaluated. If Start Date is blank, the conditional expression is evaluated continuously or until the End Date is reached.
End Date	Leave blank	The date after which the conditional expression is not evaluated. If End Date is blank, the conditional expression is evaluated continuously after the Start Date. If both Start and End Date are blank, the conditional expression is always evaluated.

4 Repeat Step 1 through Step 3 to associate the Product Form Applet with the NREC House Filter.

Creating Rules

Personalization rules control how content is targeted to users. There are three types of business rules: Business Service, Expression, or Invoke Method. For the NREC example, you write an Expression personalization rule.

Expression rules use Siebel Query Language to set the parameters that control the content displayed to users. In this case, the expression specifies that when displaying records from the Internal Product business component only show those records that have a ZIP Code field that matches the user's Postal Code attribute. The logic expressed in Siebel Query Language is:

```
[Zip Code] = GetProfileAttr('Me.Org.Postal Code')
```

NOTE: ZIP Code is a field that you added to the Internal Products business component in "Configuring the Internal Product Business Component" on page 92.

For more information about other types of personalization rules and Siebel Query Language, read *Siebel Personalization Administration Guide*.

To create a rule

1. In a Siebel employee application, navigate to Administration - Personalization > Rule Sets.
2. In the Rule Sets list, choose the NREC House Filter rule set that you created as described in "Creating Rule Sets" on page 223.
3. In the Rules list, enter a new record using the values as shown in the following table.

Field	Value	Description
Name	NREC House Filter	Choose a name that suggests the purpose of the rule.
Sequence	1	Required. Enter numbers in this field to set the order in which the rules are evaluated.
Rule Type	Expressions	Evaluates inclusion and exclusion expressions directly.
Active	TRUE	Check the box to use the rule.
Include Expression	`[ZIP Code] = GetProfileAttr('Me.Org.Postal Code')`	Used with the Expressions rule type. An expression that sets parameters to include content.

NOTE: You can type the expression directly into the Include Expression field or you can click the select button to invoke the Expression Designer. The Expression Designer provides you with a drag-and-drop interface for writing complex expressions. For instructions on using the Expression designer to enter the Include Expression for this example, read "Using the Expression Designer" on page 225.

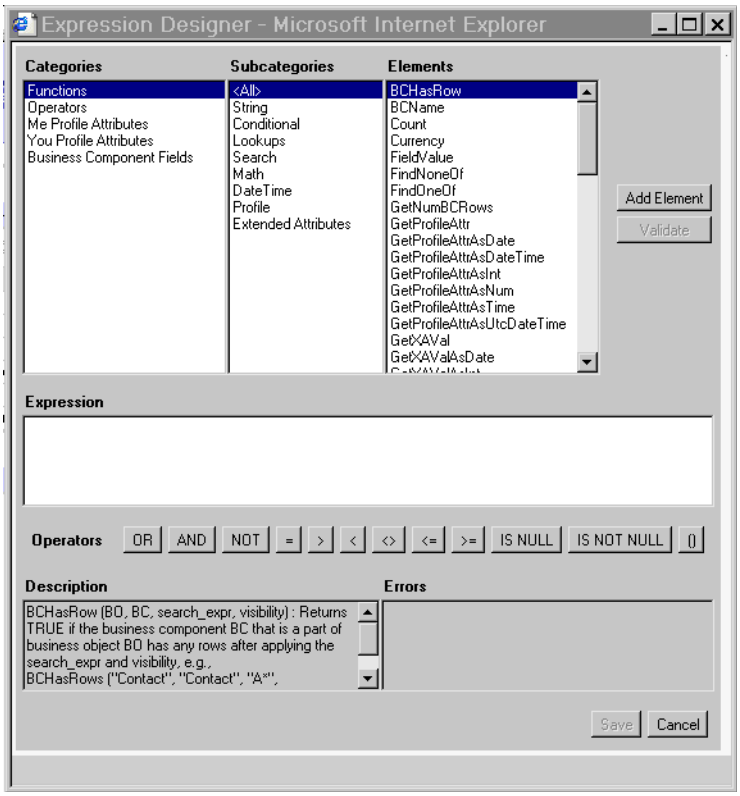
Using the Expression Designer

Use the Expression Designer to write personalization rules without having to learn Siebel Query Language. It provides you with a drag-and-drop interface and helps you validate syntax of the expression. This procedure guides you through the steps for creating the Include Expression defined in Step 3 on page 225. This procedure is an alternative to typing the expression directly into the Include Expression field.

To write a personalization rule using the Personalization Business Rules Designer

1 Click the select button in the Include Expression field.

The Expression Designer appears as shown in the following illustration.



2 Choose Business Component Fields from Categories, <All> from Subcategories, and Zip Code from Elements, and then click Add Element.

The Expression box displays [ZIP Code].

3 Place the cursor after [ZIP Code] in the Expression box.

4 Click = in the Quick bar.

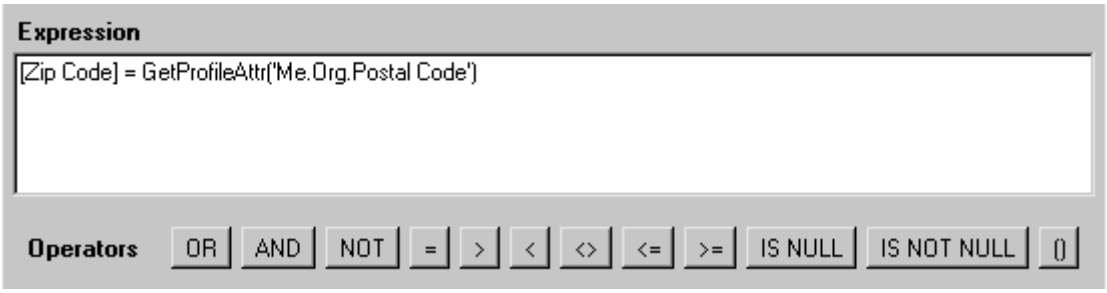
The Expression box displays [ZIP Code]=.

