CentraSite

Working with Design/Change-Time Policies

Version 9.5 SP1

November 2013

This document applies to CentraSite Version 9.5 SP1.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: IINM-UG-POLICIES-95SP1-20140410

Table of Contents

Working with Design/Change-Time Policies

This document describes how to create design/change-time policies and how to define approval policies.

The content is organized under the following sections:

Functional Scope	Describes how to create and manage design/change-time policies in CentraSite.
Using Approval Policies	Describes how to create approval policies and work with approval workflows.
Working with Email Notifications	Describes how to create policies use the Send Email Notification action. Also describes how to create email templates and upload the templates to CentraSite.
Working with Predefined Policies	Describes ways you can work with the predefined policies that perform system operations.

1 Functional Scope

This chapter covers the following topics:

Note: For descriptions of the built-in actions that you can include in a design/change-time policy, see the document *Built-In Design/Change-Time Actions Reference*.

Introduction

Design/change-time policies specify a set of *actions* that are to be executed when a specified *event* occurs to an instance of a specified *object type*. You use design/change-time policies to customize the behavior of CentraSite when certain events occur on assets and other objects in the registry (for example, during creation, modification and/or deletion events). You can use design/change-time policies to perform tasks such as obtaining approvals, executing automated tests, issuing notifications and imposing organizational standards when assets or other objects are created, modified or deleted.

Note: Design/change-time policies are only available in the *CentraSite ActiveSOA* edition.

Objects on Which Design/Change-Time Policies Can Operate

Design/change-time policies operate on the objects that CentraSite manages. The types of objects to which you can apply design/change-time policies are:

- Organizations
- Users (Note that policies you apply to User objects are enforced when events occur to the User objects in the CentraSite registry, not when events occur in the external naming directory.)
- Taxonomies
- Policies
- Assets (You can create policies that apply generally to all policy-enabled assets or to specific policy-enabled types. For more information about policy-enabled assets, see ***Applying Policies to Assets***, below.)
- Report Templates
- Lifecycle Models

Applying Policies to Assets

The type definition for an asset includes a property setting called **Policies Can Be Applied**. This property determines whether assets of the given type are *policy-enabled* (i.e., whether design/change-time policies are to be executed against them). When this property is enabled, you can create policies that are specific to the assets of that type. Additionally, instances of that type are capable of triggering design/change-time policies that target "assets" in general.

By default, the **Policies Can Be Applied** property is enabled for an asset type. If you do not want instances of particular asset type to be affected by design/change-time policies, disable this property in the asset's type definition.

Applying Policies to Virtual Types

When an asset is an instance of a virtual type, the set of policies that CentraSite applies to the asset depends on the virtual type's **Inherit Base Type Policies** setting. If the type's **Inherit Base Type Policies** option is enabled, CentraSite applies the policies of the base type to the asset *in addition to* the policies of the virtual type. For example, the **Inherit Base Type Policies** option is, by default, enabled for virtual services. Therefore, when CentraSite enforces policies for a virtual service, it applies the set of policies that are defined for the Virtual Service object type *and* the set of policies that are defined for the Service object type (the base type for the Virtual Service type).

If you disable the **Inherit Base Type Policies** option for a virtual type, CentraSite applies to the asset only the policies that are defined for the virtual type. For example, if you disable the **Inherit Base Type Policies** option for the Virtual Service object type, CentraSite applies to virtual services, only the policies that are defined for the Virtual Service type. Policies that are defined for the Service type are not applied.

The following table summarizes how the set of policies that CentraSite enforces for a virtual type is affected by the state of the **Inherit Base Type Policies** option.

If the virtual type's "Inherit Base Type Policies" option is...	And the policy is defined for the...	The instances of the virtual type will have...
ENABLED	Base Type	Base Type Policies
ENABLED	Virtual Type	Virtual Type Policies
ENABLED	Base Type —AND— Virtual Type	Base Type Policies —AND— Virtual Type Policies
DISABLED	Base Type	None
DISABLED	Virtual Type	Virtual Type Policies
DISABLED	Base Type —AND—	Virtual Type Policies

If the virtual type's "Inherit Base Type Policies" option is...	And the policy is defined for the...	The instances of the virtual type will have...
	Virtual Type	

For more information about virtual types and the **Inherit Base Type Policies** option, see the section *What is a Virtual Type?* in the document *Object Type Management*. For information about which predefined types in CentraSite are virtual types, see the section *The Predefined Asset Types Installed with CentraSite* in the same document.

Note: The **Inherit Base Type Policies** option does not affect policies that are assigned to the generic "Asset" type (i.e., policies that apply to all assets). Policies that are associated with the "Asset" type are applied to both base types and virtual types, irrespective of the **Inherit Base Type Policies** setting.

Events during Which Design/Change-Time Policies Can Be Enforced

You can apply design/change-time policies when the following events occur to an object in CentraSite.

Event	Occurs...
PreCreate	Immediately before CentraSite commits a new object to the registry.
PostCreate	Immediately after CentraSite commits a new object to the registry.
PreUpdate	Immediately before CentraSite commits an update to an existing object in the registry.
PostUpdate	Immediately after CentraSite commits an update to an existing object in the registry.
PreDelete	Immediately before CentraSite removes an object from the registry.
PostDelete	Immediately after CentraSite removes an object from the registry.
PreStateChange	Immediately before a specified lifecycle state change is made to an object.
PostStateChange	Immediately after a specified lifecycle state change is made to an object.
OnConsumerRegistration	When an asset owner accepts a pending registration request by clicking the **Apply Registration Policies** button in the **Pending Registrations** inbox. **Note:** The OnConsumerRegistration event *does not* occur when a user registers a consumer directly from an asset's **Consumer** profile. For more information about registering consumers, see the section *Consumer Provisioning and Consumer-Provider Relationship Tracking* in the document *Working with Consumer Applications*.
OnTrigger	When you use the **Run Policy Now** button in CentraSite Control to run a policy on demand. For more information about this type of event, see ***Running Policies On Demand***.

Event	Occurs...
OnCollect	When a handler process calls the collector. *This event is intended to be used only by predefined policies that perform a collection process.* For more information about collectors, see ***Collector and Handler Policies***.
OnExport	When the export handler is called during an export operation. *This event is intended to be used only by predefined policies that perform an export operation.* For more information about export handlers, see ***Collector and Handler Policies***.
OnMove	When the move handler is called during a move operation (the movement of an asset to another user or organization. *This event is intended to be used only by predefined policies that perform a move operation.* For more information about move handlers, see ***Collector and Handler Policies***.

Supported Object and Event Combinations

Not all object types support the full set of events. Some events occur only with certain types of objects. For example, a PreStateChange event occurs only on Assets, Policies and Lifecycle Models. If you create a policy for a PreStateChange event on a User object, that policy will never execute, because a PreStateChange event will never occur on a User object.

The following table identifies the events that each object type supports.

	Organization	Taxonomy	User	Policy	Lifecycle Model	Assets	Report Template
PreCreate	✓	✓	✓	✓	✓	✓	✓
PostCreate	✓	✓	✓	✓	✓	✓	✓
PreUpdate	✓	✓	✓	✓	✓	✓	✓
PostUpdate	✓	✓	✓	✓	✓	✓	✓
PreDelete	✓	✓	✓	✓	✓	✓	✓
PostDelete	✓	✓	✓	✓	✓	✓	✓
PreStateChange				✓	✓	✓	
PostStateChange				✓	✓	✓	
OnConsumerRegistration						✓	
OnTrigger	✓	✓	✓	✓	✓	✓	✓
OnCollect	✓	✓	✓	✓	✓	✓	

	Organization	Taxonomy	User	Policy	Lifecycle Model	Assets	Report Template
OnExport	✓	✓		✓	✓	✓	
OnMove			✓			✓	

Actions that Design/Change-Time Policies Can Execute

An action in a design/change-time policy instructs CentraSite to perform a specific task, such as submit a request for approval, send an email notification to a group of users, perform a specified test or validate certain properties of an object.

Built-In Actions

CentraSite includes many built-in actions that you can use to compose design/change-time policies. Built-in actions are provided in the following categories:

Category	Descriptions
WS-I Compliance	Actions that test the conformance of a Web service to the basic profiles specified by the Web Services Interoperability organization (WS-I).
Design Time	Actions that you can apply when an object is initially added to CentraSite. In general, the actions in this category are designed to be used during a PreCreate event.
Change Time	Actions that you can apply when an object within CentraSite is modified or deleted. This category contains actions that you use to obtain approvals, classify objects and perform various types of validation checks.
Global Templates	Actions that perform general tasks such as sending email notifications or setting permissions. The actions in this category can be used with nearly all event types.
ARIS	An action that notifies ARIS when changes occur to the business processes and services that have been published by ARIS to CentraSite.

For information about the built-in actions that CentraSite provides for design/change-time policies, see the document *Built-In Design/Change-Time Actions Reference*.

Custom Actions

If you need to execute a task that is not provided by a built-in action, you can create a custom action to perform the work. A custom action consists of a Java class or a Groovy script that performs the required task (for example, running a test, adding a required attribute or updating an external database).

To use a custom action, you must upload the Java class or Groovy script to the CentraSite repository and define an *action template* for it. An action template specifies the location of your custom code and identifies the parameters that it uses. After you create the action template, your custom action appears in the CentraSite Control user interface and it can be inserted into a policy just like a built-in action.

For information about adding custom actions to CentraSite, see the document *Developing Custom Actions*.

What Happens at Enforcement Time?

The policy-enforcement process begins when a CentraSite user submits a request that acts on one of the object types listed in ***Objects On Which Design/Change-Time Policies Can Operate***. Depending on the type of request the user submits, one of the events identified in ***Events During Which Design/Change-Time Policies Can Be Enforced*** can occur.

When an event occurs, CentraSite queries its database and executes policies that satisfy the following criteria:

- The policy's **Event Type** and **Object Type** settings match the given event type and object type.

  —AND—

- The policy's object-selection criteria are satisfied by the object on which the event occurred.

  —AND—

- The policy is scoped for the same organization as the object —OR— the policy is *system-wide*, meaning that it applies to all organizations.

Note: It does not matter what kind of client submits the request or how the request reaches CentraSite. Design/change-time policies are applied to all requests, regardless of whether they come from the CentraSite Control user interface, a UDDI client or a JAXR-based client. Be aware that the execution of a policy can itself trigger another policy. This can occur if an action in a policy performs an operation that is within the scope of another policy.

What Happens if an Event Triggers Multiple Policies?

When multiple policies have the same scope, all of those policies are triggered when an in-scope event occurs. To determine the order in which to execute these policies, CentraSite examines each policy's **Priority** setting.

The **Priority** setting contains a non-negative integer that indicates the policy's priority. A priority value of 0 represents the highest possible priority. The following priority values are reserved for system use. These priorities can only be assigned to **predefined policies**. You cannot assign these priority values to the regular, user-defined policies that you create using CentraSite Control.

- Values 0 through 10
- Values greater than 9999

Note: A policy's **Priority** property is used *only* when CentraSite is given multiple policies to enforce for the same event. If CentraSite has only one policy to enforce, the **Priority** property is ignored entirely.

When an event triggers multiple policies, CentraSite executes the policies serially, in priority order from lowest to highest (that is, it executes the policy with the *lowest* priority value first). Each policy in the series is executed to completion before the next one begins.

If two or more policies have the same priority value, their order is indeterminate. CentraSite will execute these policies in serial fashion after all lower priority policies and before any higher priority policies. However, you cannot predict their order

Example

If CentraSite were given the following policies to enforce for an event:

Policy	Priority
Policy A	11
Policy B	25
Policy C	11
Policy D	100
Policy E (system policy)	0

It would execute the policies in the following order:

Policy	Priority
Policy E (system policy)	0
Policy A then Policy C (or vice versa) The order of these two policies cannot be controlled or predicted because they have the same priority.	11
Policy B	25
Policy D	100

Note: The preceding example is shown for illustrative purposes. It is not a good practice to have an event trigger policies that have the same priority as exhibited by policies A and C above.

What Happens When an Action Fails?

When an action in a design/change-time policy completes its execution, it returns a completion code and a completion message. This information is written to the policy log if CentraSite is configured to log successful policies.

If the completion code indicates success, CentraSite performs the next action in the policy (if one exists) or completes the requested work on the object (for example, it commits the given change to the database).

If the completion code indicates failure, CentraSite records the error in the policy log. Then it immediately exits the policy. If the policy contains additional actions, those actions are not executed. If the policy was triggered by one of the pre-commit events (for example, during a PreCreate, PreUpdate or PreStateChange event) the requested operation is not performed. If the initial request had triggered multiple policies, any policy that had not yet been executed will be bypassed.

For information about viewing failed policies in the policy log, see ***Viewing the Policy Log*** and ***Viewing Failed Policies From Your Inbox***.

Predefined Policies Used by CentraSite

CentraSite includes a set of *predefined policies* that execute internal operations on the registry. Many of these policies relate to operations such as deleting objects, exporting objects and moving objects. By default, predefined policies are not shown in the policy list, however, you can view them by enabling the **Show Predefined Policies** option.

System policies often execute on events such as OnCollect, OnMove and OnExport. For example, the OnMove event triggers the Default Move Handler policy, which makes the changes necessary to move an asset from one organization and/or user to another. If your site has special requirements, it can override certain predefined policies. For example, if you have an asset type that is not suitably exported by CentraSite's Default Export Handler policy, you can develop your own export handler policy to export assets of that type. For more information about replacing the predefined policies installed with CentraSite, see ***Working with Predefined Policies***.

⚠ **Important:** If you belong to a role that includes the "Manage System-Wide Design/Change-Time Policies" permission, you have the ability to edit, delete and deactivate CentraSite's predefined policies. However, you should not do this. These policies perform critical functions within the registry and must not be edited, deleted, or deactivated except under the direction of a technical representative from Software AG.

Who Can Create and Manage Policies?

To create and manage (i.e., view, edit and delete) policies for a specific organization, you must belong to a role that includes the "Manage Design/Change-Time Policies" permission for that organization. By default, users in the CentraSite Administrator, Organization Administrator or Policy Administrator role have this permission.

To create and manage system-wide policies (i.e., policies that apply to all organization within an instance of CentraSite), you must belong to a role that includes the "Manage System-Wide Design/Change-Time Policies" permission. By default, users in the CentraSite Administrator and Operations Administrator role have this permission.

For more information about permissions, see the section *About Roles and Permissions* in the document *Users, Groups, Roles and Permissions*.

Adding a Design/Change-Time Policy to CentraSite

To create a design/change-time policy you must perform the following high-level steps:

1. **Create a new design/change-time policy.**
   During this step, you specify the scope of the policy and select the actions that you want it to execute. For procedures, see ***Creating a New Design/Change-Time Policy***.
2. **Optionally refine the scope.**
   During this step you can specify additional criteria to narrow the set of objects to which the policy applies. For procedures, see ***Refining the Object Scope***.
3. **Configure the policy actions.**
   During this step, you assign values to the input parameters for the individual actions. For procedures, see ***Configuring Policy Action Parameters***.

4. **Activate the policy.**
   During this step, you put the new policy into effect. For procedures, see *Activating a Design/Change-Time Policy*.

Creating a New Design/Change-Time Policy

Perform these steps to create the design/change-time policy and save it to CentraSite.

▶ To create a design/change-time policy

1 In CentraSite Control, go to **Policies > Design/Change Time**.

2 Click **Add Policy**.

3 In the **Policy Information** panel, specify the following fields:

In this field...	Do the following...
Name	Enter a name for the new policy. A policy name can contain any character (including spaces). A policy name does not need to be unique within the registry. However, to reduce ambiguity, you should avoid giving multiple policies the same name. As a best practice, we recommend that you adopt appropriate naming conventions to ensure that policies are distinctly named within your organization.
Description	*Optional*. Enter a description for the new policy. This description appears when a user displays a list of policies in the user interface.

Version	*Optional*. Specify a version identifier for the new policy.
	Note: The version identifier does not need to be numeric.
	Examples:
	`0.0a`
	`1.0.0 (beta)`
	`Pre-release 001`
	`V1-2007.04.30`
	The version identifier you enter here is the policy's public, user-assigned version identifier. CentraSite also maintains an internal, system-assigned version number for the policy. For more information about user-assigned and system-assigned version identifiers, see, *System-Assigned vs User-Assigned Version Identifiers*.

