Using the Workflow Editor to Create Task Workflows

A workflow consists of a logical flow of activation commands, including complex rollback scenarios. This logic enables you to define relationships between tasks, including sequences, branches, failure procedures, and access to Cisco ANA commands as well as the information model. Workflows are created using the Workflow Editor. The Workflow Editor can interface with an external system such as an order management system, to create a full solution for service provisioning that is user customizable and user extendable.

The Workflow Editor provides a process management GUI that acts as a powerful visual design and execution tracing tool for defining and deploying activation workflows. A workflow consists of several tasks grouped together and arranged in a hierarchy. Workflow management is supported in runtime, and includes a runtime GUI control console. After creating your workflows, you can run them locally within the editor to validate them before executing them on network elements.

Note

The Cisco ANA online help, which is launched from the Cisco ANA main toolbar by choosing Help > Help Contents, explains the enhancements that were made to the Workflow Editor for Cisco ANA. The Cisco ANA Workflow Editor online help also provides some clarifications of the vendor-provided functions. However, the vendor online help describes some features that are no longer part of the Workflow Editor that is packaged with Cisco ANA, so you should ignore those descriptions in the vendor online help.

The workflow engine is a processing module that provides services for executing workflows and for storing workflow templates (definitions). The engine resides on the gateway, using AVM 66. After a workflow is deployed (a template is created and stored), it is accessible for viewing properties and status. All deployed workflows are stored on the gateway server. The workflow engine provides default workflow inherent rollback. In addition, you can view a history of the invoked workflows using the Troubleshooting perspective.

See these topics for more specific information on workflows:

- Workflow Task Types and Activation States, page C-2
- Workflow Overview: Step-by-Step, page C-3
- Working with the Workflow Editor, page C-6
- Extending the Workflow Engine with Custom Tasks and Workflow Editor Callbacks, page C-24
Workflow Task Types and Activation States

Tasks are added to workflows to define the process. Each task performs a specific function and can be quickly added and configured using the Workflow Editor. Tasks can be classed as **predecessor** or **successor** tasks. As the names imply, a predecessor task must be completed before the next task can be executed, and a successor task is executed after a predecessor task. Each task has an associated **activation state**: 

- **Ready**—The task is ready to begin when the constraints (for example, start time or predecessors) have been satisfied.
- **Active**—The task is being executed.
- **Done**—The task is complete.
- **Abort**—The task has failed or the state has been set manually. The task can be manually reset to Ready or Done.
- **Passive**—The task exists, but is no longer relevant. For the purposes of successive tasks, the passive task is considered done.

The workflow below shows a typical task sequence:

**Figure C-1 Typical Task Sequence**

### Activation Script

The Activation Script task is located in the task toolbar of the Workflow Editor. This task executes the activation script specified in the Script Parameters tab of the Task Properties dialog box, and stores the result in a Task attribute called Result so that it can be used by tasks that follow it.

The Script Parameters tab can reference workflow attributes and task attributes. At runtime, the attributes’ values are passed into the script’s task to be used for script execution.

### Lock/Unlock Task

The Lock task has two main objectives, namely:

- To allow workflow instances to declare the resources that they use and the scope of their usage.
- To ensure that those resources are not used by any other workflow instance during that scope.

The Lock task allows you to protect any component from concurrent use by multiple workflows. You can lock an object that represents a single resource and guard the access to it. A resource’s identifier serves as the name of the lock. At any given time, a lock can only be owned by one workflow. Resources can be automatically locked during rollback.

The system prevents deadlocks before they occur. Upon detecting an imminent deadlock, the lock operation belonging to the workflow with the least progress fails. A failed lock may or may not abort the workflow.

The locking mechanism does not cover every access to every resource. Only workflows participate in the locking process. Non-workflow activities may access a resource even when it is locked by a workflow. Participation in the locking process is optional.
You can:

- Lock or unlock single or multiple resources.
- Unlock resources when a workflow terminates.
- Lock resources during rollback.

