Siebel Business Process Framework: Task UI Guide

Siebel Innovation Pack 2016

April 2016

ORACLE

Contents

Chapter 5: Using the Development Environment to Develop a Task UI

Chapter 6: Creating Steps and Connectors

Chapter 7: Developing a Task UI

Chapter 8: Examples of Developing a Task UI

Chapter 10: Customizing Task UI

1 What's New in This Release

What's New in Siebel Business Process Framework: Task UI Guide, Siebel Innovation Pack 2016

No new features have been added to this guide for this release. This guide has been updated to reflect only product name changes.

2 Overview of Siebel Task UI

This chapter describes an overview of Oracle's Siebel Task-based User Interface (Siebel Task UI), which is part of the Siebel business process framework. It includes the following topics:

About Siebel Task UI

Siebel Task UI customizes business process automation to interactions that occur with the user. A *job task* is a multiple step, interactive procedure that the user performs to complete a business function. Creating a new account and then adding a new service request to the account is an example of a business function. A job task can include branching and decision logic.

A *task UI* is a user interface that guides the user in completing a series of steps in a job task. Similar to a wizard, the task view includes a playbar that allows the user to proceed through the job task in a stepwise, guided fashion. Siebel Task UI includes the following features:

- Guides the user through the job task in a stepwise fashion
- Supports forward and backward navigation through the job task
- Allows the user to pause and resume the job task

These features can support the user in completing a job task that is not familiar, and can help increase user efficiency. Siebel Task UI allows the user to switch between multiple job tasks, and increases efficiency with completing familiar job tasks, especially in multitasking environments and in environments that are prone to interruption.

A task UI includes a set of operations that a single user performs, such as adding an account. You can also configure a task UI as a step in one or more Siebel workflow processes. A task UI can be part of a business process that crosses multiple job roles, such as the workflow process that routes an expense report through multiple levels of review and approval. A task UI can help define integration with an external system, such as setting up and provisioning an account.

Comparison of Siebel Task UI to Siebel CRM UI

A task view typically includes fewer fields, controls, and applets than a standard view. The task view removes the complexity that the user does not require to finish a job task. A task UI simplifies the interface and reduces the potential for mistakes.

Standard views come predefined with Siebel CRM. The user accesses them to do a wide variety of business functions. It can include a form, list, tree, or chart applet, and the user can use a variety of navigation techniques to navigate through records. For example, the user can use a scroll bar, screen tab, or a drop down list. A standard view can include a superset of the fields and controls that the user must use to complete a business function. A standard view provides the user with significant functionality and capability to navigate a Siebel application and to modify data, so this view requires that the user possess knowledge and skills about how to use this view. More options are available, so it is more likely that the user might make a mistake.

Example of a Task UI View

Figure 1 includes an example of a view in a task UI in a Siebel application. The user can use this view to add an account and then add a service request to the new account. This task UI provides stepwise, guided direction with backward and forward capability.

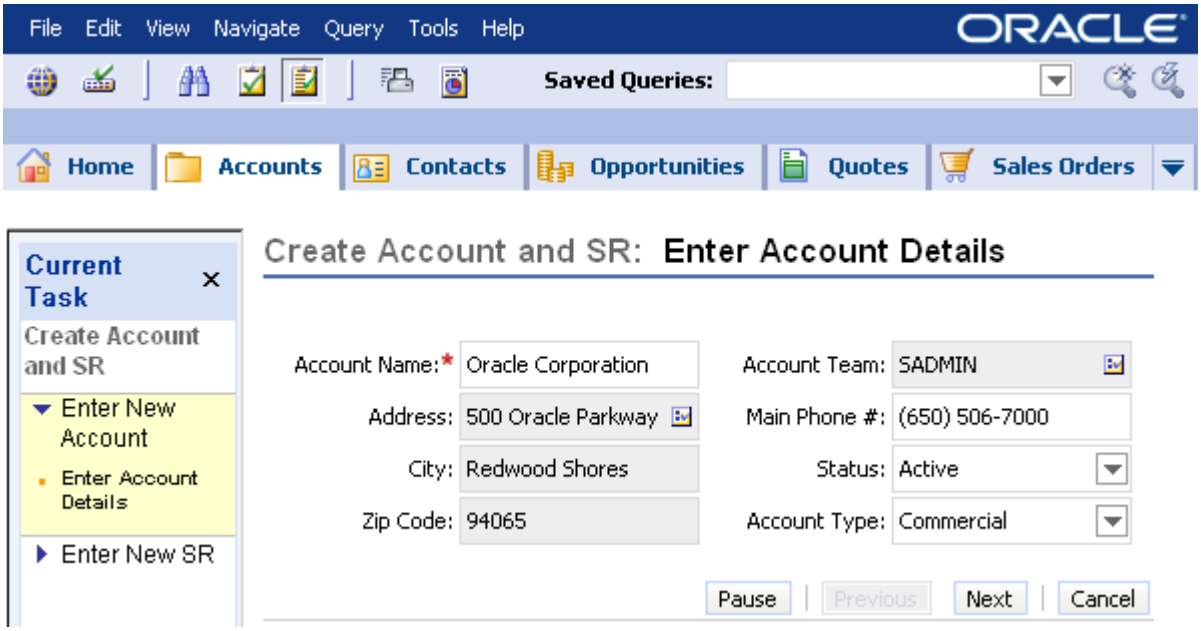


Figure 1. Example of a Task UI View

Example of a Standard View

Figure 2 includes the standard view that the user accesses to manage accounts. The power to use numerous fields that reference the underlying data, and the many navigation options available, result in an interface that requires more skill and knowledge to use accurately.

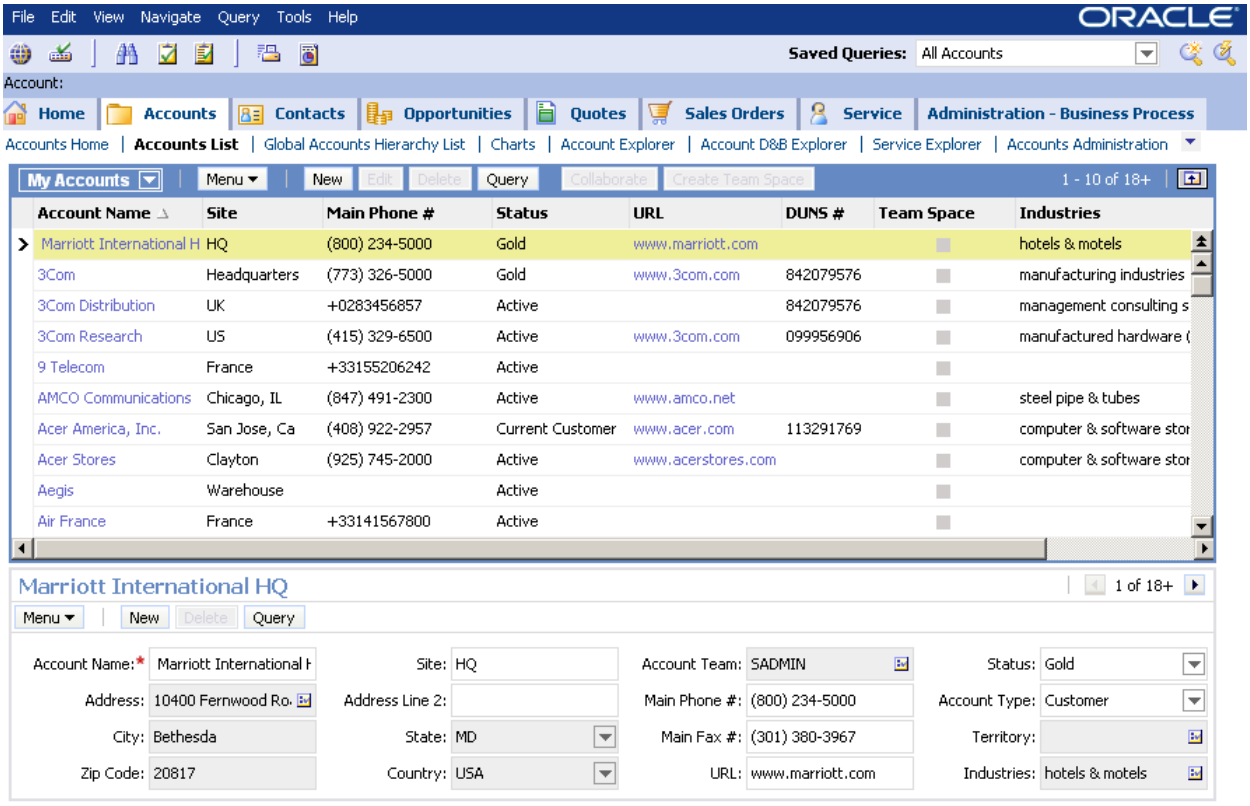


Figure 2. Example of a Standard View in a Siebel Application

Features and Benefits of Siebel Task UI

Siebel Task UI includes the following features and benefits:

- Provides direct forward and backward navigation through multiple screens and views.
- Improves efficiency through next step capability.
- Standardizes a business process.
- Provides guidance and supporting information that assists the user to accurately complete a complex business process. For example, the user can use communication features in a task UI to do a financial requirements analysis. Siebel Task UI can apply the logic required to do the analysis and then offer recommendations that the user can sell to the customer.
- Incorporates complex business logic to determine the required sequence of activities and content that the user requires at each step in a business process. For example, you can configure a task UI to display upgrade or upsell products according to the geography and current products that the customer uses.

- Uses validation to enforce rules that a business process requires. For example, a task UI can make sure a customer provides a written statement of fact within 14 days of opening a credit card dispute. Otherwise, Siebel CRM automatically closes the record.
- Integrates external data or services into a task UI. For example, calling an external credit program to determine creditworthiness of an applicant on a credit application, and then submitting identification information to the customer master database for validation.
- Coordinates multiple actions in a logical transaction that must finish successfully or that Siebel CRM must completely roll back. For example, transferring funds between financial accounts.
- Allows tracking and analysis of data for a task UI through tight integration with Oracle Business Intelligence.

Comparison of Siebel Task UI to Other Technologies

Siebel Task UI is the only technology that supports a long-running transaction when compared to other UI technologies. Siebel Task UI is closely integrated with Siebel Workflow. You cannot use Siebel Task UI with a customer facing application. It requires a High Interactivity (HI) client that supports ActiveX.

Table 1 compares features between Siebel Task UI and other UI technologies.

Table 1. Comparison of Siebel Task UI to Other UI Technologies

Feature	Siebel CRM (HI)	Siebel CRM UI (SI)	Siebel Task UI	iHelp	Smart Script	Web Channel
Employee facing	Yes	Not recommended	Yes	Yes	Yes	Not recommended
Customer facing	No	Yes	No	No	Yes	Yes
Encapsulates business logic	Some	Some	All	None	All	All
Integration with Siebel Workflow	Some	Some	Full	None	Some	Limited
Integration with Universal Inbox	Some	None	Good	None	Best	Some
Support for long-running transaction	No	No	Yes	No	No	No
Performance and scalability overhead	None	None	Some	None	Large	Some

Siebel Task UI Technology

Siebel Task UI can provide a desirable return on investment if you use it to support a business process, but it is not a universal answer to every business process. It is important that you carefully consider the factors discussed that this book describes, including the trade-offs and benefits that a task UI provides.

A task UI incorporates business logic and a guided user interface, improving performance and scalability when compared to a standard view that is designed for the advanced user. Siebel Task UI is an appropriate technology to use to support a business process that includes any of the following requirements:

- Employee facing
- Nontrivial
- Transactional
- Tight integration with a business process

Other UI Technologies

Table 1 on page 14 describes trade offs between features for other Siebel technologies. For example, SmartScript provides better integration with Universal Inbox, but at the cost of significant performance overhead and the cost of developing and maintaining a scripted solution. For this reason, SmartScript might be a more appropriate technology than Siebel Task UI to support a user who must review an expense report. Reviewing an expense report requires simple, nontransactional tasks that demand tight integration with Universal Inbox and is performed rarely enough so as not to jeopardize scalability of the entire system.

The Siebel CRM UI is more appropriate to support a simple task that an experienced user frequently performs. iHelp supports the Siebel CRM UI for simple tasks, tasks that the user frequently performs, or tasks that a novice user performs or a user who only performs the task occasionally. Web Channel is the preferred technology for an implementation that requires a customer facing user interface that includes a specific look and behavior.

Comparison of Siebel Task UI to Siebel Workflow

A task UI involves at least one task view step where a user enters data. A task UI guides the user experience to accomplish a specific business function. It is synchronous and tightly bound to a user interface. The interactions in a task UI are the result of a user who explicitly clicks the Next, Previous, Pause, and Cancel buttons to navigate through the task UI.

A workflow process is a business process that can encapsulate a series of tasks. A workflow process is not required to be synchronous and does not need to involve a user interface. A workflow process can start and stop in reply to a call from another system. If a workflow process includes an online component that involves engagement with a user, then you can configure Siebel CRM to call a task UI in the workflow process that supports this engagement.

About Predefined Task UIs

A *predefined task UI* is a task UI that comes already defined with a Siebel application. Before you create a new task UI, you can examine predefined tasks to determine if one exists that meets your design requirements, or that you can modify to meet your requirements. Predefined tasks are the tasks that Siebel Tools lists in the Tasks list immediately after you install Siebel Tools but before you add any new task. You can configure some of these predefined tasks.

Table 2 lists some predefined task UIs.

Table 2. Predefined Task UIs

Task Name	Business Object
Complete Field Task	Action
Execute Field Task Start to Finish	Action
Field Activity En Route	Action
Field Activity En Route Main Task	Action
Field Activity Execution	Action
Field Activity Execution Main Task	Action
Field Activity Invoicing	Action
Field Activity Invoicing Main Task	Action
Field Activity Main Task	Action
Field Activity Preparation	Action
Field Activity Preparation Main Task	Action
Replace Asset	Action
ADM Test Access Controlled Task	Expense
BPMRD – Create Expense Report Flow	Expense
CMEREF – Activation Order	Order Entry
Asset To Contract Task	Service Agreement
FS Asset To Contract Task	Service Agreement
FS Cover Asset SubTask	Service Agreement
FS Submit Agreement Sub Task	Service Agreement

Examining the Logic of a Task UI

You can examine the logic of a task UI.

To examine the logic of a task UI

1. In the Siebel client, use the FS Asset To Contract Task predefined task UI.

   To use this task UI, you must complete the procedures that are described in "Example of Modifying a Predefined Task UI" on page 107.

2. In Siebel Tools, in the Object Explorer, click Task.
3. In the Tasks list, query the Task Name property for FS Asset To Contract Task.
4. Open the Task Editor.

   For more information, "Opening the Task Editor" on page 51.

5. Examine the task flow that Siebel Tools displays in the Task Editor, and then compare it to the task view that Siebel CRM displays in the Siebel client:

   a. To identify a task view in the Siebel client, do the following:

      - Choose the Help menu.
      - Choose the About View menu item.
      - In the About View dialog box, note the value that Siebel CRM displays for the View.

   b. In the Task Editor, examine the task view step you identified in Step a.

About the Task Editor

The *Task Editor* is a graphical user interface in Siebel Tools that provides a declarative framework that helps you create a task UI. It allows you to use an object oriented programming language in an integrated development environment. You can drag and drop different types of steps and connectors from a palette to a canvas. It minimizes scripting, allows for dynamic modification, and encourages you to use a top-down development strategy. The Task Editor is integrated with the Siebel Workflow Editor. It simplifies the work of supporting a business process that is complex and has a long life in a Siebel application.

Example Task UI Flow in the Task Editor

Figure 3 includes an example task UI flow that Siebel Tools displays in the Task Editor.

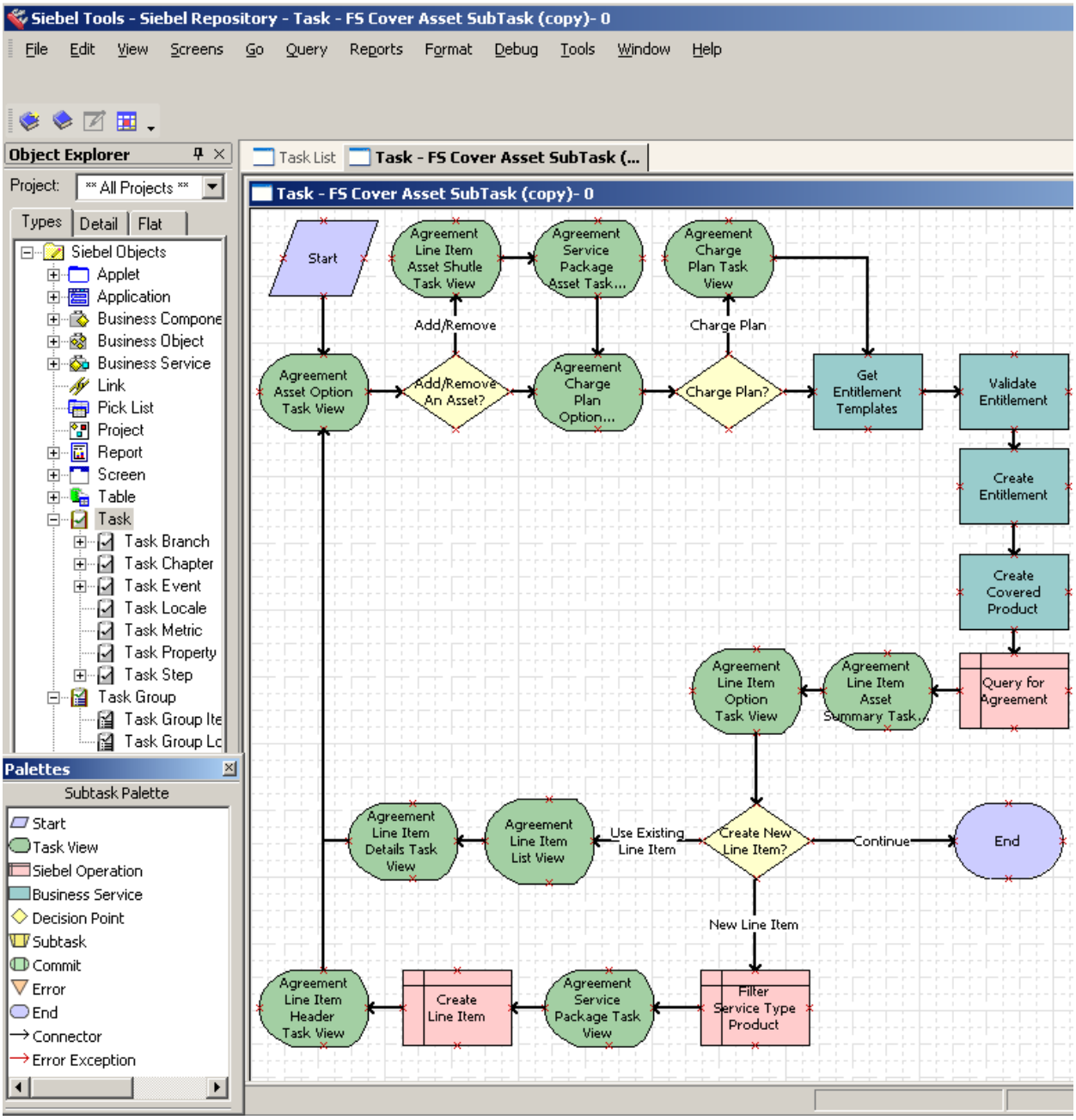


Figure 3. Example of a Task UI Flow in the Task Editor

Main Elements of the Task Editor

Figure 4 includes the main elements of the Task Editor.

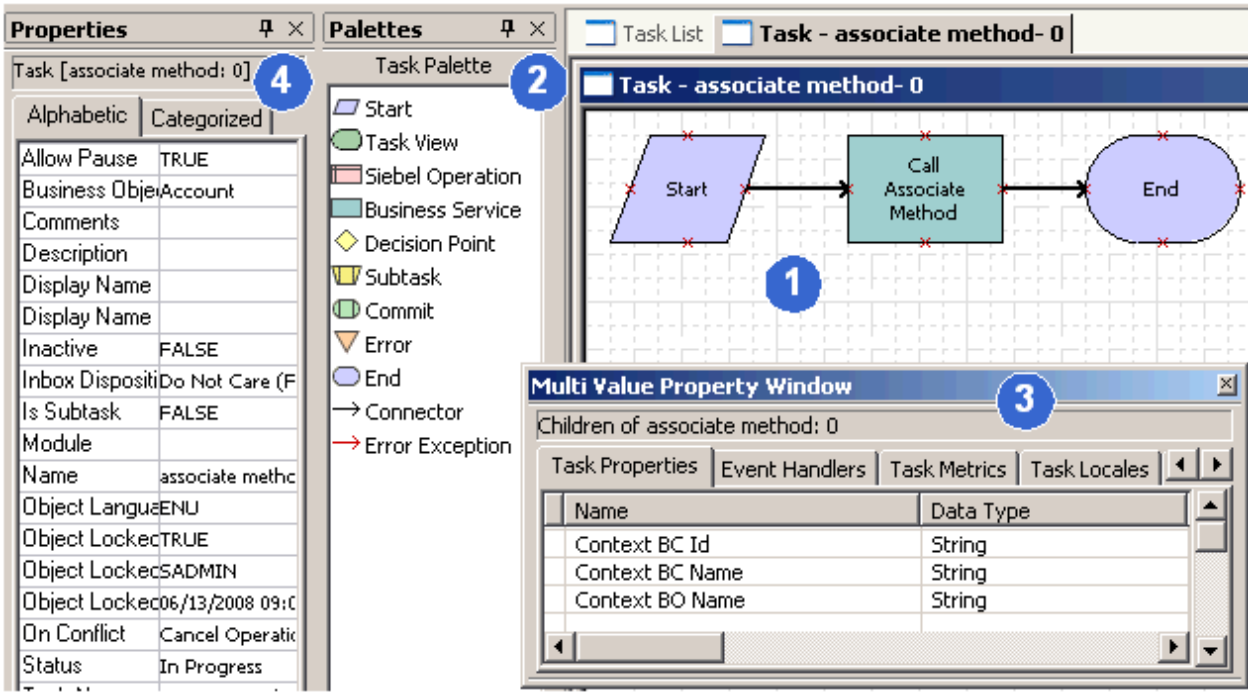


Figure 4. Graphical User Interface of the Task Editor

Explanation of Callouts

The Task Editor includes the following items:

1. **Task Editor Canvas**. A work area where you create the task UI. You right-click the canvas to access a menu that allows you to perform work that is related to creating this task.
2. **Task Palette**. A window that includes icons that represent the various step types that you can add to a task UI. To add a step, you drag, and then drop an icon from the palette to the canvas. For more information, see "Overview of Step Types" on page 66.
3. **Multi Value Property Window (MVPW)**. A window that allows you to define properties for the task UI or arguments for a task step. For more information, see "Arguments of a Task Step" on page 54.
4. **Properties Window**. A window that allows you to define properties for a task step or properties for the overall task UI. The window is context-sensitive in the following ways:
   - If you choose a step or connector in the canvas, then Siebel Tools displays properties for the step or connector in the Properties window.
   - If no step or connector is chosen in the canvas, then Siebel Tools displays properties for the overall task UI in the Properties window.

For more information, see "Using the Task Editor" on page 51.

Siebel Task UI and Siebel Workflow use the same or similar objects, such as a business service step or a decision point. Usage for the Task Editor is similar to usage for the Workflow Editor. For details about using the Workflow Editor, see *Siebel Business Process Framework: Workflow Guide*.

3 Scenario for Developing a Task UI

This chapter describes a scenario for developing a task UI. It includes the following topics:

Overview of Scenario for Developing a Task UI

The example in this topic describes how to develop a task UI. The scenario describes work that the following individuals perform:

- **Business Analyst**. Possesses detailed knowledge of the way the organization does financial planning and reporting. The analyst possesses detailed knowledge of various technologies, from personal computers to mobile personal devices, but possesses no experience using a formal programming language.
- **Application Developer**. Possesses a computer science degree and a strong technical background, including five years experience creating applications.
- **Usability Analyst**. Possesses a degree in art history and a technical background, including five years experience designing interfaces. In the last two years the interface developer created user interfaces for various customizations of a predefined Siebel application.
- **Information Technology Director**. Drives the business process management initiative at the organization that optimizes and automates business processes.

This chapter describes job roles in the context of a fictional organization. Job roles, job titles, and division of labor might vary significantly for your organization.

For more information about:

- General technical procedures, see "Roadmap for Developing a Task UI" on page 85.
- Specific technical procedures, see Chapter 8, "Examples of Developing a Task UI."

About Iterative Development

To minimize the risk of project failure, your development process can be iterative and incremental. Using an iterative technique means feedback from a phase can cause reiteration of a previous phases. For example, a significant performance issue might require a UI redesign that causes a reiteration of subsequent phases.

Using an incremental technique means that the best way to mitigate the risks of using new technology, such as Siebel Task UI, is to begin with a smaller scope, deliver it to customers, and then use customer feedback to incrementally create functionality. For brevity, this chapter does not fully describe iterations and incremental releases. However, most implementations include iterations and incremental releases, and it is recommended that you plan for them.

Example of Developing a Task UI

To develop a task UI, do the following tasks:

1. "Determining Improvement Requirements" on page 22
2. "Designing the Task UI" on page 23
3. "Developing the Task UI" on page 27
4. "Testing the Task UI" on page 28
5. "Implementing the Task UI" on page 29

Determining Improvement Requirements

This task is a step in "Example of Developing a Task UI" on page 22.

The IT director and business analyst meet for a brainstorming session where they list on a whiteboard the business processes for the organization that are in their purview. They then assess the best candidates for process improvement and determine that the main criteria include the following items:

- Return On Investment (ROI)
- Risk

The IT director and the analyst business do a careful examination, and then pick the expense reporting and reimbursement business process as a candidate for a task UI. They choose this business process because it includes a common, irregularly run job task that most employees perform.

This business process is relatively simple and well defined, making risk and investment low. The business analyst and the IT director are aware that the current way of handling this process, through email and spreadsheets, is labor intensive and error prone, making the process expensive and slow. This situation makes potential ROI very large.

The business analyst and the IT director estimate they can realize the following improvements:

- Cut costs by 30%.
- Speed up the process by 400% from the current median time of 12 days to 4 days from submission to reimbursement.
- Save the company $200,000 each year through better enforcement of company reimbursement policies.

To achieve these improvements, you can use various Siebel technology to automate a part of the process, and you can use a task UI to guide the manual part of the process and enforce rules.

Designing the Task UI

This task is a step in "Example of Developing a Task UI" on page 22.

To design the task UI, the business analyst begins by modeling the business process.

Modeling the Business Process

With low risk and high ROI, the executives at the organization approve the proposed project. The business analyst is now ready to model the business process, and names the business process Expense Reimbursement. This name reflects the fact that the objective of the business process is not simply for employees to report expenses, but for the company to reimburse employees for the valid expenses they incur while conducting business for the organization.

The next step is to separate the business process into separate activities. The business analyst answers the question *Who does what?* The answer to this question identifies the following roles:

- Submitter
- Reviewer
- Payer

The business analyst also identifies the following activities that can occur in the business process:

- Submit expense report, accomplished through a task UI
- Review expense report, accomplished through a task UI
- Pay expense report, accomplished through a business service

A favorable candidate for a business service must work as an independent entity to make sure reusability is possible. It is recommended that you use comments to document this business service to support other developers who must use the code. It must also follow a standard naming convention to make it easier for others to view the structure and understand what is happening in the code.

A favorable candidate for a task UI supports the entire business process and guides the user in performing the user role in this business process. The task must be clearly structured to make it easy for others to follow and understand.

The business analyst makes the following observations:

- Submitting an expense report is difficult to automate, so the analyst identifies it as a candidate for task UI.
- Reviewing an expense report is a candidate for automation. To minimize the risk of fraud, the business analyst notes that different users can repeat the task UI through an approval chain.
- The Oracle Accounts Payable (AP) application that the organization currently uses can automate the pay expense report, so the analyst notes it as a possible solution.

Creating the Draft Model

To finish the first iteration redesigning the business process, the analyst defines it as a workflow process, and then meets with the usability analyst and the application developer to gather feedback. The business analyst uses the Workflow Editor in Siebel Tools to model the business process as a long-running workflow process. Figure 5 includes the workflow process that is not yet executable. The illustration serves as a way to communicate the design intention with other team members when accompanied with notes from the analyst.

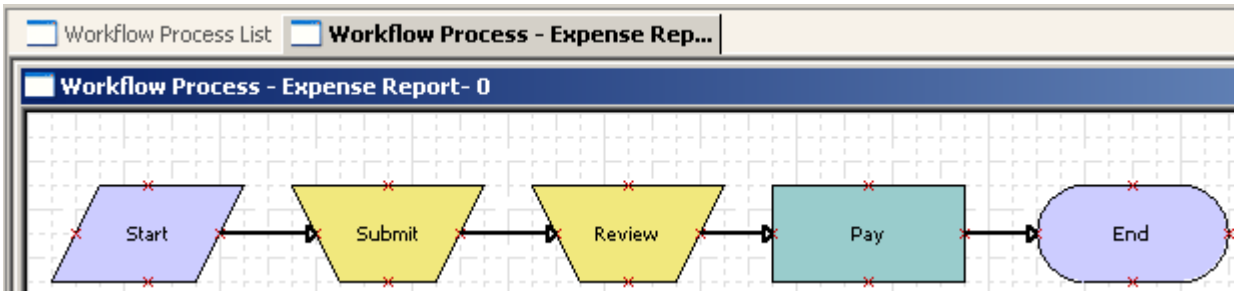


Figure 5. Draft of a Workflow Process in Siebel Tools

Other team members notice that the model is not finished because the business logic for the approval is not clarified. For example, when is a single approval sufficient, and when are more approvals required for a single expense? Also, the team notices that the model does not cover the situation where a review step finds that the expense report is ineligible and Siebel CRM must reject it.

The business analyst modeled the workflow process in Siebel Tools, so the application developer can refine the model. Figure 6 includes the revised prototype that the developer can refine into an executable workflow process. At this point, the developer and the usability analyst possess a thorough understanding of the business process for expense reimbursement.

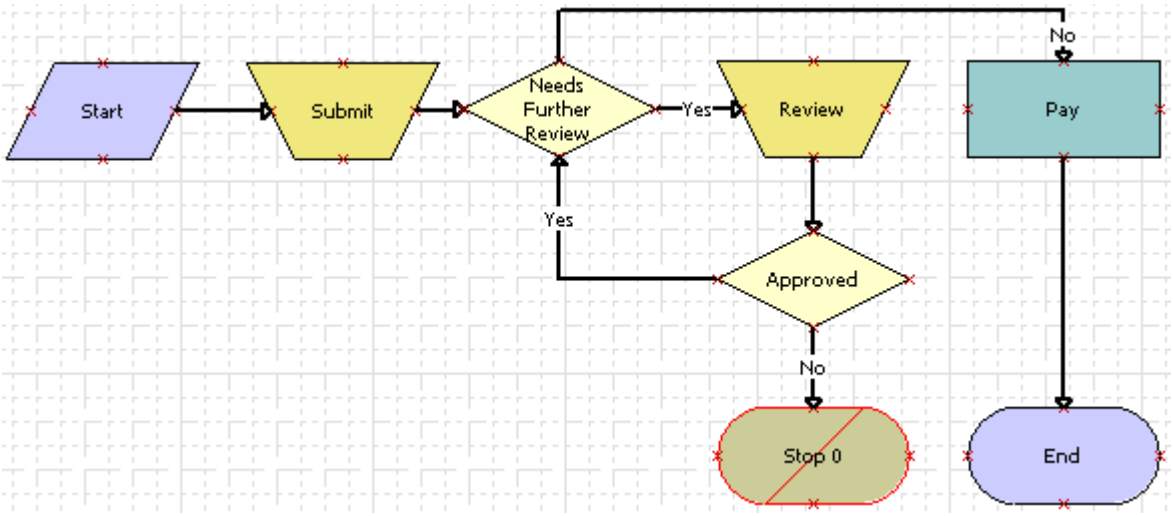


Figure 6. Refined Workflow Process in Siebel Tools

Next, the application developer creates an executable workflow process.

Creating an Executable Workflow Process

The business analyst used Siebel Tools to model the business process, which saved the application developer time because the developer can use the model that the analyst created as a starting point for iterative refinement. The application developer notices that, although logically it belongs in the business process model, the team must remove the submission task UI from the executable long-running workflow process because it must start the workflow process. The submission task must pass the ID for the expense report as an input argument. The developer decides to reuse the Object ID process property that the developer defines as an input argument to the workflow process. The developer communicates this refinement to the business analyst.

Next, the developer considers the Decision step that determines if further review of the expense report is required. The developer determines if it is best to call a business service that is dedicated to enforcing the review decision. The developer examines the Review task UI and realizes that Siebel CRM must assign it to a reviewer before Siebel CRM can instantiate it. The reviewer can use a business service step to call the Assignment Manager. This step resides immediately upstream of the review task. Figure 7 includes the executable workflow process.

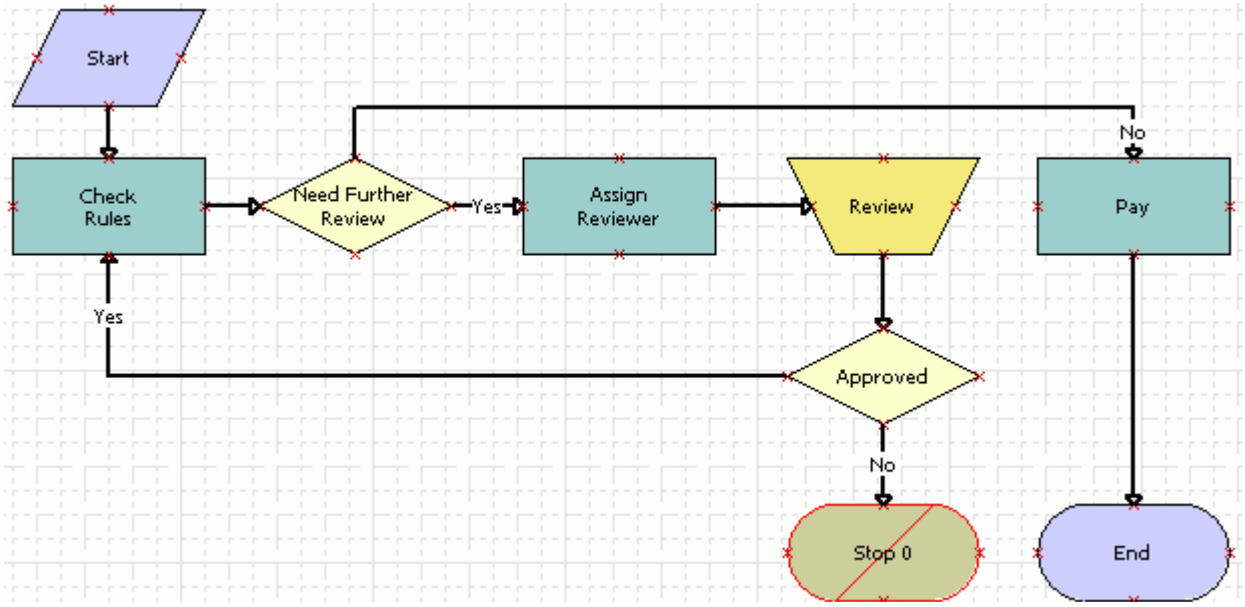


Figure 7. Executable Workflow Process in Siebel Tools

These revisions to the draft business model help to illustrate the differences that exist in the level of abstraction between a business process model and an executable workflow process. The application developer continues to refine the workflow process. For example, creating conditional branches, process properties, input arguments and output arguments, and then testing and deployment, until the team is ready to implement the workflow process in a production environment. For more information, see *Siebel Business Process Framework: Workflow Guide*.

Next, the business analyst identifies the context for the task UI.

Identifying the Context for the Task UI

The business analyst must refine the task UI. The analyst begins by identifying the context for the task. The application developer notifies the business analyst that the submission task UI starts the long-running workflow process, so the analyst realizes that the user must manually start the submission task in a standard view. The analyst also realizes that most employees will frequently do this job task.

For these reasons, the analyst decides the development team must add the submission task UI to the Common Tasks task group, and concludes that the submission task does not require context passing. The context is the current record. For example, the record being submitted. The task UI that submits the record does not need to pass information about the submitted record to the next step in the business process.

The user must be able to frequently pause this task UI. For example, to provide the user a moment to find required documents. So the analyst decides that the task must use the description for the expense report as the value of the Context field in the Universal Inbox. This configuration allows the user to distinguish between different instances of the same task.

The analyst notes that Siebel CRM opens this task UI from the long-running workflow process, so the user opens it from the Universal Inbox. For this reason, the analyst does not add the review task UI to a task group. The review task does require that Siebel CRM pass an expense report number as an input argument. A Boolean value named Approved seems to be an appropriate output argument. The appropriate context for the Universal Inbox is a concatenation of the total amount for the expense report and the name of the submitter. For example, $450.00 from Aaron Jones.

Next, the usability analyst designs the task UI.

Designing the Task UI

The usability analyst now identifies activities in each task UI, and creates a prototype for the task. The Submit Expense Report task UI includes the following activities:

1. Create expense report header.
2. Create expense items.
3. Review and submit the expense report.

The usability analyst uses the Task Editor in Siebel Tools to model the task UI. The analyst defines the three activities as three separate task view steps. The analyst is not familiar with the View and Applet editors, so the analyst enters only the view step names without linking the view steps to the task views. These views do not yet exist.

The other task UI that reviews the expense report is a simple task UI. It requires only the Review Expense Report task view step.

For an example that includes view mockups that assist with designing a task UI, see “Example of Developing a Task UI That Assists with Creating Multiple Opportunities” on page 140.

Next, the usability analyst refines the task UI.

Refining the Task UI Design

At this point, the usability analyst is ready to work with the application developer, who is familiar with the Expense Report business object and the underlying data model for the objects. Together, they sketch the view layouts on a whiteboard. The application developer identifies the business components and applets that the Submit expense report and the Review expense report task UIs require.

The team must determine whether the user must create all expense items in the same view, or create each item in a separate view. The business analysis indicates that most users do this job task frequently. The business analyst and the usability analyst agree that productivity is more important than UI simplicity, and they conclude that the user must create all items in the same view. As an advantage of this top-down technique, the usability analyst realizes the possibility of reuse for two applets in two similar views. The third view that resides in the submission task UI displays the same data that the view that resides in the review task UI displays.

Developing the Task UI

This task is a step in "Example of Developing a Task UI" on page 22.

To begin developing the task UI, the application developer creates the task.

Creating the Task UI

The application developer does the following work to create the task after the team reaches consensus on the task view layouts for the task UIs:

1. Makes required additions and modifications in the configuration for the business logic.

   This configuration might include the business object, business component, links, picklists, and so on.

2. Creates or reuses the required applets.
3. Creates or reuses the required task views.

Next, the application developer refines the task UI.

Refining the Task UI

The application developer does the following work to refine the task UI:

1. Add links to the new task views.
2. Adds an insert Operation step before the first view in the submission task UI.
3. Adds an update Operation step after the task view step in the review task UI.
4. Reviews and updates properties and multivalue properties for task steps.
5. Makes sure that the task UI contains no validation errors or warnings, including setting the Instance ID task property that the business analyst identified.

Next, the application developer creates the task group and deploys the task UI.

Creating the Task Group and Deploying the Task UI

The application developer does the following work:

1. Creates a relationship between the submission task UI and the Common Tasks task group.
2. Recompiles the SRF (Siebel Repository File) with the new UI configuration, including the task group update.
3. Clicks the Publish and Activate button to deploy the submission task UI to the development environment.

Next, the application developer sets up access control.

Setting Up Access Control

The application developer does the following work:

- Makes sure the responsibility for the test user possesses the visibility to run the submission task UI. For more information, see "Adding a Responsibility to a Task UI" on page 166.
- Makes sure the Siebel client displays the customizations in the Siebel Mobile Web Client that Siebel Tools opens after the developer publishes and activates the customizations.

Testing the Task UI

This task is a step in "Example of Developing a Task UI" on page 22. The application developer begins with unit testing.

Unit Testing

Siebel CRM displays the Submit Expense Report link in the context pane. If the user clicks this link, then Siebel CRM displays the first view of this task UI. The application developer is excited, but this joy is spoiled by the fact that the developer failed to include an Expense Description field in the Expense Header Applet. Several iterations later, the developer demonstrates the submission task UI to the business analyst, who is pleased, but who also notes that there are a few details that require refinement. Finally, version 16 of the submission task and version 22 of the SRF is ready for integration testing.

The review task UI is more difficult to unit test because it requires tight integration with a long-running workflow process. The task UI is relatively simple, so the application developer defers unit testing for the review task until the team integrates it with the long-running workflow process.

Next, the application developer does integration testing.

Integration Testing

Testing for the entire business process includes the following work:

- Publishing and activating the latest versions of the task UIs
- Publishing and activating the latest version of the long-running workflow process

- Compiling the latest UI configuration to the SRF, including business layer configuration modifications

To test the configuration in the development environment, the application developer uses the Siebel Server and the Siebel Web Client because a long-running workflow process cannot run on the Siebel Mobile Web Client. The developer uses the Task Debugger to analyze any defects that exist in the task UI logic.

Next, the application developer performs system testing.

System Testing

If the configuration appears to satisfy the business requirements, then the application developer uses Application Deployment Manager (ADM) to migrate the entire configuration to a test environment. ADM typically includes migration of new or updated LOVs that a task UI uses. The developer verifies functionality, and then tests performance and scalability.

Implementing the Task UI

This task is a step in "Example of Developing a Task UI" on page 22.

The team implements the new task UI and the long-running workflow process in the production environment. This work is best performed as part of a system wide configuration upgrade, because new functionality typically requires SRF modifications and some server downtime.

4 Siebel Task UI User Interface Elements

This chapter describes some of the user interface elements you can define in a task UI. It includes the following topics:

About the Task Pane

This topic describes the task pane. It includes the following topics:

The *task pane* is a user interface element in a task UI that provides a way to organize tasks. It also allows the user to start a task UI and to monitor progress through a task UI. Technically, the task pane provides one of the three possible panels that Siebel CRM can display in the action pane.

The *action pane* is a user interface element that displays the task pane, the iHelp Pane, and the Search Pane. If Siebel CRM displays the task pane, then it displays the tasks button that controls the appearance of the pane. It displays this button in an active state.

If a task UI is currently running, then the task pane displays the current task pane. Otherwise, it displays the context pane.

Context Pane

The *context pane* is a type of pane that displays in the task pane when the user uses a Siebel CRM client. Siebel CRM displays a task UI in the context pane in the context of the standard view. Siebel CRM registers each task that the context pane displays with a standard view.