Priority	Enter an integer that represents the priority of this policy with respect to other policies that might be triggered by the same event. The priority value determines the order in which the policies are enforced. The lower the **Priority** value, the higher the priority (i.e., 0 is the highest priority, and policies with this priority value are executed first).

In this field...	Do the following...
	- Priority values 0 through 10 and values greater than 9999 are reserved for predefined policies. You cannot assign these values to the user-defined policies that you create in CentraSite Control. - The default priority for a user-defined policy is 11. For more information about priorities, see *What Happens if an Event Triggers Multiple Policies?*

4 In the **Scope** panel, specify the object and event types to which the policy applies.

In this field...	Do the following...
Object Types	Specify the types of objects to which this policy applies. **Note:** If the object that you select is a base type that has virtual types associated with it, and the **Inherit Base Type Policies** option is enabled for certain of its virtual types, be aware that the policy you create will be applied to instances of the base type *and* instances of those virtual types. For more information about how CentraSite applies policies to base types and virtual types, see *Applying Policies to Virtual Types*. For information about which predefined types in CentraSite have virtual types associated with them, see the section *The Predefined Asset Types Installed with CentraSite* in the document *Object Type Management*.
Event Types	Specify the types of events to which this policy applies. **Note:** Not all event types occur for all objects. See *Supported Object and Event Combinations* for valid combinations. **Important:** The OnCollect, OnMove and OnExport events are designed to execute handler and collector processes. *Do not* use these events unless you are creating a handler or collector policy. The improper use of these event types can damage the registry. For information about creating collector and handler policies, see *Collector and Handler Policies*.
Organization	Specify the organization to which the policy applies or select "All" if the policy applies to all organizations. **Note:** The **Organization** list contains the names of all organizations for which you have "Manage Design/Change-Time Policies" permission. The option appears in the list if you also have "Manage System-Wide Design/Change-Time Policies" permission. **Important:** Set the system-wide property ("All") with care. You cannot change this assignment after the policy is created. For more information about the **Organization** property, see *System-Wide versus Organization-Specific Policy Enforcement*.

5 Click **Next**.

6 If you selected the PreStateChange or PostStateChange event in the previous panel and there is a lifecycle model for the object types that you have specified, CentraSite Control will ask you to select the lifecycle states that will trigger the policy. To complete this panel, do the following:

1. If you want the policy to execute immediately *before* the state is actually changed, click the **Add States** button in the **Before the Object Enters State** list and select the states that will cause this policy to execute. (Note that the **Before the Object Enters State** list will only be present if you selected the PreStateChange event in the previous panel.)
2. If you want the policy to execute immediately *after* the state is changed, click the **Add States** button in the **After the Object Entered State** list and select the states that will cause this policy to execute. (Note that the **After the Object Entered State** will only be present if you selected the PostStateChange event in the previous panel.)
3. Click **Next**.

7 From the **Available Actions** list, choose the actions that you want CentraSite to execute when it applies this policy. Keep the following points in mind when you select the actions for the policy:

- The actions shown in the **Available Actions** list are determined by the object types and event types that you specified on the **Scope** panel. If you do not see an action that you need, that action is probably not compatible with all of the object types and/or event types that you selected in the **Scope** panel.
- If necessary, you can click **Previous** to return to the **Scope** panel and change your object-type and event-type selections.
- Make sure that the actions in the **Selected Actions** list appear in the order that you want CentraSite to execute them at enforcement time. If necessary, use the controls above the list to place the actions in the correct order.
- Be aware that actions from the WS-I category cannot be combined with other types of actions. Also be aware that when you add a WS-I action to the action list, CentraSite will automatically add dependent actions to the list as necessary.

Note: For descriptions of the built-in actions that CentraSite provides for design/change-time policies, see the document *Built-In Design/Change-Time Actions Reference*.

8 Click **Finish** to save the new (as yet incomplete) policy.

9 Complete the new policy by doing the following:

1. Configure the parameters for each action on the **Actions** tab. For procedures, see ***Configuring Policy Action Parameters***.
2. *Optional.* Specify additional selection criteria to narrow the set of objects to which this policy applies. For procedures, see ***Refining the Object Scope***.

3. If the policy is to be enforced during a PreStateChange or PostStateChange event, ensure that the options on the **States** tab specify the lifecycles and state changes to which the policy applies. For procedures, see ***Configuring Policies that Execute on Lifecycle State Changes***.
4. If you want to allow other users to view, edit or delete this policy, click the **Permissions** tab and assign permissions to those users. For procedures, see ***Setting Permissions on a Policy***.
5. Activate the policy when you are ready to put it into effect. For procedures, see ***Activating a Design/Change-Time Policy***.

Activating a Design/Change-Time Policy

CentraSite does not begin enforcing a Design/Change Time policy until you *activate* it.

To activate a policy, you change the policy's lifecycle state to the “Productive” state. This state change executes CentraSite's *Automatic Policy Activation* policy.

Note: The *Automatic Policy Activation* policy is a hidden system policy. You cannot edit or delete this policy.

When you activate a policy, be aware that:

- You will not be allowed to activate the policy unless all of its parameters have been set. When you switch the policy to the Productive state, CentraSite executes the *Validate Policy Activation* policy. This policy will not allow you to switch a policy to the Productive state if the policy's parameters have not yet been set.
- Some organizations require an approval to activate a policy. If your organization has an approval action associated with the activation of a policy, CentraSite will not activate the policy until the required approvals are obtained. For more information about approval actions, see ***Using Approval Policies***.
- If an earlier version of the policy is already active, CentraSite will deactivate the old version before it activates the new one. For more information about working with versioned policies, see ***Versioning a Policy***
- When a policy becomes active, CentraSite begins enforcing it immediately. You can suspend enforcement of a policy by switching it to the Suspended state as described in ***Deactivating a Design/Change-Time Policy***.
- To activate a policy, you must have permission to change the policy to the Productive state.

To determine whether a policy is active or inactive, examine the policy's **Active** indicator on the **Policies** > **Design/Change-Time** page. The icon in the **Active** column indicates the policy's activation state as follows:

Icon	Description
🟩	Policy is active.
🟥	Policy is inactive.

The activation state of a policy is also reported next to the **State** field in the Design/Change-Time Policy Details page.

▶ To activate a policy

1. Display the Design/Change-Time Policy Details page for the policy that you want to activate. If you need procedures for this step, see *To view or edit the properties of a policy*.
2. Examine the **Actions** tabs and verify that all of the actions on this tab display the green checkmark icon in the **Parameters Set** column. If any of the actions display the red circle icon in this column, set their parameters before you continue.
3. In the **Policy Information** panel, click the **Change State** button. (If you do not see the **Change State** button, it is probably because you do not have permission to change the lifecycle state of a policy.)
4. In the **Change Lifecycle State** dialog box, select the **Productive** lifecycle state and click **OK**.
5. Examine the **State** field in the **Policy Information** panel to verify that the policy's state has been changed.

   If this state change requires approval, the **State** field will indicate that the policy is in the "pending" mode. CentraSite will automatically switch the policy to the requested state (and activate the policy) after all the necessary approvals have been obtained. For information about checking the status of objects that you have submitted for approval, see *Reviewing Requests that You Have Submitted for Approval*.

   Note: While the policy is in pending mode, it cannot be edited.

Deactivating a Design/Change-Time Policy

Deactivating a design/change-time policy causes CentraSite to suppress enforcement of the policy. You usually deactivate a policy for the following reasons:

- To suspend enforcement of a particular policy (temporarily or permanently).
- To edit a policy (for example, to modify the scope of a policy or change its action list).

To deactivate a policy, you change the policy to the Suspended state. Switching the policy to this state triggers the *Automatic Policy Deactivation* policy, which deactivates the policy. (Switching the policy to the Retired state also deactivates the policy, but you do not want to switch a policy to this state unless you intend to deactivate it permanently. After you place a policy in the Retired state, you cannot reactivate it.)

When you deactivate a policy, keep the following points in mind:

- CentraSite will not deactivate a policy if it is in the process of being executed. If you attempt to deactivate a policy while it is executing, your state change request will fail. If this occurs, wait for a period time and then try to deactivate the policy again.
- To deactivate a policy, you must have permission to change the policy to the Suspended state.

⚠ **Important:** If you belong to a role that includes the "Manage System-Wide Design/Change-Time Policies" permission, you have the ability to deactivate CentraSite's predefined policies. However, you should not do this. These policies perform critical functions within the registry and must not be deactivated except under the direction of a technical representative from Software AG.

▶ **To deactivate a policy**

1. Display the Design/Change-Time Policy Details page for the policy that you want to deactivate. If you need procedures for this step, see *To view or edit the properties of a policy*.
2. In the **Policy Information** panel, click the **Change State** button. (If you do not see the **Change State** button, it is probably because you do not have permission to change the lifecycle state of a policy.)
3. In the **Change Lifecycle State** dialog box, select the **Suspended** state (to deactivate it temporarily) or the **Retired** state (to deactivate it permanently), then click **OK**
4. Examine the **State** field in the **Policy Information** panel to verify that the policy's state has been changed.

   If this state change requires approval, the **State** field will indicate that the policy is in the “pending” mode. CentraSite will automatically switch the policy to the requested state (and deactivate the policy) after all the necessary approvals have been obtained. For information about checking the status of objects that you have submitted for approval, see *Reviewing Requests that You Have Submitted for Approval*.

Viewing the Policy List

The Design/Change-Time Policies page displays the design/change-time policies in CentraSite. Note that this list displays policies for all organizations, not just your own. It also includes system-wide policies (policies that apply to all organizations).

By default, the policy list shows only the user-defined policies in the registry. If you want to view the internal predefined policies, you must enable the **Show Predefined Policies** option. This option displays both user-defined and predefined policies. For more information about predefined policies, see *Predefined Policies Used by CentraSite*.

You can sort the list by object type or event type. To specify the sorting order, choose either **Object type** or **Event type** from the drop-down list labeled **Browse by:**.

Be aware that a policy might appear multiple times in the list. For example, if you create a policy that applies to both Assets and Report Templates, the policy will appear under the **Assets** heading and the **Report Templates** heading when you view the list by object type.

To view the policy list

- In CentraSite Control, go to **Policies > Design/Change Time** to display the policy list.
  - If you want to filter the list to see just a subset of the available policies, type a partial string in the **Search** field. CentraSite applies the filter to the **Name** column. The **Search** field is a type-ahead field, so as soon as you enter any characters, the display will be updated to show only those policies whose name contains the specified characters. The wildcard character "%" is supported.
  - If you want to see predefined policies as well as user-defined policies, enable the **Show Predefined Policies** option.

The Design/Change-Time Policies page provides the following information about each policy.

Note: Only the first six columns described below are displayed in this list by default.

You can choose to display the additional columns using the **Select Columns** button.

Column	Description	
Name	The name assigned to the policy.	
Description	Additional comments or descriptive information about the policy.	
Type	The object type(s) to which the policy applies.	
Event	The event(s) that triggers the policy.	
Organization	The organization to which the policy applies.	
	This value...	**Indicates that...**
	All	The policy is system-wide and applies to all organizations.
	`OrgName`	The policy applies to the specified organization.
	For more information about this property, see *System-Wide versus Organization-Specific Policy Enforcement*.	
Active	The policy's current enforcement state.	
	Icon	**Description**

Column	Description	
	🟩	The policy is active. CentraSite enforces this policy when events within the scope of the policy occur.
	🟥	The policy is inactive. Inactive policies exist in the registry, but they are not enforced.
Priority	The priority value assigned to the policy. For more information about when this value is used, see For more information about priorities, see *What Happens if an Event Triggers Multiple Policies?*	
Owner	The user to which the policy belongs.	
System Version	The automatically generated system-assigned version identifier for the policy. For more information about system-assigned version identifiers, see, *System-Assigned vs User-Assigned Version Identifiers*.	
Version	The user-assigned version identifier for the policy. For more information about user-assigned version identifiers, see, *System-Assigned vs User-Assigned Version Identifiers*.	
State	The policy's current lifecycle state.	

Viewing or Changing a Policy

You use the Design/Change-Time Policy Details page to examine and/or edit the properties of a policy. When editing a policy, keep the following points in mind:

- To edit a policy, you must have Modify permission on the policy. If your user account belongs to a role that has the "Manage Design/Change-Time Policies" permission for an organization, you automatically have permission to modify all of the policies in that organization. If your user account belongs to a role that has the "Manage System-Wide Design/Change-Time Policies" permission, you have permission to edit any system-wide policy.
- You cannot modify an active policy. If the policy that you want to edit is in the active state, you must deactivate it or use the versioning feature to create a new version of the policy. Creating a new version of the policy allows you to make your changes without having to deactivate the existing policy. For details, see *Versioning a Policy*.

⚠ **Important:** If you belong to a role that includes the "Manage System-Wide Design/Change-Time Policies" permission, you have the ability to edit CentraSite's predefined policies. However, you should not do this. These policies perform critical functions within the registry and must not be edited except under the direction of a technical representative from Software AG.

▶ To view or edit the properties of a policy

1. In CentraSite Control, go to **Policies > Design/Change Time** to display the policy list.
2. Locate the policy whose details you want to view or edit and choose **Details** from its context menu.
3. On the policy's details page, examine or modify the policy properties as necessary.

Field	Description
Name	The name of the policy. A policy name can contain any character (including spaces). A policy name does not need to be unique within the registry. However, to reduce ambiguity, you should avoid giving multiple policies the same name. As a best practice, we recommend that organizations adopt appropriate naming conventions to ensure the assignment of distinct policy names.
Description	*Optional*. Additional comments or descriptive information about the policy.

Version	The user-assigned version ID assigned to this policy. You may use any versioning scheme you choose for identifying different versions of a policy. The identifier does not need to be numeric.
	Examples:
	`0.0a`
	`1.0.0 (beta)`
	`Pre-release 001`
	`V1-2007.04.30`
	CentraSite also maintains a system-assigned version identifier for a policy. The system-assigned version identifier is independent from the version identifier that you specify in this field. For more information about user-assigned and system-assigned version identifiers, see ***System-Assigned vs User-Assigned Version Identifiers.***

Priority	An integer that represents the priority of this policy with respect to other policies that might be triggered by the same event. ■ Priority values 0 through 10 and values greater than 9999 are reserved for predefined policies. You cannot assign these values to the user-defined policies that you create in CentraSite Control. ■ The default priority for a user-defined policy is 11. For more information about priorities, see ***What Happens if an Event Triggers Multiple Policies?***
Actions	The settings on this tab specify the actions that CentraSite will execute when the policy is enforced. For more information about setting the properties on this tab, see ***Assigning Actions to a Design/Change-Time Policy***.

Field	Description
Scope	The settings on this tab specify the object types and event types to which the policy applies. For information about setting the properties on this tab, see *Specifying the Scope of a Design/Change-Time Policy*.
States	The settings on this tab specify the lifecycles and state changes to which this policy applies. For more information about setting the properties on this tab, see *Configuring Policies that Execute on Lifecycle State Changes*. **Note:** The **States** tab is present only if the policy's scope includes a PreStateChange or PostStateChange event.
Permissions	The settings on this tab identify the users who have instance-level permissions on the policy. For more information about the properties on this tab, see *Setting Permissions on a Policy*.

4 If you edited any of the settings on the Design/Change-Time Policy Details page, click **Save** to save the updated policy.

You can view the details page of multiple policies as follows:

▶ To view the details page of multiple policies

1 In CentraSite Control, go to **Policies > Design/Change Time** to display the policy list.

2 Locate the policies whose details you want to view, and mark their checkboxes.

3 In the **Actions** menu, choose **Details**.

The details page of each of the selected policies will now be displayed.

Specifying the Scope of a Design/Change-Time Policy

Scope refers to the set of properties that determine when a policy is enforced. For a design/change-time policy, scope is determined by the policy's **Object Types**, **Event Types** and **Organization** properties, which are described below.

