# Cisco Active Network Abstraction Workflow User Guide Version 3.6



**Americas Headquarters**
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

Text Part Number: OL-12508-01

*Cisco Active Network Abstraction Workflow User Guide Version 3.6*

# CONTENTS

# Preface

This guide describes how to create, run, manage and view workflows using the Cisco ANA Workflow Editor, Cisco ANA Manage and Cisco ANA EventVision. It is intended for use by a system administrator.

The Dralasoft Workflow Studio can be customized to meet an individual customer's specific needs. Tthe Dralasoft Workflow Studio menus and toolbars are customized for use with Cisco ANA. For a detailed explanation of the customized functionality, see Cisco ANA Workflow Editor Window, page 3.

It includes the following chapters:

- Chapter 1, “Introducing the Cisco ANA Workflow Editor”—Describes the Workflow Editor, and provides a description of tasks that are used to create workflow templates.
- Chapter 2, “Working With the Cisco ANA Workflow Editor”—Provides instructions for launching the Workflow Editor. In addition, it describes the Workflow Editor working environment, and how to operate the customized functionality.
- Chapter 3, “Getting Started”—Provides the steps required when creating, running, deploying, running and viewing a workflow.
- Chapter 4, “Managing Workflows”—Provides instructions for launching the Cisco ANA Manage application. In addition, it describes the lists of all the deployed workflow templates for the Cisco ANA gateway and the actions that can be performed.
- Chapter 5, “Customizing the Workflow Editor”—Describes extending the Workflow Engine with custom tasks and Workflow Editor callbacks.

**Note** Changes to the registry should only be carried out with the support of Cisco Professional Services.

# Obtaining Documentation, Obtaining Support, and Security Guidelines

For information on obtaining documentation, obtaining support, providing documentation feedback, security guidelines, and also recommended aliases and general Cisco documents, see the monthly *What’s New* in Cisco Product Documentation, which also lists all new and revised Cisco technical documentation, at:

http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html

CHAPTER 1

# Introducing the Cisco ANA Workflow Editor

This chapter describes the Cisco ANA Workflow Editor. In addition, it provides a description of tasks that are used to create workflow templates.

- About the Cisco ANA Workflow Editor—Provides an overview of the Workflow Editor.
- What Is a Workflow Task?—Describes tasks and their activation status.
- What Is a BQL Task?—Describes the customized BQL task.
- What Is a Lock/Unlock Task?—Describes the locking resources mechanism.
- Workflow Call (Synchronous Workflow Nesting)—Describes synchronous workflow nesting.

**Note** Changes to the registry should only be carried out with the support of Cisco Professional Services.

## About the Cisco ANA Workflow Editor

The Cisco ANA Workflow Editor is used to create and run the logical flows of activation commands, including complex rollback scenarios. This logic enables the user to define relationships between tasks, including sequences, branches, failure procedures, and access to Cisco ANA commands as well as the information model. The Workflow Editor can interface with an external system such as an order management system in order to create a full solution for service provisioning, which is user-customizable and user-extendable.

The Workflow Editor is 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.

The Workflow Editor is used to construct workflows which run gateway commands and provides complete access to the Cisco ANA live network information model. The Workflow Editor provides a nested structure. Workflow commands are also available through the Cisco ANA API.

The workflow engine resides on the gateway using AVM 66. All the deployed workflows are stored on the gateway. After a workflow is deployed, it is accessible through Cisco ANA Manage for viewing properties and status. Deployed workflows can be invoked through the Cisco ANA API using BQL. The workflow engine provides default workflow inherent rollback.

In addition, the user can view a history of the invoked workflows using Cisco ANA EventVision.

# What Is a Workflow Task?

Tasks are added to workflows to define the processes. Each task performs a specific function. Tasks can be quickly added and configured using the Workflow Editor.

Tasks can be classed as predecessor or successor tasks:

- A predecessor task is one that must be completed before the next task can be executed.
- A successor task is one that is run after a predecessor task.

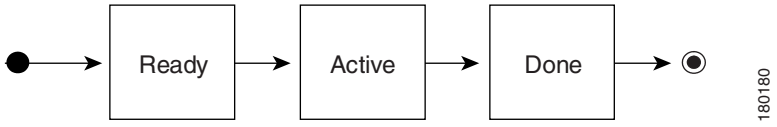
For example, if there is a workflow that consists of two tasks, Configure Switch (first task) and Configure Router (second task) then the Configure Switch task is the predecessor task the must be completed before the Configure Router (successor task) can be run.

There is an activation state associated with each task, which can be one of the following:

- **Ready—**The task is ready to begin when the constraints (for example, start time or predecessors) have been satisfied.
- **Active—**The task is being run.
- **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 indicates the typical task sequence:

***Figure 1-1 Typical Task Sequence***




# What Is a BQL Task?

The Execute BQL task is found in the Workflow Editor toolbar. This task runs the BQL command specified in the Command Template tab of the Task Properties dialog box, and stores the result in a task attribute called Result so that it can be used by scripts and other tasks.

The Command Template tab can reference workflow attributes and task attributes. At runtime, the attribute's values are substituted into the template before it is run.

# What Is a Lock/Unlock Task?

The Lock task has two main objectives:

- 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 enables the user to protect any component from concurrent use by multiple workflows. The user 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 at most 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 will fail. 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.

The user 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 sub-workflows synchronously and pass arguments to each subworkflow invocation. This enables the workflow designer to isolate the tasks running in the subworkflow as much as possible from the tasks running in the parent workflow and in other subworkflows.

The following functionality is available:

- The child workflow is run in a separate workflow. The parent workflow waits for the child workflow to terminate.
- When a child workflow is stopped it causes its parent workflow to also stop.
- 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 defined in the registry. The default value is 16.

CHAPTER 2

# Working With the Cisco ANA Workflow Editor

This chapter provides instructions for launching the Cisco ANA Workflow Editor application. In addition, it describes the Cisco ANA Workflow Editor working environment and how to operate the customized functionality.

- Starting Cisco ANA Workflow Editor—Describes how to login to the Workflow Editor.
- Cisco ANA Workflow Editor Window—Briefly describes the Workflow Editor window and the additional customized functions.
- Creating a New Workflow Template—Provides a brief overview of the workflow template creation process.
- Retrieving a Workflow Template—Describes how to retrieve a workflow template that was previously created from the local server.
- Deploying a Workflow Template—Describes how to deploy a workflow template to the server.
- Deleting a Workflow Template—Describes how to delete a workflow template from the local server.
- Viewing Workflow Properties—Describes how to view the properties of a BQL workflow.
- Working With the Task Library—Describes how to view the properties of a BQL, Workflow Call, Subflow, Lock and Unlock tasks.
- Testing a Workflow—Describes how to execute a workflow template on the local server.
- Gateway Workflow Commands and Operations—Describes how to add utility functions using BQL commands.
- Logging Out—Describes how to logout of the Workflow Editor.

# Starting Cisco ANA Workflow Editor

This section provides instructions for launching the Workflow Editor. The Workflow Editor is password protected to ensure security. Before you start using the Workflow Editor make sure you know your username, password, and the gateway IP address or hostname. Installation of the Workflow Editor is optional. The Workflow Editor can be installed using the regular client installation package. For more information, refer to *Cisco Active Network Abstraction Client Installation Guide*. When the client opens it will negotiate with the server in order to validate the license. If the license is not validated the Workflow Editor will not open.

