Siebel

Business Process Framework: Task UI Guide

October 2023

Contents

ORACLE

6 Creating Steps and Connectors   53

7 Developing a Task   71

ORACLE

ORACLE

13 Siebel Task UI Reference   195

14 Glossary   203

ORACLE

Preface

This preface introduces information sources that can help you use the application and this guide.

Using Oracle Applications

To find guides for Oracle Applications, go to the Oracle Help Center at *https://docs.oracle.com/*.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the *Oracle Accessibility Program website*.

Contacting Oracle

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit *My Oracle Support* or visit *Accessible Oracle Support* if you are hearing impaired.

Comments and Suggestions

Please give us feedback about Oracle Applications Help and guides! You can send an email to: *oracle_fusion_applications_help_ww_grp@oracle.com*.

ORACLE

1 What's New in This Release

What's New in This Release

This chapter tracks the changes in the documentation. It includes the following topics:

- *What's New in Siebel Business Process Framework: Task UI Guide, Siebel CRM 23.10 Update*
- *What's New in Siebel Business Process Framework: Task UI Guide, Siebel CRM 23.9 Update*
- *What's New in Siebel Business Process Framework: Task UI Guide, Siebel CRM 22.7 Update*
- *What's New in Siebel Business Process Framework: Task UI Guide, Siebel CRM 20.1 Update*

What's New in Siebel Business Process Framework: Task UI Guide, Siebel CRM 23.10 Update

The following table lists the changes in this revision of the documentation to support this release of the software.

Topic	Description
Removing a Task Instance from the Task Instance Monitor	New topic. Functionality added to remove task instances from the Task UI.

What's New in Siebel Business Process Framework: Task UI Guide, Siebel CRM 23.9 Update

The following table lists the changes in this revision of the documentation to support this release of the software.

Topic	Description
Creating a Custom Task	Modified topic. "To create a custom task using the Task Wizard" is modified with updated steps.
Updating to 22.7+ and Task Based UI	New topic. Task Based User Interface (TBUI) Tasks were Workspace-enabled in the 22.7 Siebel Monthly Update. Before updating to 22.7 and beyond you must make sure your deployed Tasks have the same version as the Task in your Design Repository.
Making sure both Repositories are synchronized	

What's New in Siebel Business Process Framework: Task UI Guide, Siebel CRM 22.7 Update

The following table lists the changes in this revision of the documentation to support this release of the software.

ORACLE

Topic	Description
Inspecting a Task	New topic. Describes how to inspect and preview a task in the application, prior to delivering the task to the parent Integration Workspace.
Delivering a Task	New topic. Tasks are now Workspace-enabled, which means you can edit and deliver them within a Workspace.
Binding a Task View to a Task View Step	Modified topic. The following subsection is new: *Conditions for Binding Task Views to Task Steps.*.
Examples of Developing a Task	Modified chapter. The examples in this chapter have been updated since tasks are now Workspace-enabled, which means you can edit them within a Workspace and that it is no longer necessary to activate tasks (or subtasks) or publish tasks.
Validating a Task	Modified topic. The Validate Tool is now supported in Web Tools. The Applet Menu in Web Tools has a new Validate option, which you can use to validate objects of a single object type.
Examining the Logic of a Task *About the Task Editor* *Main Elements of the Task Editor* *Implementing the Task* *Using the Task Editor* *Opening the Task Editor* *Deleting Task Steps and Connectors* *Creating a Connector* *Creating a Branch Connector* *Creating a Condition on a Branch Connector* *Creating the Condition Criteria*	Modified topics. The Task Editor is now supported in Web Tools, which means you can edit tasks within a Workspace. You click Edit Task Flow (the pencil icon) or drill down on the Task Name in the Task list to open and modify existing tasks using the Task Editor in Web Tools.
Transferring Tasks to the Siebel Mobile Web Client	Modified topic. This topic was previously called *Replicating a Task to the Siebel Mobile Web Client.*
Using the Task Instance Monitor	Modified topic. Step 2 has been updated – the Task Monitoring Configuration view is new.
Activating the Tasks and Subtasks Publishing and Activating a Task Deploying and Migrating a Task Process of Deploying a Task Preparing to Publish a Task Publishing a Task Activating a Task Migrating a Task	Obsolete topics. These topics have been removed from the guide.

ORACLE

Topic	Description
Migration Options	

What's New in Siebel Business Process Framework: Task UI Guide, Siebel CRM 20.1 Update

No new features have been added to this guide for this release (20.1 Update). This guide has been updated to reflect only product name changes.

2 Overview of Siebel Task UI

Overview of Siebel Task UI

This chapter provides 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*
- *About Predefined Tasks*
- *About the Task Editor*
- *Main Elements of the Task Editor*

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.

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

Comparing a Task View and Standard View

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 simplifies the interface and reduces the potential for mistakes.

Standard views come predefined with Siebel CRM where users access them to perform a wide variety of business functions. A standard view:

- Can include any or all of the following: a form, list, tree, or chart applet. A scroll bar, screen tab, or a drop down list can be used to navigate through the records.

ORACLE

- Can include a superset of the fields and controls that users must set or configure to complete a business function.

- Provides users with significant functionality and capability to navigate a Siebel application and to modify data. As a result, users are required to possess significant knowledge and skills about how to use standard views. With all of these options and the necessary knowledge requirement, it is possible that a user may make a mistake while using the view.

Example of a Standard View

The following image shows the Agreements Lists standard view, which you typically access to manage agreements. The power to use numerous fields that reference the underlying data and the many navigation options available result in an interface that requires a lot of skill and knowledge to use accurately.
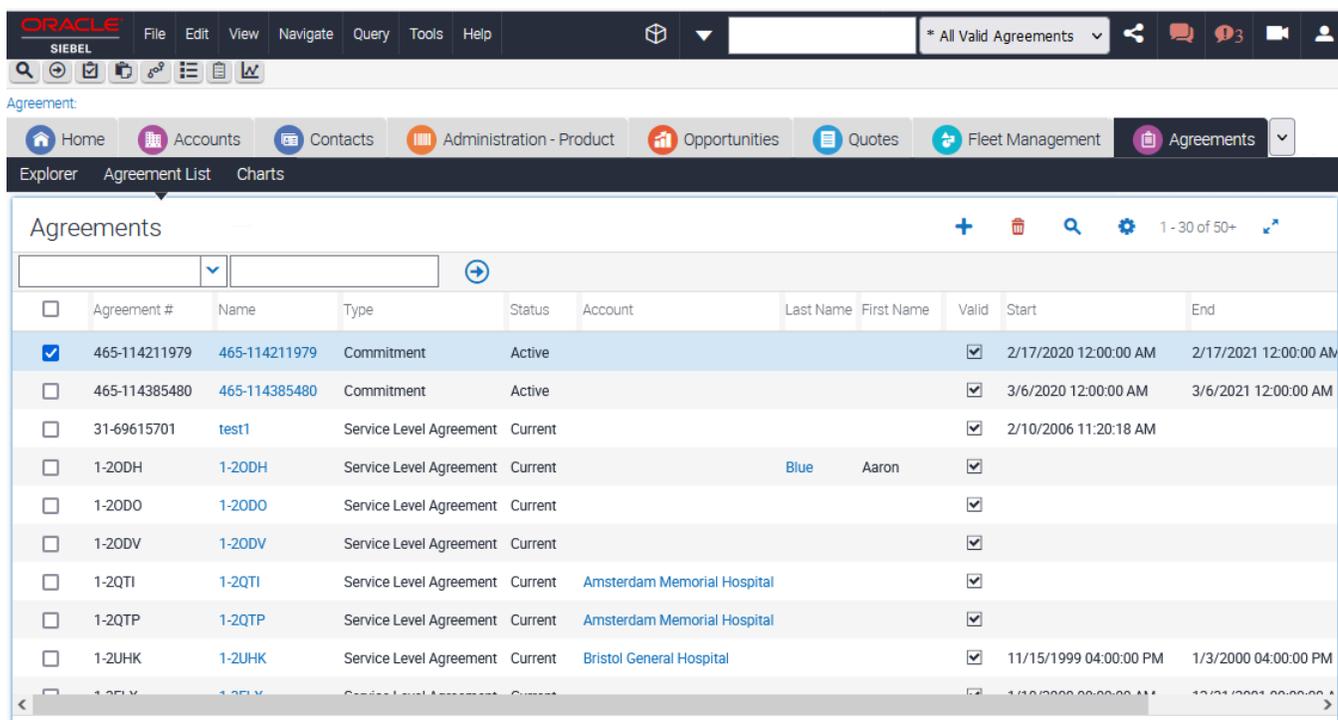
Example of a Task View

The following image is an example of a view which is using a Siebel Task to add an asset to an agreement. The task guides you through the steps involved in completing the (Assets to Agreement) task and employs forward and backward navigation functionality through the use of the following buttons: Pause, Previous, Next, Cancel.

ORACLE
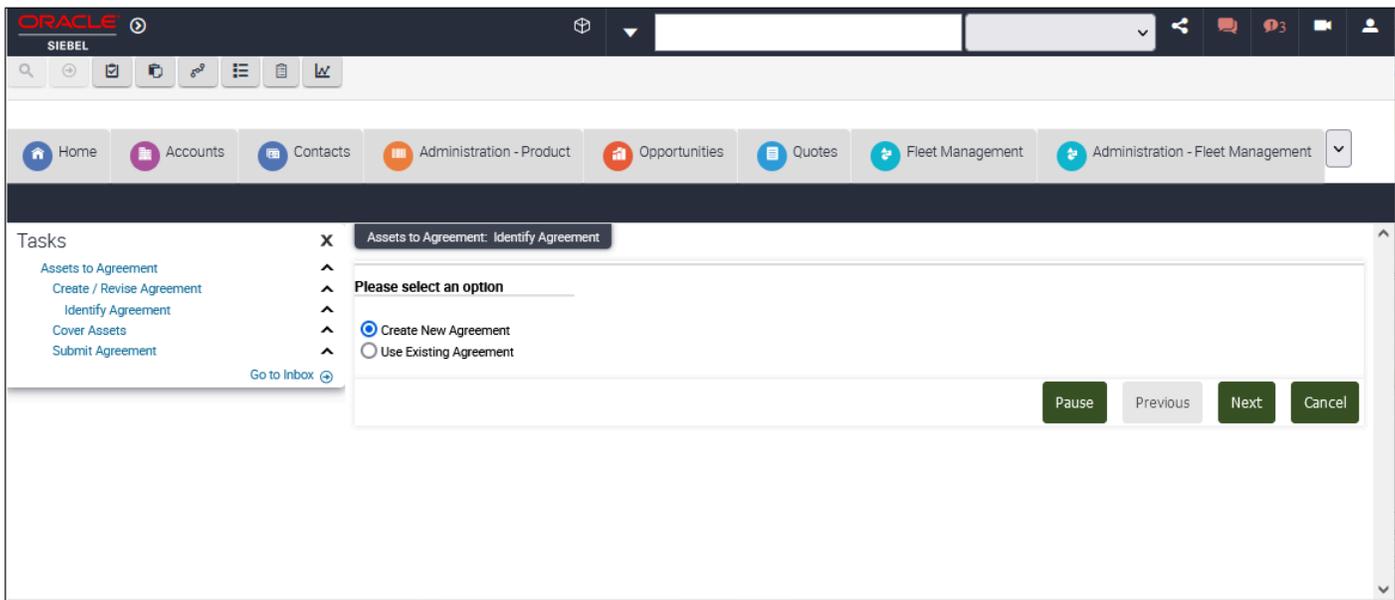
Siebel Task UI Features and Benefits

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 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 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 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. 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 task data through tight integration with Oracle Business Intelligence.

ORACLE

Comparing Siebel Task UI with Other Technologies

Siebel Task UI is the only technology that supports a long-running transaction when compared to other UI technologies available in the Siebel CRM framework. Siebel Task UI is closely integrated with Siebel Workflow.

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 provides.

A task 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:

- Nontrivial
- Transactional
- Tight integration with a business process

Other UI Technologies

The following table compares features between Siebel Task UI and other UI technologies in the Siebel CRM framework.

This table 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.

Feature	Siebel Open UI Client	Siebel Task UI	iHelp	Smart Script	Web Channel
Employee facing	Yes	Yes	Yes	Yes	Not recommended
Customer facing	Yes	Yes	No	Yes	Yes
Encapsulates business logic	Some	All	None	All	All
Integration with Siebel Workflow	Some	Full	None	Some	Limited
Integration with Universal Inbox	Some	Good	None	Good	Some
Support for long-running transaction	No	Yes	No	No	No

ORACLE

Feature	Siebel Open UI Client	Siebel Task UI	iHelp	Smart Script	Web Channel
Performance and scalability overhead	None	Some	None	Large	Some

Comparing Siebel Task UI with Siebel Workflow

A task involves at least one task view step where a user enters data. A task 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 are the result of a user who explicitly clicks the Next, Previous, Pause, and Cancel buttons to navigate through the task.

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 in the Workflow Process that supports this engagement.

About Predefined Tasks

A *predefined task* is a task that comes already defined with a Siebel application. Before you create a new task, you should examine predefined tasks to determine if one exists that meets your design requirements, or that you can modify to meet your requirements. You can configure some of the predefined tasks. The following table lists some predefined tasks.

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

ORACLE

Task Name	Business Object
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

Note: Tasks that are in an Inactive state in the repository will have to be updated to an active state.

Examining the Logic of a Task

You can examine the logic of a task by opening it in the Siebel client and then completing the steps in the following procedure. For example, to examine the logic of the FS Asset To Contract Task predefined task, open it in Siebel Tools or Web Tools and then complete the steps in the following procedure.

To examine the logic of a task

1. In the Object Explorer, click Task.
2. In the Tasks list, query the Task Name property for the FS Asset To Contract Task.
3. In the Siebel client, open the task in the Task Editor:
   - (Siebel Tools) Right-click the FS Asset To Contract Task and then select Edit Task Flow.
   - (Web Tools) Select the FS Asset To Contract Task and then click Edit Task Flow (the pencil icon). You can also drill down on the Task Name in the Task list to open the task.

ORACLE

4. Examine the task flow that appears 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, click Help, click About View, and then note the value that Siebel CRM displays for the View in the About View dialog box that opens.

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

About the Task Editor

The *Task Editor* is a graphical user interface that provides a declarative framework that helps you create a task. It allows you to use an object oriented programming language in an integrated development environment.

Using the Task Editor minimizes scripting, allows for dynamic modification, and encourages you to use a hierarchical development strategy. For example, you can select steps from the Palettes pane in the Task Editor and move them to the Canvas, where you can define the flow of the task by creating Connectors (of type Condition, Default, or Error) between the steps.

The Task Editor simplifies the work of supporting a business process that is complex and has a long life in a Siebel application.

Main Elements of the Task Editor

The Task Editor contains the following main elements, as shown in the following image:

1. **Canvas pane.** The main area of the UI where the task flow appears and where you create and define the task. For more information, see *Process of Creating a Task*.

2. **Palettes pane.** A pane that contains icons that represent the various step types you can add to a task. To add a step to a task, select it from the palette, then drag and drop the step on the canvas. For more information, see *Creating Steps and Connectors*.

> **Note:** The term *drag and drop* is used throughout this guide to describe how to move steps from the *Palettes* pane to the canvas. You move a step by first selecting the step in the Palettes pane and (with the mouse button depressed) then moving the step to the canvas (where you release the mouse button).

    To increase or decrease the size of the Palettes pane, drag (move) the pane border to resize the pane. To hide the Palettes pane, drag the pane border to reduce the size of the pane so that it disappears.

3. **Multi Value Property Window (MVPW) pane.** A pane where you define multiple properties for a task or (input and output) arguments for a task step. For more information, see *About the Task Property* and *Arguments of a Task Step*.

4. **Properties pane.** A pane where you define properties for an individual task step or for the overall task. For more information, see *About the Task Property*. The Properties pane is context-sensitive in the following ways:

   ○ If you choose a step or connector on the canvas, then the properties for the selected step or connector appear in the Properties pane.

   ○ If you select no step or connector on the canvas, then the properties for the overall task appear in the Properties pane.

   To increase or decrease the size of the Properties pane, drag (move) the pane border to resize the pane. To hide the Properties pane, drag the pane border to reduce the size of the pane so that it disappears.

Siebel Task UI and Siebel Workflow use similar objects (such as Start step, Business Service step, Decision Point, and End step). Using the Task Editor is similar to using the Workflow Editor. For more information about using the Workflow Editor, see *Siebel Business Process Framework: Workflow Guide* .

3 Siebel Task User Interface Elements

Siebel Task User Interface Elements

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

- *About the Task Pane*
- *About the Task View*
- *How Task UI Uses the Dashboard and Universal Inbox*

About the Task Pane

The *task pane* is a user interface element in a task that provides a way to organize tasks. It also allows the user to start a task and to monitor progress through a task.

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 is currently running, then the task pane displays the current task pane. Otherwise, it displays the context pane. For more information, see the following topics:

- *Context Pane*
- *Current Task Pane*
- *Task Group*
- *Task Chapter*

Managing the Task Pane Footprint

The user can manage the footprint that the task pane consumes by closing the task pane as follows:

- By clicking the Tasks button.
- By clicking Close (the x icon) in the 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 selects the name of a task in the task pane, then Siebel CRM opens that task instance, displays the first view in the task, and replaces the header task at the start of the task pane with the current task (that you just opened).

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 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.

The following image shows an example of the context pane and standard view, and highlights the following key elements:

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.** The header for the context pane is always Tasks.
3. **Task group.** In this example, the display name for the task group is Agreement Creation Task. The task group item is Assets to Agreement. The user must click Assets to Agreement to start the task. For more information, see *Task Group*.
4. **Standard view.** Provides the context. The standard view lists the task that Siebel CRM registers for this view and that the user can open. For example, if the user is currently in the Agreements view, then the context pane lists the tasks relevant to Agreements.

   Siebel CRM does not restrict the context pane to the context. Logically, it makes sense to organize tasks 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 assessment context can include a task that resides in the task pane that is related to an opportunity.
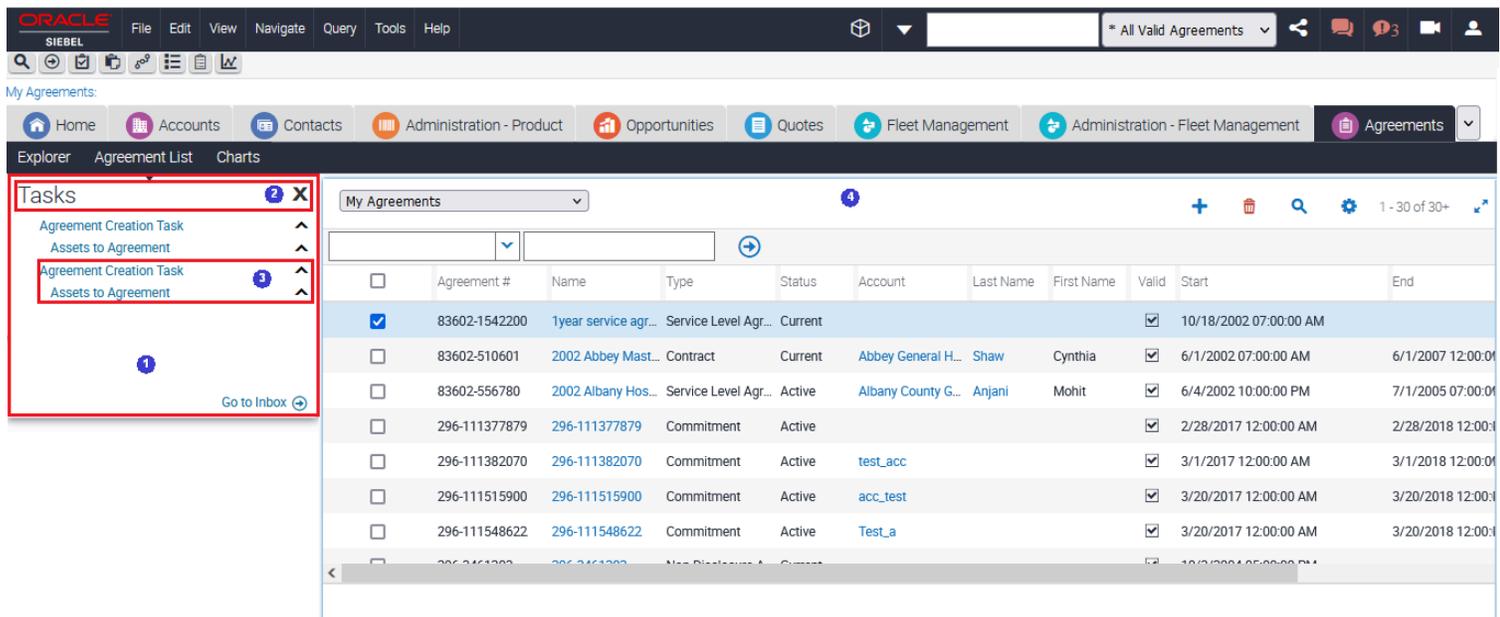
Current Task Pane

The *current task pane* is the active task pane that helps the user navigate through a task. After the user clicks a task link in the context pane, Siebel CRM displays it in the current task pane.

The following image shows an example of the current task pane, which has the following key elements:

1. **Current Task pane.** A pane that helps the user navigate through a task. In this example, the Assets to Agreement task in the current task pane contains the following steps: Create/Revise Agreement (which is selected and expanded to show the Identify Agreement substep), Cover Assets, and Submit Agreement.
2. **Task view.** A type of view that includes task applets or Siebel CRM applets. In this example, the current task view Identify Agreement. For more information, see *About the Task View*.

ORACLE

3. **Applets.** A Siebel CRM applet, task applet, or task playbar applet in a task view. For more information, see *Task Applet* and *Task Playbar Applet*.

4. **Task chapter.** A list of task steps that Siebel CRM groups under a common chapter name. In this example, Create/Revise Agreement is the task chapter and Identify Agreement is the task step. For more information, see *Task Chapter*.

If no chapters are defined for a task, then the current task pane displays only the view steps.
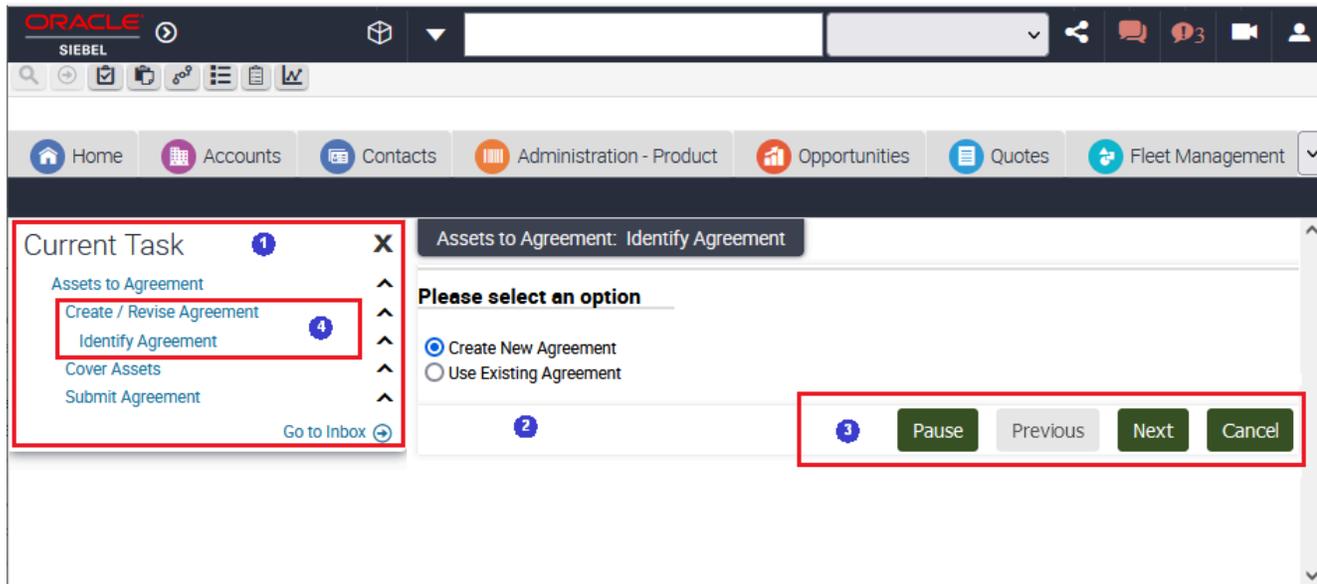
About Fast Forward and Fast Backward Navigation

Users can navigate back and forth through a task without having to repeatedly click Next or Previous by using the *fast forward* and *fast backward* navigation features.