5 Place the cursor after =.

6 Choose Me Profile Attributes from Categories, Show as String from Subcategories, Me.Org.Postal.Code from Elements, and then click Add Element.

The Expression box displays the completed rule as shown in the following illustration.

```
[ZIP Code] = GetProfileAttr('Me.Org.Postal Code')
```



7 Click Save to validate the rule.

The Expression Designer closes and the expression is displayed in the Include Expression field of the Applet Rules More Info form.

Testing

Personalization rules can be tested in a staging environment before being used in the production environment. This is done in the Test view under Administration - Personalization.

NOTE: For the current example, use the House Detail view in your configured application to enter a few house records. Be sure to enter records with several different ZIP Codes. You will define one of these values as a profile attribute when you test. Only records with that ZIP Code appear. For information about the House Detail View, read "Configuring the House Detail View" on page 87.

To test personalization rules

1 In a Siebel employee application, navigate to Administration - Personalization > Test.

The Test view appears.

2 Enter a Primary User Login and Primary User Password.

This is the user name and password of the user to use to test the personalization rules.

For example, to test NREC's personalization rule, use SADMIN and SADMIN.

3 In the Test Application field, enter the command string for the Mobile Web Client.

NOTE: You can right-click your program icon, choose Properties, and then copy and paste the command string from the Target field into the Test Application field. You may have to add the data source using the /d switch.

```
C:\<Siebel_install_dir>\BIN\siebel.exe
/c "C:\<Siebel_install_dir>\bin\ENU\ <config_file>.cfg"
/d <data_source>
```

where:

- <*Siebel_install_dir*> is the full path to the client installation directory

- <*config_file*> is the application configuration file, for example, uagent.cfg for Siebel Call Center and siebel for Siebel Sales
- <*data_source*> is the database to which to connect: Local, Sample, or ServerDataSrc

For example:

```
C:\Program Files\Siebel\7.7\client\BIN\siebel.exe
/c "C:\Program Files\Siebel\7.7\client\BIN\ENU\scw.cfg"
/d Sample
```

NOTE: You can also enter the URL for a Siebel Web Client. For example, http://<*machine_name*>/eChannel. When testing using the Siebel Web Client, the Test Mode dialog box appears after you complete Step 6 with instructions about how to launch a new instance of the application. For more information, read *Siebel Personalization Administration Guide*.

4 Click Load.

The persistent user profile attributes of the primary user are loaded in the Primary User Attributes list with a Me. prefix.

Person-related attributes have the value Person in the Source field. Organization-related attributes have the value Organization.

5 Select the Me.Org.Postal Code primary user attribute and enter a value.

For example, enter 94121.

6 Click Test.

A new instance of the specified application opens. For the current example, a new instance of Siebel Partner Portal opens.

7 Navigate to the House Detail view and test the results.

Only records that have a ZIP Code value of 94121 appear.

8 To save the test setup, click Save in the test view.

18 Implementing Siebel Remote

This chapter begins by explaining how National Real Estate Clearinghouse (NREC) uses Siebel Remote, illustrating the hardware architecture, and outlining the tasks to set up the Siebel Remote server. Next, the chapter outlines and briefly describes the tasks to set up a new Siebel Remote User. Then the chapter describes the process for synchronizing a mobile Web client.

NREC is using Siebel Remote so that its employees can connect to a Siebel Server with their laptop computers and exchange updated data and files. This process is known as *synchronization*. This chapter uses Chris Strong, a real estate agent with NREC, to detail the Siebel Remote scenario. Chris Strong is a mobile user.

Siebel Remote supports mobile computing by allowing field personnel to share current information with virtual teams of other mobile and connected users across the organization.

Typically, the mobile Web client is a laptop computer used by a field sales or service representative. A mobile Web client can download a portion of the Siebel database and the Siebel File System to a laptop. Users can access their data locally, without being connected directly to the Database Server, Siebel Server, or File System.

As Chris Strong enters and updates information in the local database on the laptop, Siebel Remote Client software tracks the changes as synchronization transactions. Subsequently, when Chris connects to the Siebel Remote server through a dial-up networking connection, these transactions are uploaded from the mobile Web client to the server during synchronization. In a similar manner, transactions occurring on the server that are applicable to Chris are tracked. During synchronization, these transactions are downloaded from the server to the laptop.

Figure 43 illustrates the main elements of the Siebel Remote architecture.

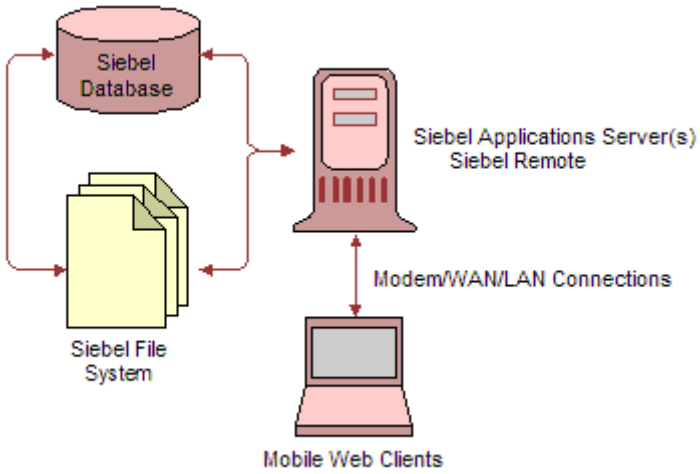


Figure 43. Siebel Remote Hardware Architecture

Setting Up the Siebel Remote Server

The Siebel Remote server runs the Siebel Remote components (such as txnproc, txnroute, and others) and manages synchronization sessions with mobile Web clients. The Siebel Remote server provides an interim storage area for data required to synchronize mobile databases with the Siebel database server.