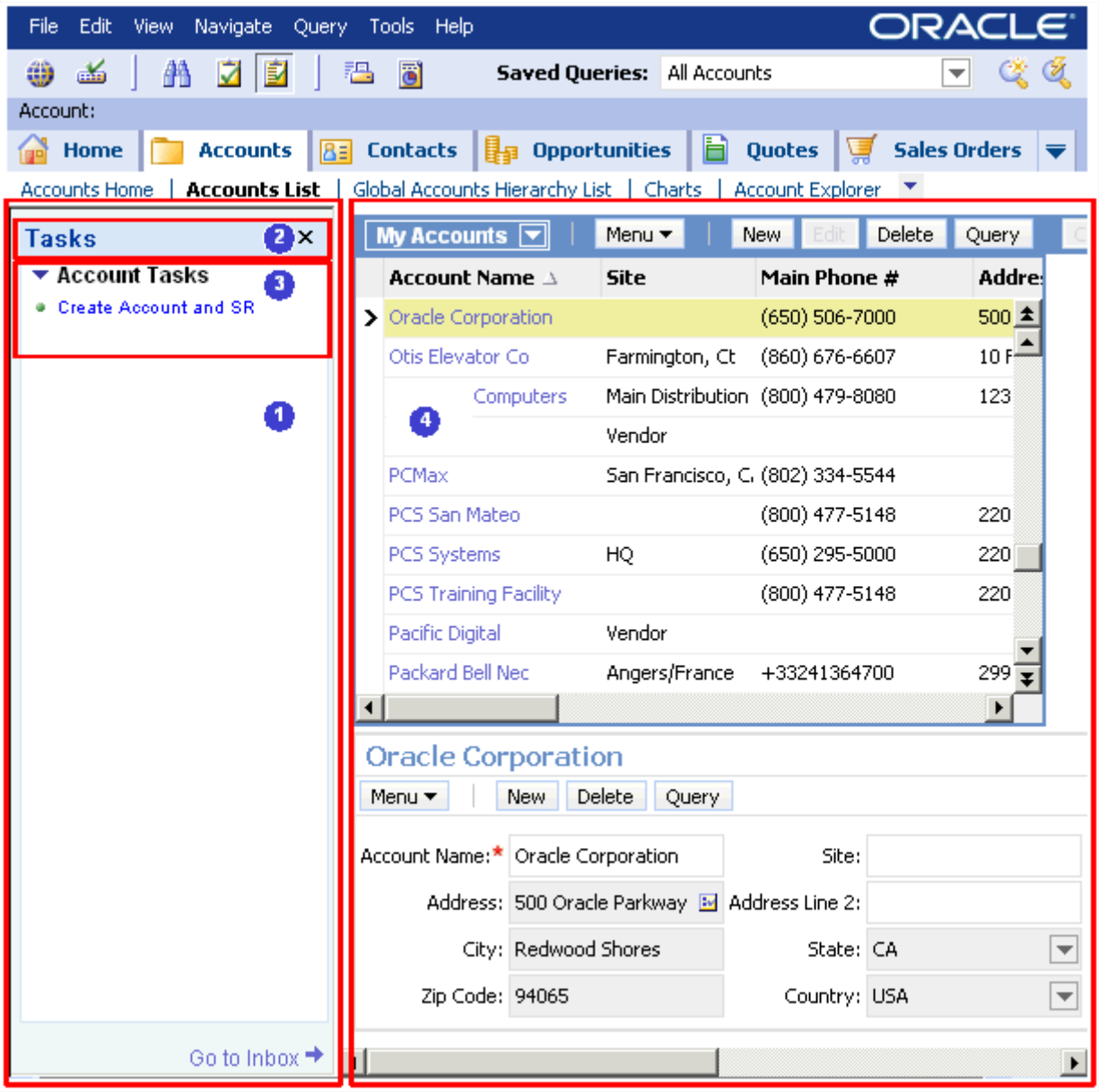


Figure 8. Example of the Context Pane

Explanation of Callouts

Figure 8 illustrates how user interface elements display when Siebel CRM displays the context pane:

1. **Context pane**. A type of pane that Siebel CRM displays in the task pane when the user is using the standard view.
2. **Context pane header**. In this example, the header for the context pane is Tasks.
3. **Task group**. In this example, the display name for the task group is Account Tasks. The task group item is Create Account and SR. It is a link that starts the task UI when the user clicks it. For more information, see "Task Group" on page 36.
4. **Standard view**. Provides the context. The standard view lists the task UIs that Siebel CRM registers for this view and that the user can open. For example, if the context is account, then the context pane lists tasks that the user can open in the account context.

   Siebel CRM does not restrict the context pane to the context. Logically, it makes sense to organize task UIs according to the standard view context, but to also allow the user to register a task in a different context. For example, a standard view for an account context can include a task that resides in the task pane that is related to an opportunity.

How the User Can Manage the Footprint That the Task Pane Consumes

The user can do any of the following to close the task pane:

- Click the Tasks button.
- Click the Close (x) icon in the upper right corner of the action pane.

Closing the task pane can be useful when screen space is critical. The user can collapse or expand task groups to manage the footprint that the task pane consumes. If the user clicks the name of a task UI in the task pane, then Siebel CRM opens a new task instance, displays the first view in the task, and replaces the header Task at the top of the task pane with the Current Task.

Current Task Pane

The *current task pane* is a type of pane that helps the user navigate through a task UI. Siebel CRM displays it in the task pane after the user clicks a task link in the context pane.

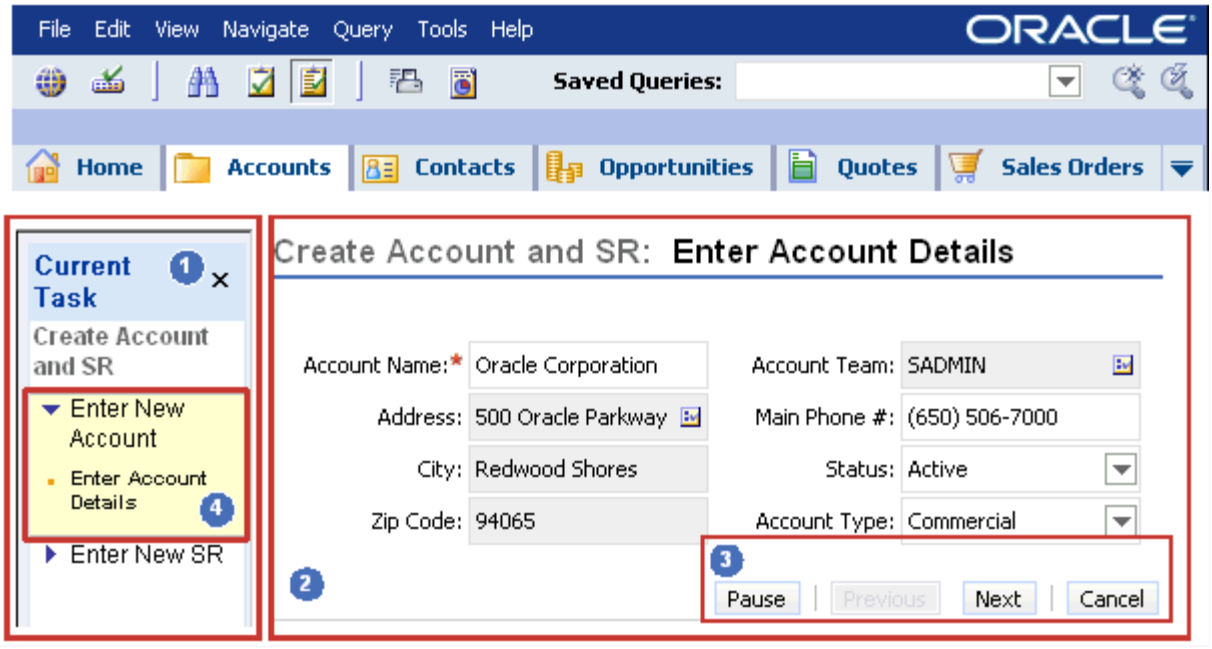


Figure 9. Example of the Current Task Pane

Explanation of Callouts

Figure 9 includes an example of the current task pane. In this example, Enter New Account is the task chapter, and Enter Account Details is the current task view:

1. **Current task pane**. A pane that helps the user navigate through a task UI.
2. **Task view**. A type of view that includes task applets or Siebel CRM applets. For more information, see "About the Task View" on page 39.
3. **Applets**. A Siebel CRM applet, task applet, or task playbar applet in a task view. For more information, see "Task Applet" on page 40 and "Task Playbar Applet" on page 41.
4. **Task chapter**. A list of task steps that Siebel CRM groups in a common chapter name. For more information, see "Task Chapter" on page 38.

If you define no chapters in the task UI, then the current task pane displays only view steps. Upon backward navigation, Siebel CRM still displays view steps that are downstream of the current step. If, on subsequent forward navigation, the user enters a different branch in the task flow, then Siebel CRM clears the view steps that it displayed for the branch that the user originally navigated.

About Fast Backward and Fast Forward Navigation

The user can use *fast backward navigation* and *fast forward navigation*, which are navigation features that allow the user to navigate back and forth through a task UI without having to repeatedly click Previous or Next.

For example, assume a user starts a large task UI and is currently working in the eighth task view. The user realizes that information in the first view requires a correction. If the user double-clicks a link in the Current Task pane, then Siebel CRM displays the view that this link references.

If the user uses fast backward navigation, then note the following:

- If the user modifies any data in the view that Siebel CRM displays, then the user must click Next from this point forward in this task UI. For example, if the user attempts to use fast forward to navigate back to the eighth view, then Siebel CRM displays a message that is similar to the following:

  ```
  You have already started modifying data on the view. You must proceed to the
  subsequent view through the Next button on the play bar.
  ```

- If the user does not modify any data, then the user can double-click any link in the Current Task pane to navigate directly to a view.

The user can use fast backward or fast forward navigation with various task UI features, such as the following:

- Pause a task.
- Transfer a task to another user.
- Navigate to another task UI chapter.
- Navigate to a subtask.

Siebel CRM enables fast backward navigation and fast forward navigation, by default.

The user can use fast backward and fast forward navigation only on views that the user has already visited in a task instance.

About the Task Progress Indicator

A task includes a Task Progress Indicator that displays the number of screens completed and the percentage of the task complete. For example, the indicator might display the following information:

Percentage Complete: 40%

Siebel CRM displays the Progress Indicator in the lower portion of the Current Task pane. It displays the percentage of views that the user has completed as part of the total number of views in the current task UI.

How Siebel CRM Determines the Percentage of Task Completion

To calculate the percentage of task completion, Siebel CRM determines the following items for each view in the task UI:

- The number of views the user completed

- The total number of views in the task UI

This percentage of task completion is the number of views completed divided by the total number of views that exist in the task UI. For example, if the user completes five views, and if a total of eight views exist in the task UI, then the percentage of the task complete is 62.5%, that Siebel CRM rounds up to 63.

To avoid run-time overhead, Siebel CRM does most of the calculations it requires to determine task completion during application design.

Decision steps, subtask steps, and loops can contribute to task complexity. To handle this complexity, Siebel CRM stores the number of views that exist in the longest possible path to reach the end step from the current view, including subtasks and alternative branches. It stores this number for each task view step. The only run-time calculation that Siebel CRM does is to track the number of views that the user completes.

For more information, see "Enabling the Task Progress Indicator" on page 208.

Task Group

A *task group* is an object type that groups task UIs that Siebel CRM displays in the context pane. The task group is a required UI element that includes a list of links to related tasks and commands that Siebel CRM displays in the context pane. You can configure a task group to display in a specific view, or across multiple views in an application.

You can use a task group to assign tasks to views according to their grouping. The task group also allows you to configure the following functionality:

- The user can open the task only with record context.
- The user can open the task without record context.

The top-level task group object references task UIs that the user can open if the business component is attached to an applet that is active and visible.

The *view task group* object type is a child of the view object type. You can use it to define the task group that Siebel CRM displays in the task pane when it displays the current view.

To view an example of a task group, see Figure 8 on page 32. For more information, see "Creating a Task Group" on page 92.

Display Name of a Task Group

You can use the Display Name property on the task group object to configure the display name of a task group. If you configure a task group across multiple views, then the task group can include the same display name. Siebel CRM gets the display name for a task UI from the task object. While a group display name is the same for all tasks in a task group, an individual task can include the same display name regardless of the task group that it references.

Task Order in Task Groups

The *task group item* is an object type that allows you to control the order that Siebel CRM uses to display task items. It is a child of the task group. Siebel CRM uses the following priority to sort task items in a task group:

1. In ascending, numerical order according to the Sequence property of the task group item.
2. In ascending, alphabetic order according to the display name of the task group or task group item. Siebel CRM does not require a unique sequence number.

Using a Task Group Across Multiple Views and Applications

You can specify the Siebel application where Siebel CRM displays a task UI. You can also add it to a view, making it available only when Siebel CRM displays this view, or you can add it to the Task Pane View, making it available globally throughout a Siebel application.

Using a Context Business Component with a Task Group

You can use the Context Business Component property of the task group item to associate a business component with a task UI. To run a task, the task can require that the context business component be the business component that an applet references. This applet is part of the current view.

You can configure a context business component for each task group item. However, associating a context business component for each task group item is meaningful only if the item that you associate is part of a task group that requires context. You can only group together task UIs that are context-sensitive.

If these requirements are met, then clicking the context-sensitive task group item causes the task UI to instantiate the task UI, and then pass it the current record of the context business component. If the business object that the task references matches the business object that the current standard view references, then the task shares the business object instance and business components with the standard view. Otherwise, you must use the following system task properties to properly instantiate the business object that the task UI references:

- Context BO Name
- Context BC Name
- Context BC Id

Using the Context Business Component and Task Display

If the Context Business Component property of the task group item is defined, then you must make sure Siebel CRM instantiates it in the current standard view, and make sure the Require Context BC property of the task group includes a check mark.

Siebel CRM uses the Require Context BC property to indicate that a task group can include a task UI that is context-sensitive. If the task group where the task resides is a candidate to render, then the context business component can constrain the display of a task in the following situations:

- If the task UI does not require a context business component, then Siebel CRM displays it.

- If the task UI requires a context business component, then Siebel CRM displays it only on the view where the context business component is the business component that one of the applets references.

Task Chapter

A *task chapter* is a list of steps that reside in a task UI that Siebel CRM groups under a common chapter name. A task chapter is an optional UI element that can provide the user with a map of what lies ahead in completing the task UI, and what work the user finished in this task UI. The task chapter allows you to configure a logical grouping of task steps and to display the chapter name with the names of task views in the current task pane.

If a task chapter is defined when a user first opens a task UI, then Siebel CRM displays only the chapter names of the task UI that it displays in the current task pane. If Siebel CRM runs the first task view step of a chapter, then it expands the chapter name in the current task pane. It does this to display the names of task views that the user finished. In the current chapter, the current task pane displays view steps that the user visited in the order of visitation, with the current view step displayed in bold.

If you assign one step to a chapter, then you must assign all other steps in the task UI to a chapter. Although you can create many chapters for a task, it is recommended that you only create as many chapters as is required.

Figure 9 on page 34 displays a task UI named Create New Account that includes two chapters. The chapter name provides details for the following parts of the task UI:

- Enter New Account
- Enter New SR

The example illustrates that the user is currently performing the Enter Account Details step.

For more information, see “Creating a Task Chapter” on page 201.

Using Color with a Task Chapter

You can use color in the Task Editor to differentiate between chapters that Siebel Tools displays. Steps that share the same color reside in the same chapter. Each chapter includes a Color property in the Multi Value Property Window where you define a chapter.

Contiguousness Requirements for a Task Step

If you assign multiple task UI steps to a single chapter, then the task steps must be contiguous. For example, if the two steps reside in different chapters, such as Chapter 2, then you cannot assign the fourth step to the same chapter as the first step, Chapter 1. Step 4 must reside in Chapter 2 or Chapter 3. Likewise, forward navigation that navigates to a previous chapter is not possible. This configuration is known as a *chapter cycle*. Validation rules prevent the flow from moving in this direction. However, you can loop task steps between chapters. For example, a step in Chapter 2 can loop back to a step in Chapter 1 after a decision point.

Using Chapters with a Subtask

You can use the Task Editor to assign a task UI step to a chapter for a parent task but not for a subtask. If you assign a chapter to a subtask step, then Siebel CRM groups the steps that reside in the subtask under the same chapter header of the subtask step. It does this when it runs the subtask.

About the Task View

This topic describes the task view. It includes the following topics:

A *task view* is a type of view that can include Siebel CRM applets and task applets that allow the user to view application data that the task UI displays, and to allow user input. The task view includes a playbar applet that includes buttons, such as Next and Previous. These buttons allow the user to proceed through the task in a stepwise, guided fashion. A task view can include optional elements to improve usability, such as a radio button group or applet message. The task view is a required element. To view an example of a task view, see Figure 9 on page 34.

The following categories describe the more common task views:

- **User decision or user choice.** A view where the user chooses from a set of options that determine how the task UI proceeds.
- **Single object data entry.** A form view that includes a built in new record.
- **Multiple object data entry.** A list view that includes a New button.
- **Task review.** A view that displays data that the user entered so far in the task instance, with options for modification.
- **Summary view.** A view that Siebel CRM displays at the conclusion of a task UI.

For more information, see the following topics:

- "Creating a Task View Step" on page 67
- "Creating a Task View" on page 88
- "Guidelines for Designing User Interface Elements" on page 213

Templates That You Can Use with a Task View

You can use a predefined web template that the IsInTask clause modifies. You do this configuration when you define a task view. This web template controls behavior that is specific to a task view. The web template does not display any of the following items:

- Applet title
- Applet menu

- Screen menu
- Thread applet
- Thread field
- Thread title

Each task view includes a title in the top left corner that indicates to the user that the view is not a standard view.

A task UI does not require a specialized task view template. You typically use the following templates for a task view:

- View Detail (Parent with Pointer)

  You can use this template for your base task view. It allows you to customize your task UI and to place applets side-by-side to provide the user a more structured view.

- View 1 Over 2 Over 1

Style Sheet for a Task UI

You can use the main_task.css style sheet to define the look and feel of a task UI. If you modify the style sheet, then Siebel CRM applies these modifications globally to all your tasks.

Task Applet

A *task applet* is an applet that Siebel CRM uses in a task view. It is an optional UI element that interacts with a transient business component. It supports a specific, finite task UI. For more information, see "Overview of Transient Data" on page 96.

A *predefined applet* is a Siebel CRM applet that Siebel CRM uses in a standard view. It interacts with fields in a predefined business component. A task view can include task applets and predefined applets.

Comparison of the Task Applet to the Standard Applet

A task applet differs from a predefined applet in the following ways:

- A task applet references a transient business component. Siebel CRM uses a transient business component to display data from a task UI that it discards when the task ends. For example, data that it displays to allow the user to choose values that create the branching condition of a task. The next task step that Siebel CRM displays might vary, depending on the value of the transient data that the user chooses.
- A task applet includes the CSSSWEFrameTask specialized frame class.
- A task applet is of type Task. A traditional applet is of type List, Form, Tree, or Chart.
- A task applet uses only on a grid Web template. For more information, see *Configuring Siebel Business Applications*.

Operations That Siebel CRM Allows in Applets

Siebel CRM comes predefined to restrict the operations that the user can perform in an applet in a task view. These restrictions prevent the user from modifying records and altering the record context in a way that might cause a problem with the underlying task logic. For more information, see "Specifying the Operations That Users Can Perform in an Applet" on page 200.

Table 3 describes the operations that Siebel CRM restricts for each type of applet.

Table 3. Operations That Siebel CRM Allows in Applets

Operation	Form	List	Task List	Pick	MVG	Association
Query	Allowed	Allowed	Allowed	Not Allowed	Not Allowed	Not Allowed
Insert	Allowed	Allowed	Allowed	Not Allowed	Allowed	Not Allowed
Delete	Allowed	Allowed	Allowed	Not applicable	Not Allowed	Not Allowed
Update	Not Allowed	Allowed	Not Allowed	Not Allowed	Allowed	Not Allowed
Next Record	Allowed	Not Allowed	Allowed	Not Allowed	Not Allowed	Not Allowed
Associate	Not applicable	Not Allowed	Not applicable	Not applicable	Not Allowed	Not applicable
Merge	Not applicable	Allowed	Not applicable	Not applicable	Not applicable	Allowed
Drilldown	Not applicable	Allowed	Not applicable	Not applicable	Not applicable	Not applicable

Task Playbar Applet

The *task playbar applet* is a type of applet that includes buttons that allow the user to control the task UI. It is a required UI element that Siebel CRM can display at the top, bottom, or top and bottom of a task view. In Figure 9 on page 34, Siebel CRM displays the task playbar applet in the lower left corner. It includes the Pause, Previous, Next, and Cancel buttons.

If necessary, you can use a custom playbar applet to modify the look and behavior of the task playbar. You can do the following:

- Use the following applet Web templates:
  - Applet Task Playbar - Bottom
  - Applet Task Playbar - Top
- Use the CSSSWEFrameTaskPlaybar frame class. This class allows the applet to handle navigation buttons and to turn on or turn off some of the buttons, depending on the business requirement.

If you define a specialized playbar applet, then you can copy and modify the following templates:

- CCAppletPlaybarButtons.swt
- CCAppletPlaybarBottom.swt
- CCAppletPlaybarTop.swt

For more information about:

- Adding a task playbar applet to a task view, see "Creating a Task View" on page 88.
- An example that controls the playbar buttons that are active, see "Controlling the Buttons of the Playbar Applet" on page 115.
- Starting an event when the user clicks a task playbar button, see "About Event Handling" on page 228.
- Reference information about the task transaction, see "About Task Transaction" on page 225.

Display Requirements for the Task Playbar

A task view must include a task playbar applet. You can position it at the top right section or lower right section of a task view that does not require vertical scrolling. If vertical scrolling is required, then you can position it at the top right section and the bottom right section of the view. From a usability standpoint, it is recommended that you avoid vertical scrolling.

Validation with Forward Navigation in the Task Playbar

If a user clicks the forward navigation button, then Siebel CRM validates the data that the current view displays, and then does the required logic. Table 4 describes validation for forward navigation and validation results.

Table 4. Validation for Forward Navigation and Validation Results

Validation	Validation Result
A validation error occurs.	A pop-up error message displays, forward navigation stops, and the user can fix the data.
Siebel CRM successfully validates data in the current view.	Siebel CRM follows the flow in the task UI, running task steps until it encounters the next view step or the next end step.
An exception occurs that Siebel CRM does not handle.	Siebel CRM stops forward navigation, displays the error message, and then returns control to the user in the same view where the user started forward navigation.
Forward play of the task UI successfully arrives at the next task view step.	Siebel CRM applies the task UI context that is associated with this step to the business component, and then displays the associated view.

Labels for the Forward Button

You can modify the Forward Button Type property on the task step to modify the label for the forward button. The label does not modify the behavior of the task. It is a hint that provides the user with an indication of what happens when the user clicks the forward button. The label can include one of the following values:

- **Next**. Used by default if the Forward Button Type property is empty and if the task UI includes more than one view. If the task UI includes only one view, then the default value is Submit.
- **Submit**. Indicates that Siebel CRM is about to save transaction data for the task UI to the Siebel database for enterprise wide consumption and, when saved, cannot be rolled back. This situation typically occurs at the end of the task. For a task that includes a commit step, Siebel CRM must display the Submit button in the last task view that it displays before it displays the commit step.
- **Finish.** Indicates that clicking the forward navigation button ends the task UI. You can use it only in a task UI where Siebel CRM fully saves the task transaction before it displays the last summary view. This situation also applies to a task that does not use a task transaction, but instead interacts directly with the Siebel database.

Backward Navigation with the Task Playbar

Clicking the Previous button causes Siebel CRM to validate the data that it displays in the current view, and then to display the last displayed view. This feature is most useful when the user makes a mistake when using a task UI and must return to an earlier view to correct this mistake. For example, if the user enters erroneous data or chooses the wrong option. Note the following:

- Siebel CRM applies the original search and sort specifications to reconstruct the view before displaying it. It also attempts to reinstate the original current record. It does not reinstate original values.
- The view displays the most current data that the transaction contains. If the user modifies this data, then Siebel CRM displays these modifications when the user navigates back to the target view.
- The user can continue backward navigation until the user reaches the first view in the task UI. The user can freely cross subtask boundaries.
- Siebel CRM disables the backward navigation button in the first view of the task UI.
- If the user uses backward navigation, and then makes a choice at a decision point that varies from the original choice that the user made, then subsequent forward navigation can cause the task UI to enter a branch in the task UI that is different from the branch that the user first pursued.

For more information, see “Disabling Backward Navigation” on page 116.

Validation with Backward Navigation

Siebel CRM does backward validation in a way that is similar to the validation it does with forward navigation with the following differences:

- If it created records in the current view in the task transaction, and if the deferred validation option is enabled, then it does not validate the record.

- If the Defer Write Record property is set to TRUE on an insert step, then it performs deferred validation. For more information, see "About the Defer Write Record Property" on page 71.
- If the user inserts data in the task transaction, and if Siebel CRM does not validate data before it runs the commit step, then it does not save the data to the Siebel database.

Siebel CRM validates transient data because it is not part of the task transaction. For more information, see "Overview of Transient Data" on page 96.

Task Pause with the Task Playbar

Clicking the Pause button in the task playbar applet causes Siebel CRM to pause the current task UI. Siebel CRM validates the current view in the same way that it validates backward navigation. If validation succeeds, and if a handler is defined for the current task, then it runs the task event handler for the pause operation. If the event handling succeeds, then it saves the current task state and the transaction for the task. It saves this information to the Siebel database. For example, it saves information about the current view, navigation history, local data, and so on.

Siebel CRM displays the predefined view that it displayed when the user started the task UI. It sets the inbox item that references the task to a paused state. This configuration allows the user to resume the paused task from the Universal Inbox or, if the task instance references a business object, then to resume the task from a view that displays tasks that reference these business objects.

For more information, see the following topics:

- "Resuming a Paused Task UI" on page 187
- "About Event Handling" on page 228
- "Disabling the Pause Button" on page 69

Implicit Pause

Siebel CRM implicitly pauses the task in any of the following situations:

- The user attempts to navigate the Web browser outside of the current task UI. For example, if the user clicks in the site map, or clicks a screen tab.
- The session for the user times out.

Task Cancel with the Task Playbar

Clicking the cancel button in a task playbar applet causes Siebel CRM to cancel the task UI. The effect of clicking the cancel button depends on the following state of this task UI:

- If the user never paused the task UI, then cancelling it leaves no trace of the task, except for timestamp metrics for the task UI.
- If the user paused the task UI at least one time, then cancelling it resets the task state to the state that existed during the last pause. Siebel CRM also rolls back task transactions:
  - If intermediate commit steps occurred since the last pause, then it rolls back the transaction to the state that existed at the last intermediate commit.

- If intermediate commit steps did not occur since the last pause, then it rolls back the transaction to the state that it saved during the last pause of this task UI.

Radio Button Group

A *radio button group* is an optional user interface element that allows the user to make a single choice from among a group of multiple choice items. Along with the combo box and list, the user uses the radio button group to make a choice that drives the task flow.

Figure 10 includes an example radio button group that displays a predefined list of options. The user can choose one of these options.

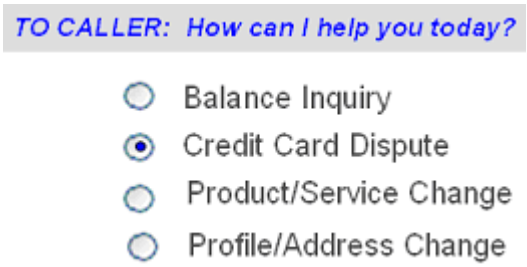


Figure 10. Example of a Radio Button Group

The input that a radio button receives determines the next task step that Siebel CRM displays, or it can determine the data it must display in the next task view. For example, a user who is a customer service representative (CSR) can choose from multiple options in reply to a question that the user poses to a caller regarding the nature of the call. The CSR asks the customer the following question:

```
How can I help you today?
```

Figure 10 illustrates how Siebel CRM displays this question as a prompt. If the customer replies that the call regards a credit card dispute, then the CSR can click the Credit Card Dispute radio button, and then the next view that Siebel CRM displays is different than the view that it displays if the issue for the caller is a balance inquiry.

The multiple radio buttons that a radio button group contains constitute a single control item. For more information, see the following topics:

- "Creating a Radio Button Group" on page 202
- "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118

Comparison of a Radio Button to a Bounded Picklist

Siebel CRM displays a radio button group differently than it displays a bounded picklist. The user can use a radio button group to view choices simultaneously without clicking the picklist. If the user makes a choice, then Siebel CRM deselects other potential choices.

Radio Buttons and Business Component Data

A task applet can include a radio button that uses data from a business component field. You can map a radio button to a field that resides in a predefined business component or to a field that resides in a transient business component. This field must be a single value field. For more information, see "Overview of Transient Data" on page 96.

Using a Radio Button with a List of Values

You can create a new set of radio buttons in the following situations:

- According to a new List of Values (LOV). In this situation, the Pick List Wizard allows you to create the LOV picklist.
- According to an existing LOV. In this situation, the LOV is already defined, so you can reuse it.

Creating a radio button group requires a LOV that Siebel CRM can use to get the choices it displays. To render a radio button group in the Siebel client, the LOV that the radio button group references must exist in the run-time environment. If you deploy the Siebel application for testing or production, then make sure you deploy the LOV to the target environment.

It is recommended that you do not use a hierarchical LOV with a radio button. If you must use a hierarchical LOV with a radio button group, then the group might function correctly at run time, but Siebel Tools cannot correctly display values during preview in Siebel Tools.

For more information, see *Configuring Siebel Business Applications*.

Applet Message

An *applet message* is an optional, free-flowing text control that displays a mix of predefined code, static text strings, and dynamic data from the Siebel database. Siebel CRM enters this data at run time. Siebel CRM displays an applet message as a line of text that can be continuous and wrapped. An applet message is similar to the personalization text that Siebel CRM displays on a home page. You can use an applet message to provide instructions to the user who performs the job task. For example, in Siebel Call Center, an applet message can represent a block of text that a customer service representative reads while addressing a customer.

You can define the static text that Siebel CRM displays in an applet message during development. Siebel CRM gets the dynamic data from business component fields. At run time, this data is part of the applet message that Siebel CRM displays while the user progresses through the task UI.

Siebel CRM statically positions an applet message as a user interface control in the task applet. Content in an applet message is read-only and does not reside inside a dialog box, so the applet message is different from other control types, such as Text or Field.

You can use an applet message in a predefined applet that references a predefined business component or in a task applet that references a transient business component. The work you do to define an applet message is the same in these situations.

You cannot modify the text format in the applet message.

For more information, see "Creating an Applet Message" on page 204 and *Siebel Object Types Reference*.

How Task UI Uses the Dashboard and Universal Inbox

A task UI uses the persistent dashboard and Universal Inbox frequently.

Persistent Dashboard

The *dashboard* is a predefined component in the Siebel client that is available in a task UI. It displays global information, such as the contact information of a caller in Siebel Call Center. This information remains on the screen when the user switches between different views. The dashboard that Siebel CRM displays in a task can include data that the task instance modifies.

Universal Inbox

The *Universal Inbox* is a feature that allows the user to display *inbox items*, which are units of work. Siebel CRM assigns these inbox items to the user. It allows Siebel CRM to assign a single owner to each inbox item. It can transfer a task UI between users for reassignment, approval, or consultation through the Universal Inbox. It stores a task in the inbox of the task owner. The inbox provides a single location where the user can find and start tasks that are paused or assigned.

A task UI depends on the Universal Inbox to allow the user to start, resume, transfer, or delete a task that Siebel CRM assigned or transferred to the user. The bottom of the task pane includes a link to the Universal Inbox to simplify navigation. If the user pauses a task, then Siebel CRM stores the state and information about the task instance in the Siebel database. It creates an inbox item in the inbox in the Siebel database. The Inbox Items List view displays the tasks that belong to a user. An inbox item displays the name of the task and the name of the user who created the task instance, by default.

How Task UI Manages a Task Instance That Displays in the Inbox

Each inbox item of type Task references only one task instance. During the duration of a task instance, the instance and the inbox item for the instance cycle through multiple states, indicated by the status field of the inbox item. The name of the inbox item is the name of the task UI, so the additional context field of the inbox item allows the user to distinguish between multiple instances of the same task. You can display some messages that are specific to this instance in the context field to help the instance owner distinguish between multiple instances, or to provide instructions that are essential to the owner.

You can use the Instance Identifier task property to configure a task UI to enter a message in the context field that is specific to a task instance. For more information, see "Modifying a Task UI to Display a Message That Is Specific to a Task Instance" on page 192.

How Users Can Resume a Paused Task UI from the Inbox

A user can click the linked name field for a paused task UI to resume it from the inbox. For more information, see "Resuming a Paused Task UI" on page 187.

A generic view of the Universal Inbox displays inbox items that Siebel CRM assigned to the current user. The user can view an inbox item of a paused task instance through an association to business data, such as an account, service request, or contact. This configuration allows the user to resume a task instance for another user without jeopardizing the integrity of the inbox. For more information, see "Creating an Association That Allows the User to Resume or Transfer a Paused Task UI" on page 187.

5 Using the Development Environment to Develop a Task UI

This chapter describes how to use the development environment to develop a task UI. It includes the following topics:

Displaying Object Types You Use to Develop a Task UI

This topic describes how to display the object types in the Object Explorer that you use to develop a task UI. Figure 11 includes the object hierarchy of the Task object type. For more information, see *Siebel Object Types Reference*.

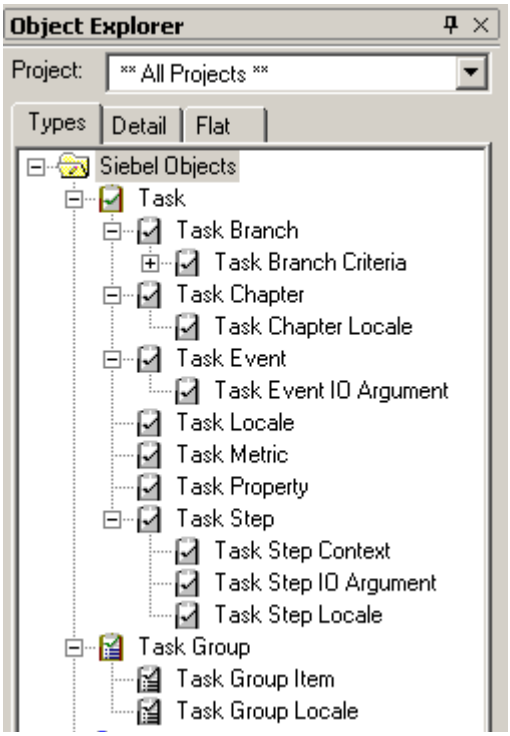


Figure 11. Hierarchy of the Task Object Type in the Object Explorer

To display object types you use to develop a task UI

1. Open Siebel Tools.
2. Choose the View menu, and then the Options menu item.
3. Click the Object Explorer tab.
4. To locate the Task tree, scroll down through the Object Explorer Hierarchy window, and then make sure the Task tree and all child objects of the Task tree include a check mark.
5. Repeat Step 4 for the following object types:
   - Task Group and all children of the task group object type.
   - View and all children of the view object type.
6. Optional. Display other object types that you might use to develop a task UI:

a Expand the Applet tree, and then make sure the Applet Message tree and all child object types of the Applet Message object type include a check mark.

b In the Applet tree, make sure Applet User Prop includes a check mark.

c In the Applet tree, expand the Control tree, and then make sure Control User Prop includes a check mark.

d Collapse the Applet tree, expand the Business Component tree, and then make sure Business Component User Prop includes a check mark.

e Collapse the Business Component tree.

f Scroll down through the Object Explorer Hierarchy window, and then make sure the Symbolic String tree and all child object types of the Symbolic String object type include a check mark.

7 Click OK.

Using the Task Editor

This topic describes how to use the Task Editor. For more information, see "About the Task Editor" on page 17.

Opening the Task Editor

You can open the task editor for a task UI.

To open the task editor

1 Locate the task UI you must modify.

For more information, see "Locating a Task UI in the Tasks List" on page 60.

2 In the Tasks list, right-click the task UI that you located in Step 1, and then choose Edit Task Flow.

Siebel Tools opens the task UI in the Task Editor. It opens the Task Palette, Properties window, and Multi Value Properties Window:

- If the palette is not visible, then choose the View menu, Windows, and then the Palette menu item.
- If the Properties window is not visible, then choose the View menu, Windows, and then the Properties Window menu item.

If the status of the task UI you chose is Completed, then Siebel Tools creates a new version of that task with a status of In Progress.

If a limited amount of space exists on your monitor, then you can stack the Object Explorer, Palette window, and Properties window on top of each other. You can then access each window through tabs near the bottom of Siebel Tools. For more information, see *Using Siebel Tools*.

Adding a Step to a Task UI

You can use the Task Editor to add a step to a task UI. You add a step, and then use the Properties window to define properties for this step. In some situations, you can use the Task Steps list to define properties for a step. You save the task UI, and then Siebel Tools displays a record for the new step in the Task Steps list, where you can modify properties.

To add a step to a task UI

1. Locate the task UI you must modify.

   For more information, see "Locating a Task UI in the Tasks List" on page 60.

2. If the value in the Status property of the task UI is Completed or Not In Use, then click Revise in the WF/Task Editor Toolbar.

   For more information, see "Using the WF/Task Editor Toolbar" on page 60.

3. Open the Task Editor.

   For more information, see "Opening the Task Editor" on page 51.

4. Drag the step type you must add from the Task Palette and drop it on the canvas.

   For more information, see "Overview of Step Types" on page 66.

5. In the Properties window, enter or modify the Name property.

   If you step out of the property, then Siebel Tools updates the step name in the canvas.

6. Optional. Enter a description of the purpose of the step.

7. Define other properties for the step, as necessary.

About the Task Property

This topic describes the task property. It includes the following topics:

Task Property and the Property Set

A *task property* is an object that stores a value that the task UI gets from the Siebel database or gets before or during processing. The following examples describe some of the ways that Siebel CRM can use a task property:

- Pass information between objects. For example, between two steps in a task UI, between a task and a subtask, or between a task and a business service. You define the task property as an input argument or output argument for the step.
- Design a decision branch that uses the value in a task property.
- Use the value in a task property in an expression.

The final value of each task property is available as a separate output argument when a task UI finishes. You can configure Siebel CRM to pass this value to other objects.

For more information about:

- Defining metrics for a task property, see "About Task Metrics" on page 234.
- The process property, which is similar to the task property, see *Siebel Business Process Framework: Workflow Guide*.
- A detailed description of properties, see *Siebel Object Types Reference*.

How a Task Property Works

A *property set* is a hierarchical structure that includes *task properties*, which are multiple sets of name and value pairs. A property set can include these task properties at each level in the hierarchy. Siebel CRM uses the property set to pass data between steps in a task UI.

Figure 12 illustrates how Siebel CRM uses a task property.

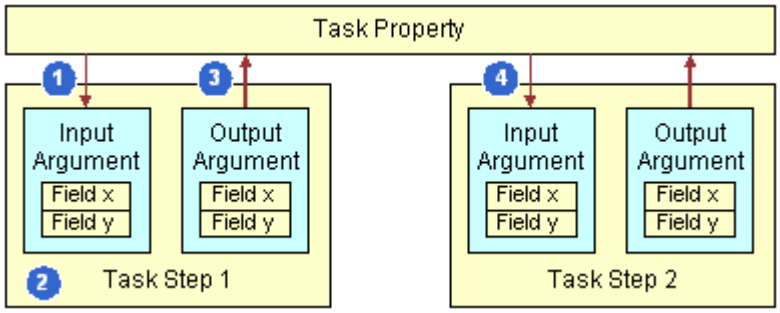


Figure 12. How Siebel CRM Uses a Task Property

Explanation of Callouts

The following steps describe an example of how Siebel CRM uses a task property:

1. Siebel CRM uses a task property to send data to Task Step 1 as an input argument.
2. Task Step 1 uses the data according to the internal configuration for the step.
3. An output argument on Task Step 1 sends data from the step to the task property.
4. An input argument on Task Step 2 brings the data that Task Step 1 uses into Task Step 2, where Siebel CRM can use it according to the internal configuration that is defined for Task Step 2.

Table 5 describes how a task property manages property sets in two different ways.

Table 5. How a Task Property Manages Property Sets

Object	Description	Work You Perform
Task Property	Store values that apply to the entire task UI.	Click the Task Editor canvas, and then use the Multi Value Property Window.
Step Argument	Communicates information between a task property and an individual step in a task UI.	Click a task step, and then use the Multi Value Property Window.

Arguments of a Task Step

You can define an argument in the Multi Value Property Window to pass information in and out of task steps. For example:

- To pass the Object ID from a parent task UI to a subtask, you define input arguments and output arguments in the Multi Value Property Window.
- If a subtask includes a return code of SUCCESS or FAILED, then you can configure Siebel CRM to send the return code to the parent task UI through an output argument.

The Task Step IO Argument is a child object of the task step object. It allows you to use an input or output argument on some types of task UI steps. Table 6 describes input arguments and output arguments that you can define for a task step.

Table 6. Input Arguments and Output Arguments of a Task Step

Input Argument	Output Argument
You can define an input argument on the following step types: ■ Business service ■ Siebel operation ■ Subtask	You can define an output argument on the following step types: ■ Business service ■ Siebel operation ■ Task view ■ Subtask ■ End

Some operations include predefined outputs. For example:

- **NumAffRows.** Used with a Query operation. Returns the number of rows returned in the query. For more information, see “Optional View Technique” on page 219.
- **NoMoreRecords.** Used with a NextRecord operation. Returns TRUE or FALSE. If no more records are available to process, then it returns TRUE.

For more information about:

- Creating arguments, see "Creating the Arguments of a Task Step" on page 82.
- The task property, see "About the Task Property" on page 52.
- Similarities between how a task UI uses a task property and how a workflow process uses a process property, see *Siebel Business Process Framework: Workflow Guide*.
- A description of the Task Step IO Argument object, see *Siebel Object Types Reference*.

How the Type Field Affects Other Fields in the Multi Value Property Window

The value you choose in the Type field in the Multi Value Property Window determines how you define other fields in the Multi Value Property Window. Table 7 describes the fields you can define.