For example, assume a user starts a large task 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 step link in the task pane, then Siebel CRM displays the view that this link references. But if the user uses the fast backward navigation feature, 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. 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 task pane to navigate directly to a view.

Siebel CRM enables the fast forward and fast backward navigation feature by default. Fast forward and fast backward navigation can be used:

- Only in views that have already been visited by the user in a task instance.

ORACLE

- With various task features, such as the following:

  - Pause a task.

  - Transfer a task to another user.

  - Navigate to another task chapter.

  - Navigate to a subtask.

About the Task Progress Indicator

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

    Percentage Complete: 40%

Siebel CRM displays a Progress Indicator in the 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.

How Siebel CRM Determines the Percentage of Task Completion

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

- The number of views the user completed

- The total number of views in the task

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

To avoid runtime 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 runtime 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*.

Task Group

A *task group* is an object type that groups tasks that Siebel CRM displays in the context pane. The task group is a required 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 highest level task group object references tasks that the user can open if the business component is attached to an applet that is active and visible.

ORACLE

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 *Context Pane*. For more information, see *Creating a Task Group*.

Task Group Display Name

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 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 Task Group Across Multiple Views and Applications

You can specify the Siebel application where Siebel CRM displays a task. 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. 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 tasks 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, 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 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.

ORACLE

Siebel CRM uses the Require Context BC property to indicate that a task group can include a task 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 does not require a context business component, then Siebel CRM displays it.

- If the task 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 that Siebel CRM groups under a common chapter name. A task chapter is an optional UI element that provides a map of what lies ahead in completing the task, and what work has been finished in the task. Use the task chapter to configure a logical grouping of task steps. You can 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, then only the chapter name appears in the task pane. If Siebel CRM runs the first task view step of a chapter, then it expands the chapter name in the task pane. It does this to display the names of task views that the user finished. In the current chapter, the task pane displays view steps that the user visited in the order of visitation. The current view step appears in bold.

If you assign one step to a chapter, then you must assign all other steps in the task 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.

The image in *Current Task Pane* displays a task named Create New Account that includes two chapters. The chapter name provides details for the following parts of the task:

- Enter New Account
- Enter New SR

The example (in *Current Task Pane*) illustrates that the user is currently performing the Enter Account Details step.

For more information, see *Creating a Task Chapter*.

Using Color with a Task Chapter

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

Using Chapters with a Subtask

You can use the Task Editor to assign a task 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

A *task view* is a type of view that can include applets and task applets. You use task views to view application data that the task displays, and to input data. The task view includes a playbar applet with buttons (such as Next and Previous)

ORACLE

which you can use to navigate through the task in a stepwise, guided fashion. A task view can include optional elements to improve usability. The task view is a required element. To view an example of a task view, see *Current Task Pane*.

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 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.

For more information, see the following topics:

- *Task Applet*
- *Task Playbar Applet*
- *Radio Button Group*
- *Applet Message*
- *Creating a Task View Step*
- *Creating a Task View*
- *Guidelines for Designing User Interface Elements*

Templates 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 doesn't display any of the following items:

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

Each task view includes a title that indicates to the user that the view is not a standard view.

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

- View Detail (Parent with Pointer). Use this template for your base task view. Use this template to configure your task and place applets side by side to provide a more structured view.
- View 1 Over 2 Over 1

Task Style Sheet

You can use the theme-xxx.css style sheet (for example theme-aurora.css, theme-redwood.css, and so on) to define the look and feel of a task. If you modify the style sheet, then Siebel CRM applies these modifications globally to all your tasks.

ORACLE

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. For more information, see *Overview of Transient Data*.

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

Comparison of the Task Applet to the Standard Applet

A task applet differs from a standard 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 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 standard 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*.

The following table describes the operations that Siebel CRM restricts for each type of applet.

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

ORACLE

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 in the header or footer area of a task view. The task playbar applet, shown in the example image in *Current Task Pane*, 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 (note that this appears in the footer area of a task view)
  - Applet Task Playbar - Top (note that this appears in the header area of a task view)
- 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
- CCAppletPlaybarBottom
- CCAppletPlaybarTop

For more information about:

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

Task Playbar Display Requirements

A task view must include a task playbar applet. Position the task playbar in the upper or lower section of a task view that does not require vertical scrolling. If vertical scrolling is required, then position the task playbar in the upper or lower section of the view also. From a usability standpoint, it is recommended that you avoid vertical scrolling.

Task Playbar Validation with Forward Navigation

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. The following table describes validation for forward navigation and validation results.

Validation	Validation Result
A validation error occurs.	An error message appears, 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, running task steps until it reaches the next view step or the next end step.
An exception occurs that Siebel CRM does not handle.	Siebel CRM stops forward navigation and displays the error message. It then returns control to the user in the same view where the user started forward navigation.

ORACLE

Validation	Validation Result
Forward play of the task successfully arrives at the next task view step.	Siebel CRM applies the task 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 includes more than one view. If the task includes only one view, then the default value is Submit.

- **Submit.** Indicates that Siebel CRM is about to save transaction data for the task to the Siebel database for enterprise-wide consumption. After saving transaction data, roll back is not possible. This situation typically occurs at the end of the task. For tasks that include a commit step, the Submit button appears in the last task view that appears before the commit step.

- **Finish.** Indicates that clicking the forward navigation button ends the task. You can use it only in a task 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.

Task Playbar Backward Navigation

Clicking the Previous button validates the data in the current view, and then returns to the last displayed view. This feature is most useful if you make a mistake during the task and need to return to an earlier view to correct this mistake. For example, if you enter erroneous data or choose 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 you modify this data, then Siebel CRM displays these modifications when you navigate back to the target view.

- You can continue backward navigation until you reach the first view in the task. You can freely cross subtask boundaries.

- Siebel CRM disables the backward navigation button in the first view of the task.

- If you use backward navigation, and then make a choice at a decision point that varies from the original choice that you made, then subsequent forward navigation can cause the task to enter a branch in the task that is different from the branch that you first pursued.

For more information, see *Disabling Backward Navigation*.

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 records are created 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*.

ORACLE

- 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*.

Task Playbar Pause

When inspecting a task, the Pause button is disabled to prevent the creation of an inbox item pointing to the task.

Clicking the Pause button in the task playbar applet pauses the current task. 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 standard view that it displayed when the user started the task. 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*
- *About Event Handling*
- *Disabling the Pause Button*

Implicit Pause

When inspecting a task, the implicit pause behavior is disabled to prevent the creation of an inbox item pointing to the task. If the user tries to implicitly pause the task, an error message similar to the following is returned:

```
Error: Task Cannot be Paused in Inspect Mode (SBL-UIF-52142)
```

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. For example, if the user clicks in the site map, or clicks a screen tab.
- The session for the user times out.

Task Playbar Cancel

Clicking the cancel button in the task playbar applet cancels the task. The effect of clicking the cancel button depends on the following task state:

- If the user never paused the task, then cancelling it leaves no trace of the task except for task timestamp metrics.
- If the user paused the task at least once, then cancelling it resets the task state to the state that existed during the last pause. Siebel CRM also rolls back task transactions as follows:
  - 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 task pause.

ORACLE

Radio Button Group

A *radio button group* is an optional user interface element that prompts the user to choose only one of a predefined list of options. Along with the combo box and list, you use the radio button group to make a choice that drives the task flow.

The following image shows an example radio button group that displays a predefined list of options. You can choose one of these options.

The input that a radio button receives determines the next task step that appears, or the data that appears in the next task view. For example, clicking the Credit Card Dispute radio button option will open the CC Dispute view for example. Clicking the Balance Inquiry radio button option will open a different view, and so on.

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*
- *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*

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*.

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.

ORACLE

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.

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.

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* and *Siebel Object Types Reference* .

How Task UI Uses the Dashboard and Universal Inbox

Task UI uses the persistent dashboard and Universal Inbox frequently.

Persistent Dashboard

The *dashboard* is a standard 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

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.

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 end 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 a Task Instance is Managed in the Inbox

Each inbox item of type Task references only one task instance. For 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, 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 to enter a message in the context field that is specific to a task instance. For more information, see *Modifying a Task to Display a Message that is Specific to a Task Instance*.

How to Resume a Paused Task from the Inbox

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

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*.

> **Note:** When a task is paused, task inbox items that do not belong to the current Workspace will also be visible in the inbox. If you try to resume any task that does not belong in the current Workspace context, then you will be unable to do so. This is because the task instance being resumed may reference items and data specific to a particular Workspace.

ORACLE

4 Task Development Scenario

Task Development Scenario

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

- *Scenario Overview*
- *Task Development Example*
- *Determining Required Task Improvements*
- *Designing the Task*
- *Developing the Task*
- *Testing the Task*
- *Implementing the Task*

Scenario Overview

The *Task Development Example* in this chapter guides you through the steps involved in developing a task. The example 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.

The job roles described here are in the context of a fictional organization. Job roles, job titles, and division of labor might vary significantly for your organization.

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.

ORACLE

Task Development Example

The typical steps involved in developing a task are as follows:

1. *Determining Required Task Improvements*
2. *Designing the Task*
3. *Developing the Task*
4. *Testing the Task*
5. *Implementing the Task*

For more information about:

- General technical procedures, see *Roadmap for Developing a Task*.

- Specific technical procedures, see *Examples of Developing a Task*

Determining Required Task Improvements

This task is a step in *Task Development Example*.

A *brainstorming* meeting takes place where the business proccesses for the orgrainzation are listed on a whiteboard. The best candidates for process improvement are assessed and it is determined that the main criteria include the following items:

- Return On Investment (ROI)

- Risk

The IT director and business analyst 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 IT director and business analyst 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 IT director and business analyst 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.

ORACLE

Designing the Task

This task is a step in *Task Development Example*.

To design the task, the business analyst begins by modeling the business process, then creates an executable Workflow Process, and so on as follows:

- *Modeling the Business Process*
- *Creating an Executable Workflow Process*
- *Identifying the Context for the Task*
- *Designing the Task*
- *Refining the Task Design*

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
- Review expense report, accomplished through a task
- 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 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.
- 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 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.

ORACLE

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 models the business process as a long-running Workflow Process. The following figure shows the steps in the Workflow Process (which is not yet executable), and the steps are as follows: Start, Submit, Review, Pay, and End. The illustration serves as a way to communicate the design intention with other team members when accompanied with notes from the analyst.
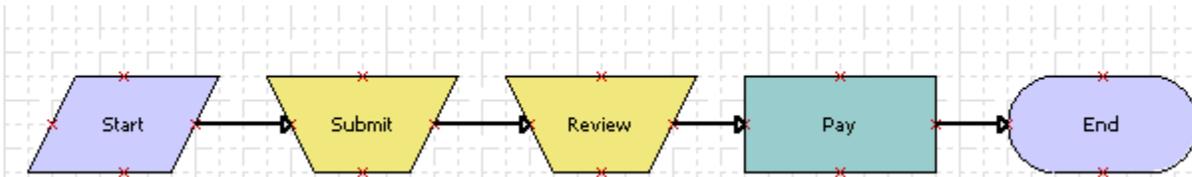
Other team members notice that the model is not finished because the business logic for the approval is not clarified. For example, when is a single approval sufficient, and when are more approvals required for a single expense? Also, the team notices that the model does not cover the situation where a review step finds that the expense report is ineligible and Siebel CRM must reject it.

The business analyst modeled the Workflow Process, so the application developer can refine the model. The following figure shows the revised prototype that the developer can refine into an executable Workflow Process and it includes the following steps:

1. A Start step.
2. A Submit step.
3. A Needs Further Review? decision step.

   a. If Yes, then continue to step 4.
   b. If No, then go to step 6.
4. A Review step.
5. An Approve decision step.

   a. If Yes, then return to step 3.
   b. If No, then the Workflow Process stops.
6. A Pay step.
7. An End step.

ORACLE
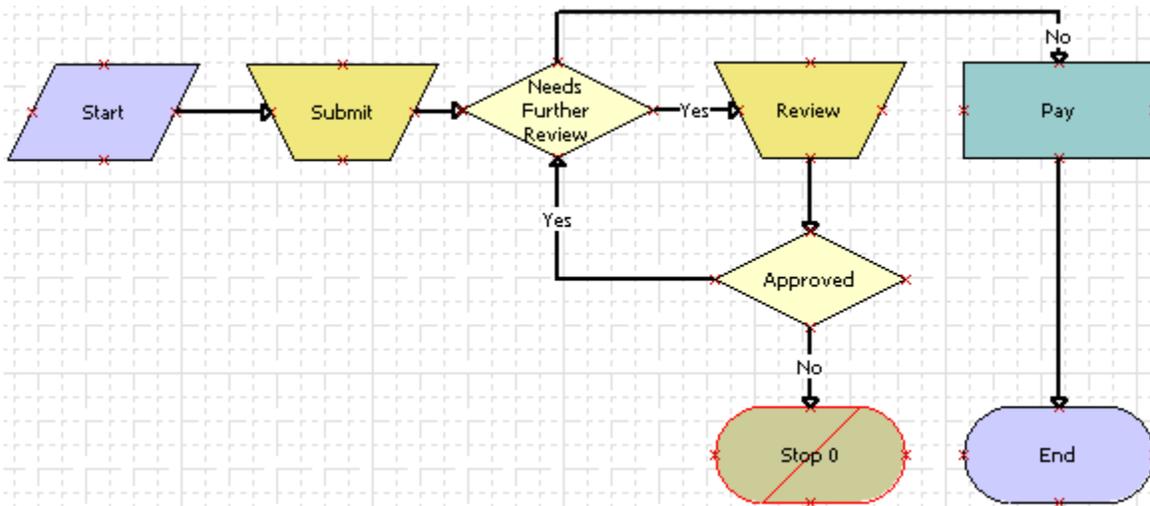
At this point, the developer and the usability analyst possess a thorough understanding of the business process for expense reimbursement.

Next, the application developer creates an executable Workflow Process.

Creating an Executable Workflow Process

The business analyst modeled 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 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 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. The following image shows the executable Workflow Process and it includes the following steps:

1. A Start step.
2. A Check Rules step.
3. A Need Further Review? decision step.
   a. If Yes, then continue to step 4.
   b. If No, then go to step 7.
4. An Assign Reviewer step.
5. A Review step.
6. An Approved decision step.
   a. If Yes, then return to step 2.

ORACLE

> **b.** If No, then the Workflow Process stops.

7. A Pay step.

8. An End step.
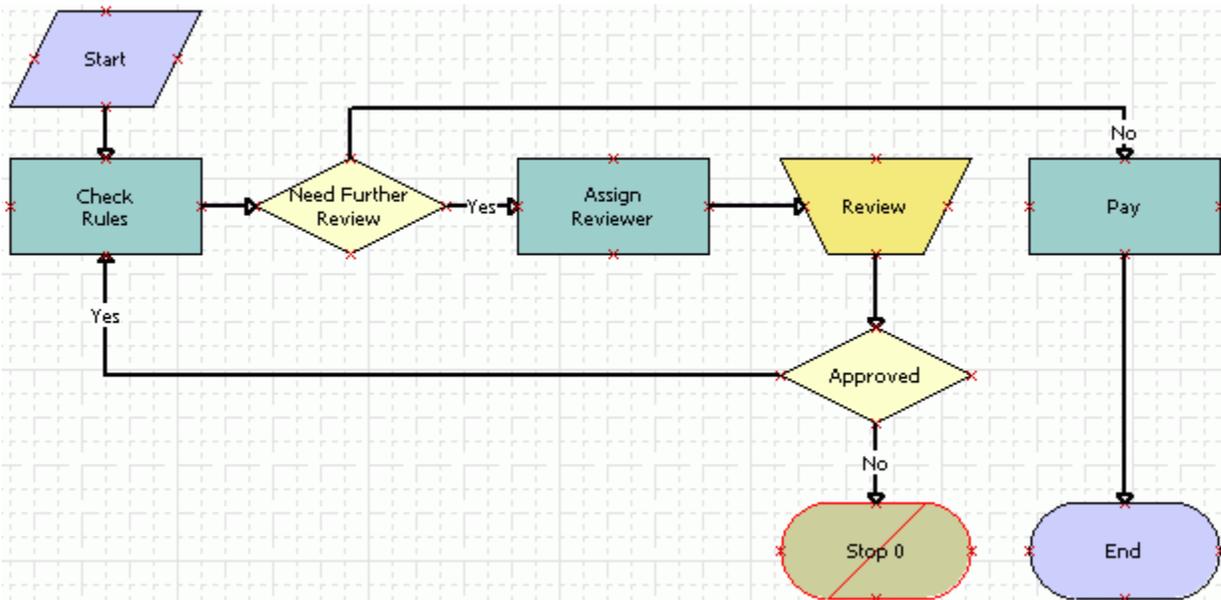
These revisions to the draft business model help to illustrate the differences that exist in the level of abstraction between a business process model and an executable Workflow Process. The application developer continues to refine the Workflow Process. For example, creating conditional branches, process 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.

Identifying the Context for the Task

The business analyst must refine the task. The analyst begins by identifying the context for the task. The application developer notifies the business analyst that the submission task 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 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 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. 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 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 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

ORACLE

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.

Designing the Task

The usability analyst now identifies activities in each task, and creates a prototype for the task. The Submit Expense Report task 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 to model the task. 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 that reviews the expense report is a simple task. It requires only the Review Expense Report task view step.

For an example that includes view mockups that assist with designing a task, see *Example of Developing a Task That Assists with Creating Multiple Opportunities*.

Next, the usability analyst refines the task.

Refining the Task 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 tasks 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 hierarchical 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 displays the same data as the view that resides in the review task displays.

Developing the Task

This task is a step in *Task Development Example*.

This task includes creating and defining the task, including task properties, steps, connectors, and so on using the Task Editor. To begin with, the application developer creates the task, then refines the task, and then sets up access control.

Creating the Task

After the team reaches consensus on the task view layouts for the task, the application developer does the following to create the task:

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.

Refining the Task

The application developer does the following to refine the task:

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

Next, the application developer sets up access control.

Setting Up Access Control

The application developer does the following to set up access control:

- Makes sure the responsibility for the test user possesses the visibility to run the submission task. For more information, see *Adding a Responsibility to a Task*.

- Makes sure the Siebel client displays the customizations.

Testing the Task

This task is a step in *Task Development Example*.

This task includes validating and testing the task to ensure there are no errors and that the task is working as required. The application developer begins with unit testing, then performs integration testing, and finally system 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. The application developer notices that the developer failed to include an Expense Description field in the Expense Header Applet. Several iterations later, the developer demonstrates the submission task to the business analyst who notes that there are a few details that require refinement.

ORACLE

The review task is more difficult to unit test because it requires tight integration with a long-running Workflow Process. The task 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 performs integration testing.

Integration Testing

To test the configuration in the development environment, the application developer inspects the Workspace in which the work is being performed in a Siebel client, such as the Call Center application. The developer can leverage the Task Debugger to analyze any defects that exist in the task logic.

Next, the application developer performs system testing.

System Testing

One the configuration satisfies the business requirements, the application developer delivers the changes to the parent Integration Workspace and the changes are migrated to the test environment, where further testing can be performed.

Implementing the Task

This task is a step in *Task Development Example*.

The task will be migrated to the production environment along with any other customizations in the next production release. It will be available to users upon their next login. In summary, the following substeps are involved in implementing the task:

1. **Deliver.** Commit the task to the parent Workspace in which you designed it. This compiles it into the Runtime Repository tables in that Integration Workspace (or MAIN).
2. **Migrate and Test.** Like any other Workspace-enabled repository object, you can migrate the task's Runtime Repository definition to downstream environments for further testing.
3. **Migrate to Production.** Once the task has been fully tested and the Integration Workspace used for testing is delivered to the Workspace tied to Production, it will be migrated to the Production environment like any other Runtime Repository object. For more information about Workspaces, see *Using Siebel Tools* .

ORACLE

5 Using the Development Environment to Develop a Task

Using the Development Environment to Develop a Task

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

- *About Workspace-Enabled Tasks*
- *Create or Open a Workspace*
- *Displaying Object Types Used to Develop a Task*
- *Opening the Task Editor*
- *Adding a Step to a Task*
- *Deleting Task Steps and Connectors*
- *About the Task Property*
- *Validating a Task*
- *Inspecting a Task*
- *Delivering a Task*
- *Task Wizards*

About Workspace-Enabled Tasks

As of Siebel CRM 22.7 Update, Tasks are Workspace-enabled. This allows you to edit tasks within a Workspace and permits parallel development to take place where multiple users work simultaneously on a task.

Workspace delivery (see *Delivering a Task*) automatically deploys and activates a task in the Application Object Manager (AOM) within a Workspace Context.

The Task pane (see *About the Task Pane*) in an application shows the tasks that qualify for the current Workspace Context in a session.

You can preview a task by clicking Inspect on the Workspace Dashboard before deploying the task in the application.

Updating to 22.7+ and Task Based UI

Task Based User Interface (TBUI) Tasks were Workspace-enabled in the 22.7 Siebel Monthly Update. Before updating to 22.7 and beyond you must make sure your deployed Tasks have the same version as the Task in your Design Repository.

ORACLE

The update process looks at the Tasks deployed to the Runtime Repository and tries to find the matching source for that Task in the Design Repository. The deployed Task had to originate from a Design Repository at some point, and the application wants to make sure that Design definition of the Task remains active and that their versions are the same.

Prior to Siebel Monthly Update 22.7 (Tasks were not Workspace-enabled) we used these two tables.

Repository	Table Name	Version Column	Task Status Column	Function of Table
Design (Tools/Web Tools)	S_TU_TASK	VERSION	STATUS_CD (IN PROGRESS, COMPLETED, NOT IN USE)	This holds the design time definition for the Task.
Runtime (Call Center)	S_WFA_DPLOY_DEF	REPOSITORY_VERSION	DEPLOY_STATUS_CD (ACTIVE, OUTDATED)	This held the runtime, compiled definition of the Task.

In Siebel Monthly Update 22.7 and beyond we use these tables:

> **Note:** The S_WFA_DPLOY_DEF Table is no longer used. S_RR_TASK is the new Table for deployed Tasks.

Repository	Table Name	Version Column	Task Status Column	Function of Table
Design (Tools/Web Tools)	S_TU_TASK	VERSION	STATUS_CD	This holds the design time definition for the Task.
Runtime (Call Center)	S_RR_TASK	VERSION_NUM	No need here because if a record is deployed to this table, it is active and ready to be used.	This holds the runtime, compiled definition of the Workspace-enabled Task.

The update process will look in your Design Repository to find the Task with the status of COMPLETED that also has the highest version. It will compare this to the same named Task record in S_WFA_DPLOY_DEF that has a status of ACTIVE and has the highest REPOSITORY_VERSION.

For example, Here we have three records with the same TASK_NAME in the Design Repository in a version prior to 22.7. The update process finds two of them are in a COMPLETED status. Out of these two records, the highest VERSION is 1. This is the record the update process expects to find in the S_WFA_DPLOY_DEF Table.

TASK_NAME	NAME	VERSION	STATUS_CD	INACTIVE_FLG
MyTask	MyTask:0	0	COMPLETED	N
MyTask	MyTask:1	1	COMPLETED	N
MyTask	MyTask:2	2	IN PROGRESS	N

ORACLE

The update process expects to find a record with a REPOSITORY_VERSION equal to the VERSION in S_TU_TASK and where the STATUS_CD is ACTIVE in the S_WFA_DPLOY_DEF Table.