Setting up the Siebel Remote server in preparation for setting up Siebel Remote users includes the following tasks.

- Setting Siebel Remote System Preferences
- Disabling Local Access to All Views
- Starting Siebel Remote Server Components
- Generating a New Database Template

For detailed information on these and other Siebel Remote administration tasks, read *Siebel Remote and Replication Manager Administration Guide*.

Setting Up a New Siebel Remote User

Setting up a mobile Web client involves certain tasks for both the Siebel Remote server and the mobile Web client. The administrator must repeat each of these steps for each mobile client.

An administrator at NREC completed the entire process of setting up each mobile client. NREC decided not to rely on end users to complete the configuration because the administrator could be more thorough and accurate in completing each task.

Setting up a new Siebel Remote user entails the following tasks.

1. Create a mobile Web client user account and privileges. For details, read "Creating a Mobile Web Client User Account and Privileges" on page 231.
2. Set up mobile Web client hardware and software. For details, read "Setting Up Mobile Client Hardware and Software" on page 231.
3. Enable network connectivity. For details, read "Enabling Network Connectivity" on page 231.
4. Establish autodial preferences. For details, read "Establishing Autodial Preferences" on page 231.
5. Set synchronization preferences. For details, read "Setting Siebel Remote Preferences" on page 232.
6. Register a mobile Web client. For details, read "Registering a Mobile Client" on page 232.
7. Run database extract for a mobile Web client. For details, read "Running Database Extract for a Mobile Web Client" on page 234.
8. Initialize a mobile Web client local database. For details, read "Initializing a Mobile Web Client Local Database" on page 235.

NOTE: For information about advanced topics, such as authentication or conflict detection and resolution, read *Siebel Security Guide* and *Siebel Remote and Replication Manager Administration Guide*.

Creating a Mobile Web Client User Account and Privileges

NREC chose not to use authentication for mobile Web clients. Therefore, NREC did not need to create any accounts or passwords for these. Because it is the easiest and most popular configuration, it is also the default configuration.

If you want to authenticate mobile Web clients, you need to create accounts for each client depending on the authentication method. For more information about this topic, read *Siebel Remote and Replication Manager Administration Guide*.

Setting Up Mobile Client Hardware and Software

Install the necessary hardware and software on the mobile Web client. This step may include:

- Configuring users.
- Installing disk drives, memory cards, and operating system software.

For more information, read *Siebel Installation Guide for Microsoft Windows*.

Enabling Network Connectivity

Install the necessary hardware and software on the mobile Web client so the mobile Web client can exchange files with the Siebel Remote server. This step may include choosing communication settings and installing networking cards, modems, and software.

For more information about network connectivity, read *Siebel Installation Guide for Microsoft Windows*.

Establishing Autodial Preferences

A mobile client user using Microsoft dial-up networking can configure Siebel Remote to automatically to establish a connection with the Siebel Remote server when the user initiates a synchronization session.

To establish autodial preferences

1. From the application-level menu, select File > Synchronize > Database.
2. In the Siebel Remote Synchronize dialog box, click Setup.
3. From the Siebel Remote Preferences dialog box, click the Connection pick list and choose the appropriate connection.
4. Click OK, and then Synchronize or close the dialog box.

NOTE: You must define your phone book entries before synchronizing.

Setting Siebel Remote Preferences

NREC is using the default settings for the Siebel Remote parameters established during installation in the client configuration file. Thus, there are no tasks for setting remote preferences.

Setting Synchronization Options

The Siebel Remote client reads configuration parameters in the Siebel configuration file (default is siebel.cfg) to specify the location of the Siebel Server. Before using Siebel Remote, you must set the values for the configuration parameters. The Siebel installation utility creates a siebel.cfg in the client bin directory with default values for each configuration parameter.

When Chris Strong and other NREC mobile users perform synchronization within the application, that is, using File > Synchronize > Database, configuration information is read from the CFG file of that particular application. For example, if Siebel Call Center is used, then configuration information is read from the uagent.cfg.

For more details about synchronization parameters and enabling encryption for synchronization networking, read *Siebel Remote and Replication Manager Administration Guide*.

Registering a Mobile Client

This section describes how to register a mobile Web client. It assumes the Siebel Administrator has previously set up Chris Strong as an employee in the Siebel application.

NOTE: It is important to make sure mobile users have the Client Status view in their responsibilities. This view helps mobile Web clients resolve data merge conflicts on their local databases by showing conflict information after synchronization. For additional detail regarding the setting up of employees, read *Siebel Applications Administration Guide*.

To register a new mobile Web client on a parent node

1. From the application-level menu, choose Site Map > Administration - Remote > Mobile Clients.
2. In the Parent Server list, choose the appropriate parent node.

   If your deployment does not use Replication Manager, then the appropriate node is HQ.
3. In the Mobile Clients list, click the menu button and choose New Record.
4. In the New Mobile Client form, enter Chris Strong in the Mobile Client field.

   CAUTION: The mobile Web client name and the User ID must be entered in uppercase letters and be eight characters or less. A good practice is to use the mobile Web client User ID (next step) as the mobile Web client name. It *can* contain only Roman, alphanumeric, and the _ or - characters. It *cannot* include spaces, periods, or other invalid characters (/ \ : * ? “ < >|) as in the DOS file naming schema. Siebel Remote uses the mobile Web client name to create inbox and outbox directories on the Siebel Server.

5 In the User ID field, click the select button and choose the User ID for Chris Strong and click OK.

The User ID is used to access Chris Strong's local database during initialization and synchronization.

6 In the Routing Model field, click the select button and choose the Sales Representative routing model and click Pick.

NOTE: Each mobile user is associated with one Routing Model. A Routing Model includes a set of routing rules. The union of the routing rules determines whether a record is routed to a mobile Web client. For more information on data routing models, read *Siebel Remote and Replication Manager Administration Guide* and the release notes documentation for your application.