To start the Workflow Editor:

**Step 1** From the Start menu, choose the **Programs** folder, then **Cisco ANA > Workflow Editor**. The Workflow Editor Login dialog box is displayed.

**Step 2** Enter your username and password.

**Step 3** Enter the gateway information in the host field as an IP address or hostname,

or

Choose a gateway from the host dropdown list.

**Note** The gateway IP address or hostname that was used when you last logged in is automatically displayed at the top of the host list.

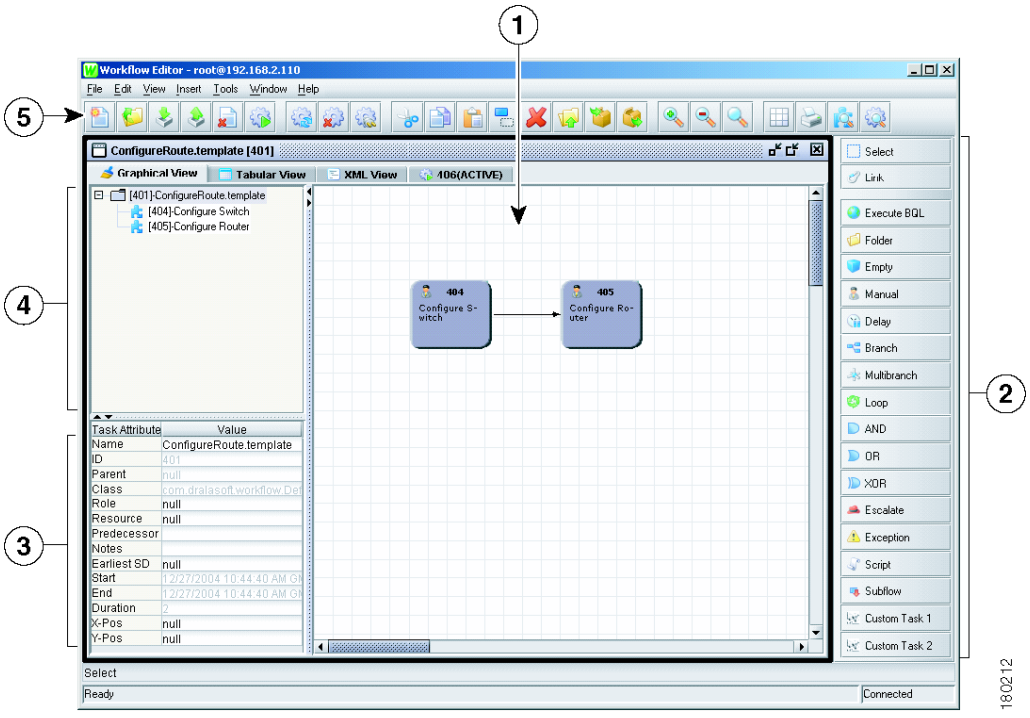
**Step 4** Click **OK**. The Workflow Editor window is displayed. For more information, see Figure 2-1

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

# Cisco ANA Workflow Editor Window

The Workflow Editor window with a template is displayed.

***Figure 2-1 Workflow Editor Window***

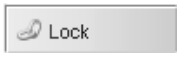
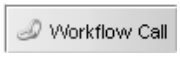



| | |
|---|---|
| 1 | Drawing area |
| 2 | Task toolbar |
| 3 | Task attribute table |
| 4 | Tree |
| 5 | Action toolbar |

The Workflow Editor window displays information on the workflow and the tasks within the workflow. The following actions are available from the Workflow Editor toolbar and from the Tools menu:

| Button | Function |
|---|---|
|  | Retrieves (loads) a workflow from the gateway. |
|  | Deletes a workflow from the gateway. |
|  | Deploys (uploads) the workflow on the gateway. |

In addition, the following tasks are available on the toolbar:

| | |
|---|---|
|   | The Execute BQL task is used to invoke commands previously created and residing on the server. This is the main task used for implementing an activation workflow. |
|   | The Lock task is used to obtain a lock on a resource. |
|   | The Unlock task is used to unlock a resource. |
|   | The Workflow Call task is used to synchronically execute or call another workflow. |
|   | The Subflow task is used to embed and execute another workflow. |

# Creating a New Workflow Template

The workflow template creation process begins by creating a new workflow template with a unique name. A blank template is created, and opened for editing. The user then adds tasks to the template in order to create the logical flow that is required. The workflow template is stored locally, and each update is automatically saved.

## Workflow Template Names

When you create the 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 is run.

# Retrieving a Workflow Template

The user can retrieve a workflow template that was previously created and deployed on the gateway or server. Once the workflow template has been retrieved, the user 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** On the toolbar, click **Retrieve Workflow**. The Retrieve Workflow template from the Server dialog box is displayed.

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

**Step 2** Choose the 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 the user has tested the workflow template locally and is satisfied, the workflow template can be deployed to the Cisco ANA server (remote server), where it will be available to all authorized users.

To deploy a workflow template:

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

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

**Note** If the workflow template already exists, the server will ask if you want to replace the existing workflow.

**Step 3** Click **OK**. The selected workflow template is deployed.

# Deleting a Workflow Template

The user 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** On the toolbar, click **Delete Workflow from Server**. The Delete Workflow Template from Server dialog box is displayed.

**Step 2** Choose 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

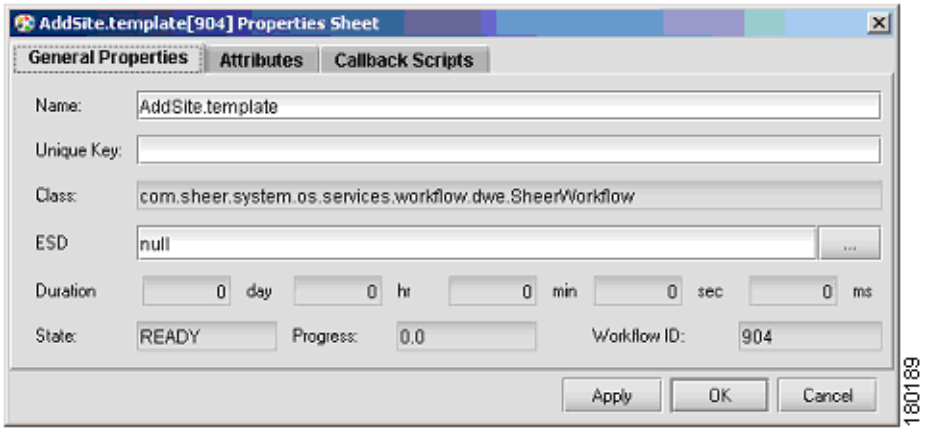
The Workflow Properties dialog box enables the user to view the workflow callback scripts. The user can view the properties of a BQL workflow.

To view workflow properties:

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

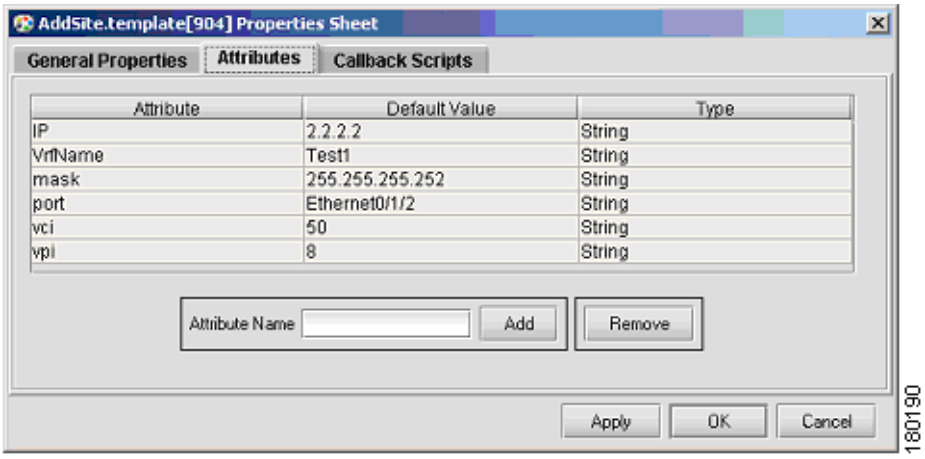
**Step 2** Right-click in the drawing area to display the menu, and choose **Workflow Properties**. The Workflow Properties dialog box for the required workflow template is displayed.