NAME	REPOSITORY_VERSION	VERSION	STATUS_CD
MyTask	1	2	ACTIVE
MyTask	0	1	OUTDATED
MyTask	0	0	OUTDATED

If they match, there is no issue. But if they do not match, perhaps the design definition has been deleted or inactivated. This means there is an active Task in the Runtime Repository with no source design definition.

Making sure both Repositories are synchronized

Before updating to 22.7+ make sure to revise Tasks that have a different deployed version than design version. Then publish and activate the Task. This will update the REPOSITORY_VERSION to the correct, matching version and set its status to ACTIVE. The steps to perform are as follows for each Task record having a different version.

If the active record (COMPLETED in the Design Repository, ACTIVE in the Runtime Repository) version for a Task in the Design Repository differs for that same Task in the Runtime Repository

1. In Siebel Tools, revise the Task in the Design Repository.

   a. Select the Task and click the Revise button.
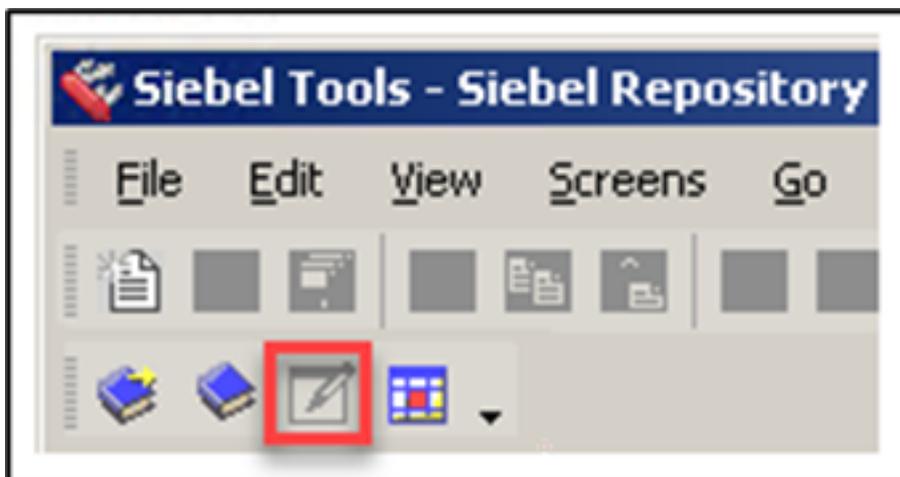
2. No need to change anything as we are only updating the version.
3. The Task will now be In Progress.

ORACLE

4. Publish the Task from Siebel Tools by clicking the Publish button.
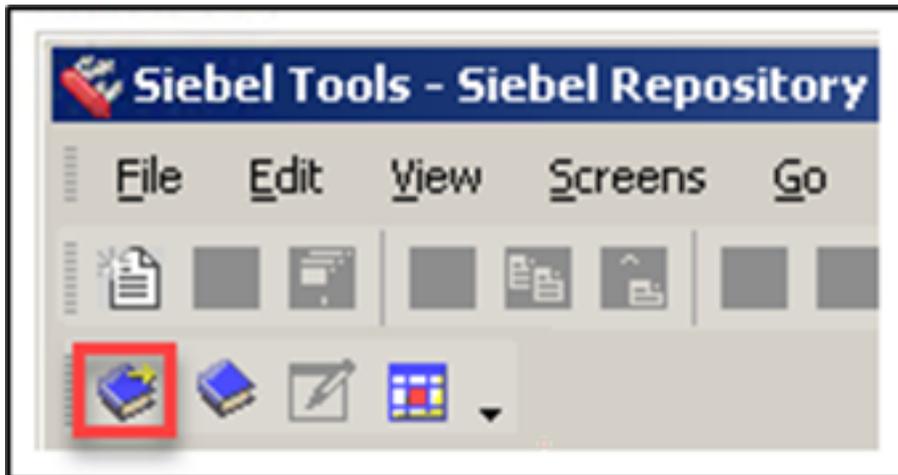
5. Now The Task is published, it must be activated.
6. Activate it by selecting the record in the Administration - Business Process screen, and then the Task Deployment view.
7. Query for the Task and click the Activate button.
8. This will update the REPOSITORY_VERSION to the same as the Design Repository Version.

Create or Open a Workspace

The following procedure shows how to create or open a Workspace. For more information about Workspaces, see *Using Siebel Tools* .

To create or open a Workspace

1. In the Workspace Explorer, locate and select the parent Integration Workspace (IWS) that you want.
2. To create a new Development Workspace from the latest IWS version, click Create on the Workspace Toolbar.
3. To open an existing Development Workspace, select it and then click Open on the Workspace Toolbar.

Displaying Object Types Used to Develop a Task

The following image shows the Task object type hierarchy in the Object Explorer. The key Task objects available under Siebel Object as shown in this image are: Task, Task Group, and View.

ORACLE
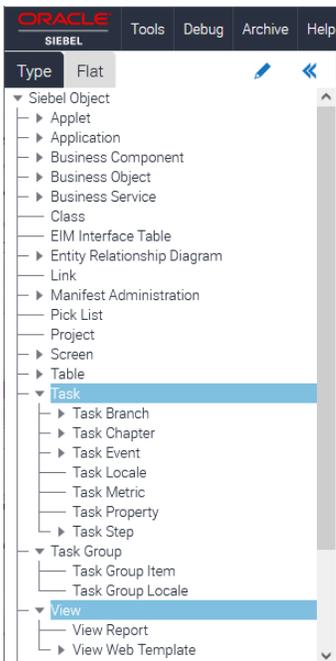
The following procedure shows how to display the object types that you use to develop a task in the Object Explorer. For more information about object types, see *Siebel Object Types Reference* .

To display the object types you use to develop a task

1. In the Object Explorer, click Edit (the pencil icon).
2. Scroll down through the Object Explorer hierarchy list until you locate the following object type(s), which you use to develop a task:

   o Task (Task Branch, Task Chapter, Task Event, Task Locale, Task Metric, Task Property, Task Step)

   o Task Group (Task Group Item, Task Group Locals)

   o View (View Reports, View Web Template)

3. Select the check box next to each of these object types.

   If you add a check mark to a top-level object type, such as Task, then a check mark is also added to all Task child objects. You can expand the parent tree to display only some child objects.
4. (Optional) Select the check box next to other object types that you might use to develop a task:

   a. Expand the Applet tree, and select the check box next to Applet Message (and all its child object types).
   b. In the Applet tree, make sure Applet User Prop includes a check mark.
   c. In the Applet tree, expand the Control tree, and select the check box next to Control User Prop.
   d. Expand the Business Component tree, and select the check box next to Business Component User Prop.
   e. Select the check box next to the Symbolic String tree (and all its child object types).
5. Click Save.

For more information about the Object Explorer and displaying object types in the Object Explorer, see *Using Siebel Tools* .

ORACLE

Opening the Task Editor

The following procedure shows how to open the task editor.

To open the task editor

1. In the Workspace Explorer, create or open a Developer Workspace – for more information, see *Create or Open a Workspace*.
2. In the Object Explorer, click Task.
3. In the Tasks list, locate and select the task you want to modify, then:
   - (Siebel Tools) Right-click the task and select Edit Task Flow.
   - (Web Tools) Click Edit Task Flow (the pencil icon). You can also drill down on the Task Name in the Tasks list to open the task that you want.

     Note: If the task you want to open is in an editable Workspace, then Edit Task Flow (the pencil icon) will be available, otherwise Preview (the eye icon) will be available to open the task in read-only mode.

   The task opens in the Task Editor along with the following panes: Canvas, Palettes, Properties, and Multi Value Property Window (MVPW). If the Palettes or Properties pane is not visible:
   - In Siebel Tools, click View, select Windows, and then select the Palettes menu item or Properties menu item.
   - In Web Tools, decrease the size of the Canvas pane by dragging the pane border. To hide the Palettes or Properties pane again, drag the pane border to reduce the size of the pane so that it disappears. For more information, see *Main Elements of the Task Editor*.

If a limited amount of space exists on your monitor, then you can hide the Object Explorer, Palettes pane, or Properties pane. You can then access each pane through tabs.

Adding a Step to a Task

You use the Task Editor to add a step to a task. You add the step and then use the Properties pane to define properties for the step. After you save the task, a record for the new step appears in the Task Steps list, where you can also modify the properties.

To add a step to a task

1. In the Workspace Explorer, create or open a Developer Workspace – for more information, see *Create or Open a Workspace*.
2. In the Object Explorer, click Task.
3. In the Tasks list, locate and open the task you want to modify – for more information, see *Opening the Task Editor*.
   The task opens in the Task Editor along with the following panes: Canvas, Palettes, Properties, and Multi Value Property Window (MVPW).

ORACLE

4. Select the step type you want to add from the Palettes pane, then drag and drop it on the canvas.

   > **Note:** The term *drag and drop* is used throughout this guide to describe how to move steps from the *Palettes* pane to the canvas. You move a step by first selecting the step in the Palettes pane and (with the mouse button depressed) then moving the step to the canvas (where you release the mouse button).

   For more information about the different step types you can add to a task, see *Overview of Step Types*.

5. In the Properties pane:

   - Enter or modify the Name property. When you step out of the property, the step name is updated on the canvas.
   - (Optional) Enter a description of the purpose of the step.
   - Define other properties for the step, as necessary.

6. Repeat step 3 and step 4 to add more steps to the task as required.

7. Add connectors to the task to define the path between the task steps.

   For information about how to add a connector to a task, see *Creating a Connector*.

Related Topics
- Creating Steps and Connectors

Deleting Steps and Connectors

The following procedure shows how to remove steps and connectors from a task.

To delete task steps and connectors

1. In the Workspace Explorer, create or open a Developer Workspace – for more information, see *Create or Open a Workspace*.

2. In the Object Explorer, click Task.

3. In the Tasks list, locate and open the task you want to modify.

   For more information, see *Opening the Task Editor*.

4. To remove a step from the task:

   - (Siebel Tools) Right-click the step and then select Delete.
   - (Web Tools) Select the step, click the Tasks Menu (cogwheel icon), select Edit, and then select Delete.

   Selecting the task step and then pressing CTRL+D (shortcut key) also deletes the task step.

5. To remove a connector from the task:

   - (Siebel Tools) Right-click the connector and then select Delete.
   - (Web Tools) Select the connector, click the Tasks Menu (cogwheel icon), select Edit, and then select Delete. In addition, dragging one end of a connector to an empty space on the canvas and then releasing the mouse button deletes the connector.

   Selecting the connector and then pressing CTRL+D (shortcut key) also deletes the connector.

ORACLE

Related Topics
- Creating Steps and Connectors

About the Task Property

This topic describes the task property and includes the following information:

- *Task Properties and the Property Set*
- *Arguments of a Task Step*
- *System and Custom Task Properties*
- *How a Subtask Uses a Task Property*
- *Viewing System Task Properties of a Task*

Task Properties and the Property Set

A *task property* is an object that stores a value that the task gets from the Siebel database or gets before or during processing. Some of the ways that Siebel CRM uses a task property are as follows:

- Pass information between objects. For example, between two steps in a task, 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 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*.
- The process property, which is similar to the task property, see *Siebel Business Process Framework: Workflow Guide* .
- Properties, including a detailed description of properties, see *Siebel Object Types Reference* .

How a Task Property Works

The following image illustrates how Siebel CRM uses a task property.

ORACLE
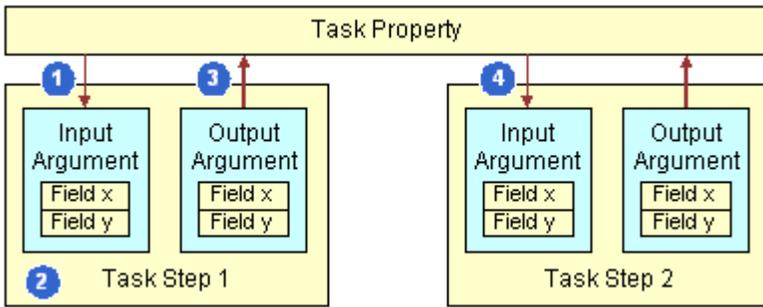
As shown in this image, the following steps describe one example of how Siebel CRM uses a task property:

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.

The following table describes how a task property manages property sets in two different ways.

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

Arguments of a Task Step

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

- To pass the Object Id from a parent task to a subtask, you define input arguments and output arguments for the subtask step.

- 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 through an output argument for the subtask step.

You can define input arguments and output arguments for the steps shown in the following table. The Task Step IO Argument is a child object of the task step object that allows you to use an input or output argument on some types of task steps.

Input Argument	Output Argument
You can define an input argument on the following step types:	You can define an output argument on the following step types

ORACLE

Input Argument	Output Argument
• Business Service • Siebel Operation • Task View • Subtask • Error	• Business Service • Siebel Operation • Task View • Subtask

Some operations include predefined outputs:

- **NumAffRows.** Used with a Query operation. Returns the number of rows returned in the query. For more information, see *Optional View Technique*.

- **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 Arguments for a Task Step*.

- The task property, see *About the Task Property*.

- Similarities between how a task uses a task property and how a Workflow Process uses a process property, see *Siebel Business Process Framework: Workflow Guide* .

- The Task Step IO Argument object, see *Siebel Object Types Reference* .

How the Type Field Affects Other Fields in the MVPW Pane

The value you choose in the Type field in the Multi Value Property Window (MVPW) pane determines how you define other fields in the MVPW pane. The following table describes the fields you can define.

Type Field Option	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 Can Modify*.
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. Click the down arrow 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.

ORACLE

Type Field Option	Work You Perform
	• Define a string in the Value field. The value you enter defines the literal value for the argument.
Task Property	Do the following: • Choose Task Property in the Type field. • Define the Property Name field. For more information, see *About the Task Property*.
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.

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 is automatically created when you create a new task, 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 includes a set of system task properties. If you create a new task, then the following system task properties are automatically created:

- 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*.

The following table describes system task properties. For more information, see *Viewing System Task Properties of a Task*.

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 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. An example of an error code is as follows: SBL-BPR-00515.

ORACLE

Task Property	Description
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 creates a unique identifier each time the task executes and places it in this property.
Object Id	The ROW_ID of the active record in the primary business component.
Siebel Operation Object Id	The object identifier of an object that Siebel CRM updates, creates, or queries during an operation step. It enters data into this system property when the 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 one of the following types:

- String
- Number
- Binary
- Date
- Hierarchy
- Integration Object
- Strongly Typed Integration Object

You use the Multi Value Property Window (MVPW) pane to define a custom task property. For more information, see *Creating a Task Property*.

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 pass data from the parent task into the subtask. 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 in the parent task.
- An output argument allows Siebel CRM to return information from a subtask in the parent task. 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 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

ORACLE

the task properties of the subtask. To choose this value, you can use the list in the Task Input field. The Multi Property Value Window (MPVW) displays 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*.

Why Siebel CRM Passes Hierarchical Data by Reference

A parent task and a subtask include separate local task properties, so passing arguments causes Siebel CRM to move the data between the parent task and the subtask. Task properties of type Hierarchy are automatically passed by reference (which avoids copying the data) to the subtask. All other types of task properties are passed by value and are copied to the subtask.

Viewing System Task Properties of a Task

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

To view system task properties of a task

1. Locate and select the task that you want modify.
2. To view the task properties for the selected task from the Tasks list, expand the Task tree in the Object Explorer, and then click Task Property.

   The Task Properties list shows all the task property definitions for the selected task.
3. (Optional) To view the task properties for the selected task in the MVPW pane, open the Task Editor.

   The Task Editor shows the MVPW pane with all the task property definitions for the selected task. For more information, see *Opening the Task Editor*.

Validating a Task

Before you deliver a task (see *Delivering a Task*), you must validate the task to ensure that it does what it is supposed to do without error. For more information, see *Validating a Task*.

Inspecting a Task

After you validate a task, you can inspect and preview the task in the application, prior to delivering the task to the parent Integration Workspace, as shown in the following procedure. This procedure to inspect a task is optional.

ORACLE

> **Note:** When inspecting a task, the Pause button is disabled to prevent the creation of an inbox item pointing to the task. Implicit pause behavior is also disabled to prevent the creation of an inbox item pointing to the task. If a user tries to implicitly pause a task, an error message will be returned as follows: `Error: Task Cannot be Paused in Inspect Mode (SBL-UFI-52142).`

To inspect a task (optional)

1. Go to the Workspace Dashboard in your client application and open the Workspace that is being used for your task development.
2. Click Inspect on the Workspace Toolbar.
3. Navigate to the view where the task is to be executed (for more information, see *Adding a Task Group to a View*).
4. Inspect and verify the task.

Delivering a Task

Delivering a task to its parent integration branch makes it active in any Application Object Manager (AOM) with that Workspace Context. For more information about Workspaces, see *Using Siebel Tools* .

To deliver a task

1. Modify the task in the Developer Workspace.
2. Test the updated task.
3. Deliver the task to its parent Integration Workspace.

   a. Go to the Workspace Dashboard in your client application and open the Workspace that is being used for your task development.
   b. Click Deliver on the Workspace Toolbar.

On delivering a task, the runtime definition of the task is created as an entry in the S_RR_TASK table.

During FullPublish, the S_RR_TASK table and related tables are reset to reflect the version of the task that remains in the MAIN Workspace.

> **Note:** If there are different runtime definitions of the same task in the runtime table, modified in different Workspaces, then the appropriate definition will be picked up based on the Workspace Context defined in the AOM definition or selected by the user from the Workspace Dashboard at runtime.

About Deploying a Task

A *deployed task* is a task that is active. You create a task in your development environment, and then you deploy it to a testing or production environment.

As soon as you deliver a task to the Integration Workspace, it is compiled into the Runtime Repository and is active in that Workspace. If a migration is performed from that Workspace to any downstream Runtime Repository environment, then the task will also be active in that environment.

Inactivating a Deployed Task

The following procedure shows how to inactivate a deployed task.

To inactivate a deployed task

1. On the Task object in the Siebel Repository, set the Inactive Flag to TRUE.
2. Deliver the change to the parent Integration Workspace.

    Any user who is leveraging that Workspace, whether in the Design Repository (DR) instance or in any downstream Runtime Repository (RR) environment migrated from that Integration Workspace, will ignore the inactive task.

Related Topics
- Adding a Responsibility to a Task
- Transferring Tasks to the Siebel Mobile Web Client

Task Wizards

The following wizards are available to help you create a task:

- **Task Wizard.** Use this wizard to create an object definition for a task, including the first steps that you must create to run a task.

- **Task Applet Wizard.** Use this wizard to create a task applet.

- **Task View Wizard.** Use this wizard to create task views, where you include and display the objects that each task requires.

- **Transient Business Component Wizard.** Use this wizard to create a transient business component and fields for the transient business component. Make sure that you use the correct classes, table, and class set. For more information about transient data and transient business component, see *Overview of Transient Data*.

ORACLE

6 Creating Steps and Connectors

Creating Steps and Connectors

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

- *Overview of Step Types*
- *Creating a Start Step*
- *Creating a Task View Step*
- *Creating a Siebel Operation Step*
- *Creating a Business Service Step*
- *Creating a Decision Point*
- *Creating a Subtask Step*
- *Creating a Commit Step*
- *Creating an Error Step*
- *Creating an End Step*
- *Creating a Connector*
- *Creating a Branch Connector*
- *Creating an Error Exception Connector*
- *Creating Arguments for a Task Step*
- *Creating a Task Property*

Overview of Step Types

You use the Palettes pane in the Task Editor to define a task. The following table describes the different step types that are available in the Palettes pane.

Step Type	Description
Start	Defines the input conditions that must be met to run a task instance. For more information, see *Creating a Start Step*.
Task View	Displays a task view in the Siebel client. For more information, see *Creating a Task View Step*.
Siebel Operation	Includes an Insert, Update, or Query operation that interacts with a business component record. For more information, see *Creating a Siebel Operation Step*.
Business Service	Calls a business service that allows you to run a predefined or custom action. For more information, see *Creating a Business Service Step*.

ORACLE

Step Type	Description
Decision Point	Presents one or more decision conditions that the user must evaluate to determine the next step to run. For more information, see *Creating a Decision Point*.
Subtask	Calls a task. The subtask task can be independent of the calling task. For more information, see *Creating a Subtask Step*.
Commit	Explicitly saves task data that Siebel CRM stores in temporary storage to the Siebel database. For more information, see *Creating a Commit Step*.
Error	Creates an error and returns control to the current view. For more information, see *Creating an Error Step*.
End	Indicates the end of the task. For more information, see *Creating an End Step*.

The following table describes the connectors that you can add to define the task flow in the Palettes pane.

Connector Type	Description
Connector	Determines the direction of flow between steps in a task. A connector can be one of the following types: • **Connector.** For more information, see *Creating a Connector*. **Condition..** Includes a condition that must be met to continue through a path. For more information, see *Creating a Branch Connector*. • **Default.** Does not include a condition. If a condition is not met for another connector, then the task flows down the default connector. For more information, see *Creating a Branch Connector*. • **Error Exception.** For more information, see *Creating an Error Exception Connector*. • **User Defined Exception.** This is defined in the following row.
User Defined Exception	Handles deviation from normal processing, such as a system error or a custom error that you define. For more information, see *Creating an Error Exception Connector*.

Related Topics
- Main Elements of the Task Editor
- Using the Development Environment to Develop a Task

Creating a Start Step

A *start step* is a type of Task UI step that indicates the starting point of a task. There can be only one start step in a task. You can optionally create logic and runtime events on the connector that emanates from the start step when you create a task.

ORACLE

To create a start step

- Select the Start step from the Palettes pane, then drag and drop it on the canvas.

  For more information, see *Adding a Step to a Task*.

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 for the task playbar and control how Siebel CRM saves information to the Siebel database. For more information, see *About the Task View*.

To be able to use a task view in a task, the task view must meet the following conditions:

- It must be active
- It must have the same business object as the Task
- It must not already be bound to another task
- It must be of type Task

To create a task view step

1. Select the Task View step from the Palettes pane, then drag and drop it on the canvas.

   For more information, see *Adding a Step to a Task*.

ORACLE

2. In the Properties pane, set the following properties:

    a. Set the Task View property.

       Doing this binds the task view step to a task view. Drill down on (or double-click) the task view step to view and edit the task view. The Task Editor displays an approximation of how Siebel CRM renders the view at runtime.

       The applets and view must already have been created before you can set the Task View property, otherwise the Task Editor will not display any options in the Task View property.

    b. Set the following properties:

       - Display Name
       - Display Name Type

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

    c. Set the following properties as required:

       - Disable Cancel
       - Disable Pause
       - Disable Previous

       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.

    d. Set the Forward Button Type property.

       For more information, see *Task Playbar Validation with Forward Navigation*.

    e. Set the following properties as required:

       - 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* and *Siebel Object Types Reference* .

3. (Optional) Set the Task Step Context property in the Multi Value Property Window (MVPW) pane.

   This 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*.

4. (Optional) If you set the Task Step Context, then add output arguments, as required.

   The task view step does not return an output argument. You use the Output Arguments tab in the MVPW pane 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 MVPW*.

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 Playbar Pause*.

ORACLE

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*.

Using the Display Name with a Looping Task

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 (*).

ORACLE

You can create a Siebel operation step for any business component that is contained in the business object that the task references. If necessary, you may need to add the business component to the business object. For more information on adding business components to business objects, see *Configuring Siebel Business Applications* .

Some business component fields are not available for modification. For more information, see *Business Component Fields That a Task Can Modify*.

To create a Siebel operation step

1. Select the Siebel Operation step from the Palettes pane, then drag and drop it on the canvas.

    For more information, see *Adding a Step to a Task*.
2. In the Properties pane:

    a. Set the Operation property.

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

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

    c. For updates or insertions, ensure that every required field is added to the Siebel operation step.

    For more information, see *Identifying the Business Component Field for a Siebel Operation Step*.
3. (Optional) Create a search specification for the Siebel operation step.

    For more information, see *Defining the Context for a Task Step*.
4. (Optional) In the Multi Value Property Window (MVPW) pane, add input arguments and output arguments.

    For more information, see *Creating an Output Argument for a Task Step*.

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. This allows the user to enter data over a series of task views, without committing the changes until the user has had a chance to finish entering the data, while keeping the task view less cluttered.