Table 7. How the Type Field Affects Other Fields in the Multi Value Property Window

Type You Chose	Work You Perform
Business Component	Do the following: ■ Choose Business Component in the Type field. ■ Define the business component and business component fields. For more information, see "Business Component Fields That a Task UI Can Modify" on page 225.
Expression	Do the following: ■ Choose Expression in the Type field. ■ Define an expression in the Value field. Siebel CRM evaluates this expression at run time to determine the value. You can click the down arrow that Siebel Tools displays in the Value field to open the Expression Builder. It only displays this arrow after you choose Expression in the Type field.
Literal	Do the following: ■ Choose Literal in the Type field. ■ Define a string in the Value field. The value you enter defines the literal value for the argument.

Table 7. How the Type Field Affects Other Fields in the Multi Value Property Window

Type You Chose	Work You Perform
Task Property	Do the following: ■ Choose Task Property in the Type field. ■ Define the Property Name field. Task Property is available only for an input argument. For more information, see "About the Task Property" on page 52.
Output Argument	Define the Output Argument field. You can choose a task property in the Output Argument field. The list allows you to choose a task property that is of type In/Out or Out. Output Argument is available only for an output argument.

System and Custom Task Properties

A task property can be a system task property or a custom task property.

System Task Property

A *system task property* is a type of task property that Siebel Tools automatically creates when you create a new task UI, such as when you create a new record in the Tasks list. It is similar to the *system process property* in a workflow process. Every object definition for a task UI includes a set of system task properties. If you create a new task UI, then Siebel Tools adds the following system task properties:

- Context BC Id
- Context BO Name
- Context BC Name

If a task uses a standard view through a context-sensitive task group, then Siebel CRM enters data into these properties. For more information, see "Task Group" on page 36.

Table 8 describes system task properties. For more information, see "Viewing System Task Properties of a Task UI" on page 58.

Table 8. System Task Properties

Task Property	Description
Context BC Id	The ID of the current record in the business component that the Context BC Name task property identifies when Siebel CRM starts this task instance. If the Context BC Id property is: ■ **Empty.** The Siebel operation queries the new record that the task UI creates. ■ **Not empty.** The Siebel operation queries the Context BC record.
Context BC Name	The name of the business component that includes the record that the Context BC Id task property identifies.
Context BO Name	The name of the business object that the standard view references. Siebel CRM starts the task instance from this view.
Error Code	An error symbol for the task instance. Siebel CRM enters it if a step returns an error. The error code is SBL-BPR-00515.
Error Message	The text that describes the error. Siebel CRM enters data in this property if a step returns an error. For example, the following text is the error message for error code SBL-BPR-00515: `Error Executing Searchspecs at Task View Step Task View1`
Instance Identifier	The object identifier of the task instance. Siebel CRM enters data in this object when it runs the task UI.
Object Id	The Siebel Row Id of the work item that Siebel CRM is processing, which is the active row of the primary business component.
Siebel Operation Object Id	The object identifier of an object that Siebel CRM updates, creates, or queries during a Siebel operation step. It enters data in this system property when a Siebel operation runs. If a query returns multiple values, then Siebel CRM sets the Object Id property to an asterisk (*).

Custom Task Property

A *custom task property* is a property that you explicitly define to meet your design requirements. A custom task property can be of the following types:

- String
- Number
- Binary
- Date
- Hierarchy

- Integration Object
- Strongly Typed Integration Object

You can use the Multi Value Property Window to define a custom task property. For more information, see "Creating a Task Property" on page 83.

How a Subtask Uses a Task Property

Similar to the subprocess step of a workflow process, you can configure Siebel CRM to pass information in and out of a subtask through a task property in an input argument or output argument according to the following logic:

- An input argument allows Siebel CRM to enter data in a task property in the subtask with information from the parent task UI. If Siebel CRM starts a subtask, then it initializes the task property of the subtask with the value of the input argument of the subtask step.
- An output argument allows Siebel CRM to enter information from a subtask in the parent task UI. If the subtask returns data to the parent, then the parent can read the task properties of a subtask through the output argument of the subtask step.

You can use an input argument to pass a system task property, such as the object ID, from a parent task UI to a subtask. If the subtask references a different business component, then you must configure Siebel CRM to pass the row ID of the target object as the subtask Object ID task property. For a subtask step, the receiving end of an input argument is one of the task properties of the subtask. To choose this value, you can use the list in the Task Input field. Siebel Tools displays the the task properties of the subtask that are of type In/Out or In in this list.

If the subtask creates a child object, then the Object ID that Siebel CRM passes to a subtask must be null, and it must not be the Object ID of the parent.

For more information, see "Creating a Subtask Step" on page 73.

Why Siebel CRM Passes Hierarchical Data by Reference

A parent task UI and a subtask include separate local task properties, so passing arguments causes Siebel CRM to copy the data between the parent task and the subtask. Copying hierarchical data for a hierarchical task property can consume resources. For this reason, Siebel CRM passes hierarchical data by reference between the parent task and a subtask.

Viewing System Task Properties of a Task UI

You can use the Task Properties list or the Multi Value Property Window to view system task properties of a task UI.

To view system task properties of a task UI

1. Locate the task UI you must modify.

   For more information, see "Locating a Task UI in the Tasks List" on page 60.

2 Optional. To view task properties in the list:

   a In the Object Explorer, expand the Task tree, and then click Task Property.

      The Task Properties list displays object definitions for the task properties of a task.

3 Optional. To view task properties in the Multi Value Property Window, open the Task Editor.

   The Task Editor displays the Multi Value Property Window and the system task properties. For more information, "Opening the Task Editor" on page 51.

Using the Tasks List

You can use the Tasks list in Siebel Tools to create a task UI. You can right-click the task in the Tasks list to display a menu that allows you to work on this task. Table 9 describes some of the items that this menu contains.

Table 9. Some Menu Items That the Right-Click Menu Displays in the Tasks List

Menu Item	Description
New Record	Creates a new object definition for a task UI. For more information, see "Using the Tasks List to Create a Custom UI" on page 87.
Delete Record	Deletes a task UI. If the status for a task UI is Completed, then you cannot delete it. You can do the following work to delete a completed task: ■ Click Expire in the WF/Task Editor toolbar. This command modifies the Status property for the task to Not In Use. ■ Delete the task.
Copy Record	Copies a task UI. Does a cascade copy of an existing task UI. A cascade copy copies the object definition for the task UI and the object definitions of the child objects of this task UI. You must rename the new copy immediately after the cascade copy finishes to make sure the new copy is unique.
Add to Archive	Archives a task UI in a SIF file. For more information, see the topics about working with archive files in *Using Siebel Tools*.
Validate	Validates a task UI. For more information, see "Validating a Task UI" on page 151.
Edit Task Flow	Opens the Task Editor for a task UI. For more information, see "Using the Task Editor" on page 51.
Reset Version	Resets the version number of a task UI to 0. You can reset the version only if the original version 0 of the task UI no longer exists in the SRF (Siebel Repository File).

Locating a Task UI in the Tasks List

This topic describes how to locate the object definition for a task UI in the Tasks list.

To locate a task UI in the Tasks list

1. Open Siebel Tools.
2. If necessary, display the Task object hierarchy.

   For more information, see "Preparing Siebel Tools to Create a Custom Task UI" on page 86.
3. In the Object Explorer, click Task.
4. In the Tasks list, query the Task Name property for the task UI you must modify.

   If the task UI exists, then Siebel Tools displays the object definition for it. Querying for the task UI in this way results in Siebel Tools displaying a single, isolated record. This technique helps to make sure you choose the correct task UI in the Tasks list when you modify child objects of the task UI or when you do other work, such as publishing or revising.

Using the WF/Task Editor Toolbar

The *WF/Task Editor toolbar* is a toolbar in Siebel Tools that you use to deploy a task UI. It allows you to do the following work on a task UI:

- Publish and activate
- Publish
- Revise
- Expire

You must use the WF/Task Editor toolbar to publish and activate a task UI, or to expire a task UI. For more information about the work that these buttons perform, see "Process of Deploying a Task UI" on page 164.

Figure 13 includes the WF/Task Editor toolbar.

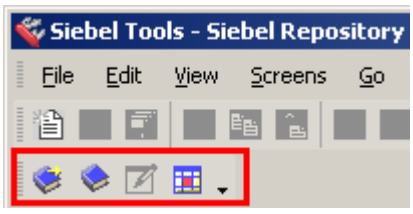


Figure 13. WF/Task Editor Toolbar in Siebel Tools

Table 10 describes the buttons that the WF/Task Editor Toolbar contains.

Table 10. Buttons That the WF/Task Editor Toolbar Contains

Button Name	Description
Publish	Moves the object definition of a task UI from the repository tables that reside in your development environment to the run-time tables that reside on the Siebel client. You can then use the administrative screens in the Siebel client to activate this task UI. If you click Publish, then Siebel Tools modifies the status of the task UI from In Progress to Completed. You can choose more than one task before you click the Publish button or the Publish/Activate button to publish and activate multiple tasks.
Publish/Activate	Publishes and activates a task UI with a single button click in Siebel Tools while in local mode. You must do the following work to use the Publish/Activate button: ■ Set the VerCheckTime parameter. For more information, see "Preparing Siebel Tools to Create a Custom Task UI" on page 86. ■ Use this button to activate a task UI only for debugging on a Siebel Web Client. ■ To view the task, you must restart any Siebel client that is currently running. If you run or test a task UI in a server environment, then do not use the Publish/Activate button. Instead, use the Publish button to publish the task, and then use the Activate button in the task deployment view in the Siebel client. For more information, see "Publishing and Activating a Task UI in Siebel Tools" on page 167.
Revise	Creates a new version of the task UI for editing. Siebel Tools increments the version property by 1, and then displays this new version in the Tasks list.
Expire	Sets the status of the task UI to expired.

Displaying the WF/Task Editor Toolbar

You must display the WF/Task Editor toolbar the first time you use it.

To display the WF/Task Editor toolbar

- In Siebel Tools, choose the View menu, Toolbars, and then the WF/Task Editor Toolbar menu item.

  To identify a button in the toolbar after you display the toolbar, position your mouse over an icon in the toolbar, and then note the name of the button that Siebel Tools displays in a small pop-up message.

Revising a Task UI

You can use the WF/Task Editor toolbar to revise a task UI.

To revise a task UI

1. In Siebel Tools, in the Object Explorer, click Project.
2. In the Projects list, query the Name property of the project that includes the task UI you must modify.
3. To lock the project, click the Locked property.

   The Locked property must include a check mark.
4. Locate the task UI you must modify.

   For more information, see "Locating a Task UI in the Tasks List" on page 60.
5. To create a new version of the task UI that you can edit, click the Revise button in the WF/Task Editor toolbar.

   Starting with Siebel CRM version 8.1.1.9, you can compare different versions of a task UI. For more information, see the topic about Comparing Different Versions of a Workflow Process or Task UI in *Using Siebel Tools*.
6. Make your modifications in the new version of the task UI.
7. Deploy the task UI.

   For more information, see "Process of Deploying a Task UI" on page 164.
8. Unlock the project you locked in Step 3.

Expiring a Task UI

You can use the WF/Task Editor toolbar to expire a task UI.

To expire a task UI

1. Locate the task UI you must modify.

   For more information, see "Locating a Task UI in the Tasks List" on page 60.

2 Click the Expire button in the WF/Task Editor toolbar.

For more information, see "Revising a Task UI" on page 62.

Publishing in Local Mode

If you work in local mode and you must use the Publish/Activate button rather than the Publish button, then it is not necessary to activate the task UI separately in the run-time client.

Wizards You Use to Create a Task UI

You can use the following wizards to help create a task UI:

- **Task Wizard.** Helps you create an object definition for a task UI, including the first steps that you must include to run a task.
- **Task Applet Wizard.** Helps you create a task applet.
- **Task View Wizard.** Helps you create task UI views, and makes sure you include and display the objects that each task UI requires.
- **Transient Business Component Wizard.** Helps you create a transient business component and fields for the transient business component, making sure you use the correct classes, table, and class set. For more information, see "Overview of Transient Data" on page 96.

6 Creating Steps and Connectors

This chapter describes how to create task UI steps and connectors. It includes the following topics:

Overview of Step Types

Figure 14 includes the different types of steps and connectors that Siebel Tools displays in the Task Palette of the Task Editor. For more information, see "Using the Task Editor" on page 51.

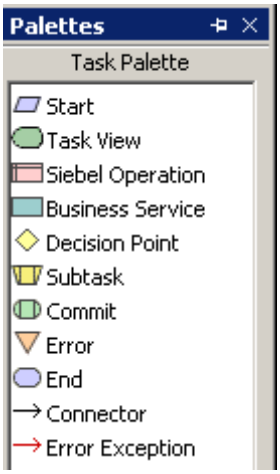


Figure 14. Step Types and Connectors in the Task Palette

Table 11 describes the step types and connectors that are available in the Task Palette.

Table 11. Step Types and Connectors in the Task Palette

Step Type	Description
Start	Defines the input conditions that must be met to run a task instance.
Task View	Displays a task view in the Siebel client.
Siebel Operation	Includes an Insert, Update, or Query operation that interacts with a business component record.
Business Service	Calls a business service that allows you to run a predefined or custom action.
Decision Point	Evaluates a condition to determine the next step to run.
Subtask	Calls a task UI. The subtask task can be independent of the calling task.
Commit	Explicitly saves task data that Siebel CRM stores in temporary storage to the Siebel database.
Error	Creates an error and returns control to the current view.
End	Indicates the end of the task UI.
Connector	Determines the direction of flow between steps in a task UI.
Error Exception	Handles deviation from normal processing, such as a system error or a custom error that you define.

Creating a Start Step

A *start step* is a type of task UI step that indicates the starting point of a task. A task must include only one start step. You can create conditional logic and run-time events on the connector that emanates from the start step when you create a workflow process. You are not required to create a condition branch that emanates from the start step for a task UI.

To create a start step

- Add a start step to a task UI.

  For more information, see "Adding a Step to a Task UI" on page 52.

Creating a Task View Step

A *task view step* is a type of task UI step that displays a task view in the Siebel client. The task view step allows you to map a business process to the user interface. Several properties on the task view step, such as Disable Cancel, allow you to specify behavior of the task playbar and control how Siebel CRM saves information to the Siebel database. For more information, see "About the Task View" on page 39.

To create a task view step

1 Add a task view step to a task UI.

For more information, see "Adding a Step to a Task UI" on page 52.

2 Choose the task view step you added in Step 1.

3 In the Properties Window, set the Task View property.

Siebel Tools binds the task view step to a task view. You can double-click the task view step to view and edit the task view. For more information, see "Drilling Down on a Task View Step" on page 69.

You must create the applets and view before you can do Step 3. If you do not do this, then Siebel Tools does not display any values in the Task View property.

4 Set the following properties:

- Disable Cancel
- Disable Pause
- Disable Previous

You can set the disable property for the button to TRUE to disable a navigation button. For example, to disable the Cancel button, set the Disable Cancel property to TRUE.

5 Set the following properties:

a Display Name

b Display Name Type

Display Name Type controls how Siebel CRM displays the task step of the current task pane. Siebel CRM uses it and the Display Name property. For more information, see "Controlling the Display Name of a Step" on page 69.

6 Set the Forward Button Type property.

For more information, see "Validation with Forward Navigation in the Task Playbar" on page 42.

7 Set the following properties:

- Retain Applet SearchSpec
- Retain Task SearchSpec
- Retain User SearchSpec

Search specifications affect how Siebel CRM preserves the business component state across task views. For more information, see "Record Context Is Lost" on page 160 and *Siebel Object Types Reference*.

8 Optional. Set the Task Step Context property.

The Task Step Context property defines the search specification that Siebel CRM uses to filter data for the view step. For more information, see "Defining the Context for a Task Step" on page 193.

9 Optional. If you create the task step context, then you can add output arguments, as required.

The task view step does not return an output argument. You can use the Output Arguments tab in the Multi Value Property Window to update a task property. For example, you can copy data that the user enters in a business component field to a task property. For more information, see "How the Type Field Affects Other Fields in the Multi Value Property Window" on page 55.

10 Choose the File menu, and then choose the Save menu item.

Drilling Down on a Task View Step

You configure Siebel CRM to bind the task view step to a task view. You can then double-click the task view step in the Task Editor, and then Siebel Tools displays the Task View UI editor that displays the view that it binds to the task view step. The Task View UI editor displays an approximation of how Siebel CRM renders the view at run time.

Disabling the Pause Button

You can set the Disable Pause property on the task view step to TRUE to disable the Pause button on the playbar. Disabling the pause button does not prevent implicit pause, but you can use it to provide a hint to the user that pausing is not recommended. For example, in a summary view. For more information, see "Task Pause with the Task Playbar" on page 44.

Controlling the Display Name of a Step

You can use the Display Name and Display Name Type properties of the task view object to suppress steps that Siebel CRM displays in the current task pane. If you set Display Name Type to Unique, and if consecutive task views use the same display name, then Siebel CRM displays this name only one time in the current task pane.

To control the display name of a step

- Do one of the following:
  - Enter a value in the Display Name property for the first step. Leave Display Name empty for subsequent steps.

    If the Display Name property is empty in subsequent steps, then Siebel CRM displays the step for the first view as the current step even if the user navigates to a subsequent step. This option is useful if the user must perceive a series of views as one logical step.
  - Set the Display Name Type property to Unique.

    For more information, see "Using the Display Name with a Looping Task UI" on page 70.

Using the Display Name with a Looping Task UI

Setting the Display Name Type property to Unique causes Siebel CRM to add the value that the Display Name property contains to the current task pane the first time the user encounters the step. Siebel CRM does not add it on a subsequent step. This configuration is useful in a loop. For example, a loop where the user enters multiple line items, where each line item represents one iteration of the loop. If the type is set to Unique, then Siebel CRM displays the step in the current task pane the first time through the loop, but not during subsequent iterations.

If a loop includes:

- **One view.** Setting the Display Name Type property to unique is sufficient.
- **More than one view.** You can set the Display Name Type property to Unique and leave the Display Name property empty. Siebel CRM completes the Display Name and marks it as Unique for the first view in the loop. Other views in the loop include an empty Display Name. Siebel CRM then displays the Display Name of the first view in the current task pane for the entire loop.

For more information, see *Siebel Object Types Reference*.

Creating a Siebel Operation Step

A *Siebel operation step* is a type of task UI step that performs an operation on business component data. Example operations include Insert, Update, and Query. Siebel CRM runs a Siebel operation step, and then the Siebel Operation Object ID process property stores the row ID of the record where Siebel CRM performed the operation. If the query operation returns multiple records, then the property stores an asterisk (*).

You can create a Siebel operation step for a business component that references the same business object that the task UI references. To update a business component that does not reference this business object, you must use Siebel Tools to configure the business component to reference the business object.

Some business component fields are not available for modification. For more information, see "Business Component Fields That a Task UI Can Modify" on page 225.

To create a Siebel operation step

1. Add a Siebel operation step to a task UI.

   For more information, see "Adding a Step to a Task UI" on page 52.

2. In the Properties window, set the Operation property.

   If you must configure Siebel CRM to update or insert a field that includes dependencies, then make sure the field is valid. For example, if your task UI updates the area and subarea fields of a service request, then you must make sure the value that the user chooses for the subarea field is valid for the area field. For more information, see *Siebel Business Process Framework: Workflow Guide*.

3. In the Business Component property, choose the business component where this Siebel operation step uses data.

4 If you must configure Siebel CRM to create a field, or if the Siebel operation step performs an insert operation, then make sure you add the field to the Siebel operation step.

   For more information, see "Identifying the Business Component Field for a Siebel Operation Step" on page 71.

5 Optional. Create a search specification for the Siebel operation step.

   For more information, see "Defining the Context for a Task Step" on page 193.

6 Optional. In the Multi Value Property Window, add input arguments and output arguments.

   For more information, see "Creating an Output Argument on a Task Step" on page 82.

About the Defer Write Record Property

Setting the Defer Write Record property to TRUE on a Siebel operation step allows the user to provide data for a required field in the subsequent task view step before the Siebel operation step attempts an insert. If the user attempts to provide data for a Siebel operation that already occurred, then Siebel CRM displays an error that a required field includes no data. This situation occurs when the following logic exists in your task UI:

1 A Siebel operation step performs an insert operation.

2 Task view step follows the Siebel operation in Step 1. The user uses this task view step to provide the data that the insert requires.

To fix this problem, the Defer Write Record property defers the insert operation until a commit operation occurs that allows the user to provide the required data before Siebel CRM does the insert operation. The Defer Write Record property is not specific to a task UI. For more information, see "Using the Defer Write Record Property When You Disable Task Transaction" on page 159 and *Siebel Object Types Reference*.

Identifying the Business Component Field for a Siebel Operation Step

You can use the Fields tab in the Multi Value Property Window to create a field for a Siebel operation step.

To identify the business component field for a Siebel operation step

1 Open the Task Editor.

   For more information, "Opening the Task Editor" on page 51.

2 Make sure the Business Component property of the Siebel operation step you must modify contains a value.

3 Choose the Siebel operation step, and then open the Multi Value Property Window.

4 In the Multi Value Property Window, click the Fields tab.

5 Right-click in the list area of the Multi Value Property Window, and then choose New Record.

6 In the Field Name field, choose the business component field that this step must update.

The dropdown list for the Field Name field lists the business component fields for the business component that you specify in the Business Component property of the Siebel operation step.

7 Set the Type field.

For more information, see "How the Type Field Affects Other Fields in the Multi Value Property Window" on page 55.

8 Set the Value field.

9 If you create multiple fields, and if the user must enter field values in a specific order, then define this order in the Preferred Sequence field, beginning with number 1.

10 Optional. Enter comments.

For information about updating a field that uses a multivalue group, see *Siebel Business Process Framework: Workflow Guide*.

Creating a Business Service Step

A *business service step* is a type of task UI step that calls a business service. The following items describe actions that some predefined business services perform:

- **Notification.** The Outbound Communication Server business service sends a notification to an employee or a contact.
- **Assignment.** Assignment Manager calls the Synchronous Assignment Manager Request business service to assign an object in a task UI.
- **Server task.** You can use one of the following business services to run a server task for a server component:
  - Synchronous Server Requests business service
  - Asynchronous Server Requests business service

For more information about business services, see the following items:

- "Guidelines for Using a Business Service" on page 217
- *Siebel Business Process Framework: Workflow Guide*

To create a business service step

1 Add a business service step to a task UI.

For more information, see "Adding a Step to a Task UI" on page 52.

2 In the Business Service Name property, choose the name of the business service that this step calls.

3 In the Business Service Method property, choose the method that Siebel CRM uses to call the business service.

The choices that Siebel Tools displays for this property depend on the business service that you define in Step 2.

4 Create input arguments and output arguments.

For more information, see "How the Type Field Affects Other Fields in the Multi Value Property Window" on page 55.

Creating a Decision Point

A *decision point* is a type of task UI step that evaluates one or more conditions to determine the next step to run in a task. You can create a condition on a connector that emanates from the decision point. You cannot create a condition directly on the decision point.

To create a decision point

1 Add a decision point to a task UI.

For more information, see "Adding a Step to a Task UI" on page 52.

2 Create a branch connector for the decision point.

For more information, see "Creating a Branch Connector" on page 78.

Creating a Subtask Step

A *subtask step* is a type of task UI step that allows you to start a separate task in a task. You can use it to do the following:

- Reuse common sequences that you define in a task UI to decrease development and maintenance cost.
- Modularize a task UI. If a task is so large that it becomes difficult to develop and manage, then you can separate it into smaller subtasks.
- Separate a large task into multiple tasks to improve readability of the task UI in the Task Editor.
- Maintain a clean, consistent, and intuitive programming model.

For more information, see "Using a Transient Business Component in a Subtask" on page 104.

To create a subtask step

1 In the Object Explorer, click Task.

2 Make sure the subtask that the subtask step calls is defined correctly.

For more information, see "Characteristics of a Subtask" on page 74.

3 Add a subtask step to a task UI.

For more information, see "Adding a Step to a Task UI" on page 52.

4 In the Subtask Name property, choose the subtask that this subtask step calls.

5 Optional. Create input arguments or output arguments for the subtask.

For more information, see "How the Type Field Affects Other Fields in the Multi Value Property Window" on page 55.

Comparing a Subtask to a Subprocess

Table 12 compares a subtask that resides in a task UI to a subprocess that resides in a workflow process.

Table 12. Comparison of Subtask to Subprocess

Object	Siebel Task UI	Workflow Process
Process Instance	A parent task UI and a subtask share the same process instance.	A parent workflow process and a subprocess run in separate instances.
Process Properties and Task Properties	Starting a subtask does not create a new instance. Siebel CRM creates a new context each time it starts a subtask. The context stores local task properties. A subtask and a parent task UI do not share task properties. For more information, see "How a Subtask Uses a Task Property" on page 58.	Starting a subprocess creates a new instance of the workflow process. A subprocess includes an independent set of process properties.
Business Object	A parent task UI and a subtask must reference the same business object. They must include the same business object type and share the same business object instance.	A parent workflow process and a subprocess can reference different business objects, and can use two different business object instances.
Input Arguments and Output Arguments	Siebel CRM can pass information in and out of a subtask through an input argument or an output argument. An input argument allows it to enter data in the task property of a subtask. It can get this data from the parent task. For more information, see "How a Subtask Uses a Task Property" on page 58.	The operation of an input argument or output argument is similar to the subprocess step in a workflow process.

Characteristics of a Subtask

A subtask includes the following characteristics:

- A task UI can include one or more subtask steps.
- A subtask can include a subtask.
- You cannot use a subtask as a parent task.
- A subtask can include a commit step.
- An object definition must exist for the subtask before you can reference it in a subtask step of the parent task.
- The Is Subtask property of the task UI that a subtask step references must be TRUE. You cannot modify this value after you set it.
- The boundaries of a subtask are not visible in the Siebel client. For example, if the user clicks Previous or Next, then the user can cross the subtask boundary in either direction.
- You can define an event handler only for a parent task, and not a subtask. For more information, see "How Siebel CRM Handles an Event That Occurs in a Subtask" on page 229.
- You must explicitly define a task UI as the parent task or the subtask at design time. Each parent task UI includes a task state. The task state stores information that is essential to the run-time task instance. This information includes the pointer to the current step, task object Id, and the navigation path. A subtask does not require an individual task state, but it does require an independent set of local task properties.
- The parent task and the subtask pass data through input arguments and output arguments so that they can communicate with each other.
- You can deploy and activate a subtask in the same way that you deploy and activate a parent task. If you publish a task that includes a subtask, then you must publish the subtask before you publish the parent task. This sequence makes sure the subtask is available to the parent task in the run-time environment.

Using the Exception Branch in a Subtask

You can use an exception branch in a subtask. You cannot use an exception branch to enter or leave a subtask step. This restriction makes sure that a subtask can only exit from the end step of the subtask, and in the forward direction.

Creating a Commit Step

A *commit step* is a type of task UI step that explicitly saves task data from temporary storage to the Siebel database. The end step also saves temporary data to the Siebel database when a task finishes. The commit step allows you to configure Siebel CRM to save data before the task reaches the end step. You can create an exception branch that emanates from the commit step to handle an error that results from persistent temporary data.

You must use a commit step in the following situations:

- To save data at an intermediate save point
- To make sure the user cannot modify some data

For more information, see "Commit Interim Data Technique" on page 223, and "Using a Transient Business Component with a Commit Step" on page 104.

To create a commit step

- Add a commit step to a task UI.

  For more information, see "Adding a Step to a Task UI" on page 52.

Creating an Error Step

An *error step* is a type of task UI step that creates an error message and returns control to the current view. It allows you to configure Siebel CRM to display an error message in the Siebel client. This functionality is typically required if an error message that a business service returns is not sufficiently clear to the user. The error step in a task UI supports an exception in a way that is similar to how the Stop step works in a workflow process. For example, assume a business service step in a task UI calls an external system, the external system is down, and an error occurs. An exception branch and an error step handle this error.

If the error step runs, then Siebel CRM does the following work:

- Replaces the existing error message with the error message that you define for the error step. This configuration allows you to display an informative and contextual message in the Siebel client. Siebel CRM displays this error message in a dialog box on top of the current view.
- Resets the task UI to the most recent view step that it displayed before it encountered the error step. If Siebel CRM runs an error step before it displays the first view step in a task UI, then it cancels the task.

You can use an error step in an exception branch or as part of the normal logic of a task UI to handle an expected error.

To create an error step

1. Add an error step to a task UI.

   For more information, see "Adding a Step to a Task UI" on page 52.

2. In the Error Code property of the error step, choose a predefined error code from the picklist, or define a custom error message.

   For more information, see "Creating a Custom Error Message" on page 77.

How Siebel CRM Handles an Error While a Task UI Runs

If an exception occurs in a task step while the task UI runs, and if no exception branch is defined, then Siebel CRM displays a generic error message in a dialog box that it displays on top of the current task view. The user can acknowledge the error message, and then correct the data in the task view, or can navigate to a previous task view to correct the data. This configuration is known as *default exception handling*.

How Siebel CRM Handles an Error When the User Pauses a Task UI

No exception handling occurs if a user pauses a task UI. If a paused task fails, then Siebel CRM displays a typical error message, not a pause error. The user can cancel or continue the task.

How Siebel CRM Handles an Error When the User Cancels a Task UI

If the user clicks Cancel, then Siebel CRM displays a dialog box that prompts the user for confirmation. For example, if the user is about to perform a delete or undo operation in a task UI, and then clicks OK to continue.

Creating a Custom Error Message

You can use a custom error code and an input argument to create a custom error message for an error step. You can use the input argument to define the text of the custom error message.

To create a custom error message

1. Add an error step to a task UI.

   For more information, see "Adding a Step to a Task UI" on page 52.

2. In the Error Code property of the error step, choose an error code that starts with WF_ERR_CUSTOM.

   For example, WF_ERR_CUSTOM_1.

3. Make sure the error step is chosen.

4. Use the Multi Value Property Window to add an input argument:

   a. In the Input Argument field, enter a substitution variable.

      For example, %1. You can use a substitution variable in the error message to represent the input argument for an error step. A percent symbol (%) identifies a substitution variable.

   b. Choose a Type for the input argument, which is the source of the value that Siebel CRM uses for the error message text.

   c. Define the remaining fields, according to the Type you set in Step b.

      For more information, see "Creating an Input Argument on a Task Step" on page 82.

   d. Step off the record to save the modifications.

5. Optional. Repeat Step 4 to create more input arguments.

Creating an End Step

The *end step* is a type of task UI step that instructs Siebel CRM to end the task instance, and then to transfer data that currently resides in temporary storage to the Siebel database. It also provides one last chance to modify a task property that the task output arguments return from the object that called the task. Each task must include only one end step.

To create an end step

1 Add an end step to a task UI.

For more information, see "Adding a Step to a Task UI" on page 52.

2 Optional. Define an output argument for the end step.

An output argument allows you to configure Siebel CRM to store a value that it creates while a task UI runs. It stores this value in a task property. It can then pass this value between objects in a task UI, such as between a subtask and a parent task. For more information, see "About the Task Property" on page 52.

Creating a Connector

The *connector* is an object in a task UI that allows you to establish flow between task steps.

To create a connector

1 Add a connector to a task UI.

Do the work described in "Adding a Step to a Task UI" on page 52, with the following differences:

- Instead of dragging and then dropping a step type, drag and then drop a connector.
- When you drag the connector from the Task Palette to the canvas, drop the start point of the connector onto the preceding step.
- Before you connect two steps, place the steps to be connected on a horizontal plane, with the preceding step placed on the left and the subsequent step placed on the right. Place the steps in close proximity to one another. Siebel Tools automatically connects the start point to the preceding step and the end point to the subsequent step when you drop the connector.

2 Drag the connector end point onto the subsequent step.

A connector end point includes an arrow.

A connector end point that is colored white is not connected correctly. A red end point indicates that the connector is connected correctly. You must make sure the ends of every connector in the task UI are red.

3 Optional. Enter text in the Label property.

Siebel Tools displays this text as a label on the connector in the Task Editor.

Creating a Branch Connector

A *branch connector* is a type of connector in a task UI where you create a condition. Similar to how it operates in a workflow process, a branch connector in a task UI can be conditional or nonconditional. Siebel CRM uses a condition and a decision point to determine the path to follow in the task UI according to criteria that you define. A different action can occur depending on the path that it follows.

You can configure branching in a task UI on branch connectors that emanate from a decision point. You can create a separate condition for each of these connectors. You do not create conditions on the decision point. To determine the next step to run, the decision point evaluates the conditions you define on the outgoing connectors.

You can define the following values in the Type property of a connector:

- **Condition**. Includes a condition. Allows you to create a condition that must be met to continue through this path.
- **Default**. Does not include a condition. If the condition is not met for another connector, then the task UI flows down the Default connector.

To view an example that includes a condition, see "Revising the Task UI" on page 124.

To create a branch connector

1. Create a connector.

   For more information, see "Creating a Connector" on page 78.

2. Enter or modify the Name property of the branch connector.

   The branch name must be unique.

3. Set the Type property of the branch connector to Condition.

   For more information, see the description for the task branch object in *Siebel Object Types Reference*.

4. Create conditions that apply to this branch.

   For more information, see "Creating a Condition on a Branch Connector" on page 79.

5. In the Comments property, enter text that describes the branch logic.

6. Optional. Repeat Step 1 through Step 5 for each additional branch connector you must create.

   If your task UI must proceed along more than one conditional connector and one default connector, then you must create a separate branch connector for each logical path.

7. Create a default branch.

   CAUTION: You must create a Default connector to handle the situation where an item does not meet any of the conditions you create.

Creating a Condition on a Branch Connector

You can create a condition on a branch connector to control task flow. For example, the following are some conditions that you can create according to the value of a priority field:

- If the priority is high, then the task UI follows a branch that sends an email to a vice president.
- If the priority is medium, then the task UI follows a branch that sends an email to an individual contributor.

You can use the Compose Condition Criteria dialog box to create a condition. For detailed usage information, see *Siebel Business Process Framework: Workflow Guide*.

To create a condition on a branch connector

1 Open the Task Editor.

For more information, "Opening the Task Editor" on page 51.

2 In the Task Editor, right-click the branch connector where you must create a condition.

3 Choose Edit Conditions.

Siebel Tools displays the Compose Condition Criteria dialog box. Siebel Tools constrains the Values that it lists in this dialog box according to the business object that the Business Object property of the task UI references.

4 In the Compose Condition Criteria dialog box, use the Compare To list to choose one of the values that the following table describes.

Value	Description
Applet	Uses the value that an applet field contains for the condition comparison.
Business Component	Uses the value that a business component field contains for the condition comparison or when you create an expression.
Expression	Uses an expression to evaluate a specific value.
Task Property	Compares the value that a task property of a task instance contains to a defined value.

5 Choose an Operation to evaluate the values.

6 Enter an Object and Field, if necessary.

7 Enter one or more values in the Values window.

You can enter multiple values in the Values window. If you enter multiple values, then Siebel CRM uses an OR operator between each value.

8 If you chose Expression in the Compare To field, then enter the expression in the Values window.

For more information, see *Siebel Developer's Reference*.

9 Click OK.

10 Save your modifications.

11 Close the Task Editor.

Creating Multiple OR Conditions

You can create multiple conditions for each branch. Siebel CRM treats multiple conditions with the AND operator. You can use multiple expressions to create multiple OR conditions. The following example illustrates an expression that uses the OR operator to compare a business component field to the date for today:

```
([Close Date] <= Today()) OR ([Name] = 'Opportunity test1')
```

For more information, see *Siebel Business Process Framework: Workflow Guide*.

Branching and Parallel Processing

You cannot define the order that Siebel CRM uses to evaluate conditions, so it is important that you define branching conditions so that they are mutually exclusive. Siebel Task UI does not support parallel processing. Make sure you create conditions so the task UI can proceed along only one connector. If you create conditions so that flow can proceed simultaneously along multiple connectors, then Siebel CRM cannot predict run-time behavior.

Creating an Error Exception Connector

The *error exception* is a type of connector that resides in a task UI. It handles the following types of errors:

- **System.** For example, a failure that occurs when Siebel CRM sends an email notification.
- **User.** For example, a user attempts to submit an incomplete order.

Siebel Tools displays an error exception as a red connector in the Task Editor.

To create an error exception

1. Create a connector.

   For more information, see "Creating a Connector" on page 78.

2. In the Type property of the connector, choose one of the following values:

   - Error Exception
   - User Defined Exception

   The task controller treats a user defined exception the same way it treats an error exception.

3. Double-click the exception connector.

4. In the Compose Condition Criteria dialog box, create conditions that apply to the exception.

   For information about how to use the Compose Condition Criteria dialog box, see *Siebel Business Process Framework: Workflow Guide*.

Creating the Arguments of a Task Step

This topic describes how to create the arguments of a task step. For more information, see "Arguments of a Task Step" on page 54.

Creating an Input Argument on a Task Step

An input argument allows you to configure Siebel CRM to pass data to some types of steps in a task UI at run time.

To create an input argument on a task step

1. Open the Task Editor for the task UI you must modify.

   For more information, "Opening the Task Editor" on page 51.

2. In the Task Editor, choose the step where you must create an input argument.
3. In the Multi Value Property Window, click the Input Arguments tab.

   For a Siebel operation step, you use the Fields tab rather than the Input Arguments tab.

4. Right-click in the list area of the Multi Value Property Window, and then choose New Record.
5. In the Input Argument field, choose an input argument.

   This value is the destination of the input argument. For a Siebel operation step, choose the Field Name.

6. In the Type field, choose a type.
7. Define remaining fields for the input argument, according to the type you defined in Step 6.

   For more information, see "How the Type Field Affects Other Fields in the Multi Value Property Window" on page 55.

   Some fields are not available. For more information, see "Business Component Fields That a Task UI Can Modify" on page 225.

Creating an Output Argument on a Task Step

An output argument allows you to configure Siebel CRM to store the value that results from processing one of the following step types:

- Business service step
- Siebel operation step
- Task view step
- End step

Siebel CRM stores this value in a task property. It can then pass this value to another object, such as the input argument of a subsequent task step.

To create an output argument on a task step

1 Open the Task Editor for the task UI you must modify.

For more information, "Opening the Task Editor" on page 51.

2 In the Task Editor, choose the step where you must create an output argument.

3 In the Multi Value Property Window, click the Output Arguments tab.

4 Right-click in the list area of the Multi Value Property Window, and then choose New Record.

5 In the Property Name field, choose a property name.

This is the name of the task property that stores the value of the output argument.

6 In the Type field, choose a type.

7 Define remaining fields for the output argument, according to the type you defined in Step 6.

For more information, see "How the Type Field Affects Other Fields in the Multi Value Property Window" on page 55.

Creating a Task Property

A task property can pass a value between objects in a task flow, such as between a task step and another task step, or between a task step and a subtask. Siebel Tools automatically defines a set of system task properties when you create a task object. You can use these system task properties to pass information between task objects. You can also define a custom task property to meet your design requirements. For more information, see "About the Task Property" on page 52.

To create a task property

1 Open the Task Editor.

For more information, "Opening the Task Editor" on page 51.

2 Click the canvas. Do not click a task step or connector.

The Multi Value Property Window displays tabs that are specific to the task UI.

3 In the Multi Value Property Window, click the Task Properties tab.

4 Right-click in the list area of the Multi Value Property Window, and then choose New Record.

5 Enter a value in the Name field.

6 Choose the Data Type for the task property.

7 Optional. In the Default field, assign a default value for the task property.

8 Optional. Modify the value in the following fields:

- Access Mode

- In/Out

Siebel Tools sets the value in the Access Mode field to R/W and the value in the In/Out field to None, by default. Typically, you do not need to modify these values. However, you can make modifications according to the following items:

- You can modify the Access Mode field to make the task property read only.
- You can modify the In/Out field to one of the following values:
  - In
  - Out
  - In/Out

  The In/Out field allows you to use the task property as input, output, or input and output for the task UI. The default value is None. This configuration confines the task property to the task instance.

9 If you chose Integration Object as the Data Type in Step 6, then choose the Integration Object that Siebel CRM must use for the task property.

10 Optional. Enter comments in the Comments field, as necessary.

11 Step off the record to save your modifications.

7 Developing a Task UI

This chapter describes how to develop a task UI. It includes the following topics:

Roadmap for Developing a Task UI

To develop a task UI, perform the following processes and tasks:

1. Determine improvement requirements and design the task UI. For an example that describes this step, see "Example of Developing a Task UI" on page 22.
2. Process of Creating a Task UI on page 85
3. Process of Deploying a Task UI on page 164
4. Process of Testing a Task UI on page 151
5. Conditional. Migrating a Task UI on page 167
6. Administering a Task UI on page 171

For examples that describe how to develop a task UI, see Chapter 8, "Examples of Developing a Task UI." For more information, see "Customizing Task UI" on page 181.

Process of Creating a Task UI

This process is a step in "Roadmap for Developing a Task UI" on page 85.

To create a task UI, perform the following tasks and processes:

1. Preparing Siebel Tools to Create a Custom Task UI on page 86
2. Creating a Custom Task UI on page 86
3. Diagramming a Task UI on page 88
4. Creating a Task View on page 88
5. Binding a Task View to a Task View Step on page 91
6. Creating a Task Group on page 92
7. Adding a Task Group to a View on page 93
8. Refining a Task UI on page 94

Preparing Siebel Tools to Create a Custom Task UI

To create a custom task UI, you begin by preparing Siebel Tools.

To prepare Siebel Tools to create a custom task UI

1. Make sure the following parameter is set in the Siebel section of the tools.cfg file in the \TOOLS_ROOT\bin*language* directory:

   ```
   EnableToolsConstrain = FALSE
   ```

   If this parameter does not exist, then add it. Setting EnableToolsConstrain to false allows you to assign a display name to a task object.

2. Set the following parameter in the Workflow section of the configuration file:

   ```
   VerCheckTime = -1
   ```

   Setting VerCheckTime to -1 (negative one) allows you to use the Publish/Activate button. The configuration file is typically in the language directory on the client. For example:

   Siebel\81\21031\MWC\BIN*lang**configuration file*

   where:

   - *lang* is the language directory, such as ENU.
   - *configuration file* is the configuration file for the Siebel application, such as uagent.cfg.

3. Display object types that you use to develop a task UI.

   For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.

4. If necessary, display business service object types.

   A business service object can include hidden properties. To display hidden properties in a list in Siebel Tools for the following objects, make sure the Hidden property of the object is set to FALSE:

   - Business service
   - Business service method
   - Business service method argument

   For more information, see *Siebel Business Process Framework: Workflow Guide*.

Creating a Custom Task UI

The Task Wizard helps you define the minimal set of properties that a task UI requires. For more information about task properties that Siebel Tools automatically defines when you create a new task object, see "About the Task Property" on page 52.