**Workflow Call (Synchronous Workflow Nesting)**

Synchronous workflow nesting allows workflow designers to invoke subworkflows synchronously, and pass arguments to each sub-workflow invocation. This enables the workflow designer to isolate the tasks executing in the sub-workflow as much as possible from the tasks executing in the parent workflow and in other sub-workflows.

The following functionality is available:

- The child workflow is executed in a separate workflow. The parent workflow waits for the child workflow to terminate.
- When a child workflow is aborted, it causes its parent workflow to also abort.
- The child workflow has a separate scope for attributes.
- The output of the child workflow is directed to the parent workflow.
- The parent workflow can pass parameters to its child workflow.
- The correct rollback sequence is maintained throughout the depth of the lineage.
- The child workflows are not visible through the API. The user interacts directly with the parent.
- Delete and abort operations on parent workflows are delegated to child workflows.

*Note*  
The maximum workflow nesting depth is 16.

**Workflow Overview: Step-by-Step**

These steps describe how to create and manage your workflows.

<table>
<thead>
<tr>
<th>Table C-1</th>
<th>Create and Manage Workflows</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td>Create a command using Command Builder and preview it.</td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td>Define tasks and workflows:</td>
</tr>
<tr>
<td>a.</td>
<td>Copy the Command Builder scripts to the command template tabs for the Workflow Editor Activation Scripts</td>
</tr>
<tr>
<td>b.</td>
<td>Edit the Command Builder scripts, if required.</td>
</tr>
<tr>
<td>c.</td>
<td>Add workflow and task attributes.</td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td>Test the workflow locally.</td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td>Deploy the workflow on the gateway.</td>
</tr>
</tbody>
</table>
Understanding the Workflow Editor User Interface

Figure C-2 shows the Workflow Editor window with a template. You can open the Workflow Editor by clicking the Workflow icon in the Inventory perspective main toolbar:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Tooltip</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>Workflow</td>
<td>Opens the Workflow Editor</td>
</tr>
</tbody>
</table>

Note: The Workflow Editor window appears empty when it is opened. You can create a new workflow or retrieve a workflow that was created previously.

Table C-1: Create and Manage Workflows (continued)

<table>
<thead>
<tr>
<th>Step</th>
<th>To Perform this Task...</th>
<th>See...</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Run the workflow on the gateway.</td>
<td>Running a Workflow on the Gateway, page C-24</td>
</tr>
<tr>
<td>6</td>
<td>View the results of the workflow.</td>
<td>Running a Workflow on the Gateway, page C-24</td>
</tr>
</tbody>
</table>

Figure C-2: Workflow Editor Window

| 1 | Drawing Area | 4 | Tree |
| 2 | Task Toolbar | 5 | Action Toolbar |
| 3 | Task Attribute Table | | |
The Workflow Editor window contains these actions which have been customized and added to the action toolbar:

<table>
<thead>
<tr>
<th>Icon</th>
<th>Tooltip</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Retrieve Workflow from Server</td>
<td>Retrieves and loads a workflow from the gateway server</td>
</tr>
<tr>
<td></td>
<td>Delete Workflow from Server</td>
<td>Deletes a workflow from the gateway server</td>
</tr>
<tr>
<td></td>
<td>Deploy Workflow</td>
<td>Uploads a workflow to the gateway server</td>
</tr>
<tr>
<td></td>
<td>Help</td>
<td>Launches vendor online help</td>
</tr>
</tbody>
</table>

**Note** The Cisco ANA online help, which is launched from the Cisco ANA main toolbar by choosing Help > Help Contents, explains the enhancements that were made to the Workflow Editor for Cisco ANA. The Cisco ANA Workflow Editor online help also provides some clarifications of the vendor-provided functions. However, the vendor online help describes some features that are no longer part of the Workflow Editor that is packaged with Cisco ANA, so you should ignore those descriptions in the vendor online help.