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:

1. A Siebel operation step performs an insert operation.
2. A task view step follows the Siebel operation in the first step. 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. For more information, see *Disabling Task Transactions* 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 (MVPW) pane to create a field for a Siebel operation step.

To identify the business component field for a Siebel operation step

1. In the Task Editor (see *Opening the Task Editor*), make sure the Business Component property of the Siebel operation step that you want to modify contains a value.
2. Select the Siebel operation step.
3. In the MVPW pane, click the Fields tab and create a new record.
4. Create a new record:
    - In the Field Name field, select the business component field that this step must update. The drop down 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.
    - Set the Type field. For more information, see *How the Type Field Affects Other Fields in the MVPW*.
5. 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.
6. (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 a few example actions that some predefined business services perform:

- **Notification.** The Outbound Communications Manager 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*
- *Siebel Business Process Framework: Workflow Guide*

To create a business service step

1. Select the Business Service step from the Palettes pane, then drag and drop it on the canvas.

ORACLE

For more information, see *Adding a Step to a Task*.

2. In the Properties pane:

   a. In the Business Service Name property, select the name of the business service that the business service step calls.

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

      The choices that appear for this property depend on the business service that you define in step a.

3. In the Multi Value Property Window (MVPW) pane, create input arguments and output arguments as required.

   For more information, see *How the Type Field Affects Other Fields in the MVPW*.

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. Select the Decision Point step from the Palettes pane, then drag and drop it on the canvas.

   For more information, see *Adding a Step to a Task*.

2. Create a branch connector for the decision point.

   For more information, see *Creating a Branch Connector*.

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 are required for multiple tasks to decrease development and maintenance cost.

- Modularize a task. 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 in the Task Editor.

- Maintain a clean, consistent, and intuitive programming model.

For more information, see *Using a Transient Business Component in a Subtask*.

To create a subtask step

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

   For more information, see *Characteristics of a Subtask*.

2. Select the Subtask step from the Palettes pane, then drag and drop it on the canvas.

   For more information, see *Adding a Step to a Task*.

3. In the Subtask Name property (in the Properties pane), select the subtask that this subtask step calls.

4. (Optional) Create input arguments or output arguments for the subtask.

   For more information, see *How the Type Field Affects Other Fields in the MVPW*.

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 a forward direction.

Comparing a Subtask to a Subprocess

The following table compares a subtask that resides in a task to a subprocess that resides in a Workflow Process.

Object	Task UI	Workflow Process
Process Instance	A parent task 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 do not share task properties. For more information, see *How a Subtask Uses a Task Property*.	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 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*.	The operation of an input argument or output argument is similar to the subprocess step in a Workflow Process.

Subtask Charactieristics

A subtask includes the following characteristics:

- A task can include one or more subtask steps.
- A subtask can include a subtask.

ORACLE

- 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 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*.

- You must explicitly define a task as the parent task or the subtask at design time. Each parent task 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.

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 a commit step to handle an error that occurs during the commit. For example, if a commit attempts to save a record that is missing a required field, you may wish to have the exception step return you to the step that should have collected that required field.

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*, and *Using a Transient Business Component with a Commit Step*.

To create a commit step

- Select the Commit step from the Palettes pane, then drag and drop it on the canvas.

  For more information, see *Adding a Step to a Task*.

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 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 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:

- 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 which opens in the current view.

- Resets the task 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, 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 to handle an expected error.

To create an error step

1. Select the Error step from the Palettes pane, then drag and drop it on the canvas.

   For more information, see *Adding a Step to a Task*.
2. In the Error Code property (in the Properties pane), select a predefined error code from the picklist, or define a custom error message.

   For more information, see *Creating a Custom Error Message*.

Error Handling While a Task Runs

If an exception occurs in a task step while the task runs, and if no exception branch is defined, then Siebel CRM displays a generic error message in a dialog box that opens in 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*.

Error Handling When User Pauses a Task

No exception handling occurs if a user pauses a task. 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.

Error Handling When User Cancels a Task

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, 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. Select the Error step from the Palettes pane, then drag and drop it on the canvas.

   For more information, see *Adding a Step to a Task*.
2. In the Error Code property (in the Properties pane), choose an error code that starts with WF_ERR_CUSTOM.

   For example, WF_ERR_CUSTOM_1.

ORACLE

3. Select the error step and in the Multi Value Property Window (MVPW) pane, 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. Select 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 selected in step b.

      For more information, see *Creating an Input Argument for a Task Step*.

   d. Step off the record to save the changes.

4. Optional. Repeat the previous step 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. Select the End step from the Palettes pane, then drag and drop it on the canvas.

   For more information, see *Adding a Step to a Task*.

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 runs. It stores this value in a task property. It can then pass this value between objects in a task, such as between a subtask and a parent task. For more information, see *About the Task Property*.

Creating a Connector

The *connector* is an object in Task UI that allows you to define the flow between task steps.

To create a connector

1. Go to the Tasks list and locate and select the task you want to modify.

2. Define the flow or path of the task by adding connectors as follows:

   o (Siebel Tools) Select the Connector from the Palettes pane, then drag and drop it on the canvas. Select the connector's start point on the canvas and attach it to a step, then select and move the other end (end point) of the connector and connect it to the next step in the flow.

   o (Web Tools) Use the mouse to draw a connector line from the connector node on one step to the connector node on another step.

3. Select a connector and in the Properties pane:

ORACLE

    a. Enter text in the Label property. This text appears as a label on the connector.

    b. Set the Type property of the connector to Connector (which is the default value).

Other types of connector that you can define include the following:

- **Condition Connector.**. Includes a condition that must be met to continue through a path. For more information, see *Creating a Branch Connector*.
- **Default Connector.** Does not include a condition. If a condition is not met for another connector, then the task flows down the default connector. For more information, see *Creating a Branch Connector*.
- **Error or User Defined Exception Connector.** For more information, see *Creating an Error Exception Connector*.

4. Repeat steps 2 and 3 until every step in the task is connected correctly.

Related Topics

- Creating a Branch Connector
- Creating a Condition on a Branch Connector
- Creating Multiple OR Conditions
- Branching and Parallel Processing
- Creating an Error Exception Connector

Creating a Branch Connector

A *branch connector* is a type of connector in Task UI where you create a condition. Similar to how it operates in a Workflow Process, a branch connector in a task can be conditional or nonconditional. Siebel CRM uses a condition and a decision point to determine the path to follow in the task 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 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 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 flows down the Default connector.

To view an example that includes a condition, see *Revising the Task*.

To create a branch connector

1. Create a connector as described in *Creating a Connector*.

ORACLE

2. Select the connector and in the Properties pane:
    a. Enter or modify the Name property of the branch connector. The branch name must be unique.
    b. Set the Type property of the branch connector to Condition.
       This value for this property is typically Condition (if defining a decision condition on the connector) or Default (if using this connector as an exit route). You can set the Type property to other values as required. For more information, see the description for the task branch object in *Siebel Object Types Reference*.
    c. Create conditions that apply to this branch as described in *Creating a Condition on a Branch Connector*.
    d. In the Comments property, enter text that describes the branch logic.
3. (Optional) Repeat the previous steps for each additional branch connector you want to create.
   If your task must proceed along more than one conditional connector and one default connector, then you must create a separate branch connector for each logical path.
4. Create a default branch connector.

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

You 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 (see *Opening the Task Editor*) for the task you want to modify.
2. Double-click the branch connector where you want to create the condition – the Compose Condition Criteria dialog box opens. The following also opens the Compose Condition Criteria dialog box:
    o (Siebel Tools) Right-click the connector and select Edit Condition.
    o (Web Tools) Select the connector, click the Tasks Menu (cogwheel icon) and select Edit Conditions.
3. In the Compose Condition Criteria dialog box, use the Compare To list to select one of the values described in the following table. The content of this dialog box varies according to the business object that the task references.

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.

ORACLE

Value	Description
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.

4. Select an Operation to evaluate the values.
5. Enter an Object and Field, if necessary.
6. Enter one or more values in the Values window.

   If you enter multiple values, then Siebel CRM uses an OR operator between each value.
7. 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* .
8. Click OK.
9. Close the Task Editor.

Creating Multiple OR Conditions

You can create multiple conditions for each branch connector. 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 can proceed along only one connector. If you create conditions so that the flow can proceed simultaneously along multiple connectors, then Siebel CRM cannot predict runtime behavior.

Creating an Error Exception Connector

The error exception is a type of connector that resides in 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.

To create an error exception

1. Create a connector as shown in *Creating a Connector*.

ORACLE

> **Note:** Each step in your task flow can have one entry connector and one exit connector. Create your error connection first, as shown in the following step, and then connect your step to another step.

2. Select the connector that you created in step 1 and in the Type property of the connector, select one of the following values:

    ○ Error Exception

    ○ User Defined Exception

    The Task UI framework treats a user defined exception the same way it treats an error exception.

3. Double-click the exception connector you just created and in the Compose Condition Criteria dialog box that opens, 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 Arguments for a Task Step

This topic describes how to create the arguments for a task step. It includes the following topics:

- *Creating an Input Argument for a Task Step*
- *Creating an Output Argument for a Task Step*

For more information, see *Arguments of a Task Step*.

Creating an Input Argument for a Task Step

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

To create an input argument for a task step

1. Open the Task Editor (see *Opening the Task Editor*) for the task you want to modify.
2. In the Task Editor, select the step where you want to create an input argument.
3. In the Multi Value Property Window (MVPW) pane, click the Input Arguments tab.

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

4. Create a new record:

    ○ In the Input Argument field, select an input argument. This value is the destination of the input argument. For a Siebel operation step, select the Field Name.

    ○ In the Type field, select a type.

5. Define the remaining fields for the input argument according to the type you defined in the previous step.

    For more information, see *How the Type Field Affects Other Fields in the MVPW*. Some fields are not available. For more information, see *Business Component Fields That a Task Can Modify*.

ORACLE

Creating an Output Argument for 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 (see *Opening the Task Editor*) for the task you want to modify.
2. In the Task Editor, select the step where you want to create an output argument.
3. In the Multi Value Property Window (MVPW) pane, click the Output Arguments tab.
4. Create a new record:

   - In the Property Name field, select a property name. This is the name of the task property that stores the value of the output argument.
   - In the Type field, select a type.

5. Define remaining fields for the output argument according to the type you defined in the previous step.

   For more information, see *How the Type Field Affects Other Fields in the MVPW*.

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. The Siebel Task UI framework automatically defines a set of system task properties when you create a task. You can use these system-defined task properties as well as custom-defined task properties to pass information between steps to meet your design requirements. For more information, see *About the Task Property*.

To create a task property

1. In the Task Editor (see *Opening the Task Editor*), click the canvas – do not click a task step or a connector.

   The Multi Value Property Window (MVPW) pane displays tabs that are specific to the task.
2. In the MVPW pane, click the Task Properties tab.
3. Create a new record:

   - Enter a value in the Name field.
   - Choose a Data Type for the task property.
   - (Optional) In the Default field, assign a default value for the task property.

ORACLE

○ (Optional) Modify the value in the following fields:

  - Access Mode. The default value is R/W.
  - In/Out. The default value is None.

  Modify these values (typically, you do not need to modify them) according to the following:

  - In the Access Mode field, select Read Only.
  - In the In/Out field, select one of the following values: In, Out, or In/Out.

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

○ If you select Integration Object as the Data Type, then choose the Integration Object that Siebel CRM must use for the task property.

○ (Optional) Enter comments in the Comments field, as necessary.

7 Developing a Task

Developing a Task

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

- *Roadmap for Developing a Task*
- *Process of Creating a Task*
- *Developing a Task that Uses Transient Data*

Roadmap for Developing a Task

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

1. Determine improvement requirements and design the task – this step is described in *Task Development Example*.
2. *Process of Creating a Task*
3. *Validating and Inspecting a Task*
4. *Process of Testing a Task*
5. *Administering a Task*

For examples that describe how to develop a task, see *Examples of Developing a Task*. For more information, see *Customizing a Task*.

Process of Creating a Task

This process is a step in *Roadmap for Developing a Task*.

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

1. *Creating a Custom Task*
2. *Diagramming a Task*
3. *Creating a Task View*
4. *Editing the Layout of a Task View*
5. *Binding a Task View to a Task View Step*
6. *Creating a Task Group*
7. *Adding a Task Group to a View*
8. *Refining a Task*

Creating a Custom Task

This task is a step in *Process of Creating a Task*.

ORACLE

You can use the Tasks list or Task Wizard to create a custom task. Since the Task Wizard makes sure you define the minimal set of properties and objects that your task requires, it is strongly recommended that you use the Task Wizard, instead of the Tasks list, to create custom tasks. For more information about task properties that are automatically defined when you create a new task object, see *About the Task Property*.

To create a custom task using the Tasks list

1. In the Object Explorer, click Task.
2. In the Tasks list, create a new task.

   At a minimum, define the following properties:

   - Task Name.
   - Project.
   - Business Object.
   - Indicate if the task is a subtask.

     Never modify the Is Subtask property of a task or a subtask after you create it. For more information, see *Diagramming a Task*.

To create a custom task using the Task Wizard

1. Open an editable Workspace.
2. Start the New Object Wizards (click the magic wand icon) and choose the New Task icon.
3. Click Start.
4. Enter these properties.

   a. Task Name: Name of the Task. It should be unique in the Workspace.
   b. Display Name: Name as it should appear in the Task Pane for the end user to choose.
   c. Business Object: The Name of the Business Object that will handle the data. Type it or choose it from the list.
   d. Transient Business Component: The Name of the Transient Business Component (TBC) that you will use to store data while the Task executes. Tasks do not require a TBC, but if you need one for your Task, it should have already been created to appear in the list. If you want to create the TBC later, leave this field blank and update the Task definition later with your desired TBC name. For more information, see *Determining When to Use a Transient Business Component*.
   e. Is Subtask: If this Task is intended to be a Subtask check this box.
5. When satisfied with your choices click the Next button.
6. The next View displays your choices as read-only so that you may confirm them.
7. Click Finish to create the Task or Previous to adjust the values.
8. The wizard displays the status of your Task's creation. If there were any issues those will be displayed in the pane at the bottom of the wizard view.
9. After clicking Finish, a popup window allows you to click OK to edit the Task definition or Cancel to return to the View you were on before starting the wizard.

Diagramming a Task

This task is a step in *Process of Creating a Task*.

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

To diagram a task

1. Open the Task Editor (see *Opening the Task Editor*) and add one or more logic steps to the task.
2. A task 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*.
3. Add connectors to define the path of the task or task flow.

   For more information, see *Creating a Connector*.
4. Repeat the previous steps to add logic steps and connectors until every step in the task is connected correctly.

Creating a Task View

This task is a step in *Process of Creating a Task*.

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*
- *Guidelines for Designing User Interface Elements*

To create a task view

1. Make sure that the following items that the task view references exist – these objects must exist before you create the task view using the Task Wizard:

   ○ Standard applets.

   ○ Task applets.

   ○ Transient business components. For more information, see *Overview of Transient Data*.

2. Start the Task View Wizard and in the New View dialog box, define the following items and then click Next:

   ○ Project name

   ○ View name

   ○ View title

   ○ Business object of the view

   ○ Upgrade behavior

3. In the View Web Layout - Select Template dialog box, select a Web template for your view and then click Next.
4. In the Web Layout - Applets dialog box, select the applets to display in the task view and then click Next.

   This dialog box lists predefined applets. You can select task applets in a subsequent dialog box.

ORACLE

5. In the Task View - Pick Task dialog box, do one of the following and then click Next:

   o Click Yes to specify a task applet for this view.

     Note that the task applet that you want to associate with this view must already be created. For more information, see *Overview of Transient Data* and *Task Applet*.
   o Click No if you do not need to specify a task applet for this view.

6. If you click No in the previous step (about specifying a task applet), then go to step 9.
7. In the Task View - Select Task dialog box, click the task where Siebel CRM must display the task view and then click Next.
8. In the Task View - Task Applets dialog box, select the applets that Siebel CRM must display in the task view and then click Next.
9. In the Task View - Select Playbar Applet dialog box, select the playbar applet to appear in the header and footer of your view and then click Next.

   You must include at least one playbar applet in a task view. To improve usability, it is recommended that you include only one playbar applet. If you use only one playbar, then leave one of the windows in this dialog box empty. For more information, see *Task Playbar Applet*.

10. In the Finish dialog box, click Finish.

    The new view opens in the Web Layout Editor. The Finish dialog box displays the properties that will be used to create the new view.

11. Edit the layout of the task view.

    For more information, see *Editing the Layout of a Task View*.

Editing the Layout of a Task View

This task is a step in *Process of Creating a Task*.

The Web Layout Editor allows you to edit the mapping that the Task UI framework 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 the Object Explorer, click View.
2. In the Views list, query the Name property for the view you want to modify.
3. Edit the Web layout, as necessary.
4. Preview the view.

   The view in preview mode appears similar to the way it is rendered at runtime in the Siebel client.
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.

ORACLE

   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

This task is a step in *Process of Creating a Task*.

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 the task view. For more information, see *Creating a Task View Step*.

To bind a task view to a task view step

1. Open the Task Editor (see *Opening the Task Editor*) for the task where Siebel CRM must bind a task view.
2. Bind a task view to a task view step as follows:

   ◦ (Siebel Tools) Right-click the task view step, select Bind Task View and in the dialog box that opens, select a task view and then click OK.

     This dialog box lists the task views that you can define for the task. If the task view that you require is not included in the list, then click New to create a new one using the Task View Wizard.

     > **Note:** *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

   ◦ (Web Tools) Select the task view step and in the Properties pane, select a task view from the Task View drop down list.

     The name of the view, that the task view step references, appears as the label for the task view step in the Task Editor.
3. Define properties for the task view step.

   Some properties, such as Disable Cancel, let you specify behavior for the task playbar. The Forward Button Type property lets you specify when to save information to the Siebel database. For more information, see *Task Playbar Applet*.

Conditions for Binding Task Views to Task Steps

You can bind a task view to a task view step if it meets the following conditions:

- The task view is active.

- The task view is of type Task.

- The task view and the task in which the task view will be used has the same Business Object. For example, if a task uses the Contact Business Object as specified by its Business Object property, then the task view that

ORACLE

you want to use in the task view step must also use the same Contact Business Object (also specified by its Business Object property).

- The task view has not already been bound to a different task.

Creating a Task Group

This task is a step in *Process of Creating a Task*.

This topic describes how to create a task group. For more information, see *Task Group* and *Displaying Object Types Used to Develop a Task*.

To create a task group

1. 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*.
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 *Task Group*.

2. Make sure the task group you added in the previous step is selected.
3. Under the Task Group, add a new Task Group Item.
4. 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 appear in the Action Invoked property, so you must define the Type property first.
Action Invoked	Choose the task that you want to start. The Action Invoked property instructs the task to open when the user clicks the link in the context pane. For more information, see *Context Pane*.

ORACLE

Property	Description
Context Business Component	If you placed a check mark in the Require Context BC option in step 1, then select 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 about task order in task groups, see *Task Group*.

Adding a Task Group to a View

This task is a step in *Process of Creating a Task*.

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 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.

For more information, see *Task Group* and *Displaying Object Types Used to Develop a Task*. A number of factors determine how a task appears in the task pane. For more information, see *Task Does Not Appear in the Task Pane*.

To add a task group to a view

1. Select the view where you want to add the task group.
2. Add a new View Task Group under this view.
3. (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 select this application in the Application property.

If there are additional views under which this Task Group should be available, then add the Task Group to those views as well.

Refining a Task

This task is a step in *Process of Creating a Task*.

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

ORACLE

To refine a task

1. Add conditional logic to your task.

You create conditions and values that affect task flow to add conditional logic to a task. 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*.

2. Add a subtask.

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

3. 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*.

4. Create logic to handle an error condition.

You can create logic in your task 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*.

5. 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*.

6. 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, and what work the user finished in the task. For more information, see *Other Options for Customizing a Task*.
- ○ You can use various usability techniques to improve the user experience. For more information, see *Techniques to Improve Task Usability*.

7. Create logic to resume a paused task.

If the user must pause a task 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*.

Developing a Task that Uses Transient Data

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

- *Overview of Transient Data*
- *Determining When to Use a Transient Business Component*
- *Creating a Transient Business Component*
- *Creating a Task Applet*
- *Transferring Data with a Transient Business Component*
- *Guidelines for Using a Transient Business Component*
- *About the Multirecord Transient Business Component*

ORACLE

Overview of Transient Data

Transient data is a type of data that is relevant only in the context of a particular task instance. The answer to the following question that Siebel CRM displays in a task 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. 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. Siebel CRM accesses the records of a transient business component only in the context of a specific task or subtask.

A task 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. 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*.
- 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 that Assists with Adding an Opportunity and an Activity*.
- For an example that does not use transient data, see *Example of Developing a Task that Assists with Adding an Opportunity*.

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.
- Cleared after a commit step or when the user finishes a task.
- 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 CRM 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.

ORACLE

How Siebel CRM Manages Transient Data

The Task UI framework 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 CRM 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 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 finishes.

For more information, see *Creating a Transient Business Component*.

Determining When to Use a Transient Business Component

This topic describes how to determine whether to use a transient business component or not.

If one of the following situations is true, then you must use a transient business component. If none of these situations are true, then do not need to use a transient business component. Use a business component instead.

- A radio button or decision point determines the task flow.

- The task must temporarily store data. For more information, see *Storing Data Temporarily While Gathering the Information Required to Create a Complete Record*.

- 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*.

- 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 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.

For more information, see *Creating a Transient Business Component*.

Creating a Transient Business Component

If your task 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*.

To create a transient business component

1. Determine if your task requires a transient business component.

    If a transient business component is not required by your task, then quit this procedure. For more information, see *Determining When to Use a Transient Business Component*.

2. Start the New Object Wizards (click the magic wand icon), go to the Task tab and create a new Transient Business Component.

3. Configure the properties shown in the following table for the new transient business component.

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.
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*.

4. Click Finish.

    The wizard automatically does the following:

    ○ 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.

5. Create a new Field for the Transient Business Component.

ORACLE

6. Configure the following properties for the new field you created in the previous step:

    o Name

    o Type

    o 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:

    o The Column property is set automatically.

    o The Column and Join properties are read only.

    o You cannot define a multivalue field.

    o The Task UI framework allows a calculated field. It does not assign a column to a calculated field.

7. Repeat step 5 and 6 for each additional field you want to define.

Setting the Type Property of a Business Component

Note the following about setting a business component's Type property:

- 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.

The New Object Wizards 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

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*.

Creating a Task Applet

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

If your task requires a task applet, then you can create it before you create the task view. For more information, see *Creating a Task View*. For an example of creating a task applet, see *Creating the Task Applet*.

To create a task applet

1. Make sure that the following objects exist before you create the task applet:

    ○ The object definition of the task

    ○ The transient business component that the task applet references

2. Start the New Object Wizards (click the magic wand icon), go to the Task tab and create either a new Task Form Applet or Task List Applet.

    > **Note:** *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

3. In the General dialog box that appears, configure the properties shown in the following table for the new task applet.

Property	Description
Project	Select the project that this applet references.
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 will show in the Siebel client. To display no title, then leave the Display Title property empty.
Task	Select the task that this applet references. If you do not specify a task, then you can reuse this applet with multiple tasks. If you specify a task, then you cannot use this applet with any other task.
Upgrade Behavior	Select Preserve. This setting preserves the modification during an upgrade.
Transient Business Component	Select the transient business component that stores data for this applet.

4. Click Next.

ORACLE

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

   To choose a field, select it in the Available Fields window and move it to the Selected Fields window (by using the navigation arrows between the two windows).

6. Click Next.

   The Finish dialog box appears showing the values that will be used to define the new Task Applet. Click Back to return to an earlier dialog and make modifications, as necessary.

7. Click Finish.

   The task applet is created and the Web Template Layout Editor opens.

8. 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. In the Task Editor (see *Opening the Task Editor*), select 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.