To create a custom task UI

1. In Siebel Tools, choose the File menu, and then the New Object menu item.
2. In the New Object Wizards dialog box, click the Task tab.

   You can also right-click in the Tasks list, and then choose New Task Wizard to start the Task Wizard.
3. Choose the Task icon, and then click OK.
4. Use the wizard to create a new task UI:
   a. Define a project for the task UI.
   b. Define the name and display name for the task UI.

      Siebel CRM displays this display name in the task pane in the Siebel client.
   c. Indicate if the task UI is a subtask.

      You must never modify the Is Subtask property of a task UI or a subtask after you finish creating it. For more information, see "Diagramming a Task UI" on page 88.
   d. Define the business object that the task UI references.
   e. If necessary, define the transient business component that the task UI references.

      For more information, see "Determining if You Must Use a Transient Business Component" on page 97.
5. Click Finish.

   Siebel Tools creates a new object definition for the task UI, and then opens the Task Editor with the start and end steps already added.
6. Optional. To revise the task UI during development, use the Tasks list, as necessary.

Using the Tasks List to Create a Custom UI

You can use the Tasks list to create a custom task UI. However, it is strongly recommended that you use the Task Wizard instead of the Tasks list to create a custom task UI. The Task Wizard makes sure you define the minimal set of properties and objects that your task UI requires. For more information, see "Creating a Custom Task UI" on page 86.

To use the Tasks list to create a custom UI

1. In Siebel Tools, lock the project for the new task UI. If necessary, create a new project.
2. In the Object Explorer, click Task.
3. In the Tasks list, create a new task UI.

   At a minimum, define the following properties:
   - Task Name.
   - Project.

- Business Object.
- Indicate if the task UI is a subtask.

  You must never modify the Is Subtask property of a task UI or a subtask after you finish creating it. For more information, see “Diagramming a Task UI” on page 88.

Diagramming a Task UI

This task is a step in “Process of Creating a Task UI” on page 85.

The Task Editor allows you to create a visual representation of the entire task UI, including decision points and conditional logic. You can define the details of each step when you add each step in the Task Editor, or you can finish the entire flow, and then enter these details. Siebel Tools automatically creates the start and end steps for you when you create a new task object.

To diagram a task UI

1. Open the Task Editor.

   For more information, “Opening the Task Editor” on page 51.

2. Add one or more logic steps to the task UI.

   A task UI can include one or more steps that include some form of logic, such as a business service step, decision point, subtask step, Siebel operation step, and so on. More than one of each type of these steps can exist. For more information, see “Adding a Step to a Task UI” on page 52.

3. Add connectors to define the task flow.

   For more information, see “Creating a Connector” on page 78.

4. Repeat Step 2 and Step 3 until you correctly connect every step that the task UI contains.

Creating a Task View

This topic describes how to create a task view. It is strongly recommended that you use the Task View Wizard to create a task view. For more information, see the following topics:

- “About the Task View” on page 39
- “Guidelines for Designing User Interface Elements” on page 213

To create a task view

1. Make sure the following items that the task view references exist:
   - Standard applets.
   - Task applets.

- Transient business components. For more information, see "Overview of Transient Data" on page 96.

These objects must exist before you use the Task View Wizard.

2. In Siebel Tools, choose the File menu, and then the New Object menu item.
3. Click the Task tab, click the Task View icon, and then click OK.
4. In the Task View Wizard, in the New View dialog box, define the following items:
   - Project name
   - View name
   - View title
   - Business object of the view
   - Upgrade behavior

   Siebel Tools uses the value you enter in the View Name to enter data in the Name property of the task view. At run time, Siebel CRM displays the value you enter for the view title in the Siebel client.
5. Click Next.
6. In the View Web Layout - Select Template dialog box, choose a Web template for your view, and then click Next.
7. In the Web Layout - Applets dialog box, choose the applets that display in the task view, and then click Next.

   This dialog box lists predefined applets. You can choose task applets in a subsequent dialog box.
8. In the Task View - Pick Task dialog box, do one of the following:
   - Click Yes to specify a task applet.
   - Click No if you do not need to specify a task applet.

   This dialog box allows you to define a task applet for this view. If you click Yes, then you must create the task applet before you use this wizard. For more information, see "Overview of Transient Data" on page 96 and "Task Applet" on page 40.
9. Click Next.
10. If you clicked No in Step 8, then go to Step 13.
11. In the Task View - Select Task dialog box, click the task UI where Siebel CRM must display the task view, and then click Next.
12. In the Task View - Task Applets dialog box, choose applets that Siebel CRM must display in the task view, and then click Next.

13 In the Task View - Select Playbar Applet dialog box, choose the playbar applet that displays at the top and the bottom of your view, and then click Next.

   It is recommended that you include only one playbar applet to improve usability. If you use only one playbar, then leave one of the windows in this dialog box empty.

   You must use at least one playbar applet in a task view.

   For more information, see "Task Playbar Applet" on page 41.

14 In the Finish dialog box, click Finish.

   Siebel Tools opens the new view in the Web Layout Editor. The Finish dialog box displays the properties that Siebel Tools uses to create the new view.

15 If you set the Business Object property of the task view to Account, Contact, or Opportunity, then set the Visibility Applet Type property of the task view to All.

   For more information, see "Creating a Task View for Accounts, Contacts, or Opportunities" on page 90.

16 Edit the layout of the task view.

   For more information, see "Editing the Layout of a Task View" on page 90.

Creating a Task View for Accounts, Contacts, or Opportunities

Siebel CRM denormalizes some business component data in intersection tables so that all the columns that it requires to do sort or search operations reside on a single table. This design is preferable to using columns that reside in the base table and in the intersection tables for performance reasons. A view that includes the Visibility Applet Type set to All does not require an intersection table to determine record visibility. For example, opportunities use the intersection tables for organizations.

To create a task view for accounts, contacts, or opportunities

- If you set the Business Object property of a task view to Account, Contact, or Opportunity, then set the Visibility Applet Type property of this view to All.

Editing the Layout of a Task View

The Web Layout Editor allows you to edit the mapping that Siebel Tools uses between applets that exist in the view and placeholders that exist in the template.

To edit the layout of a task view

1 In Siebel Tools, in the Object Explorer, click View.

2 In the Views list, query the Name property for the view you must modify.

3 In the Views list, right-click the view you located in Step 2, and then choose Edit Web Layout.

4 Preview the view:

a. Right-click in the Web Layout Editor, and then choose Preview.

   Siebel Tools displays the view in a preview mode that is similar to the way Siebel CRM renders it at run time in the Siebel client.

b. To return to editing mode, right-click in the Web Layout Editor, and then choose Preview.

5. Modify the applet sequence, as necessary:

   a. In the Object Explorer, expand the View tree, and then click View Web Template.

   b. Choose a row in the Web Templates list.

   c. In the Object Explorer, expand the View Web Template tree, and then click View Web Template Item.

   d. To define the sequence for each item, in the View Web Template Items list, use the Item Identifier property.

6. Modify the applet mode, as necessary:

   a. In the Object Explorer, click View Web Template.

   b. In the Web Templates list, choose a row.

   c. In the Object Explorer, click View Web Template Item.

   d. In the View Web Template Items list, use the Applet Mode property to define the applet mode for each item.

For more information about:

- Using the Web Layout Editor, see the topic on creating screens and views in *Configuring Siebel Business Applications*.
- Using an external editor to modify the Web template file, see *Using Siebel Tools*.

Binding a Task View to a Task View Step

You must configure Siebel CRM to bind the task view to the task view step so that it displays a task view and a task view step for this task view. For more information, see "Creating a Task View Step" on page 67.

To bind a task view to a task view step

1. Open the Task Editor for the task UI where Siebel CRM must bind a task view.

   For more information, see "Using the Task Editor" on page 51.

2. Right-click the task view step where Siebel CRM must bind a task view, and then choose Bind Task View.

3. In the following dialog box, choose a task view, and then click OK:

```
Please Bind a Task View From the Following List
```

This dialog box lists the task views that you define for the task UI. If it does not list the task view that you require, then you can click New to create a new one. If you click New, then the New Task View Wizard starts.

If you click OK, then Siebel Tools displays the name of the view that the task view step references. It displays this name as the label for the task view step in the Task Editor.

4 Define properties for the task view step.

Some properties on the task view step, such as Disable Cancel, allow you to specify behavior of the task playbar. The Forward Button Type property allows you to specify when Siebel CRM saves information to the Siebel database. For more information, see "Task Playbar Applet" on page 41.

Creating a Task Group

This topic describes how to create a task group. For more information, see "Task Group" on page 36.

To create a task group

1 Make sure Siebel Tools is properly prepared to create a task group.

For more information, see "Preparing Siebel Tools to Create a Custom Task UI" on page 86.

2 In the Object Explorer, click Task Group.

3 In the Task Groups list, add a new task group using values from the following table.

Property	Description
Name	Enter a name for the task group.
Project	Choose a project for the task group.
Display Name - String Override	Optional. Enter a value for the task group label. Siebel CRM displays this label in the context pane in the Siebel client. For more information, see "Context Pane" on page 32.
Require Context BC	Optional. If this task group includes a task that is context-sensitive, then make sure the Require Context BC option includes a check mark. For more information, see "Using a Context Business Component with a Task Group" on page 37.

4 Make sure the task group you added in Step 3 is still chosen.

5 In the Object Explorer, expand the Task Group tree, and then click Task Group Item.

6 In the Task Group Items list, add a new task group item.

Siebel CRM displays the task group items that you define in this step as a list of links under the task group label in the context pane.

7 Define properties for the new task group item, using values from the following table.

Property	Description
Type	Choose Task. The value you define in the Type property determines the values that Siebel Tools displays in the Action Invoked property, so you must define the Type property first.
Action Invoked	Choose the task UI that you must start. The Action Invoked property instructs the task UI to open when the user clicks the link in the context pane. For more information, see "Context Pane" on page 32.
Context Business Component	If you placed a check mark in the Require Context BC option in Step 3, then choose a Context Business Component. Otherwise, leave this property empty.
Sequence	Optional. Enter a numerical value. The Sequence property allows you to define the order that Siebel CRM uses to display the links in the context pane. For more information, see "Task Order in Task Groups" on page 37.

Adding a Task Group to a View

To add a task group to a view, do the procedure described in one of the following topics:

- "Adding a Task Group to a View" on page 93
- "Adding a Task Group to Multiple Views" on page 94

For more information, see "Task Group" on page 36.

A number of factors determine how Siebel CRM displays a task UI in the task pane. For more information, see "Task UI Does Not Display in the Task Pane" on page 160.

Adding a Task Group to a View

This topic describes how to add a task group to a view so that Siebel CRM displays it only in a specific view. For example, for a task UI that assists the user with creating a new opportunity, you can define the view so that the task group only displays in the task pane when the user is currently in the Opportunity List View. You can also limit display of the task group according to the Siebel application that is running in the Siebel client, such as Siebel Call Center.

To add a task group to a view

1 In the Object Explorer, click View.

2 In the Views list, query the Name property for the view where you must add a task group.

3 In the Object Explorer, expand the View tree, and then click View Task Group.

4 If necessary, display the View Task Group object type.

For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.

5 In the View Task Groups list, add a new view task group.

6 Optional. Configure Siebel CRM to display the task group only in a single Siebel application.

If you require Siebel CRM to display the task group only in a single Siebel application, then choose this application in the Application property.

Adding a Task Group to Multiple Views

This topic describes how to add a task group so that Siebel CRM displays it across multiple views. For example, you can add the task group to the Task Pane View for a task UI that assists the user with creating a new opportunity but where the user must access the task from across multiple views.

To add a task group to multiple views

1 In Siebel Tools, in the Object Explorer, click View.

2 In the Views list, query the Name property for Task Pane View.

3 In the Object Explorer, expand the Views tree, and then click View Task Group.

4 In the View Task Groups list, add a new view task group.

5 Click the Task Group property, and then choose the task group from the Task Groups pop-up applet.

At run time, Siebel CRM displays this task group globally across views in a Siebel application.

Refining a Task UI

This task is a step in "Process of Creating a Task UI" on page 85.

This topic describes how to refine your task UI to more precisely meet design requirements.

To refine a task UI

- Add conditional logic to your task UI.

  You create conditions and values that affect task flow to add conditional logic to a task UI. You create branching in a task on branch connectors that emanate from a decision point. If your task includes a decision point, then you must create conditional branching. For more information, see "Creating a Branch Connector" on page 78.

- Add a subtask.

  If your task UI includes more than 15 steps, then consider separating the task into one or more subtasks. For more information, see "Creating a Subtask Step" on page 73.

- Add a commit step.

  You can use a commit step to save temporary data to the Siebel database. For more information, see "Creating a Commit Step" on page 75.

- Create logic to handle an error condition.

  You can create logic in your task UI to handle an error that occurs at run time, such as a business service that returns an internal error message. For more information, see "Creating an Error Step" on page 76.

- Collect task metrics.

  Task metrics collect and store task data that Siebel CRM regularly saves in a data warehouse. You can use an Online Analytical Processing (OLAP) tool, such as Oracle Business Intelligence, to analyze this data. For more information, see "About Task Metrics" on page 234.

- Refine UI objects:

  - Various UI objects can improve the user experience. For example, a task chapter can provide the user with a map of what lies ahead in completing the task UI, and what work the user finished in the task. For more information, see "Other Options for Customizing a Task UI" on page 200.

  - You can use various usability techniques to improve the user experience. For more information, see "Techniques to Improve the Usability of a Task UI" on page 218.

- Create logic to resume a paused task UI.

  If the user must pause a task UI and at some future point the same or another user must resume it, then you can create a relationship between the task instance and a business object instance. For more information, see "Resuming a Paused Task UI" on page 187.

Developing a Task UI That Uses Transient Data

This topic describes how to develop a task UI that uses transient data. It includes the following topics:

Overview of Transient Data

Transient data is a type of data that is relevant for a limited time period. In a task UI, this time period is the duration of a task instance. The answer to the following question that Siebel CRM displays in a task UI is a typical example of transient data:

```
What would you like to do next?
```

A *transient business component* (TBC) is a type of business component whose records exist only during the duration of a task instance. It allows you to configure Siebel CRM to create data that is specific to a task instance that it can display and that the user can edit and access in a task UI. Using a transient business component is conditional, depending on your data manipulation requirements.

Transient data is dynamic and is tied to a transient business component in a task UI. Siebel CRM accesses the records of a transient business component only in the context of a specific task UI or subtask.

A task UI can assign transient data to the input method arguments of a task step or to a task property through the output argument of a task step.

In contrast to a business component, a transient business component stores data that Siebel CRM requires for the duration of the task instance but that it can discard when the user finishes the task UI. Siebel CRM might require this data to support the following situations:

- To perform an intermediate calculation.
- To store information about the decisions the user makes.
- To temporarily store data. For more information, see "Storing Data Temporarily While Gathering the Information Required to Create a Complete Record" on page 103.
- For other purposes where Siebel CRM does not need to store data in the Siebel database.

For examples, see the following topics:

- For an example that uses transient data, see "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.
- For an example that does not use transient data, see "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

Characteristics of a Transient Business Component

The following characteristics describe a transient business component:

- Stores transient data, including data that controls task flow.
- Defined in a way that is similar to how you define a business component except that it references the S_TU_LOG table and is of the transient type.
- Deployed the same way that you deploy a business component, which is through the SRF (Siebel Repository File).
- Cleared after a commit step or when the user finishes a task UI.

- Supports a calculated field. The Calculated Field property indicates if the transient business component calculates the value of the field. If set to TRUE, then the transient business component calculates the value.
- Does not support multivalue fields.
- Does not support an explicit join.
- Is not stored in the Siebel database.
- Does not support an insert, copy, or delete operations on a business component record. You can use a business component to insert, copy, or delete a business component record.

How Siebel CRM Manages Transient Data

Siebel Tools automatically enters in the following properties of a transient business component when you create it. These properties are not editable:

- The Name property specifies the Siebel database column for the table.
- The Join properties define a logical join between the base table of a business component and another table. If Siebel CRM creates a new record, then it automatically enters data in the Join properties according to the Type and Text length properties.

Siebel CRM forces all columns to active at run time to avoid problems with field activation.

Siebel CRM stores transient data in the S_TU_LOG table and, if necessary, in S_TU_LOG_X_* extension tables:

- The TASK_ID_VAL column stores the task instance ID.
- The BC_NAME column stores the name of the business component.

Siebel CRM finishes using a transient business component, and then a background service cleans the S_TU_LOG table according to the value in the TASK_ID_VAL column. If a task UI is running on a Siebel Mobile Web Client that is connected to a local database, then Siebel CRM deletes the transient data as soon as the task UI finishes.

For more information, see "Creating a Transient Business Component" on page 98.

Determining if You Must Use a Transient Business Component

This topic describes how to determine if you must use a transient business component.

To determine if you must use a transient business component

1. If one the following situations is true, then you must use a transient business component. If the following situations are not true, then do not use a transient business component. Use a business component instead:
   - A radio button or decision point determines the task flow.

- The task UI must temporarily store data. For more information, see "Storing Data Temporarily While Gathering the Information Required to Create a Complete Record" on page 103.
- You cannot use the Defer Write Record property to store transient data. For more information, see "Using the Defer Write Record Property of a Siebel Operation Step Instead of a Transient Business Component" on page 104.
- You must control a search specification. For example, if a user must make a choice that includes a predefault value, such as in reply to the following question:

  ```
  In what month did the transaction occur?
  ```

- If a task UI must modify transient data while the task runs:
  - To use fields from different business components in a single applet. For example, to allow the user to create a new account and enter contact data in a single task applet.
  - To modify the appearance of the user interface depending on run-time conditions. For example, according to a choice that the user makes, UI controls can map to different fields of a business component. If the user chooses Credit Card as a payment method, then fields for credit card type, number, and expiration date are available at run time.
  - To determine if a record already exists before committing a new record that might be a duplicate.

2 If you determined in Step 1 that you must use a transient business component, then create one now.

  For more information, see "Creating a Transient Business Component" on page 98.

Creating a Transient Business Component

If your task UI requires transient data, then you must create a transient business component. It is strongly recommended that you use the Transient Business Component Wizard to create a transient business component. For more information, see "Overview of Transient Data" on page 96.

To create a transient business component

1 Determine if your task UI must use a transient business component. If your task UI does not require a transient business component, then quit this procedure.

  For more information, see "Determining if You Must Use a Transient Business Component" on page 97.

2 In Siebel Tools, choose the File menu, and then the New Object menu item.

3 In the New Object Wizards dialog box, choose the Task tab.

4 Choose the Transient BusComp icon, and then click OK.

5 In the New Business Component Wizard, define properties in the New Business Component dialog box using values from the following table.

Property	Description
Project	Choose a value from the drop down list. The wizard only lists projects that are locked.
Name	Enter a text string that describes the task UI.
MultiRecord TBC checkbox	Optional. If you are creating a multirecord transient business component, then make sure the MultiRecord TBC option includes a check mark. For more information, see "About the Multirecord Transient Business Component" on page 104.

6 Click Finish.

The wizard automatically does the following work:

- Assigns the transient business component to the S_TU_LOG table.
- Sets the Type property of the business component to Transient.
- Displays a new record for the transient business component in the Business Components list.

7 In the Object Explorer, expand the Business Component tree, and then click Field.

8 In the Fields list, create a new field for the transient business component.

9 Define values for the following properties of the field you created in Step 8:

- Name
- Type
- Text Length

CAUTION: The Type property is a required property. The Text Length property is not a required property. However, it is recommended that you define a value for the Text Length property so that Siebel CRM can accurately determine the size of the Column property. If a field of type DTYPE_TEXT includes an undefined text length, then Siebel CRM uses a length of 255 characters. If the actual length is shorter than 255 characters, and if you do not define the Text Length property, then Siebel CRM might create an unnecessary join to an extension table, or even fail to map the field to the correct corresponding column.

Note the following:

- Siebel Tools automatically enters data in the Column property.
- The Column and Join properties are read only.
- You cannot define a multivalue field.
- Siebel Tools allows a calculated field. It does not assign a column to a calculated field.

10 Repeat Step 8 and Step 9 for each additional field you must define.

Setting the Type Property of a Business Component

Note the following:

- The Type property for a business component is Nontransient.
- The Type property for a transient business component is Transient.

If you set the Type property for a business component to Transient or Nontransient, then you cannot modify it, even through a copy operation.

Creating a Transient Business Component Manually

You can use the Business Components list to manually create a transient business component. However, it is recommended that you use the Transient Business Component Wizard to create a transient business component instead of using the Business Components list. Siebel Tools automatically defines the Column property and the Join property and sets them to read only. Make sure you set the following properties to exactly the same values that the Transient Business Component Wizard uses:

- Class
- Table
- Type

It is strongly recommended that you use the Transient Business Component Wizard to make sure that you correctly define the CSSBCTaskTransient class and the CSSBCTaskTransientBase class. For more information, see "Classes That Siebel CRM Uses with a Transient Business Components" on page 105.

Creating a Task Applet

If your task does not involve transient data, then you can use a standard applet instead. For more information, see Task Applet on page 40.

If your task UI does require a task applet, then you can create it before you create the task view. For more information, see "Creating a Task View" on page 88. For an example of creating a task applet, see "Creating the Task Applet" on page 121.

To create a task applet

1. Make sure the following objects exist:
   - The object definition of the task UI
   - The transient business component that the task applet references

   These objects must exist before you create the task applet.
2. In Siebel Tools, choose the File menu, and then the New Object menu item.
3. In the New Object Wizards dialog box, click the Task tab.
4. Choose one of the following icons, and then click OK:

- Task Form Applet
- Task List Applet

5 In the General dialog box, define properties using values in the following table.

Property	Description
Project	Choose the project that this applet must reference.
Name	Enter a name for this applet. The name cannot be the same as the name of an existing applet.
Display Title	Optional. Enter the display title that Siebel CRM displays in the Siebel client. To display no title, you can leave the Display Title property empty.
Task	Choose the task UI that this applet must reference. If you do not choose a task, then you can reuse this applet with multiple tasks. If you choose a task, then you cannot use this applet with any other task.
Upgrade Behavior	Choose Preserve. This setting preserves this modification during an upgrade.
Transient Business Component	Choose the transient business component that stores data for this applet.

6 Click Next.

7 In the Web Layout - Fields dialog box, choose fields from the transient business component that provide data for the layout.

To choose a field, click it in the Available Fields window, and then click the arrow that points toward the Selected Fields window. You can hold down the SHIFT key to choose multiple fields.

8 Click Next.

The Finish dialog box displays. This dialog box displays properties that Siebel Tools uses to define the new task applet. You can click Back to return to an earlier dialog and make modifications, if necessary.

9 Click Finish.

Siebel Tools creates the task applet and then opens the Web Template Layout Editor.

10 Adjust the applet layout and modify properties for the controls, as necessary.

Transferring Data with a Transient Business Component

Siebel CRM transfers data between business components in the same way regardless of whether or not a transient business component is involved. You can configure Siebel CRM to transfer data between a transient business component or a nontransient business component.

To transfer data from a transient business component to a task property

1 Open the Task Editor.

For more information, "Opening the Task Editor" on page 51.

2 In the Task Editor, choose the step that displays after the assignment occurs.

This step is typically a task view step that allows the user to modify the data of a transient business component.

3 In the Multi Value Property Window, click the Output Arguments tab.

4 Assign a field of the transient business component to a task property.

5 For the assignment type, use Business Component.

To transfer data between a transient business component and another business component

1 Open the Task Editor.

For more information, "Opening the Task Editor" on page 51.

2 Add a Siebel operation step using values from the following table.

Property	Value
Operation	Update

3 Create an output argument:

a Choose the Siebel operation step you added in Step 2.

b In the Multi Value Property Window, click the Output Arguments tab.

c Right-click in the list area of the Multi Value Property Window, and then choose New Record.

d Define the new output argument using values from the following table.

Property	Description
Property Name	Choose the property that contains the data you must transfer.
Type	Choose Business Component.
Business Component	Choose the business component that receives the data you are transferring.
Business Component Field	Choose the business component field that receives the data you are transferring.

This step updates the target business component.

Guidelines for Using a Transient Business Component

If you use a transient business component, then use the following guidelines:

- Do not use a transient business component unless your task UI requires it. For more information, see "Determining if You Must Use a Transient Business Component" on page 97.
- If you use a transient business component that references the Contact business object, then you must add it to the Contact business object. Otherwise, Siebel CRM might create a runtime error.
- Siebel CRM might consider data that resides in a task property as transient data. The source of the data determines how Siebel CRM can display it. You must consider the following capabilities when you decide to store transient data in a transient business component or in a task property:
  - Siebel CRM can display data that it gets from a transient business component in the Siebel client.
  - Siebel CRM cannot display data that it gets from a task property in the Siebel client.
- You can enter data from a transient business component into a task property or a business component field.

Storing Data Temporarily While Gathering the Information Required to Create a Complete Record

If a task UI uses more than one view to get the information it requires to create a complete record, then you must use a transient business component. Task UI cannot insert a record in a business component until it collects a value for every field that the business component requires for a complete record. For example, assume you create a task UI that includes three separate views that do the following:

- In the first view, the user enters the opportunity name.
- In the second view, the user enters the close date.
- In the third view, the user enters the currency.

An opportunity record must include a value for each of the following required fields:

- Opportunity Name
- Close Date
- Currency

You must use a transient business component to collect this information across views in the task UI. You can configure the task UI to copy data from the transient business component to the business component record when this task UI finishes.

Using a Transient Business Component in a Subtask

A transient business component can include one or multiple instances for each context. You can configure Siebel CRM to use multiple instances when you create the transient business component. The context of the transient business component resides at the level of the subtask. The subtask and the parent task do not share the same instance of the transient business component even if the subtask uses the same transient business component that the parent task UI uses. If no instance of a transient business component exists in the current context, then Siebel CRM creates a new instance when it queries the transient business component. For more information, see "Creating a Subtask Step" on page 73.

Using a Transient Business Component with a Commit Step

Siebel CRM erases transient data that resides in the S_TU_LOG table when a commit step saves data. You can configure Siebel CRM to create a new instance of the transient business component and display it in a task view step after Siebel CRM clears the commit data and the transient data. Siebel CRM does not clear new data from the S_TU_LOG table. If more modifications occur in the transient business component, then Siebel CRM updates the same record in the S_TU_LOG table. It does not clear the older data and it does not create a new record until it issues a commit. For more information, see "Creating a Commit Step" on page 75.

Using the Defer Write Record Property of a Siebel Operation Step Instead of a Transient Business Component

The Defer Write Record property on a Siebel operation step postpones a write operation until the task flow encounters a subsequent commit step. If the only data storage requirement for your task UI is to store the data that is involved with a Siebel operation step, then you can use the Defer Write Record property instead of a transient business component and task applet. For more information, see "About the Defer Write Record Property" on page 71.

About the Multirecord Transient Business Component

A *multirecord transient business component* is a type of transient business component that can include more than one record for a given task UI context. The following examples describe situations where you might use a multirecord transient business component:

- To allow the user to choose a single record from several records during the course of completing a task UI. For example, the user might create several plans but near the end of the task UI choose the plan that most closely matches the business requirement. While the task runs, Siebel CRM saves each plan as a separate record in the multirecord transient business component. It saves only a single record to the Siebel database after the task finishes.
- To automate the creation of records for a merge. For example, assume your deployment must automate record creation for a merge. This operation is similar to mimicking a three way merge where Siebel CRM automatically creates three records for a user, and then asks the user to choose the record that it must resolve. The record that the user chooses is the only record that it saves to the Siebel database. It clears the remaining records.

A multirecord transient business component can do some of the same operations that a standard business component can do, such as insert, update, delete, copy, or undo. You can use a multirecord transient business component in a Siebel operation step in the same way that you use a standard business component.

Siebel CRM clears the records in a multirecord transient business component when the task UI finishes or if the user cancels it. If the user pauses the task, then these records remain in temporary storage.

Classes That Siebel CRM Uses with a Transient Business Components

Table 13 describes classes that Siebel CRM uses with a single record transient business component and with a multirecord transient business component. The default setting is single record.

If you create a transient business component, then it is strongly recommended that you use the New Business Component Wizard to make sure that you properly define the CSSBCTaskTransient class, the CSSBCTaskTransientBase class, or a class that Siebel CRM derives from these classes.

Table 13. Classes That Siebel CRM Uses With a Transient Business Component

Type	Description	Class
Single record transient business component	Supports a single record in a task UI context.	Uses the CSSBCTaskTransient class or a class that Siebel CRM derives from the CSSBCTaskTransient class. The CSSBCTaskTransient class makes sure only one row exists for each context for each single record transient business component. The context of a transient business component resides at the level of the subtask. This means that even if a subtask uses the same single record transient business component with the parent task, then the two subtasks do not share the same transient business component record. This specialized CSSBCTaskTransient class also filters the single record for the current context and the current business component. It runs a default query on the first Get function or Set function, and it creates a new record if no such record exists. That is, if the following method returns no rows: `Execute`
Multirecord transient business component	Supports multiple records in a task UI context.	Uses the CSSBCTaskTransientBase class or a class that Siebel CRM derives from the CSSBCTaskTransientBase class.

8 Examples of Developing a Task UI

This chapter describes examples of developing a task UI. It includes the following topics:

Example of Modifying a Predefined Task UI

To modify the predefined FS Asset To Contract Task, perform the following tasks:

The work that you do to modify a predefined task UI varies depending on the task you must modify. In this topic you modify the relatively complex FS Asset To Contract Task. Siebel CRM displays the following message in the Siebel client when it runs this task UI:

```
Select a Service Package to be Added
```

You must create some products with Type equal to Service to make sure Siebel CRM displays data in this view. You activate the task, add product data, and then verify task functionality. A significantly different configuration might be required to configure a different predefined task UI.

For more information, see "About Predefined Task UIs" on page 16.

Activating the Task UIs and Subtasks

This task is a step in "Example of Modifying a Predefined Task UI" on page 107.

In this topic, you activate the task UIs and subtasks.

To activate the task UIs and subtasks

1. In the Siebel client, navigate to the Administration - Business Process screen, and then the Task Deployment view.
2. Query the Name column for FS Asset To Contract Task, and then click Activate.
3. Repeat Step 2 for the following tasks:
   - FS Cover Asset SubTask
   - FS Submit Agreement Sub Task
4. Verify that Siebel CRM displays each task UI in the Active Tasks list and that the Deployment Status for each task is Active.
5. Navigate to the Administration - Application screen, and then the Tasks view.
6. Verify that each of the tasks you activated in Step 2 and Step 3 includes the Siebel Administrator responsibility:
   a. In the Registered Tasks list, click each view in succession.
   b. As you click each view, in the Responsibilities list, verify that a record exists with Siebel Administrator in the Responsibility field.

      If this record does not exist, then add it now. Leave the Allow Delete and Allow Transfer fields with their default check mark.

   For more information, see "Adding a Responsibility to a Task UI" on page 166.

Adding the Product Data for the Asset to Contract Task

This task is a step in "Example of Modifying a Predefined Task UI" on page 107.

In this topic, you add the product data.

To add the product data for the Asset to Contract Task

1. Navigate to the Administration - Product screen, and then the Products view.
2. Create a new product using values from the following table.

Field	Description
Name	Choose Technical Support 24x7.
Type	Choose Service.
Service Product	Make sure this field includes a check mark.

3. Make sure the following product is chosen:

   `Technical Support 24x7`

4 Click the Product Definitions link.

Note the default record that Siebel CRM displays in the Versions list with Version set to Work Space.

5 Define the Start Date field to be midnight of the date for today.

For example, if today is July 1, 2013, then enter the following date:

```
7/1/2013 12:00:00 AM
```

Leave the end date field empty.

6 In the Products list, click Release.

Siebel CRM releases the product version for this configuration, causing a new version to display in the Versions list with 1 in the Version field and the following value in the Start Date field:

```
7/1/2013 12:00:00 AM
```

7 Navigate to the Administration - Contracts screen, and then the Term Templates view.

8 Create a term template using values in the following table.

Field	Value
Term #	TS Term 1
Term Name	Technical Support Terms
Type	Standard
Description	This is a test Agreement Term Save record.

9 Log out of the Siebel client.

Verifying the Functionality of the Asset to Contract Task

This task is a step in "Example of Modifying a Predefined Task UI" on page 107.

In this topic, you verify the functionality of the Asset to Contract task.

To verify the functionality of the Asset to Contract task

1 Log in to the Siebel client, click the Accounts screen tab, and then click the Tasks icon.

2 Verify that the task pane displays the task UI named Agreement Creation Task.

3 Click the following link that Siebel CRM displays under the Agreement Creation Task:

```
Assets to Agreement
```

4 Verify that the task pane displays the following task groups:

- Create / Revise Agreement

- Cover Assets
- Submit Agreement.

5 In the Identify Agreement task view, make sure the Use Existing Agreement radio button includes a check mark, and then click Next.

6 In the Select an Agreement for Revision task view, choose an agreement from the list, and then click Next.

7 On the Agreement Detail task view, note that Siebel CRM enters data in some fields according to the choice you made in Step 6, and then click Next.

8 On the Add Service Package task view, click Continue.

The choice you make alters the task flow:

- If you click Continue, then Siebel CRM runs the FS Submit Agreement Sub Task.
- If you click Use Existing Line Item, then Siebel CRM displays the Agreement Line Item List View.

To verify this logic in the Task Editor, you can examine conditions on connectors that emanate from the Create New Line Item decision point. For more information, see "Examining the Logic of a Task UI" on page 16.

You can chose one of the other options on a subsequent test of this task UI to view how the task logic varies according to the condition that is met.

9 On the Add Agreement Terms task view, click Add, add the Technical Support Terms term, and then click Next.

Siebel CRM displays the Agreement Terms Task View of the subtask named FS Submit Agreement Sub Task. You can verify this logic in the Task Editor.

10 On the Submit Agreement for Approval task view, click Continue, and then click Next.

11 On the Generate Agreement Document task view, click Auto Document, and then click Submit.

If you click Submit, then Siebel CRM updates the base table with the modification that you made to the agreement. If you add a new agreement, then the All Agreements List view displays a new record for the agreement.

Example of Developing a Task UI That Assists with Adding an Opportunity

The example in this topic describes how to develop a task UI that assists the user with adding an opportunity. To develop this example, perform the following tasks:

1 Creating and Diagramming the Task UI on page 111

2 Creating the Applet on page 112

3 Creating the Task View on page 114

4 Binding the Task View To the Task View Step on page 115

5 Controlling the Buttons of the Playbar Applet on page 115

6 Creating the Task Group on page 116

7 Publishing and Activating a Task UI in Siebel Tools on page 167

8 Verifying the Task UI on page 117

The data manipulation and storage requirements for this example do not involve transient data, so you do not use a transient business component or a task applet. Instead, you use a standard business component and create a standard applet. For more information, see the following topics:

- "Overview of Transient Data" on page 96
- "Task Applet" on page 40

Creating and Diagramming the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

In this topic, you create and diagram the task UI.

To create and diagram the task UI

1 Make sure Siebel Tools is prepared to create a task UI.

   For more information, see "Preparing Siebel Tools to Create a Custom Task UI" on page 86.

2 In the Object Explorer, click Project.

3 In the Projects list, create a new project named Opportunity Task UI.

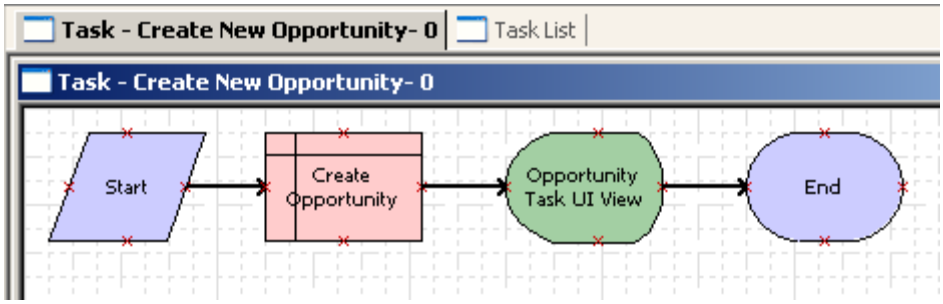
   You can use this project to support this development effort. For more information, see *Using Siebel Tools*.

4 Use the Task Wizard to create a new task UI using values from the following table.

Property	Description
Project	Choose Opportunity Task UI.
Task Name	Enter Create New Opportunity.
Display Name	Enter Create New Opportunity.
Business Object	Choose Opportunity.
Transient Business Component	Leave this field empty.
Subtask	Make sure this option does not include a check mark.

   For more information, see "Creating a Custom Task UI" on page 86.

5 Diagram the task UI to resemble the flow that the following diagram displays:



Siebel Tools displays the label for the task view step after you bind the view in a subsequent step in this example. For more information, see "Diagramming a Task UI" on page 88.

6 Click the Create Opportunity step, and then use the Properties window to define properties using values in the following table.

Property	Value
Business Component	Opportunity
Defer Write Record	TRUE
Operation	Insert

For more information, see "About the Defer Write Record Property" on page 71.

7 Save your work and then close the Task Editor.

Creating the Applet

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

In this topic, you create a new standard applet that the user uses to enter information about an opportunity.

To create the applet

1 In Siebel Tools, choose the File menu, and then the New Object menu item.

2 In the New Object Wizards dialog box, click the Applets tab.

3 Click the Form Applet icon and then click OK.

4 In the General dialog box, define fields using values from the following table, and then click Next.

Property	Description
Project	Choose Opportunity Task UI.
Name	Enter Opportunity Task UI Form Applet.
Display Title	Enter Opportunity.
Business Component	Choose Opportunity.
Upgrade Behavior	Choose Admin.
Use Grid Layout	Make sure this option includes a check mark.

5 In the Web Layout - Form dialog box, accept the default Edit Mode, and then click Next.

6 In the Web Layout - Fields dialog box, move the following fields from the Available Fields window to the Selected Fields window, and then click Next:

- Name
- Description
- Primary Revenue Amount
- Primary Revenue Close Date

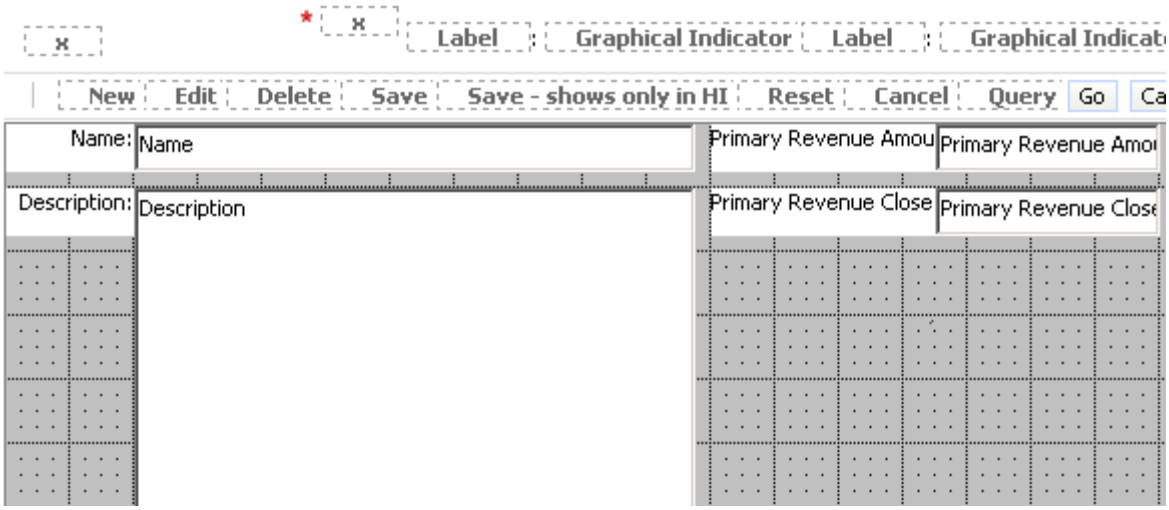
7 In the Web Layout - Fields dialog box, move fields that Siebel Tools lists in the Available Controls window to the Selected Controls window, and then click Next.

Siebel Tools might already display fields for this dialog box in the Selected Controls window.

8 In the Finish dialog box, review your settings and then Click Finish.

Siebel Tools creates the new applet and then opens the Web Layout Editor.

9 In the Applet Web Template window, reposition controls and labels until the applet resembles the layout that the following diagram displays:



10 Choose the Description control, and then use the Properties window to set the property described in the following table.

Property	Value
HTML Type	TextArea

You set the HTML Type property, and then Siebel Tools displays a scroll bar along the right side of the Description control.

11 Save your work and then close the Applet Web Template window.

Creating the Task View

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

In this topic, you create a task view that displays the applet that allows the user to enter opportunity details. You defined this applet in "Creating the Applet" on page 112.

To create the task view

1 Create a task view.

For more information, see "Creating a Task View" on page 88.

2 In the New View dialog box, enter values described in the following table.

Property	Value
Project	Opportunity Task UI
View Name	Opportunity Task UI View
View Title	Opportunity
Business Object	Opportunity
Upgrade Behavior	Admin

3 In the View Web Layout - Select Template dialog box, choose the following template:

View Detail (Parent with Pointer)

4 In the Web Layout - Applets dialog box, chose the following applet, and then click Next:

Opportunity Task UI Form Applet

The Opportunity Task UI Form Applet is the custom applet you created in "Creating the Applet" on page 112.

5 In the Task View - Pick Task dialog box, click No, and then click Next.

6 In the Task View - Select Playbar Applet dialog box, choose the Task Playbar Applet - Bottom applet as the bottom playbar applet, and then click Next.

7 Click Finish, verify that the layout is correct, and then close the Web Layout Editor.

Binding the Task View To the Task View Step

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

In this topic, you bind a task view to the task view step.

To bind the task view to the task view step

- Bind the task view step that resides immediately downstream of the Create Opportunity Siebel operation step to the Opportunity Task UI View.

  For more information, see "Binding a Task View to a Task View Step" on page 91.

Controlling the Buttons of the Playbar Applet

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

You configure properties on the task view step to control the buttons that are active in the playbar applet. You also set the display name for the task view.

To control the buttons of the playbar applet

1 In the Task Editor, click the Opportunity Task UI View step, and then use the Properties window to define properties described in the following table.

Property	Description
Disable Previous	Choose TRUE.
Display Name - String Override	Enter Opportunity Details.
Forward Button Type	Choose Submit.

2 Save your modifications and then close the Task Editor.

Disabling Backward Navigation

You can set the Disable Previous property on the task view step to TRUE to prevent backward navigation. You can use this feature for some situations, such as a summary page where Siebel CRM already saved the task transaction to the Siebel database, or with a business process that started processing the transaction, so going back to modify the transaction is no longer possible.