7 In the Language(s) field, click the select button, and choose English as the preferred language for Chris Strong.

Using a language preference the mobile user can download data in a preferred language, or languages, for the following dock objects that contain Translation Tables: LOV, Product, Literature, Catalog, Catalog Category. This helps optimize the size of the local databases.

For more information about language preferences, read *Siebel Remote and Replication Manager Administration Guide*.

8 Complete the remaining fields as appropriate.

The Sync Password field is used by the Synchronization Manager if the authentication method in the Siebel Server Component Parameters is set to Siebel. Set the password in this field and give it to the mobile user.

The App Server Name field is not populated until the Database Extract is run for the mobile user. At the time the mobile Web client record is created (S_NODE) the APP_SERVER_NAME field is NULL.

NOTE: If you use EIM to load mobile user records, records without an HQ node as the parent node do not appear in the Mobile Clients view. The parent node is stored in the following two columns: EIM_NODE.par_name and EIM_NODE.par_node_type_cd. Although these columns are not required for EIM, they are required for Siebel Remote. When you enter mobile users using the Mobile Clients view, these columns are populated by default.

9 Navigate to Administration - Applications > Responsibilities > Responsibilities list, and choose the corresponding Responsibility with the Routing Model suffix.

This is required because Chris Strong's data routing model, Sales Representative, is one of the following:

- Sales Representative
- Minimal Data
- Analyst
- Sales Manager

The corresponding Responsibility with the Routing Model suffix relates to the data routing model assigned in Step 6 above. For more information about corresponding routing models and how these help optimize the size of local databases, read *Siebel Remote and Replication Manager Administration Guide*.

a In the Users list, add a new record.

b In the Add Users Selection dialog box, select the mobile user and click OK.

Running Database Extract for a Mobile Web Client

The database extract process retrieves data visible to Chris Strong from the server database. It retrieves data according to routing rules that define the level of access to information for Chris Strong. It creates compressed files that contain data to be loaded into the local database when Chris initializes the laptop.

Before running a database extract for Chris Strong, you must make sure that your organization's reporting hierarchies are updated and that Chris has a valid position in your organization's hierarchy. The resulting information is used by the application's routing rules, and may affect the outcome of the database extract. For more information on positions, read *Siebel Applications Administration Guide*.

To run a database extract for a mobile Web client

1 From the application-level menu, choose Site Map > Administration - Server > Enterprise Operations.

2 Click the Component Jobs tab, and click New.

3 In the Component/Job field, choose Database Extract from the pick list.

4 In the Component Job Parameters list, click New and add the necessary parameters.

   The required parameter for Database Extract is Client Name.

   The value for the Client Name parameter is CSTRONG, the name of Chris Strong's mobile Web client.

5 From the Component Jobs menu, choose Submit request.

The mobile client database is extracted. This may take a few minutes.

Server Directory Tree After Running Database Extract

The Database Extract program creates the appropriate directories for Chris Strong on the Remote server.

NOTE: The installation program also places a directory named txnproc in the docking directory within the Siebel server root directory. Do *not* modify the contents of this directory under any circumstances.

The following example shows a portion of the server directory tree after you run Database Extract for Chris Strong.

```
siebel
  docking
     cstrong
```

```
        inbox
        outbox
    txnproc
```

For more information about this topic, read "Sample Directory Tree After Running Database Extract" on page 66.

Initializing a Mobile Web Client Local Database

The volume of information that must be downloaded from the Siebel Remote server to initialize a mobile Web client's database is usually substantial. Establish a LAN (rather than a modem or WAN) connection between the server and the mobile Web client for this process.

Alternatively, the local database can be initialized from a CD-ROM or other media—if compressed files have been copied into the folder specified as FileSystem parameter. For more information about extracting to a CD, read *Siebel Remote and Replication Manager Administration Guide*.

NOTE: To initialize a mobile Web client database, the TableOwner parameter in the CFG file must be set to Siebel (the default).

To initialize the mobile Web client database

1. Establish a connection between the Siebel Remote server and Chris Strong's laptop.
2. In the mobile Web client's Siebel program group, click the Siebel Remote icon.

   NOTE: Verify that the icon is pointing to the appropriate CFG file. The default is siebel.cfg.
3. In the Siebel Remote Parameters dialog box, enter the information as shown in the following table.

Field	Entry
Client Name	CSTRONG (registered Siebel client name).
User Name	CSTRONG (login name).
Password	CSTRONG (If an authenticator password was assigned, enter that instead of CSTRONG).

4. Click Continue.
5. Monitor the process for errors.

To initialize the mobile client database during login

- Log in to the local database when starting the application. When Siebel Business Applications cannot find a local database, it attempts to initialize the local database. Follow the prompts.

After the initialization completes, Chris Strong's laptop is ready for use in the field. Chris Strong needs to resynchronize with the Siebel Remote Server on a frequent basis. Instructions for this process are included in the next section, "Synchronizing a Mobile Web Client" on page 236. There is an auto synchronization option to help maintain the frequency of synchronization.

Synchronizing a Mobile Web Client

This section describes the processes for synchronizing a mobile Web client.

Routing and Merging

On the Siebel Remote server, the Transaction Router and Transaction Merger components continuously route and apply transactions for mobile clients. Figure 44 illustrates the processes that occur when a mobile client is synchronized.