*Figure 2-2 Workflow Properties Dialog Box - General Properties Tab*



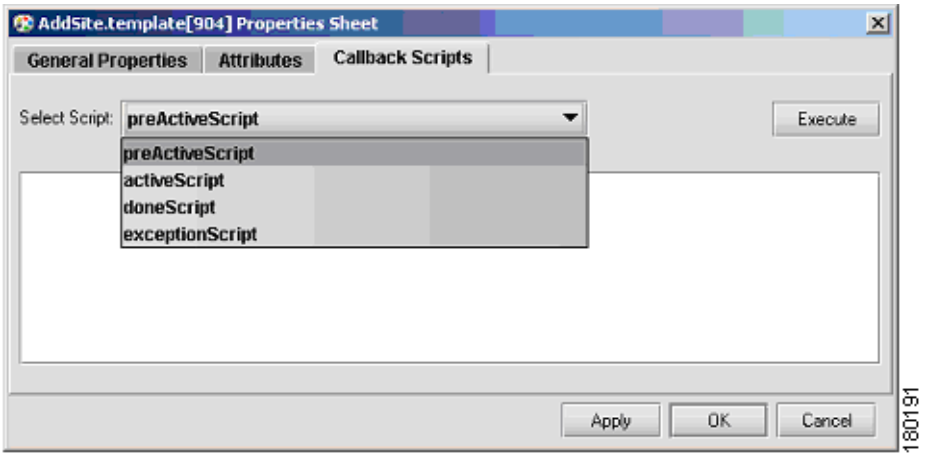
**Step 3** Choose the Attributes tab. The attributes of the BQL workflow are displayed.

***Figure 2-3 Attributes Tab***




**Step 4** Choose the Callback Scripts tab.

***Figure 2-4 Callback Scripts Tab***




The Select Script list enables the user to choose one of the following options in order to activate the script:

- **preActiveScript—**Activates the script before the workflow is active.
- **activeScript—**Activates the script when the workflow becomes active.
- **doneScript—**Activates the script when the workflow is successfully completed.
- **exceptionScript—**Activates 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:

- Execute BQL Task
- Workflow Call Task
- Subflow Task
- Lock Task
- Unlock Task

## Execute BQL Task

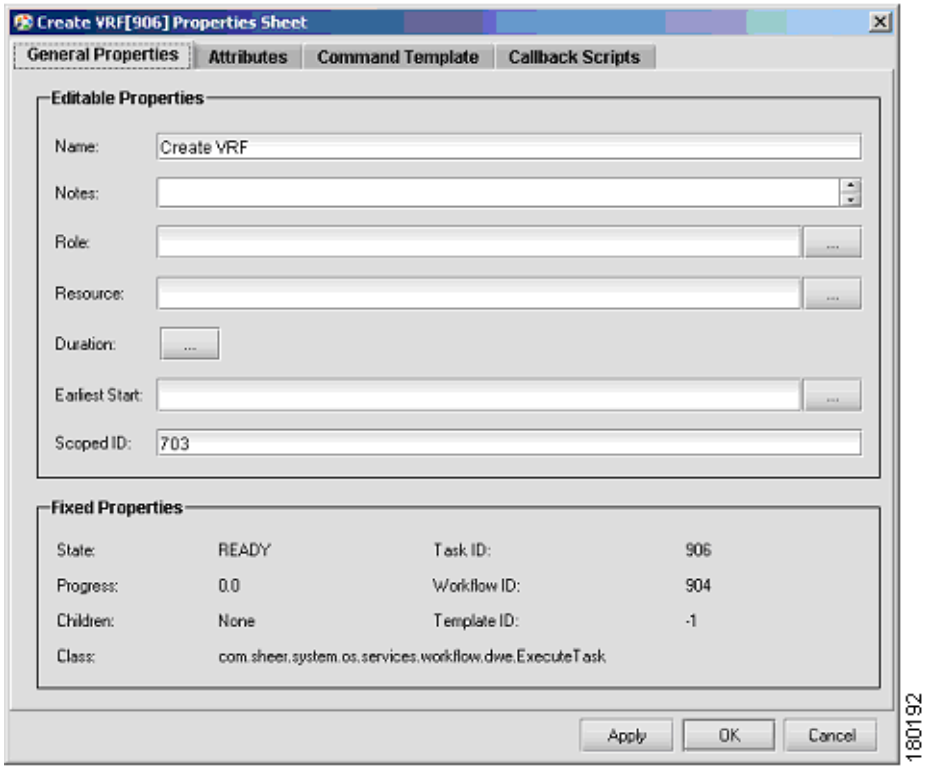
To view the properties of an Execute BQL task:

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

**Step 2** Choose the required BQL task in the drawing area.

**Step 3** Right-click to display the menu, and choose **Task Properties**. The Task Properties dialog box for the required task is displayed.

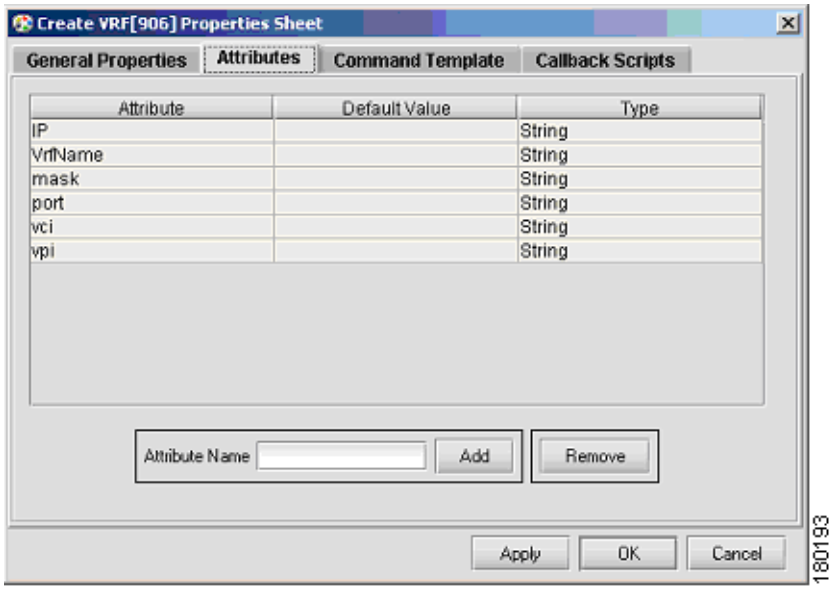
*Figure 2-5 Task Properties Dialog Box - General Properties Tab*