Property	Description
Object Types	The list of object types to which this policy applies. A design/change-time policy can be applied to the objects listed in *Objects On Which Design/Change-Time Policies Can Operate*. **Note:** If the object that you select is a base type that has virtual types associated with it, and the **Inherit Base Type Policies** option is enabled for certain of its virtual types, be aware that the policy you create will be applied to instances of the base type *and* instances of those virtual types. For more information about how CentraSite applies policies to base types and virtual types, see *Applying Policies to Virtual Types*. For information about which predefined

Property	Description
	types in CentraSite have virtual types associated with them, see the section *The Predefined Asset Types Installed with CentraSite* in the document *Object Type Management*. You can optionally restrict a policy to specific instances of the selected object types by specifying additional object-selection criteria. For procedures, see *Refining the Object Scope*.
Event Types	The list of event types to which the policy applies. A design/change-time policy can be triggered by the events listed in *Events During Which Design/Change-Time Policies Can Be Enforced*. **Note:** Not all event types occur for all object types. See *Supported Object and Event Combinations*. **Important:** The OnCollect, OnMove and OnExport events are designed to execute handler and collector processes. *Do not* use these events unless you are creating a handler or collector policy. The improper use of these event types can damage the registry. For information about creating collector and handler policies, see *Collector and Handler Policies*.
Organization	Determines whether the policy belongs to a specific organization or is system-wide. For more information about the **Organization** property, see *System-Wide versus Organization-Specific Policy Enforcement*. **Important:** You cannot change the value of the **Organization** property if the policy is system-wide.

System-Wide versus Organization-Specific Policy Enforcement

The **Organization** property specifies the organization to which the policy applies. When the **Organization** property is set to "All," it indicates that the policy is *system-wide*. When the **Organization** property specifies a particular organization, it indicates that the policy is *organization-specific*.

Organization-Specific Policies

An organization-specific policy is enforced on objects that belong to the same organization as the organization to which the policy applies. For example, if you have a policy that executes when User objects are updated and its **Organization** property specifies organization ABC, CentraSite will only execute that policy when User objects *in organization ABC* are updated.

Points to keep in mind when working with organization-specific policies:

- You can create organization-specific policies for any organization on which you have "Manage Design/Change-Time Policies" permission. For example, if have you "Manage Design/Change-Time Policies" permission for organization ABC and XYZ, you can create organization-specific policies for either organization.
- You set the **Organization** property when you create a policy. After a policy is created, you cannot change this property.

- At enforcement time, CentraSite selects policies based on the organization to which the object belongs, *not* the organization to which the requestor belongs. For example, if a user from organization XYZ edits an asset in organization ABC, CentraSite applies organization ABC's policies (the organization to which the asset belongs), not organization XYZ's policies (the organization to which the requestor belongs).

System-Wide Policies

A system-wide policy is enforced for all organizations. For example, if you create a system-wide policy that executes when an asset is created, CentraSite will enforce the policy whenever *any* user in *any* organization adds an asset to the catalog.

To create a system-wide policy, you must belong to a role that has "Manage System-Wide Design/Change-Time Policies" permission. In a standard CentraSite configuration, only users in the CentraSite Administrator role and the Policy Administrator role have this permission.

System-wide policies are useful for managing many types of objects. For example, they are often used to assign users to certain server-wide groups or to enforce server-wide naming conventions on objects. However, organization-specific policies are often better choices for asset-related policies, because they enable an organization to tailor its policies to its own development processes and methodologies.

Modifying the Scope of a Design/Change-Time Policy

You use the **Scope** tab on the Design/Change-Time Policy Details page to specify a policy's scope.

Scope changes are limited by the set of actions currently selected on the **Actions** tab. That is, CentraSite will not allow you to save a policy if its scope includes object types and/or events that are not compatible with the current set of specified actions. In some cases, you might need to clear actions from the **Actions** tab in order to select the object types and event types you need on the **Scope** tab. For more information about policy and action compatibility, see *Policy Scope and Action Scope*.

Note: You cannot change the **Organization** property on the **Scope** tab. After a policy has been created, its **Organization** property cannot be changed.

To modify the scope of a design/change-time policy

1 Display the Design/Change-Time Policy Details page for the policy whose scope you want to modify. If you need procedures for this step, see *To view or edit the properties of a policy*.

2 If the policy is active, deactivate it. You cannot change the scope of an active policy. If you need procedures for this step, see *Deactivating a Design/Change-Time Policy*.

3 Select the **Scope** tab and specify the following:

- In the **Object Types** and **Event Types** lists, select the object types and event types to which the policy applies.
- *Optional*. In the **Apply policy to objects that meet the following criteria** section, specify additional selection criteria to narrow the set of objects to which this policy will be applied. For procedures, see *Refining the Object Scope*.

4 Click **Save** to save the modified policy.

Note: If the selected object and/or event types are not compatible with the current set of actions in the action list, CentraSite will not permit you to save the policy. You must correct the policy's action list or its scope to save the policy successfully.

5 When you are ready to put the policy into effect, activate it as described in *Activating a Design/Change-Time Policy*.

Refining the Object Scope

If you want to further restrict the set of objects to which the policy is applied, you can specify additional selection criteria in the **Apply policy to objects that meet the following criteria** section of the **Scope** tab. Using this section, you can filter objects by Name, Description and/or Classification attributes.

Filtering By Name and Description

You can filter objects based on their Name and/or Description attributes using any of the following comparison operators:

Comparison Operator	Description
Equals	Selects objects whose Name or Description value matches a given string of characters. For example, you would use this operator if you wanted to apply a policy only to Taxonomy objects with the **Description** value "Project IDs".
Not Equals	Selects objects whose Name or Description value *does not match* a given string of characters. For example, you would you use this operator if you wanted to apply a policy to all Taxonomies *except* those with the **Description** value "Project IDs".
Contains	Selects objects whose Name or Description property includes a given string of characters anywhere within the property's value. For example, you would use this operator if you wanted to apply a policy to Application Server objects that had the word "Fairfax" anywhere in their **Description** property.
Starts With	Selects objects whose Name or Description property begins with a given string. For example, you would use this operator if you wanted to apply a policy only to Web services whose name begins with the characters "UTIL-".

When specifying match strings for the comparison operators described above, keep the following points in mind:

- Match strings *are not* case-sensitive. If you define a filter for names that start with "ABC", it will select names starting "abc" and "Abc" (and other variations) as well as "ABC".
- Wildcard characters are not supported. That is, you cannot use characters such as * or % to represent "any sequence of characters". These characters, if present in the match string, are simply treated as literal characters that are to be matched.

Filtering By Classification Attribute

You can also filter objects based on the way in which they are classified. When you filter objects in this way, CentraSite applies the policy to objects that have at least one classification attribute whose value matches a specified taxonomy category. For example, you could use a classification filter to apply a policy to those Application Servers objects that are classified as "JBoss" servers.

When you filter objects by classification, CentraSite inspects all of an object's classification attributes at enforcement time. If any of those attributes contain the exact category specified by the selection criteria, the policy is executed.

Note: To satisfy the selection criteria, the attribute value in the object must match the category specified in the selection criteria *exactly*. Sub-categories of the specified category *are not* considered to be matches. For example, say you have a taxonomy category called "Project ABC", and that category has the subcategories "Project ABC Design", Project ABC Development" and "Project ABC Deployment". If you filter for category "Project ABC", CentraSite will apply the policy to objects that are classified by the specific category "Project ABC", but not objects that are classified by that category's sub-categories.

How to Refine the Object Scope

Use the following procedure to specify additional criteria for selecting the objects to which you want the policy applied.

To refine the object scope

1. Select the policy's **Scope** tab.
2. If you want to filter by Name or Description, take the following steps in the **Apply policy to objects that meet the following criteria** section of the tab.
    1. Select **Name** or **Description** in the first field.
    2. Select the comparison operator in the second field.
    3. Specify the match string in the third field.
3. If you want to filter by object classification, take the following steps in the **Apply policy to objects that meet the following criteria** section of the tab.

1. Select **Classification** in the first field.
2. Click **Browse** and select the category by which you want to filter objects.

4 If you want to specify additional criteria, click the plus button and repeat steps 2 and 3.

⚠ **Important:** If you specify multiple filters, the policy is applied only if the object matches *all the selection criteria* (i.e., the selection criteria is combined using an AND operator, not an OR).

Configuring Policies that Execute on Lifecycle State Changes

If you create a policy that executes on the PreStateChange or PostStateChange event type, you must configure the policy's **States** tab. The settings on this tab identify the specific state changes that will trigger the policy. This tab also specifies whether the policy is to be executed before or after the object is switched to a specified state.

When creating policies that execute on state changes, keep the following points in mind:

- Policies that are triggered by a state change execute when an object *switches to* a specified state (called the *target state*). The object's state prior to the change is immaterial. For example, if you have a lifecycle model with the states: *Test*, *Production* and *Offline*, and you have a policy that specifies the *Offline* target state, that policy will execute anytime the object switches to the *Offline* state. It does not matter whether the transition occurs from the *Test* state or the *Production* state.
- Policies that are triggered by a state change are executed regardless of whether the state change is initiated from the CentraSite Control UI, the API (e.g., a custom client program) or another policy.
- You cannot specify a target state on the **States** tab unless that state has already been defined in a lifecycle model. Additionally, the lifecycle model must be active. In other words, you cannot completely configure a policy that executes on a state change until you have created and activated the lifecycle model whose state(s) will trigger the policy.
- If you configure the policy to execute before the object's state is changed (i.e., on a PreStateChange event), and any action in the policy fails, the state change will not occur.

Assigning Actions to a Design/Change-Time Policy

The **Actions** tab on the Design/Change-Time Policy Details page specifies the list of actions that you want CentraSite to execute when it enforces the policy. CentraSite executes actions in the order in which they appear in the list.

The action list can include any built-in or custom actions that are compatible with the policy's scope (as currently specified on the policy's **Scope** tab).

Note: For descriptions of the built-in actions that you can include in a design/change-time policy, see the document *Built-In Design/Change-Time Actions Reference*.

Policy Scope and Action Scope

Like a policy, an action has a declared scope. The scope of an action is declared in the **Object Types** and **Event Types** properties in the action's *action template*. An action template is an object that defines a policy action that is available within CentraSite. For more information about action templates, see the document *Developing Custom Actions*.

A policy can only include actions whose scope *matches or exceeds* the policy's own scope. For example, if you had an action ABC with the following scope:

Action ABC's Scope			
Object Type(s):	Service		
Event Type(s):	PostCreate PostUpdate		

Action ABC

You could use this action in policies 1 and 2 below, because these policies include only objects and events that are encompassed by scope of the action. However, you could not use the action in policies 3 or 4, because these policies include objects and/or events that the action does not support.

Policy #1 Scope		Compatible with Action ABC?
Object Types(s):	Service	Yes
Event Type(s):	PostCreate	Yes

Policy #1

Policy #2 Scope		Compatible with Action ABC?
Object Types(s):	Service	Yes
Event Type(s):	PostCreate PostUpdate	Yes

Policy #2

Policy #3

Policy Scope		Compatible with Action ABC?
Object Types(s):	Service Report Template (*out of scope*)	No
Event Type(s):	PostCreate	Yes

Policy #4

Policy Scope		Compatible?
Object Types(s):	Service	Yes
Event Type(s):	PostCreate PostUpdate PostDelete (*out of scope*)	No

You can use the following procedure to examine the scope for a particular action.

To examine the scope of an action

1. In CentraSite Control, go to **Policies > Action Templates** to display the list of action templates that exist on your instance of CentraSite.
2. Locate the action whose scope you want to examine and choose **Details** from its context menu.
3. On the Edit Action Template page, select the action's **Scope** tab. This tab specifies the object types and event types with which the action can be used.

Note: Virtual types and base types are treated as distinct object types with respect to policy action scope. A policy action that is scoped to a base type cannot be inserted into a policy that is scoped only for the virtual type(s) associated with the base type, nor can an action that is scoped for a particular virtual type be inserted into a policy that is scoped specifically for the base type. The scope of a policy action is not affected by a virtual type's **Inherit Base Type Policies** option. (In other words, the **Inherit Base Type Policies** option does not enable you to insert actions that are scoped for a virtual type into a policy that is scoped for a base type, or vice versa. Virtual types and base types are treated simply as different types when CentraSite determines which actions are compatible with the specified scope of a policy.)

Modifying the Action List

Use the following procedure to modify the action list for a policy.

▶ **To modify the action list for a design/change-time policy**

1 Display the Design/Change-Time Policy Details page for the policy whose action list you want to edit. If you need procedures for this step, see *To view or edit the properties of a policy*.

2 If the policy is active, deactivate it. You cannot change the action list of an active policy. If you need procedures for this step, see *Deactivating a Design/Change-Time Policy*.

3 Select the **Actions** tab to display the list of actions associated with the policy.

4 To add actions to, delete actions from or modify the order of actions in the list, do the following.

   1. Click **Edit Actions List**.
   2. Use the controls in the **Edit Assigned Actions** dialog box to add actions to the list and/or delete actions from the list.

      When editing the list of actions, keep the following points in mind:

      - This dialog only displays actions that support the policy's current scope. If you need to specify actions for object or event types that are outside of the current scope, you must modify the policy's scope first (on the **Scope** tab) and then update the action list. For more information about the scope of a policy, see *Policy Scope and Action Scope* and *Specifying the Scope of a Design/Change-Time Policy*.
      - Make sure the actions in the **Assigned Actions** list appear in the order that you want CentraSite to execute them.
      - Be aware that actions from the WS-I category cannot be combined with other types of actions. Also be aware that when you add a WS-I action to the action list, CentraSite will automatically add dependent actions to the list as necessary.
   3. Click **OK** to save the modified list.

5 Use the procedure in *Configuring Policy Action Parameters* to configure the parameter settings for any new actions that you might have added to the list in the preceding steps or to make any necessary updates to the parameter values for existing actions in the list.

Note: For information about the parameter settings for the built-in actions provided by CentraSite, see the document *Built-In Design/Change-Time Actions Reference*.

6 When the action list is complete and you have configured all of the input parameters for the actions correctly, click **Save** to save the updated policy.

Configuring Policy Action Parameters

Most policy actions have input parameters that you must set to configure the action's enforcement behavior.

When you display the **Actions** tab on the Design/Change-Time Policy Details page, the icon in the **Parameters Set** column indicates whether the action has input parameters that need to be set.

This Icon...	Indicates that...
⭘	The action has required input parameters that have not yet been set.
✓	All of the action's required input parameters have been set. **Note:** This icon automatically appears for actions that have no input parameters.

▶ To configure the input parameters for a policy action

1 Display the Design/Change-Time Policy Details page for the policy whose actions you want to configure. If you need procedures for this step, see *To view or edit the properties of a policy*.

2 If the policy is active, deactivate it. You cannot modify the actions of an active policy. If you need procedures for this step, see *Deactivating a Design/Change-Time Policy*.

3 On the **Actions** tab do the following for each action in the list:

   1. Click the action whose parameters you want to examine or set.
   2. In the Edit Action Parameters page, set the parameters as necessary.

      Note: Required parameters are marked with an asterisk.

   3. Click **Save** to save the parameter settings.

4 After you configure the parameters for all of the actions in the list, click **Save** to save the updated policy.

Setting Permissions on a Design/Change-Time Policy

Be default, all users have View permissions on the design/change-time policies in the registry.

Users who belong to a role that includes the "Manage Design/Change-Time Policies" permission for an organization have Full permission on the policies that belong to the organization. Users who belong to a role that includes the "Manage System-Wide Design/Change-Time Policies" permission, have Full permission on all system-wide policies. To enable other users to modify and/or delete policies, you must modify the policy's instance-level permission settings.

You can modify the instance-level permissions for a policy by executing a design/change-time policy or by specifying the permissions manually on the **Permissions** tab in CentraSite Control. Procedures for both ways are provided later in this section.

When setting permissions on policies, keep the following points in mind:

- To set permissions on a organization-specific policy, you must belong to a role that has the "Manage Design/Change-Time Policies" for the organization to which the policy belongs or have the Full instance-level permission on the policy itself.
- To set permissions on a system-wide policy, you must belong to a role that has the "Manage System-Wide Design/Change-Time Policies" or have the Full instance-level permission on the policy itself.
- You can assign permissions to any individual user or group defined in CentraSite.
- The groups to which you can assign permissions include the following system-defined groups:

Group Name	Description
Users	All users within a specified organization.
Members	All users within a specified organization and its child organizations.
Everyone	All users of CentraSite *including guest users* (if your CentraSite permits access by guests).

- If a user is affected by multiple permission assignments, the user receive the union of all the assignments. For example, if group ABC has Modify permission on a policy and group XYZ has Full permission on the same policy, users that belong to both groups will, in effect, receive Full permission on the policy.

Using a Design/Change-Time Policy to Set Permissions

You can include the Set Permissions action in a design/change-time policy to set instance-level permissions on a policy. You can use this action to automatically assign permissions to a policy during any of the following events.