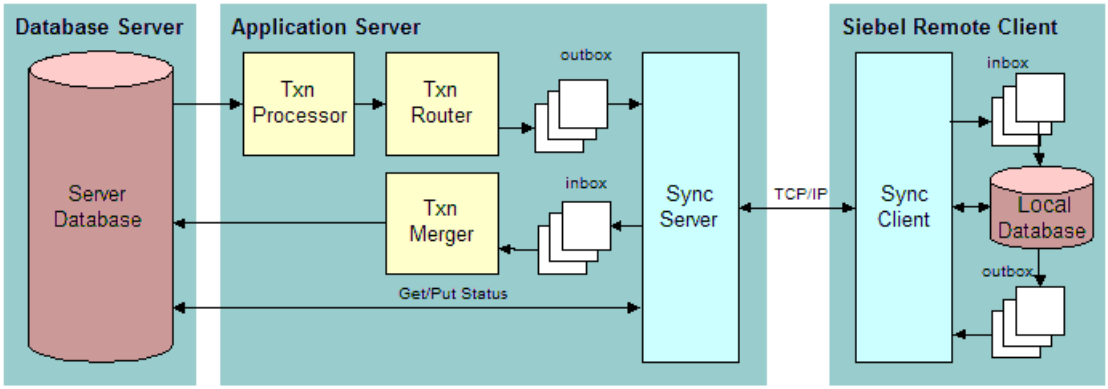


Figure 44. Synchronizing a Mobile Web Client (Routing and Merging)

For more detailed information about routing and merging, read *Siebel Remote and Replication Manager Administration Guide*.

Synchronization Session

To synchronize an existing mobile Web client, the mobile user launches Siebel Remote, either from within the Siebel mobile Web client or in stand-alone mode. Siebel Remote executes the following steps:

- Connects
- Validates mobile Web client
- Check for correct version
- Checks for database extract
- Retrieves transactions and file attachments
- Sends transactions and file attachments

- Applies changes to the local database
- Disconnects
- Applies changes to the server database
- Cleans up

For more detailed information about synchronization steps, read *Siebel Remote and Replication Manager Administration Guide*.

How Changes Are Propagated To and From a Mobile Web Client

Between synchronization sessions, the Siebel Remote server prepares transactions applied to the database server by both mobile and connected users. Siebel Server components write the transactions to a separate directory for each mobile user, such as Chris Strong. These transactions, combined with items from the file system, are downloaded to Chris during the next synchronization session. Items from the file system include updated, published, or requested marketing literature, correspondence templates, and other types of file attachments.

A similar process occurs on Chris's laptop as well, although without the server component.

Process Flow for Changes by Connected Users

This section describes the process flow for downloading changes on the server database to local databases, such as Chris's. The flow begins from the time a connected user creates a new opportunity until it appears in Chris's local database.

This specific process flow includes an example of a telesales representative in a Call Center. The telesales representative talks to potential customers responding to a new NREC advertising campaign.

The telesales representative decides to create a new opportunity record for one of the more promising responses. The figure below illustrates this flow and includes the following groups of steps.

1. The telesales representative creates a new opportunity record—a transaction saved in the opportunities table on the server database. A copy is saved to the master transaction log.

2 A mobile user invokes a synchronization session from the laptop.

The mobile user can use the Siebel client while the Siebel Remote client applies the changes to the local database, as shown in Figure 45.

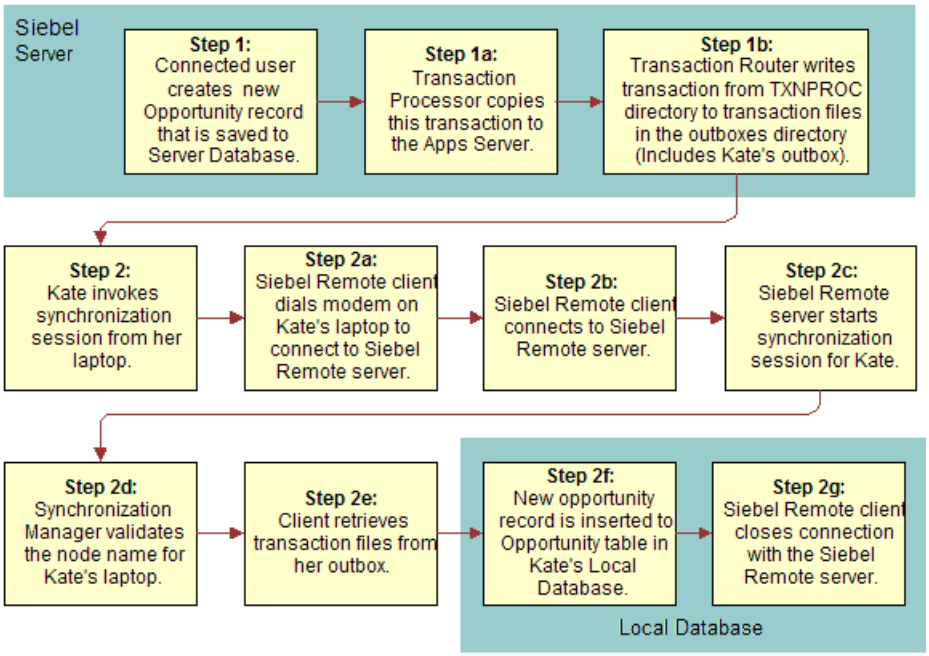


Figure 45. Process Flow for Changes by Connected Users

CAUTION: Users must never directly modify the local transaction log. The Siebel Remote synchronization client automatically purges the local transaction log table.

For more information about process flow for changes by connected users, read *Siebel Remote and Replication Manager Administration Guide*.

Process Flow for Changes Made by Mobile Users

Mobile Web clients, such as Chris, use a local database to store data for user access. The local database contains Siebel Business Applications tables that store user data. The local database also contains a local transaction log to store transactions created by the mobile user. Siebel Remote forwards these transactions to the Siebel Remote server when the client synchronizes.

This section provides a description of each phase of the process flow, from the time when Chris modifies the new opportunity until the time when the modifications appear in the Server database. For this example, assume that Chris is meeting with the potential new client, represented by the opportunity record entered by the telesales representative in the previous section.

The figure below illustrates this flow and includes the following groups of steps.