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

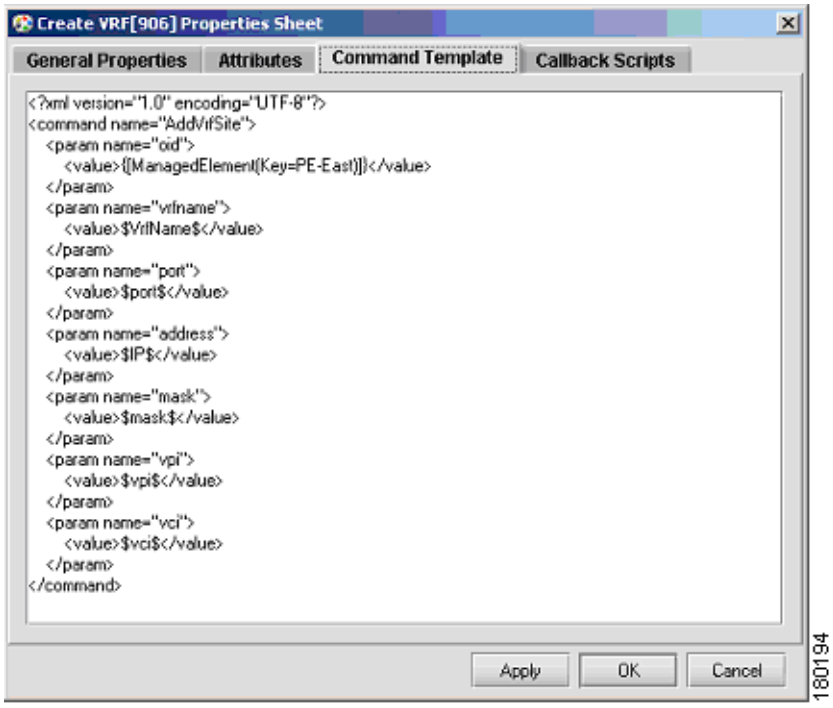
**Step 4** Choose the Attributes tab. The attributes of the task are displayed.

***Figure 2-6 Attributes Tab***




**Step 5** Choose the Command Template tab.

***Figure 2-7 Command Template Tab***




Use the Command Template tab to enter the command template XML to create a command instance descriptor (CID) used for running the gateway command or activation script.

The format of the template is similar to a CID, with the option for setting parameters that would be evaluated on execution:

- $Attribute name$—Will be evaluated by that name from a workflow attribute.
- $Task name:Attribute name$—Will be evaluated by that name by a task attribute.

If an invalid attribute or task name is entered, the evaluation of the command template will fail and stop the task. The following error message is displayed:

```
Failed to create command, task [Task] not performed, aborting task.
Reason: Attribute <Attribute name> evaluation failed, aborting command creation
```

**Note** Each BQL task includes a task attribute called RollbackEnabled. This attribute defines whether or not this BQL task command will be rolled back if the workflow stops. The default value is True.

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

# Workflow Call Task

The Workflow Call task is used to synchronically run or call another workflow, which will be recognized as a child of this workflow.

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

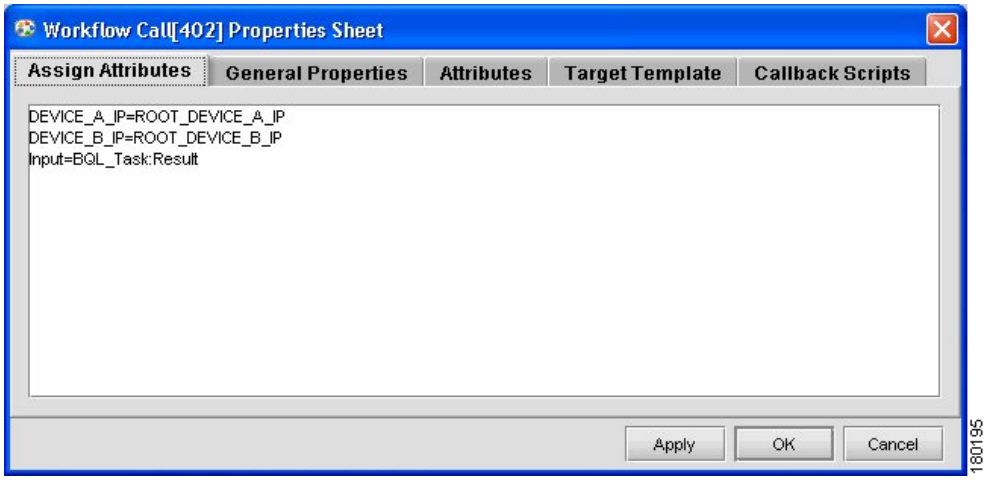
To view the properties of a Workflow Call task:

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

**Step 2** Choose the required Workflow Call task in the drawing area.

**Step 3** Right-click to display the menu, and choose **Task Properties**. The Task Properties dialog box for the required task is displayed.

*Figure 2-8 Task Properties Dialog Box - Assign Attributes Tab*



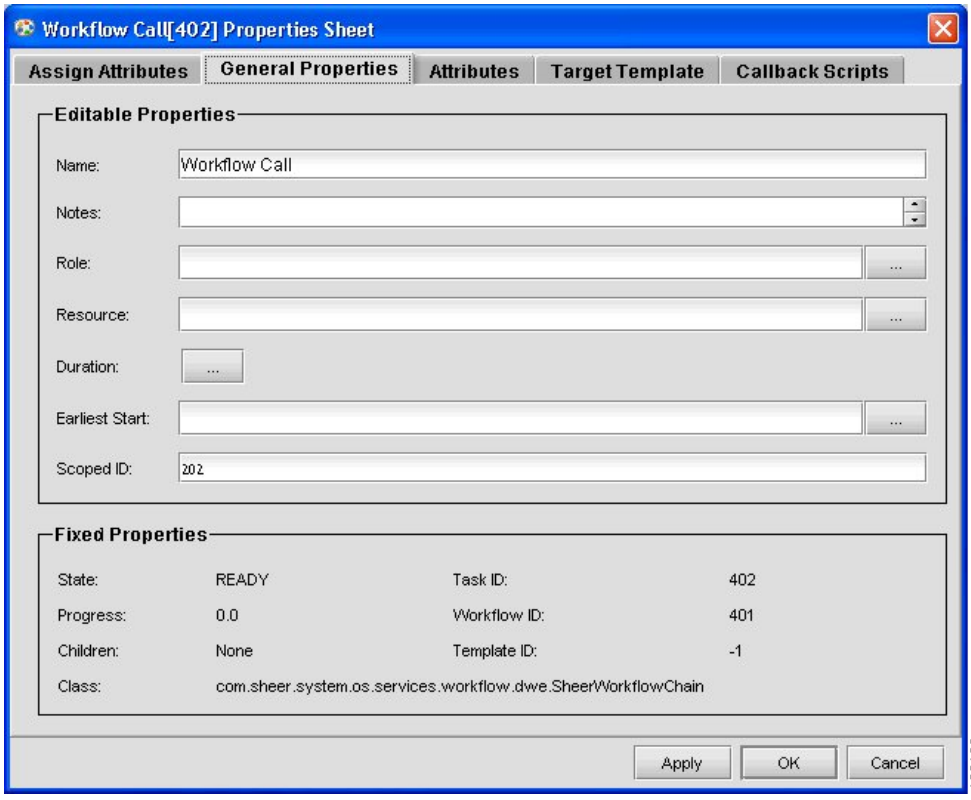
The Assign Attributes tab enables the user to pass parameters as attributes to child workflows.