2. In the Multi Value Property Window (MVPW) pane, click the Output Arguments tab.

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

4. For the assignment type, use Business Component.

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

1. In the Task Editor (see *Opening the Task Editor*), add a Siebel operation step with the value shown in the following table.

Property	Value
Operation	Update

2. Create an output argument:

   a. Select the Siebel operation step you added in step 2.

   b. In the MVPW pane, click the Output Arguments tab.

   c. Create a new (output argument) record and configure the properties shown in the following table.

Property	Description
Property Name	Select the property that contains the data you want to transfer.

ORACLE

Property	Description
Type	Select Business Component.
Business Component	Select the business component that will receive the data you are transferring.
Business Component Field	Select the business component field that will receive the data you are transferring.

At runtime, this configuration will update the target business component.

Guidelines for Using a Transient Business Component

It is recommended that you adhere to the following guidelines when using a Transient Business Component (TBC):

- If you use a TBC that references a particular Business Object, then you must add that TBC to the Business Object or you will receive an error. Note that a TBC does not require a link to be defined on the Business Object Component.
- Siebel CRM might consider data that resides in a task property as transient data. The source of the data determines how Siebel CRM will display it. Consider the following capabilities when deciding whether to store transient data in a TBC or in a task property:
  - Siebel CRM displays data that it gets from a TBC 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 TBC 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 uses more than one view to get the information it requires to create a complete record, then you must use a transient business component. The task 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 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. You can configure the task to copy data from the transient business component to the business component record when this task finishes.

ORACLE

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 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*.

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*.

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 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*.

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 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. For example, the user might create several plans but near the end of the task 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 finishes or if the user cancels it. If the user pauses the task, then these records remain in temporary storage.

ORACLE

Classes that Siebel CRM Uses with a Transient Business Components

The following table 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 Transient 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.

Type	Description	Class
Single record transient business component	Supports a single record in a task 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 context.	Uses the CSSBCTaskTransientBase class or a class that Siebel CRM derives from the CSSBCTaskTransientBase class.

ORACLE

8   Examples of Developing a Task

Examples of Developing a Task

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

- *Example of Modifying a Predefined Task*
- *Example of Developing a Task that Assists with Adding an Opportunity*
- *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*
- *Example of Developing a Task that Assists with Adding an Account and a Service Request*
- *Example of Developing a Task That Assists with Creating Multiple Opportunities*

Examples of Modifying a Predefined Task

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

1. *Adding the Product Data for the Asset to Contract Task*
2. *Verifying the Functionality of the Asset to Contract Task*

The work that is required to modify a predefined task varies depending on the task you want to modify. In this example, 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:

```
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 deliver the task, add product data, and then verify task functionality. A significantly different configuration might be required to configure a different predefined task.

For more information, see *About Predefined Tasks*.

Adding the Product Data for the Asset to Contract Task

This task is a step in *Example of Modifying a Predefined Task*.

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 with the values shown in the following table.

Field	Description
Name	Choose Technical Support 24x7.

ORACLE

Field	Description
Type	Choose Service.
Service Product	Make sure this field includes a check mark.

3. Make sure the following product is selected:

   `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 on today's date.

   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 with the values shown 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*.

In this topic, you verify the functionality of the Asset to Contract task. For more information, see *Verifying Functionality of a Task*.

To verify the functionality of the Asset to Contract task

1. In the Siebel client, navigate to the Accounts screen and then click Tasks (the clipboard icon).
2. Verify that the task pane displays the task 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 that the Use Existing Agreement radio button includes a check mark and then click Next.
6. In the Select an Agreement for Revision task view, select an agreement from the list and then click Next.
7. In the Agreement Detail task view, note that Siebel CRM enters data in some fields according to the choice you made in the previous step and then click Next.
8. In the Add Service Package task view, click Continue.

   The choice you make here alters the task flow as follows:

   - 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*.

   You can chose one of the other options on a subsequent test of this task to view how the task logic varies according to the condition that is met.
9. In 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. In the Submit Agreement for Approval task view, click Continue and then click Next.
11. In 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 that Assists with Adding an Opportunity

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

1. Create or open a Developer Workspace – see *Create or Open a Workspace*
2. *Creating the Task*
3. *Creating the Applet*
4. *Creating the Task View*
5. *Binding the Task View to the Task View Step*
6. *Controlling Playbar Applet Buttons*
7. *Creating the Task Group*
8. Validate the task in the Developer Workspace – see *Validating a Task*
9. (Optional) Preview and inspect the task in the Developer Workspace – see *Inspecting a Task*
10. Deliver the task - see *Delivering a Task*.
11. *Verifying the Task*

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*

- *Task Applet*

Creating the Task

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity*.

In this topic, you create the task and define the task flow.

To create the task

1. In the Workspace Explorer, open your Developer Workspace – for more information, see *Create or Open a Workspace*.
2. In the Object Explorer, click Project.
3. In the Projects list, create a new project named Opportunity Task.

   You can use this project to support this development effort. For more information, see *Using Siebel Tools*.
4. Create a new task with the properties shown in the following table. For more information, see *Creating a Custom Task*.

Property	Description
Project	Select Opportunity Task.

ORACLE

Property	Description
Task Name	Enter (type in) Create New Opportunity.
Display Name	Enter (type in) Create New Opportunity.
Business Object	Select Opportunity.
Transient Business Component	Leave this field empty.
Subtask	Make sure this option does not include a check mark.

5. Add the following steps and connectors until your task resembles the flow illustrated in the following image:
   a. A Start step.
   b. A Create Opportunity step.
   c. An Opportunity Task View step.
   d. An End step.
   e. The following connectors: A connector between step (a) and (b), step (b) and (c), and step (c) and (d).
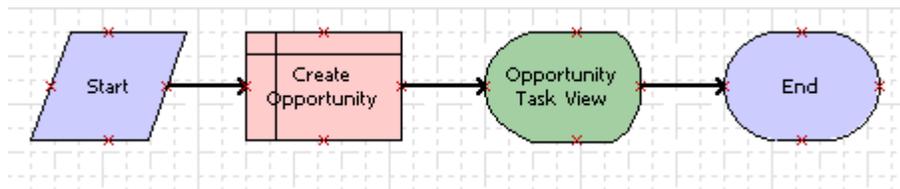
The label for the task view step appears after you bind the view (in a subsequent step in this example). For more information, see *Diagramming a Task*.

6. Select the Create Opportunity step and in the Properties pane, configure the properties for the step as shown in the following table.

Property	Value
Business Component	Opportunity
Defer Write Record For more information, see *About the Defer Write Record Property*.	TRUE
Operation	Insert

ORACLE

7. Close the Task Editor.

Creating the Applet

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity*.

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

To create the applet

1. Start the New Object Wizards (click the magic wand icon), go to the Applets tab, click the Form Applet icon, and then click OK.

   Note: *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

2. In the General dialog box that appears, configure the properties shown in the following table and then click Next.

Property	Description
Project	Select Opportunity Task.
Name	Enter (type in) Opportunity Task Form Applet.
Display Title	Enter Opportunity.
Business Component	Select Opportunity.
Upgrade Behavior	Select Admin.
Use Grid Layout	Make sure this option includes a check mark.

3. In the Web Layout - Form dialog box, accept the default Edit Mode and then click Next.
4. 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

5. In the Web Layout - Fields dialog box, move fields from the Available Controls list to the Selected Controls list as required and then click Next.

ORACLE

6. In the Finish dialog box, review your settings and then Click Finish.

   The new applet is created and opens in the Web Layout Editor.

7. In the Applet Web Template window, reposition the controls and labels until the applet layout resembles that shown in the following table.

Field Label	Corresponding Control (Field Input Box)	Field Label	Corresponding Control (Field Input Box)
Name:	<Field name>	Primary Revenue Amount:	<Primary Revenue Amount>
Description:	<Field description>	Primary Revenue Close Date:	<Primary Revenue Close Date>

8. Choose the Description control and in the Properties pane, set the property shown in the following table.

Property	Value
HTML Type	TextArea

   When you set the HTML Type property, a scroll bar appears along the side of the Description control.

9. Close the Applet Web Template window.

Creating the Task View

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity*.

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*.

To create the task view

1. Start the Task View Wizard and in the New View dialog box, configure the properties shown in the following table.

Property	Value
Project	Opportunity Task
View Name	Opportunity Task View
View Title	Opportunity

ORACLE

Property	Value
Business Object	Opportunity
Upgrade Behavior	Admin

> **Note:** *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

2. In the View Web Layout - Select Template dialog box, select the following template: View Detail (Parent with Pointer).
3. In the Web Layout - Applets dialog box, select 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*.
4. In the Task View - Pick Task dialog box, click No, and then click Next.
5. In the Task View - Select Playbar Applet dialog box, select the Task Playbar Applet - Bottom applet as the footer playbar applet, and then click Next.
6. Click Finish, verify that the layout is correct, and then close the Web Layout Editor.

For more information about creating a task view, see *Creating a Task View*.

Binding the Task View to the Task View Step

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity*.

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 View.

  For more information, see *Binding a Task View to a Task View Step*.

Controlling the Buttons of the Playbar Applet

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity*.

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 playbar applet buttons

1. In the Task Editor, select the Opportunity Task View step.

ORACLE

2. In the Properties pane, configure the properties shown in the following table.

Property	Description
Disable Previous	Select TRUE.
Name - String Override	Enter Opportunity Details.
Forward Button Type	Select Submit.

3. 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 that Assists with Adding an Opportunity*.

In this topic, you create a task group for the Create New Opportunity task. 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*.

To create the task group

1. Create the task group as follows:

    a. Create a new task group with the properties shown in the following table.

Property	Value
Name	Opportunity Task Group
Project	Opportunity Task
Display Name - String Override	Opportunity Tasks

ORACLE

b. Create a new task group item with the properties shown in the following table.

Property	Value
Action Invoked	Create New Opportunity
Type	Task
Sequence	1

For more information, see *Creating a Task Group*.

2. Configure the task group to display in a specific view.

   o In the Object Explorer, click View and then select Opportunity List View.

   o Select Opportunity Task Group from the pop-up applet that appears when you drill down on the Task Group property in the View Task Group list.

   o Configure the property shown in the following table.

Property	Value
Sequence	2

For more information, see *Adding a Task Group to a View*.

Verifying the Task

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity*.

In this topic, you verify that the task delivers the required functionality. For more information, see *Verifying Functionality of a Task*.

> **Note:** Responsibilities and their relationship to Views can be Workspace enabled in your development environment. If they have been Workspace enabled in your development environment and you are working in that environment, then you can only modify them in an editable Workspace. You do not need an editable Workspace to create and edit Responsibilities and their relationship to Views in your Production environment.

To verify the task

1. In the Siebel client, navigate to the Administration - Application screen, then the:
   - Tasks view to add the responsibilities for new tasks.
   - Views view to add the responsibilities for new views.

   For more information, see *Adding a Responsibility to a Task*.
2. In the Siebel client, click Tasks (the clipboard icon) and then verify that the task pane does not display in the Opportunity Tasks task group. You defined this task to display only when the Opportunity List View is the current view.
3. Navigate to the Opportunities screen, then the Opportunities List view.
4. Verify that the task pane now displays the Opportunity Tasks task group.
5. In the task pane, click the Create New Opportunity link, 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*.

   If the layout requires any adjustment, then you can rapid prototype the applet layout. For more information, see *Rapid Prototyping on Applet Layout*.
6. Verify the following:
   - The Pause, Submit, and Cancel buttons are active.
   - The Previous button is disabled.
7. Enter a value in the following fields, and then click Submit:
   - Name
   - Description
   - Primary Revenue Amount

   Siebel CRM closes the task and then displays the standard Opportunity List View.
8. Verify that Siebel CRM includes the opportunity you defined in the previous step in the opportunity list and that the opportunity fields include the values you entered in the task.

Example of Developing a Task that Assists with Adding an Opportunity and an Activity

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

1. Create or open a Developer Workspace – see *Create or Open a Workspace*
2. *Creating the Task*
3. *Creating the Transient Business Component*
4. *Creating the Picklist*
5. *Creating the Task Applet*
6. *Adding the Task Applet to the View*
7. *Creating the Task View*

ORACLE

8. *Revising the Task*
9. *Creating the Condition Criteria*
10. Validate the task in the Developer Workspace – see *Validating a Task*
11. (Optional) Preview and inspect the task in the Developer Workspace – see *Inspecting a Task*
12. Deliver the task - see *Delivering a Task*.
13. *Verifying the Functionality of the Task*

In this example, the user uses a task 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*.

Creating the Task

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*.

In this topic, you create the task and define the task flow.

To create the task

- Create and test the task that is described in *Example of Developing a Task that Assists with Adding an Opportunity*.

Creating the Transient Business Component

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*.

This task 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*
- *Radio Button Group*

To create the transient business component

1. Start the New Object Wizards (click the magic wand icon), go to the Task tab and create a new Transient Business Component with the properties shown in the following table.

   Note: *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

Property	Description
Project	Select Opportunity Task.
Name	Enter (type in) TBC for Opportunity.
MultiRecord TBC checkbox	Make sure this property does not contain a check mark.

ORACLE

Property	Description

For more information, see *Creating a Transient Business Component*.

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

3. In the Fields list, create a new field with the properties shown in the following table.

Property	Description
Name	Enter Create Activity.
Picklist	Select a picklist. This choice only acts as a placeholder. You define the actual picklist in a subsequent step in this procedure.
Text Length	Enter (type in) 10.
Type	Select DTYPE_TEXT.

4. Make sure the Create Activity record is selected 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 with the properties shown in 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 that Assists with Adding an Opportunity and an Activity*.

In this topic, you create the picklist that the Create Activity field references.

To create the picklist

1. Start the New Object Wizards (click the magic wand icon), go to the General tab, click Picklist and then click OK.

   Note: *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

ORACLE

2. Use the Pick List Wizard to create the picklist:

  a. In the Pick List dialog box, configure the properties shown in the following table and then click Next.

Property	Value
Project	Opportunity Task
Business Component	TBC for Opportunity
Field	Create Activity

  b. In the Pick List Type dialog box, accept the default Static picklist 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:

   - 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:

   - Enter the following text for the name in the first window: Values for Yes No Prompt.
   - Enter the following value in the middle window: Yes
   - Click Enter.

   The Task Flow Editor displays the following value in the Current values window: Yes.
  f. In the List of Values dialog box:

   - Enter the following text in the middle window: No.
   - Click Enter.

   The Task Flow Editor 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.
3. In the Object Explorer, click Business Component.
4. In the Business Components list, query the Name property for the following value: TBC for Opportunity.
5. In the Object Explorer, expand the Business Component tree, and then click Field.
6. In the Fields list, query the Name property for the Create Activity field that you defined previously.
7. 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 that Assists with Adding an Opportunity and an Activity*.

In this topic, you create the task applet that allows the user to add an activity. For more information, see *Task Applet*.

To create the task applet

1. Use the Task Form Applet Wizard to create a new task applet:

   a. Start the New Object Wizards (click the magic wand icon), go to the Task tab, select the Task Form Applet icon, and then click OK.

   b. In the General dialog box that appears, configure the properties shown in the following table.

Property	Value
Project	Opportunity Task
Applet Name	Opportunity Task Next Step
Display Title	Next Step
Task Applet is Associated With	Create New Opportunity
Upgrade Behavior	Admin
Transient Business Component	TBC for Opportunity

   Note: *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

   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*.

2. In the Applet Web Template window, reposition the Create Activity control and label so that they are side by side as shown in the following table.

Create Activity Label	Create Activity Control Options
Create Activity:	There are two options as follows: ○ Yes

ORACLE

Create Activity Label	Create Activity Control Options
	○ No

The No option is not visible when the Task Flow Editor shows the template. You must resize the Create Activity control to show the No option.

3. Close the Web Layout Editor.

Adding the Task Applet to the View

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*.

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 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 with the properties shown in the following table.

Property	Value
Name	Opportunity Task Next Step
Item Identifier	6
Applet	Opportunity Task Next Step
Applet Mode	Edit

5. In the View Web Template Items list, choose the item named Task Playbar Applet - Bottom, and then set the property shown in the following table.

Property	Value
Item Identifier	8

ORACLE

Creating the Task View

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*.

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*.

To create the task view

1. Start the Task View Wizard and in the New View dialog box, configure the properties shown in the following table.

Property	Value
Project	Opportunity Task
View Name	Opportunity Activity Task View
View Title	Create Activity
Business Object	Opportunity
Upgrade Behavior	Admin

> **Note:** *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

2. In the View Web Layout - Select Template dialog box, select the following template: View Detail (Parent with Pointer).
3. In the Web Layout - Applets dialog box, move the following applets from the Available Applets list to the Selected Applets list:

    o Opportunity Task Form Applet

    o Activity Form Applet

    The Opportunity Task Form Applet is the custom applet you created in *Creating the Applet* 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.
4. In the Task View - Pick Task dialog box, click No.
5. In the Task View - Select Playbar Applet dialog box, select the Task Playbar Applet - Bottom applet as the footer playbar applet.
6. Click Finish, verify that the layout is correct, and then close the Web Layout Editor.

Revising the Task

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*.

In this topic, you revise the task so that it allows the user to add an activity.

To revise the task

1. In the Tasks list, query the Task Name property for Create New Opportunity task.
2. In the Workspace Explorer, create a new Developer Workspace (see *Create or Open a Workspace*), open the task and modify it (by adding step d, e, and f) until it resembles the flow shown in the following image.

   a. A Start step.
   b. A Create Opportunity step.
   c. An Opportunity Task View step.
   d. A Create Activity decision step.
   e. A Create Activity step.
   f. An Opportunity Activity Task View step.
   g. An End step.
   h. The following connectors: A connector between step (a) and (b), step (b) and (c), step (c) and (d), step (d) and (e) labelled Yes, step (e) and (f), step (f) and (g), and (d) and (g) labelled No.
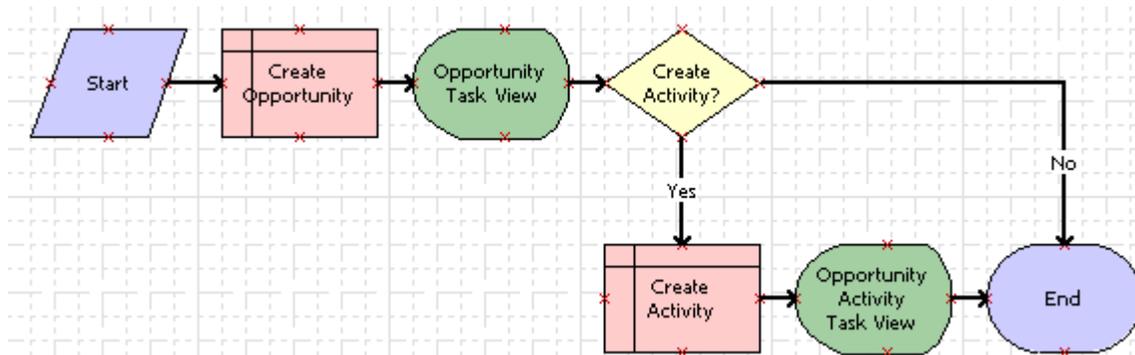
   The Task Editor displays the label of the Opportunity Activity Task View after you bind the view in a subsequent step in this example. For more information, see *Diagramming a Task*.

   For more information about Workspaces, see *Using Siebel Tools* .
3. Select the Opportunity Task View step and in the Properties pane, define the property shown in the following table.

Property	Value
Forward Button Type	Next

4. Select the Create Activity step and in the Properties pane, define the properties shown in the following table.

ORACLE

Property	Value
Business Component	Action
Defer Write Record	True
Operation	Insert

5. Select the Opportunity Activity Task View step and in the Properties pane, define the properties shown in the following table.

Property	Value
Display Name - String Override	Create Activity
Forward Button Type	Submit
Task View	Opportunity Activity Task View

6. Select the No connector and in the Properties pane, define the property shown in the following table.

Property	Value
Type	Default

Creating the Condition Criteria

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*.

In this topic, you create the condition criteria.

To create the condition criteria

1. Select the Yes connector, click the Tasks Menu (the cogwheel icon), and then select Edit Conditions.

   For more information, see *Creating a Branch Connector*.

ORACLE

2. In the Compose Condition Criteria dialog box that opens, do the following:

   a. In the Compare To picklist, select 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 runtime value of a field from the transient business component (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 select in the Compare To picklist.

   b. In the Object picklist, select the TBC for Opportunity.

   c. In the Operation picklist, accept the following option: 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 runtime value of the Create Activity field of the business component.

   d. In the Field picklist, select Create Activity.

   e. In the Values section, click New, select Yes in the Pick Value dialog that opens, and then click OK.

   f. Click Add.

      The condition is added to the Conditions list in the Compose Condition Criteria dialog box. In this example, you add only a single condition (but another condition can be added if required in a different situation). The following table shows the values that should be set for this example condition in the Compose Condition Criteria dialog box.

Field Label	Field Control	Field Value
Compare To	Drop down list	Business Component
Operation	Drop down list	All Must Match (Ignore Case)
Object	Drop down list	TBC for Opportunity
Field	Drop down list	Create Activity
Values	Text box	Yes

      At runtime, if the Create Activity field contains the Value:

        - **Yes**, then Siebel CRM directs the task flow down the Yes branch.

        - **No**, then Siebel CRM directs the task flow down the No branch.

   g. Click OK.

3. Close the Task Editor.

Verifying the Functionality of the Task

This task is a step in *Example of Developing a Task that Assists with Adding an Opportunity and an Activity*.

In this topic, you verify that the task delivers the required functionality. For more information, see *Verifying Functionality of a Task*.

To verify the functionality of the task

1. In the Siebel client, navigate to the Administration - Application screen, then the Views view and add the responsibilities for the new view (Opportunity Activity Task View). For more information, see *Adding a Responsibility to a Task*.
2. In the Siebel client, navigate to the Opportunities screen and then the Opportunities List view.
3. Click Tasks (the clipboard icon) and 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*
   - 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 first portion of the view includes the same opportunity details that were displayed in the Enter Opportunity Details view.
   b. Verify that the second portion of the view displays fields that allow the user to provide details about the activity.
   c. Verify that the Opportunity field data in the second portion of the view 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 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.

8. In the Opportunity Name field, drill down on the Test Opportunity record.
9. Go to 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 in Step 6.

ORACLE

Example of Developing a Task that Assists with Adding an Account and a Service Request

The example in this topic describes how to develop a task that assists the user with adding an account and a service request for that account. To develop this example, perform the following tasks:

1. Create or open a Developer Workspace – see *Create or Open a Workspace*
2. *Creating the Task*
3. *Creating the Applets*
4. *Creating the Task Views*
5. *Binding Task Views to Task View Steps*
6. *Controlling the Buttons of the Playbar Applet*
7. *Creating Chapters for the Task*
8. *Creating the Task Group*
9. Validate the task in the Developer Workspace – see *Validating a Task*
10. (Optional) Preview and inspect the task in the Developer Workspace – see *Inspecting a Task*
11. Deliver the task - see *Delivering a Task*.
12. *Verifying the Task*
13. (Optional) *Rapid Prototyping on Applet Layout*

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*
- *Task Applet*

Creating the Task

This task is a step in *Example of Developing a Task that Assists with Adding an Account and a Service Request*.

In this topic, you create the task and define the task flow.

To create the task

1. In the Workspace Explorer, open your Developer Workspace – for more information, see *Create or Open a Workspace*.
2. In the Object Explorer, click Project.
3. In the Projects list, create a new project named Account Task.

   You can use this project to support this development effort. For more information, see *Using Siebel Tools* .
4. Create a new task with the values shown in the following table. For more information, see *Creating a Custom Task*.

Property	Description
Project	Select Account Task.

ORACLE

Property	Description
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*.

5. Add the following steps and connectors until your task resembles the flow illustrated in the following image:

   a. A Start step.
   b. A Create Account step.
   c. An Account Entry Task View step.
   d. A Create SR step.
   e. An SR Entry Task View.
   f. An End step.
   g. The following connectors: A connector between step (a) and (b), step (b) and (c), step (c) and (d), step (d) and (e), and (e) and (f).
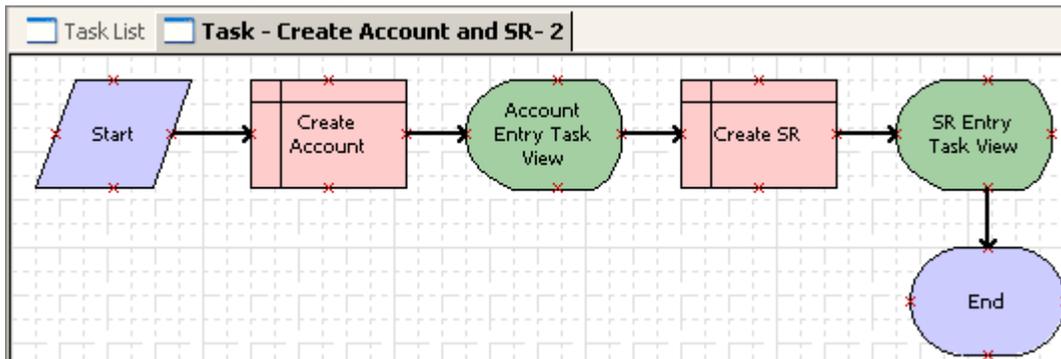
The labels for the task view steps appear after you bind the views (in a subsequent step in this example). For more information, see *Diagramming a Task*.

6. Select the Create Account step and in the Properties pane, configure the properties shown in the following table.

Property	Value
Business Component	Account

ORACLE

Property	Value
Defer Write Record For more information, see *About the Defer Write Record Property*.	TRUE
Operation	Insert

7. Select the Create SR step and in the Properties pane, configure the properties shown in the following table.

Property	Value
Business Component	Service Request
Defer Write Record	TRUE
Operation	Insert

8. Close the Task Editor.

Creating the Applets

This task is a step in *Example of Developing a Task that Assists with Adding an Account and a Service Request*.

In this topic, you reuse standard applets. For more information, see *Reusing a Standard Applet*.

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 the controls and their labels so the layout of the (Account Entry for Task UI) applet resembles that shown in the following table.

Label, Column 1	Corresponding Control, Column 2	Label, Column 3	Corresponding Control, Column 4
New, Edit, Delete	Buttons	Save, Reset, Cancel	Buttons
Account Name:	Text Box	Account Team:	Text Box

ORACLE

Label, Column 1	Corresponding Control, Column 2	Label, Column 3	Corresponding Control, Column 4
Address:	Text Box	Main Phone #:	Text Box
City:	Text Box	Status:	Text Box
Zip Code:	Text Box	Account Type:	Text Box

    e. Close the Applet Web Template window.

    For more information, see *Reusing a Standard Applet*.

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 the controls and their labels so the layout of the (SR Entry for Task UI) applet resembles that shown in the following table.

Label, Column 1	Corresponding Control, Column 2	Label, Column 3	Corresponding Control, Column 4
New, Edit, Delete	Buttons	Save, Reset, Cancel	Buttons
Account:	Text Box	Area:	Text Box
SR #:	Text Box	Subarea:	Text Box
Owner:	Text Box	Priority:	Text Box
Date Opened:	Text Box	Severity:	Text Box
Status:	Text Box	Substatus:	Text Box
Summary:	Text Box	(Summary) Text Box continued	(Summary) Text Box continued
Description:	Text Box	(Summary) Text Box continued	(Summary) Text Box continued

ORACLE

Label, Column 1	Corresponding Control, Column 2	Label, Column 3	Corresponding Control, Column 4

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 before the grid.

e. Close the Applet Web Template window.

Creating the Task Views

This task is a step in *Example of Developing a Task that Assists with Adding an Account and a Service Request*.

In this topic, you create task views that display the applets you created in *Creating the Applets*.

Note: *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

To create the task views

1. Use the Task View Wizard to create a task view that allows the user to enter account data:

   a. In the New View dialog box, configure the values shown 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*.

ORACLE

    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 footer 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*.

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, configure the values shown 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, select the View Basic template.

    c. In the Web Layout - Applets dialog box, select the following applet: SR Entry for Task.

The SR Entry for Task applet is one of the custom applets you created in *Creating the Applets*.

    d. In the Task View - Pick Task dialog box, click No.

    e. In the Task View - Select Playbar Applet dialog box, select the following applet as the footer 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*.

Binding Task Views to Task View Steps

This task is a step in *Example of Developing a Task that Assists with Adding an Account and a Service Request*.

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*.

ORACLE

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 that Assists with Adding an Account and a Service Request*.

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, select the Account Entry Task View step.
2. In the Properties pane, define the properties shown in the following table.

Property	Value
Disable Previous	TRUE
Display Name - String Override	Enter Account Details
Forward Button Type	Next

3. Select the SR Entry Task View step and in the Properties pane, define the properties shown 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

This task is a step in *Example of Developing a Task that Assists with Adding an Account and a Service Request*.

In this topic, you create two new chapters for the Create New Account task. The task chapter is an optional UI element that assists user navigation. For more information, see *Task Chapter*.

To create chapters for the task

1. Create two new chapters with the values shown in the following table.

Name	Display Name - String Override	Sequence
Chapter 1	Enter New Account	10
Chapter 2	Enter New SR	20

2. Assign the following steps to Chapter 1:

   ○ Start
   ○ Create Account
   ○ Account Entry

3. Assign the following steps to Chapter 2:

   ○ Create SR
   ○ SR Entry Task View
   ○ End

4. Close the Task Editor.

For more information, see *Creating a Task Chapter*.

Creating the Task Group

This task is a step in *Example of Developing a Task that Assists with Adding an Account and a Service Request*.

In this topic, you create a task group for the Create New Account task. 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*.

To create the task group

1. Create the task groups:

   a. Create a new task group with the values shown in the following table.

Property	Value
Name	Account Tasks
Project	Account Task UI

ORACLE

Property	Value
Display Name - String Override	Account Tasks

b. Create a new task group item with the values shown in the following table.

Property	Value
Action Invoked	Create Account and SR
Type	Task
Sequence	1

For more information, see *Creating a Task Group*.

2. Configure the task group to display across views.

Select Account Tasks in the Task Groups pop-up applet when you define the Task Group property in the View Task Groups list, and then define the property shown in the following table.

Property	Value
Sequence	2

For more information, see *Adding a Task Group to a View*.

Verifying the Task

This task is a step in *Example of Developing a Task that Assists with Adding an Account and a Service Request*.

In this topic, you verify that the task delivers the required functionality. For more information, see *Verifying Functionality of a Task*.

Note: Responsibilities and their relationship to Views can be Workspace enabled in your development environment. If they have been Workspace enabled in your development environment and you are working in that environment, then you can only modify them in an editable Workspace. You do not need an editable Workspace to create and edit Responsibilities and their relationship to Views in your Production environment.

To verify the task

1. In the Siebel client, navigate to the Administration - Application screen, then the:
    - Tasks view to add the responsibilities for the new task (Create Account and SR).
    - Views view to add the responsibilities for the new views (Account Entry Task View and SR Entry Task View).

    For more information, see *Adding a Responsibility to a Task*.
2. In the Siebel client, click Tasks (the clipboard icon) and then verify that the task pane displays a task group named Account Tasks.

    To verify that Siebel CRM displays the task 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 Create Account and SR link.
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*.
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 *Example of a Siebel Task View*.

    If the layout requires adjustment, then you can rapid prototype the layout of the applet. For more information, see *Rapid Prototyping on Applet Layout*.
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 the previous step.
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 view and query the Account Name field for the name you entered previously.
14. Verify that Siebel CRM displays the new account.
15. Drill down 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 previously.

ORACLE

Rapid Prototyping on Applet Layout

| **Note:** This task applies only to Siebel Tools.

This task is an optional step in *Example of Developing a Task that Assists with Adding an Account and a Service Request*.

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 do rapid prototyping on applet layout

1. Open Siebel Tools or Web Tools.
2. In the Workspace Explorer, create a Developer Workspace – for more information, see *Create or Open a Workspace*.
3. In the Applet Web Templates list, select the record you want to modify and then choose Edit Web Layout.
4. Modify the layout and close the Applet Web Template when finished.
5. Deliver the Workspace – this is similar to *Delivering a Task*.
6. Log in to the Siebel client, navigate to the task you are modifying and then examine your modifications.

Example of Developing a Task that Assists with Creating Multiple Opportunities

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

1. Create or open a Developer Workspace – see *Create or Open a Workspace*
2. *Creating the Transient Business Component*
3. *Creating the Business Service*
4. *Creating the Task*
5. *Creating the Task Applet*
6. *Creating the Task View*
7. *Binding the Task View to the Task View Step*
8. *Controlling the Buttons of the Playbar Applet*
9. *Creating the Task Group*
10. Validate the task in the Developer Workspace – see *Validating a Task*
11. (Optional) Preview and inspect the task in the Developer Workspace – see *Inspecting a Task*
12. Deliver the task - see *Delivering a Task*.
13. *Verifying the Task*

Creating the Transient Business Component

This task is a step in *Example of Developing a Task That Assists with Creating Multiple Opportunities*.

In this topic, you create the transient business component for the task. For more information, see *Overview of Transient Data*.

ORACLE

In this topic, you create the task and define the task flow.

To create the transient business component

1. In the Workspace Explorer, open the Developer Workspace – for more information, see *Create or Open a Workspace*.
2. In the Object Explorer, click Project.
3. In the Projects list, create a new project named Multiple Opportunity Task.

   You can use this project to support this development effort. For more information, see *Using Siebel Tools* .
4. Use the New Object Wizards to create a new object:

   > **Note:** *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

   o In the New Object Wizards dialog box, go to the Task tab, click the Transient BusComp icon, and then click OK.
   o In the New Business Component dialog box, define the new business component with the values shown in the following table.

Property	Description
Project	Choose Multiple Opportunity Task.
Name	Enter MultiRecordTBC.
MultiRecordTBC	Make sure this property includes a check mark.

   o Click Finish.
5. In the Object Explorer, expand the Business Component tree, and then click Single Value Field.
6. In the Single Value Fields list, create a new field with the values shown in 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 That Assists with Creating Multiple Opportunities*.

In this topic, you create the business service that the task UI calls.

ORACLE

To create the business service

1. In the Object Explorer, click Business Service and create a new record with the values shown in the following table.

Property	Value
Name	MultiTBC_Insert
Project	Multiple Opportunity Task UI
Class	CSSService
Display Name - String Override	MultiTBC_Insert

2. Select the MultiTBC_Insert business service record that you created in the previous step, click the Applet Menu (the cogwheel icon) and then select Edit Server Scripts.
3. In the Scripting Language dialog that opens, select eScript and then click OK.

   The script editing window opens with the Service_PreInvokeMethod function selected.
4. Position the cursor in the script editing window and do the following:
   - Right-click, and then select Select All.
   - Right-click, and then select Cut.
5. Paste the following script into 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);
```