1. As a result of the meeting, Chris makes changes to the Opportunity record in the local database. Chris enters these changes immediately after the meeting while working offline.
2. The modified opportunity record is saved to the Opportunities table in Chris's local database.
3. The transaction record is saved to the Local Transaction log.
4. Chris synchronizes the laptop. The Siebel Remote client extracts pending transactions from the Local Transaction log into transaction files (.dx). The client then places these DX files in the outbox directory on Chris's laptop. Siebel Remote connects to the server and the DX file is sent to the server and inserted in the database as shown in Figure 46.

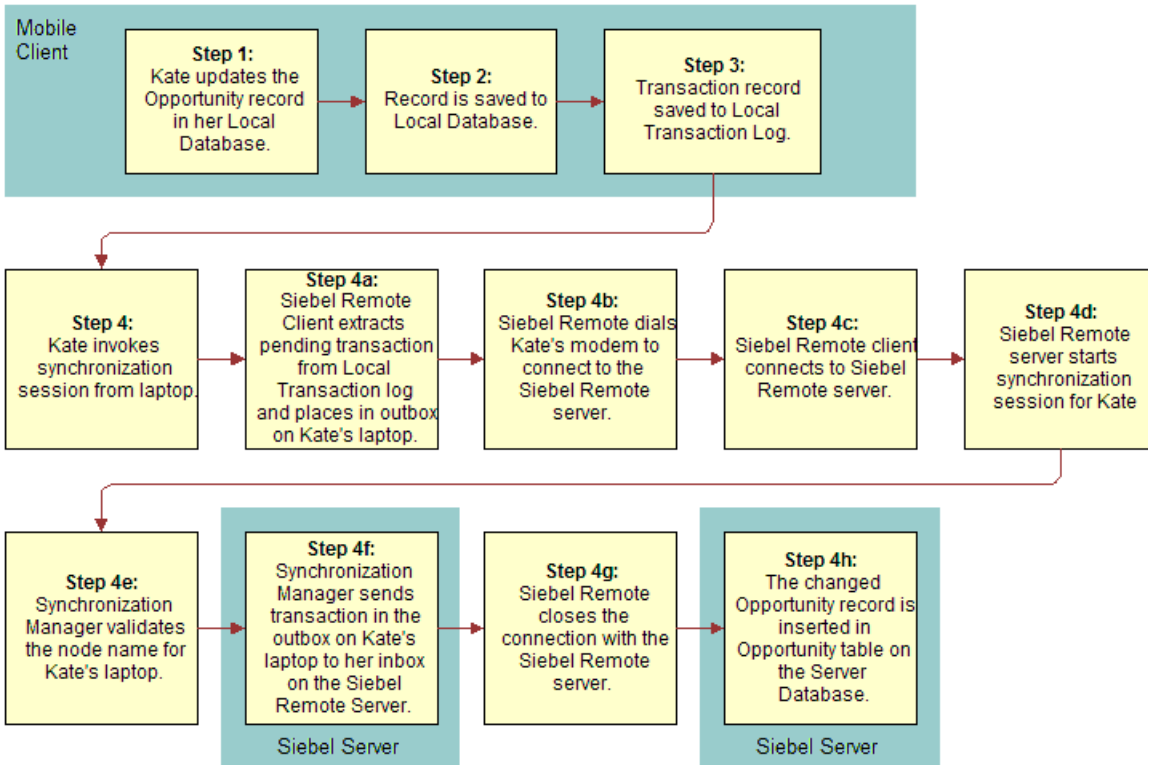


Figure 46. Process Flow for Changes by Mobile Users

For more information about process flow for changes by mobile users, read *Siebel Remote and Replication Manager Administration Guide*.

Synchronizing a Mobile Web Client Machine

Chris must synchronize frequently to obtain and view possible updates in the server database. Also, there may be updates to the store of documentation, marketing literature, and sales brochures in the file system.

Siebel 7.7 introduces the TrickleSync option to help maintain the frequency of synchronization. Frequent synchronization by mobile users can improve the performance of the application. After initializing their local database, mobile users enable or disable auto-synchronization from User Preferences > DB Synchronization. For more information read, *Siebel Remote and Replication Manager Administration Guide*.

To synchronize a mobile Web client

1. Start a Siebel application on the Mobile Web client machine.
2. Choose File > Synchronize Database.
3. Choose the synchronization options.

   For options, read *Siebel Installation Guide for Microsoft Windows*.
4. Click Synchronize.

19 Deploying the Application

This chapter covers the steps that NREC took to migrate its data and application from the test to production environments. Many of the tasks in this section were covered in previous chapters.

This chapter includes the following topics:

Migrating Data from the Test Environment to Production

After completing your testing, you are ready to move data from the test environment to the production environment.

The data you must move includes the following:

- **Tools configurations, including schema changes.** Use the Repository Migration Utility to move the latest Tools configuration from your test environment to your production environment. The procedure is the same as the one you followed in moving configurations from development to test. Refer to the instructions in Chapter 12, "Migrating to the Test Environment."
- **Modified files, such as SRF files, Web templates, image files, and cascading style sheets.** You must copy any changes you made to Web templates and related files, as described in Chapter 12, "Migrating to the Test Environment."
- **Transactional data, such as accounts, contacts, opportunities, and so on.** You have a snapshot of this data in your test environment, after having completed an EIM import. However, this snapshot may by now be out-of-date, as updates may have continued to be made to the data in your legacy application. Therefore, rather than copying this user data from test to production, do another EIM import from your legacy application—but this time directly into your production environment. For more information about EIM imports, read Chapter 13, "Using EIM to Load Data Into the Test Environment" and *Siebel Enterprise Integration Manager Administration Guide*.
- **Setup data, such as employees, positions, responsibilities, and so on.** You must copy this information from either your test environment or your legacy application to your production environment.
- **Program data, specifically Assignment Manager rules and Workflow processes and policies, and Personalization rules and expressions.** You must copy this information from your test environment to your production environment.

This chapter gives instructions for the last two items—migrating setup data and program data from test to production.