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

**Step 4** Choose the General Properties tab. The general properties of the task are displayed.

***Figure 2-9 General Properties Tab***

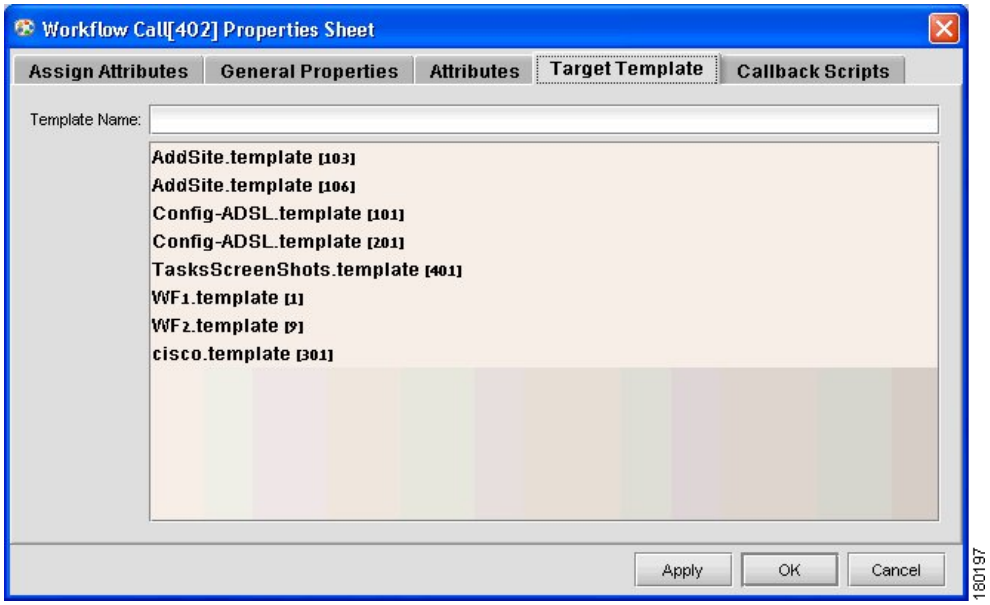



**Note** The Earliest Start option is not supported.

**Step 5** Choose the Attributes tab. The attributes of the task are displayed.

**Step 6** Choose the Target Template tab.

*Figure 2-10 Target Template Tab*




The Target Template tab enables the user to choose the workflow template that will be defined as the child workflow.

**Step 7** Click **OK**. The Task Properties dialog box is closed.

# Subflow Task

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

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

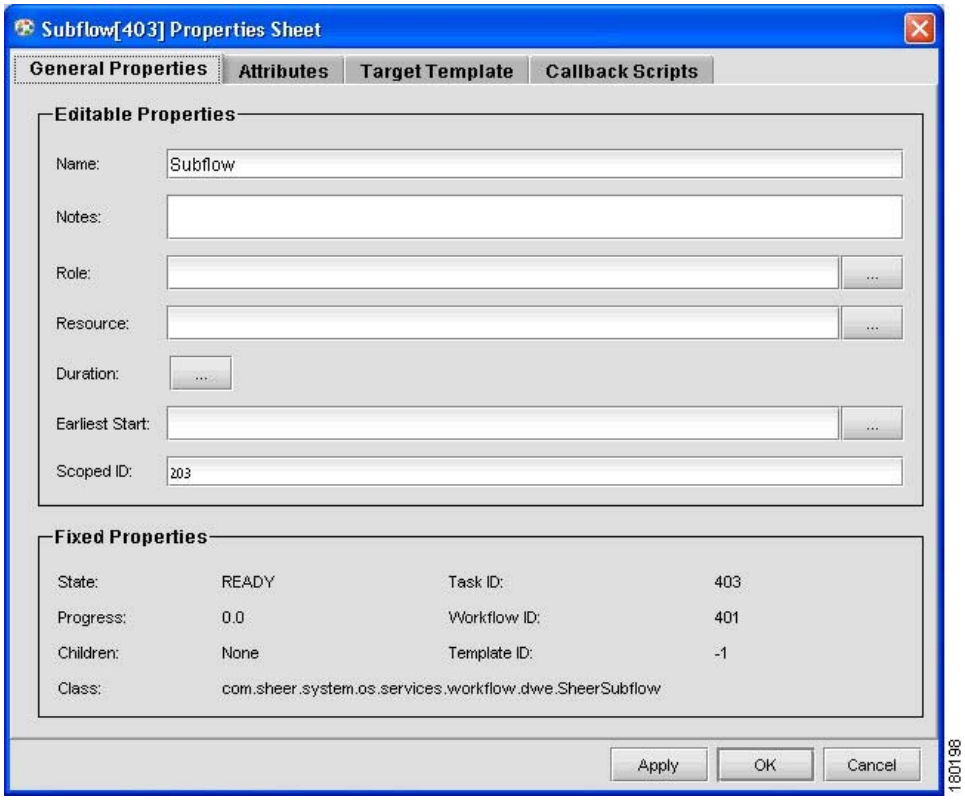
To view the properties of a Subflow task:

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

**Step 2** Choose the required Subflow task in the Drawing Area.

**Step 3** Right-click to display the menu, and choose **Task Properties**. The Task Properties dialog box for the required task is displayed.

*Figure 2-11 General Properties Tab - Subflow*

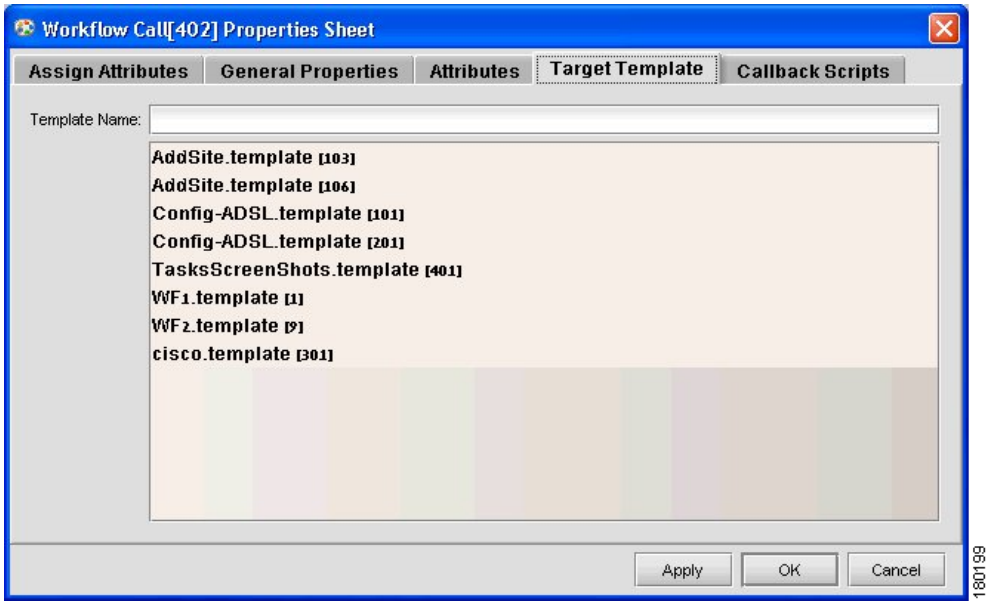



**Note** The Earliest Start option is not supported.

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

**Step 5** Choose the Target Template tab.

*Figure 2-12 Target Template Tab - Subflow*




The Target Template tab enables the user to choose the workflow template that will be embedded.