ORACLE

```
//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());
}
}
currentBC = null;
currentTBC = null;
currentBO = null;
return (CancelOperation);
}
```

6. Close the script editing window and then click Yes in the Confirm dialog that opens.
7. In the Object Explorer, expand the Business Service tree and then click Business Service Method.
8. In the Business Service Methods list, create a new business service with the values shown in the following table.

Property	Value
Name	InsertRecordToTBC
Display Name - String Override	MultiTBCInsertBSM

Creating the Task

This task is a step in *Example of Developing a Task That Assists with Creating Multiple Opportunities*.

In this topic, you create the task and define the task flow.

To create the task

1. In the Workspace Explorer, open your Developer Workspace and create a new task with the values shown in the following table.

Property	Description
Project	Choose Multiple Opportunity Task.

ORACLE

Property	Description
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* and *Create or Open a Workspace*.

2. Add the following steps and connectors until your task resembles the flow illustrated in the following image:

   a. A Start step.
   b. An Insert Opportunity step.
   c. A Mullti Opportunity Task View step.
   d. An End step.
   e. The following connectors: A connector between step (a) and (b), step (b) and (c), and step (c) and (d).
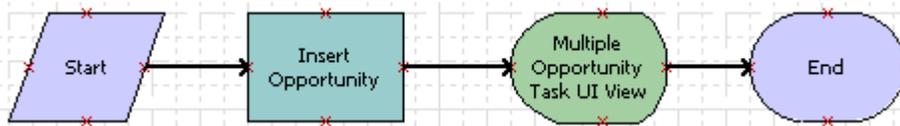
The label for the task view step appears after you bind the view (in a subsequent step in this example). For more information, see *Diagramming a Task*.

3. Select the Insert Opportunity step and in the Properties pane, configure the properties shown in the following table.

Property	Value
Business Service Name	MultiTBC_Insert
Business Service Method	InsertRecordToTBC

4. Close the Task Editor.

Creating the Task Applet

This task is a step in *Example of Developing a Task That Assists with Creating Multiple Opportunities*.

ORACLE

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*.

To create the task applet

1. Start the New Object Wizards (click the magic wand icon), go to the Task tab, select the Task List Applet icon and then click OK.

   > **Note:** *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

   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.

2. In the General dialog box, configure the properties shown in 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

3. 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.
4. In the Finish dialog box, review your settings and then Click Finish.

   Siebel creates the new applet and then opens the Web Layout Editor.
5. Close the Applet Web Template window.

Creating the Task View

This task is a step in *Example of Developing a Task That Assists with Creating Multiple Opportunities*.

In this topic, you create a task view that displays the applet that allows the user to enter multiple opportunities. For more information, see *Creating a Task View*.

ORACLE

To create the task view

1. Start the Task View Wizard and in the New View dialog box, set the values shown in the following table.

   > **Note:** *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

Property	Value
Project	Multiple Opportunity Task UI
View Name	Multiple Opportunity Task UI View
View Title	Multiple Opportunities
Business Object	Opportunity
Upgrade Behavior	Admin

2. In the View Web Layout - Select Template dialog box, choose the following template:

   View Detail (Parent with Pointer)
3. In the Web Layout - Applets dialog box, click Next.

   Do not move any applets to the Selected Applets window.
4. In the Task View - Select Task dialog box, choose Create Multiple Opportunities, and then click Next.
5. 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*.
6. Click Next.
7. In the Task View - Select Playbar Applet dialog box, choose the Task Playbar Applet - Bottom applet as the footer playbar applet, and then click Next.
8. 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 That Assists with Creating Multiple Opportunities*.

In this topic, you bind a task view to the task view step.

ORACLE

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*.

Controlling the Buttons of the Playbar Applet

This task is a step in *Example of Developing a Task That Assists with Creating Multiple Opportunities*.

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, select the Opportunity Task View step.
2. In the Properties pane, define the properties shown in the following table.

Property	Value
Disable Previous	TRUE
Display Name - String Override	Enter Multiple Opportunities
Forward Button Type	Submit

3. Close the Task Editor.

Creating the Task Group

This task is a step in *Example of Developing a Task That Assists with Creating Multiple Opportunities*.

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*.

To create the task group

1. Create the task group:
   a. Create a new task group with the values shown in the following table.

Property	Value
Name	Multiple Opportunities Task Group

ORACLE

Property	Value
Project	Multiple Opportunity Task UI
Display Name - String Override	Multiple Opportunities

b. Create a new task group item with the values shown in the following table.

Property	Value
Action Invoked	Create Multiple Opportunities
Type	Task
Sequence	1

For more information, see *Creating a Task Group*.

2. Configure the task group to display in a specific view:

   o In the Views list, query the Name property for Opportunity List View.

   o 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 shown in the following table.

Property	Value
Sequence	2

For more information, see *Adding a Task Group to a View*.

Verifying the Task

This task is a step in *Example of Developing a Task That Assists with Creating Multiple Opportunities*.

In this topic, you verify that the task delivers the required functionality. For more information, see *Verifying Functionality of a Task*.

To verify the task

1. In the Siebel client, navigate to the Administration - Application screen, then the:

ORACLE

- ○ Tasks view to add the responsibilities for the new task (Create Multiple Opportunities).

- ○ Views view to add the responsibilities for the new view (Multiple Opportunity Task UI View).

For more information, see *Adding a Responsibility to a Task*.

2. In the Siebel client, click Tasks (the clipboard icon) and then verify that the task pane does not display the Multiple Opportunities task group. You defined this task to display only when the Opportunity List View is the current view.

3. Navigate to the Opportunities screen, then the Opportunities List view.

4. Verify that the task pane now displays the Multiple Opportunities task group.

5. 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*.

   If the layout requires adjustment, then you can rapid prototype the layout of the applet. For more information, see *Rapid Prototyping on Applet Layout*.

6. Verify that the following buttons are active:

   - ○ Pause

   - ○ Submit

   - ○ Cancel

7. Verify that the Previous button is disabled.

8. Enter multiple opportunities and then click Submit.

   Siebel CRM closes the task and then displays the standard Opportunity List View.

9. Verify that the opportunity list includes the opportunities you created in the previous step.

ORACLE

9 Testing, Troubleshooting, and Deploying a Task

Testing, Troubleshooting, and Deploying a Task

This chapter describes how to test, troubleshoot, and deploy tasks. It includes the following topics:

- *Process of Testing a Task*
- *Troubleshooting a Task*
- *Transferring Tasks to the Siebel Mobile Web Client*

Process of Testing a Task

This process is a step in *Roadmap for Developing a Task*.

You must test a task before you deliver it. Testing a task verifies that the task you release runs properly and does not cause conflicts with any other existing task. To test a task, perform the following tasks:

1. Validate the task – see *Validating a Task*.
2. (Optional) Run the Task Debugger – see *Using the Task Debugger*.
3. (Optional) Preview and inspect the task – see *Inspecting a Task*.
4. Verify task functionality – see *Verifying Functionality of a Task*.
5. (Optional) Troubleshoot the task – see *Troubleshooting a Task*.
6. Modify data collected during testing – see *Modifying How Siebel CRM Logs Data During Testing*.

Validating a Task

This task is a step in *Process of Testing a Task*.

The *Validate Tool* is an error correction tool that enforces the semantic consistency of a task. You can use the Validate Tool before you inspect and deliver a task to ensure that there are no configuration errors. For example, you can use the Validate Tool to make sure an error process does not include an error process. The following procedure describes how to validate a task.

To validate a task

1. In the Object Explorer, locate and select the task that you want to validate.
2. Click the Applet Menu (the cogwheel icon) and then select the Validate option.
3. In the Validate dialog box that opens, click Options.
4. In the Validation Options dialog box that opens, review the validation rules that are available and specify which rules to enforce or ignore during validation:
   a. Select/deselect the check box next to each rule that you want to enforce/ignore.
   b. (Optional) Click Advanced Options to open the Advanced Options dialog where you can select objects to exclude from (object type, Workspace or batch) validation.

ORACLE

    c. (Optional) Once the list of rules to enforce or ignore is set, click Save to save the information to a text file. This helps if you need to review or share the information later with your team.

    d. (Optional) Select the check box next to the following options as required:

        - **Do Not Report Warnings.** Select this option to report only errors and ignore warnings.

        - **Abort Validation After <x_number> Errors.** Select this option and then enter a number (for example 5) so that object validation will stop after <x_number> of errors are identified.

    e. Click OK.

    Note that if you select Ignore All or deselect all the rules in the Validation Options dialog, then you have chosen to ignore the rules for this task and will receive the following message:

```
There are no rules currently enforced. You may update the list
of rules to be enforced from the 'Options' button.
```

5. In the Validation dialog box, click Start to start the validation process.

    Siebel starts validating the task and returns the results in the Validation dialog when finished:

       ○ For example:

```
Total tests failed: 0 or
```

```
Total tests failed: 3
```

       ○ Validation Errors and Warnings appear in the Validate list. Each row in the Validate list identifies a rule violation for the task. Not every error is fatal. Some errors are only warnings. For more information, see *Errors That the Validate Tool Detects*.

       ○ Select an error in the Validate list and then click Go To to drill down on the object that causes the error.

       ○ (Optional) Click Save As to save the validation results to a `validation.log` file on your local machine.

       Depending on your browser naming conventions, the validation results will be saved to `validation(1).log` if *validation.log* already exists or `validation(2).log` if *validation(1).log* already exists, and so on.

    For more information about validating objects including batch validation, see *Using Siebel Tools* .

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 the following table.

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.
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.

ORACLE

Symptom or Error Message	Solution
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 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 contains.

Using the Task Debugger

This task is a step in *Process of Testing a Task*.

The *Task Debugger* is a tool that you can run to test a task. 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. The debugger displays and modifies these values while you use the Task UI framework.

Making the Debug Mode Available

You can step through the task 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 menu 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 debug mode available when using Siebel Web Client

1. In the Siebel client, navigate to the Administration - Server Configuration screen, then the Servers view.
2. In the Siebel Servers list, select 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 (which appears after 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*.

To make debug mode available when using Siebel Developer Client

1. Make sure Siebel Developer Client is closed.
2. 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:

ORACLE

```
D:\Siebel\bin\enu
```

3. In the InfraUIFramework section, set the EnableRestrictedMenu parameter to TRUE, then save and close the configuration file.

   ```
   EnableRestrictedMenu = TRUE
   ```

   If EnableRestrictedMenu is not in the InfraUIFramework section, then add it.
4. Restart the client.
5. Run the Task Debugger.

   For more information, see *Running the Task Debugger*.

   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.

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.

To run the task debugger

1. Log in to the Siebel client.
2. From the Tools menu, select the Debug Mode menu item.
3. Click Open Task Pane to open the task pane.

   The Open Task Pane icon appears next to the Site Map icon.
4. In the task pane, drill down on the task that you want to debug.

   Siebel CRM opens the task and then runs the start step of the task. It also opens the Task Properties watch window for the task, which includes the properties for the step that it most recently run. A line titled Last Action appears under the name of the task you are debugging. This line indicates the last navigation action that was performed. For example, Next.
5. Click Continue in the watch window.

   Siebel CRM runs the next step in the task. 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, click one of the playbar buttons in the task view.
7. Repeat the previous two steps 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, do one of the following:
   - Click Stop in the watch window.
   - Click Cancel in a task view.

Verifying Functionality of a Task

This task is a step in *Process of Testing a Task*.

ORACLE

To finish testing for a task, 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*.

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*
- *Setting Log Levels on the Siebel Client*
- *Collecting Task Timestamp Metrics*
- *Collecting Task Property Metrics*
- *Disabling Task Transactions*

Using Logging on the Siebel Server

Event logging provides a convenient, nonintrusive way to trace components that require further examination. Similar to Siebel Workflow, the Siebel Task UI framework 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, select the component you want to trace. For example: Siebel Call Center Object Manager (ENU).
3. To view the configurable event types for the component that you selected in the previous step, 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*.
   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. Open the prompt from a directory on the Siebel Server.

ORACLE

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

   Run the following command for this example from the 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:

```
TskExec TskState 3 0000000243fc1350:0 2006-02-22 23:49:49 Task state transition changes : Action invoked:
'Navigate', Current State: 'Navigate', Next State: 'Navigate'.
TskNav Oper 3 0000000243fc1350:0 2006-02-22 23:49:49 Task engine requested to navigate to next view.
TskNav Oper 3 0000000243fc1350:0 2006-02-22 23:49:49 Task engine requested to navigate to next step: 'Task
 View 1'.
TskNav PathChange 3 0000000243fc1350:0 2006-02-22 23:49:49 Pushing frame to stack : '1*05*Start0*1*00*'.
TskNav PathChange 3 0000000243fc1350:0 2006-02-22 23:49:49 Pushing 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 -
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (0) : TYPE =
WfTaskViewInfo
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (0) : ViewName =
RequiredFieldView
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (0) : child count
 =
0TskPresInfo ViewInfo 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (0) : child count =
 0
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (1) : TYPE =
WfTaskPlaybarInfo
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (1) : Submit =
 HIDDEN
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (1) : Next =
 ENABLED
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (1) : Prev =
 ENABLED
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (1) : Pause =
 ENABLED
TskPresInfo ViewInfo 3 0000000243fc1350:0 2006-02-22 23:49:49 Task presentation view info (1) : Cancel =
 ENABLED
```

Collecting Task Timestamp Metrics

You can configure Siebel CRM to collect the timestamp metrics for a task after you deploy the task by delivering the Workspace. For more information, see *Delivering a Task* and *About Task Metrics*.