In addition, the following customized tasks have been added to the task toolbar:

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Obtains a lock on a resource</td>
</tr>
<tr>
<td></td>
<td>Unlocks a resource</td>
</tr>
<tr>
<td></td>
<td>Synchronically executes or calls another workflow</td>
</tr>
<tr>
<td></td>
<td>Embeds and executes another workflow</td>
</tr>
</tbody>
</table>

**Roles Required to Use the Workflow Editor**

Table C-2 lists the roles that are required to use the Workflow Editor. For more information on roles, see Creating and Managing Users, Passwords, and Scopes, page 13-34.

<table>
<thead>
<tr>
<th>Task</th>
<th>Role Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating a workflow</td>
<td>Viewer, Network Operator, Configurator, Administrator</td>
</tr>
<tr>
<td>Opening a workflow</td>
<td>Viewer, Network Operator, Configurator, Administrator</td>
</tr>
<tr>
<td>Closing a workflow</td>
<td>Viewer, Network Operator, Configurator, Administrator</td>
</tr>
<tr>
<td>Saving as/renaming a workflow</td>
<td>Viewer, Network Operator, Configurator, Administrator</td>
</tr>
</tbody>
</table>
Working with the Workflow Editor

These topics provide instructions for launching the Workflow Editor feature. The Workflow Editor window contains these actions which have been customized and added to the action toolbar.

- Creating a New Workflow Template, page C-6
- Retrieving a Workflow Template, page C-7
- Deploying a Workflow Template, page C-7
- Deleting a Workflow Template, page C-8
- Viewing Workflow Properties, page C-8
- Working with the Task Library, page C-10
- Executing (Testing) a Workflow Locally, page C-23
- Running a Workflow on the Gateway, page C-24

Creating a New Workflow Template

The workflow template creation process begins when you create a new workflow template with a unique name. A blank template is created, and opened for editing. You then add tasks to the template to create the logical flow that is required. The workflow template is stored locally, and each update you make is automatically saved.

Reviewers: The following is from the ANA 3.6.2 Workflow doc. (I do said it applies to 4.1 also.)

When you create a unique name for each workflow template, do not include the following wildcard characters:

- "_" denotes any single character
- "%" denotes a zero or many characters

If the "_" and "%" characters are included in the workflow template, you will see the following message in the AVM66 log when you try to run the template or reference it in a subflow:

"WARN [13 21:00:08,248] - dralasoft.workflow - Task aborted. Task: 245886, Workflow: 245885 java.lang.IllegalArgumentException: Template AA_BB.template is ambiguous, templates ids are: 245874, 245873"
The following examples show workflow template names that can lead to ambiguity if they are deployed together:

- WFTLM_MUESTRA.template and WFTLM#MUESTRA.template
- WFTLM%MUESTRA.template and WFTLM###MUESTRA.template

The ambiguity only occurs when the template containing the wildcard characters runs.

For information about creating new workflows, see the Workflow Editor online help by clicking the Help icon in the Workflow Editor action toolbar (see Understanding the Workflow Editor User Interface, page C-4).

**Retrieving a Workflow Template**

You can retrieve a workflow template that was previously created and deployed on the gateway or server. Once the workflow template has been retrieved you can do one of the following:

- Edit the workflow template
- Deploy the workflow template
- Execute the workflow template
- Delete the workflow template

To retrieve a workflow template:

**Step 1**
In the toolbar, click the Retrieve Workflow from Server icon. The Retrieve Workflow Template from Server dialog box is displayed.

The Retrieve Workflow Template from Server dialog box displays the list of existing workflow templates.

**Step 2**
Select the required workflow template that you want to load from the list.

**Step 3**
Click **OK**. A confirmation message is displayed.

**Step 4**
Click **OK**. The required workflow template is opened and displayed in the Workflow Editor window.

**Deploying a Workflow Template**

After you have tested the workflow template locally, you can deploy the workflow template to the gateway, where it is available to all authorized users.

To deploy a workflow template:

**Step 1**
In the toolbar, click the Deploy Workflow icon. A confirmation message is displayed.

**Step 2**
Click **Yes**. A success message is displayed.

**Note**
If the workflow template already exists the server asks you if you want to replace the existing workflow.
Deleting a Workflow Template

You can delete a workflow template from the server.