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

# Lock Task

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

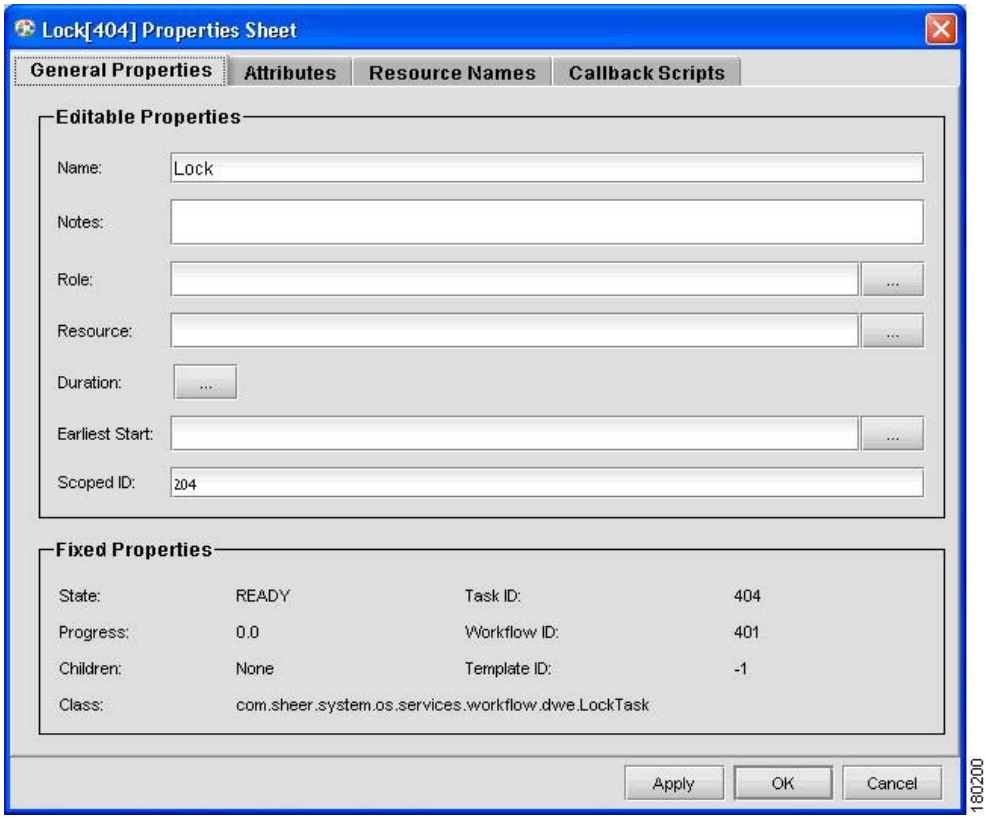
To view the properties of a Lock task:

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

**Step 2** Choose the required lock task in the drawing area.

**Step 3** Right-click to display the menu, and choose **Task Properties**. The Task Properties dialog box for the required task is displayed.

*Figure 2-13 General Properties Tab - Lock*

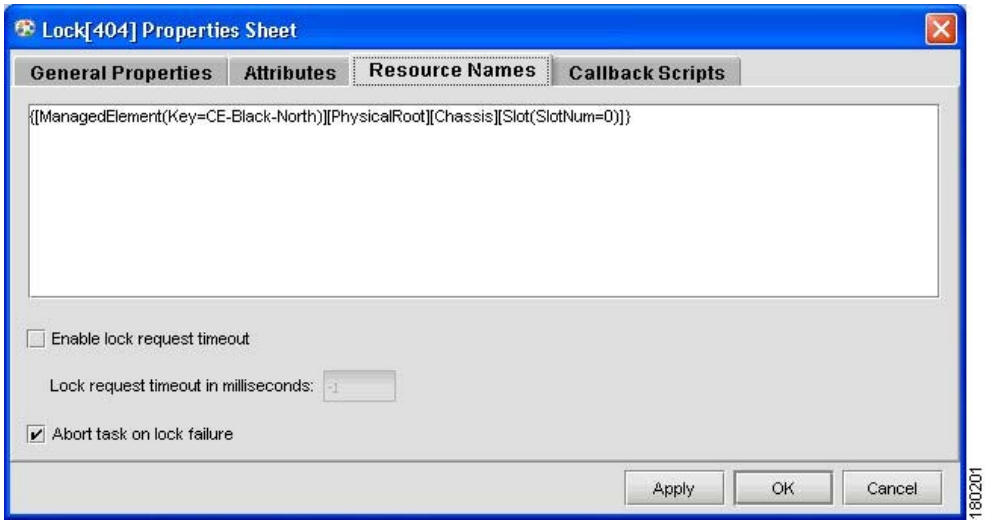



**Note** The Earliest Start option is not supported.

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

**Step 5** Choose the Resource Names tab.

*Figure 2-14 Resource Names Tab - Lock*



The Resource Names tab enables the user to specify the resources that should be locked.

**Note** It is also possible to lock multiple resources.

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

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

- $Attribute name$—Will be evaluated from a workflow attribute by that name.
- $Task name:Attribute name$—Will be evaluated by a task attribute by that name.

The result of the lock task is stored in an attribute named Result in the lock task as follows:

- `Success - lock successfully obtained.`

  or

- `Failed due to timeout - lock failed due to timeout`

  or

- `Failed due to deadlock - lock failed due to a deadlock`

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

# Unlock Task

The Unlock task is used to unlock specific locked resources.

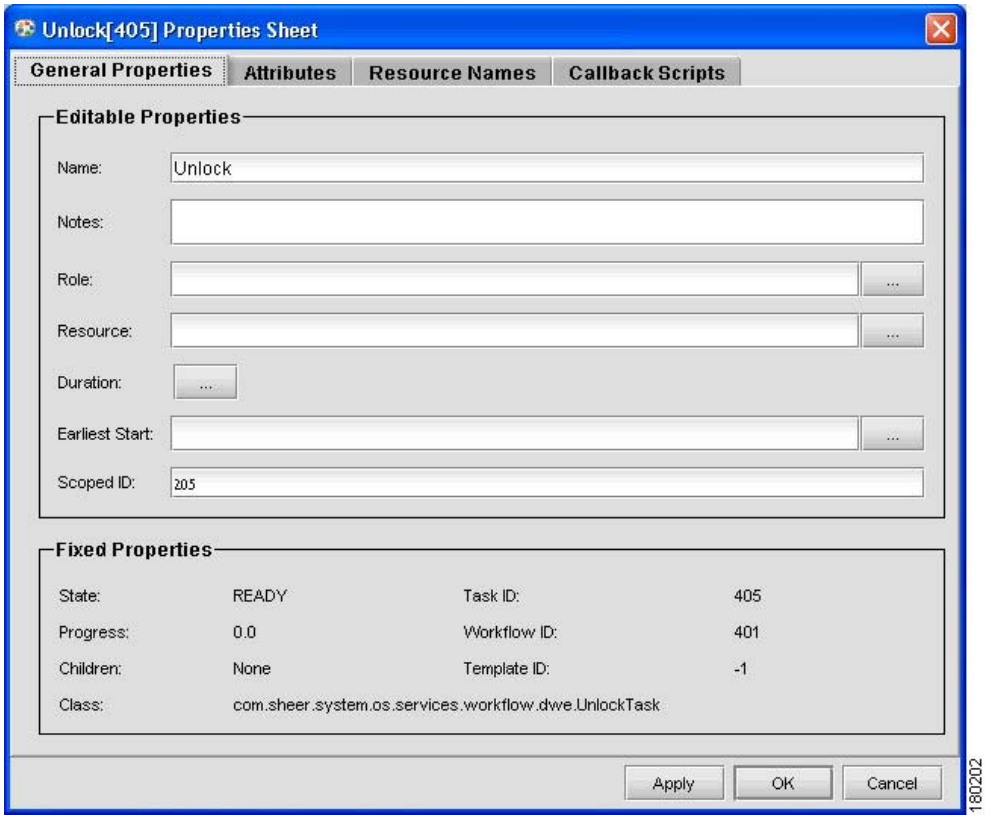
To view the properties of an Unlock task:

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

**Step 2** Choose the required unlock task in the drawing area.

**Step 3** Right-click to display the menu, and choose **Task Properties**. The Task Properties dialog box for the required task is displayed.