To collect timestamp metrics for a task

1. Open the Siebel client.

ORACLE

2. Navigate to the Administration - Business Process screen, then the Task Monitoring Configuration view.
3. Set the Analytics Level field to one of the following values:
   - Timestamp
   - All

These values allow Siebel CRM to collect timestamp metrics at runtime.

Collecting Task Property Metrics

You can configure Siebel CRM to collect the property metrics of a task after you deploy it by delivering the Workspace. For more information, see *Delivering a Task* and *About Task Metrics*.

To collect property metrics of a task

1. Locate the task where you want to collect property metrics.
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	Select the required metric from the list of predefined metric names.
Property Name	Select a task property from the list of properties that are defined for the task. The runtime value of the metric is set to the runtime value of the task property that Siebel CRM references. Modify the value only for customizable metrics and not for system metrics, whose values are set by the task.
Inactive	Select 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. Deliver the task – see *Delivering a Task*.
5. Open the Siebel client.
6. Navigate to the Administration - Business Process screen, then the Task Monitoring Configuration 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 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
   ```

Disabling Task Transactions

In some testing situations it might be beneficial to temporarily disable transactions for a task. You can disable task transactions at the business component level and at the task level.

ORACLE

—

For more information, see *About Task Transaction*.

To disable transactions at the task level

1. Locate the task that you want to modify.
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 references. For more information, see *Disabling Task Transactions*.

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 Used to Develop a Task*.
2. In the Business Components list, query the Name property for the business component that is involved with the task transaction.
3. In the Object Explorer, expand the Business Component tree, and then click Business Component User Prop.
4. 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
     ```

5. 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 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 the following procedure).

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.

If you set the Transactional property for a task 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.

For more information, see *About the Defer Write Record Property*.

ORACLE

Troubleshooting a Task

This topic describes how to troubleshoot problems that you might encounter when you develop a task UI. It includes the following topics:

- *Task Does Not Appear in the Task Pane*
- *Record Context Is Lost*
- *Build View Errors Occur with Task*
- *Troubleshooting Other Errors*

Task Does Not Display in the Task Pane

If Siebel CRM does not display your task in the task pane, then do the following:

1. Make sure you add the responsibility that you assign to the user to the task. For more information, see *Adding a Responsibility to a Task*.
2. Make sure the task references a view task group. Siebel CRM displays a task group that 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*.
3. 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*.
4. Make sure the parent task group references a view.
5. Make sure object definitions exist in the repository for the following items:

   - Task
   - Task group
   - Metadata
6. To deploy a task, make sure that it is not marked as Inactive in the Repository and deliver it to its parent Integration Workspace.
7. Make sure you properly defined the Context Business Component property. For more information, see *Task Group*.
8. For a task to show up in a given Siebel CRM application, such as Siebel Call Center, make sure that it is associated with a View associated with a Screen in that application. For more information, see *Adding a Task Group to a View*.

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.

ORACLE

- 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.
- 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 step, depending on the value of the Retain View Search Spec property.

Specialized Code That Starts a Task

If any of the following conditions apply, then specialized code might start a task and might also 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 occur.

To debug these situations, do the following:

- 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.

ORACLE

Build View Errors Occur with Task

The following build view errors might occur if the user navigates through a task and if the mode for this task 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.

- 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. To resolve the problem, look for it in the Symptom or Error Message column in the following table. For more information, see *Overview of Transient Data*.

Symptom or Error Message	Solution
A text field does not display properly when a task runs. The task 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.	Select the task that Siebel CRM links to the applet.
A task step includes a forward button type of Submit.	Select a view step and modify the type to Submit. For more information, see *Task Playbar Validation with Forward Navigation*.
Siebel CRM does not list a task in the Application - Administration screen, Tasks view.	Deliver the task to the Integration Workspace.
A transient business component is missing when a task 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:

ORACLE

Symptom or Error Message	Solution
	• Make sure you registered the views or applets in the Views view of the Application - Administration screen. • Make sure your current Workspace context includes the Task definition. • Make sure the responsibility for the user possesses the visibility to and can run the task. For more information, see *Adding a Responsibility to a Task*.
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 View. For more information, see *Adding a Task Group to a View*.
Siebel CRM does not display the task in the task group.	Add the task to the task group and then deliver your changes.
Siebel CRM does not display the task group or the name of the task 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*.
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*.
A condition does not include a condition that matches a run-time value.	For more information, see *Creating a Branch Connector*.

Transferring Tasks to the Siebel Mobile Web Client

Siebel Task UI data is transferred from Siebel Server to the Siebel Mobile Web Client when database initialization runs on the mobile client. Subsequent Siebel Server-side changes to Siebel Task UI are pushed to the Siebel Mobile client via the Siebel Anywhere Repository Upgrade Kit.

To roll back the changes, you must first roll back the changes on the (Siebel) server side, then re-generate the Siebel Anywhere Repository Upgrade Kit with the rolled back version and re-distribute it on the server side.

Siebel Mobile client synchronization detects new changes, facilitates database synchronization between the Siebel Server and Siebel Mobile client and updates the Tasks in the Siebel Mobile client.

To receive the Paused Task Inbox Items, from Siebel Server-to-Siebel Mobile client (and vice versa), you must set the replication level for the task.

ORACLE

Replication Level

Setting the replication level for a task, transfers the Task Inbox Items for the mobile client user (from Siebel Server to the mobile client, and vice versa) whenever the Siebel Mobile client synchronizes. During synchronization, the following happens:

- It replicates to the Siebel Mobile Web Client a task inbox item that the user creates on the Siebel Server.
- It replicates to the Siebel Server a task inbox item that the user creates on the Siebel Mobile Web Client.

Note: Setting the replication level does not automatically transfer Siebel Task Repository changes to the Siebel Mobile client. To get the Siebel Repository changes, you must create a Siebel Anywhere Repository Upgrade Kit. For more information on how to create an Upgrade Kit, see *Siebel Anywhere Administration Guide* .

To set the replication level

1. In the Siebel client, navigate to the Administration - Business Process screen, then the Task Monitoring Configuration view.
2. In the Tasks list, query the Name field for the record that you want to replicate.

   For example, search for the name of the task.
3. Set the Replication field of the record to one of the following as required:

   - **Regional.** For a Siebel Mobile Web Client that connects only to a regional database.
   - **All.** For a Siebel Mobile Web Client that connects to a regional or a local database.

ORACLE

10 Administering a Task

Administering a Task

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

- *Adding a Responsibility to a Task*
- *Using the Task Instance Monitor*
- *Using Task Reports*
- *Allowing Task Transfer*
- *Transferring a Task Instance*
- *Deleting a Task Instance From the Inbox*
- *Removing Temporary Data After a Task Finishes*
- *Configuring Siebel CRM to Resolve Task Transaction Conflicts*

This chapter is a step in *Roadmap for Developing a Task*.

> **Note:** If a task is active in the Design Time Repository (DR), it will automatically be migrated to downstream Runtime Repository (RR) environments, and be available to any calling process, including runtime events.

Adding a Responsibility to a Task

The Registered Task Administration view shows whether a user can run, transfer, or delete a task. You must add the responsibility that Siebel CRM assigns to the user to the registered task so that the user can access the task.

The way you administer access control for a task 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

1. Log in to the Siebel client.
2. Register the task:
   a. Navigate to the Administration - Application screen, then the Tasks view.
   b. In the Registered Tasks list, click New.
   c. In the Task Name field, select the task that you want to register and then click OK.

      Siebel CRM displays a list that includes deployed tasks.
3. Add a responsibility to the task you registered in the previous step:
   a. Make sure the task you registered is selected 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 and then click OK.

    If you typically log in with administrator privileges, then add the Siebel Administrator responsibility for testing purposes.

    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 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*, and *How Task UI Uses the Dashboard and Universal Inbox*.

4. In the Registered Tasks list, click Clear Cache.

5. Log out of the Siebel client.

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 you want to monitor:

    a. Navigate to the Administration - Business Process screen, then the Task Monitoring Configuration view.

    b. In the Monitoring Configuration list, select the record you want to monitor.

    c. In the Monitoring Level field, select a monitoring level.

    Siebel CRM sets the monitoring level for every task to None, by default. You must set the monitoring level to a value other than None to collect data for a task. For more information, see *Task Instance Monitor Monitoring Levels*.

3. Run the task that you set up to monitor in the previous step.

4. Navigate to the Administration - Business Process screen, then the Task Instance Monitor view.

5. In the Task Name field, select the task you set up to monitor previously.

    The Task Instance Monitor displays information only for tasks that are active. All fields are read-only.

6. Examine the information about the task instance.

    For more information, see *Task Instance Monitor Fields*.

7. To examine the information about the task steps for this task instance, do the following:

    a. Click the Task Step Instances tab.

    This tab lists all the steps that constitute the task. It includes information about each task step, including the Start Time, End Time, Status, and so on.

    b. Review the information about the process properties for the step.

    The step you select in the Task Step Instances list references these process properties.

ORACLE

Task Instance Monitor Monitoring Levels

The following table describes the monitoring levels you can adjust in the task instance monitor. Siebel CRM uses the monitoring level that you set for each task, even if the monitoring level is different for a subtask. For example, assume you set the monitoring level to Progress for the parent task 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
- Does not display a new record for any new instance of the subtask

If a task has an error step and if a task instance of the task runs the error step, then Siebel CRM still displays a record for the task instance according to the monitoring level you set.

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 the relevant step in *Using the Task Instance Monitor*.
Debug	Displays the same information as the Detail monitoring level.

Task Instance Monitor Fields

The following table describes the 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.

ORACLE

Field	Description
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.
Instance Id	A value that uniquely identifies an instance of a task or subtask.
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 Status=In Error, then Siebel CRM displays the task in a different font color (that is, red).
Task Id	A value that uniquely identifies a task or subtask.
Task Name	The value that appears in the Display Name property for the task.
Workspace Name	The name of the Workspace.
Workspace Version	The Workspace version number.

Removing a Task Instance from the Task Instance Monitor

If a Task instance contains a status of Stopped, Completed, Completed Abnormal or In Error, then you can remove it from the Task Instance Monitor view. If you must remove a Task instance that is paused, then you must first resume and complete or cancel the instance. This is not specific to instances of a particular Task. It applies to all instances of any Task instance that are in the previously mentioned statuses.

> **Note:** Task Instances that are **Stopped** or **In Error** are deleted regardless of the date you specify. All other status codes will use your specified date.

To remove a Task Instance from Task Instance Monitor

1. Navigate to the Administration - Business Process screen, Task Instance Monitor view, then the Task Process Instances view.
2. Click Purge in the Task Process Instances Applet.
3. In the Task Instance Monitor Purge dialog box that appears, specify a date.
4. Click Purge.

   The Task Instances with the said status and before the specified date are removed from the Task Instance Monitor and will no longer appear in the Task Instance Monitor view.

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 *Task Instance Monitor Fields*.

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. Set the monitoring level and then run the task.

   For more information, see *Using the Task Instance Monitor*.
2. Log in to the Siebel client with administrator privileges.
3. Navigate to the Administration - Business Process screen, then the Task Reports view.
4. Click Reports (the Reports button.
5. Select an output format for the report - options include the following:

   o Task Report - MHTML

   o Task Report - PDF

   o Task Report - PPT

   o Task Report - RTF

6. In the File Download dialog box, select one of the following options:

   o **Open** to open the report in a separate window according to the output format you selected in the previous step.

   o **Save** to save the report to a location that you specify.

ORACLE

Allowing Task Transfer

You can transfer a task to another user by default. If you disable task transfer in Siebel CRM, then the owner field becomes read-only and the Transfer button is disabled for the current task item selected 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, then the Tasks view.
3. In the Registered Tasks list, select the task where you want to control task transfer.
4. In the Responsibilities list, select the responsibility where you want to allow task transfer, and then make sure that the Allow Transfer option includes a check mark.

   To disable task transfer for the selected responsibility, deselect the Allow Transfer check box.

Transferring a Task Instance

The owner of a task instance can use the Universal Inbox to transfer a task.

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*.

To transfer a task instance

1. In the Siebel client, navigate to the Inbox screen and then the Inbox Items List view.
2. Select the task you want to 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.
5. In Transfer To dialog box that opens, select the owner to whom you want to transfer the task instance in the Owner field.
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 the inbox
   - Displays the task instance in the inbox of the new owner
   - Enters a check mark in the Completed field in the My Inbox Items list
   - Adds the task instance to the My Completed Items list

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, then the Tasks view.
2. Select the task that you want to delete.
3. Make sure the Allow Delete option includes a check mark.
4. Navigate to the Inbox screen, then the Inbox Items List view.
5. Select the task that you want to delete.
6. Click the Detail tab and then click Delete.

    Siebel CRM removes the task instance from the inbox.

Removing Temporary Data After a Task 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*.

To remove temporary data after a task finishes

1. Add a business service step to your task immediately after a commit step, with the values shown in 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 the previous step.

ORACLE

Siebel CRM does the following:

- o If you specify a value for the Task Id input argument, then Siebel CRM deletes only temporary data for the task that the Task Id identifies.
- o 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*.

Configuring Siebel CRM to Resolve Duplicate Conflicts

The following procedure shows 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. Display the Business Component User Prop object type.
   For more information, see *Displaying Object Types Used to Develop a Task*.
2. In the Object Explorer, click Business Component.
3. In the Business Components list, locate the business component you want to modify.
4. Expand the Business Component tree in the Object Explorer and then click Business Component User Prop.
5. In the Business Component User Properties list, add a new record with the value shown in the following table.

Name	Description
Dup Conflict In Task	Enter one of the following values: o **Resolve.** Siebel CRM writes the duplicate record with the Conflict Id field set to the value of Id, by default. o **Fail.** Siebel CRM stops the task transaction. o **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.** This 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**. This is caused by one of the following reasons:
  - o Siebel CRM deletes a record in a task transaction, and then updates this record outside the task transaction.

ORACLE

o   Siebel CRM updates a record in a task transaction, and then deletes this record outside the task
      transaction.

Siebel CRM typically detects an update conflict or a delete conflict during a commit operation or when it receives data in
the task transaction.

To configure Siebel CRM to resolve update and delete conflicts

- Create a task property with the value shown in the following table. For more information, see *Creating a Task Property*.

Property	Description
On Conflict	Use one of the following values: o **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. o **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.

ORACLE

11 Customizing a Task

Customizing a Task

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

- *Starting a Task*
- *Resuming a Paused Task*
- *Creating a Subtask*
- *Defining the Context for a Task Step*
- *Creating a Task Event*
- *Using a Business Service Step to Call a Workflow Process*
- *Other Options for Customizing a Task*

Starting a Task

A task 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*
- *Creating a Menu Item to Start a Task*
- *Creating an iHelp Link to Start a Task*
- *Creating a Workflow Process to Start a Task*
- *Creating a Script to Start a Task*

Creating a Button to Start a Task

You can create a button on an applet that users can click to start a task, as shown in the following procedure. For more information, see *Configuring Siebel Business Applications* .

To create a button to start a task

1. 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 Used to Develop a Task*.
2. In the Applets list, query for the applet where you want to add the button (that will start a task).
3. In the Object Explorer, expand the Applet tree, and then click Control.

ORACLE

4. In the Controls list, add a new control with the values shown in 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 in LaunchTask, which is case sensitive and all one word.

5. Make sure the control you defined previously is still selected.
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 with the values shown in the following table.

Property	Description
Name	Select Task Name
Value	Enter the name of the task.

Creating a Menu Item to Start a Task

You can create a menu item on an applet that the user can click to start a task. For more information, see *Configuring Siebel Business Applications* .

To create a menu item to start a task

1. Under Commands in the Object Explorer, create a new command with the values shown in the following table.

Property	Description
Name	Type in a name for the command (which describes the logic that the command performs).
Business Service	Select Task UI Service (SWE).
	CAUTION: 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.

ORACLE

Property	Description
Target	Select Server.
Method	Select LaunchTask.
Method Argument	Enter the name of the task.

The task to open is defined in the properties of the command, so you must define a new command for each task.

2. Add a menu item to the applet menu that calls the command that is defined for the task.

Creating an iHelp Link to Start a Task

You can use an iHelp link to start a task. The following link types are available in the iHelp pane:

- An iHelp item
- A task

To create an iHelp link to start a task

1. In the Siebel client, navigate to the Administration - iHelp screen, 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, select an iHelp item.
4. Click Revise and then click the More Info tab.
5. In the Related Task field, select a task.

   To display a link for a task 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*.
7. Click the iHelp icon to view the link that resides in the iHelp item that Siebel CRM displays to start the task.

Creating a Workflow Process to Start a Task

You can start a long-running Workflow Process that starts a task. For more information about how to do this, see *Siebel Business Process Framework: Workflow Guide* .

ORACLE

Creating a Script to Start a Task

You can use a browser script or a server script to start a task. Your script calls the Task UI Service business service. It then passes the name of the task to the LaunchTaskFromScript business service method to start the task. Processing varies depending on the type of script that you use as follows:

- **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 calls the server script in the context of the Object Manager at the required pre- or post-event.

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 and makes sure the user possesses the license and responsibility to run the task. For more information, see *Adding a Responsibility to a Task*.
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 UI framework, designating it as the context business component.
3. If any of the following situations exist, then an error is created:
    - The task requires a business component that is not the same as the context business component.
    - The task is not active in the current Workspace context.
4. If Siebel CRM does not encounter an error, then it runs the task and opens the task pane.

Note the following restrictions when using a script with a task:

- Calling a task 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. It cannot pass any other parameters.
- You can use a script only to start a task. You cannot use a script to interact with a task in any other way.

For more information, see *About Event Handling*.

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)
{
 var returnValue = "ContinueOperation";
 try
 {
 if (name == "Test")
 {
 var inputPropSet;
 var outputPropSet;
 var taskUIsvc;

 inputPropSet = theApplication().NewPropertySet();

 outputPropSet = theApplication().NewPropertySet();

 taskUIsvc = theApplication().GetService("Task UI Service (SWE)");
```

ORACLE

```
        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 this script --!>


        outputPropSet = taskUIsvc.InvokeMethod("LaunchTaskFromScript",inputPropSet);

        returnValue = "CancelOperation";
        }
        }
        finally
        {
        inputPropSet = null;
        outputPropSet = null;
        taskUIsvc = null;
        }
        return(returnValue);
    }
```

Example of a Server Script

In this example, the server script opens the Create a Contact task UI:

```
    function WebApplet_PreInvokeMethod (MethodName)
    {
     var returnValue = ContinueOperation;
     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 this script --!>

     taskUIsvc.InvokeMethod("LaunchTaskFromScript",inputPropSet,outputPropSet);

     returnValue = CancelOperation;
     }
     }
     finally
     {
     inputPropSet = null;
     outputPropSet = null;
     taskUIsvc = null;
     }
     return(returnValue);
    }
```

ORACLE

Resuming a Paused Task

This topic describes the configuration options to resume a paused task and includes the following information:

- *Creating an Association that Allows the User to Resume or Transfer a Paused Task*
- *Process of Creating a View that Allows a User to Transfer a Paused Task*
- *Modifying a Task to Display a Message that is Specific to a Task Instance*

For more information, see the following topics:

- *Task Playbar Pause*
- *How Task UI Uses the Dashboard and Universal Inbox*
- *Allowing Task Transfer*
- *Transferring a Task Instance*

Creating an Association that Allows the User to Resume or Transfer a Paused Task

This topic describes how to configure a task instance to reference a business component instance. A user can pause a task. 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, begins with a task 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 tasks 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

1. Determine where to locate the business service step in the task.

   For more information, see *Locating the Business Service Step in a Task*.
2. Add a business service step to the task using values described in the following table.

Property	Value
Business Service Name	Task Administration
Business Service Method	Associate

ORACLE

3. Make sure the business service step you added in the previous step is still chosen.

4. In the Multi Value Property Window (MVPW) pane, 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 references.
Type	Literal
Value	Enter the ROW_ID of the business component record that the task references.

5. In the MVPW pane, 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 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*.

Locating the Business Service Step in a Task

The task 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 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 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 with a Long-Running Workflow Process

If a long-running Workflow Process creates a task but never starts it, then Siebel CRM cannot associate this task with a business object. To avoid this situation, you must configure Siebel CRM to start the task manually or automatically.

ORACLE

Process of Creating a View that Allows a User to Transfer a Paused Task

To create a view that allows a user to transfer a paused task, perform the following tasks:

1. *Creating a New Link to Support Task Transfer*
2. *Modifying the Business Object to Support Task Transfer*
3. *Creating a Standard View to Support Task Transfer*
4. *Modifying a Screen to Support Task Transfer*

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*. 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	Select the business component where you want to use task transfer.
Child Business Component	Select UInbox Item Context.
Source Field	Select Id.
Destination Field	Select Item Context Id.
No Update	Select TRUE.
No Delete	Select TRUE.
No Insert	Select TRUE.
Search Specification	Enter the following code:

ORACLE

Property	Description

	`(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:
	
	    o 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*.

3. Select the new link, click the Applet Menu (the cogwheel icon), select the Validate option, 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*.

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 want to 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	Select UInbox Item Context.
Child Business Component	Select the new link.

5. Select the new business object, click the Applet Menu (the cogwheel icon), select the Validate option, 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*.

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. Start the New Object Wizards, go to the General tab, select View and then click OK.

   Note: *Task Wizards* and New Object Wizards are currently supported in Siebel Tools only – they are not supported in Web Tools.

ORACLE

2. Follow the prompts in the wizard to define the new view with the values shown in the following table.

Property	Description
Business Object	Select the business object that you modified in the *Modifying the Business Object to Support Task Transfer* topic.
View Name	Enter a descriptive name for the view. For example, ObjectName - Paused Tasks.
Upgrade Behavior	Select Preserve. This setting preserves this modification during an upgrade.
Web Template	Select a master and detail template. For example, View Basic.
Master Applet	Select a master applet according to the master business component that is defined.
Child Applet	Select Task Item Context List Applet.

3. Edit the Web layout, as necessary.
4. Select the new view, click the Applet Menu (the cogwheel icon), select the Validate option, 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*. 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 want to 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	Select the view that you modified in the *Creating a Standard View to Support Task Transfer* topic.
Sequence	Define a sequence that correctly positions this view on the site map.
Type	Select Detail View.
Parent Category	Select a value.

Property	Description
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	Select TRUE.
Display In Site Map	Select TRUE.
Upgrade Behavior	Select Preserve. This setting preserves this modification during an upgrade.

5. Select the new screen view record, click the Applet Menu (the cogwheel icon), select the Validate option, and then click Start.
6. Compile your modifications and then deploy the task UI by delivering the workspace.
7. Make sure you can access the new view while you use the responsibilities.

Modifying a Task 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.

- A timestamp.

You can modify a task 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. When the user pauses or finishes a task, 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 property.

- The Instance Identifier property is limited to 200 characters in length.

- You can define an instance identifier only on a task step where you can define an output argument. For more information, see *Arguments of a Task Step*.

- If the user clicks Pause on the first view, then Siebel CRM does not display information from the Instance Identifier.

ORACLE

To modify a task to display a message that is specific to a task instance

1. Add an output argument to a task step.
2. In the MVPW pane, 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 that 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 to create a subtask, then make sure that the Create As a Subtask option includes a check mark. For more information, see *Creating a Subtask Step*.

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. In the Task Editor (see *Opening the Task Editor*), choose a Siebel operation step or a task view step.
2. In the Multi Value Property Window (MVPW) pane, go to the Task Step Context tab and create a new record.
3. Enter a Name for the task step context, and then choose a Type.