**Note**
A workflow template does not have to be open or displayed in the Workflow Editor window before it can be deleted.

To delete a workflow template:

**Step 1**
In the toolbar, click the Delete Workflow from Server icon. The Delete Workflow Template from Server dialog box is displayed.

**Step 2**
Select the template that you want to delete from the list.

**Step 3**
Click OK. A success message is displayed.

**Step 4**
Click OK. The workflow template is deleted from the local server.

Viewing Workflow Properties

The Workflow Properties dialog box lets you view the workflow callback scripts. You can view the properties of an Activation Script workflow.

To view workflow properties:

**Step 1**
Create or retrieve the required workflow template.

**Step 2**
Right-click anywhere in the drawing area, and choose Workflow Properties from the popup menu. The Workflow Properties dialog box for the required workflow template is displayed.
For information about the Workflow Properties dialog box, see the Workflow Editor online help by clicking the Help icon in the action toolbar.

**Step 3**  
Click the **Attributes** tab. The attributes of the Activation Script workflow are displayed.

![Workflow Properties Dialog Box - General Properties Tab](image1)

![Attributes Tab](image2)
Step 4  Click the **Callback Scripts** tab.

*Figure C-5  Callback Scripts Tab*

The Select Script drop-down list lets you choose one of the following options to execute (activate) the script:

- **preActiveScript**—Executes the script before the workflow is active.
- **activeScript**—Executes the script when the workflow becomes active.
- **doneScript**—Executes the script when the workflow is successfully completed.
- **exceptionScript**—Executes the script if one of the tasks in the workflow fails.

The Execute button runs the script for testing purposes.

Step 5  Click **OK**. The Workflow Properties dialog box is closed.

## Working with the Task Library

This section describes viewing the properties of the following tasks:

- **Activation Script Task**, page C-11
- **Workflow Call Task**, page C-14
- **Subflow Task**, page C-16
- **Lock Task**, page C-18
- **Unlock Task**, page C-20
- **Notes on Delay and Escalate Tasks**, page C-22
Activation Script Task

The Activation Script task is used to invoke commands previously created and residing on the gateway. This is the main task used for implementing an activation workflow.

The command template can include the following parameters:

- `$Attribute name$` is evaluated as a workflow attribute.
- `$Task name:Attribute name$` is evaluated as a task attribute.

To view the properties of an Activation Script task:

**Step 1** Create or retrieve the required workflow template.

**Step 2** Right-click the required Activation Script in the drawing area, and choose **Task Properties** from the popup menu. The Task Properties dialog box for the required task is displayed.

---

**Note**

The Earliest Start option is not supported.

For information about the Task Properties dialog box, see the Workflow Editor online help by clicking the Help icon in the action toolbar.
Step 3  Click the Attributes tab. The attributes of the task are displayed.

Figure C-7  Activation Script Properties Dialog Box - Attributes Tab

Step 4  Click the Script Parameters tab.
The Script Parameters tab lets you specify the following:

- **Script Name**—The name of the Activation Script that resides on the gateway server.
- **Target OID**—The OID of the VNE on which to run the script. Supported values are fixed strings, attribute references (as is shown in Figure C-8), or a template with embedded attribute references; for example, `{[ManagedElement(Key=$devicename$)]}`.
- **Arguments**—The arguments passed to the script according to the script signature. The name should be the same as the name of the script parameter as defined in Command Builder. Supported values are fixed strings, attribute references (as is shown in Figure C-8), or a template with embedded attribute references; for example, `{[ManagedElement(Key=$devicename$)]}`.

**Step 5** Click **OK**. The Task Properties dialog box is closed.
Workflow Call Task

The Workflow Call task is used to synchronically execute/call another workflow, which is recognized as a child of this workflow.

**Note**

The maximum workflow nesting depth is 16.

To view the properties of a Workflow Call task:

1. Create or retrieve the required workflow template.
2. Right-click the required Workflow Call task in the drawing area, and choose Task Properties from the popup menu. The Task Properties dialog box for the required task is displayed.