- PostCreate
- PreStateChange
- PostStateChange
- OnTrigger

For more information about using the Set Permissions action, see the description of this action in the document *Built-In Design/Change-Time Actions Reference*.

Setting Permissions Using the Permission Tab in CentraSite Control

Use the following procedure to assign instance-level permissions to a policy from the Permission tab in CentraSite Control.

▶ To assign permissions to a policy

1 Display the Design/Change-Time Policy Details page for the policy whose permissions you want to modify. If you need procedures for this step, see *To view or edit the properties of a policy*.

2 On the Policy details page, click the **Permissions** tab.

3 To add users or groups to the **Users / Groups** list, do the following:

   1. Click **Add Users / Groups**.
   2. Select the users and groups to which you want to assign permissions.

      If you want to filter the list, type a partial string in the **Search** field. CentraSite applies the filter to the **Users/Groups** column.

String	Description
`b`	Displays names that contain "b"
`bar`	Displays names that contain "bar"
`%`	Displays all users and groups

   3. Click **OK**.

4 To remove a user or group from the **Users / Groups** list, select the check box beside the group name or user ID and click **Delete**.

5 Use the **View**, **Modify** and **Full** check boxes to assign specific permissions to each user and/or group in the **Users / Groups** list as follows:

Permission	Allows the selected user or group to...
View	View the policy. **Note:** Disabling this permission will not prevent a user from accessing the policy. CentraSite implicitly grants users View permission on all design/change-time policies within an instance of CentraSite. This implicit permission that CentraSite grants to a user cannot be not revoked by disabling the **View** permission on this tab.
Modify	View and edit the policy.
Full	View, edit and delete the policy. This permission also allows the selected user or group to assign instance-level permissions to the policy.

6 Click **Save** to save the new permission settings.

Note: If you have given users Modify or Full permissions on the policy and you want them to be able to work with the policies in CentraSite Control, be sure the users belong to a role that has the "Use the Policies UI" permission.

Running Policies on Demand

If you create a policy for the OnTrigger event, you can use the **Run** button on the details page to run the policy “on demand.”

Many of the built-in actions in CentraSite support the OnTrigger event. For example, you can run the WS-I actions on demand. You can also use the OnTrigger event to execute policies that set permissions on certain types of objects or change the state of an object.

Who Can Run a Policy on Demand?

You can run a policy on demand if you have view permission on the policy.

To Which Objects Is the Policy Applied?

When you run a policy on demand, CentraSite queries the objects in the organization to which the policy applies and selects objects that satisfy the following conditions.

- The object is one that is within the policy's object scope.
- The object is one on which you have View permission.
- The object's name, description and/or classification properties satisfy the object-selection criteria on the policy's **Scope** tab (if any).

Note: If the policy's **Organization** property is set to "All " (meaning that it is a system-wide policy), then CentraSite queries *all organizations* for objects that satisfy the conditions listed above.

CentraSite executes the policy's actions on each object in the result set produced by this query (henceforth, referred to as the *target set*).

If an action in the policy performs an update or delete operation on the objects in the target set, be aware that these operations will only execute successfully if you have the appropriate Modify or Full permission on the target object. If you do not have the required permissions, the action that performs the edit or delete operation will fail and the failure will be reported in the policy log.

Keep in mind also that, as with other policies, a policy that you execute on demand might trigger other policies. This will occur anytime an action in the policy performs an operation that is within the scope of another policy.

What Happens When the Policy is Executed?

When you run a policy on demand, CentraSite executes the policy against each object in the target set. Results are written to the policy log and are also displayed in the results window in the user interface.

For example, if you have a policy that contains actions 1, 2 and 3 and the target set contains objects A, B and C, the policy will iterate over the objects in the target set as follows:

Iteration 1: Execute actions 1, 2 and 3 on object A

Iteration 2: Execute actions 1, 2 and 3 on object B

Iteration 3: Execute actions 1, 2 and 3 on object C

If an action returns a failure code during an iteration of the policy, CentraSite writes the failure message to the policy log and immediately exits that iteration of the policy. If the target set contains additional objects, CentraSite applies the policy to the next object in the target set.

There is one exception, namely if the Send Email Notification action returns a failure code; in this case, CentraSite writes the failure message to the policy log and performs the next action in the policy (if one exists).

Running a Policy on Demand

Use the following procedure to execute a policy on demand.

▶ To run a policy on demand

1. Display the Design/Change-Time Policy Details page for the policy that you want to execute. If you need procedures for this step, see *To view or edit the properties of a policy*.
2. Examine the **Scope** tab and verify that the **Object Types** property and the criteria in the **Apply policy to objects that meet the following criteria** section of the tab (if any) identify the precise set of objects to which you want the policy applied.
3. Examine the **Actions** tab and verify that the action list contains the set of actions that you want CentraSite to execute and that the parameters for all actions in the list are set properly.
4. Click **Run**.

   If the **Run** button does not appear on the Design/Change-Time Policy Details page, it is most likely because:

  - The policy has not been activated.
  - The policy's Event Type property does not include the OnTrigger event.

5. When the policy completes, examine the results window to determine whether all iterations of the policy executed successfully. (CentraSite also writes these results to the policy log, so you can view them later.) For information about viewing the policy log, see ***Viewing the Policy Log***.

Deleting a Policy

You delete a policy to remove it from CentraSite permanently.

CentraSite is installed with a system-wide policy called *Check State Validation Policy for Policy*. This policy will not allow you to delete a policy unless the policy is in the New or Retired state.

Important: If you belong to a role that includes the "Manage System-Wide Design/Change-Time Policies" permission, you have the ability to delete CentraSite's predefined policies. However, you should not do this. These policies perform critical functions within the registry and must not be deleted except under the direction of a technical representative from Software AG.

In addition to being in the New or Retired state, the following conditions must also be met in order to delete a policy:

- The policy must not be in-progress.
- The policy must be inactive.
- You must have Full permission on the policy.

To delete a policy

1. In the CentraSite Control, go to **Policies > Design/Change Time** to display the policy list.
2. Enable the checkbox next to the name of the policy that you want to delete.
3. Click **Delete**.

Note: When you delete a policy that is an intermediate version, CentraSite also deletes all previous versions of the policy.

You can delete multiple policies in a single step. The rules described above for deleting a single policy apply also when deleting multiple policies.

Important: If you have selected several policies where one or more of them are predefined policies (e.g., collector and handler policies), you can use the **Delete** button to delete the policies. However, as you are not allowed to delete predefined policies, only policies you have permission for will be deleted. The same applies to any other policies for which you do not have the required permission.

To delete multiple policies in a single operation

1. In CentraSite Control, go to **Policies > Design/Change Time** to display the policy list.
2. Mark the checkboxes of the policies that you want to delete.
3. From the **Actions** menu, choose **Delete**.

Copying a Policy

A design/change-time policy can become quite complex, especially if it includes many policy actions. Instead of creating a new policy "from scratch", it is sometimes easier to copy an existing policy that is similar to the one you need and edit the copy.

When you create a copy of a policy, be aware that:

- To create a copy of a policy, you must have permission to manage design/change-time policies for the organization specified in the scope of the original policy. For example, if the original policy is scoped for organization ABC, you will not be permitted to create a copy of that policy unless you have permission to manage design/change-time policies for organization ABC. (This is because the copied policy has the same organizational scope as the original.)
- When CentraSite creates a copy of a policy, the new copy of the policy is identical to the original one except that:
  - The new policy's system-assigned version identifier is always set at 1. (For additional information about system-assigned version numbers, see *System-Assigned vs User-Assigned Version Identifiers*.)
  - Ownership of the new policy is assigned to the user who created the copy.
- Like all new policies, the copied policy begins it lifecycle in the New state and it is marked as inactive.
- There is no expressed relationship between the copy and the original policy (i.e., CentraSite does not establish any type of association between the two policies).

In general, a copied policy no different than a policy that you create from scratch.

To copy a policy

1. In CentraSite Control, go to **Policies > Design/Change Time** to display the policy list.
2. Locate the policy that you want to copy and choose **Create Copy** from its context menu.

3 Modify the new policy as necessary and then save it.

Versioning a Policy

When you need to make changes to an existing policy, creating a new version of the policy is an efficient way to accomplish this task. Versioning a policy enables you to create a new version of a policy (which is an identical copy of the existing policy) and make your changes to the new version. When you are ready to put the updated policy into effect, you simply "activate" the new version of the policy. When you activate the new version, CentraSite automatically deactivates and retires the old version of the policy.

When you create a new version of a policy, be aware that:

- You can only create a new version from the *latest version* of a policy. For example, if a policy already has versions 1.0, 2.0 and 3.0, CentraSite will only allow you to create a new version of the policy from version 3.0. (It makes no difference whether the policy that you are versioning is active or inactive. You can version a policy in either mode.)
- To create a new version of a policy, you must have permission to manage design/change-time policies for the organization specified in the scope of the original policy. For example, if the original policy is scoped for organization ABC, you will not be permitted to create a new version of that policy unless you have permission to manage design/change-time policies for organization ABC.
- When CentraSite creates a new version of a policy, it produces a version that is identical to the previous version, except that:
  - The new policy's system-assigned version identifier is incremented by one. (For additional information about system-assigned version numbers, see ***System-Assigned vs User-Assigned Version Identifiers***.)
  - Ownership of the new policy is assigned to the user who created the new version.
- Like all new policies, the new version begins its lifecycle in the New state and is marked as inactive.
- CentraSite automatically establishes a relationship between the new version of the policy and the previous version. This relationship enables several capabilities and features in CentraSite that relate to versioned policies.

▶ To version a policy

1 In CentraSite Control, go to **Policies > Design/Change Time** to display the policy list.

2 Locate the *most recent version* of the policy for which you want to create a new version and choose **Create New Version** from its context menu.

3 Modify the new version of the policy as necessary and then save it.

4. When you are ready to put the new version into effect, activate the new policy. CentraSite will automatically deactivate and retire the previous version.

   Note: If you activate the new version of the policy while CentraSite is in the middle of executing the old version, your activation request will fail. If this occurs, wait for a period time and then try to activate the new version of the policy again.

System-Assigned vs User-Assigned Version Identifiers

CentraSite maintains two version identifiers for a policy: a *system-assigned identifier* and a *user-assigned identifier*.

- The system-assigned identifier is a version number that CentraSite maintains for its own internal use. CentraSite automatically assigns this identifier to a policy when the policy is created. You cannot delete it or modify it. A policy's system-assigned identifier is numeric and always has the format *`MajorVersion.Revision`*. A policy always begins with a system-assigned version identifier of 1.0. The *`MajorVersion`* number is incremented by one each time you create a new version of a policy (e.g., 1.0, 2.0, 3.0).

  A policy's system-assigned version number is shown in the **System Version** column on the Design/Change-Time Policies page and in the **System Version** field of the policy's detail page.
- The user-assigned identifier is an optional identifier that you can assign to a distinguish a specific version of a policy. This identifier does not need to be numeric. For example, you might use a value such as "V2.a (beta)" to identify a version.

  A policy's system-assigned version number is shown in the **Version** column on the Design/Change-Time Policies page and in the **Version** field of the policy's detail page.

Viewing the Policy Log

The policy log contains information about policy that CentraSite has executed. By default, CentraSite only logs information about policies that fail. However, you can optionally configure CentraSite to log information about policies that resulted in success, info, warning, and failure alerts. For information about configuring the logging facility for policies, see the section *Configuring Logs* in the document *Logging*.

Note: Over time, the policy log can grow quite large, especially if you are logging information about successful policies. To prevent the policy log from growing too large, you should purge it periodically. For information about purging the policy log, see the section *Purging Logs* in the document *Logging*.

The following procedure describes how to view the policy log. To view this log, you must belong to a role that includes the "View Policy Log" permission.

- If you belong to the CentraSite Administrator role, you can view all entries in the policy log.
- If you belong to the Organization Administrator role for an organization, you can view the log entries for policies that were triggered by users in your organization.
- If you do not belong to either of these roles, but you have the "View Policy Log" permission, you can view the log entries for the policies that you triggered.

Note: From your inbox on the My CentraSite page, you can view the list of policies that failed during events that you triggered. You do not need special permissions to view this log. For procedures, see *Viewing Failed Policies From Your Inbox*.

To view the policy log

1. In CentraSite Control, go to **Administration > Logs > Policy Log**.
2. Complete the following fields to specify which type of log entries you want to view:

<table>
<tr><th>In this field...</th><th colspan="2">Specify...</th></tr>
<tr><td><b>Object Name</b></td><td colspan="2"><i>Optional</i>. A pattern string that describes the names of the objects (of <b>Object Type</b>) whose log entries you want to view.
You can provide the exact name or use a pattern string consisting of a character sequence and/or the % wildcard character (which represents any string of characters). For example, if you specify the pattern string ‘A%', CentraSite displays entities whose names start with ‘A'.
Leave <b>Entity Name</b> empty to view all names.</td></tr>
<tr><td><b>Policy Type</b></td><td colspan="2">The type of policy whose log entries you want to view. To view the log entries for design/change-time policies, select <b>Design/Change Time</b> from the drop-down list (if it is not already selected).</td></tr>
<tr><td><b>Object Type</b></td><td colspan="2">The object type whose log entries you want to view.</td></tr>
<tr><td><b>Event Type</b></td><td colspan="2">The event type whose log entries you want to view.</td></tr>
<tr><td rowspan="5"><b>Policy Status</b></td><td colspan="2">The policy execution status that you want to view.
A policy's execution status is the result set of each of its action's execution result. CentraSite writes the following policy execution status to the policy log depending on the log configuration.</td></tr>
<tr><td><b>Icon</b></td><td><b>Description</b></td></tr>
<tr><td>🟩</td><td><i>Success</i>. Displays policies that have resulted in success alert.</td></tr>
<tr><td>ℹ</td><td><i>Info</i>. Displays policies that have resulted in informational alert.</td></tr>
<tr><td>△</td><td><i>Inprogress</i>. Displays policies that have resulted in inprogress alert.</td></tr>
</table>

In this field...	Specify...	
	⚠	*Warning*. Displays policies that have resulted in warning alert.
	●	*Failure*. Displays policies that have resulted in failure alert.
	Note: For details on a policy's action result see the section ***Viewing the Policy's Action Result*** , and for information on log configuration see the section *Configuring Logs* in the document *Logging*.	
Execution Date	*Optional*. The time period that you want to examine. Leave the **From** and **To** fields empty to view log entries for all dates.	

3 Click **Search** to retrieve the specified log entries.

4 To view details for a particular entry in the returned list, click the name of the policy.

> **Note:** If a policy included a WS-I action, the log entry for the policy will include a link to the results of the WS-I action.

Viewing the Policy's Action Result

A policy's execution status basically depends on each of its action's execution result.

When an action in a design/change-time policy completes its execution, it returns a completion code and a completion message. If the completion code indicates success, informational or warning, CentraSite performs the next action in the policy (if one exists) or completes the requested work on the object (for example, it commits the given change to the database) and writes the information to the policy log. The policy log displays a success, informational and/or warning alert accordingly if configured to log these alerts.

However, if the completion code indicates failure, CentraSite records the error in the policy log. Then it immediately exits the policy. If the policy contains additional actions, those actions are not executed. If the policy was triggered by one of the pre-commit events (for example, during a PreCreate, PreUpdate or PreStateChange event) the requested operation is not performed. If the initial request had triggered multiple policies, any policy that had not yet been executed will be bypassed.

The policy's execution status is a result set of each of its actions result.

The following table summarizes how a policy's execution status is affected by each of its action's execution result.

The policy's execution status will have...	Icon...	Policy Action A	Policy Action B	Policy Action C	Policy Action D	Policy Action E
SUCCESS	🟩	SUCCESS	SUCCESS	SUCCESS	SUCCESS	SUCCESS
INFO	ℹ	SUCCESS	SUCCESS	INFO	SUCCESS	SUCCESS
INPROGRESS	△	INFO	INPROGRESS	SUCCESS	SUCCESS	SUCCESS
WARNING	⚠	SUCCESS	INPROGRESS	INFO	WARNING	WARNING
FAILURE	🔴	SUCCESS	WARNING	SUCCESS	SUCCESS	FAILURE

Note: If you have used a client jar from the version 8.2.2, then CentraSite will show policy's result status as “Success” in the client side; but however on the server side the policy log will continue to store the policy's result status as “Informational” or “Warning”. This is because, a client jar used from versions of CentraSite prior to version 8.2.5 do not support the above policy status.