Creating the Task Group

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

In this topic, you create a task group for the Create New Opportunity task UI. For this example, you create the task group so that Siebel CRM displays it only when it also displays the Opportunity List View. For more information, see "Task Group" on page 36.

To create the task group

1 Create the task group:

a Create the new task group using values from the following table.

Property	Value
Name	Opportunity Task Group
Project	Opportunity Task UI
Display Name - String Override	Opportunity Tasks

b Create the new task group item using values from the following table.

Property	Value
Action Invoked	Create New Opportunity
Type	Task
Sequence	1

For more information, see "Creating a Task Group" on page 92.

2 Define the task group to display in a specific view.

Choose Opportunity Task Group from the pop-up applet when you define the Task Group property in the View Task Groups list, and then define the property described in the following table.

Property	Value
Sequence	2

For more information, see "Adding a Task Group to a View" on page 93.

Verifying the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

In this topic, you verify that the task UI delivers the required functionality. For more information, see "Verifying Functionality of a Task UI" on page 154.

To verify the task UI

1 Log in to the Siebel client, click the Tasks icon, and then verify that the task pane does not display in the Opportunity Tasks task group.

You defined this task UI to display only when the Opportunity List View is the current view.

2 Click the Opportunities screen tab, and then click the Opportunities List link.

3 Verify that the task pane now displays the Opportunity Tasks task group.

4 In the task pane, click the Create New Opportunity link, and then verify that Siebel CRM displays the Enter Opportunity Details view and that it correctly arranges the controls.

The layout must be similar to the layout that you defined in "Creating the Applet" on page 112.

If the layout requires adjustment, then you can rapid prototype the applet layout. For more information, see "Rapid Prototyping the Layout of an Applet" on page 139.

5 Verify the following:

- The Pause, Submit, and Cancel buttons are active.

- The Previous button is disabled.

6 Enter a value in the following fields, and then click Submit:

- Name
- Description
- Primary Revenue Amount

Siebel CRM closes the task UI and then displays the standard Opportunity List View.

7 Verify that Siebel CRM includes the opportunity you defined in Step 6 in the opportunity list, and that the opportunity fields include the values you entered in the task UI.

Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity

The example in this topic describes how to develop a task UI that assists the user with adding a new opportunity and an activity for this opportunity. To develop this example, perform the following tasks:

1 Creating and Diagramming the Task UI on page 118

2 Creating the Transient Business Component on page 119

3 Creating the Picklist on page 120

4 Creating the Task Applet on page 121

5 Adding the Task Applet to the View on page 122

6 Creating the Task View on page 123

7 Revising the Task UI on page 124

8 Creating the Condition Criteria on page 126

9 Publishing and Activating a Task UI in Siebel Tools on page 167

10 Verifying the Functionality of the Task UI on page 127

In this example, the user uses a task UI to enter information in the fields of an opportunity. Siebel CRM displays a radio button group that allows the user to choose to add an activity. Siebel CRM stores this choice as transient data, so you use a transient business component and a task applet. For more information, see "Overview of Transient Data" on page 96.

Creating and Diagramming the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

In this topic, you create and diagram the task UI.

To create and diagram the task UI

- Create and test the task UI that is described in "Example of Developing a Task UI That Assists with Adding an Opportunity" on page 110.

Creating the Transient Business Component

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

This task UI uses a radio button group to provide the user an option to add an activity to the opportunity. It is not necessary to store the reply that the user provides in the Siebel database. Instead, you use a transient business component to store the reply. For more information, see the following topics:

- "Overview of Transient Data" on page 96
- "Radio Button Group" on page 45

To create the transient business component

1. Use the Business Component Wizard to create a transient business component. In the New Business Component dialog box, define properties using values from the following table.

Property	Description
Project	Choose Opportunity Task UI.
Name	Enter TBC for Opportunity.
MultiRecord TBC checkbox	Make sure this property does not contain a check mark.

   For more information, see "Creating a Transient Business Component" on page 98.

2. In the Object Explorer, expand the Business Component tree, and then click Field.
3. In the Fields list, create a new field using values from the following table.

Property	Description
Name	Enter Create Activity.
Picklist	Choose a picklist. This choice only acts as a placeholder. You define the actual picklist in a subsequent step in this procedure.
Text Length	Enter 10.
Type	Choose DTYPE_TEXT.

4. Make sure the Create Activity record is still chosen in the Fields list.
5. In the Object Explorer, expand the Field tree, and then click Pick Map.

6 In the Pick Maps list, create a new pick map using values from the following table.

Property	Value
Field	Create Activity
Picklist Field	Value

Creating the Picklist

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

In this topic, you create the picklist that the Create Activity field references.

To create the picklist

1 In Siebel Tools, choose the File menu, and then the New Object menu item.

2 In the General tab of the New Object Wizards dialog box, choose Pick List, and then click OK.

3 Use the Pick List Wizard to create the pick list:

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

Property	Value
Project	Opportunity Task UI
Business Component	TBC for Opportunity
Field	Create Activity

b In the Pick List Type dialog box, accept the default Static pick list type and then click Next.

c In the Pick List Definition dialog box, accept the following default value and then click Next:

Create New Pick List

d In part two of the Pick List Definition dialog box, do the following work:

- Enter the following text for the name: Yes No Prompt
- Accept the following default: Create New List of Values
- Click Next.

e In the List of Values dialog box, do the following work:

- Enter the following text for the name in the top window: Values for Yes No Prompt
- Enter the following value in the middle window: Yes
- Click Enter.

Siebel Tools displays the following value in the Current values window: Yes

f In the List of Values dialog box, do the following work:

- Enter the following text in the middle window: No
- Click Enter.

Siebel Tools displays the following value in the Current values window: Yes and No

g Click Next.

h In the Pick List Definition dialog box, leave all fields empty, and then click Next.

i In the Finish dialog box, click Finish.

4 In the Object Explorer, click Business Component.

5 In the Business Components list, query the Name property for the following value:

TBC for Opportunity

6 In the Object Explorer, expand the Business Component tree, and then click Field.

7 In the Fields list, query the Name property for the Create Activity field that you defined in Step 3 on page 119.

8 In the Picklist property, remove the value you entered earlier, and then choose the following value from the pop-up applet:

Yes No Prompt

Creating the Task Applet

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

In this topic, you create the task applet that allows the user to add an activity. For more information, see Task Applet on page 40.

To create the task applet

1 Use the Task Form Applet Wizard to create a new task applet:

a In the Task tab of the New Object Wizards dialog box, choose the Task Form Applet icon.

b In the General dialog box, define properties using values from the following table.

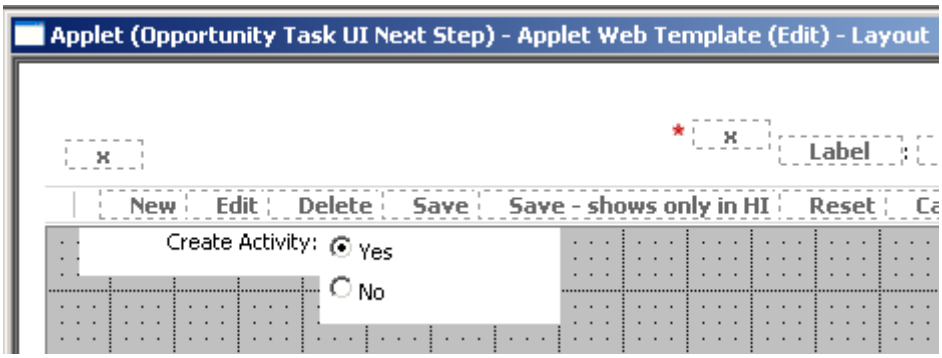
Property	Value
Project	Opportunity Task UI
Applet Name	Opportunity Task UI Next Step
Display Title	Next Step
Task Applet is Associated With	Create New Opportunity

Property	Value
Upgrade Behavior	Admin
Transient Business Component	TBC for Opportunity

c In the Web Layout - Fields dialog box, move only the Create Activity field to the Selected Fields window.

For more information, see "Creating a Task Applet" on page 100.

2 In the Applet Web Template window, reposition the Create Activity control and label until the applet resembles the layout that the following diagram displays:



The No option is not visible when Siebel Tools displays the template. You must resize the Create Activity control to display the No option.

3 Save your work, and then close the Web Layout Editor.

Adding the Task Applet to the View

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

In this topic, you add the task applet to the view.

To add the task applet to the view

1 In the Object Explorer, click View.

2 In the Views list, query the Name property for Opportunity Task UI View.

3 In the Object Explorer, expand the View tree, expand the View Web Template tree, and then click View Web Template Item.

4 In the View Web Template Items list, add a new view web template using values from the following table.

Property	Value
Name	Opportunity Task UI Next Step
Item Identifier	6
Applet	Opportunity Task UI Next Step
Applet Mode	Edit

5 In the View Web Template Items list, choose the item named Task Playbar Applet - Bottom, and then set properties using values from the following table.

Property	Value
Item Identifier	8

Creating the Task View

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

In this topic, you create a new task view that allows the user to add an activity to the opportunity. For more information, see "Creating a Task View" on page 88.

To create the task view

- Use the Task View Wizard to create a new task view:

  a In the New View dialog box, enter values described in the following table.

Property	Value
Project	Opportunity Task UI
View Name	Opportunity Activity Task View
View Title	Create Activity
Business Object	Opportunity
Upgrade Behavior	Admin

  b In the View Web Layout - Select Template dialog box, choose the following template:

    - View Detail (Parent with Pointer)

  c In the Web Layout - Applets dialog box, move the following applets from the Available Applets window to the Selected Applets window:

- Opportunity Task UI Form Applet
- Activity Form Applet

The Opportunity Task UI Form Applet is the custom applet you created in "Creating the Applet" on page 112 that allows the user to enter details about the opportunity. The Activity Form Applet is a predefined applet that allows the user to enter details about the activity.

d. In the Task View - Pick Task dialog box, click No.

e. In the Task View - Select Playbar Applet dialog box, choose the Task Playbar Applet - Bottom applet as the bottom playbar applet.

f. Click Finish, verify that the layout is correct, and then close the Web Layout Editor.

Revising the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

In this topic, you revise the task UI so that it allows the user to add an activity.

To revise the task UI

1. In the Object Explorer, click Task.
2. In the Tasks list, query the Task Name property for Create New Opportunity.

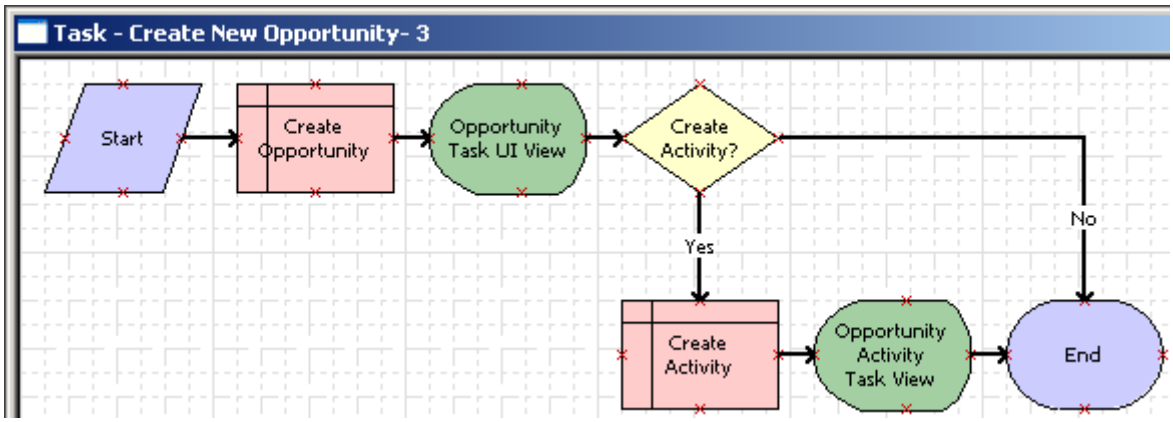
   If Siebel Tools returns more than one record, then choose the record that includes the highest number in the Version property.
3. In the WF/Task Editor Toolbar, click the Revise icon.

   Siebel Tools creates a new record in the Tasks list. This new record possesses the highest number in the Version property. For more information, see "Using the WF/Task Editor Toolbar" on page 60.
4. In the Tasks list, modify the new revision using values from the following table.

Property	Value
Transient BC	TBC for Opportunity

The Transient BC property identifies the transient business component.

5 Open the Task Editor for the new version, and then modify the task UI until it resembles the flow that the following diagram displays:



Siebel Tools displays the label of the Opportunity Activity Task View after you bind the view in a subsequent step in this example. For more information, see "Diagramming a Task UI" on page 88.

6 Choose the Opportunity Task UI View step, and then use the properties window to define values described in the following table.

Property	Value
Forward Button Type	Next

7 Choose the Create Activity step, and then use the properties window to define values described in the following table.

Property	Value
Business Component	Action
Defer Write Record	True
Operation	Insert

8 Choose the Opportunity Activity Task View step, and then use the properties window to define values described in the following table.

Property	Value
Display Name - String Override	Create Activity
Forward Button Type	Submit
Task View	Opportunity Activity Task View

9 Click the No connector, and then use the properties window to define values described in the following table.

Property	Value
Type	Default

Creating the Condition Criteria

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

In this topic, you create the condition criteria.

To create the condition criteria

1 Right-click the Yes connector, and then choose Edit Conditions.

Siebel Tools displays the Compose Condition Criteria dialog box. For more information, see "Creating a Branch Connector" on page 78.

2 In the Compare To picklist, choose Business Component.

You can use the Compare To picklist to indicate whether Siebel CRM must use an object, expression, or process property in the comparison. In this example, Siebel CRM compares the run-time value of a field from the TBC for Opportunity business component, as defined in the Object picklist. Items in the Object picklist are context-sensitive. Siebel CRM updates these items according to the value that you choose in the Compare To picklist.

3 In the Object picklist, choose TBC for Opportunity.

4 In the Operation picklist, accept the following default:

All Must Match (Ignore Case)

You can use the Operation field to create the condition in this example. The value you define in the Values section of the Compose Condition Criteria dialog must match the run-time value of the Create Activity field of the business component.

5 In the Field picklist, choose Create Activity.

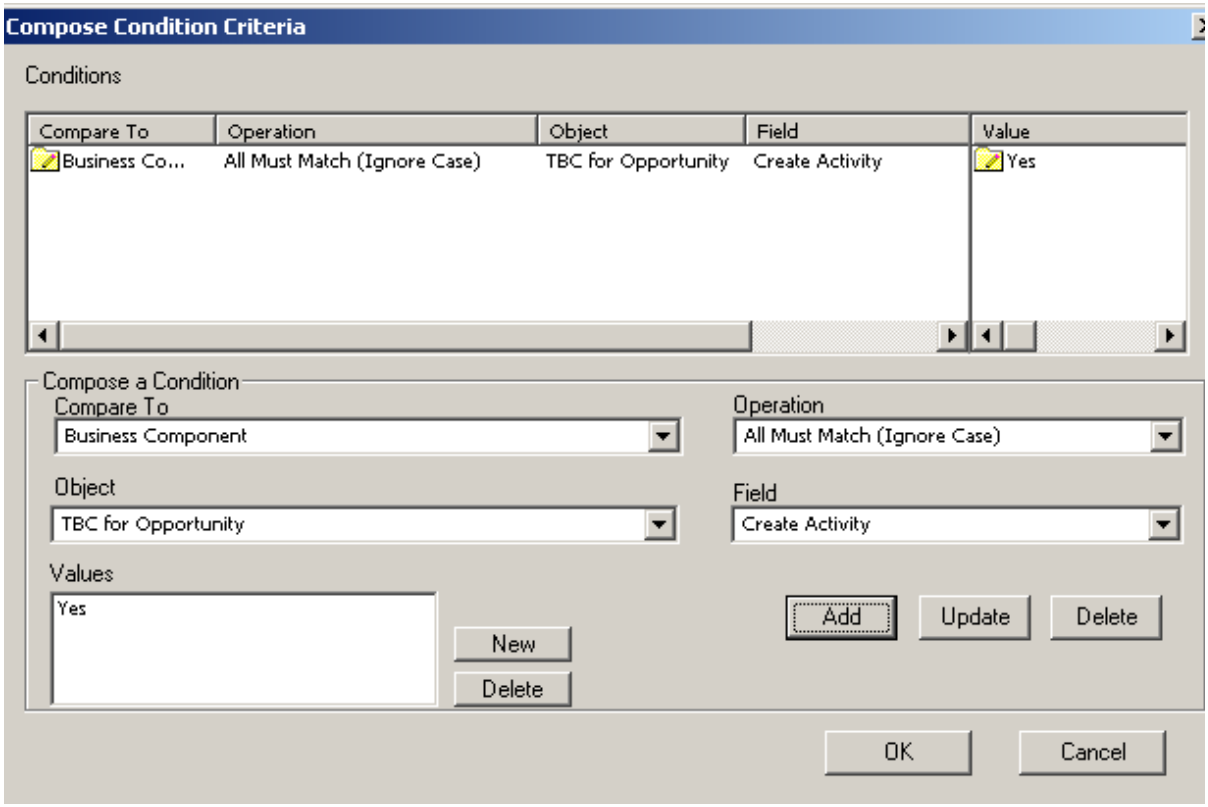
6 In the Values window, click New.

7 In the Pick Value dialog box, choose Yes, and then click OK.

8 In the Compose Condition Criteria dialog box, click Add.

Siebel Tools adds the condition you created in the lower portion of the Compose Condition Criteria dialog box to the Conditions list in the upper portion of the dialog box. At this point, you can add another condition. For this example, you add only a single condition.

The condition you set in the Compose Condition Criteria dialog box must resemble the condition that the following diagram displays:



At run time, if the Create Activity field contains:

- **Yes.** Siebel CRM directs the task flow down the Yes branch.
- **No.** Siebel CRM directs the task flow down the No branch.

9 Click OK.

10 Save your work, and then close the Task Editor.

Verifying the Functionality of the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Adding an Opportunity and an Activity" on page 118.

In this topic, you verify that the task UI delivers the required functionality.

To verify the functionality of the task UI

1. Log in to the Siebel client, click the Opportunities screen tab, and then click the Opportunities List link.
2. Click the Tasks icon.
3. In the task pane, click the Create New Opportunity link, and then verify that the Enter Opportunity Details view does the following:
   - Displays the same controls you examined in "Verifying the Task UI" on page 117
   - Includes the Create Activity radio button group
4. Enter Test Opportunity in the Name field, enter a Description, click Yes, and then click Next.
5. Verify that the Create Activity view displays correctly:
   a. Verify that the top portion of the view includes the same opportunity details that Siebel CRM displayed in the Enter Opportunity Details view.
   b. Verify that the bottom portion of the view displays fields that allow the user to provide details about the activity.
   c. Verify that Siebel CRM enters data in the Opportunity field in the bottom portion of the view that includes the opportunity name that you entered in the earlier view.
6. Do the following:
   a. Enter some text in the Description field of the activity.
   b. Define other activity fields at your discretion.
   c. Click Submit.

   Siebel CRM closes the task UI and then displays the standard Opportunity List View.
7. Verify the following:
   - The opportunity list includes the opportunity you created in Step 4.
   - The opportunity fields include the values you entered in the task UI.
8. In the Opportunity Name field, drill down on the Test Opportunity record.
9. Click the Activities view tab, and then verify that the standard Opportunity Detail - Activities View displays an activity record that includes the activity details you entered in the task UI in Step 6.

Example of Developing a Task UI That Assists with Adding an Account and a Service Request

The example in this topic describes how to develop a task UI that assists the user with adding an account and a service request for that account. To develop this example, perform the following tasks:

1. Creating and Diagramming the Task UI on page 129
2. Creating the Applets on page 131

3 Creating the Task Views on page 133

4 Binding Task Views To Task View Steps on page 135

5 Controlling the Buttons of the Playbar Applet on page 135

6 Creating Chapters for the Task UI on page 136

7 Creating the Task Group on page 137

8 Publishing and Activating a Task UI in Siebel Tools on page 167

9 Verifying the Task UI on page 138

The data manipulation and storage requirements for this example do not involve transient data, so you do not use a transient business component or a task applet. Instead, you use standard business components and reuse standard applets. For more information, see:

- “Overview of Transient Data” on page 96
- “Task Applet” on page 40

Creating and Diagramming the Task UI

This task is a step in “Example of Developing a Task UI That Assists with Adding an Account and a Service Request” on page 128.

In this topic, you create and diagram the task UI.

To create and diagram the task UI

1 Make sure Siebel Tools is prepared to create a task UI.

   For more information, see “Preparing Siebel Tools to Create a Custom Task UI” on page 86.

2 In the Object Explorer, click Project.

3 In the Projects list, create a new project named Account Task UI.

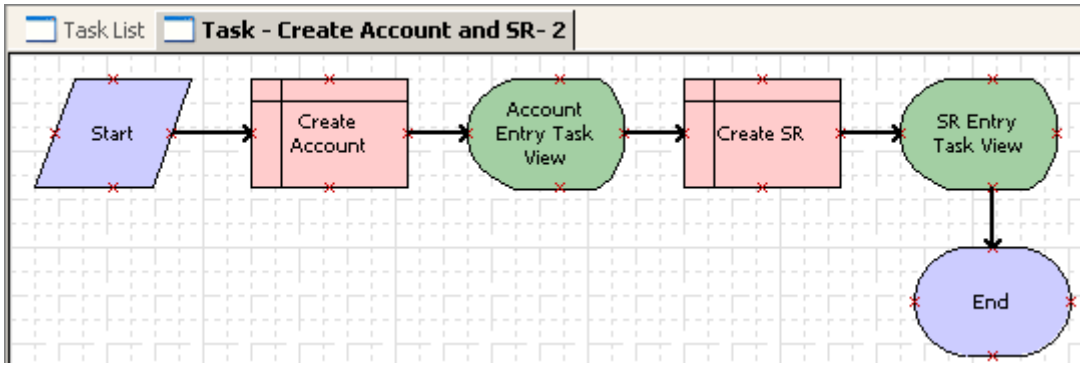
   You can use this project to support this development effort. For more information, see *Using Siebel Tools*.

4 Use the Task Wizard to create a new task UI, using values in the following table.

Property	Description
Project	Choose Account Task UI.
Task Name	Enter Create Account and SR.
Display Name	Enter Create new Account and SR.
Business Object	Choose Account.
Transient Business Component	Leave this field empty.
Subtask	Make sure this property does not include a check mark.

For more information, see "Creating a Custom Task UI" on page 86.

5 Diagram the task UI so that it resembles the flow that the following diagram displays:



Siebel Tools displays labels for the task view steps after you bind the views in a subsequent step in this example. For more information, see "Diagramming a Task UI" on page 88.

6 Click the Create Account step, and then use the Properties window to define properties using values in the following table.

Property	Value
Business Component	Account
Defer Write Record	TRUE
Operation	Insert

For more information, see "About the Defer Write Record Property" on page 71.

7 Click the Create SR step, and then use the Properties window to define properties using values in the following table.

Property	Value
Business Component	Service Request
Defer Write Record	TRUE
Operation	Insert

8 Save your work, and then close the Task Editor.

Creating the Applets

This task is a step in "Example of Developing a Task UI That Assists with Adding an Account and a Service Request" on page 128.

In this topic, you reuse standard applets. For more information, see "Reusing a Standard Applet" on page 206.

To create the applets

1 Copy a standard applet:

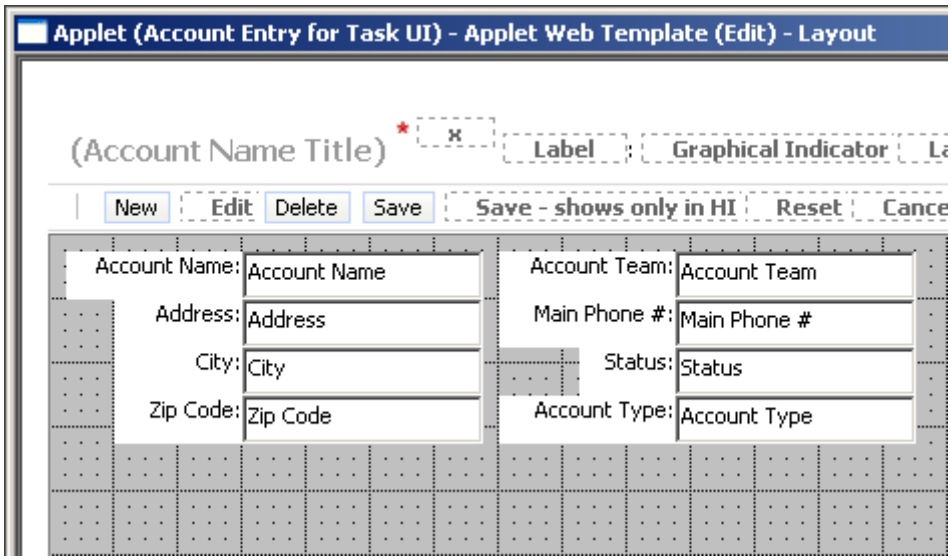
   a Copy the Account Entry Applet.

   b To name the new applet, use the following text:

      - Account Entry for Task UI

   c In the Project property, enter Account Task UI.

d In the Applet Web Template window, delete and reposition controls and their labels so the applet resembles the layout that the following diagram displays:



e Save your work and then close the Applet Web Template window.

For more information, see "Reusing a Standard Applet" on page 206.

2 Copy a standard applet:

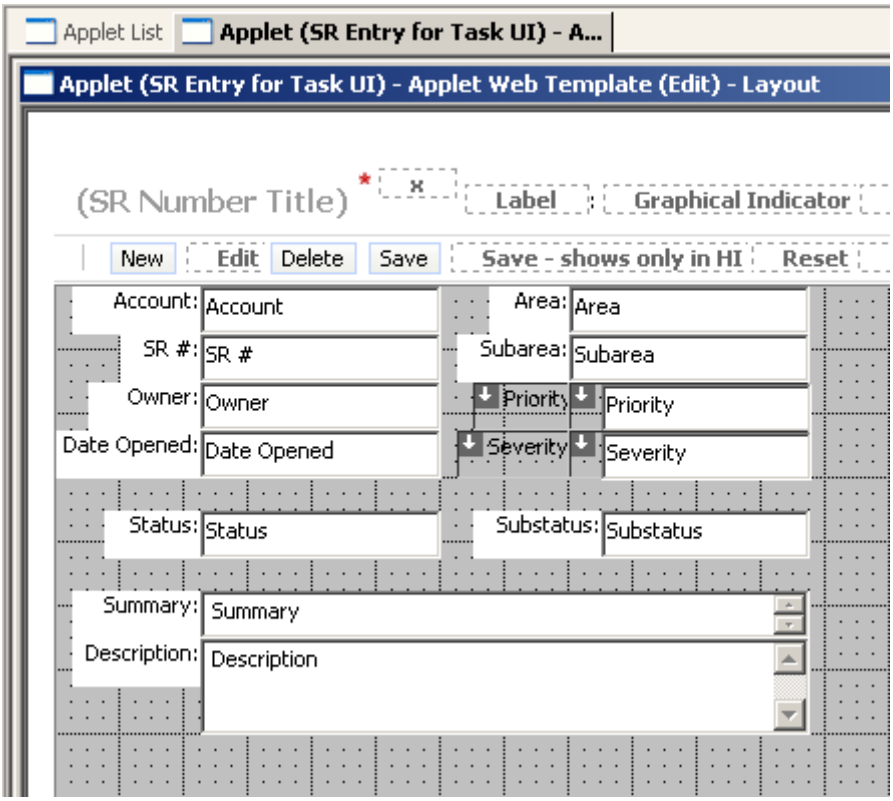
a Copy the Service Request Detail Applet.

b Use the following text to name the new applet:

- SR Entry for Task UI

c In the Project property, enter Account Task UI.

d In the Applet Web Template window, delete and reposition controls and their labels so the applet resembles the layout that the following diagram displays:



Reuse of this applet includes the fields that the insert operation requires. You can remove the following items to simplify layout:

- Remove unused controls and labels from the grid.
- Remove the Verify button and the Verify Best Time button from the area immediately above the grid.

e Save your work, and then close the Applet Web Template window.

For more information, see "Reusing a Standard Applet" on page 206.

Creating the Task Views

This task is a step in "Example of Developing a Task UI That Assists with Adding an Account and a Service Request" on page 128.

In this topic, you create task views that display the applets you created in "Creating the Applets" on page 131.

To create the task views

1 Use the Task View Wizard to create the task view that allows the user to enter account data:

a In the New View dialog box, enter values described in the following table.

Property	Value
Project	Account Task UI
View Name	Account Entry Task View
View Title	Enter Account Details
Business Object	Account
Upgrade Behavior	Admin

b In the View Web Layout - Select Template dialog box, choose the View Basic template.

c In the Web Layout - Applets dialog box, chose the following applet:

- Account Entry for Task UI

The Account Entry for Task UI applet is one of the custom applets you created in "Creating the Applets" on page 131.

d In the Task View - Pick Task dialog box, click No.

e In the Task View - Select Playbar Applet dialog box, choose the following applet as the bottom playbar applet:

- Task Playbar Applet - Bottom

f Click Finish, verify that the layout is correct, and then close the Web Layout Editor.

For more information, see "Creating a Task View" on page 88.

2 Use the Task View Wizard to create the task view that allows the user to enter data about the service request:

a In the New View dialog box, enter values described in the following table.

Property	Value
Project	Account Task UI
View Name	SR Entry Task View
View Title	Enter SR Details
Business Object	Account
Upgrade Behavior	Admin

b In the View Web Layout - Select Template dialog box, choose the View Basic template.

c In the Web Layout - Applets dialog box, chose the following applet:

- SR Entry for Task UI

  The SR Entry for Task UI applet is one of the custom applets you created in "Creating the Applets" on page 131.

d. In the Task View - Pick Task dialog box, click No.

e. In the Task View - Select Playbar Applet dialog box, choose the following applet as the bottom playbar applet:

- Task Playbar Applet - Bottom

f. Click Finish, verify that the layout is correct, and then close the Web Layout Editor.

For more information, see "Creating a Task View" on page 88.

Binding Task Views To Task View Steps

This task is a step in "Example of Developing a Task UI That Assists with Adding an Account and a Service Request" on page 128.

You must configure Siebel CRM to bind a task view to each task view step in the Create New Account task UI.

To bind task views to task view steps

1. Configure Siebel CRM to bind the task view step that resides immediately downstream of the Create Account Siebel operation step to the Account Entry Task View.

   For more information, see "Binding a Task View to a Task View Step" on page 91.

2. Configure Siebel CRM to bind the task view step that resides immediately downstream of the Create SR Siebel operation step to the SR Entry Task View.

Controlling the Buttons of the Playbar Applet

This task is a step in "Example of Developing a Task UI That Assists with Adding an Account and a Service Request" on page 128.

This topic describes how to modify properties on the task view step to control the buttons that are active in the playbar applet. You also set the display name for the task view.

To control the buttons of the playbar applet

1 In the Task Editor, click the Account Entry Task View step, and then use the Properties window to define properties described in the following table.

Property	Value
Disable Previous	TRUE
Display Name - String Override	Enter Account Details
Forward Button Type	Next

2 Click the SR Entry Task View step, and then use the Properties window to define properties described in the following table.

Property	Value
Disable Previous	FALSE
Display Name - String Override	Enter SR Details
Forward Button Type	Submit

Creating Chapters for the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Adding an Account and a Service Request" on page 128.

In this topic, you create two new chapters for the Create New Account task UI. The task chapter is an optional UI element that assists user navigation. For more information, see "Task Chapter" on page 38.

To create chapters for the task UI

■ Create the task chapter:

a Create two new chapters using values from the following table.

Name	Display Name - String Override	Sequence
Chapter 1	Enter New Account	10
Chapter 2	Enter New SR	20

b Assign the following steps to Chapter 1:

- ❑ Start
- ❑ Create Account
- ❑ Account Entry

c Assign the following steps to Chapter 2:

- Create SR
- SR Entry Task View
- End

d Save your modifications and close the Task Editor.

For more information, see "Creating a Task Chapter" on page 201.

Creating the Task Group

This task is a step in "Example of Developing a Task UI That Assists with Adding an Account and a Service Request" on page 128.

In this topic, you create a task group for the Create New Account task UI. The task group is a required UI element that specifies the group to display in the task pane when Siebel CRM displays the current view. For more information, see "Task Group" on page 36.

To create the task group

1 Create the task groups:

a Create a new task group using values from the following table.

Property	Value
Name	Account Tasks
Project	Account Task UI
Display Name - String Override	Account Tasks

b Create a new task group item using values from the following table.

Property	Value
Action Invoked	Create Account and SR
Type	Task
Sequence	1

For more information, see "Creating a Task Group" on page 92.

2 Define the task group to display across views.

Choose Account Tasks from the Task Groups pop-up applet when you define the Task Group property in the View Task Groups list, and then define the property described in the following table.

Property	Value
Sequence	2

For more information, see "Adding a Task Group to Multiple Views" on page 94.

Verifying the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Adding an Account and a Service Request" on page 128.

In this topic, you verify that the task UI delivers the required functionality. For more information, see "Verifying Functionality of a Task UI" on page 154.

To verify the task UI

1 Log in to the Siebel client.

2 Click the Tasks icon, and then verify that the task pane displays a task group named Account Tasks.

To verify that Siebel CRM displays the task UI globally, click several different screen tabs, such as Home, Accounts, Contacts and Sales Orders. The Account Tasks task group must remain visible regardless of the standard view that Siebel CRM displays.

3 In the task pane, click the following link:

Create Account and SR

4 Verify that the task pane displays the following chapters:

- Enter New Account
- Enter New SR

The type of task pane that Siebel CRM displays varies depending on the context that the user uses to view the pane. At this point in this procedure, Siebel CRM displays the current task pane. For more information, see "About the Task Pane" on page 31.

5 Verify that Siebel CRM displays the Enter Account Details view and that it correctly arranges the controls.

The layout must be similar to the layout displayed in Figure 1 on page 12.

If the layout requires adjustment, then you can rapid prototype the layout of the applet. For more information, see "Rapid Prototyping the Layout of an Applet" on page 139.

6 Verify that the following buttons are active:

- Pause
- Next
- Cancel

7 Verify that Siebel CRM disables the Previous button.

8 Enter a value in the Account Name field and then click Next.

9 Verify that Siebel CRM does the following:

- Displays the Enter SR Details view
- Automatically enters data in the Account field that includes the account name you entered in Step 8

10 Make sure the required fields include values.

11 Enter some text in the Summary field.

12 Click Submit.

13 Navigate to the My Accounts list, and then query the Account Name field for the name you entered in Step 8.

14 Verify that Siebel CRM displays the new account.

15 Drilldown on the Account Name field.

16 Click the Service Requests view tab.

17 Verify that Siebel CRM displays one new SR for the account, and that this SR includes values you entered in Step 10.

Rapid Prototyping the Layout of an Applet

This task is a step in "Example of Developing a Task UI That Assists with Adding an Account and a Service Request" on page 128.

You can do rapid prototyping on the layout of the applet. It is not necessary to recompile the entire task and the objects that the task references.

To rapid prototype layout of an applet

1 If the Siebel client is open, then close it.

2 In Siebel Tools, in the Object Explorer, expand the Applet tree, and then click Applet Web Template.

3 In the Applet Web Templates list, right-click the applet web template you must modify, and then choose Edit Web Layout.

4 Modify the layout, save your modifications, and then close Applet Web Template.

5 In the Applets list, right-click the record for the applet you modified in Step 4, and then choose Compile Selected Objects.

6 Log in to the Siebel client, navigate to the task UI you are modifying, and then examine your modifications.

Example of Developing a Task UI That Assists with Creating Multiple Opportunities

The example in this topic describes how to develop a task UI that assists the user with adding multiple opportunities. To develop this example, perform the following tasks:

Creating the Transient Business Component

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

In this topic, you create the transient business component for the task UI. For more information, see "Overview of Transient Data" on page 96.

To create the transient business component

1 Make sure Siebel Tools is prepared to create a task UI.

For more information, see "Preparing Siebel Tools to Create a Custom Task UI" on page 86.

2 In the Object Explorer, click Project.

3 In the Projects list, create a new project named Multiple Opportunity Task UI.

You can use this project to support this development effort. For more information, see *Using Siebel Tools*.

4 Choose the File menu, and then the New Object menu item.

5 In the New Object Wizards dialog box, choose the Task tab, click the Transient BusComp icon, and then click OK.

6 In the New Business Component dialog box, define the new business component using values from the following table.

Property	Description
Project	Choose Multiple Opportunity Task UI.
Name	Enter MultiRecordTBC.
MultiRecordTBC	Make sure this property includes a check mark.

7 Click Finish.

8 In the Object Explorer, expand the Business Component tree, and then click Single Value Field.

9 In the Single Value Fields list, create a new field using values from the following table.

Property	Value
Name	OptyName
Type	DTYPE_TEXT

Creating the Business Service

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

In this topic, you create the business service that the task UI calls.

To create the business service

1 In the Object Explorer, click Business Services.

2 In the Business Services list, create a new record using values from the following table.

Property	Value
Name	MultiTBC_Insert
Project	Multiple Opportunity Task UI
Class	CSSService
Display Name - String Override	MultiTBC_Insert

3 In the Business Services list, right-click the MultiTBC_Insert business service, and then choose Edit Server Scripts.

4 In the Scripting Language dialog box, choose eScript, and then click OK.

Siebel Tools opens the script editing window and chooses the Service_PreInvokeMethod function.

5 Make sure the cursor is positioned in the script editing window.

6 Choose the Edit menu, and then the Select All menu item.

7 Choose the Edit menu, and then the Cut menu item.

8 Paste the following script in the script editing window:

```
function Service_PreInvokeMethod (MethodName, Inputs, Outputs)
{
 if (MethodName == "InsertRecordToTBC")
 {
     var currentBC;
        var currentTBC;
        var currentBO;
        var isSuccess;
        var iCounter;
        var indata;
        var isRecord;

        currentBO = TheApplication().GetBusObject ("Opportunity");
        currentTBC = currentBO.GetBusComp ("TBUI81MultiRecordTBC");
        currentBC = currentBO.GetBusComp ("Opportunity");

        //Inserts data in the TBC
        iCounter = 3;
        do
        {
                 indata="OptyMTBC"+iCounter;
                currentTBC.NewRecord(0); // creating a new record
              currentTBC.SetFieldValue("OptyName", indata); //setting name field
```

```
                currentTBC.WriteRecord(); // writing record in the multirecord TBC
                  iCounter--;
            } while(iCounter);


        //Get the data from TBC and insert in BC
            currentTBC.ActivateField ("OptyName");
            currentTBC.ClearToQuery();
            currentTBC.SetSearchSpec ("OptyName", "*");
            currentTBC.ExecuteQuery(ForwardBackward);
            isRecord = currentTBC.FirstRecord();
            iCounter = 0;
            if(isRecord)
            {
                    do
                    {
                            indata = currentTBC.GetFieldValue("OptyName");
                            currentBC.NewRecord(0); //This creates the new record
                            currentBC.ActivateField("Name");
                            currentBC.SetFieldValue("Name", indata);
    // This is setting the field value for opportunity name
                         currentBC.WriteRecord(); // writing the record to database.
                            iCounter++;
                    } while(currentTBC.NextRecord());
            }
      }
      return (CancelOperation);
    }
```

9 Close the script editing window, and then click Yes in the Confirm dialog box.

10 In the Object Explorer, expand the Business Service tree, and then click Business Service Method.

11 In the Business Service Methods list, create a new business service using values from the following table.

Property	Value
Name	InsertRecordToTBC
Display Name - String Override	MultiTBCInsertBSM

Creating and Diagramming the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

In this topic, you create and diagram the task UI.

To create and diagram the task UI

1 Make sure Siebel Tools is prepared to create a task UI.

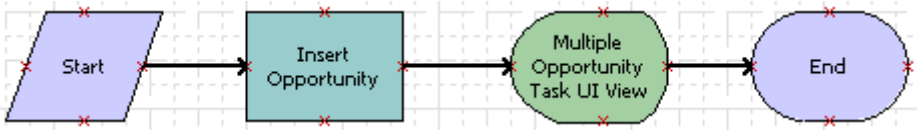
For more information, see "Preparing Siebel Tools to Create a Custom Task UI" on page 86.

2 To create a new task UI, use the Task Wizard using values from the following table.

Property	Description
Project	Choose Multiple Opportunity Task UI.
Task Name	Enter Create Multiple Opportunities.
Display Name	Enter Create Multiple Opportunities.
Business Object	Choose Opportunity.
Transient Business Component	Choose MultiRecordTBC.
Subtask	Make sure this property does not include a check mark.

For more information, see "Creating a Custom Task UI" on page 86.

3 Diagram the task UI until it resembles the flow that the following diagram displays.



Siebel Tools displays the label for the task view step after you bind the view in a subsequent step in this example. For more information, see "Diagramming a Task UI" on page 88.

4 Click the Insert Opportunity step.

5 To define properties for the Insert Opportunity step, use the Properties window using values in the following table.

Property	Value
Business Service Name	81MultiTBC_Insert
Business Service Method	InsertRecordToTBC

6 Save your work, and then close the Task Editor.

Creating the Task Applet

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

In this topic, you create a new task applet that the user uses to enter opportunity information in the task UI. For more information, see "Creating a Task Applet" on page 100.

To create the task applet

1 In Siebel Tools, choose the File menu, and then the New Object menu item.

2 In the New Object Wizards dialog box, click the Task tab, choose the Task List Applet icon, and then click OK.

You create a list applet because the business need for this example requires that the user simultaneously create and view all opportunities that the user adds for the task instance. As an alternative, you can create a form applet where the user enters one opportunity at a time, and then uses buttons on the playbar to loop through the same form repeatedly until the user enters all opportunities.

3 In the General dialog box, define fields using values from the following table, and then click Next.

Property	Value
Project	Multiple Opportunity Task UI
Name	Multiple Opportunity Task Applet
Display Title	Multiple Opportunity
Task Applet is Associated With	Create Multiple Opportunities
Upgrade Behavior	Admin
Business Component	MultiRecordTBC

4 In the Web Layout - Fields dialog box, move the OptyName field from the Available Fields window to the Selected Fields window, and then click Next.

5 In the Finish dialog box, review your settings, and then Click Finish.

Siebel Tools creates the new applet and then opens the Web Layout Editor.

6 Save your work and then close the Applet Web Template window.

Creating the Task View

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

In this topic, you create a task view that displays the applet that allows the user to enter multiple opportunities.

To create the task view

1 Create a task view.

For more information, see "Creating a Task View" on page 88.