*Figure 2-15 General Properties Tab - Unlock*

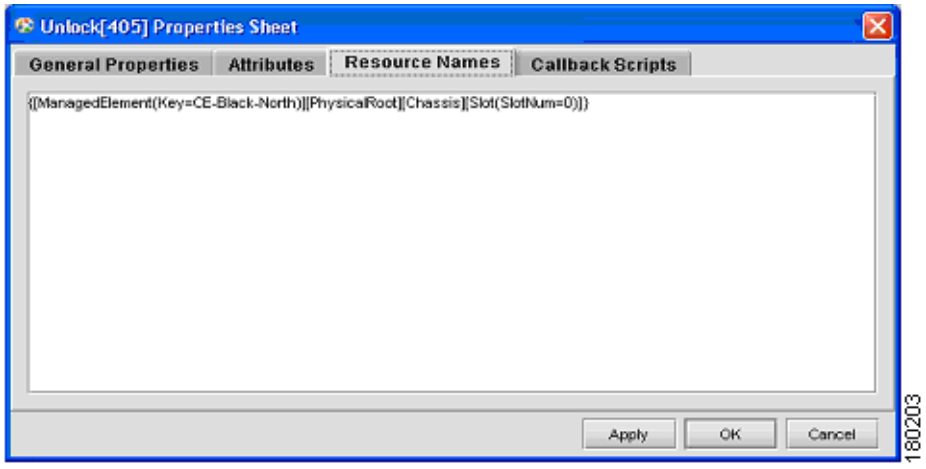



**Note** The Earliest Start option is not supported.

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

**Step 5** Choose the Resource Names tab.

*Figure 2-16 Resource Names Tab - Unlock*



The Resource Names tab enables the user to specify the resources that should be unlocked. In the text area, enter the resources names to be unlocked. Each resource name should be on a separate line.

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

- $Attribute name$—Will be evaluated from a workflow attribute by that name.
- $Task name:Attribute name$—Will be evaluated by a task attribute by that name.

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

# Testing a Workflow

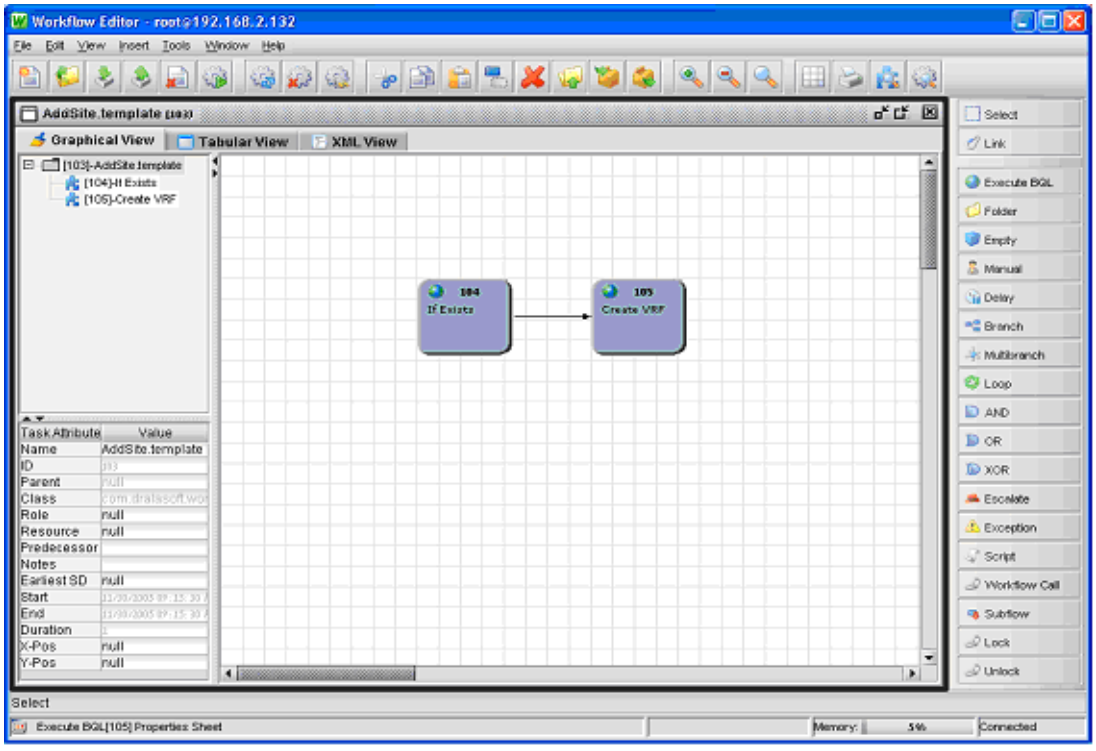
The user can test a BQL workflow locally on the client. This section describes testing a workflow that includes a BQL task.

To test a BQL workflow:

**Step 1** Create a new workflow in the Workflow Editor window (see Creating a New Workflow Template, page 2-4) or retrieve a workflow (see Retrieving a Workflow Template, page 2-5).

**Step 2** On the toolbar, click **Execute Workflow**. The Workflow Editor window changes and displays an additional set of views, as shown in the example.

***Figure 2-17 Execute Workflow***



The new tab that is created in the upper row and displayed in the Workflow Editor window reflects the activation status. For more information about activation status, see Chapter 1, "What Is a Workflow Task?".

In addition, the following views are displayed in the window and relate to the activation of the workflow:

- **Graphical View—**Displays the workflow as a graphical presentation.
- **Tabular View—**Displays a table of all 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:

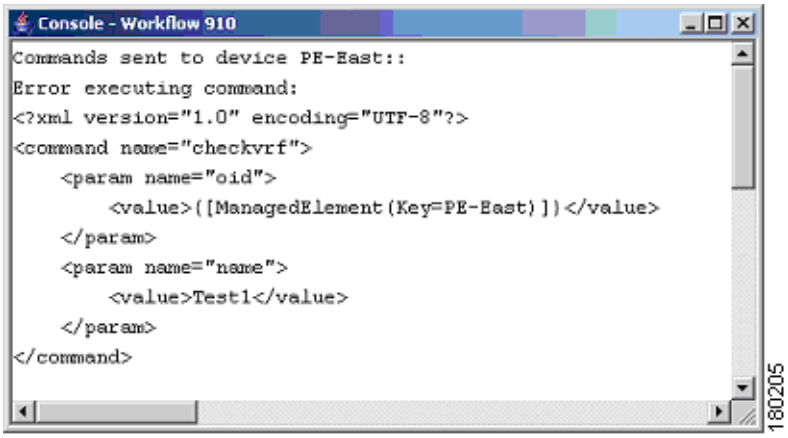
- **Activate—**Activates the workflow.