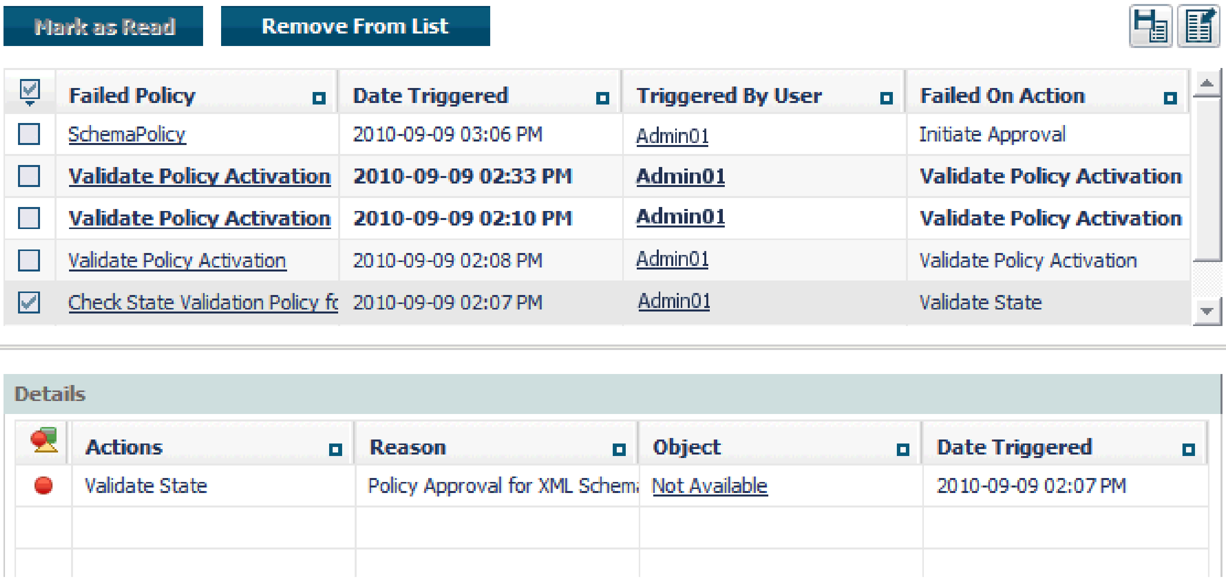
Viewing Failed Policies From Your Inbox

Your inbox on the My CentraSite page includes the **Failed Policies** link, which displays the list of policies that failed during events that you initiated.

When you click the **Failed Policies** link in your inbox, CentraSite Control opens a two-pane screen. The upper pane displays the list of logged policy failures that occurred during events that you initiated. The lower pane displays detailed information for a selected failure.

A failure stays in your **Failed Policies** list until you explicitly clear it from the list using the **Remove from List** button or the underlying log entry is purged from the policy log.

Note: When you clear an entry from this list using the **Remove from List**, you do not remove the entry from the underlying policy log. You simply eliminate it from your inbox display.



☑	Failed Policy	Date Triggered	Triggered By User	Failed On Action
☐	SchemaPolicy	2010-09-09 03:06 PM	Admin01	Initiate Approval
☐	**Validate Policy Activation**	**2010-09-09 02:33 PM**	**Admin01**	**Validate Policy Activation**
☐	**Validate Policy Activation**	**2010-09-09 02:10 PM**	**Admin01**	**Validate Policy Activation**
☐	Validate Policy Activation	2010-09-09 02:08 PM	Admin01	Validate Policy Activation
☑	Check State Validation Policy fc	2010-09-09 02:07 PM	Admin01	Validate State

	Actions	Reason	Object	Date Triggered
●	Validate State	Policy Approval for XML Schem	Not Available	2010-09-09 02:07 PM

To view the policy log

1 In CentraSite Control, go to **Home** > **My CentraSite**.

2 In the **Policy Log** section of the Inbox, click **Failed Policies**.

3 Examine the list of failures in the upper pane of the **Failed Policies** window.

4 If you want to examine the details for a reported failure, click in any non-linked area of the row that contains the failure log entry. The details for the selected failure will appear in the lower pane.

This Icon...	Indicates That...
🟩	Indicates that the action had resulted in success alert.
ℹ	Indicates that the action had resulted in informational alert.
△	Indicates that the action had resulted in inprogress alert.
⚠	Indicates that the action had resulted in warning alert.
●	Indicates that the action had resulted in failure alert.

Exporting and Importing Design/Change-Time Policies

Using the import and export features, you can export a design/change-time policy from one instance of CentraSite and import it into another.

The following sections provide specific information about exporting and importing policies. Before you use the import and export features with policy objects, review the general information provided in the document *Importing/Exporting Registry Objects*.

- Exporting Design/Change-Time Policies
- Importing Design/Change-Time Policies

Exporting Design/Change-Time Policies

When exporting a design/change-time policy, keep the following points in mind:

- You can export a policy that is active or inactive. You do not need to deactivate a policy to export it.
- The export process does not export the following objects that a policy references.
  - Lifecycle State
  - Organization

  When an administrator imports the policy on the target instance, the import process assigns these properties as described in ***Importing Design/Change-Time Policies***.
- The export process does not export the policy's instance-level permissions. When an administrator imports the policy on the target instance, the import process assigns instance-level permissions as described in ***Importing Design/Change-Time Policies***.
- If the policy includes a custom action, CentraSite will export the action template for the custom action along with the action's associated Java archive file (if the action has been implemented as a Java program) or Groovy script (if the action has been implemented as a Groovy script).
- If the policy's object scope includes a custom asset type, CentraSite will export that type along with the policy. (Note that many of the predefined types that are installed with CentraSite are treated as "custom" asset types and will be exported if they are within the policy's scope. For a list of the predefined types that are treated as custom types, see the section *The Predefined Asset Types Installed with CentraSite* in the document *Object Type Management*.)
- If the policy scope is "Assets", then the type information will not be exported. So any basic attributes referenced in the policies have to be edited in the target registry (for example, if the policy has assertions such as "Set Attribute value" or "Validate Attribute value"). There is one exception, namely if the policy uses classification attributes in the policy assertions; in this case, the type information is included in the export.

- The export process will export the parameter values assigned to the actions in the policy. If the parameter value is a reference to an instance of one of the following object types, you must export the referenced object and import it on the target instance of CentraSite before you import the policy.
  - Organization
  - User
  - Group

  If referenced User/Group objects do not already exist on the target instance of CentraSite when you import the policy, the import will remove the references to these objects and the import will be successful. After the import, you can edit the policy to reference the necessary user/group objects. If a parameter references any other type of object, the export process will export the referenced object with the policy, and that object will be imported as necessary into the target registry.
- If the policy executes on a PreStateChange or PostStateChange event, the lifecycle model and state information on the policy's **States** tab will be exported. If the specified lifecycle model and states do not exist in the target instance of CentraSite when the policy is imported, the import process will fail (i.e., the policy will not be imported).

To export a policy

1. In CentraSite Control go to **Policies > Design/Change Time** to list the available policies.
2. Locate the policy that you want to export and choose **Export** from its context menu.
3. Specify the options in the **Export** dialog box and click **OK**. If you need information about these options, see the section *Exporting and Importing Objects Using CentraSite Control* in the document *Importing/Exporting Registry Objects*.
4. Save the archive file when prompted to do so.
5. Examine the export log that is displayed by CentraSite Control and check for any errors that occurred during the export process.

Importing Design/Change-Time Policies

When importing a design-time policy, keep the following points in mind.

- If the policy, or any related object in the archive already exists in the target instance of CentraSite, the existing object will be overwritten according to the conditions described in the topic *What Happens When an Imported Object Already Exists in the Target Registry?* in the section *Introduction* of the document *Importing/Exporting Registry Objects*.
- To import a policy successfully, you must belong to a role that includes the "Manage Design/Change-Time Policies" permission. If the archive contains a system-wide policy or custom action templates, you must also belong to a role that has the "Manage System-Wide Design/Change-Time Policies" permission.

- When the imported policy *is added* to the registry, the import process assigns the Organization, Lifecycle State and Activation State properties to the imported policy as follows.

Property	Value assigned to new policy
Organization	The organization of the user that is performing the import, unless the policy is a system-wide policy. If the policy is system-wide, the policy's **Organization** attribute will remain set to "All".
Lifecycle State	The initial state for new policies. **Note:** If you are using the lifecycle model for policies that is installed with CentraSite, the imported policy's state will be set to "New."
Activation State	Inactive

  Additionally, CentraSite assigns instance-level permissions to the imported policy just as though you created the policy manually. (In other words, the imported policy receives the same permission settings as the policies you create from scratch.)

- When an imported policy *replaces* (updates) an existing policy in the target registry, all of the policy's properties, except for its permission settings, are updated according to the policy object in the archive. This includes the policy's organizational scope, its lifecycle state, and its activation state (i.e., whether the policy is active or inactive). If the referenced organization and/or lifecycle model does not already exist on the target registry, the import process will fail. Also, be aware that the import process will replace the policy on the target regardless of whether the target policy is currently active or inactive. Due to this behavior, you might want to import only new versions of a policy, and not use the import process to directly replace a version of a policy that already exists.
- If the archive file contains a reference to an object that is not already present in the target registry or is not included in the archive file itself, the policy will not be imported.
- If design/change-time policies exist for the events that the import process initiates (e.g., the creation of a policy), those policies will be triggered.

To import a design/change-time policy

1. In CentraSite Control, go to **Policies** > **Design/Change Time**.
2. Click the **Import** icon.
3. Click **Browse** and select the zip file containing the policy that you want to import.
4. If you want the importer to automatically replace the policy if it already exists, select **Allow replace of existing objects**.
5. Click **OK**.
6. When the import process is complete, check the import log to make sure that the policy and its associated objects were imported successfully.
7. If the import process was successful, open the policy in CentraSite Control and do the following:

1. Inspect the parameter settings for each action in the policy to ensure that they are set properly. Specify appropriate values as necessary.
2. Inspect the other properties assigned to the policy and ensure that they are set appropriately.
3. Activate the policy.

Customizing the Predefined Lifecycle Model for Policies

CentraSite provides a lifecycle model for policies, which consists of four states: New, Productive, Suspended and Retired. Predefined policies associated with this lifecycle model activate and deactivate a policy as appropriate when you switch the policy's lifecycle state. The model is also structured in a way that allows CentraSite to automatically deactivate the old version of a policy when you activate a new one.

Because of the complex nature of this lifecycle model and its associated policies, you can make only the following kinds of changes to it:

- You can edit the lifecycle model's name, description and permission settings.
- You can rename the states in the lifecycle model.

To make these changes, you must create a new version of the lifecycle model and make your modifications to the new version as described in the section *Updating a Lifecycle Model* in the document *Customizing Lifecycle Management*.

You can also associate additional policies with the states in the lifecycle model. (Do not modify or delete the predefined policies that are associated with this lifecycle model, however.)

⚠ **Important:** Make only the kinds of customization described above. Do not add states to the model. Do not remove states from the model. Do not modify any of its transitions.

2 Using Approval Policies

This chapter covers the following topics:

About Approval Policies

CentraSite's approval-management framework enables you to create polices that trigger approval processes when certain time events occur in the registry. For example, you might create a policy that requires a system architect to review and approve all assets before they are switched to a productive state.

To impose an approval process on a change time event, you create an *approval policy* for the event. An approval policy is a policy that contains one of CentraSite's built-in *approval actions*.

Note: In this guide, the term *approval policy* is used to generally refer to policies that you use to perform approvals. Technically speaking, an approval policy is no different than an ordinary design/change-time policy. It is simply one that includes an approval action. An approval policy can also include other actions (assuming they are within the policy's scope).

Note: Approval policies are available only in *CentraSite ActiveSOA*.

The Approval Actions

CentraSite provides the following actions for obtaining approvals. To impose an approval process on an event, you include one of these actions in your policy.

Action Name	Description
Initiate Approval	This action submits a request to a designated group of approvers (referred to as the *approval group*). For information about using this action, see ***Using the Initiate Approval Action***.
Initiate Group-dependent Approval	This action submits a request to the approval group *only* if the requestor belongs to a specified user group. For information about using this action, see ***Using the Initiate Group-dependent Approval Action***.

What Happens When an Approval Action Is Enforced?

When a user performs an operation that triggers an approval policy, CentraSite initiates an approval workflow and submits the user's request to the designated group of approvers. Approvers receive the approval request in their “inbox” in CentraSite Control. Approvers whose user account includes a valid e-mail address also receive an email message informing them that a request is awaiting their approval. You can configure an approval action to send an email notification to other specified users, too.

Note: For CentraSite to issue email messages, an administrator must first configure CentraSite's email server settings. For procedures, see the section *Configuring the Email Server* in the document *Basic Operations*.

CentraSite does not execute the user's requested operation until it obtains the necessary approvals. If an approver rejects the request, CentraSite notifies the requestor and immediately exits the policy. It does not perform the user's requested operation nor does it execute any remaining actions in the approval policy. If other policies were to be executed against the user's request (i.e., if the request triggered lower priority policies in addition to the approval policy) those policies are not executed.

Using the **Inbox** on the My CentraSite page in CentraSite Control, users can view the status of the requests that they have submitted for approval. Approvers also use the **Inbox** to review and authorize requests that require their approval.

- For information about checking the status of requests that you have submitted for approval, see ***Reviewing Requests that You Have Submitted for Approval***.
- For information about reviewing requests that require your approval, see ***Approving a Request***.

Auto-Approval

When the user who submits a request is also an authorized approver for the requested operation, the request is *auto-approved*. (In other words, the requestor's approval is granted implicitly.)

The way in which a request is handled after it is auto-approved depends on whether the approval workflow is configured to execute in *Anyone* or *Everyone* mode.

- *In Anyone mode*, an auto-approval completes the approval process. Such requests do not formally initiate an approval workflow, however, they do appear in the Approval History log (the log will indicate that the request was auto-approved).
- *In Everyone mode*, the requestor's approval is registered and then the request is submitted to the remaining approvers in the approval group.

For more information about these two approval modes, see ***Approval Modes***.

Note: The auto-approval process also occurs when an approval action is invoked and all of its specified approver groups are empty or all users in the specified groups are inactive.

Approval Modes

You configure an approval action to operate in one of the following modes:

- **Anyone**
  In Anyone mode, a request can be approved or rejected by any single user in the approver group. In this mode, only one user in the group is required to approve or reject the request. This is the default mode.
- **Everyone**
  In Everyone mode, a request must be approved by all users in the approver group (it does not matter in which order the approvals are obtained). A rejection by any approver in the group will cause the request to be rejected.

What Types of Events and Objects Can Be Approved?

You can add approval policies for the following combinations of events and object types.

Event Type	Supported Object Types	Supported Approval Actions
PreStateChange	Asset Policy	Initiate Approval Initiate Group-dependent Approval
OnConsumerRegistration	Asset	Initiate Approval Initiate Group-dependent Approval

Using the Initiate Approval Action

You use the Initiate Approval action when you want to define an approval process that applies to *all of the users* who submit requests that trigger the policy. If you need to apply the approval process selectively, that is, if only certain groups of users require approval, or if different groups of users require authorization from different groups of approvers, use the **Initiate Group-dependent Approval** action instead.

The parameters required to define the action include the name of the approval flow that the action initiates, the name of the approver groups (i.e. the groups of users who are allowed to approve requests that trigger the policy), and email addresses of users who should be informed of the progress of the action.

The parameters are described in the description of the Initiate Approval action in the document *Built-In Design/Change-Time Actions Reference*.

Using the Initiate Group-Dependent Approval Action

When you want a policy to initiate an approval process for some groups of requestors and not for others, or when you need to route requests to different approvers based on the user group to which a requestor belongs, you use the *Initiate Group-dependent Approval* action.

The parameters required to define the action include the name of the approval flow that the action initiates, the name of the approver groups (i.e. the groups of users who are allowed to approve requests that trigger the policy), the names of the related triggering groups (i.e. the groups of members whose requests require approval), and email addresses of users who should be informed of the progress of the action.

The parameters are described in the description of the action Initiate Group-Dependent Approval in the document *Built-In Design/Change-Time Actions Reference*.

You can route approvals to different approver groups based on the triggering group to which the requestor belongs. For example, you could configure the action to route requests to the approver groups Approvers-A and Approvers-B when a requestor belongs to a particular triggering group.