2 In the New View dialog box, enter values described in the following table.

Property	Value
Project	Multiple Opportunity Task UI
View Name	Multiple Opportunity Task UI View
View Title	Multiple Opportunities
Business Object	Opportunity
Upgrade Behavior	Admin

3 In the View Web Layout - Select Template dialog box, choose the following template:

View Detail (Parent with Pointer)

4 In the Web Layout - Applets dialog box, click Next.

Do not move any applets to the Selected Applets window.

5 In the Task View - Select Task dialog box, choose Create Multiple Opportunities, and then click Next.

6 In the Task View - Task Applets dialog box, move the following applet from the Available Applets window to the Selected Applets window:

Multiple Opportunity Task Applet

The Multiple Opportunity Task Applet is the custom applet you created in "Creating the Task Applet" on page 145.

7 Click Next.

8 In the Task View - Select Playbar Applet dialog box, choose the Task Playbar Applet - Bottom applet as the bottom playbar applet, and then click Next.

9 Click Finish, verify that the layout is correct, and then close the Web Layout Editor.

Binding the Task View to the Task View Step

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

In this topic, you bind a task view to the task view step.

To bind the task view to the task view step

- Bind the task view step to the Multiple Opportunity Task UI View.

  For more information, see "Binding a Task View to a Task View Step" on page 91.

Controlling the Buttons of the Playbar Applet

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

You modify properties on the task view step to control the buttons that are active in the playbar applet. You also set the display name for the task view.

To control the buttons of the playbar applet

1 In the Task Editor, click the Opportunity Task UI View step, and then use the Properties window to define properties described in the following table.

Property	Value
Disable Previous	TRUE
Display Name - String Override	Enter Multiple Opportunities
Forward Button Type	Submit

2 Save your modifications and then close the Task Editor.

Creating the Task Group

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

In this topic, you create a task group for the Create Multiple Opportunities task. For this example, you define the task group so that Siebel CRM displays it only when it also displays the Opportunity List View. For more information, see "Task Group" on page 36.

To create the task group

1 Create the task group:

a Create the new task group using values from the following table.

Property	Value
Name	Multiple Opportunities Task Group
Project	Multiple Opportunity Task UI
Display Name - String Override	Multiple Opportunities

b Create the new task group item using values from the following table.

Property	Value
Action Invoked	Create Multiple Opportunities
Type	Task
Sequence	1

For more information, see "Creating a Task Group" on page 92.

2 Define the task group to display in a specific view:

- In the Views list, query the Name property for Opportunity List View.
- Choose Multiple Opportunities Task Group from the Task Groups pop-up applet when you define the Task Group property in the View Task Groups list, and then define the property described in the following table.

Property	Value
Sequence	2

For more information, see "Adding a Task Group to a View" on page 93.

Verifying the Task UI

This task is a step in "Example of Developing a Task UI That Assists with Creating Multiple Opportunities" on page 140.

In this topic, you verify that the task UI delivers the required functionality. For more information, see "Verifying Functionality of a Task UI" on page 154.

To verify the task UI

1. Log in to the Siebel client, click the Tasks icon, and then verify that the task pane does not display the Multiple Opportunities task group.

   You defined this task UI to display only when the Opportunity List View is the current view.

2. Click the Opportunities screen tab, and then click the Opportunities List link.
3. Verify that the task pane now displays the Multiple Opportunities task group.
4. In the task pane, click the Create Multiple Opportunities link, and then verify that Siebel CRM displays the correct view and arranges the controls correctly.

   The layout must be similar to the layout you defined in "Creating the Task Applet" on page 145.

   If the layout requires adjustment, then you can rapid prototype the layout of the applet. For more information, see "Rapid Prototyping the Layout of an Applet" on page 139.

5. Verify that the following buttons are active:
   - Pause
   - Submit
   - Cancel
6. Verify that the Previous button is disabled.
7. Enter multiple opportunities and then click Submit.

   Siebel CRM closes the task UI and then displays the standard Opportunity List View.

8. Verify that the opportunity list includes the opportunities you created in Step 6.

9 Testing, Troubleshooting, Deploying, and Administering a Task UI

This chapter describes how to test, troubleshoot, deploy, and administer a task UI. It includes the following topics:

Process of Testing a Task UI

This process is a step in "Roadmap for Developing a Task UI" on page 85. To test a task UI, perform the following tasks:

1. Validating a Task UI on page 151
2. Optional. Using the Task Debugger on page 152
3. Verifying Functionality of a Task UI on page 154
4. Optional. Troubleshooting a Task UI on page 159

You must test a task UI before you deploy it. Testing a task UI verifies that the task UI you release to the production environment runs properly and does not cause conflicts with another existing task UI.

This topic also describes how to modify data that Siebel CRM collects during testing. For more information, see "Modifying How Siebel CRM Logs Data During Testing" on page 155.

Validating a Task UI

This task is a step in "Process of Testing a Task UI" on page 151.

The Validate tool in Siebel Tools is an error correction tool that you can use before you deploy a task UI. Validation enforces the semantic consistency of a task. For example, you can use validation to make sure an error process does not include an error process.

To validate a task UI

1. Locate the task UI you must validate.

   For more information, see "Locating a Task UI in the Tasks List" on page 60.

2. In the Tasks list, right-click the task UI, and then choose Validate.

3 In the Validate dialog box, click Start.

Siebel Tools displays the following message in the bottom left corner of the Validate dialog box:

```
Starting validation
```

If your task contains an error, then Siebel Tools displays a warning that describes the error.

4 If Siebel Tools reports a problem after it finishes the validation, then use the information that it displays in the Errors window to help correct the error.

Validate the task UI until you resolve all errors. If the task contains no errors, then Siebel Tools displays the following message in the bottom left corner of the Validate dialog box:

```
Total tests failed: 0
```

5 Click Cancel to close the Validate dialog box and end the validation.

Errors That the Validate Tool Detects

This topic describes problems you might encounter when you use the Validate tool. To resolve the problem, look for it in the Symptom or Error Message column in Table 14.

Table 14. Errors That the Validate Tool Can Detect

Symptom or Error Message	Solution
Connector is not attached correctly.	Make sure you correctly attach all connectors.
Conditional branch is not defined for a Decision point.	Make sure you define at least one conditional branch that emanates from each decision point in the task UI.
Business service or business service method is not defined for a business service step.	Make sure you define the Business Service Name and Business Service Method properties for each business service step in the task UI.
Business component is missing from a Siebel operation step.	Make sure you define the Business Component property for each Siebel operation step that the task UI contains.
Name of the subtask is not defined for a subtask step.	Make sure you define the Subtask Name property for each subtask step that the task UI contains.

Using the Task Debugger

This task is a step in "Process of Testing a Task UI" on page 151.

The *Task Debugger* is a tool that you can run to test a task UI. You can use the Debug Mode in the Siebel client when you use the Task Debugger to do the same work that you expect the user to perform. The debugger allows you to read a watch window that displays values of various properties that are involved with the task UI. The debugger displays and modifies these values while you use the task UI.

Making the Debug Mode Available

You can step through the task UI in debug mode. *Debug Mode* is a restricted menu item that Siebel CRM displays on the Tools menu in the Siebel client. If the EnableRestrictedMenu parameter is set to:

- **TRUE.** Siebel CRM displays the Debug Mode item.
- **FALSE.** Siebel CRM does not display the Debug Mode menu item.

The procedure you perform depends on the type of Siebel client you use.

To make the debug mode available when using the Siebel Mobile Web Client

1. Use a text editor to open the configuration file (.cfg) for the Siebel application.

   For example, to open the configuration file for Siebel Call Center, navigate to the following directory, and then open the uagent.cfg file:

   ```
   D:\Siebel\81\21031\MWC\BIN\ENU
   ```

2. In the InfraUIFramework section, set the EnableRestrictedMenu parameter to TRUE.

   For example:

   ```
   EnableRestrictedMenu = TRUE
   ```

   If EnableRestrictedMenu is not in the InfraUIFramework section, then add it.

3. Save your modifications, and then close the file.
4. Run the Task Debugger.

   For more information, see “Running the Task Debugger” on page 154.

   If the EnableRestrictedMenu parameter is set to something other than True or False, or if it is not defined in the InfraUIFramework section, then Siebel CRM does not display the Debug Mode menu item unless you possess access to the Admin - Restricted Menu Items view.

To make the debug mode available when using the Siebel Web Client

1. In the Siebel client, navigate to the Administration - Server Configuration screen, and then the Servers view.
2. In the Siebel Servers list, choose a Siebel Server.
3. Click the Components view tab.
4. In the Components list, query the Component field for the application object manager.

   For example, query the Component field for the following text:

   Call Center Object Manager (ENU)

5. Click the Parameters view tab that Siebel CRM displays below the Components list.
6. In the Component Parameters list, query the Parameter field for EnableRestrictedMenu.
7. Set the value to True to enable debug mode.

8 Run the Task Debugger.

For more information, see "Running the Task Debugger" on page 154.

Running the Task Debugger

You can choose the Tools menu and then the Debug Mode menu item in the Siebel client to toggle the Debug Mode on or off. You can toggle this mode before, during, or after running a task UI.

To run the task debugger

1 Log in to the Siebel client.

2 Choose the Tools menu, and then the Debug Mode menu item.

3 To open the task pane, click the Open Task Pane icon.

Siebel CRM displays the Open Task Pane icon in the same row of icons where it displays the Site Map icon.

4 From the task pane, drill down on the task UI that you must debug.

Siebel CRM opens the task UI and then runs the start step of the task UI. It displays the Task Properties watch window for this task UI, including the properties for the step that it most recently run. It displays a line titled Last Action under the name of the task UI you are debugging. This line indicates the last navigation action that it performed. For example, Next.

5 Click Continue in the watch window.

Siebel CRM runs the next step in the task UI. For example, if the next step is a task view step, then it displays a task view.

6 To continue to run and debug the task UI, click one of the playbar buttons in the task view.

7 Repeat Step 5 and Step 6 until you reach the end step.

Debugging ends when you reach the end step.

8 Optional. To end a debugging session before you reach the end step, you can do one of the following:

- Click Stop in the watch window.
- Click Cancel in a task view.

Verifying Functionality of a Task UI

This task is a step in "Process of Testing a Task UI" on page 151.

To finish testing for a task UI, you can verify that the task includes the functionality that is required to meet your business requirements. You run the task in the Siebel client, and then verify that the task functionality matches the functionality that you described during the planning phase. To view an example, see "Verifying the Task UI" on page 117.

Modifying How Siebel CRM Logs Data During Testing

This topic describes how to modify the way that Siebel CRM logs data during testing. It includes the following topics:

Using Logging on the Siebel Server

Event logging provides a convenient, nonintrusive way to trace components that require further examination. Similar to Siebel Workflow, Siebel Task UI provides for tracing through event logging. You set trace levels on a server component to turn on logging on the Siebel Server.

To use logging on the Siebel Server

1. In the Siebel client, navigate to the Administration - Server Configuration screen, Servers, Components, and then the Events view.
2. In the Components applet, choose the component you must trace.

   You can choose the following components:

   - Call Center Object Manager
   - Custom Application Object Manager
3. To view the configurable event types for the component that you chose in Step 2, click the Events tab.

   Siebel CRM sets the log level to 1, by default.
4. Modify the value of the log level to 3, 4, or 5.

   For more information, see "About Event Logging" on page 232.

   You can increase the tracing level for the following events to increase the number of entries that Siebel CRM logs:

   - Task UI Object Manager Log
   - Task UI Conflict Log
   - Task UI Union Sql Log

If you increase the Component Event Configuration Log Level value, then Siebel CRM creates more tracing information. For more information, see *Siebel System Monitoring and Diagnostics Guide*.

Setting Log Levels on the Siebel Client

You can set the log level of the Siebel client.

To set log levels on the Siebel client

1 Open a command prompt.

You can open the prompt from a directory on the Siebel Server.

2 Set the SIEBEL_LOG_EVENTS environment variable.

For example, set the logging level for the following log events:

- Set the TskNav log event to 3
- Set the TskExec log event to 3
- Set the StpExec log event to 4
- Set the TskPresInfo log event to 4

You run the following command for this example from a command prompt:

```
set SIEBEL_LOG_EVENTS = TskNav =3, TskExec=3, StpExec=4, TskPresInfo=4
```

Example of a Log File

This following file is an example of a log file that displays information about the TskExec, TskNav, and TskPresInfo log events:

```
TskExecTskState30000000243fc1350:02006-02-22 23:49:49Task state transition changes : Action invoked:
'Navigate', Current State: 'Navigate', Next State: 'Navigate'.
TskNavOper30000000243fc1350:02006-02-22 23:49:49Task engine requested to navigate to next view.
TskNavOper30000000243fc1350:02006-02-22 23:49:49Task engine requested to navigate to next step: 'Task View
1'.
TskNavPathChange30000000243fc1350:02006-02-22 23:49:49Pushing frame to stack : '1*05*Start0*1*00*'.
TskNavPathChange30000000243fc1350:02006-02-22 23:49:49Pushing frame to stack : '1*011*Task View
03011*8#Bookmark4#8#viewName21#EnvironmentDetailView6#bAdmin1#06#viewId0#14#bIgnoreContext1#03#18#frameBo
okmarkArray18#frameBookmarkArray1#4#size1#33#1#21#210#6#rowIds0#16#extraInformation0#14#ToggleSequence2#-
111#columnNames0#11#InQueryMode1#013#SavedShowMode4#Edit9#frameName17#EnvironmentDetail16#RuntimeClassNam
e19#CSSSWEFrameBookmark8#ShowMode4#Edit8#UniqueID1#30#1#11#110#6#rowIds0#16#extraInformation0#14#ToggleSe
quence2#-111#columnNames0#11#InQueryMode1#013#SavedShowMode4#Base9#frameName28#Task Playbar Applet -
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (0) : TYPE =
WfTaskViewInfo
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (0) : ViewName =
RequiredFieldView
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (0) : child count =
0TskPresInfoViewInfo0000000243fc1350:02006-02-22 23:49:49Task presentation view info (0) : child count = 0
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (1) : TYPE =
WfTaskPlaybarInfo
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (1) : Submit = HIDDEN
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (1) : Next = ENABLED
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (1) : Prev = ENABLED
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (1) : Pause = ENABLED
TskPresInfoViewInfo30000000243fc1350:02006-02-22 23:49:49Task presentation view info (1) : Cancel = ENABLED
```

Collecting Timestamp Metrics of a Task UI

You can configure Siebel CRM to collect timestamp metrics after you deploy the task UI. For more information, see “About Task Metrics” on page 234.

To collect timestamp metrics of a task UI

1. Deploy the task UI.

   For more information, see "Deploying and Migrating a Task UI" on page 163.

2. Open the Siebel client.
3. Navigate to the Administration - Business Process screen, and then the Task Deployment view.
4. Set the Analytics Level field to one of the following values:
   - Timestamp
   - All

   These values allow Siebel CRM to collect timestamp metrics at run time.

Collecting Property Metrics of a Task UI

You can configure Siebel CRM to collect property metrics of a task UI after you deploy it. For more information, see "About Task Metrics" on page 234.

To collect property metrics of a task UI

1. Locate the task UI where you must collect property metrics.

   For more information, see "Locating a Task UI in the Tasks List" on page 60.

2. In the Object Explorer, expand the Task tree, and then click Task Metric.
3. In the Task Metrics list, create a new task metric, using values from the following table.

Property	Description
Metric Name	Choose the required metric from the list of predefined metric names.
Property Name	Choose a task property from the list of properties that are defined for the task UI. Siebel Tools sets the run-time value of the metric to the run-time value of the task property that Siebel CRM references. You modify the value only for customizable metrics, and not for system metrics, whose values are set by Siebel Task UI.
Inactive	Choose FALSE. The option in the Inactive column does not include a check mark, by default. It allows you to turn off a metric and turn it on later.

4. Deploy the task UI.

   For more information, see "Deploying and Migrating a Task UI" on page 163.

5. Open the Siebel client.
6. Navigate to the Administration - Business Process screen, and then the Task Deployment view.

7 For testing purposes, set the Analytics Level field to All.

This step turns on property metrics so Siebel CRM collects them at run time.

8 Run the task UI at least one time, and then query the tables where Siebel CRM stores the data.

For example, enter the following query:

```
select s.row_id, s.flow_name, s.PARENT_FLOW_NAME, s.ROOT_FLOW_NAME,
s.FLOW_INST_ID_VAL, s.flow_type_cd, s.created from siebel.s_wfa_anly_info as s
```

9 Optional. Use DML to modify the stored data.

For example, to delete a record, enter the following query:

```
delete from siebel.s_wfa_anly_info
```

Disabling Task Transactions

In some testing situations it might be beneficial to temporarily disable transactions for a task UI. You can disable task transactions at the business component level and at the task UI level.

For more information, see "About Task Transaction" on page 225.

To disable transactions at the task level

1 Locate the task UI that you must modify.

For more information, see "Locating a Task UI in the Tasks List" on page 60.

2 Set the Transactional property to FALSE.

CAUTION: Disabling task transaction can result in unpredictable behavior. You must thoroughly test disabling task transaction before you use it in a production environment.

If you set the Transactional property to FALSE, then task transaction is off. Siebel CRM immediately saves to the base tables any updates that occur on business components that the task UI references. For more information, see "Using the Defer Write Record Property When You Disable Task Transaction" on page 159.

To disable transaction at the business component level

1 If necessary, display the Business Component User Prop object.

For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.

2 In Siebel Tools, in the Object Explorer, click Business Component.

3 In the Business Components list, query the Name property for the business component that is involved with the task transaction.

4 In the Object Explorer, expand the Business Component tree, and then click Business Component User Prop.

5 In the Business Component User Properties list, query the Name property for Immediate Commit In Task.

You can do the following to view the predefined business components that include an Immediate Commit In Task user property:

- In the Object Explorer, click the Flat tab.
- In the Object Explorer, click Business Component User Prop.
- In the Business Component User Properties list, query the Name property for the following string:

```
Immediate Commit In Task
```

6 Set the value property to TRUE.

If the Value property for the Immediate Commit In Task user property is TRUE, then Siebel CRM turns off task transaction, and it immediately saves to the base tables any updates that occur on the business components that the task UI references. For a business component to be part of the task transaction, the task transaction must be turned on at the business component level and at the task level. For more information, see "Using the Defer Write Record Property When You Disable Task Transaction" on page 159.

Using the Defer Write Record Property When You Disable Task Transaction

If you set the Immediate Commit In Task business component user property to TRUE, and if a Siebel operation step references the business component where this property resides, and if the Operation property for this Siebel operation step is set to Insert, then do not set the Defer Write Record property in this Siebel operation step to TRUE. This requirement applies to any Siebel operation step in any task UI.

If you set the Transactional property for a task UI to FALSE, and if this task includes a Siebel operation step, and if the Operation property for this step is set to Insert, then do not set the Defer Write Record property for this Siebel operation step to TRUE. This requirement applies to any Siebel operation step in this task UI.

For more information, see "About the Defer Write Record Property" on page 71.

Troubleshooting a Task UI

This topic describes how to troubleshoot problems that you might encounter when you develop a task UI. It includes the following topics:

Task UI Does Not Display in the Task Pane

If Siebel CRM does not display your task UI in the task pane, then do the following work:

- Make sure you add the responsibility that you assign to the user to the task UI. For more information, see "Adding a Responsibility to a Task UI" on page 166.
- Make sure the task UI references a view task group. Siebel CRM displays a task group that the the Task Pane View references before it displays a view that is specific to a task group that it arranges sequentially. For more information, see "Creating a Task Group" on page 92.
- Make sure the view task group references one of the following items:
  - The current standard view in the current application
  - The Task Pane View

  For more information, see "Adding a Task Group to a View" on page 93.
- Make sure the parent task group references a view.
- Make sure object definitions exist in the repository for the following items:
  - Task UI
  - Task group
  - Metadata
- Make sure you compile the objects that your task UI references to the SRF (Siebel Repository File). Example objects include the task applet, business component, and so on. For more information, see "Preparing to Publish a Task UI" on page 164.
- Make sure the following is true:
  - You publish the task UI to the current run-time database.
  - This publication includes an activation date that has passed or is not defined.
  - This publication includes an expiration date that has not passed or is not defined.

  For more information, see "Publishing a Task UI" on page 164.
- Make sure you properly defined the Context Business Component property. For more information, see "Using the Context Business Component and Task Display" on page 37.
- Make sure you define the task UI as a task that is running in a Siebel application. For example, Call Center. For more information, see "Adding a Task Group to a View" on page 93.

Record Context Is Lost

If Siebel CRM loses the record context while the user navigates between views, even though no Query operation exists, then you must make sure the following items are true:

- The search specifications across views are consistent.
- The search specifications for the view step match.

Inconsistent Search Specifications Across Views

If all of the following situations exist, then Siebel CRM repeats a search:

- The first view includes an applet search specification or a view search specification, but the second view does not include either of these search specifications.
- The Retain Applet Search Spec property or the Retain View Search Spec property is set to FALSE.

The Retain Applet Search Spec property or the Retain View Search Spec property is set to FALSE, so Siebel CRM repeats the search, and then it loses the record context. The second view is supposed to display data from the business component without the constraint that the search specification of the first view imposes.

You must configure Siebel CRM to retain the search specification to avoid this situation. To do this, you set the Retain Applet Search Spec property or the Retain View Search Spec property to TRUE.

Guidelines for Defining a Search Specification

To more clearly define the scope of the search specification, it is recommended that you set the constraint for a search specification on the business component or on the view step rather than on the applet. Note the following situations:

- A search specification on a business component applies throughout the task UI.
- A search specification on a view step applies only for the step where you define it.
- A search specification on a view step applies until Siebel CRM runs the next task UI step, depending on the value of the Retain View Search Spec property.

Specialized Code That Starts a Task UI Might Cause a Loss of Context

If any of the following situations apply, then specialized code might start a task UI and can cause a loss of record context:

- The steps include an identical applet.
- The steps include identical search specifications.
- The Retain Applet Search Spec property is TRUE.
- The Retain View Search Spec property is TRUE.
- Repeat runs of the task UI occur.

To debug this situation, you can perform the following work:

- Use logging with the ObjMgrSqlLog event set to 5 to inspect the search specification that Siebel CRM uses for each SQL statement.
- Review your log for the CSSSqlObj::InvalidateSQLCursor call.
- Determine if performing this call creates an error.

Build View Errors Occur with Task UI

The following build view errors might occur if the user navigates through a task UI, and if the mode for this task UI is set to In Task UI:

- Siebel CRM does not delete the previous view so an active view still exists. In this situation, Siebel CRM displays a dialog box in the Siebel client. You can dismiss the message, correct the problem, and then try to navigate or cancel the task UI.
- Siebel CRM deletes the previous view, so no active view is available. In this situation, Siebel CRM displays an HTML page in the Siebel client that describes the error.

Troubleshooting Other Errors

This topic describes problems you might encounter when you develop a task UI. To resolve the problem, look for it in the Symptom or Error Message column in Table 15. For more information, see "Overview of Transient Data" on page 96.

Table 15. Other Problems That Can Occur When You Develop a Task UI

Symptom or Error Message	Solution
A text field does not display properly when a task UI runs. The task UI uses a transient business component with this text field.	Modify the property of the field to Text. The default is RadioButton.
You cannot find a transient business component in the Business Component property of a task step.	Add the transient business component to the corresponding business object: ■ In the Object Explorer, expand the Business Object tree, and then click Business Object Component. ■ In the Business Object Components list, add the required configuration.
Siebel CRM does not display a Task applet while you define a view.	Choose the task UI that Siebel CRM links to the applet.
A task step includes a forward button type of Submit.	Choose a view step and modify the type to Submit. For more information, see "Validation with Forward Navigation in the Task Playbar" on page 42.
Siebel CRM does not list a task UI in the Application - Administration screen, Tasks view.	Publish and activate the task UI.

Table 15. Other Problems That Can Occur When You Develop a Task UI

Symptom or Error Message	Solution
A transient business component is missing when a task UI runs.	Add the transient business component to the corresponding business object: ■ In the Object Explorer, expand the Business Object tree, and then click Business Object Component. ■ In the Business Object Components list, add the required configuration.
The Siebel client does not display views or applets that you defined.	Do the following work: ■ Make sure you registered the views or applets in the Views view of the Application - Administration screen. ■ Make sure you compiled the views and applets to the SRF. ■ Make sure the responsibility for the user possesses the visibility to and can run the task UI. For more information, see "Adding a Responsibility to a Task UI" on page 166.
Siebel CRM does not display the name of the task group in the task pane.	Define a display name, and then update the display name in the task group.
Siebel CRM does not display the task group in the task pane.	Make sure you add the task group to the standard UI.
Siebel CRM does not display the task UI in the task group.	Add the task UI to the task group, and then compile your modifications.
Siebel CRM does not display the task group or the name of the task UI in the standard UI.	Compile the standard UI that contains the objects.
Siebel CRM does not display chapter definitions.	For more information, see "Creating a Task Chapter" on page 201.
Siebel CRM does not display a view name.	Define a display name and update the display name in the view.
Siebel CRM displays an error that indicates that a required field includes no data.	Set the Defer Write Record property on the Siebel operation step to TRUE. For more information, see "About the Defer Write Record Property" on page 71.
A condition does not include a condition that matches a run-time value.	For more information, see "Creating a Branch Connector" on page 78.

Deploying and Migrating a Task UI

This chapter describes how to deploy and migrate a task UI. It includes the following topics:

Process of Deploying a Task UI

This process is a step in "Roadmap for Developing a Task UI" on page 85.

To deploy a task UI, perform the following tasks:

1. Preparing to Publish a Task UI on page 164
2. Publishing a Task UI on page 164
3. Activating a Task UI on page 165
4. Adding a Responsibility to a Task UI on page 166

A *deployed task* is a task UI that is published and activated. You create a task UI in your development environment, and then you deploy it to a testing or production environment. You can use Siebel Tools to publish the task and then activate it in the Siebel client. This process deploys task objects from the SRF (Siebel Repository File) to the run-time environment.

As an alternative, you can publish and activate a task UI in Siebel Tools. For more information, see "Publishing and Activating a Task UI in Siebel Tools" on page 167.

Preparing to Publish a Task UI

You must compile the SRF before you publish a task UI. If you add or modify repository objects that are part of the task UI or that this task references, then you must compile the SRF. Example objects in the SRF that the task might require include business components, business services, and views.

To prepare to publish a task UI

1. Validate the task UI.

   For more information, see "Validating a Task UI" on page 151.

2. Compile modified objects:

   a. In Siebel Tools, choose the Tools menu, and then the Compile Projects menu item.

   b. In the Object Compiler dialog box, click Locked Projects, and then click Compile.

Publishing a Task UI

Publishing a task UI makes the task visible in the Siebel client. Siebel CRM reads the object definition of the task UI from the SRF when you publish it, and then writes the definition and deployment parameters in XML format to the run-time environment. If your task UI includes a subtask step, then you must publish the subtask first so that the subtask is available to the parent task UI in the run-time environment.

To publish a task UI

1 Locate the task UI you must modify.

For more information, see "Locating a Task UI in the Tasks List" on page 60.

2 Make sure the status property is In Progress.

3 On the WF/Task Editor toolbar, click the Publish icon.

For more information, see "Using the WF/Task Editor Toolbar" on page 60.

If you click Publish, then Siebel Tools prompts you to validate your task or to continue.

4 Click Yes to continue.

5 Make sure the status property is Completed.

If Siebel Tools modifies the status from In Progress to Completed, then the task UI is available in the Siebel client.

Activating a Task UI

This topic describes how to activate a task UI.

To activate a task UI

1 Log in to the Siebel Client.

2 Navigate to the Administration - Business Process screen, and then the Task Deployment view.

The Published Tasks list displays task UIs that are published from the repository tables. If you revise and publish a task UI, then Siebel CRM increments the repository version of the task. If you click the task name link, then Siebel CRM displays the child items of the task. These child items are the object definitions of the published task.

3 In the Published Tasks list, query the Name field for the task UI you must activate.

4 In the Published Tasks list, click Activate.

Siebel CRM validates the format, and then modifies the deployment status of the task UI to Active. For more information, see "About the Deployment Status" on page 166

5 Verify that the Deployment Status field of the Active Tasks list displays a value of Active for the task UI that you activated in Step 3.

6 Optional. In the Active Tasks list, define deployment parameters for the task UI:

a In the Activation Date/Time field, set the activation date.

It is not necessary to set an activation date or an expiration date. If you leave these fields empty, then the task UI is active and does not expire.

b In the Expiration Date/Time field, set the expiration date.

c Set the replication to None, unless you are deploying the task UI to a Siebel Mobile Web Client.

If you are deploying the task UI to a Siebel Mobile Web Client, then see "Replicating a Task UI to the Siebel Mobile Web Client" on page 170.

About the Deployment Status

If the deployment status of the previous active version of a task UI exists when you activate a task, then Siebel CRM modifies the deployment status to Outdated. You can view this modification in the Active Tasks applet that Siebel CRM displays in the lower portion of the Task Deployment screen.

The deployment status of a deployed task can include one of the following values:

- **Active**. A task UI that is active is the basis for a new task instance. Only one active version of a task with a specific name can exist.
- **Outdated**. A task instance that is already in progress or that is in the inbox can continue to use the outdated object definition. A new instance does not use the outdated definition.
- **Inactive**. If a task UI is inactive, then no new or paused task instance can use the inactive definition.

Adding a Responsibility to a Task UI

You publish and activate a task UI, specify the responsibilities that can run it, and then define the access rights that Siebel CRM must include in these responsibilities in the inbox. The Registered Task Administration view allows you to determine if a user can run, transfer, or delete a task UI. You must add the responsibility that Siebel CRM assigns to the user to the registered task so that this user can access this task UI.

If a user possesses the privileges that Siebel CRM requires to run a task UI, then this user also possesses the privileges required to access the views that this task UI uses.

You administer access control for a task UI in a way that is similar to how you administer access control for a view.

For more information, see *Siebel Security Guide*.

To add a responsibility to a task UI

1. Log in to the Siebel client.
2. Register the task UI:
   a. Navigate to the Administration - Application screen, and then the Tasks view.
   b. In the Registered Tasks list, click New.
   c. In the Task Name field, choose the task UI that you must register, and then click OK.

      Siebel CRM displays a list that includes deployed tasks. For more information, see "Process of Deploying a Task UI" on page 164.
3. Add a responsibility to the task UI you registered in Step 2:
   a. Make sure the task UI you registered in Step 2 is still chosen in the Registered Tasks list.
   b. In the Responsibilities list, click New.

c In the Tasks dialog box, query for the responsibility that must include access to the task UI, and then click OK.

If you typically log in with administrator privileges, then, for testing purposes, add the Siebel Administrator responsibility.

d In the Responsibilities list, make sure each of the following fields include a check mark:

- **Allow Delete.** Allows the user to delete a paused task UI that Siebel CRM displays in the Universal Inbox.
- **Allow Transfer.** Allows the user to transfer a paused task.

The default value for each of these properties includes a check mark. For more information, see "Resuming a Paused Task UI" on page 187, and "Universal Inbox" on page 47.

4 In the Registered Tasks list, click Clear Cache.

5 Log out of the Siebel client.

Publishing and Activating a Task UI in Siebel Tools

You can publish and activate a task UI in Siebel Tools instead of publishing it in Siebel Tools and then activating it in the Siebel client. For more information, see "Process of Deploying a Task UI" on page 164.

To publish and activate a task UI in Siebel Tools

1 Validate the task UI.

For more information, see "Validating a Task UI" on page 151.

2 Compile locked projects.

For more information, see "Preparing to Publish a Task UI" on page 164.

3 To publish and activate the task UI, do the following:

a Make sure the latest version of the task UI you are publishing is chosen in the Tasks list.

b In the WF/Task Editor Toolbar, choose the Publish/Activate button.

For more information, see "Using the WF/Task Editor Toolbar" on page 60.

Do not set the optional fields.

4 Optional. For testing purposes, add the Siebel Administrator responsibility to the task UI.

For more information, see "Adding a Responsibility to a Task UI" on page 166.

5 If the Siebel client is open, then clear the cache, and then log out of the Siebel client.

Migrating a Task UI

This topic is a step in "Roadmap for Developing a Task UI" on page 85.

Migration is the process of moving object definitions between environments. For example:

- Between a development environment and a test environment
- Between a test environment and a production environment

This topic describes deployment and migration options that are available with a task UI. Deployment and migration options for a task UI are similar to the deployment and migration options that are available for a workflow process. For more information, see *Siebel Business Process Framework: Workflow Guide*.

Before you migrate data, make sure the data that the task UI requires exists in the target environment. For example, if the task UI requires custom entries in the List Of Values (LOV) table, then make sure these entries exist and are active.

To migrate a task UI

1. Choose the migration option that most closely meets your requirements.

   For more information, see "Comparison of Migration Options" on page 169.

2. Do the migration.

   For more information, see the relevant documentation for the migration option you chose in Step 1.

Comparison of Migration Options

Table 16 compares options for migrating a task UI.

Table 16. Comparison of Options for Migrating a Task UI

Migration Option	Description
Application Deployment Manager (ADM)	Use ADM in the following situations: ■ You are migrating customization data for your entire enterprise, including data from a task UI. ■ You must migrate more than 10 task UI deployments at one time. ADM is a tool that streamlines the process of deploying enterprise customization data, such as views, responsibilities, assignment rules, tasks, and workflow processes. A task deployment package in ADM includes the Siebel Repository File (SRF), tasks, subtasks, and their run-time settings, such as activation and expiration times. For more information, see *Siebel Application Deployment Manager Guide*.
Repository Import and Export (REPIMEXP)	Use REPIMEXP in the following situations: ■ You are rolling out your release and you must migrate most or all repository objects. ■ You do not need to use ADM or you cannot use ADM. REPIMEXP is a manual process that you can use for repository migration. You can use it to migrate repository objects in bulk, including object definitions for a task UI. REPIMEXP is a command line utility that resides in the BIN directory of your Siebel installation. Enter the following command in a command prompt to view usage options: `repimexp /help` If you use REPIMEXP, then you cannot pick and choose the task UIs to migrate. You can use one of the other migration options to migrate a single task UI or to migrate only some task UIs.
Import and export	Use import and export in the following situations: ■ You can migrate task UIs in increments of approximately 10 or less. ■ You must back up sets of objects. ■ You must migrate objects. In Siebel Tools, you can export objects from the SRF to an archive file (SIF), and then import objects from the archive file back to the SRF. You can include individual objects or entire projects in an archive file. Example individual objects include business components, applets, views, and task UIs. For more information, see *Using Siebel Tools*.
Publish	For more information, see "Publishing a Task UI" on page 164.

Replicating a Task UI to the Siebel Mobile Web Client

This topic describes how to configure a task UI to run on a Siebel Mobile Web Client that can run with a regional database or a local database. The Inbox screen is visible in the Siebel Mobile Web Client. Siebel CRM does the following, by default:

- Replicates to the Siebel Mobile Web Client a task UI inbox item that the user creates on the Siebel Server.
- Replicates to the Siebel Server a task UI inbox item that the user creates on the Siebel Mobile Web Client.
- Does not automatically replicate to the Siebel Mobile Web Client any object definitions for a task UI that reside in the SRF (Siebel Repository File).

To replicate a task UI to the Siebel Mobile Web Client

1. In the Siebel client, navigate to the Administration - Workflow Process screen, Task Deployment, and then the Published Task view.
2. In the Active Tasks list, query the Name field for the task deployment record that you must replicate.

   For example, search for the name of the task UI.
3. Modify the Replication field of the task deployment record according to the following requirements:
   - To Regional for a Siebel Mobile Web Client that connects only to a regional database
   - To All for a Siebel Mobile Web Client that connects to a regional or a local database

How Siebel CRM Runs a Task UI Across Nodes

A run-time instance of a task UI is the underlying object of a task UI inbox item. Siebel CRM does not replicate a task run-time instance between the Siebel Server and the Siebel Mobile Web Client. A user can run a task inbox item only on the same node where Siebel CRM creates this task inbox item. For example, if the node is a server, then the user can only run the instance from this Siebel Server. If the user attempts to run a task UI inbox item from an incorrect node, then Siebel CRM returns an error.

Siebel CRM allows a task transfer only between users that reside on the same node. A user can transfer a task inbox item that Siebel CRM creates on the Siebel Server or regional node to other users that reside on the same server or regional node. If a user creates a task inbox item while using a Siebel Mobile Web Client, and if the user connects this client to a local database during login, then Siebel CRM cannot transfer this task inbox item.

Setting Synchronization Interval for a Task UI

You can set the synchronization interval for a task UI that Siebel CRM replicates to the Siebel Mobile Web Client to manage how frequently Siebel CRM does synchronization. For more information, see *Siebel Remote and Replication Manager Administration Guide*.

Administering a Task UI

This topic is a step in "Roadmap for Developing a Task UI" on page 85.

This topic describes how to administer a task UI. It includes the following topics:

This topic is a step in "Roadmap for Developing a Task UI" on page 85.

Using the Task Instance Monitor

The *Task Instance Monitor* is an administrative tool that you can use to view a detailed status of active tasks and tasks that Siebel CRM has recently run.

To use the Task Instance Monitor

1. Log in to the Siebel client with administrator privileges.
2. Adjust the monitoring level for the task UI you must monitor:
   a. Navigate to the Administration - Business Process screen, and then the Task Deployment view.
   b. In the Child Items list, locate the task UI you must monitor.
   c. In the Monitoring Level field, choose a monitoring level.

      Siebel CRM sets the monitoring level for every task UI to None, by default. You must set the monitoring level to a value other than None to collect data for a task UI. For more information, see "Monitoring Levels of the Task Instance Monitor" on page 172.
3. Run the task UI that you set up to monitor in Step 2.
4. Navigate to the Administration - Business Process screen, and then the Task Instance Monitor view.

5 In the Task Name field, locate the task UI you set up to monitor in Step 2.

The Task Name field uses the following format:

name:*number*

where:

- *name* is the name of the task UI.
- *number* is the repository version of the task.

For example:

BPF-Acceptance:32

The Task Instance Monitor displays information only for task UIs that are active and published. All fields are read-only.

6 Examine information about the task instance.

For more information, see "Fields That Siebel CRM Displays in the Task Instance Monitor" on page 173.

7 To examine information about task steps for this task instance, do the following:

a Click the Task Step Instances tab.

The Task Step Instances tab lists all the steps that constitute this task UI. It includes information for each task step. Example information includes Start Time, End Time, Status, and so on.

b To view information about process properties for the step, examine details in the bottom applet.

Siebel CRM displays information about process properties in the bottom applet. The step you choose in the Task Step Instances list references these process properties.

Monitoring Levels of the Task Instance Monitor

Table 17 describes the monitoring levels you can adjust in the task instance monitor. Siebel CRM uses the monitoring level that you set for each task UI, even if the monitoring level is different for a subtask. For example, assume you set the monitoring level to Progress for the parent task UI and to None for a subtask. In this situation, Siebel CRM does the following:

- Displays a new record for any new instance of the parent task UI
- Does not display a new record for any new instance of the subtask

If a task UI includes an error step, and if a task instance of this task UI runs the error step, then Siebel CRM still displays a record for the task instance according to the monitoring level you set.

Table 17. Monitoring Levels You Can Set in the Task Instance Monitor

Monitoring Level	Description
None	Siebel CRM does not display a new record for any task instance in any of the following views: ■ Task Instance Monitor ■ Task Step Instances ■ Task Report
Status	Siebel CRM displays a new record in the Task Instance Monitor view and the Task Report view for any new task instance. It does not display a new record in the Task Step Instances view.
Progress	Siebel CRM displays a new record for any new task instance in each of the following views: ■ Task Instance Monitor ■ Task Step Instances ■ Task Report
Detail	Siebel CRM displays the same information that it displays for the Progress monitoring level, plus information about process properties. To view this information, see Step 7 on page 172.
Debug	Displays the same information as the Detail monitoring level.

Fields That Siebel CRM Displays in the Task Instance Monitor

Table 18 describes fields that Siebel CRM displays in the Task Instance Monitor.

Table 18. Fields That Siebel CRM Displays in the Task Instance Monitor

Field	Description
Application	The name of the Siebel application where Siebel CRM runs this task instance. For example, Siebel Call Center.
Current Step Name	The step name of the currently active step of this task instance.
Duration	The total amount of time that has elapsed between the Start Time and the End Time.
End Time	The time of day that the user completed or canceled the task UI.
Instance Id	A value that uniquely identifies an instance of a task UI or subtask.

Table 18. Fields That Siebel CRM Displays in the Task Instance Monitor

Field	Description
Owner Id	The name of the user who started the task instance.
Resume Time	The time of day when the user resumed a paused task.
Server Name	The name of the Siebel Server where the task instance runs.
Start Time	The time of day that the user started this task instance.
Status	Displays the status. It can include one of the following values: ■ Running ■ Paused ■ Stopped ■ Completed ■ In Error If the Status is In Error, then Siebel CRM displays the font color for this task UI in red.
Task Id	A value that uniquely identifies a task UI or subtask.
Task Name	The value that Siebel Tools displays in the Display Name property for the task UI.

Using Task Reports

You can create a task report that includes historical information about completed task instances. It can include information from fields that Siebel CRM displays in the Task Instance Monitor, such as Start Time, End Time, and so on. For more information, see "Fields That Siebel CRM Displays in the Task Instance Monitor" on page 173.

You can use task reports in the Siebel client. Siebel CRM includes a report record even if the user cancels or stops a task instance. For example, if the user cancels a task instance, then Siebel CRM includes information about all task steps that the user completed.

To use task reports

1. Make sure the file system includes an att folder.

   For example:

   ```
   \machine\fs\att
   ```

   Siebel CRM uses this att folder to create a report.

2. Set the monitoring level and then run the task UI.

   For more information, see "Using the Task Instance Monitor" on page 171.

3 Log in to the Siebel client with administrator privileges.

4 Navigate to the Administration - Business Process screen, and then the Task Reports view.

5 Click the following Reports icon:

6 Choose one of the following output formats for the report:

- Task Report - MHTML
- Task Report - PDF
- Task Report - PPT
- Task Report - RTF

7 In the File Download dialog box, choose one of the following options:

- **Open.** Siebel CRM opens the report in a separate window according to the output format you choose in Step 6. For example, if you choose PDF, then Siebel CRM uses the program that you configure your computer to use to open a PDF file.
- **Save.** Siebel CRM saves the report to a location that you specify.

Allowing Task Transfer

Siebel CRM allows a user to transfer a task UI to another user, by default. If you configure Siebel CRM to not allow a task transfer, then it makes the owner field read-only and disables the Transfer button for the current task item that it displays in the Inbox Items List view.