The Assign Attributes tab lets you pass parameters as attributes to child workflows. An attribute can be assigned to a child workflow in two ways:

- By specifying a workflow attribute in a parent workflow, as follows:
  
  `attribute_in_child_workflow=attribute_in_parent_workflow`

- By specifying a task in a parent workflow followed by a colon (`:`) and the attribute name, as follows:
  
  `attribute_in_child_workflow=task_in_parent_workflow:attribute_name`

For information about the Task Properties dialog box, see the Workflow Editor online help by clicking the Help icon in the action toolbar.
**Step 3**  
Click the **General Properties** tab. The general properties of the task are displayed.

![Workflow Call Task Properties Dialog Box - General Properties Tab](image)

**Note**  
The Earliest Start option is not supported.

**Step 4**  
Click the **Attributes** tab. The attributes of the task are displayed.

**Step 5**  
Click the **Target Template** tab.
Figure C-11  Workflow Call Task Properties Dialog Box - Target Template Tab

The Target Template tab lets you choose the workflow template that is defined as the child workflow.

**Step 6**
Click OK. The Task Properties dialog box is closed.

### Subflow Task

The Subflow task is used to embed and execute another workflow in this workflow instance.

**Note**
The maximum workflow nesting depth is defined in the registry. The default value is 16.

To view the properties of a Subflow task:

**Step 1**
Create or retrieve the required workflow template.

**Step 2**
Right-click the required Subflow task in the drawing area, and choose Task Properties from the popup menu. The Task Properties dialog box for the required task is displayed.
Note  The “Earliest Start” option is not supported.

For information about the Task Properties dialog box, see the Workflow Editor online help by clicking the Help icon in the action toolbar.

Step 3  Click the Attributes tab. The attributes of the task are displayed.

Step 4  Click the Target Template tab.
The Target Template tab lets you select the workflow template that is embedded.

**Step 5**
Click **OK**. The Task Properties dialog box is closed.

---

**Lock Task**

The Lock task is used to create a lock on specific resources.

To view the properties of a Lock task:

**Step 1**
Create or retrieve the required workflow template.

**Step 2**
Right-click the required Lock task in the drawing area, and choose **Task Properties** from the popup menu. The Task Properties dialog box for the required task is displayed.
Note  The Earliest Start option is not supported.

For information, see the Workflow Editor online help by clicking the Help icon in the action toolbar.

Step 3  Click the Attributes tab. The attributes of the task are displayed.

Step 4  Click the Resource Names tab.
The Resource Names tab lets you specify the resources that should be locked.

**Note**  
It is possible to lock multiple resources as well.

**Reviewers: The following info was added from the 3.6.2 guide.**

In the text area, enter the resource names to be locked. Each resource name should be on a separate line. Resource names can be passed as parameters that are resolved during task execution to either workflow or task attributes:

- `$Attribute name$` is evaluated from a workflow attribute by that name.
- `$Task name:Attribute name$` is evaluated by a task attribute by that name.

The result is stored in an attribute named Result (in the Lock task) and is one of the following:

- **Success** - lock successfully obtained.
- **Failed due to timeout** - lock failed due to timeout.
- **Failed due to deadlock** - lock failed due to a deadlock.

**Step 5**  
Click OK. The Task Properties dialog box is closed.

**Unlock Task**

The Unlock task is used to unlock specific locked resources.

To view the properties of an Unlock task:

**Step 1**  
Create or retrieve the required workflow template.

**Step 2**  
Right-click the required Unlock task in the drawing area, and choose **Task Properties** from the popup menu. The Task Properties dialog box for the required task is displayed.
Figure C-16  Unlock Task Properties Dialog Box - General Properties Tab

Note  The “Earliest Start” option is not supported.

For information about the Task Properties dialog box, see the Workflow Editor online help by clicking the Help icon in the action toolbar.