Points to consider when using the Initiate Group-dependent Approval action:

- If a requestor does not belong to any of the groups specified in the **Triggering Groups** parameter, CentraSite does not even initiate an approval workflow. Approval is waived and CentraSite simply executes the next action in the policy. (Be aware that, because the request does not enter the approval framework, requests that are waived do not appear in the Approval History log.)
- The UI dialog allows you to combine triggering groups and approval groups into sets, where each set defines one or more triggering groups and the associated approver groups. You can specify multiple sets, and CentraSite processes each set in the order given in the dialog. When it encounters a set whose **Triggering Groups** parameter includes a user group to which the requestor belongs, it immediately initiates an approval workflow based on that set and ignores any remaining sets in the dialog. In other words, if the requestor is a member of multiple **Triggering Groups**, approval is determined by whichever of those groups appears first in the dialog.
- If a requestor is a member of both **Triggering Groups** and a member of **Approver Group** in the same triggering group/approver group combination, the request is **auto-approved**.

Switching the State of an Object when an Approval Request is Rejected

By default, an object's lifecycle state is not changed when an approval request is rejected. For example, let's say that object ABC is in the "Tested" state and an approval request is submitted to switch object ABC to the "Production" state. If the approval request is rejected, object ABC stays in the "Tested" state. For some approval workflows, however, you might want to switch objects to a particular state when they are rejected. To do this you use the **Reject State** parameter.

⚠ **Important:** If you use this option, make sure that the lifecycle model provides a transition from the state(s) that an object might be in when the approval policy executes and the state that you specify in the **Reject State** parameter. Otherwise, the approval engine will not be able to switch the target object to the specified state when a rejection occurs.

Also be aware that you can specify only one state in the **Reject State** parameter. Therefore, if an approval policy applies to objects with different lifecycle models, the **Reject State** can apply to only one of those models. For example, let's say you use the same approval policy for both XML schemas and services, but these two asset types follow different lifecycle models. If you set the **Reject State** to a state in the lifecycle model for XML schemas, only XML schemas will switch to this state when an approval request is rejected. Services, when rejected, will simply remain in their current state. If you want to specify one reject state for XML schemas and another for services, you must create a separate approval policy for each type.

Adding an Approval Policy to CentraSite

To create an approval policy, you must perform the following general steps:

1. If one does not already exist, create a user group composed of the individuals who are authorized to approve the type of request that triggers the policy. For information about creating user groups that represent authorized approvers, see *Approver Groups*.
2. Create a design/change-time policy with the appropriate scope (event type and object type) and into this policy, insert an approval action.
   - For information about the event types and object types with which you can use an approval action, see *What Types of Events and Objects Can Be Approved?*.
   - For general information about creating policies, see *Adding a Design/Change-Time Policy to CentraSite*.
   - For specific information about using policies with the PreStateChange event type, see *Using Approvals with PreStateChange Events*.

Including Multiple Actions in an Approval Policy

An approval policy can include actions in addition to the approval action. For example, you might create a policy like the example shown below, which validates a particular attribute in the asset and executes a custom action before it initiates the approval process.

Example

```
Validate Attribute Value
MyCustomAction
Initiate Approval
```

The example above illustrates how you can execute policy actions before you initiate the approval process. You can also insert actions after the approval action as long as those actions DO NOT attempt to modify the object on which the policy is acting. When an object enters an approval process, CentraSite locks the object to prevent any modifications to the object while it is undergoing approval. The object remains locked until the approval policy *and all additional policies that are triggered by the same event* are complete.

If an approval policy includes an action that attempts to update the object after approval process has been initiated, that action will fail. When this occurs, CentraSite immediately exits the policy and reverts the object to its previous state.

The following shows an approval policy that includes an action after the approval action. This policy will execute successfully, because the action following the approval action simply sends out an email notification. It does not attempt to modify the asset on which the policy is acting.

Example A (correct)

```
Validate Classification ↩
Set Instance and Profile Permissions ↩
Initiate Approval
Send Email Notification
```

The following shows an approval policy that would not execute successfully. In this example, the Set Instance and Profile Permissions action follows the Initiate Approval action. Because the asset is locked at this point in the policy, the Set Instance and Profile Permissions action will fail and the asset will revert to its previous lifecycle state.

Example B (incorrect)

```
Validate Classification ↩
Initiate Approval
Set Instance and Profile Permissions ↩
Send Email Notification
```

Tip: As a best practice, avoid executing any additional actions after the approval action in an approval policy. If there are actions that you need to execute after approval is granted, place those actions in a separate policy that executes on the PostStateChange event.

Using Approvals with PreStateChange Events

The PreStateChange event occurs when you change the lifecycle state of an object.

You can use an approval policy with the PreStateChange event to prevent users from switching the following types of objects to certain lifecycle states (e.g., to the Productive state) without first getting the required approvals:

- Policy
- Asset
- Lifecycle model

To create an approval policy that executes on a PreStateChange, you must perform the following general steps:

1. Make sure that the state change(s) that will trigger the policy are defined in an existing lifecycle model. If the lifecycle model, with the appropriate state, has not yet been defined, you must create it before you create the approval policy. For procedures, see the document *Customizing Lifecycle Management*.
2. Create a design/change-time policy with the following scope:

   Event Type: PreStateChange

   Object Type: Policy, Asset or Lifecycle Model

   For procedures, see *Adding a Design/Change-Time Policy to CentraSite*.
3. In the **Before the Object Enters State** section of the policy's **States** tab, specify the state change that requires approval. For procedures, see *Configuring Policies that Execute on Lifecycle State Changes*.
4. On the policy's **Actions** tab, specify and configure the approval action that is to be executed when an in-scope object switches to the state specified in the preceding step. If other actions

are to be executed before or after the approval action, insert those actions on the **Action** tab, too. For procedures, see ***Assigning Actions to a Design/Change-Time Policy***.

Caution: Only certain kinds of actions can be included *after* the approval action in an approval policy. Some actions, if they occur after the approval action, will cause the policy to fail. For information about what kind of actions can follow an approval action, see "**Including Multiple Actions in an Approval Policy**".

Using Approvals with OnConsumerRegistration Events

The OnConsumerRegistration event occurs when an asset owner reviews a consumer registration request and accepts the request by clicking the **Apply Registration Policies** button in the **Pending Registrations** view on the My CentraSite page.

Note: Because asset owners are not required to review consumer registrations that are initiated from an asset's **Consumers** profile, an approval policy that executes OnConsumerRegistration is not triggered when a consumer is registered in this manner. The only kinds of consumer registrations that are subject to an approval policy are those that are initiated using the **Register As Consumer** menu command.

As described in the section *Consumer Provisioning and Consumer-Provider Relationship Tracking* in the document *Working with Consumer Applications*, an organization must have a consumer-registration policy to process the consumer registrations that are initiated using the **Register As Consumer** menu command. At a minimum, this policy must include the Register Consumer action, because this action performs the work of actually registering a consumer (that is, it establishes the actual relationship between the asset and the specified consumers). If, in addition to the asset owner, you want designated individuals to review and approve the registration request, place an approval action before the Register Consumer action.

Note: The approval process that is imposed by a consumer-registration policy occurs *in addition* to the review and approval that is required by the asset owner. That is, the asset owner always reviews the registration first, and if he or she accepts the registration, the request proceeds through the approval process defined by the consumer-registration policy.

The following procedure describes the general steps you use to create a consumer-registration policy that includes an approval action.

1. Create a design/change-time policy with the following scope:
   - **Event Type:** OnConsumerRegistration
   - **Object Type:** Asset (of any type)

   If you need procedures for this step, see ***Adding a Design/Change-Time Policy to CentraSite***.

2. On the policy's **Actions** tab, add the following actions. Make sure the approval action *precedes* the Register Consumer action.

   - Initiate Approval —OR— Initiate Group-dependent Approval
   - Register Consumer

   If you need procedures for adding actions to a policy, see *Assigning Actions to a Design/Change-Time Policy*.

3. Configure the approval action's input parameters. If you need procedures for this step, see *Configuring Policy Action Parameters*.

4. Insert additional actions before and/or after this pair of actions as necessary.

   The following example shows an action list that obtains the required approval, executes the registration process, and then grants instance-level permissions to the consumers that the policy registers.

Example

```
Initiate Approval
Register Consumer ↩
Set Consumer Permission
```

Approver Groups

An approver group is simply a user group that identifies the set of individual who are authorized to approve a submitted request. An approver group can be composed of users from any organization.

Note: If you want approvers to be able to review the details for an object that they are asked to approve, make those users have View permission on the object. For example, if the users in group ABC will be required to approve assets that are switched to a certain lifecycle state, make sure that the users in group ABC have View permission on the assets that they will be asked to approve. Without View permission, approvers will not be able to examine the details of the assets that users submit to them for approval.

Changing the Membership of an Approver Group

Changing the membership of an approver group *does not* affect requests that are already pending approval. When CentraSite submits a request to the approval engine, it assigns the users from the specified approver group to that request. The request retains its assigned set of approvers throughout the entire approval process.

For example, let's say that approval policy P1 uses approver group AG1, and that AG1 contains users A and B. If a user submits a request that triggers P1, users A and B will become the designated approvers for that request. Let's say that while this request is waiting for approval, an administrator modifies group AG1 and replaces users A and B with users X and Y. This change will have *no effect* on the request that is awaiting approval. Users A and B will continue to its designated approvers. The changes to group AG1 will only affect new requests that policy P1 submits for approval.

Reviewing Requests that You Have Submitted for Approval

In the Approval History log, CentraSite maintains a record of every request that users submit for approval. You can use the following procedure to view your requests and examine their status.

Note: The list displays *all* requests that have been submitted on your behalf, including requests that were auto-approved.

To view requests that you have submitted for approval

1 In CentraSite Control, go to **Home** > **My CentraSite** to display the My CentraSite page.

2 Click **Menu** > **Inbox** > **Approvals** > **Approval Requests** to display the list of requests that you have submitted for approval.

The **Status** column in the **Approval Requests** list indicates the state of each request as follows:

Status	Description
Pending	The request has been submitted for approval, but has not yet been processed by the required approvers.
Approved	The request has been submitted and approved by the required approvers.
Rejected	The request has been submitted and rejected. The operation you requested was not executed.
Auto-Approved	The request was **auto-approved**. This occurs when you submit a request for which you are also an authorized approver.

3 To examine the details for a particular request (including a list of the individuals who are authorized to approve the request), click any "non-linked" area in the row that contains the request. CentraSite Control

The details for the request will appear in the **Approval Flow Information** panel.

Approving a Request

If you are an approver, CentraSite places requests in your **Pending Approvals** inbox for your review and approval.

To view and approve requests in your inbox

1. In CentraSite Control, go to **Home** > **My CentraSite** to display the My CentraSite page.
2. Click **Menu** > **Inbox** > **Approvals** > **Pending Approvals** to display the list of requests that require your approval.
3. Choose the request that you want to review by clicking any "non-linked" area within the row that contains the request.

   Note: If you want to examine the object on which the approval is requested, click the object name in the **Approval Request** column. If you have View permissions on the object, you will be allowed to view the object's details.

4. In the **Comment** text box, type a comment. (e.g., *"Request rejected. Add required specifications to this asset and resubmit"*.)
5. Click the **Accept** or **Reject** button as appropriate to approve or reject the request.

Viewing Your Approval History

The **Approval History** link in your inbox displays all requests for which you were an authorized approver (i.e., the list includes any request whose approver group included you as a member).

To view your approval history

1. In CentraSite Control, go to **Home** > **My CentraSite** to display the My CentraSite page.
2. Click **Menu** > **Inbox** > **Approvals** > **Approval History** to display the list of requests for which you were an authorized approver.
3. To examine the details for a particular request, including the list of other authorized approvers, choose the approval workflow in the **Approval Request** column.

Viewing the Approval History Log

Use the following procedure to display the Approval History log. This log contains a record of all approval requests that have been submitted to CentraSite. To view the Approval History log, you must belong to a role that has the "View Approval History" permission. If you belong to the CentraSite Administrator role, you will see all entries in the Approval History log. Otherwise, you will see only the set of approval requests that were triggered within your organization.

For additional information about the approval log, see the document *Logging*.

To view the Approval History log

- In CentraSite Control, go to **Administration** > **Logs** > **Approval History**.

Reverting the State of an Object that is Pending Approval

Occasionally, you might need to revert a request that has been submitted for approval. For example, if a request that has already been submitted to the approval engine requires the approval of a user who has left the company, you will need to back that request out of the approval engine and re-submit it (after updating the approver group, of course).

When you have an approval request that is stuck in the "pending" mode, a user in the CentraSite Administrator role can use the following procedure to revert the object to its previous state so that the condition can be corrected and the object can be resubmitted for approval.

Note: Reverting the lifecycle state of an asset does not undo any attribute changes that might have been made by policies that were executed by the original state-change event. It simple returns the asset's lifecycle property to its previous state. If other attribute changes occurred during the state-change event, you will need to undo those changes manually.

To revert the state of an object that is pending approval

1 In CentraSite Control, use one of the following steps to display the list that contains the object whose pending state you want to revert.

To revert the state of this type of object...	Do this in CentraSite Control,..
Asset	Go to **Asset Catalog** > **Browse**.
Design/Change-Time Policy	Go to **Policies** > **Design/Change-Time**.
Run-Time Policy	Go to **Policies** > **Run-Time**.
Lifecycle Model	Go to **Administration** > **Lifecycles** > **Models**.

2 Locate the object whose state you want to revert and choose **Revert Pending State** from its context menu.

3 **Note:** For assets, you can also perform the **Revert Pending State** command from the **Actions** menu on the asset detail page.

Using the Approval Service API

CentraSite provides a Web service that enables you to create applications and/or integrate third-party workflow tools with CentraSite's approval queue. This service provides operations that enable you to obtain the list of pending requests for an approver and approve or reject those requests via a Web service. The Web service also provides operations for viewing the approval history log.

For more information about using the Approval service, see the section *Approval Service* in the document *CentraSite Web Service Interfaces*.

3 Working with EMail Notifications

Certain policy actions, such as the Send Email Notification action and the approval actions, send email messages to users when specified events occur. For example, you might use the Send Email Notification action to alert a certain group of administrators when an asset switches to a particular state. Or, you might issue an email alert to certain users when an approval request is rejected.

Setting the Email-Related Parameters in an Email Notification Action

Actions that send email notifications to users (such as the Send Email Notification action) require you to specify the following input parameters.

Set this parameter...	To specify...

`Users`
— AND/OR —
`Groups`
	The users to whom the email message is to be sent. You can use the `Users` parameter to specify individual users, the `Groups` parameter to specify groups of users, or both.

`Subject`	The text that will appear on the subject line of the email.

`Use Email Template`
— OR —
`Custom Message`
	The body of the message. You can specify the body of the email by typing a message directly into the `Custom Message` parameter or by using an email template. For more information about these options, see ***Using a Custom Message in an Email Notification Action*** and ***Using Email Templates with Policy Actions***.

You can include *substitution tokens* in the subject line and/or the body of the email message. Substitution tokens enable you to incorporate run-time information into the email. For example, you can use the `${user.name}` token to insert (into the email message or the subject line) the name of the user who triggered the policy. For a complete list of the supported substitution tokens, see the description of the action Send Email Notification in the section *Built-In Actions for Design/Change-Time Policies* of the document *Built-In Design/Change-Time Actions Reference*.

Using a Custom Message in an Email Notification Action

One way to specify the body of the email message is to simply type the message directly into the `Custom Message` parameter. If you specify the body of the message in this way, you must set the `Format` parameter to indicate whether the message is to be sent as plain text or as an HTML document. When you use HTML format, you must enclose the message text in <html> and <body> tags.

Example of a Plain Text Message

```
Virtual Service ${entity.name} has been placed in production by ${user.name}.
To view the virtual service, go to: ${entity.URL}
```

Example of an HTML Message

```
<html>
<body>
<p>Virtual Service <b>${entity.name}</b> has been placed in production by ↩
<b>${user.name}</b>.</p>
<p>To view the virtual service, go to: <a href="${entity.URL}">${entity.URL}</a></p>
</body>
</html>
```

Using Email Templates with Policy Actions

To generate the body of an email message from an email template, the email template must exist in CentraSite's repository. If the template does not already exist, you must create it and upload it to the repository using the EmailTemplateManager command-line utility. (Several predefined templates are available for you to use with various policy actions. For a list, see ***Predefined Email Templates Installed with CentraSite***.)