Moving Setup Data

You must move setup data into your production environment. Setup data includes information about employees, positions, responsibilities, and so on. Because the setup data in your test environment may be obsolete by the time you are ready to move into production, it is a good idea to move the data into the production server directly from your legacy application, using EIM for the import. For example, use the EIM_EMPLOYEE interface table to import employee data from your legacy application into the production server.

For information about using EIM, read *Siebel Enterprise Integration Manager Administration Guide*.

Moving Program Data

You must move program data, specifically Assignment Manager, Workflow and Personalization data from the test environment (where you created it) to the Production environment, as described in this section.

Moving Assignment Manager Data

You can use EIM to move your Assignment Manager data.

To move Assignment Manager data

- Use EIM, using the following interface tables.
  - EIM_ASGN_GRP
  - EIM_ASGN_WL
  - EIM_ASG_GRPDTL

For more information about EIM, read *Siebel Enterprise Integration Manager Administration Guide*.

Moving Workflow Data

Moving Workflow data consists of moving data about workflow processes and moving data about workflow policies and groups. Moving each of the two types of data requires you to follow a different procedure.

To export a process

1. Log in to the application in the test environment.
2. Navigate to Administration - Business Process > Workflow Processes > All Processes.
3. Select the process or processes you want to export. To select more than one process, press and hold the CTRL key while selecting the processes.
4. From the Processes list menu, choose Export Workflow.

   The XML workflow process definition appears.

5 From the XML dialog box menu, choose File > Save As.

6 Enter the file path, file name and the XML file name extension, and then click Save.

The process or processes are exported. If you selected more than one process to export, the processes are saved to one XML file.

NOTE: When exporting a process containing subprocesses, you must also export the subprocesses. Subprocesses are not exported automatically.

To import a process definition

1 Log in to the application in the production environment.

2 Navigate to Administration - Business Process > Workflow Processes > All Processes.

3 From the applet menu, choose Import Workflow.

4 Select a path and file name of the process to import.

5 Click Open.

The process imports with a status of In Progress.

NOTE: If a process definition of the same name exists in the target environment, the newly imported process definition's version number increments by one.

To move workflow policies and groups

- Use EIM, using the following interface tables:
  - EIM_WFM_GROUP
  - EIM_WFM_RULE
  - EIM_WFM_ACTION

For more information about EIM, read *Siebel Enterprise Integration Manager Administration Guide*.

Moving Personalization Data

Personalization rules, events, and actions can be exported as an XML file for later importation into another Siebel environment.

Exporting Personalization Data as an XML File

Personalization data rules, events, and actions are exported in one XML file.

To export personalization data as an XML file

1 Navigate to any Administration - Personalization view, for example Views.

2 Click the menu button, and then choose XML Export.

The File Download window appears.

3 Click the Save This File To Disk option button, and then click OK.

A dialog box appears prompting you specify a file name and storage location.

The default file name is personalization.ooo.xml.

4 Save the XML file.

Importing Personalization Data

Personalization rules, events, and actions can be imported from an XML file generated by a previous export. This process can take several minutes.

To import personalization data from an XML file

1 Navigate to any Administration - Personalization view, for example Views.

2 Click the menu button, and then choose XML Import.

A dialog box appears.

3 Click Browse.

A dialog box appears prompting you for a file name.

4 Choose a file, and then click OK.

5 Click Submit.

The personalization data is imported. When the process is finished, a message displays how many records had conflicts and were inserted, updated, and skipped.

Rolling Out to End Users

As described in "Migrating Data from the Test Environment to Production" on page 241, migrating data provides the production server with the latest data. The next logical step in the deployment is to roll out the applications to users. This section follows the NREC example to show how NREC rolled out the application to end users.

Rolling Out to Siebel Web Client Users

With the exception of the Web browser, the applications for Web Client users reside on the server. Therefore, as long as the client machines have a Web browser, you need only tell users the URLs to use to log in to for each application.

Rolling Out to Mobile and Dedicated Web Client Users

The Siebel Packager utility creates custom software installation packages for distribution to end users. This installer contains the Siebel Mobile and Dedicated Client executables and your custom configuration. The installer creates a Siebel environment on the end users' machines that duplicates the environment on the administrator's client machine.

After the Siebel Packager utility assembles the software into a single, self-extracting file, you can distribute this installer to your users. For example, NREC chose network distribution, as described in "Making Your Customized Installer Available to End Users" on page 248. However, Siebel Packager packages can also be distributed in the following ways:

- **Siebel Anywhere.** You can use Siebel Anywhere to distribute and execute the installation automatically for both dedicated and mobile users. However, because receiving Siebel Anywhere kits requires the Siebel client to be already installed on the user's machine, you cannot use Oracle's Siebel Anywhere for an initial rollout. You can use it only for upgrades. For more information, read *Siebel Anywhere Administration Guide*.
- **CD-ROM.** You can copy the customized software package onto CD-ROMs for distribution to end users.
- **Other methods.** You can distribute the program through any other file distribution mechanism, such as email or FTP.

Preparing to Use Siebel Packager Utility

You must complete the procedures in this section before using the Siebel Packager utility.

To prepare to use the Siebel Packager utility

1. Make sure the Siebel Packager utility has been installed on a client machine in your Production environment. This initial installation serves as a model for other installations, which may be performed by running the Siebel Packager utility against the initial installation.
   - Select Custom during client installation and be sure to select the Packager Utility option.
   - The rest of this chapter refers to the root-level directory of the client installation as *siebel_clnt_root*.
   - The Siebel Packager utility uses the files from this client installation (or another client installation, as specified when running the Siebel Packager utility) for the installer that it creates.

   NOTE: Be sure to customize this model Siebel client installation so that it is identical to how you intend to package it. When creating the custom installer, the Siebel Packager utility reproduces this model installation exactly.
2. Copy any changed files to the appropriate directories under *siebel_clnt_root*, or under the root directory of another installation that you use to create the custom installation package.