To allow task transfer

1 Log in to the Siebel client with administrator privileges.

2 Navigate to the Administration - Application screen, and then the Tasks view.

3 In the Registered Tasks list, choose the task UI where you must control task transfer.

4 In the Responsibilities list, choose the responsibility where you must allow task transfer, and then make sure the Allow Transfer option includes a check mark.

To disable task transfer, make sure the Allow Transfer option does not include a check mark.

5 Step off the record to save your modifications.

Transferring a Task Instance

The owner of a task instance can use the Universal Inbox to transfer a task UI.

Siebel CRM constrains the list of users that the user can choose to transfer a task instance. It constrains this list to users whose responsibility is added for the task. For more information, see "Adding a Responsibility to a Task UI" on page 166.

To transfer a task instance

1. In the Siebel client, navigate to the Inbox screen, and then the Inbox Items List view.
2. Choose the task UI you must transfer.
3. Make sure the Transfer button is available.

   If Siebel CRM disables the Transfer button, then you cannot transfer this task instance.
4. Click Transfer.

   Siebel CRM displays the Transfer To dialog box.
5. In the Owner field, choose the owner to whom you must transfer the task instance.
6. In the Comments field, enter a reason for the transfer.
7. Click OK, and then refresh the screen.

   Siebel CRM does the following:

   - Closes the pop-up applet
   - Removes the task instance from your inbox
   - Displays the task instance in the inbox of the new owner
   - Enters a check mark in the Completed field in your My Inbox Items list
   - Adds the task instance to your My Completed Items list

Deactivating a Deployed Task UI

This topic describes how to deactivate a task UI that you deployed to the Siebel client.

To deactivate a deployed task UI

1. In the Siebel Client, navigate to the Administration - Business Process screen, and then the Task Deployment view.
2. In the Active Tasks list, query the Name field for the task UI you must deactivate.
3. Choose the task UI, and then click Deactivate Task.

   Siebel CRM modifies the deployment status of the task UI to Inactive.

Deleting a Deployed Task UI

This topic describes how to delete a task UI that you deployed to the Siebel client.

To delete a deployed task UI

1. In the Siebel Client, navigate to the Administration - Business Process screen, and then the Task Deployment view.
2. In the Active Tasks list, query the Name field for the task UI you must delete.
3. Click the task UI, and then click Delete Task.

   CAUTION: Deleting a deployed task UI affects running instances of this task and might corrupt data. If you must delete a deployed task UI, then it is recommended that you deactivate the task first, and then delete it. For more information, see "Deactivating a Deployed Task UI" on page 176.

Deleting a Task Instance From the Inbox

You can delete a task instance from the inbox.

To delete a task instance from the inbox

1. In the Siebel client, navigate to the Administration - Application screen, and then the Tasks view.
2. Choose the task UI that you must delete.
3. Make sure the Allow Delete option includes a check mark.
4. Step off the record to save your modifications.
5. Navigate to the Inbox screen, and then the Inbox Items List view.
6. Choose the task UI that you must delete.
7. Click the Detail tab, and then click Delete.

   Siebel CRM removes the task instance from the inbox.

Removing Temporary Data After a Task UI Finishes

Siebel CRM stores temporary data for a task instance in the S_TU_LOG table. The user finishes a task instance, and then the Siebel Object Manager clears the temporary data. This behavior varies depending on the following data source where the user connects:

- **Server.** For performance reasons, a batch process runs every five minutes that deletes used records from the S_TU_LOG table.
- **Local.** Deletes rows from the S_TU_LOG table as soon as the user finishes the task instance.
- **Sample.** Never deletes rows from the S_TU_LOG table.

You can run the Task Log Cleanup Service business service to manually delete temporary data that Siebel CRM stores for a finished task instance. This service allows you to specify when Siebel CRM deletes temporary data rather than performing the delete at a predefined interval.

For more information, see "Transparent Storage" on page 226.

To remove temporary data after a task UI finishes

1 Add a business service step to your task UI immediately after a commit step. Use values from the following table.

Property	Value
Business Service Name	Task Log Cleanup Service
Business Service Method	CleanTaskLog

2 Optional. Define the Task Id input argument for the business service that you added in Step 1.

Siebel CRM does the following:

- If you specify a value for the Task Id input argument, then Siebel CRM deletes only temporary data for the task UI that the Task Id identifies.
- If you do not specify a value, then Siebel CRM deletes all expired rows from the S_TU_LOG table.

Configuring Siebel CRM to Resolve Task Transaction Conflicts

This topic describes how to configure Siebel CRM to resolve task transaction conflicts. For more information, see "About Task Transaction" on page 225.

Configuring Siebel CRM to Resolve Duplicate Conflicts

This topic describes how to configure Siebel CRM to resolve duplicate conflicts. A *duplicate conflict* is a type of conflict that occurs when a unique key violation exists in the RDBMS during a commit.

To configure Siebel CRM to resolve duplicate conflicts

1 Open Siebel Tools.

2 Display the Business Component User Property object type.

For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.

3 In the Object Explorer, click Business Component.

4 In the Business Components list, locate the business component you must modify.

5 In the Object Explorer, expand the Business Component tree, and then click Business Component User Prop.

6 In the Business Component User Properties list, add a new record using values from the following table.

Name	Description
Dup Conflict In Task	Enter one of the following values: ■ **Resolve**. Siebel CRM writes the duplicate record with the Conflict Id field set to the value of Id, by default. ■ **Fail**. Siebel CRM stops the task transaction. ■ **Ignore new record**. Siebel CRM skips the duplicate record but saves the rest of the task transaction.

Configuring Siebel CRM to Resolve Update and Delete Conflicts

You can configure Siebel CRM to resolve the following types of conflicts:

- **Update conflict.** Occurs if Siebel CRM modifies the same record inside and outside of the task transaction. Similar to the standard UI, Siebel CRM detects the update conflict according to the Modification Id system field.
- **Delete conflict.** Caused by one of the following reasons:
  - Siebel CRM deletes a record in a task transaction, and then updates this record outside the task transaction.
  - Siebel CRM updates a record in a task transaction, and then deletes this record outside the task transaction.

Siebel CRM detects an update conflict or a delete conflict in the following situations:

- When it gets data in the task transaction
- During a commit

To configure Siebel CRM to resolve update and delete conflicts

- Create a task property using values from the following table.

Property	Description
On Conflict	Use one of the following values: ■ **Continue operation**. Siebel CRM overwrites modifications that the user saves outside of the task transaction with modifications that the user makes in the task transaction. If the user deletes the record outside of the task transaction, then Siebel CRM returns an error message. ■ **Cancel operation**. Siebel CRM displays an error message that describes the conflict. Data that resides in conflicting fields in the task transaction are lost. The user must manually reenter values in the fields to resolve this conflict.

For more information, "Creating a Task Property" on page 83.

10 Customizing Task UI

This chapter describes how to customize a task UI. It includes the following topics:

Starting a Task UI

A task UI typically starts when the user clicks a task group item that Siebel CRM displays as a link in a task group in the task pane. This topic describes options for starting a task other than through the task pane. It includes the following topics:

Creating a Button to Start a Task UI

You can create a button on an applet that a user can click to start a task UI. For example, a button labeled Update Contact Info in the standard My Contacts list. For more information, see *Configuring Siebel Business Applications*.

To create a button to start a task UI

1. In Siebel Tools, display the Control User Prop object type.

   Control User Prop is a child of the Control object type, which is a child of the Applet object type. For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.

2. In the Applets list, query the Name property for the applet that contains the control that starts the task UI.

3 In the Object Explorer, expand the Applet tree, and then click Control.

4 In the Controls list, add a new control using values from the following table.

Property	Description
HTML Type	Choose MiniButton
Caption	Define the Caption property so the user can readily identify the control.
Method Invoked	Enter LaunchTask. This option is not available from the list of values. You must use the keyboard to manually type it in the property.

5 Make sure the control you defined in Step 3 is still chosen.

6 In the Object Explorer, expand the Control tree, and then click Control User Property.

7 In the Control User Props list, add a new user property using values from the following table.

Property	Description
Name	Choose Task Name
Value	Enter the name of the task UI.

Creating a Menu Item to Start a Task UI

You can create a menu item on an applet that the user can click to start a task UI. For more information, see *Configuring Siebel Business Applications*.

To create a menu item to start a task UI

1 In Siebel Tools, in the Object Explorer, click Command.

2 In the Commands list, create a new command using values from the following table.

Property	Description
Name	Enter text that describes the logic that the command performs.
Business Service	Choose Task UI Service (SWE). **CAUTION:** You must make sure that the business service you specify does not include a browser script. A business service only works with a server script. If an applet menu item on the Siebel Server calls a business service that includes a browser script, then the business service fails.
Target	Choose Server.
Method	Choose LaunchTask.
Method Argument	Enter the name of the task UI.

The task UI to open is defined in the properties of the command, so you must define a new command for each task UI.

3 Add a menu item to the applet menu that calls the command that is defined for the task UI.

Creating an iHelp Link to Start a Task UI

You can use the following types of links in the iHelp Pane:

- An iHelp item
- A task UI

To create an iHelp link to start a task UI

1 In the Siebel client, choose the Administration - iHelp screen, and then the All iHelp Items view.

2 Create an iHelp item.

For more information, see *Siebel Applications Administration Guide*.

3 In the iHelp Items list, choose an iHelp item.

4 Click Revise, and then click the More Info tab.

5 In the Related Task field, choose a task UI.

To display a link for a task UI in the iHelp Pane, you must create a relationship between the task and an iHelp item on the iHelp Pane. For more information, see *Siebel Applications Administration Guide*.

6 To activate the iHelp item, click Activate.

   If the Activate button is not available, then click the Responsibility tab, and make sure the Active Flag option includes a check mark for your responsibility. For more information, see "Adding a Responsibility to a Task UI" on page 166.

7 Click the iHelp icon to view the link that resides in the iHelp item that Siebel CRM displays to start the task UI.

Creating a Workflow Process to Start a Task UI

You can start a long-running workflow process that starts a task UI. For more information about how to do this, see *Siebel Business Process Framework: Workflow Guide*.

Creating a Script to Start a Task UI

You can use a browser script or a server script to start a task UI. Your script calls the Task UI Service business service. It then passes the name of the task UI to the LaunchTaskFromScript business service method to start the task UI. You must compile the script into proxy JavaScript objects or to the SRF (Siebel Repository File). Processing varies depending the following type of script you use:

- **Browser script.** Siebel CRM handles the InvokeMethod methods for a script on the client objects, and then passes them to their server equivalents.
- **Server script.** Siebel CRM compiles the JavaScript files that include the scripted methods to the SRF, and then calls them from the Object Manager at the required preevent or postevent.

In this situation, browser script or server script does not use the CanInvokeMethod method.

Siebel CRM does the following work at run time:

1 The Task UI Service examines the name of the task UI and makes sure the user possesses the license and responsibility to run the task. For more information, see "Adding a Responsibility to a Task UI" on page 166.

2 If Siebel CRM allows the user to run the task, then the service passes the pointer of the active business component to the Task Controller, designating it as the context business component.

3 If any of the following situations exist, then the task controller creates an error:

   - The task requires a business component that is not the same as the context business component.
   - The task is not activated.

4 If Siebel CRM does not encounter an error, then it runs the task UI and opens the task pane.

Note the following restrictions when using a script with a task UI:

- Calling a task UI from a script requires *UI context*, meaning that Siebel CRM can reference a record in an applet. You must not configure Siebel CRM to start a task from a script that does not include UI context, such as a script for a workflow process, or from an event where the UI did not finish processing, such as the Applet_Load event.

- A script can pass only the name of the task UI. It cannot pass any other parameters.
- You can use a script only to start a task UI. You cannot use a script to interact with a task in any other way.

For more information, see "About Event Handling" on page 228.

Example of a Client Script

In this example, the browser script locates the Create a Contact task UI, and then starts the task:

```
function Applet_PreInvokeMethod (name, inputPropSet)
{

   try
   {

      if (name == "Test")
      {

         var inputPropSet;
         var outputPropSet;
         var taskUIsvc;

         inputPropSet = theApplication().NewPropertySet();

         outputPropSet = theApplication().NewPropertySet();

         taskUIsvc = theApplication().GetService("Task UI Service (SWE)");

         inputPropSet.SetProperty("TaskName","Create a Contact");

         <!-- Note: because taskUIsvc.Invokemethod() is required to pass outputPropSet,
         the outputPropSet is created. outputPropSet is not used to send results back
         to the task UI--!>

         taskUIsvc.InvokeMethod("LaunchTaskFromScript",inputPropSet,outputPropSet);

         return ("CancelOperation");

      }

   }

   catch(e)
   {

      theApplication().alert("Error" + e.toString());

   }

   finally
   {
   }

   return ("ContinueOperation");
```

```
}
```

Example of a Server Script

In this example, the server script opens the Create a Contact task UI:

```
function WebApplet_PreInvokeMethod (MethodName)
{

   try
   {

      if (MethodName == "Test")
      {

         var inputPropSet;
         var outputPropSet;
         var taskUIsvc;

         inputPropSet = TheApplication().NewPropertySet();

         outputPropSet = TheApplication().NewPropertySet();

         taskUIsvc = TheApplication().GetService("Task UI Service (SWE)");

         inputPropSet.SetProperty("TaskName","Create a Contact");

         <!-- Note: because taskUIsvc.Invokemethod() is required to pass outputPropSet,
         the outputPropSet is created. outputPropSet is not used to send results back
         to the task UI--!>

         taskUIsvc.InvokeMethod("LaunchTaskFromScript",inputPropSet,outputPropSet);

         return (CancelOperation);

      }

   }

   catch(e)
   {

      TheApplication().RaiseErrorText("Error" + e.toString());

   }

   finally
   {
   }

   return (ContinueOperation);

}
```

Resuming a Paused Task UI

This topic describes configuration options to resume a paused task UI. It includes the following topics:

For more information, see the following topics:

Creating an Association That Allows the User to Resume or Transfer a Paused Task UI

This topic describes how to configure a task instance to reference a business component instance. A user can pause a task UI. The same or another user can resume the task at a later time, retaining the task state. You can configure a task to reference a business object instance so that Siebel CRM can transfer a paused task between users.

For example, assume you are a customer service representative (CSR) in a 100 person IT contact center for a computer manufacturer. The primary business process you perform, diagnosing problems with IT computer systems, beings with a task UI that requires complex information from a customer. Occasionally, the customer might call you back with diagnostic information that requires you to pause the task. Another CSR might answer the phone when the customer calls, so this CSR must access the paused task.

The CSR must look up the customer and view a list of the task UIs that are open for the CSR when the call comes in, regardless of who started or paused that task. The CSR must take ownership of the paused task and resume it. To use this pause and transfer functionality, you must associate the task with a business object instance so that Siebel CRM can retain the current task state.

To create an association to resume or transfer a paused task UI

1. Determine where you must locate the business service step in the task UI.

   For more information, see "Locating the Business Service Step in the Task UI" on page 188.

2 Add a business service step to the task UI using values described in the following table.

Property	Value
Business Service Name	Task Administration
Business Service Method	Associate

3 Make sure the business service step you added in Step 2 is still chosen.

4 In the Multi Value Property Window, add a new input argument using values described in the following table.

Field	Value
Input Argument	ObjectId The ObjectId identifies the business component record that the task UI references.
Type	Literal
Value	Enter the RowId of the business component record that the task UI references.

5 In the Multi Value Property Window, add a new input argument using values described in the following table.

Field	Value
Input Argument	ObjectType The ObjectType is the name that Siebel CRM uses for the association. It uses this configuration as part of the search specification. The ObjectType is not required to match the business component name, but including it can help to keep objects synchronized.
Type	Literal
Value	Name of the business component that the task UI references.

6 If necessary, customize the user interface.

For more information, see "Process of Creating a View That Allows a User to Transfer a Paused Task UI" on page 189.

Locating the Business Service Step in the Task UI

The task UI must call the Associate method of the Task Administration business service as early as possible in the task flow. The configuration varies depending on the following:

- **The task UI creates a new record that must be associated.** Siebel CRM can do this call between the view that first validates the new record and the next subsequent commit step. The user cannot view the associated record outside of the task UI before Siebel CRM saves the task transaction.
- **The task UI updates an existing record.** Siebel CRM must create the association with this record before it displays the first view in the task UI. In this situation, the ObjectId parameter of the Associate method typically uses the value that the Context BC Id task property contains.

Limitations of Associating a Task UI with a Long-Running Workflow Process

If a long-running workflow process creates a task UI but never starts it, then Siebel CRM cannot associate this task UI with a business object. To avoid this situation, you must configure Siebel CRM to start the task manually or automatically.

Process of Creating a View That Allows a User to Transfer a Paused Task UI

To create a view that allows a user to transfer a paused task UI, perform the following tasks:

1. Creating a New Link to Support Task Transfer on page 189
2. Modifying the Business Object to Support Task Transfer on page 190
3. Creating a Standard View to Support Task Transfer on page 191
4. Modifying a Screen to Support Task Transfer on page 191

This topic describes how to configure the user interface for a custom business object to be *multicall capable*, which is a configuration that allows you to define a task view on a custom screen that allows the user to view a paused task for a parent record. For example, you can add a service request view that includes a parent service request applet and a child applet that includes a list of paused inbox items. This configuration allows the user to transfer a paused task to another user from a view rather than from the Inbox Items List view.

Creating a New Link to Support Task Transfer

This task is a step in "Process of Creating a View That Allows a User to Transfer a Paused Task UI" on page 189. It describes how to create a new link to allow for task transfer.

To create a new link to support task transfer

1. In the Object Explorer, click Link.

2 In the Links list, create a new link using values described in the following table.

Property	Description
Parent Business Component	Choose the business component where you must use task transfer.
Child Business Component	Choose UInbox Item Context.
Source Field	Choose Id.
Destination Field	Choose Item Context Id.
No Update	Choose TRUE.
No Delete	Choose TRUE.
No Insert	Choose TRUE.
Search Specification	Enter the following code: `(LookupName('WF_INST_STAT_CD', [Task Status]) = 'PAUSED' OR LookupName('WF_INST_STAT_CD', [Task Status]) = 'TRANSFERRED') AND [Item Context Object Name] = '`*`object_name`*`'` where: ■ *object_name* is the name you use to associate the task UI. This name must match the string that you use in the definition of the task UI when you associate this task with the business object. It does not need to match the name of the business object.

For more information, see "Resuming a Paused Task UI" on page 187.

3 Right-click the new link, choose Validate, and then click Start.

Modifying the Business Object to Support Task Transfer

This task is a step in "Process of Creating a View That Allows a User to Transfer a Paused Task UI" on page 189.

This topic describes how to modify the business object to support task transfer.

To modify the business object to support task transfer

1 In the Object Explorer, click Business Object.

2 In the Business Objects list, query the Name property for the business object where you must use task transfer.

3 In the Object Explorer, expand the Business Object tree, and then click Business Object Component.

4 In the Business Object Components list, create a new business object component using values described in the following table.

Property	Description
Bus Comp	Choose UInbox Item Context.
Child Business Component	Choose the new link.

5 Right-click the business object, choose Validate, and then click Start.

Creating a Standard View to Support Task Transfer

This task is a step in "Process of Creating a View That Allows a User to Transfer a Paused Task UI" on page 189.

This topic describes how to create a new standard view to allow for task transfer.

To create a standard view to support task transfer

1 In Siebel Tools, choose the File menu, and then the New Object menu item.

2 In the New Object Wizards dialog box, on the General tab, choose View, and then click OK.

3 Follow the prompts in the wizard to define the new view, using values described in the following table.

Property	Description
Business Object	Choose the business object that you modified in the "Modifying the Business Object to Support Task Transfer" on page 190 topic.
View Name	Enter a descriptive name for the view. For example, ObjectName - Paused Tasks.
Upgrade Behavior	Choose Preserve. This setting preserves this modification during an upgrade.
Web Template	Choose a master and detail template. For example, View Basic.
Master Applet	Choose a master applet according to the master business component that is defined.
Child Applet	Choose Task Item Context List Applet.

4 Edit the Web layout, as necessary.

5 Right-click the new view, choose Validate, and then click Start.

Modifying a Screen to Support Task Transfer

This task is a step in "Process of Creating a View That Allows a User to Transfer a Paused Task UI" on page 189. This topic describes how to modify a screen to support task transfer.

To modify a screen to support task transfer

1. In the Object Explorer, click Screen.
2. In the Screens list, query the Name property for the screen where you must define the task transfer view.
3. In the Object Explorer, expand the Screen tree, and then click Screen View.
4. In the Screen Views list, add a new screen view using values described in the following table.

Property	Description
View	Choose the view that you modified in the "Creating a Standard View to Support Task Transfer" on page 191 topic.
Sequence	Define a sequence that correctly positions this view on the site map.
Type	Choose Detail View.
Parent Category	Choose a value.
Viewbar Text	Define a value. For example, for a task UI, you can use the following symbolic string: `SBL_TASKS-1004224752-2S0`
Menu Text	Define a value. For example, for a task UI, you can use the following symbolic string: `SBL_TASKS-1004224752-2S0`
Display In Page	Choose TRUE.
Display In Site Map	Choose TRUE.
Upgrade Behavior	Choose Preserve. This setting preserves this modification during an upgrade.

5. In the Screen Views list, right-click the new record, choose Validate, and then click Start.
6. Compile your modifications and then deploy the task UI.
7. Make sure you can access the new view while you use the responsibilities.

Modifying a Task UI to Display a Message That Is Specific to a Task Instance

In some situations, you might need to display a message that is specific to a task instance in the Inbox Items List view. For example:

- Information that helps the instance owner to distinguish between instances that include the same task name.
- Instructions that the current instance owner requires to complete the task UI.

- A timestamp.

You can modify a task UI to display a message that is specific to a task instance. Siebel CRM displays this message in the Inbox Context field in the Inbox Items List view. If the user pauses or finishes a task UI, then Siebel CRM displays information from the Instance Identifier task property in this context field.

The following conditions apply:

- You can use an expression for the Instance Identifier.
- The Instance Identifier property is limited to 200 characters in length.
- You can define the instance identifier only on a task step where you can define an output argument. For more information, see "Arguments of a Task Step" on page 54.
- . If the user clicks Pause on the first view, then Siebel CRM does not display information from the Instance Identifier.

To modify a task UI to display a message that is specific to a task instance

1. Add an output argument to a task UI step.
2. In the Multi Value Property Window, click the Property Name field for the output argument, and then pick Instance Identifier from the Property Name dialog box.
3. Define the remaining fields in the same way that you define a typical output argument.

Creating a Subtask

You create a new subtask in the same way as you create a parent task UI. The only difference is that you must make sure the value for the Is Subtask property includes a check mark. If you use the Task Wizard, then make sure the Create As a Subtask option includes a check mark. For more information, see "Creating a Subtask Step" on page 73.

Defining the Context for a Task Step

You can create a search specification on a Siebel operation step or a task view step that filters data. You define a Task Step Context object, which is a child of the Task Step object type. For more information, see *Siebel Object Types Reference*.

To define the context for a task step

1. Open the Task Editor.

   For more information, "Opening the Task Editor" on page 51.
2. In the Task Editor, choose a Siebel operation step or a task view step.
3. In the Multi Value Property Window, click the Task Step Context tab.
4. In the Multi Value Property Window, right-click the list area, and then choose New Record.

5 Enter a Name for the task step context, and then choose a Type.

If you set Type to Expression in Step 5, then enter the name of a business component in the Expression Business Component field.

6 Enter a Search Specification for the context.

CAUTION: It is recommended that you define the search specification for the Siebel operation step as efficiently as possible so that the specification matches only the smallest set of rows that are necessary to meet the business requirement. A search specification that identifies a large set of rows can severely degrade performance.

- If you set the Type field to Literal, then enter a literal value in the form of an expression. For example, enter the following text:

```
= 100
```

- If you set the Type field to Expression, then enter an expression. For example, enter the following text:

```
[Status] LIKE '*Open*'
```

The Expression Business Component evaluates the expression. For example, you might use the following search specifications for a Siebel operation step that performs a query operation:

```
"Repeatable " + timestamp()

"Iteration " + [&Iteration]
```

The Expression Builder does not examine the format. Make sure you enter an expression that includes the correct format.

7 Optional. Create a filter business component.

For more information, see "How Siebel CRM Uses a Filter Business Component" on page 195.

Example of Defining the Context for the Step of a Task UI

Figure 15 illustrates an example that includes an update operation.

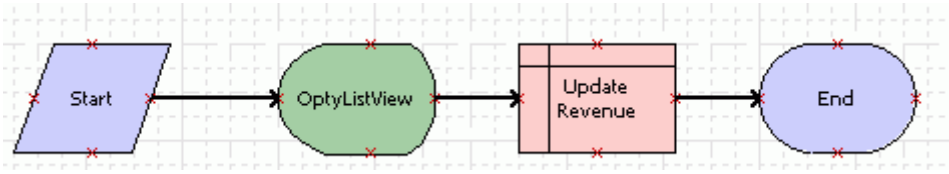


Figure 15. Example Task UI with Update Operation

In this example, the OptyListView task view step lists records in the Opportunity business component. The Siebel operation step named Update Revenue does the following:

- Performs an update operation on the Opportunity business component.

- Includes an input argument named Primary Revenue Amount of type Literal with a value of 50000.

If the user clicks Query, sets a criteria, and clicks Go, then Siebel CRM refreshes the view to display only the records that meet the query criteria. For example, the user might query for all records that includes a Lead Quality that is set to poor. If the user chooses one of these records and clicks Submit, then Siebel CRM sets the revenue amount to 50000 for the record in the Opportunity business component.

How Siebel CRM Uses a Filter Business Component

A filter business component provides the group of records where Siebel CRM performs the context search. Siebel CRM uses the following logic to identify the records that it displays:

- If the Is User Search Spec property is FALSE, then the user can only query records that Siebel CRM displays in the view, by default.
- If the Is User Search Spec property is TRUE, then the user can query all records. The Search Specification property determines the default records.

Table 19 describes example properties of a context search. In this example, the Is User Search Spec property is FALSE, so Siebel CRM displays only the records that include Opty as part of the name.

Table 19. Example Properties for a Context Search

Property	Value
Name	Opportunity
Type	Literal
Expression Business Component	Opportunity
Filter Business Component	Opportunity
Search Specification	[Name] like 'Opty*'
Is User Search Spec	FALSE

Creating a Task Event

A *task event* is an optional object type that you can define for a task UI. You can define the following task events:

- Pause
- Resume
- PreCancel
- PostCancel
- PostComplete
- Delete

For more information, see "About Event Handling" on page 228, and *Siebel Object Types Reference*.

To create a task event

1. If necessary, display the Task object hierarchy.

   For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.

2. Locate the task UI you must modify.

   For more information, see "Locating a Task UI in the Tasks List" on page 60.

   You cannot configure a subtask to reference an event. If the Is Subtask property of the task UI is set to TRUE, then you cannot add a task event to this task.

3. In the Object Explorer, expand the Task tree, and then click Task Event.
4. In the Task Events list, create a new task event:
   a. In the Name property, choose a name.
   b. Identify a business service or a workflow process to handle the event.

      For more information, see "Specifying a Business Service or a Workflow Process to Handle a Task Event" on page 196.

   c. Optional. Define input arguments and output arguments for the event.

      For more information, see "Creating Input Arguments and Output Arguments for a Task Event" on page 197.

Specifying a Business Service or a Workflow Process to Handle a Task Event

You can specify a business service or a workflow process to handle a task event.

To specify a business service or a workflow process to handle a task event

1. In the Task Events list, locate the task event you must modify.
2. Specify a business service or a workflow process to handle the task event:
   - To use a business service to handle the event, do the following:
     - Define the Business Service Name property.
     - Define the Business Service Method property.

- To use a workflow process to handle the event, choose the name of the workflow process in the Workflow Process property.

  If the Workflow Process property includes a value, then Siebel Tools ignores the Business Service Name and Business Service Method properties and the workflow process handles the event.

  If you choose Workflow Process to handle the event, then you cannot define input arguments and output arguments. To use a workflow process as a handler for a business service, you define the following properties:

Property	Value
Business Service	Workflow Process Manager
Method	RunProcess

  For more information, see "Using a Business Service Step to Call a Workflow Process" on page 199.

Creating Input Arguments and Output Arguments for a Task Event

You can create input arguments and output arguments for a task event.

Creating an Input Argument for a Task Event

This topic describes how to create an input argument for a task event that uses a business service.

To create an input argument of a task event

1. Locate the task UI you must modify.

   For more information, see "Locating a Task UI in the Tasks List" on page 60.

2. In the Object Explorer, expand the Task tree, and then click Task Event.
3. In the Task Events list, locate the task event you must modify.
4. In the Object Explorer, expand the Task Event tree, and then click Task Event IO Argument.
5. In the Task Event IO Arguments list, add a new task event IO argument.
6. In the Input/Output property, choose Input.

   Do not modify the Name property. Name is a system defined property.

7. In the Argument property, choose the name of an argument.

   The picklist displays the input arguments that are available for the business service method that Siebel Tools displays in the Business Service Method property of the parent task event. You choose an argument, and then Siebel Tools enter data in the Name property.

8 In the Type property, choose a source type.

You can choose one of the following values:

- Business Component
- Expression
- Literal
- Task Property

The type you choose identifies the source for the value of the input argument.

9 Define the remaining properties, according to the Type you chose in Step 8. Use values from the following table.

Type	Description
Business Component	Do the following work: ■ Choose a business component in the Business Component property. ■ Choose a business component field in the Business Component Field property.
Expression	Use the Value property to define an expression. Siebel CRM evaluates this expression at run time to determine the value that it uses for the input argument.
Literal	Use the Value property of the input argument to define a literal value. for the input argument, Siebel CRM.
Task Property	Choose a task property in the Property Name property.

Siebel Tools disables properties that are not applicable for the type you choose.

Creating an Output Argument for a Task Event

This topic describes how to create an output argument for a task event that uses a business service.

To create an output argument for a task event

1 Locate the task UI you must modify.

For more information, see "Locating a Task UI in the Tasks List" on page 60.

2 In the Object Explorer, expand the Task tree, and then click Task Event.

3 In the Task Events list, choose the task event you must modify.

4 In the Object Explorer, expand the Task Event tree, and then click Task Event IO Argument.

5 In the Task Event IO Arguments list, add a new task event IO argument.

6 In the Input/Output property, choose Output.

Do not modify the Name property. The Name property is a system defined property.

7 In the Type property, choose a source type.

You can choose one of the following values:

- Business Component
- Expression
- Literal
- Output Argument

The type you choose identifies the source for the value of the output argument.

8 Define the following properties, according to the Type you chose in Step 7:

- Business Component
- Expression
- Literal
- Output Argument

For more information, see "How the Type Field Affects Other Fields in the Multi Value Property Window" on page 55.

Siebel Tools disables properties that are not applicable for the type you choose.

9 In the Property Name property, choose the name of a task property.

For example, to enter the value of the output argument in the Object Id task property, choose Object Id.

Using a Business Service Step to Call a Workflow Process

You can use a business service step to call a workflow process.

To use a business service step to call a workflow process

1 Add a business service step to a task UI using values from the following table.

Property	Value
Business Service	Workflow Process Manager
Method	RunProcess

2 In the Multi Value Property Window, add an input argument to the business service step you added in Step 1 using values from the following table.

Property	Description
Input Argument	Choose ProcessName.
Type	Choose Literal.
Value	Enter the name of the workflow process.

3 Define more input arguments, as necessary.

To pass a value from a task UI to a workflow process, the input argument must use the same name as the process property of the workflow process. For more information, see "Creating the Arguments of a Task Step" on page 82 and *Siebel Business Process Framework: Workflow Guide*.

Other Options for Customizing a Task UI

This topic describes optional UI objects you can create for a task UI. It includes the following topics:

Specifying the Operations That Users Can Perform in an Applet

Siebel CRM applies some restrictions on the operations that a user can do in an applet, such as No Insert, No Update, and so on. You can modify these restrictions. If you set the EnableStandardMethods applet user property to TRUE, then the following occurs:

- Siebel CRM applies the restrictions that it defines on the applet.
- The user can use the applet to modify a record that Siebel CRM displays in a task view the same way that this user modifies this record in a predefined view. For example, the user can do a query, advance the record pointer, edit multiple records simultaneously, and so on.

The user cannot do a MergeRecord operation in a list applet on a task view even if you set the EnableStandardMethods user property to TRUE.

For more information, see "Operations That Siebel CRM Allows in Applets" on page 41.

To specify the operations that users can perform in an applet

1. Log in to Siebel Tools.
2. If necessary, display the Applet User Prop object.

   For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.
3. In Siebel Tools, in the Object Explorer, click Applet.
4. In the Applets list, query the Name property for the applet you must modify.
5. In the Object Explorer, expand the Applet tree, and then click Applet User Prop.
6. In the Applet User Properties list, create a new record using values in the following table.

Property	Value
Name	EnableStandardMethods
Value	TRUE

Creating a Task Chapter

This topic describes how to create a task chapter. For more information, see "Task Chapter" on page 38.

To create a task chapter

1. Create a task chapter:
   a. Open the task UI you must modify in the Task Editor.

      For more information, see "Using the Task Editor" on page 51
   b. Right-click the canvas, and then choose Show Chapters.

      Colors in the task UI steps disappear, leaving the task steps white.
   c. In the Multi Value Property Window, click the Chapters tab.
   d. Right-click in the list area of the Multi Value Property Window, and then choose New Record.
   e. Define a value in the Name property.
   f. Define a value for the Display Name - String Reference property.

      Siebel CRM displays this value as the task chapter title in the Siebel client. To display a custom value, do the following work:

      - Define a value in the Display Name - String Override property.
      - Leave the Display Name - String Reference property empty.

   g. Choose a color from the pop-up Color applet, and then click OK.

h Choose a Sequence.

Siebel Tools sorts the records in the Chapters tab according to the sequence number, by default. If you define a new chapter, then Siebel Tools sets the Sequence property to the next available number in the series. For example, 10, 20, 30, and so on. If you define more chapters in a task UI, then you can insert new chapters in the sequence without having to renumber the sequence. For example, you could insert 25 between 20 and 30.

If you define a chapter but you do not assign a task step to this chapter, then Siebel Tools creates an error when you validate the task UI.

2 Assign task steps to a task chapter:

a In the Task Editor, right-click a step, and then choose Assign Chapter.

To simultaneously assign multiple steps to a chapter, you can depress the shift key as you click each step that you must include in the chapter.

b Choose a chapter, and then click OK.

Siebel Tools limits the list to the task chapters that are defined in the task UI.

c Repeat Step a and Step b until you have assigned each step in the task UI to a chapter.

Although using chapters is not required, if you assign one step in a task UI to a chapter, then you must assign every step in the task to a chapter.

d Save your work.

Deleting a Task Chapter

You can delete a task chapter.

To delete a task chapter

1 In the Multi Value Property Window, click the Chapters tab.

2 Right-click the chapter you must delete, and then choose Delete Record.

Siebel Tools removes the chapter assignment from the applicable steps in the Chapters view. The steps remain but Siebel Tools does not assign them to a chapter.

Creating a Radio Button Group

The user can use the radio button group to make a choice in a task UI. The input that the radio button receives can determine the next step of a task, or it can determine the data that Siebel CRM gets and then displays in the subsequent task view. For more information, see “Radio Button Group” on page 45.

To create a radio button group

1 In Siebel Tools, create a field on the business component that supplies data for the radio button group.

2 Optional. Define a predefault value for the field you created in Step 1.

For more information, see "Defining the Default Value for a Radio Button Group" on page 203.

3 In the Object Explorer, click Pick List.

4 In the Picklists list, right-click, and then choose New Pick List Wizard.

5 Follow the prompts in the New Pick List Wizard, define the Pick List, and then choose or define the LOV.

6 Make sure the Business Component property of the new picklist is set to PickList Generic.

7 Configure the business component field you defined in Step 1 to reference the picklist:

a In the Object Explorer, click Business Component.

b In the Business Components list, query the Name property for the business component.

c In the Object Explorer, expand the Business Component tree, and then click Field.

d In the Fields list, query the Name property for the field.

e In the Picklist property, choose the new picklist.

8 In the Object Explorer, click Applet.

9 In the Applets list, query the Name property for the applet where you must add the radio button group.

Siebel CRM cannot display a radio button group in a list applet. If you define a radio button group in a list applet, then Siebel CRM displays a radio button as a text entry field in the Siebel client.

10 In the Object Explorer, expand the Applet tree, and then click Control.

11 In the Controls list, create a new control using values from the following table.

Property	Description
Name	Enter text that describes the control.
Field	Choose the field that you defined for the radio button group.
HTMLType	Choose RadioButton.

Defining the Default Value for a Radio Button Group

Setting a default value for a radio button group allows you to set the choice that Siebel CRM displays in this group. Setting the default value for a radio button group is optional. If you set a default value, then make sure you choose the default value carefully. If you do not explicitly define a default value, then Siebel CRM displays the first value that the LOV lists in alphabetic, ascending order.

To define the default value for a radio button group

1 In the Object Explorer, click Business Component.

2 In the Business Components list, query the Name property for the business component that the applet references.

3 In the Object Explorer, expand the Business Component tree, and then click Field.

4 In the Fields list, query the Name property for the field that supplies data to the radio button group.

5 In the Predefault Value property, define the default value for the radio button group.

Only one default value can exist for a radio button group.

Creating an Applet Message

An applet message allows you to combine static text and dynamic data in a message that Siebel CRM displays in the Siebel client. You can place an applet message in an applet. For more information, see "Applet Message" on page 46.

To create an applet message

1 If necessary, display the applet message and symbolic string object types, and their child object types.

For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.

2 In Siebel Tools, define a symbolic string that includes the text of the applet message:

a In the Object Explorer, click Symbolic String.

b In the Symbolic Strings list, create a new symbolic string.

In the Current String Value property, use %1, %2, and so on as placeholders for the dynamic data. Make sure you use an ascending number sequence to avoid a run-time error. For example, use %1, %2, %3, and so on.

3 In the Object Explorer, click Applet.

4 In the Applets list, query the Name property for the applet where you must define the applet message.

5 In the Object Explorer, expand the Applet tree, and then click Applet Message.

6 In the Applet Messages list, add an applet message object to the applet:

a In the Text Message property, choose the symbolic string you defined in Step 2.

b Define the other properties, as necessary.

7 In the Object Explorer, expand the Applet Message tree, and then click Applet Message Variable.

8 In the Applet Message Variables list, define one record for each value that Siebel CRM must substitute in the applet message:

a In the Value property, enter a substitution number.

For example, enter the number 1 to represent the %1 substitution you defined in Step 2.

b In the Field property, choose the business component field whose value must display for the placeholder in the message.

c Define the other properties, as necessary.

For example, the following configuration causes Siebel CRM to substitute the %1 placeholder in the applet message with the value of the Opportunity Name field:

Property	Value
Value	1
Field	Opportunity Name

9 Define the layout of the applet message.

For more information, see "Defining the Layout of the Applet Message" on page 205.

Defining the Layout of the Applet Message

This topic describes how to define the layout of the applet message.

To define the layout of the applet message

1 In the Object Explorer, click Applet.

2 In the Applets list, query the Name property for the applet where you must define the applet message.

3 In the Object Explorer, expand the Applet tree, and then click Control.

4 In the Controls list, add a new control for the applet message you created in Step 6 on page 204 using values from the following table.

Property	Description
Field	Enter the name of the applet message.
Field Type	Choose Message.

5 In the Applets list, right-click, and then choose Edit Web Layout.

6 Drag and then drop a Label control from the Palette to the applet grid layout. Place this control where Siebel CRM must display the applet message.

7 Optional. To modify the width of a control for text, do the following work:

a Position the mouse over the left edge or right edge of the control so that Siebel Tools changes the cursor to a line that includes arrows at each end of the line.

b Click and hold down the left mouse button.

c To modify the size of the control, drag the cursor.

The text in the control wraps in the edit and preview modes. Siebel CRM wraps the applet message text in the Siebel client at run time. This message includes the dynamic text that replaces the placeholder text.

8 In the Properties window, set the HTML Display Mode property to FormatData.

   Setting the HTML Display Mode property to FormatData allows line returns and spacing in the text string.

9 Optional. If you did not enter text for the message when you defined the applet message, then you can enter it now:

   a Double-click the control.

   b In the text box in the layout editor, enter the text for the message.

Writing an Independent Message

An applet message uses a symbolic string, so you cannot predict with precision how Siebel CRM renders the message at run time. You must write the message so that it forms a sentence that is grammatically correct and that is less than 250 characters in length. Do not write a message that depends on the symbolic string to concatenate the message. For example, to instruct the user to make sure data is complete, you can write the following message:

```
Make sure the information you enter is complete.
```

Do not write the following:

```
Make sure the information
```

Reusing a Standard Applet

If the applet you use in a task UI does not interact with transient data, then, as an option, you can copy and modify a standard, predefined applet instead of creating a new one. For more information, see Overview of Transient Data on page 96.

To reuse a standard applet

1 Identify a standard applet:

   a In the Siebel client, examine the Siebel application for an applet that closely matches your display requirements.

   b Locate a candidate applet.

   c Choose the Help menu, and then the About View menu item.

   d Note the value in the Applets section.

      If Siebel CRM lists more than one applet, then note the applet name that most closely matches the functionality of the candidate applet.

   e In Siebel Tools, query the Name property of the Applets list for the applet you identified in Step b.

   f In the Applets list, right-click, and then choose Edit Web Layout.

   g Verify that the layout resembles the applet you identified in Step b.'

   h Close the Web Layout Editor.

2 Copy a standard applet:

a In the Applets list, right-click, and then choose Copy Record.

Siebel Tools performs a cascade copy of the applet. You can view the status bar that Siebel Tools displays along the bottom of the Siebel Tools user interface to monitor the status of the copy operation. Siebel Tools finishes the copy, and then displays the new record in the Applets list and places the cursor in the Name property.

b In the Name property, enter a name for your custom applet that indicates how the user uses the applet.

For example, Account Entry for task UI.

c If necessary, define the Project property.

d In the Applets list, right-click the applet, and then choose Edit Web Layout.

e In the Applet Web Template window, delete unnecessary controls.

Task UI is intended to simplify the user interface, so it is recommended that you remove controls and labels that the user does not require to finish the task. Make sure you do not delete fields that the user requires to finish the task. For example, an insert operation for an opportunity requires the following fields:

- Opportunity Name
- Close Date
- Currency

f In the Applet Web Template window, reposition controls and their labels until the applet resembles the required layout for the task UI.

g Save your work, and then close the Applet Web Template window.