The following describes how to create an email template and upload it to the repository. It also describes how to edit a template, download a template, delete a template and obtain a list of the templates that already exist in the repository.

Note: To work with email templates, you must have access to the command line on the machine where CentraSite is installed. Additionally, if you want to upload templates or delete templates, you must have a CentraSite user account that belongs to the CentraSite Administrator role. To view the list of email templates or to download a template, you simply need a CentraSite user account. (In other words, your CentraSite account does not require any explicit permissions. All CentraSite users have View permission on email templates.)

Creating a Custom Email Template

An email template is a text file that contains an HTML document. Your HTML document should include the <html> and <body> tags as shown in the example below. (The inclusion of a <head> tag is optional. CentraSite does not require this tag in an email template.)

Example of an Email Template

```
<html>
<body>
   <p>Virtual Service <b>${entity.name}</b> has been placed in production by ↩
<b>${user.name}</b>.</p>
  <p>To view the virtual service, go to: <a href="${entity.}">${entity.URL}</a></p>
</body>
</html>
```

Adding an Email Template to CentraSite

To add an email template to CentraSite, you must create a script file that executes the EmailTemplateManager utility. Then you must run the script file with the -t *`filename`* input parameter.

⚠ **Important:** The template's filename is used to uniquely identify the template within CentraSite. If you upload a template that has the same filename name as a template that already resides in the repository, the new template will automatically *replace* the existing one.

- From a Java Class
- From a Command Line

From a Java Class

To add an Email Template to CentraSite from a Java Class

1 Create an email template file as described in ***Creating an Email Template***. Copy the file somewhere within the file system of the machine where CentraSite is installed.

2 Create a script file as described in ***Creating a Script File for the EmailTemplateManager Utility***.

3 Execute the script file with the following parameters:

```
yourScriptFile -t templateFile -dbuser yourCSUserID -dbpassword yourPassword
```

Example

```
myScript -t d:\myDirectory\myEmailTemplate.html -dbuser jcallen -dbpassword
j45Hk19a
```

From a Command Line

To add an email template to CentraSite, run the utility with the following input parameters:

- Under Windows:
- Under Linux:
- Input Parameters

Under Windows:

Use the following procedure to add an email template under Windows:

1. Open *ConfigEmailTemplates.cmd* in a text editor.
2. Add the following property statement:

```
<CentraSiteInstallDir>\utilities\ ConfigEmailTemplates.cmd -dbuser <USERNAME> ↩
-dbpassword <PASSWORD> [-dburl <CENTRASITE-URL>] -t <TEMPLATE-FILE>
```

where, `<CentraSiteInstallDir>` is the CentraSite installation directory. By default, this is the *CentraSite* folder under `<SuiteInstallDir>`.

Under Linux:

Use the following procedure to add an email template under Linux:

1. Open *ConfigEmailTemplates.sh* in a text editor.
2. Add the following property statement:

```
<CentraSiteInstallDir>\utilities\ ConfigEmailTemplates.sh -dbuser <USERNAME> ↩
-dbpassword <PASSWORD> [-dburl <CENTRASITE-URL>] -t <TEMPLATE-FILE>
```

where, `<CentraSiteInstallDir>` is the CentraSite installation directory. By default, this is the *CentraSite* folder under `<SuiteInstallDir>`.

Input Parameters

The following table describes the complete set of input parameters that you can use with the EmailTemplateManager utility:

Parameter	Description
`USERNAME`	*Required*. Your CentraSite user ID.
`PASSWORD`	*Required*. The password for your CentraSite user account.
`CENTRASITE-URL`	The fully qualified URL for the CentraSite registry/repository. If you omit this parameter, the importer assumes that the registry/repository resides at `http://localhost:53307/CentraSite/CentraSite`. **Note:** If the registry/repository is running on a different machine and port number, you can use this parameter to specify its location instead of using the individual -h and -p parameters. (If you specify the -dburl parameter with the -h and/or -p parameters, the -h and -p parameters will be ignored.)
`TEMPLATE-FILE`	The URI (file: or http:) of the email template.

Viewing the List of Email Templates on CentraSite

To list the email templates that exist in the CentraSite repository, do either of the following:

- From a Java Class
- From a Command Line

From a Java Class

To list the email templates in the repository

1 Create a script file as described in ***Creating a Script File for the EmailTemplateManager Utility***.

2 Execute the script file with the following parameters:

```
yourScriptFile -list -dbuser yourCSUserID -dbpassword yourPassword
```

Example

```
myScript -list -dbuser jcallen -dbpassword j45Hk19a
```

From a Command Line

To list the email templates in the repository, run the utility with the following input parameters and options:

- Under Windows:
- Under Linux:

- Input Parameters and Options

Under Windows:

Use the following procedure to list the email templates under Windows:

1. Open *ConfigEmailTemplates.cmd* in a text editor.
2. Add the following property statement:

```
<CentraSiteInstallDir>\utilities\ ConfigEmailTemplates.cmd -dbuser <USERNAME> ↩
-dbpassword <PASSWORD> -list
```

   where, `<CentraSiteInstallDir>` is the CentraSite installation directory. By default, this is the *CentraSite* folder under `<SuiteInstallDir>`.

Under Linux:

Use the following procedure to list the email templates under Linux:

1. Open *ConfigEmailTemplates.sh* in a text editor.
2. Add the following property statement:

```
<CentraSiteInstallDir>\utilities\ ConfigEmailTemplates.sh -dbuser <USERNAME> ↩
-dbpassword <PASSWORD> -list
```

   where, `<CentraSiteInstallDir>` is the CentraSite installation directory. By default, this is the *CentraSite* folder under `<SuiteInstallDir>`.

Input Parameters and Options

The following tables describe the complete set of input parameters and options that you can use with the EmailTemplateManager utility:

Parameter	Description
USERNAME	*Required*. Your CentraSite user ID.
PASSWORD	*Required*. The password for your CentraSite user account.
CENTRASITE-URL	The fully qualified URL for the CentraSite registry/repository. If you omit this parameter, the importer assumes that the registry/repository resides at `http://localhost:53307/CentraSite/CentraSite`. **Note:** If the registry/repository is running on a different machine and port number, you can use this parameter to specify its location instead of using the individual -h and -p parameters. (If you specify the -dburl parameter with the -h and/or -p parameters, the -h and -p parameters will be ignored.)

Option	Description
`-list`	Lists all the published email Templates.

Updating an Existing Email Template on CentraSite

To update an existing template in the repository, do the following:

1. Download the existing template from the repository using the procedure in ***Downloading an Email Template from CentraSite***.
2. Edit the template as necessary.
3. Upload the updated template file to the repository using the procedure in ***Adding an Email Template to CentraSite***.

Important: If you want to update an existing template, make sure the template file that you upload has exactly the same filename as the one you want to replace.

Downloading an Email Template from CentraSite

To download a template from the CentraSite repository, do either of the following:

- From a Java Class
- From a Command Line

From a Java Class

To download an email template from the repository

1 Create a script file as described in ***Creating a Script File for the EmailTemplateManager Utility***.

2 Execute the script file with the following parameters:

```
yourScriptFile -download templateName -tolocation targetDirectory -dbuser yourC-
SUserID -dbpassword yourPassword
```

Example

```
myScript -download myEmailTemplate.html -tolocation d:\myDir\mySubDir -dbuser
jcallen -dbpassword j45Hk19a
```

From a Command Line

To download a template from the repository, run the utility with the following input parameters:

- Under Windows:
- Under Linux:
- Input Parameters

Under Windows:

Use the following procedure to download a template under Windows:

1. Open *ConfigEmailTemplates.cmd* in a text editor.
2. Add the following property statement:

```
<CentraSiteInstallDir>\utilities\ ConfigEmailTemplates.cmd -dbuser <USERNAME> ↩
-dbpassword <PASSWORD> [-dburl <CENTRASITE-URL>] -download <EMAIL-TEMPLATE> ↩
-tolocation <LOCATION>
```

where, `<CentraSiteInstallDir>` is the CentraSite installation directory. By default, this is the *CentraSite* folder under `<SuiteInstallDir>`.

Under Linux:

Use the following procedure to download a template under Linux:

1. Open *ConfigEmailTemplates.sh* in a text editor.
2. Add the following property statement:

```
<CentraSiteInstallDir>\utilities\ ConfigEmailTemplates.sh -dbuser <USERNAME> ↩
-dbpassword <PASSWORD> [-dburl <CENTRASITE-URL>] -download <EMAIL-TEMPLATE> ↩
-tolocation <LOCATION>
```

where, `<CentraSiteInstallDir>` is the CentraSite installation directory. By default, this is the *CentraSite* folder under `<SuiteInstallDir>`.

Input Parameters

The following table describes the complete set of input parameters that you can use with the EmailTemplateManager utility:

Parameter	Description
USERNAME	*Required*. Your CentraSite user ID.
PASSWORD	*Required*. The password for your CentraSite user account.
CENTRASITE-URL	The fully qualified URL for the CentraSite registry/repository. If you omit this parameter, the importer assumes that the registry/repository resides at `http://localhost:53307/CentraSite/CentraSite`. **Note:** If the registry/repository is running on a different machine and port number, you can use this parameter to specify its location instead of using the individual -h and -p parameters. (If you specify the -dburl parameter with the -h and/or -p parameters, the -h and -p parameters will be ignored.)
EMAIL-TEMPLATE	The name of an existing email template.
LOCATION	The location to save an existing email template.

Deleting an Email Template from CentraSite

To delete an email template from the CentraSite repository, do either of the following:

Note: The EmailTemplateManager utility will not allow you to delete any of the predefined email templates that are installed with CentraSite. Neither will it allow you to delete any template that is used by an existing policy.

- From a Java Class
- From a Command Line

From a Java Class

To delete an email template from the repository

1 Create a script file as described in ***Creating a Script File for the EmailTemplateManager Utility***.

2 Execute the script file with the following parameters:

```
yourScriptFile -delete templateName -dbuser yourCSUserID -dbpassword yourPassword
```

Example

```
myScript -delete myEmailTemplate.html -dbuser jcallen -dbpassword j45Hk19a
```

From a Command Line

To delete an email template from the repository, run the utility with the following input parameters:

- Under Windows:
- Under Linux:
- Input Parameters

Under Windows:

Use the following procedure to delete a template under Windows:

1. Open *ConfigEmailTemplates.cmd* in a text editor.
2. Add the following property statement:

```
<CentraSiteInstallDir>\utilities\ ConfigEmailTemplates.cmd -dbuser <USERNAME> ↩
-dbpassword <PASSWORD> [-dburl <CENTRASITE-URL>] -delete <EMAIL-TEMPLATE>
```

   where, `<CentraSiteInstallDir>` is the CentraSite installation directory. By default, this is the *CentraSite* folder under `<SuiteInstallDir>`.

Under Linux:

Use the following procedure to delete a template under Linux:

1. Open *ConfigEmailTemplates.sh* in a text editor.
2. Add the following property statement:

```
<CentraSiteInstallDir>\utilities\ ConfigEmailTemplates.sh -dbuser <USERNAME> ↩
-dbpassword <PASSWORD> [-dburl <CENTRASITE-URL>] -delete <EMAIL-TEMPLATE>
```

   where, `<CentraSiteInstallDir>` is the CentraSite installation directory. By default, this is the *CentraSite* folder under `<SuiteInstallDir>`.

Input Parameters

The following table describes the complete set of input parameters that you can use with the EmailTemplateManager utility:

Parameter	Description
USERNAME	*Required*. Your CentraSite user ID.
PASSWORD	*Required*. The password for your CentraSite user account.
CENTRASITE-URL	The fully qualified URL for the CentraSite registry/repository. If you omit this parameter, the importer assumes that the registry/repository resides at `http://localhost:53307/CentraSite/CentraSite`. **Note:** If the registry/repository is running on a different machine and port number, you can use this parameter to specify its location instead of using the individual -h and -p parameters. (If you specify the -dburl parameter with the -h and/or -p parameters, the -h and -p parameters will be ignored.)
EMAIL-TEMPLATE	The name of an existing email template.

Creating a Script File for the EmailTemplateManager Utility

The EmailTemplateManager is a Java class whose `main()` method executes when you execute the EmailTemplateManager from the command line. To ensure that the CLASSPATH and other environment variables are set properly when you execute this utility, you must create a script file that calls the EmailTemplateManager as described below.

Creating a Script File for Windows (a .bat file)

Create a script file that looks as follows if CentraSite is running under Windows.

```
@echo off
set JAVAEXE=fullPathToJava.exe
set REDIST=CentraSiteHomeDirectory\redist
set BASEDIR=%~dp0
cd /d %REDIST%

REM build CLASSPATH with all files from jar directory
set LOCAL_CLASSPATH=
for %%I in (".\*.jar") do call "CentraSiteHomeDirectory\bin\cfg\lcp.cmd" %%I

%JAVAEXE% -cp %LOCAL_CLASSPATH% com.centrasite.util.EmailTemplateManager %*
cd /d %BASEDIR%
```

Example

```
@echo off
REM
REM Run Email Template Manager Utility
REM
set JAVAEXE=D:\software\java\jdk1.5.0_12\bin\java
set REDIST=C:\SoftwareAG\CentraSite\redist
set BASEDIR=%~dp0
cd /d %REDIST%

REM build CLASSPATH with all files from jar directory
set LOCAL_CLASSPATH=
for %%I in (".\*.jar") do call "C:\SoftwareAG\CentraSite\bin\cfg\lcp.cmd" %%I

%JAVAEXE% -cp %LOCAL_CLASSPATH% com.centrasite.util.EmailTemplateManager %*
cd /d %BASEDIR%
```

Creating a Script File for Unix (C-shell script)

Create a script file that looks as follows if CentraSite is running under Unix.

```
set javaexe="fullPathToJava.exe"
set redist="fullPathToJava.exe/redist"
set mainjar="CentraSiteUtils.jar"
set delim='\:'
cd "$redist"
set cl=""
foreach j ( `ls *.jar` )
 if ($cl != "") set cl=${cl}${delim}
 set cl=${cl}${j}
end
setenv CLASSPATH ${mainjar}${delim}${cl}
$javaexe com.centrasite.util.EmailTemplateManager $*
```

Example

```
#!/bin/csh
#
# Run Email Template Manager Utility
#
set javaexe="/.../softwareag/cjp/v16/bin/java"
set redist="/.../softwareag/CentraSite/redist"
set mainjar="CentraSiteUtils.jar"
set delim='\:'
# build CLASSPATH with all files from jar directory
cd "$redist"
set cl=""
foreach j ( `ls *.jar` )
  if ($cl != "") set cl=${cl}${delim}
  set cl=${cl}${j}
end
```

```
setenv CLASSPATH ${mainjar}${delim}${cl}
$javaexe com.centrasite.util.EmailTemplateManager $*
```

Executing the EmailTemplateManager Script File

To run the EmailTemplateManager, execute your script file on the machine where CentraSite is installed.

Note: CentraSite must be running when you execute the script file.

When you execute your script file, you must include input parameters on the command line to specify what you want the EmailTemplateManager to do. You must also include parameters to specify your CentraSite user ID and password. For example, to delete a template from the directory, you would run your script as follows:

`yourScriptFile -delete templateName -dbuser yourCSUserID -dbpassword yourPassword`

Example

```
myScript -delete myEmailTemplate.html -dbuser jcallen -dbpassword j45Hk19a
```

Note: If you execute your script without supplying any input parameters, the EmailTemplateManager will display the list of all supported parameters.

Using the -dburl Input Parameter

The EmailTemplateManager utility assumes that the CentraSite registry/repository is running at `http://localhost:53307/CentraSite/CentraSite` (i.e., it assumes that it is installed on the same machine as the EmailTemplateManager utility). If you installed the registry/repository component on a different machine than the CentraSite Application Server Tier (CAST) or if you configured the registry/repository to run on a different port than 53307, you will need to use the -dburl parameter to specify the address of the registry/repository when you run your script.

Example

```
myScript -delete myEmailTemplate.html -dburl ↩
http://rubicon:53307/CentraSite/CentraSite -dbuser jcallen -dbpassword j45Hk19a ↩
```

Predefined Email Templates Installed with CentraSite

The following predefined email templates are installed with CentraSite. These templates are provided for you to use with the policy actions listed below if you do not want to create your own email templates. For information about the policy actions, see the section *Built-In Actions for Design/Change-Time Policies* in the document *Built-In Design/Change-Time Actions Reference*.