   If you set Type to Expression, then enter the name of a business component in the Expression Business Component field.
4. 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.

   o 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`.
   o 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]
   ```

ORACLE

The Expression Builder does not examine the format so make sure you enter an expression that includes the correct format.

5. Optional. Create a filter business component.

For more information, see *How Siebel CRM Uses a Filter Business Component*.

Example of Defining the Context for a Task Step

The following image illustrates an example that includes an update operation.
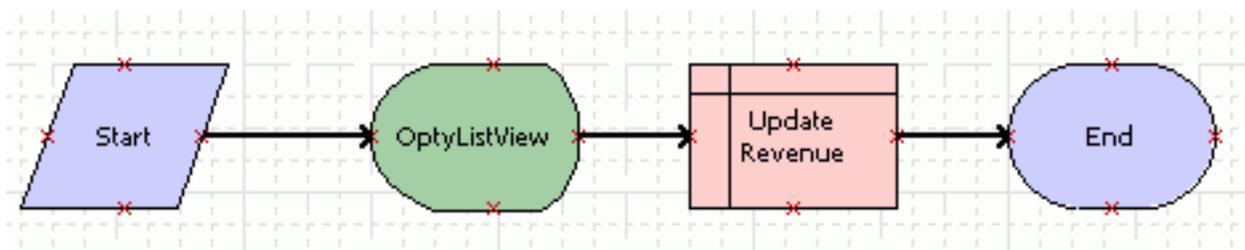
In this example, the OptyListView task view step lists records in the Opportunity business component. The Siebel operation step named Update Revenue does the following:

- 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. This is the default behavior.

- If the Is User Search Spec property is TRUE, then the user can query all records. The Search Specification property determines the default records.

The following table 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.

Property	Value
Name	**Opportunity**
Type	**Literal**

ORACLE

Property	Value
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. You can define the following task events:

- Pause
- Resume
- PreCancel
- PostCancel
- PostComplete
- Delete

For more information, see *About Event Handling*, and *Siebel Object Types Reference* .

To create a task event

1. Display the Task object hierarchy if necessary.

   For more information, see *Displaying Object Types Used to Develop a Task*.
2. In the Object Explorer, locate the task that you want to modify.

   You cannot configure a subtask to reference an event. If the Is Subtask property of the task 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, select 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*.
   c. (Optional) Define input arguments and output arguments for the event.

      For more information, see *Creating an Input Argument for a Task Event* and *Creating an Output Argument for a Task Event*.

ORACLE

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 want to 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, select the name of the Workflow Process in the Workflow Process property.

     If the Workflow Process property includes a value, then the Business Service Name and Business Service Method properties are ignored and the Workflow Process handles the event.

     If you select 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, 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*.

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 that you want to modify.
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 want to 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, select Input.

   Do not modify the Name property. Name is a system defined property.

ORACLE

7. In the Argument property, select the argument name.

   This picklist shows the input arguments that are available for the business service method that appear in the Business Service Method property of the parent task event. You select an argument and then Siebel enters the data in the Name property.

   > **Note:** In the Argument property you must enter a value. Either pick a value from the Pick List or type one in yourself if one is not available. The Pick List is not bounded.

8. In the Type property, select a source type. The following options are available:

   - Business Component
   - Expression
   - Literal
   - Task Property

   The type you select identifies the source for the value of the input argument.

9. Define the remaining properties, according to the Type selected in the previous step, with the values shown in 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 runtime 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.
Task Property	Choose a task property in the Property Name property.

Note that some properties will be disabled if they do not apply for the selected Type.

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 that you want to modify.
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 want to modify.

ORACLE

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, select Output.

   Do not modify the Name property. The Name property is a system defined property.

7. In the Type property, select a source type. The following options are available:

   - 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 selected in the previous step:

   - Business Component
   - Expression
   - Literal
   - Output Argument

   For more information, see *How the Type Field Affects Other Fields in the MVPW*.

   Note that some properties will be disabled if they do not apply for the selected Type.

9. In the Property Name property, select the name of a task property.

   For example, to enter the value of the output argument in the Object Id task property, select Object Id.

Using a Business Service Step to Call a Workflow Process

You can use a business service step to call a Workflow Process, as shown in the following procedure.

To use a business service step to call a Workflow Process

1. Add a business service step to a task with the values shown in the following table.

Property	Value
Business Service	Workflow Process Manager
Method	RunProcess

2. In the Multi Value Property Window (MVPW) pane, add an input argument to the business service step (you added in the previous step) with the values shown in the following table.

ORACLE

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 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 Arguments for a Task Step* and *Siebel Business Process Framework: Workflow Guide* .

Other Options for Customizing a Task

This topic describes optional UI objects that you can create for a task. It includes the following information:

- *Specifying the Operations That Users Can Perform in an Applet*
- *Creating a Task Chapter*
- *Creating a Radio Button Group*
- *Creating an Applet Message*
- *Reusing a Standard Applet*
- *Hiding the Task Pane*
- *Enabling the Task Progress Indicator*

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*.

To specify the operations that users can perform in an applet

1. If necessary, display the Applet User Prop object.

ORACLE

For more information, see *Displaying Object Types Used to Develop a Task*.

2. In the Object Explorer, click Applet.
3. In the Applets list, query the Name property for the applet you want to modify.
4. In the Object Explorer, expand the Applet tree and then click Applet User Prop.
5. In the Applet User Properties list, create a new record with the values shown 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*.

To create a task chapter

1. Create a task chapter:

   a. Open the task you want to modify in the Task Editor (see *Opening the Task Editor*).
   b. In the Multi Value Property Window (MVPW) pane, click the Task Chapters tab and then create a new record.
   c. Define a value in the Name property.
   d. Define a value for the Display Name - String Reference property.
   e. (In Siebel Tools only) Select a color from the pop-up Color applet, and then click OK.

      Siebel CRM displays this value as the task chapter title in the Siebel client. To display a custom value, do the following:

      - Define a value in the Display Name - String Override property.
      - Leave the Display Name - String Reference property empty.

   f. Select a Sequence.

      Records in the Task Chapters tab are sorted according to the sequence number. If you define a new chapter, the Sequence property will be set to the next available number in the series in increments of 10, 20, 30, and so on. If you want to insert more chapters, you can renumber chapters to be between these values. For example, you could create a new chapter and number it 25 to place it between chapters 20 and 30.

      If you define a chapter but you do not assign a task step to the chapter, then an error occurs when you validate the task.

2. Assign task steps to a task chapter:

    a. Select a step in the Task Editor and then click Assign Chapter.

       To simultaneously assign multiple steps to a chapter, depress the shift key as you click each step that you want to include in the chapter.

    b. Select a chapter and then click OK.

    c. Repeat the previous two substeps until you have assigned each step in the task to a chapter.

       Although using chapters is not required, if you assign one step in a task to a chapter, then you must assign every step in the task to a chapter.

Deleting a Task Chapter

You can delete a task chapter.

To delete a task chapter

1. In the Multi Value Property Window (MVPW) pane, click the Task Chapters tab.
2. Select the chapter you want to delete, and then choose Delete Record.

The chapter is removed from the applicable steps in the Chapters view. The steps remain but they are not assigned to a chapter.

Creating a Radio Button Group

The user can use the radio button group to make a choice in a task. 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*.

To create a radio button group

1. 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 the first step.

   For more information, see *Defining the Default Value for a Radio Button Group*.

3. In the Object Explorer, click Pick List.
4. Use the Pick List Wizard to create the picklist.

   Follow the prompts in the New Pick List Wizard, define the picklist and then select or define the LOV.

5. Make sure the Business Component property of the new picklist is set to PickList Generic.
6. Configure the business component field you defined in the first step 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, select the new picklist.

7. In the Object Explorer, click Applet.

8. In the Applets list, query the Name property for the applet where you want to 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.

9. In the Object Explorer, expand the Applet tree and then click Control.

10. In the Controls list, create a new control with the values shown in the following table.

Property	Description
Name	Type in text that describes the control.
Field	Select the field that you defined for the radio button group.
HTMLType	Select 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*.

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 Used to Develop a Task*.

2. Define a Symbolic String that includes the text for the applet message:

   a. In the Object Explorer, click Symbolic String.

   b. In the Symbolic Strings list, create a new symbolic string.

ORACLE

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 runtime 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 want to 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, select the symbolic string you defined in a previous step.
   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 replace the %1 substitution you defined in a previous step.

   b. In the Field property, select 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*.

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 want to 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 *Creating an Applet Message* with the values shown in the following table.

Property	Description
Field	Enter the name of the applet message.

ORACLE

Property	Description
Field Type	Choose Message.

5. Edit the Applet Web Template:

   ○ Drag and drop a Label control from the Palette to the applet grid layout.

   ○ (Optional) To modify the width of the control (for text), do the following:

      i. Position the mouse over the border (or edge) of the control so that the Web Template Editor changes the cursor to a line that includes arrows at each end of the line.

      ii. Drag the border to modify the size of the control.

      The text in the control wraps in edit and preview modes. Siebel CRM wraps the applet message text in the Siebel client at runtime. This message includes the dynamic text that replaces the placeholder text.

6. In the Properties pane, 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.

7. (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, type in the text for the message.

Reusing a Standard Applet

If the applet you use in a task does not interact with transient data, then you can copy and modify a standard, predefined applet instead of creating a new one. For more information, see *Overview of Transient Data*.

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. From the Help menu, select 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 the Object Explorer, query the Name property of the Applets list for the applet you identified previously.

   f. Select the Applet Web Template you want to edit, then choose Edit Web Layout.

   g. Verify that the layout resembles the applet you identified previously.

   h. Close the Web Layout Editor.

ORACLE

2. Copy a standard applet:
   a. Select the applet you want to duplicate, then choose Copy Record.
      Wait for the applet to be copied. This may take a few moments if there are many child controls or other objects in the applet. When the copy operation is complete, the cursor will be placed in the Name field, as a unique name is required for the new applet.
   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.
   c. Select the Applet Web Template you want to edit, then choose Edit Web Layout.
   d. In the Applet Web Template Editor, delete unnecessary controls.
      The Task UI framework 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
   e. In the Applet Web Template Editor, reposition controls and their labels until the applet resembles the required layout for the task.

Hiding the Task Pane

Siebel CRM automatically displays the Task Pane by default. Siebel CRM does the following if you configured to hide the Task Pane:

- Hides the Task Pane for any task, including any completed, paused, or cancelled task instance. The Task Pane remains hidden even if the user logs out and then 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.

If you configure a Siebel application to hide the Task Pane, then no users can use the Task Pane to start a task in the application. Other ways exist to start a task. For more information, see *Starting a Task*.

For more information, see *About the Task Pane*.

To hide the Task Pane

1. Display the Application User Prop object type.
   For more information, see *Displaying Object Types Used to Develop a Task*.
2. In the Object Explorer, click Application.
3. In the Applications list, locate the Siebel application you want to modify.
4. In the Object Explorer, expand the Application tree, and then click Application User Prop.
5. In the Application User Props list, create a new record with the values shown in the following table.

Property	Description
Name	HideTaskPane

ORACLE

Property	Description
Value	True

6. 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*.

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 the task does not display the Task Progress Indicator.

3. Log in to the Siebel client.
4. (Optional) Configure the task to display the task percentage:

   a. Navigate to the Administration - Business Process screen, then the Task Monitoring Configuration view.
   b. Query the Name column for the task you want to modify.
   c. Click Compute Percentage.
5. Navigate to a view that includes the task that must display the Task Progress Indicator.
6. Verify that the task displays the Task Progress Indicator.

Note: Task Progress Indicator does not work when the task is in inspect/preview mode.

ORACLE

12 Guidelines and Techniques for Task Development

Guidelines and Techniques for Task Development

This chapter describes guidelines and techniques for developing a task. It includes the following topics:

- *Guidelines for Developing a Task*
- *Techniques to Improve Task Usability*

Guidelines for Developing a Task

This topic describes guidelines for developing a task. It includes the following information:

- *Guidelines for Organizing the Task Flow*
- *Guidelines for Designing Task Functionality*
- *Guidelines for Designing User Interface Elements*
- *Guidelines for a Multilingual Task*
- *Guidelines for Using a Business Service*

Guidelines for Organizing the Task Flow

Use the following guidelines when organizing the task flow:

- **Envision a simple task.** Envision the display of information as a series of linked pages or steps when you design a task. 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.

- **Simplify task complexity.** Make sure the task 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.** Separate the task into a number of logical chunks. A task 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 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. 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 around logical commit points of the task transaction.** For more information, see *About Task Transaction*.

ORACLE

Guidelines for Designing Task Functionality

Use the following guidelines when designing the functionality of the task:

- **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.** How frequently the user performs the task can affect how the task 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. Design your task 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.**

- **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. Depending on the complexity and length of a task, 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.** Remain in the task. 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. 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 UI framework 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 or cancels the task, then you avoid having Siebel CRM create an orphan file.

ORACLE

Guidelines for Using a Script to Start a Task

If you use a script to start a task, then use the following guidelines:

- Make sure an active UI context exists before the user starts a task.

- Do not start a task from the middle of a run-time event, such as the WriteRecord event.

- Do not create a task 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 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. A script must only call a task when the business component includes at least one record.

Only limited support exists for starting a task from a script.

Guidelines for Designing User Interface Elements

This topic provides guidelines that you can use to design user interface elements. It includes the following information:

- *Guidelines for Designing a Task View*
- *Guidelines for Using a Form Applet and a List Applet*
- *Guidelines for Reusing Views and Applets*
- *Guidelines for Designing Buttons and Menus*
- *Guidelines for Using the Default Focus*
- *Guidelines for Designing the Structure and Content of a Page*
- *Guidelines for User Interface Styles to Avoid*

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.** Configure the task to display only data that is specific to the current step. This technique helps to simplify the UI.

- **Focus feedback that the task 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 the task 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*.

- **Use a radio button or picklist.** You can configure the task 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.

ORACLE

- 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 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*.

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 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, particularly if the user only creates a single record.

Guidelines for Reusing Views and Applets

To enforce the simpler layout requirements of a task, 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 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 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.

ORACLE

- **Make sure you deploy the task (by delivering the Workspace) before you add a button or menu to start the task.** You must deliver the task to the Integration Workspace (or MAIN) before the user can use a button or menu item to start it. If you do not do so, then Siebel CRM displays a runtime error in the Siebel client.

- **Do not create menus and buttons that perform the same function.** For example, avoid including *header and footer* task playbar applets on the same view. Instead, include only the header task playbar applet or only the footer 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

Use the following guidelines to design the structure of a page that displays in a task:

- **Include a title for the task 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.

The following table describes the different types of content that Siebel CRM can display on a page and their recommended usage. Make sure that a page only includes content that supports the purpose of the page.

Content Type	Recommended Usage
Overview Page	Displayed at the start of a task. Provides an overview of the goal of the task 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. Includes only data that is relevant or meaningful to the current task 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 to make adjustments.
Summary Page	Displayed at the end of a task. 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.

ORACLE

Guidelines for User Interface Styles to Avoid

Avoid using the following:

- **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. 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

You can use values that are independent of any given language to design a task that Siebel CRM can run in multiple languages. Pay close attention to the following:

- 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*.

- 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 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* .

ORACLE

Guidelines for Using a Business Service

A business service allows you to run a predefined or custom action in a task. 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. 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 cannot call a business service that includes browser script. If a task 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 Task Usability

This topic describes techniques you can use to improve the usability of a task. It includes the following topics:

- *Split View Technique*
- *Optional View Technique*
- *Mixed View Technique*
- *Mixed Applet Technique*
- *Business Component Method Technique*
- *Refine Query Technique*
- *Commit Interim Data Technique*
- *Other Usability Techniques*

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 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.

ORACLE

Example of the Split View Technique

Assume a task includes account data that includes two hundred single value fields and eighty multivalue fields in the Account business component. This task 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 must display data that the user can review and if this data is optional, then you can make the view optional. This following procedure shows how to use the optional view technique.

To use the optional view technique

1. Open the Task Editor (see *Opening the Task Editor*) for the task you want to modify.
2. Add a Siebel operation step with the values shown in the following table:

Property	Value
Operation	Query

3. (Optional) In the Multi Value Property Window (MVPW) pane, click the Output Arguments tab and then add an output argument with the values shown in the following table.

Property	Value
Type	Output Argument
Output Argument	NumAffRows For more information, see *Arguments of a Task Step*.

For more information, see *Creating an Output Argument for a Task Step*.

4. Add a Decision step:

   a. Place this decision step immediately after the Siebel operation step you added previously.
   b. Add a branch with the values shown in the following table.

Property	Value
Type	Condition

ORACLE

Property	Value

    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 with the values shown in 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 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 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*.

Mixed Applet Technique

If a task must display data from several business components in a single applet, then you can create a task that uses a transient business component to hold data for the mixed applet. The task can save the data to a nontransient business component. The task saves this data immediately following the view that includes the mixed applet. For more information, see *Overview of Transient Data*.

ORACLE

Example of the Mixed Applet Technique

The predefined CMEREF - Activation Order task 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, 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:

- 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, the task 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, one choice at a time:

1. The user makes a choice, and then clicks Next on the task playbar applet.
2. The task calls the business logic and in the correct order.
3. If an exception occurs, then the task displays an error dialog.
4. If an exception does not occur, then the task can display the next view.

This following procedure shows 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 with the values shown in the following table.

Property	Value
Class	CSSBCAdapterSvc

ORACLE

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.
4. Call the business component method from a business service step in the task.

   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 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 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, 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 runs. You can configure a task 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 UI framework 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 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 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 allows the user to use the name to refine the query. For example:

```
[Name] LIKE '*300 minutes*'
```

This task 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 resides in the Siebel Communications application.

ORACLE

Commit Interim Data Technique

A task 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. The following procedure shows how to use the commit interim data technique.

To use the commit interim data technique

1. Open the Task Editor (see *Opening the Task Editor*) for the task you want to modify.
2. Add a view step that reviews data that the user enters in the task.
3. At the point in the task where data that the user entered must become visible to other users, set the Forward Button Type property to Submit.

   For more information, see *Task Playbar Validation with Forward Navigation*.
4. Add a commit step immediately downstream of the view step you added previously.

Example of the Commit Interim Data Technique

In the predefined Field Activity Main Task task, 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 and choose a different status before committing modifications that are visible to other users.

Other Usability Techniques

The following table describes other usability techniques that you can use.

Usability Technique	Recommended Usage
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*.
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*.
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.

ORACLE

Usability Technique	Recommended Usage
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 captures customer information that is useful in the customer dashboard, then configure Siebel CRM to push this data to the Customer Dashboard.

ORACLE

ORACLE

13   **Siebel Task UI Reference**

Siebel Task UI Reference

This chapter includes reference information for Siebel Task UI. It includes the following topics:

- *Business Component Fields That a Task Can Modify*
- *About Task Transaction*
- *About Event Handling*
- *About Event Logging*
- *About Task Metrics*

Business Component Fields That a Task Can Modify

A task cannot modify the following types of business component fields:

- Fields that use a multivalue group
- Calculated fields

A task 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 Multivalue Group

To update a field that uses a multivalue group, you can define a business component for this field and then create a link to the related child and parent business components. 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 create a link between the two business components and use it to add the child (Account Team) business component 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

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.

ORACLE

- 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 level. For more information, see *Disabling Task Transactions*.

For more information, see *Configuring Siebel CRM to Resolve Task Transaction Conflicts*.

Atomicity

Siebel CRM can save as a group the operations that it performs in a task 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
- Prevents task persistence across sessions and users.

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. 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 Finishes*.

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 UI framework uses a locking feature to handle any concurrency issues in the server component. It makes sure that no more than one transaction for each task 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 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 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 to handle a service request while on a phone call with a customer.
2. Siebel CRM passes the task instance ID of the task 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 that the call center representative is running.

How Siebel CRM Shares Temporary Storage

The sharing of temporary storage is disabled by default in Siebel CRM. To enable the sharing of temporay storage, you must define the TASKTEMPSTORAGESHARING task property with a data type of String. If this property is not defined, then Siebel CRM will display an error indicating 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, then Siebel CRM creates a new task instance. The relationship that exists between a task instance and a task is similar to the relationship that exists between a business component (such as Account) and a particular instance of that component (such as, Company X). Each task instance includes a state that consists of task properties and navigation history. Siebel CRM typically shares data that resides in

ORACLE

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*.

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 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 calls an event. The following operations support event handling:

- Pause

- Resume

- Cancel

- Delete

- Complete

If Siebel CRM starts an operation in a task, 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 pre-event 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 pre-event handlers.

For example, the Cancel operation includes two event handlers: a pre-event handler and a post-event handler:

- The PreCancel event handler runs when the user cancels a task 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 UI framework 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* 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 calls a task event when the user is in a subtask, then Siebel CRM sends the task event to the parent task and runs the corresponding event handler that the parent task references. It runs this event handler in the context of the parent task.

ORACLE

Operations That Call an Event Handler

The following table describes the operations that call an event handler.

Operation	Description
Pause	The Pause operation saves the following information in Siebel database: • Task state and task instance. • An inbox item. The user can use this inbox item to restart the task instance.
Resume	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	The Cancel operation stops the task transaction. For more information, see *Cancel Operation Events*.
Delete	The Delete operation removes a task instance from the following items: • Inbox • Siebel database If a parent task starts a task, then the subtask sends the parent task an Abort status. The task utility service creates this event.
Complete	The Complete operation ends the task instance in the following situations: • The user clicks Finish or Submit. • Siebel CRM finishes the task instance automatically. When the task finishes, Siebel CRM starts the PostComplete event.
Post Complete	The PostComplete operation allows the task to do more work after it finishes running. The following examples describe how Siebel CRM uses the PostComplete operation: • A user runs a task 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 it has been submitted, a Workflow Process must run. This requirement makes sure no unwanted data exists. For example, a user can follow multiple paths in a task and create records in these paths. The user can navigate backward in the task 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.

Cancel Operation Events

The following table describes the (PreCancel and PostCancel) events that the Cancel operation uses.

ORACLE

Event Type	Description	Usage
PreCancel	The PreCancel event occurs in the context of the task, 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 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.
PostCancel	The PostCancel event occurs after the PreCancel event finishes, outside of the context of the task 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.

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*.

The following table describes event logging for tasks and includes the following information:

- Event types that are related to the task 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 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 higher than the default parameter affects performance. You must reset the trace level to the default parameter after you finish troubleshooting.

Event Type	Level	Symbol	Information Symbol
Workflow Definition Loading	3	DfnLoad	**Step.** Steps and branch names that Siebel CRM loaded.
Workflow Definition Loading	5	DfnLoad	**Chapter.** Chapter names that Siebel CRM loaded.
Task Execution	3	TskExec	**Operation.** 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.
Task Execution	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	**Operation.** Includes information about the navigation operations that the user performed in a task instance.
Task Navigation	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.
Task Navigation	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.
Workflow Step Execution	4	StpExec	Includes information about a task step and input or output arguments of the task step. The following symbols are included: • **Condition.** 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 UI framework 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:

ORACLE

Event Type	Level	Symbol	Information Symbol
			• 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 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*.

Timestamp Metrics

A *timestamp metric* is a type of metric that stores timestamps for a task. 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, 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 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

ORACLE

14 **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.

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. Determines direction through the task according to input.

End Step

A type of task step that instructs the runtime task 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 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.

ORACLE

Error Step

A type of task step that displays a custom error message in the Siebel client. Allows you to configure a task to display a localized error message.

Long-running Workflow Process

A persistent Workflow Process that can last for hours, days or months. A task can be integrated 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 – this 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.

Subtask Step

A type of task step that calls another task. It allows you to start a separate task in a task.

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 steps, allowing bidirectional navigation, branching, and the ability to pause and resume the task.

ORACLE

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 can pause, resume, cancel, delete, or complete a task. 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 Instance

A single instance of a task. Siebel CRM creates a new task instance when the user or a long-running Workflow Process starts a task. The relationship that exists between a task instance and a task is similar to the relationship between a business component (such as Account) and a particular instance of that component (such as, Company X).

Task Metric

A type of metric that collects and stores task data that Siebel CRM regularly saves to the Siebel database. Siebel CRM can store timestamp metrics and property metrics for a task.

ORACLE

Task Object Definition

The metadata component of a task. The object definition for the task includes the task 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 one or more task sessions.

Task Step

A child object of a task. Each task includes several steps. Examples of task steps include the following: Start, Business Service and End step.

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. 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, the Task UI framework stores this transactional data in temporary storage and does not save it to the Siebel database unless it encounters a commit step or reaches the end of the task. If the user cancels the task, then it removes this transactional data from temporary storage.

ORACLE