Hiding the Task Pane

Siebel CRM automatically displays the Task Pane, by default. Siebel CRM does the following work if you configure it to hide the Task Pane:

- Hides the Task Pane for any task UI, including any completed, paused, or cancelled task instance. This hiding persists even if the user logs out and then logs back in to the client. Siebel CRM also hides any features that it normally displays in the Task Pane, such as the Task Progress Indicator.
- Disables the shortcut key for the Task Pane. This shortcut key is Ctrl + Shift + Y.

If you configure a Siebel application to hide the Task Pane, then no users can use the Task Pane to start a task UI in this application. Other ways exist to start a task UI. For more information, see "Starting a Task UI" on page 181.

For more information, see "About the Task Pane" on page 31.

To hide the Task Pane

1 Log in to Siebel Tools.

2 Display the Application User Prop object type.

For more information, see "Displaying Object Types You Use to Develop a Task UI" on page 50.

3 In the Object Explorer, click Application.

4 In the Applications list, locate the Siebel application you must modify.

5 In the Object Explorer, expand the Application tree, and then click Application User Prop.

6 In the Application User Props list, create a new record using values from the following table.

Property	Description
Name	HideTaskPane
Value	True

7 Compile your modifications.

8 In the Siebel client, test your modifications.

Enabling the Task Progress Indicator

This topic describes how to enable the Task Progress Indicator. For more information, see "About the Task Progress Indicator" on page 35.

To enable the Task Progress Indicator

1 Use a text editor to open the configuration file (.cfg) for the Siebel application.

For example, to open the configuration file for Siebel Call Center, navigate to the following directory, and then open the uagent.cfg file:

```
D:\Siebel\81\21031\MWC\BIN\ENU
```

2 Set the EnableTaskProgress parameter in the following Task section:

```
[Task]
EnableTaskProgress = TRUE
```

If you set EnableTaskProgress to FALSE, or if it is not defined, then Siebel Task UI does not display the Task Progress Indicator.

3 Log in to the Siebel client.

4 Optional. Configure Siebel Task UI to display the task percentage:

a Navigate to the Administration - Business Process screen, and then the Task Deployment view.

b Query the Name column for the task UI you must modify.

c Click Compute Percentage.

5 Navigate to a view that includes the task UI that must display the Task Progress Indicator.

6 Verify that Siebel Task UI displays the Task Progress Indicator.

11 Guidelines and Techniques for Developing a Task UI

This chapter describes guidelines and techniques for developing a task UI. It includes the following topics:

Guidelines for Developing a Task UI

This topic describes guidelines for developing a task UI. It includes the following topics:

Guidelines for Organizing the Task Flow

Use the following guidelines when you organize the task flow:

- **Envision a simple task UI.** Envision the display of information as a series of linked pages or steps when you design a task UI. Although in many situations you can configure a task step in a single page, it is not required. If the step is complex, then you can separate it into multiple pages. Simplicity is an important design principle for a task UI.
- **Simplify task complexity.** Make sure the task UI is self contained and is a single point-to-point flow. During the design phase, consider the number of steps and decisions. If necessary, separate one task into multiple tasks.
- **Chunk the task UI.** Separate the task UI into a number of logical chunks. A task UI must guide the user through a task that is easy to follow and review. A task that is too long might disorient the user. A rule of thumb is to create chunks that include between five and seven items.
- **Use the Chapter feature.** You can configure task UI to use the Chapter feature to assist the user in understanding the overall job task and to monitor completion of the job task. The chapter feature provides the user with an outline of the task UI. This feature helps the user understand the work that the user already performed and how much work the user must still perform at a point in the overall task. The individual steps in a chapter can vary depending on choices that the user makes.

- **Organize the task UI around logical commit points of the task transaction.** For more information, see "About Task Transaction" on page 225.

Guidelines for Designing Task Functionality

Use the following guidelines when you design the functionality of the task UI:

- **Design for the user profile.** Consider the experience and the job role of the user. Consider questions such as job turnover, and computer aptitude.
- **Design for how frequently the user performs the task UI.** How frequently the user performs the task can affect how the task UI is designed. A task that the user performs infrequently might require more guidance because the user is not provided an opportunity to internalize expertise.
- **Enforce forward navigation.** You can enforce forward navigation but allow for review and editing of work that the user performs in the task UI. Design your task UI so that the task guides the user forward through the task in a way that allows the user to edit and review portions of the task that the user has finished. Navigation that requires the user to click Previous is acceptable, but the user must not be required to repeatedly click Previous through a long list of views.
- **Allow the user to enter record data across multiple views.** Allowing the user to enter record data across multiple views can help to organize the fields that the user must use to create a complete record that is very complex.
- **Allow the user to define a query.** Use a separate view for each of the following:
  - Use one view for the query.
  - Use another view for the query results.
- **Use the form view and list view correctly.** Do the following:
  - If each record includes a large number of required fields, then loop through a form view.
  - If each record includes a small number of required fields, then use a list view.

  This technique helps to minimize horizontal and vertical scrolling of the list view.
- **Use the same task view in multiple view steps in a task UI.**
- **Choose a display style for the step of the current task pane.** Make sure Siebel CRM uses the correct style according to the complexity of the task UI. Depending on the complexity and length of a task UI, you can configure Siebel CRM to do one of the following:
  - Display all steps simultaneously.
  - Display only a subset of steps simultaneously.
- **Avoid using a link in a task UI.** Remain in the task UI. Avoid a script or run-time event that pauses the task and then leaves the task.
- **Include a review page.** A review page allows the user to review and modify data before Siebel CRM saves this data. You can place a review page at the end of a task UI. For a long task, you can place a review page at multiple locations throughout the task.

- **Configure Siebel CRM to perform an operation that occurs outside of the Siebel database only if the user clicks Submit.** The task controller does not roll back modifications that Siebel CRM makes outside of the Siebel database, such as with saving an attachment. It is recommended that you configure Siebel CRM to perform this type of operation only after the user clicks Submit. If the user steps back through the task UI or cancels the task, then you avoid having Siebel CRM create an orphan file.

Guidelines for Using a Script to Start a Task UI

If you use a script to start a task UI, then use the following guidelines:

- Make sure an active UI context exists before the user starts a task UI.
- Do not start a task UI from the middle of a run-time event, such as the WriteRecord event.
- Do not create a task UI from a script and then immediately pause the task to the inbox. The Workflow Task step can perform this logic, but you cannot use it through a script. If you define a task UI to start from a script, then Siebel CRM starts it immediately when the script runs and then displays the first view in the Siebel client.
- Make sure a business component record is available to the task UI. A script must only call a task when the business component includes at least one record.

Only limited support exists for starting a task UI from a script.

Guidelines for Designing User Interface Elements

This topic describes guidelines that you can use to design user interface elements.

Guidelines for Designing a Task View

Use the following guidelines when you design a task view:

- **Avoid redundancy.** For example, avoid using multiple buttons that perform the same function. Displaying more than one button that performs the same function increases uncertainty and unnecessarily increases the complexity of the view.
- **Focus the content of the task UI.** Configure task UI to display only data that is specific to the current step. This technique helps to simplify the UI.
- **Focus feedback that the task UI provides.** The current task pane provides feedback to the user about the progress that the user makes in a task instance. Make sure this feedback is succinct and relates only to the job task that the user is currently performing.
- **Use an applet message.** If you configure task UI to combine dynamic data, such as with a business component and a transient business component, then you can use an applet message. For more information, see Overview of Transient Data on page 96.
- **Use a radio button or picklist.** You can configure task UI to use a radio button or picklist to help the user make a decision.
- **Override default method disabling.** In most situations, you must disable a default method. The following are common exceptions:

  - A New button on a list applet for nonloop operation.
  - A Query button that constrains data.
- **Avoid using a button that navigates the user to a different view.** A button or a script must not navigate the user to a different view. If this situation occurs, then the task UI pauses. A service that calls a server component must not assume that it works on task data that Siebel CRM has not saved.
- **Avoid modifying the look and feel of a template.** You typical do not need to modify a template. You can use the IsInTask, target Task UI Service (SWE) to determine whether Siebel CRM must run a section of a template only in task mode.
- **Avoid using a search specification on an applet.** Instead of using a search specification on an applet, use a search specification on a task view step. This technique helps to avoid confusion if you reuse an applet across task views.
- **Use a consistent approach to page design.** For example, when designing a page, do the following:
  - Begin with the page type. For example, Overview, Work, Review, or Summary.
  - Proceed with the page title.
  - Finish completing the task design.

For more information, see "About the Task View" on page 39.

Guidelines for Using a Form Applet and a List Applet

Use a list applet to do the following:

- Display a list of items.
- Allow the user to create a list of items.

You might find you do not use a list view in a task UI as often as you use a list view in the standard UI. Apply the following guidelines:

- Use a list applet for a list or summary view.
- If the user must modify data in a record, then use a form applet. The form applet focuses user attention on a single record.
- Use a Siebel operation step to allow the user to create a new record. Use a view step that includes a form applet that displays a new record that includes empty fields that the user finishes. Siebel CRM typically uses a list view that allows the user to click New to create a new record in the standard UI. This configuration is not necessary in a task UI, particularly if the user only creates a single record.

Guidelines for Reusing Views and Applets

To enforce the simpler layout requirements of a task UI, it is recommended that you create a new applet and view or modify a predefined view and applet. You can use the CSSSWEFrame template to avoid performance problems that might occur with specialized code when you create a new applet.

Guidelines for Designing Buttons and Menus

If you design buttons and menus, then use the following guidelines:

- **Do not configure Siebel CRM to open a task UI that creates a new record from outside the task pane.** If Siebel CRM does not yet display a record, then you might need to disable a button that opens a task. A button or menu requires at least one record from the business component that the task UI references. Siebel CRM must reference this business component when it creates a task instance. If the business component includes no records, then Siebel CRM cannot reference it and cannot create a task instance, and an error occurs.
- **Make sure you deploy the task UI before you add a button or menu to start the task.** You must activate the task before the user can use a button or menu item to start it. If you do not activate it, then Siebel CRM displays a run-time error in the Siebel client.
- **Do not create menus and buttons that perform the same function.** For example, avoid including the top and bottom task playbar applets on the same view. Instead, include only the top task playbar applet or only the bottom task playbar applet.

Guidelines for Using the Default Focus

Include a *default focus* that prompts the user to perform an action. For example:

- **Place the default focus on the Next button.** On an overview page, the user typically clicks Next.
- **Define the cursor to display in the first editable field.** On a content page, the user typically edits the first editable field.

Guidelines for Designing the Structure and Content of a Page

This topic describes guidelines for designing the structure and content of a page.

Guidelines for Designing the Structure of a Page

Use the following guidelines to design the structure of a page that displays in a task UI:

- **Include a title for the task UI that describes the goal of the task.** For example: Create a New Account.
- **Include a page title that is concise.** Each page title can be an explicit statement of the purpose of the page.
- **Include a page explanation.** Briefly elaborate on the purpose of the page.

Guidelines for Designing the Content of a Page

Table 20 describes types of content that Siebel CRM can display on a page and their recommended usage. Make sure the page only includes content that supports the purpose of the page.

Table 20. Types of Content That Siebel CRM Can Display on a Page

Page Content Type	Recommended Usage
Overview Page	Displayed at the start of a task UI. Provides an overview of the goal of the task UI that can help the user to determine to proceed with the task. For example, the overview page in a driver license renewal task can indicate that completing the task requires a social security number and a driver license number.
Step Page	Displayed throughout a task UI. Includes only data that is relevant or meaningful to the current task UI step.
Review Page	Displayed before a commit step. Allows the user to review data before Siebel CRM saves it. Allows the user to review, edit, and navigate in the task UI to make adjustments.
Summary Page	Displayed at the end of a task UI. Provides a log and confirmation of work that the user performed. This page cannot include editable data fields because it is a confirmation of information that Siebel CRM already saved to the Siebel database.

Guidelines for User Interface Styles to Avoid

Avoid using the following items:

- **Avoid using a drilldown.** For example, consider the Contact screen in the standard UI that displays a list of contacts. If the user clicks the Account Name field in the list view of the Contact screen, then Siebel CRM navigates the user to the Account screen that displays the chosen account. This functionality is known as a *drilldown*.

  Siebel CRM does not allow a drilldown in a task UI. Using a drilldown requires Siebel CRM to pause the task so that it can navigate the user to the standard UI view that the drilldown references.

- **Avoid using a button on an applet.**
- **Avoid a design style that uses a vertical or a horizontal scroll bar.** The scroll bar disorients the user.
- **Avoid including too many applets in a view.** Keep the design of each view as simple as possible.
- **Avoid redundancy.** Avoid including multiple buttons or applets that perform the same function.

Guidelines for a Multilingual Task UI

You can use values that are independent of any given language to design a task UI that Siebel CRM can run in multiple languages. You must pay close attention to the following items:

- Display names that use symbolic strings. Do not use overrides.
- Use LIC (language independent code) instead of a display value.
- Values for the Task Property.
- Field values for a business component and transient business component. For more information, see Overview of Transient Data on page 96.
- Siebel operation steps.
- Conditions.
- LOVs.
- Multilingual LOVs (MLOVs). If a LOV type is Multilingual, then Siebel CRM uses language independent code to store the data that the LOV binds.
- Marked for translation.

About Literal Values in a Multilingual Environment

A literal value depends on a language, so a task UI cannot reference a literal value in a dynamic picklist and function correctly in a multilingual environment. A task that references a literal value is language dependent and you must not configure Siebel CRM to run it in a multilingual environment. A literal value in a dynamic picklist can include content that the user creates. These values reference content rather than metadata or an LOV, so they are variable and the task cannot reliably interpret them. For more information, see *Siebel Global Deployment Guide*.

Guidelines for Using a Business Service

A business service allows you to run a predefined or custom action in a task UI. Code that Siebel CRM calls synchronously or asynchronously cannot view data that Siebel CRM has not saved from temporary storage to base tables. To avoid this problem, you can add a commit step before you start a server task for a server component.

Using Script with a Business Service

You can use Siebel VB or Siebel eScript to create your own custom business service that Siebel CRM calls from a task UI. A server script in a business service operates on the temporary instances of a business component and updates the S_TU_LOG table. It does not update the base tables.

CAUTION: A task UI cannot call a business service that includes browser script. If a task UI calls a business service that includes browser script, and if that task runs on the Siebel Server, then the business service fails.

Techniques to Improve the Usability of a Task UI

This topic describes techniques you can use to improve the usability of a task UI. It includes the following topics:

Split View Technique

A business component record can include a large number of fields. Dependencies might exist between fields in this record that require the user to enter information in the fields in a specific order. A view that displays all business component fields simultaneously is difficult to use. A task UI can guide the user through completing these fields.

You can use a form applet that references the same business component to split data entry for a single record into multiple views. Navigating backward and forward between these views displays different fields of the same record while preserving the field values that the user enters.

It is recommended to keep fields that depend on one another in the same view. This technique avoids a problem if modifying a field value in one view causes Siebel CRM to modify a field value in another view. For example, through a pick map or the On Field Update user property.

Example of the Split View Technique

Assume a task UI includes account data that includes two hundred single value fields and eighty multivalue fields in the Account business component. This task UI uses the following views to display these account fields:

- The first view displays identification data, such as Name, Location, DUNS, and CSN.
- The second view displays classification data, such as Account Type, Industry, Territory.
- The third view displays communication data, such as Primary Address, Phone Number, Fax Number, and URL.

Optional View Technique

If a task UI must display data that the user can review, and if this data is optional, then you can make the view optional.

Using the Optional View Technique

This topic describes how to use the optional view technique.

To use the optional view technique

1. Open the Task Editor for the task UI you must modify.

   For more information, "Opening the Task Editor" on page 51.

2. Add a Siebel operation step using values from the following table:

Property	Value
Operation	Query

3. Optional. In the Multi Value Property Window, click the Output Arguments tab, and then add an output argument using values from the following table.

Property	Value
Type	Output Argument
Output Argument	NumAffRows For more information, see "Arguments of a Task Step" on page 54.

   For more information, see "Creating an Output Argument on a Task Step" on page 82.

4. Add a Decision step:

   a. Place this decision step immediately after the step you added in Step 2.

   b. Add a branch using values from the following table.

Property	Value
Type	Condition

   c. In the conditional branch, test that the value of the NumAffRows task property is a literal value that equals 0.

d Add another branch using values from the following table.

Property	Value
Type	Default

5 Add a view step immediately downstream of the decision step.

This view step displays the optional data.

6 To bypass the optional view, use the condition branch.

Example of the Optional View Technique

An activity field can include instructions. If no instructions exist, then the task UI can skip the view that displays them.

Mixed View Technique

If an applet references a single business component, and if the boundary of the business component is not intuitive or obvious to the user, then you can create a task UI that displays a logical set of data in a view. You can configure Siebel CRM to combine data from multiple business components into a single task view step. A task can include more than one applet, so the user does not need to worry about boundaries between business components.

Example of the Mixed View Technique

A view displays a list of items in an order and asks if the user must add a new item. Siebel CRM uses a single applet to display the list of ordered items according to the Order Item business component. A second applet includes the following items:

- A question that Siebel CRM displays as an applet message.
- The answer to the question, stored as a field in a transient business component and displayed as a radio button. For more information, see Overview of Transient Data on page 96.

Mixed Applet Technique

If a task UI must display data from several business components in a single applet, then you can create a task UI that uses a transient business component to hold data for the mixed applet. The task UI can save the data to a nontransient business component. The task UI saves this data immediately following the view that includes the mixed applet. For more information, see Overview of Transient Data on page 96.

Example of the Mixed Applet Technique

The predefined CMEREF - Activation Order task UI is an example of the mixed applet technique. It asks for the following information in a single applet:

- Account name
- Customer first name
- Customer last name

At a later point in this task UI, it uses this data to query an existing account and contact. If it does not find a record, then it does one the following:

- Creates a new account or a new contact
- Creates a new account and a new contact

Business Component Method Technique

The standard UI requires the user to click a button or to choose a menu item to call a section of business logic. This business logic typically calls a business component method and is not readily available as a business service method. The problem with this kind of user experience is that the user must possess knowledge about the following items:

- The user must click a button to run the logic
- The user must understand the result of clicking the button
- The user must understand the sequence to use to click multiple buttons

To avoid this complexity, task UI provides the following capabilities:

- Display unambiguous choices in the Siebel client
- Describe the result of clicking a button
- Provide backward navigation
- Enforce the sequence to use to click multiple buttons

You can provide the user with multiple choices in a task UI, one choice at a time:

1. The user makes a choice, and then clicks Next on the task playbar applet.
2. The task UI calls the business logic and in the correct order.
3. If an exception occurs, then the task UI displays an error dialog.
4. If an exception does not occur, then the task UI can display the next view.

Using the Business Component Method Technique

This topic describes how to use the business component method technique.

To use the business component method technique

1 If an adapter business service does not exist, then create one using values from the following table.

Property	Value
Class	CSSBCAdapterSvc

2 Make sure the value of the Business Component user property specifies the business component whose method or methods it is adapting.

3 Define a method with the same name as the business component method that Siebel CRM must call from the task UI.

4 Call the business component method from a business service step in the task UI.

This business service step must reside downstream of a task view step so that the Next step flows to the business service step and calls the business component method.

If you enable backward navigation to the selector view, and if this view displays an option that logically negates the underlying business logic that Siebel CRM performs, then you might need to adjust the flow of your business process so that it is compatible with backward navigation and the business logic.

Example of the Business Component Method Technique

The predefined Field Activity Invoicing task UI displays an option to automatically create an Invoice. The user clicks AutoInvoice in the Invoice applet to create this invoice in the standard UI. The predefined task calls an adapter business service method after the selector view. This task UI eliminates the requirement to use the AutoInvoice button in the Invoice applet.

Refine Query Technique

It is recommended that you define a query for a task UI, typically as a search specification in a task view step. However, sometimes it is not possible to know at design time the exact search specification that the user requires at run time, and Siebel CRM might display a large number of records when the search specification for the task UI runs. You can configure a task UI that uses query-by-example to allow the user to refine this query.

Siebel CRM uses the following logic in the refine query technique:

1 The task view step includes a broad search specification.

2 The task controller sets this search specification as an anonymous search specification on the business component, using the & (ampersand) prefix to substitute task properties that the search specification references.

3 The task view includes a Refine Query button that calls the Refine Query method on the business component. This method uses the search specification set in Step 1 but allows the user to refine the search criteria.

4 The user refines the search specification, clicks Execute Query in the task view, and then Siebel CRM displays the task UI only in query mode.

5 Siebel CRM uses the refined search specification set in Step 3 to run the query.

If Siebel CRM does not display the fields that the search specification in Step 1 references, then the user cannot remove this query even if the user is required to remove it.

Example of the Refine Query Technique

The predefined Create Order task UI includes a task view step that allows the user to choose a number of products. To search products, the user can use the customer zip code, but this query can return hundreds of products. The Create Order task UI allows the user to use the name to refine the query. For example:

```
[Name] LIKE '*300 minutes*'
```

This task UI does not display the zip code, so the user can only narrow the original search specification and cannot broaden it. The predefined Create Order task UI resides in the Siebel Communications application.

Commit Interim Data Technique

A task UI might require a long time to finish but the business requirement dictates that data that the task updated be visible to other users as soon as possible.

Using the Commit Interim Data Technique

This topic describes how to use the commit interim data technique.

To use the commit interim data technique

1 Open the Task Editor for the task UI you must modify.

For more information, “Opening the Task Editor” on page 51.

2 Add a view step that reviews data that the user enters in the task.

3 At the point in the task UI where data that the user entered must become visible to other users, set the Forward Button Type property to Submit.

For more information, see “Validation with Forward Navigation in the Task Playbar” on page 42.

4 Add a commit step immediately downstream of the view step you added in Step 2.

Example of the Commit Interim Data Technique

In the predefined Field Activity Main Task task UI, if the user updates the Status field, then Siebel CRM saves it to the Siebel database so that other users can view it, regardless of whether or not the task instance finishes. The following logic is involved in this example:

1. A Siebel operation step updates the activity status.
2. A task view step includes a Submit button. This task view step is immediately downstream of the Siebel operation step described in Step 1.
3. A commit step saves data to the Siebel database. This commit step is immediately downstream of the step described in Step 2.

This behavior makes sure the user understands that Siebel CRM saves the modifications, and allows the user to step back through the task UI and choose a different status before committing modifications that are visible to other users.

Other Usability Techniques

Table 21 describes other usability techniques you can use.

Table 21. Other Usability Techniques

Usability Technique	Description
Use the Defer Write Record property.	For more information, see "Using the Defer Write Record Property of a Siebel Operation Step Instead of a Transient Business Component" on page 104.
Place all required fields on a single view.	Placing all required fields on a single view helps the user to finish the transaction. If necessary, you can use a transient business component to store interim data. For more information, see Overview of Transient Data on page 96.
Avoid using too many pop-up elements.	Too many pop-up elements can be distracting.
Use a list view as a picklist.	You can use a list view as a picklist to allow the user to pick records.
Use the selector.	The Selector forces the user to choose between several options that are available.
Use the hierarchical selector.	The Hierarchical Selector forces the user to make several mutually dependent decisions. Make sure you avoid cascading a series of selector views with only relevant options.
Push data to the Customer Dashboard.	If a task UI captures customer information that is useful in the customer dashboard, then configure Siebel CRM to push this data to the Customer Dashboard.

A Reference Materials for Siebel Task UI

This appendix describes reference information for Siebel Task UI. It includes the following topics:

Business Component Fields That a Task UI Can Modify

A task UI cannot modify the following types of business component fields:

- Fields that use a multivalue group
- Calculated fields

A task UI can do the following for a field that Siebel CRM defines as read-only at the business component level:

- Use another field or a user property to read it.
- Cannot modify it.

Updating a Field That Uses a Multi Value Group

To update a field that uses a multivalue group, you can define a business component for this field, and then use Siebel Tools to link the business component to the object. For example, the Account Team field uses a multivalue group, so you cannot use the Account business component to update it. You can create a business component named Account Team and then use Oracle's Siebel Tools to reference it to the Account business object. You can then use a Siebel operation step to update the Account Team business component. For more information, see *Configuring Siebel Business Applications*.

About Task Transaction

This topic describes some of the features of task transaction.

Task transaction is a feature that allows you to configure Siebel CRM to keep data for a task instance separate from the base tables until it explicitly saves the task data to the Siebel database. Task transaction allows data in a task instance to last for an arbitrarily long amount of time while maintaining reliability, performance, and scalability. Task transaction includes the following features:

- Task transaction data can persist only for a moment or it can last for days or months.
- A task transaction that persists for a long time is typically an inbox record.
- Task transaction data can span multiple user sessions, process boundaries, and database or application server restarts.
- The Cancel operation stops a task transaction and rolls back the modifications that Siebel CRM saved before cancellation.
- You can turn task transaction off at the business component level and at the task UI level. For more information, see "Disabling Task Transactions" on page 158.
- Task transaction is dependent on a feature that resides on the Siebel Server that is external to the underlying Relational Database Management System (RDBMS).

For more information, see "Configuring Siebel CRM to Resolve Task Transaction Conflicts" on page 178.

Atomicity

Siebel CRM can save as a group the operations that it performs in a task UI or it can stop them. Atomicity is traditionally achieved using a transactional feature that is embedded in the RDBMS. However, this feature is not appropriate for a task transaction because it is limited to a single database connection that can result in the following problems:

- Limits the duration of the transaction
- Jeopardizes reliability of the Siebel system, performance, and scalability

Isolation

A modification that a user makes in a task transaction is visible only in this task transaction until Siebel CRM saves it. For example, an insert, update, or delete operation can perform a save operation. If Siebel CRM saves a modification that resides outside of the task transaction, then this modification is visible in the task transaction as soon as Siebel CRM queries the modified record again.

Transparent Storage

Siebel CRM dedicates a set of generic tables to store data modifications in the task transaction. It maps columns in the generic tables to columns in the Siebel database tables. This configuration hides the complexity of the storage details from the person who develops the task UI. However, you must configure Siebel CRM to clean up storage of a task transaction after it saves and ends this transaction. For more information, see "Removing Temporary Data After a Task UI Finishes" on page 177.

Transparent Data Retrieval

Siebel CRM merges the data that resides in task transaction storage with the data that resides in the Siebel database tables, and with complete support for data filtering. A search specification is an example of data filtering. Siebel CRM supports declarative data ordering only for a hierarchical business component. A sort specification is an example of declarative data ordering. For other configurations, ordering works as expected only when Siebel CRM does not modify data that resides in the task transaction.

Sharable Temporary Data

The Object Manager stores the data that the user creates or modifies during a task instance. It stores this data in temporary storage in a special database table but does not save it to the base tables until the user finishes the task or reaches a commit step. Sharable temporary data can include temporary data that spans multiple server components. A thread that is separate from the initiating application can pass the task instance ID to another server component, and then return the information from this other server component back to the task.

Siebel CRM starts and shares temporary storage in the following way:

1. If the Server Request Broker detects that the current Object Manager is running a task transaction, then it passes the task transaction ID to the server component that Siebel CRM called.
2. If the Object Manager in the server component detects the task transaction ID, then the Object Manager uses this ID to start the same task transaction. As a result, the session for the Object Manager gains access to the data in this task transaction. The Object Manager ends the task transaction before the server component thread ends.
3. The task controller uses a locking feature to handle the concurrency issue in the server component. It makes sure that no more than one transaction for each task UI is active at a time, regardless of how many threads Siebel CRM runs in the server component.

This solution can handle multiple server calls. For example, if a task UI calls a server component, and this server component then calls another server component, then these server components possess access to the same task transaction.

Example with Assignment Manager

Assume that a task UI must schedule a job. It does the following work to evaluate the skills and availability that an individual possesses, and then to assign and schedule the job:

1. A call center representative runs a task UI To handle a service request while on a phone call with a customer.
2. Siebel CRM passes the task instance ID of the task UI that the user runs to the Assignment Manager server component.
3. The Assignment Manager server component determines if a field service engineer is available.
4. The Assignment Manager server component returns this information to the task UI that the call center representative is running.

How Siebel CRM Shares Temporary Storage

Siebel CRM disables sharing of temporary storage, by default. To make this sharing available, you must define the TASKTEMPSTORAGESHARING task property with a data type of String. If you do not define this property, then Siebel CRM displays an error that indicates that it cannot access temporary storage.

An asynchronous call to the Server Request Manager cannot access temporary storage data even if sharing of temporary storage data is available.

Task Transaction and Task Instance

If the user, or if a long-running workflow process starts a task UI, then Siebel CRM creates a new task instance. The relationship that exists between a task instance and a task UI is similar to the relationship that exists between an object and a class in an object oriented programming language. Each task instance includes a state that consists of task properties and navigation history. Siebel CRM typically shares data that resides in the business object that the task references, except for data that resides in a transient business component, which is unique to a specific task instance. For more information, see Overview of Transient Data on page 96.

The duration of a task instance can coincide with the duration of the task transaction. However, the task instance might not use a task transaction, or it might use a series of task transactions through the use of intermediate commit steps.

A paused task UI is a task instance that Siebel CRM stores and that the user can resume from the inbox.

About Event Handling

A *task event* is a type of event that allows you to define the processing that Siebel CRM does when an operation in a task UI calls an event. The following operations support event handling:

- Pause
- Resume
- Cancel
- Delete
- Complete

If Siebel CRM starts an operation in a task UI, then it calls a task event to handle the operation. The event handler for each operation performs the required action according to the semantics of each operation. A *preevent handler* is a type of handler that runs before Siebel CRM performs the operation for the handler, such as Resume, Pause, or Delete. Most handlers are preevent handlers.

For example, the Cancel operation includes two event handlers: a preevent handler and a postevent handler:

- The PreCancel event handler runs when the user cancels a task UI and before the temporary storage data rolls back.

- The PostCancel event handler starts after the PreCancel event handler runs and after Siebel CRM rolls back the temporary storage after the task finishes. For example, if the user clicks Finish or Submit, then the PostComplete event handler starts.

Siebel CRM does the following work to run an event handler that starts before the user starts the operation:

- If the event handler succeeds, then the task controller finishes the operation that the user started.
- If the event handler fails, then Siebel CRM stops the operation that the user started and returns the error to the object that called the event.

For more information, see "Creating a Task Event" on page 195 and *Siebel Object Interfaces Reference*.

How Siebel CRM Handles an Event That Occurs in a Subtask

You cannot configure a subtask to reference an event handler. If a task UI calls a task event when the user is in a subtask, then Siebel CRM sends the task event to the parent task UI and runs the corresponding event handler that the parent task references. It runs this event handler in the context of the parent task.

Operations That Call an Event Handler

This topic describes operations that call an event handler.

Pause Operation

The Pause operation saves the following information in the Siebel database:

- Task state and task instance.
- An inbox item. The user can use this inbox item to restart the task instance.

Resume Operation

The Resume operation restores the state and history of a task instance and resumes a paused task instance. The user can click a Task inbox item in the Inbox Items List view to resume a task instance.

Cancel Operation

The Cancel operation stops the task transaction and rolls back the modifications that Siebel CRM saved before the user cancelled this task instance.

Events That the Cancel Operation Uses

Table 22 describes handling for the PreCancel and PostCancel events.

Table 22. Handling for the PreCancel and PostCancel Events

Event Type	Description	Usage
PreCancel	Note the following: ■ The PreCanncel event occurs in the context of the task UI, before the rollback. ■ This event handler can compensate for a modification that occurs in an external system. ■ The Cancel operation calls a PreCancel event before Siebel CRM rolls back temporary storage or before it deletes the task instance. ■ If the user cancels a task, then the PreCancel event handler starts before Siebel CRM rolls back the temporary storage of the Object Manager. ■ If the task includes a parent flow, then the parent flow receives an Abort status from the task. ■ Rollback does not occur before this event, so it can access the context data of a task. The context data includes the field values that existed before the rollback occurred that is part of the Cancel operation. ■ If an error occurs, then the task UI remains on the same view and displays an error in the Siebel client.	You can configure Siebel CRM to use the Pre Cancel event handler in the following situations: ■ If the handler fails, then the Cancel event must quit. ■ An external system must perform some action.

Table 22. Handling for the PreCancel and PostCancel Events

Event Type	Description	Usage
PostCancel	Note the following: ■ The PostCancel event occurs after the PreCancel event finishes, outside of the context of the task UI, and after the rollback. ■ You can use the PostCancel event to compensate for the temporary storage data of the Object Manager that Siebel CRM saves to the Siebel database during a commit step. ■ The Cancel operation calls a PostCancel event after Siebel CRM rolls back the temporary storage due to the Cancel operation. The PostCancel event does not require access to temporary storage or the task context. ■ If an error occurs, then Siebel CRM navigates the user to the view that it displayed after the task cancelled or finished. It displays the next standard view regardless of whether the handler succeeds or fails.	You can configure Siebel CRM to use the Post Cancel event handler in the following situations: ■ If the handler fails, then the Cancel event must not stop. ■ Access to temporary storage is not required.

Delete Operation

The Delete operation removes a task instance from the following items:

- Inbox
- Siebel database

If a parent task starts a task UI, then the subtask sends the parent task an Abort status. The task utility service creates this event.

Complete Operation

The Complete operation ends the task instance in the following situations:

- The user clicks Finish or Submit.
- Siebel CRM finishes the task instance automatically.

If the task finishes, then Siebel CRM starts the PostComplete event.

PostComplete Operation

The PostComplete operation allows the task UI to do more work after it finishes running. The following examples describe how Siebel CRM uses the PostComplete operation:

- A user runs a task UI that updates an account record. If the user submits the modifications to the account, then Siebel CRM passes the modifications and information about who modified the record as inputs to a business service. This configuration creates an audit trail on the account record.
- A task instance finishes running and, after submittal, a workflow process must run. This requirement makes sure no unwanted data exists. For example, a user can follow multiple paths in a task UI and create records in these paths. The user can navigate backward in the task UI and forward again down other paths. In this situation, a cleanup process that occurs at the end of the task session can eliminate unwanted data.

In these situations, it is important that the task instance finishes. The PostComplete event handler can handle a failure that occurs when Siebel CRM creates an audit trail or during cleanup. Siebel CRM does not abort this task instance.

About Event Logging

The event log collects information about different operations that occur while a task instance runs. It can print information that you can analyze. For more information, see "Modifying How Siebel CRM Logs Data During Testing" on page 155.

Table 23 describes the following information:

- Event types that are related to Siebel Task UI that Siebel CRM displays in the Event Configuration View.
- Information about the events that Siebel CRM traces.
- Information symbols that Siebel CRM displays in the log files to identify the events. Siebel Task UI shares some event symbols with Siebel Workflow.
- The Log Level column indicates the minimum level that Siebel CRM uses to turn on tracing.

CAUTION: Setting a trace level above the default parameter affects performance. You must reset the trace level to the default parameter after you finish troubleshooting.

Table 23. Event Logging for Siebel Task UI

Event Type	Level	Symbol	Information Symbol
Workflow Definition Loading	3	DfnLoad	**Step.** Steps and branch names that Siebel CRM loaded.
	5	DfnLoad	**Chapter.** Chapter names that Siebel CRM loaded.

Table 23. Event Logging for Siebel Task UI

Event Type	Level	Symbol	Information Symbol
Task Execution	3	TskExec	**Oper.** Includes information about various operations that the user performs during the duration of a task instance. Example operations include create, delete, pause, restore, instantiate, or cancel a task instance.
	5	TskExec	Siebel CRM includes the following symbols: ■ **TskState.** Details about the state transition that occurs during the duration of a task instance. ■ **TskInbox.** Internal details about the interactions that occur for the task instance with the Universal Inbox.
Task Navigation	3	TskNav	**Oper.** Includes information about the navigation operations that the user performed in a task instance.
	4	TskNav	**NavPath.** Includes information about modifications that occur to backward and forward navigation. Includes logs if Siebel CRM inserts a new frame or deletes a frame from backward or forward stacks.
	5	TskNav	**DumpStack.** Dumps frames in the forward and backward navigation stacks after Siebel CRM restores or resumes a task instance.
Workflow Process Execution	4	PrcExec	**Create.** Includes information about the creation of a new task instance.

Table 23. Event Logging for Siebel Task UI

Event Type	Level	Symbol	Information Symbol
Workflow Step Execution	4	StpExec	Includes information about a task step and input argument or output arguments of the task step. The following symbols are included: ■ **Cond.** Information about how Siebel CRM evaluates a branch condition. ■ **Create.** Information about how Siebel CRM creates a step instance. ■ **End.** Information about Siebel CRM completes a step instance. ■ **Task.** Information about calling a business service. ■ **TaskArg**. Information about arguments to a business service.
Task Presentation Information	4	TskPresInfo	Includes information that the task controller provides to an external component. SWE is an example of an external component. **ViewInfo.** Information about the entire structure of the task view, including the following items: ■ View name ■ Playbar ■ Current task pane ■ Records that are pending validation

About Task Metrics

A *task metric* is a type of metric that collects and stores data about a task UI that Siebel CRM regularly saves to a data warehouse. You can use an online analytical processing (OLAP) tool, such as Oracle Business Intelligence, to analyze this data. You can collect timestamp metrics and property metrics. For more information, see "Modifying How Siebel CRM Logs Data During Testing" on page 155.

Timestamp Metrics

A *timestamp metric* is a type of metric that stores timestamps for a task UI. Siebel CRM requires the time when an event occurs so that it can measure the timeliness of a business process. For example, Siebel CRM requires the time a task starts and ends so that it can determine the amount of time that a user spends on a task.

If task metrics are turned on when you deploy a task UI, then Siebel CRM stores the following timestamps for the task:

- Task Started

- Task Paused
- Task Resumed
- Task Cancelled
- Task Completed

The collection and persistence of these timestamps incur a minimal and finite performance and scalability overhead. Siebel CRM does not collect timestamp metrics at the step level because the affect on performance is not acceptable.

Property Metrics

A *property metric* is a type of metric that stores metrics for a task property. It allows you to collect data about a task UI whose source is a task property. You can define a property metric to satisfy the requirements to measure business performance. The following tables store data for property metrics:

- S_WFA_ANLY_INFO
- S_WFA_ANLY_PROP
- S_WFA_ANLY_TS
- S_WFR_PROC_MTRC

Glossary

business service step

A type of task step that calls a business service. Allows you to call a business service that runs a predefined or custom action in a task UI.

business task

A logical unit of work that the user must accomplish. For example, filing an expense report or entering a new prospect in Siebel CRM.

commit step

A type of task step that explicitly saves temporary task data to the Siebel database. You can use a commit step to transfer data from temporary storage to the Siebel database while the task runs. The end step saves the temporary data to the Siebel database when the task finishes.

decision point

A type of a task step that evaluates conditions on an outgoing condition branch to determine the next step to run. Establishes conditional logic in a task UI. Determines direction through the task UI according to input.

end step

A type of task UI step that instructs the run-time task UI to end the task instance and then save temporary data to the Siebel database. It also provides one last chance to modify any task properties that the task output arguments returned from the calling object.

error handling

A technique to gracefully intercept and handle processing errors that occur while a task UI runs. Explicit error handling uses a combination of exception branches and error steps. Exception branches can end with an error step. The semantics of the error step are similar to the default exception handling.

error step

A type of task UI step that displays a custom error message in the Siebel client. Allows you to configure Siebel Task UI to display a localized error message.

long-running workflow process

A persistent workflow process that can last for hours, days, or months. A task UI allows for integration with a long-running workflow process. You can use a long-running workflow process to create an assigned task. Typically, this occurs when the task finishes and a workflow process starts that creates an inbox record for the next user who must process more information.

Siebel operation step

A type of task step that performs an operation on business component data. Example operations include insert, update, and delete.

start step

A type of task step that starts a task UI.

subtask step

A type of task step that calls another task UI. It allows you to start a separate task UI in a task UI.

task

A logical work unit that a user performs to finish a business operation. From the perspective of the user, a task is the view of a logical work unit that the user must perform. Siebel CRM displays a task in the Siebel client as a link that the user can click in the task pane. It then displays the view where the user performs this work unit.

task UI

A solution that allows you to create a user interface that guides the user experience in a Siebel application. Uses an interface that is similar to a wizard that guides the user through a series of tasks, allowing bidirectional navigation, branching, and the ability to pause and resume the task UI.

task branch

A child object of a task that directs task flow to proceed from one step to another. Different types of task branches exist just as different types of branches in a workflow process exist. A conditional branch can direct task flow from a decision point. An exception task branch can direct task flow if an error occurs. An error step can customize an error message.

task chapter

An object that groups task steps. It informs the user of tasks that the user can perform as a group. It informs the user of the tasks that this user must complete.

task event

A type of event that handles a task operation. A user pause, resume, cancel, delete, or complete a task UI. These actions are known as task operations. A task event handles the operation when a task operation occurs. The event handler for each operation performs the required action.

task group

A grouping feature that controls the tasks that Siebel CRM displays in the context pane. You can use a task group to assign tasks to views according to a grouping. The task group allows you to determine if the task must assume the record context when the user starts the task, or if the user can start the task as a subtask task.

task instance

A single instance of a task UI. Siebel CRM creates a new task instance when the user or a long-running workflow process starts a task. The relationship between a task instance and a task UI is similar to the relationship between an object and a class in an object oriented programming language.

task metric

A type of metric that collects and stores task data that Siebel CRM regularly saves to a data warehouse. Siebel CRM can store timestamp metrics and property metrics for a task UI.

task object definition

The metadata component of a task UI. The object definition for the task includes the task UI and the properties of the task flow. For example, view names. The task definition refers to the many items required to run a task instance, including definitions for task views, applets, business components, LOVs, and other metadata.

task property

Object that stores data that is local to a task or a subtask. A task property is analogous to a local variable in a programming language. The task property defines the characteristics of a task. You can use it to pass data in and out of a task, which is similar to input arguments and output arguments in a programming language.

task session

A type of task session that begins when the user opens or resumes a task and ends when the user pauses, completes, or cancels a task. This configuration implies that a task instance can span zero, one, or more task sessions.

task step

A child object of a task. Each task includes several steps. Similar to using the Workflow Editor to develop a workflow process, you use the Task Editor to create conditions and values for a task step. Examples of task steps include the start step, business service step, and end step.

task UI framework

A set of solutions that allows you to implement a task UI. It includes the development, run-time, and administrative features that allow you to implement a task UI.

task view step

A type of task step that displays a user interface view in the Siebel client. It allows the user to interact with application data in a task UI. A task view includes a playbar applet that allows the user to control how the task runs and navigates.

temporary storage

A type of storage that stores transactional data that is specific to a task instance. When a task runs, Oracle's Siebel Task UI stores this transactional data in temporary storage and does not save it to the Siebel database unless it encounters a commit step. If the user cancels the task UI, then it removes this transactional data from temporary storage. If the user finishes the task, then it saves this transactional data to the Siebel database.

Index