Template Name	Description	Meant to be used with...
ApprovalNotification.html	Default email template used when an approval request is approved.	The `Send Approval Email` parameter in the Initiate Approval or Initiate Group-Dependent Approval action.
AutoApprovalNotification.html	Template used in earlier versions of CentraSite to indicate that a request had been auto-approved. This template is no longer used by CentraSite.	The Send Approval Email parameter in the Initiate Approval or Initiate Group-Dependent Approval action. (This template would only be useful in cases where all approval requests processed by the policy action were expected to be auto-approved.)
ChangeNotification.html	Default email template used for change notifications.	The `Use Email Template` parameter in the Send Email Notification action.
PendingNotification.html	Default email template used when an approval request is submitted to approvers.	The `Send Pending Approval Email` parameter in the Initiate Approval or Initiate Group-Dependent Approval action.
RejectApprovalNotification.html	Default email template used when an approval request is rejected.	The `Send Rejection Email` parameter in the Initiate Approval or Initiate Group-Dependent Approval action.

4 Working with Predefined Policies

System policies are predefined, design-time policies that CentraSite uses to perform internal operations (e.g., identifying the components associated with a given object) and registry-wide governance functions (e.g., ensuring the validity of policies prior to activation). Policies that are classified as predefined policies are system-wide in scope and execute at priority levels that are reserved for predefined policies. System policies are applied to assets regardless of the asset type's **Policies can be applied** property.

By default, predefined policies are not displayed by CentraSite Control. To view predefined policies, you must enable the **Show Predefined Policies** option on the Design/Change-Time Policy page.

If you belong to a role that includes the "Manage System-Wide Design/Change-Time Policies" permission, you have the ability to edit, delete and deactivate CentraSite's predefined policies. However, you should not do this. These policies perform critical functions within the registry and must not be edited, deleted, or deactivated except under the direction of a technical representative from Software AG.

The following sections describe predefined policies.

Collector and Handler Policies

The collector and handler policies are used to delete, move and export *composite objects* in a consistent way. Composite objects are objects that are made up, in part, of other registry objects. A Service object, for example, includes Operation objects, Binding objects and Interface objects. When you delete, move, or export a Service object, you want CentraSite to delete, move, or export the Service object *and* its related components. The collector and handler policies provide a way to consistently identify the set of registry objects that make up a composite object and treat those objects as a unit when performing delete, move and export operations.

To understand collectors and handlers, you must understand the concepts of *shared components*, *nonshared components* and *required objects*. If you are not already familiar with these concepts, review the material in the section *Working with Composite Objects* in the document *Object Type Management*.

- The Default Export Handler

What Are the Roles of the Collector and Handler Policies?

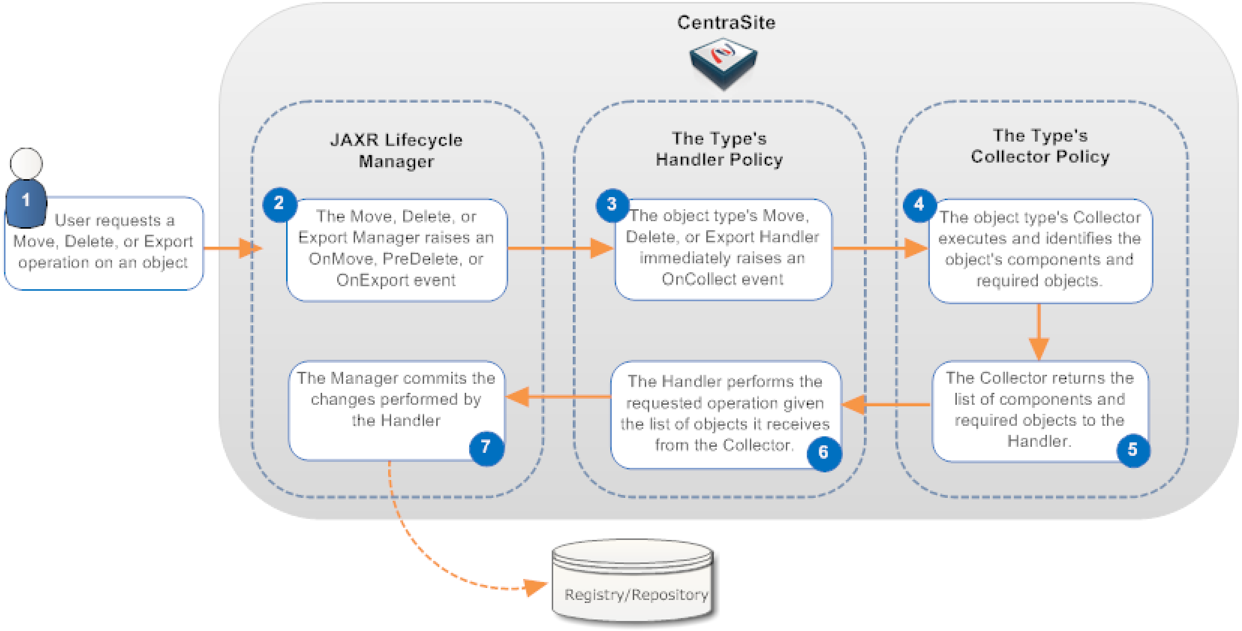
Handler policies and collector policies operate together to delete, move and export composite objects.

- The collector policy produces a list of the components (shared and nonshared) and required objects that are associated with a given instance of a composite object.
- The handler policy performs the delete, move, or export operation (depending on the type of handler that has been invoked) on the given object based on the list of components and required objects the handler receives from the collector.

For example, when you delete an instance of a Service object, the Service delete handler policy invokes the Service collector policy to identify the set of components and required objects associated with that particular instance of a Service (e.g., its operations, bindings, interfaces, XML schemas, supporting documents and so forth). The delete handler then deletes the Service object and all of the nonshared components that were identified by the Service collector.

The following diagram illustrates how the handler and collector policies interact during a move, delete, or export operation.

The Handler and Collection Process


CentraSite
JAXR Lifecycle Manager
The Type's Handler Policy
The Type's Collector Policy
1 User requests a Move, Delete, or Export operation on an object
2 The Move, Delete, or Export Manager raises an OnMove, PreDelete, or OnExport event
3 The object type's Move, Delete, or Export Handler immediately raises an OnCollect event
4 The object type's Collector executes and identifies the object's components and required objects.
5 The Collector returns the list of components and required objects to the Handler.
6 The Handler performs the requested operation given the list of objects it receives from the Collector.
7 The Manager commits the changes performed by the Handler
Registry/Repository

How Collectors and Handlers are Associated with a Type

An asset type must specify the handler and collection policies that are to be used to delete, move and export instances of its type. This is accomplished by defining a "HasHandler" association between the Type object and its handler and collection policies.

All types must have a "HasHandler" association to the four types of policies listed below.

Kind of Policy	Description
Collector policy	A policy that executes on the OnCollect event and generates a list of the nonshared components, shared components and required objects associated with a given instance of the type.
Delete Handler policy	A policy that executes on the PreDelete event and deletes the components for a given instance of the type.
Move Handler policy	A policy that executes on the OnMove event and moves the components for a given instance of the type to another user and/or organization.
Export Handler policy	A policy that executes on the OnExport event and exports the components and required objects for a given instance of the type.

⚠ **Important:** An asset type must have one (and only one) of each of the four kinds of policies shown above.

The Collector and Handler Policies Provided with CentraSite

The following sections describe the predefined collector and handler policies installed with CentraSite.

The Default Collector and Handler Policies

The following table lists the set of default collector and handler policies that CentraSite uses for deleting, moving and exporting registry objects. The default collector and handler policies are used by many of the predefined types installed with CentraSite.

CentraSite also assigns the default handler policies to custom types that you create. CentraSite does not assign the default collector policies to custom types; this means that the default handler policies will continue as if the collector returned no objects.

Because CentraSite uses these policies for many of the predefined types and also assigns the handler policies to new types by default, you must not edit, delete or deactivate them.

Policy Name	For more information
Default Collector	See *The Default Collector*.
Default Delete Handler	See *The Default Delete Handler*.
Default Move Handler	See *The Default Move Handler*.
Default Export Handler	See *The Default Export Handler*.

Specialized Collector and Handler Policies for Assets

The following table lists the specialized collector and handler policies that CentraSite uses for deleting, moving and exporting instances of certain predefined asset types.

The component lists for the various object types mentioned in the table are described in the section *Working with Composite Types* in the document *Object Type Management*.

You must not edit, delete, or deactivate any of the following policies.

Policy Name	Description
Collector Policy For BPEL Process	Performs the collection process for BPEL Process objects. For a list of the components that this collector returns, see the BPEL Process component list.
Collector Policy For Process	Performs the collection process for Process objects. For a list of the components that this collector returns, see the Process component list.
Collector Policy for REST Service	Performs the collection process for REST Service objects. For a list of the components that this collector returns, see the XML/REST Service component list.
Collector Policy for Schema	Performs the collection process for XML Schema objects. For a list of the components that this collector returns, see the XML Schema component list.
Collector Policy for Virtual REST Service	Performs the collection process for Virtual REST Service objects. For a list of the components that this collector returns, see the Virtual XML/REST Service component list.
Collector Policy for Virtual Service	Performs the collection process for Virtual Service objects. For a list of the components that this collector returns, see the Virtual Service component list.
Collector Policy for Virtual XML Service	Performs the collection process for Virtual XML Service objects. For a list of the components that this collector returns, see the Virtual XML/REST Service component list.
Collector Policy for Web Service	Performs the collection process for Service objects. For a list of the components that this collector returns, see the Service component list.
Collector Policy for WS-Policy	Performs the collection process for WS-Policy objects. For a list of the components that this collector returns, see the WS-Policy component list.

Policy Name	**Description**
Collector Policy for XML Service	Performs the collection process for XML Service objects. For a list of the components that this collector returns, see the XML/REST Service component list.
Collector Policy for IS Service Interface	Performs the collection process for Integration Server (IS) Service Interface objects. For a list of the components that this collector returns, see the IS Service Interface component list.
Virtual Service Export Handler	Performs the export process for Virtual Service objects.

Specialized Collector and Handler Policies for Other Registry Objects

The following table lists the set of collector and handler policies that CentraSite uses for deleting, moving and exporting instances of certain registry objects that are not assets. You must not edit, delete, or deactivate these policies.

Policy Name	**Description**
Asset Type Export Handler	Performs the export process for asset type definitions.
Collector Policy For Lifecycle Model	Performs the collection process for lifecycle models.
Collector Policy For Policy	Performs the collection process for Policy objects.
Default User Move Handler	Performs the move operation on users.
Taxonomy and Category Export Handler	Performs the export process for taxonomies and their categories.

The Default Collector

The Default Collector policy identifies the components and required objects for a given object. It does this by looking for specific kinds of associations between the composite object and other objects in the registry.

How the Collector Locates the Components that are Associated with a Composite Object

To identify the components of a composite object, the collector finds objects that are related to the composite object by an *aggregation relationship* or a *reverse-aggregation relationship*.

- An aggregation relationship is indicated by the presence of an *aggregated* Relationship attribute in the composite object. Like a regular Relationship attribute, an aggregated Relationship attribute associates an asset with other objects in the registry. However, an aggregated Relationship attribute additionally indicates that the associated objects are components of the object that contains the aggregated Relationship attribute.
- A reverse-aggregation relationship is indicated by a *reverse-aggregated* Relationship attribute that is present in another registry object and points back to the object on which the collection is being performed. Like a regular Relationship attribute, a reverse-aggregated Relationship attribute also associates an asset with other objects in the registry. However, a reverse-aggregated Rela-

tionship attribute additionally indicates that the object that contains the reverse-aggregated attribute is a component of the object at the end of the relationship.

Conceptually, both the aggregated and reverse-aggregated forms of the Relationship attribute establish a parent-child relationship between the object being collected (the parent) and objects that are its components (its children). However, the aggregated form expresses the relationship from the perspective of the parent object (i.e., the aggregated Relationship attribute exists in the composite object and identifies the object's components), whereas, the reverse-aggregated form expresses the relationship from the perspective of a child object (i.e., the reverse-aggregated Relationship attribute exists in a component object and identifies the composite object to which it belongs). The components that make up a composite object can be identified using aggregated Relationship attributes, reverse-aggregated Relationship attributes or a combination of the two.

Aggregated Relationship attributes are specified during the type definition for a composite object. Reverse-aggregated Relationship attributes are specified during the type definition of a component object.

How the Default Collector Determines whether a Component is Shared or Nonshared

In the list of components that the Default Collector returns to a handler, a component is marked as *shared* or *nonshared*. To determine whether a component is shared or nonshared, the Default Collector checks whether the component is associated with any objects other than the one on which the collection is being performed.

- If the component is only associated with the object on which the collection is being performed, the Default Collector marks it as a nonshared object.
- If the component is associated with other objects in addition to the one on which the collection is being performed, the Default Collector marks it as a shared object.

How the Default Collector Locates the Required Objects that are Associated with a Composite Object

In addition to components, the Default Collector also locates the required objects that are associated with a given object. It does this based on presence of required-object Relationship attributes in the composite object.

Like aggregated and reverse-aggregated Relationship attributes, a required-object Relationship attribute identifies objects that are to be collected. When a composite object contains a required-object Relationship attribute, the Default Collector collects the objects that the attribute references and marks them as "required objects" in the list that it returns to the handler.

In addition to those required objects that the Default Collector locates based on the required-object Relationship attributes that it finds in a composite object, the collector also returns the following items as required objects:

- The Type object associated with the object on which the collection is being performed

- The repository items (i.e., supporting documents and other attached files) associated with the object on which the collection is being performed

Working with the Default Collector

The Default Collector policy is used by many of the predefined asset types installed with CentraSite. Do not edit, delete or deactivate this policy.

⚠ **Important:** The Default Collector is triggered by an OnCollect event. The sole purpose of this event is to trigger the collector policy for a given object type. Do not attempt to use the OnCollect event to create additional policies that execute before or after the Default Collector or any other collector policy. Doing this can cause handlers to fail. Only one policy should execute when an OnCollect event occurs, and that policy should be the collection policy for the object type on which the OnCollect event occurs.

The Default Delete Handler

The Default Delete Handler policy deletes 1) the object on which the delete operation was requested and 2) all of that object's nonshared components (as identified by the list that the collector returns to the handler).

The Default Delete Handler policy is used by many of the predefined asset types installed with CentraSite. CentraSite also uses this handler to delete policies and lifecycle models. Do not edit, delete or deactivate this policy.

The Default Delete Handler is automatically assigned to new types that you add to CentraSite.

The Default Delete Handler is different than the other handler policies in that it executes on an event (the PreDelete event) that is also used to trigger user-defined policies. The Default Delete Handler has a priority of 1, which ensures that it executes before any user-defined policies that are also scoped for the PreDelete event.

The Default Move Handler

The Default Move Handler policy moves 1) the object on which the move operation was requested and 2) all of that object's nonshared components (as identified by the list that the collector returns to the handler).

The Default Move Handler policy is used by many of the predefined asset types installed with CentraSite. Do not edit, delete or deactivate this policy.

The Default Move Handler is automatically assigned to new types that you add to CentraSite.

⚠ **Important:** The Default Move Handler is triggered by an OnMove event. The purpose of this event is to trigger the move handler policy for a given object type. Do not attempt to use the OnMove event to create additional policies that execute before or after the Default Move

Handler (or any other move handler policy). Doing this could cause the handler to fail. Only one policy should execute when an OnMove event occurs, and that policy should be the move handler policy for the object type on which the OnMove event occurs.

The Default Export Handler

The Default Export Handler policy generates an export archive file that contains 1) the object on which the export operation was requested and 2) all of that object's nonshared components, shared components and required objects (as identified by the list that the collector returns to the handler).

The Default Export Handler policy is used by many of the predefined asset types installed with CentraSite. CentraSite also uses this handler to export policies. Do not delete, deactivate or modify this policy.

The Default Export Handler is automatically assigned to new types that you add to CentraSite.

⚠ **Important:** The Default Export Handler is triggered by an OnExport event. The purpose of this event is to trigger the export handler policy for a given object type. Do not attempt to use the OnExport event to create additional policies that execute before or after the Default Export Handler (or before/after any other export handler policy). Doing this can cause the handler policy to fail. Only one policy should execute when an OnExport event occurs, and that policy should be the export handler policy for the object type on which the OnExport event occurs.