Step 3 Click the Attributes tab. The attributes of the task are displayed.
Step 4 Click the Resource Names tab.
The Resource Names tab lets you specify the resources that should be unlocked.

**Reviewers: The following text was added from the 3.6.2 guide.**

In the text area, enter the resource names to be unlocked. Each resource name should be on a separate line.

Resource names can be passed as parameters that are resolved during task execution to either workflow or task attributes:

- `$Attribute name$` is evaluated from a workflow attribute by that name.
- `$Task name:Attribute name$` is evaluated by a task attribute by that name.

**Step 5**  Click **OK**. The Task Properties dialog box is closed.

---

**Notes on Delay and Escalate Tasks**

The Delay and Activation Scripts both allow you to configure a delay property, which specifies the following:

- For Delay tasks, it specifies a task’s sleep time.
- For Escalate tasks, it specifies the amount of time to wait before escalating (starting another task in the workflow).

However, how this affects the workflow depends on whether the workflow engine is configured to use absolute time or a *business calendar*. If the `useCalendar` property is set to true, the workflow uses the business calendar, which is configured using the `CALENDAR.DWE` system property, defined in the engine XML file. See the vendor online help for more information (see *Understanding the Workflow Editor User Interface*, page C-4).
Executing (Testing) a Workflow Locally

Reviewers: Can users test activation scripts locally?

You can execute (test) an Activation Script workflow locally on the client. For more information about executing a workflow, see the Workflow Editor online help by clicking the Help icon in the action toolbar. This section describes executing (testing) a workflow that includes a Command Builder task.

To execute a workflow:

Step 1 Create a new workflow in the Workflow Editor window (see Creating a New Workflow Template, page C-6) or retrieve a workflow (see Retrieving a Workflow Template, page C-7).

Step 2 In the toolbar, click the Execute Workflow icon. The Workflow Editor window changes and displays a new tab near the top of the window, reflecting the activation status (see Workflow Task Types and Activation States, page C-2). For example, the new tab might say READY.

In addition, the following views are displayed in the window (lower row) and relate to the state of your workflow test on your local machine:

- Graphical View—Displays the workflow as a graphical presentation.
- Tabular View—Displays a table of all of the tasks in the workflow.
- Workflow State Analysis—Displays an analysis of the activation status of the workflow.
- XML View—Displays the XML code of the current workflow.

In addition, the following buttons are displayed at the bottom of the window:

- Activate—Activates the workflow.

Note This button toggles to Abort when Activate is clicked. Clicking Abort aborts the workflow.

- Copy—Copies the workflow.
- Delete—Deletes an executing workflow.

Step 3 Click Activate. The Console window is displayed.

The Console window displays the results of each task in the workflow.
Extending the Workflow Engine with Custom Tasks and Workflow Editor Callbacks

Step 4 Click to close the Console window. The Workflow Editor window is displayed.

The tasks displayed in the Graphical View tab (lower row) display the activation status of each task using an oval shape and various colors (in the bottom right corner), as shown in the example.

*Figure C-19 Tasks Displayed In the Graphical View Tab*

These colors change as the status of the task changes during the execution process, as follows:

- Blue—Ready
- Green—Active
- Red—Aborted
- Grey—Done
- Light Gray—Inactive

For more information about activation status, see page 2.

Step 5 Review the results of the execution of the workflow in the Workflow Editor window.

Once the workflow has been executed (tested) successfully locally on the client, you can deploy the workflow to the gateway. For more information, see Deploying a Workflow Template, page C-7.

Running a Workflow on the Gateway

You can use standardized Cisco ANA APIs to run workflows on the gateway, and to receive status about those workflows. For information on how to do this, see Cisco Active Network Abstraction Information Model Primer.

Extending the Workflow Engine with Custom Tasks and Workflow Editor Callbacks

You can develop and install custom tasks for your special project needs. For more information, see these sections of the vendor online help:

- Customization > Task and Workflow Property Dialogs
- Engine API > Customizing Engine Behavior

To launch the vendor online help, click the question mark icon, as described in Understanding the Workflow Editor User Interface, page C-4.