   Such files may include custom repository (SRF) or configuration (CFG) files, and Web template files.

3 Make sure that you have sufficient free disk space on the client on which the Siebel Packager utility is installed.

   During the packaging process, the Siebel Packager utility temporarily requires three times the amount of disk space required by the Siebel client software you are packaging, plus two times the disk space required by the third-party software you are packaging.

4 Create a custom siebel.ini file.

   The siebel.ini file controls the following behavior of the installer that the Siebel Packager utility creates:

   - Whether or not the Siebel client installation program checks that the appropriate versions of third-party software have been installed on the client machine
   - Whether or not the data sources that the Siebel application relies on are installed and how they are configured
   - Which installation dialog boxes users are presented at run time and which installation parameters they can specify

   The siebel.ini file used by the standard Siebel client installer prompts the user for parameters—not usually the desired behavior for an end-user installation. Create a siebel.ini file customized to your environment.

   Make your changes to the siebel.ini file located in the \siebel_client\packager\temp\package_name directory using a standard text editor. This file is largely self-documented to help you in customizing it. If you need more information, read *Siebel Installation Guide for Microsoft Windows*.

Preparing Siebel Components for Packaging

You can use the Siebel Packager utility to package the Siebel client installation software. Of the different installable components that make up a Siebel application, NREC chose to package the ones described in Table 14.

Table 14. Software Installed in the Test Environment

Component	Description
BIN	Siebel Executable Files (Binaries) located in the \siebel_clnt_root\bin directory, including the required DLL files, configuration files, and the Siebel executable. Include every file in this directory in your self-extracting installer, except the user preferences file, user.prf, and the session file, siebel.ses. You may want to replace the siebel.cfg file with your customized configuration file.
FONTS	Contains font files, located in the siebel_clnt_root\fonts directory.
LOCAL	Location of the local database, located in the siebel_clnt_root\local directory. Local databases are unique to individual users and must not be packaged.
LOCALE	Language-specific files, located in the siebel_clnt_root\locale. directory. Do not omit this module when creating a package.

Table 14. Software Installed in the Test Environment

Component	Description
MSGTEMPL	Message files used by the client, located in the siebel_clnt_root\msgtempl directory.
Objects	Object Configuration Template Files (Configured Objects) located in the siebel_clnt_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 Siebel Packager utility.
PUBLIC	Contains HTML, JavaScript and image files for Siebel Web Client, located in the siebel_clnt_root\public directory.
SQLTEMPL	Contains SQL scripts used by the Siebel Web Client, located in the siebel_clnt_root\sqltempl directory.
WEBTEMPL	Contains Web templates, located in the siebel_clnt_root\webtempl directory.

For information about other types of files that can be packaged, read *Siebel Installation Guide for Microsoft Windows*.

Running the Siebel Packager Utility

The Siebel Packager wizard guides you through the windows to help you create the custom client installer. NREC is deploying a single-language version of their application. This requires two separate installers—one for BASE and one for ENU (U.S. English). Both packages must use the same package name. You need a separate installer for each language component, including BASE. For example, if NREC were planning a multilingual roll out, it would require an installation of BASE as well as each additional language pack.

To create the BASE package

1. From the Windows Start menu, choose Programs > Siebel Client 7.7 > Packager.
2. Click OK at the Choose Setup Language screen.

   The Siebel Client Packager, Step 1 of 4 - Directory Definition dialog box appears.
3. Create the BASE package.
   a. Enter a Package Name. For example, NREC Package.
   b. Leave the default values in the remaining fields.
   c. Click Next.
4. In the Module Definition dialog box, choose Packager from the Modules list and click Remove and then Click Next.

   You do not want to include Packager itself as a module.
5. In the Packaging window, click Start.

   The utility displays progress information while it executes.

6 When the utility is complete, click OK and then Next.

7 In the Self-extracting Archive dialog box, click Exit to finish.

To create the language package

1 Go to start > Programs > Siebel Client 7.7 > Packager, and click OK at the Choose Setup Language dialog box.

2 In the Directory Definition screen:

   a Choose <LANGUAGE> in the Language Packs list (for example, choose ENU for U.S. English).

   b Enter a Package Name (for example, NREC Package).

   c Leave the default values in the remaining fields.

   d Click Next.

   The Module Definition dialog box appears.

3 Choose Packager from the Modules list, click on Remove to remove it, and then click Next.

   The Packaging dialog box appears.

4 Click Start to begin creating the package.

5 When the Siebel Packager utility is complete, click OK, and then click Next to continue.

6 In the Self-extracting Archive dialog box, click Exit to finish.

NOTE: The parameters used in the siebel.ini file need to be modified so that the installation on the end-user machines is “silent.” A silent install is one that does not display any windows or dialog boxes to the user. The siebel.ini file can be edited by clicking on the Edit siebel.ini file in the Siebel Packager utility. For more information about creating Base and Language Packages, and about editing the siebel.ini file, read *Siebel Installation Guide for Microsoft Windows*.

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. As described earlier, you can distribute your customized installation program to end users in a number of ways. Because NREC chose to distribute the installer using network distribution, this section describes that distribution method. For instructions about other distribution methods, read *Siebel Installation Guide for Microsoft Windows*.

To distribute a self-extracting installer over a LAN (if you created a self-extracting archive)

1 Put the self-extracting installer (packager_name.exe) in a network-accessible directory. Make sure that all users have access to this directory.

2 Send an email to users explaining how to copy and extract the package from this location. Consider telling users how to FTP the self-extracting archive to their machines and install it from there.

To distribute an installer over a LAN (if you did not create a self-extracting archive)

1. Put the package directory in a network-accessible location to which all users have access.
2. Send an email to users to tell them how to install the package from this location.

After or during this rollout, you are ready to provide user training on the applications.

Index

D

E

M

N

O

P

R

S

T

U

V

W