**Note** This button toggles to Abort when Activation is clicked. Click Abort to stop the workflow.

- **Copy—**Copies the workflow.
- **Delete—**Deletes a running workflow.

**Step 3** Click **Activate**. The Console window is displayed.

***Figure 2-18 Console Window***




The Console window displays the results of each task in the workflow.

**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 colors (in the bottom right corner), as shown in the example.

***Figure 2-19 Tasks Displayed In the Graphical View Tab***

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

- **Blue—**Ready
- **Green—**Active
- **Red—**Abort
- **Grey—**Done
- **Light Grey—**Passive

For more information about activation status, see Chapter 1, "What Is a Workflow Task?".

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

Once the workflow has been tested successfully locally on the client, the user can deploy the workflow to the gateway. For more information, see Deploying a Workflow Template, page 2-5.

# Gateway Workflow Commands and Operations

This section describes how to add utility functions using BQL commands.

## Deleting a Template

To delete a template from the server, use the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Delete">
    <param name="oid">
        <value>{[WorkflowTemplate(Name=template-name)]}</value>
    </param>
</command>
```

## Running a Workflow

To execute a workflow based on an installed template, use the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="RunWorkflow">
    <param name="templateOid">
        <value>{[WorkflowTemplate(Name=template-name)]}</value>
    </param>
    <param name="workflowAttributes">
        <value>
            <IWorkflowAttribute>
                <ID type="Oid">{[WorkflowAttribute(Name=attribute1-name)]}</ID>
                <Value type="String">attribute1-value</Value>
            </IWorkflowAttribute>
            <IWorkflowAttribute>
                <ID type="Oid">{[WorkflowAttribute(Name=attribute2-name)]}</ID>
                <Value type="String">attribute2-value</Value>
            </IWorkflowAttribute>
```

…and so on for other attributes…

```
        </value>
    <IMObject_Array/>
    </param>
</command>
```

This command returns the OID of the new workflow. For example:

```
------ com.sheer.framework.imo.Oid ------
{[Workflow(Id=801)]}
-----------------------------------------
```

# Aborting a Workflow

To abort a workflow, which includes, stopping it if it is running and rolling back all the activation scripts that it has executed, use the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="AbortWorkflow">
    <param name="oid">
        <value>{[Workflow(Id=workflow-id)]}</value>
    </param>
</command>
```

If successful, this command returns nothing.

# Deleting a Workflow

To delete a workflow that is not active, namely, one that has already been completed or was aborted, use the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="Delete">
    <param name="oid">
        <value>{[Workflow(Id=workflow-id)]}</value>
    </param>
</command>
```

If successful, this command returns nothing.

# Getting Workflow Output

To get the output of a workflow, both during its testing and afterwards, use the following BQL command:

```
<?xml version="1.0" encoding="UTF-8"?>
<command name="GetWorkflowOutput">
    <param name="oid">
        <value>{[Workflow(Id=workflow-id)]}</value>
    </param>
</command>
```

This command returns the output as a string. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<IMO>
    <Output type="String">
    …
```

progress messages and script testing results

```
    …
    </Output>
</IMO>
```

# Logging Out

When you have finished working with the Workflow Editor you can log out of the application. Any open workflow templates are automatically saved when you log out.

To log out of the Cisco ANA Workflow Editor:

**Step 1** From the File menu, choose **Exit**,

or

Click in the top right-hand corner to close the Workflow Editor window.

The Workflow Editor is closed.

CHAPTER 3

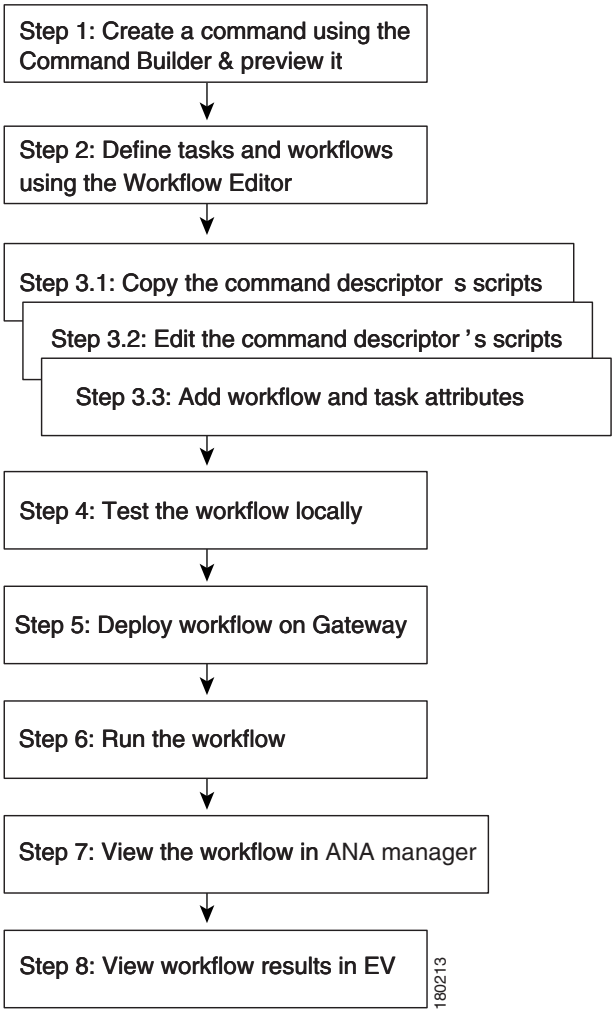
# Getting Started

This chapter provides the steps required to create, test, deploy, run and view a workflow.

## Step-By-Step

The flow below describes the steps required when working with and managing workflows. The steps are described in the order in which they must be performed.

*Figure 3-1 Flow For Creating, Testing, Deploying, Running and Viewing a Workflow*




**Step 1** Create a command using the Command Builder and preview it. For more information, refer to the *Cisco Active Network Abstraction Command Builder User Guide*.

**Step 2** Define tasks and workflows using the Workflow Editor. For more information, see Chapter 2, "Working With the Cisco ANA Workflow Editor".

**Step 3** Copy the command descriptor's scripts—The user can copy the command descriptor's scripts that were defined using the Command Builder to the Sheer Workflow Editor BQL Tasks command template tabs.

Edit the command descriptor's scripts—The user can edit the command descriptor's scripts that were defined using the Command Builder to the Sheer Workflow Editor BQL Tasks command template tabs, as required.

Add workflow and task attributes—The user can add workflow and task attributes using the Sheer Workflow Editor.

**Step 4** Test the workflow locally. For more information, see Testing a Workflow, page 2-18.

**Step 5** Deploy workflow on gateway. For more information, see Deploying a Workflow Template, page 2-5.

**Step 6** Run the workflow. The user can run the workflow using a BQL command. For more information, see Running a Workflow, page 2-20.

**Step 7** Manage and view the workflow in Cisco ANA Manage. For more information, see Chapter 4, “Managing Workflows”.

**Step 8** View workflow results in EventVision. For more information, see Viewing the Workflow History Using Cisco ANA EventVision, page 4-6.

CHAPTER 4

# Managing Workflows

This chapter provides instructions for launching Cisco ANA Manage. In addition, it describes the lists of all the deployed workflow templates for the Cisco ANA gateway, and the actions that can be performed.

- Starting Cisco ANA Manage—Describes how to open the Cisco ANA Manage application.
- Workflow Engine Branch—Describes the Workflow Engine branch that enables the user to manage templates and scripts during runtime.
- Viewing the List of Templates and Template Properties—Describes how to view the list of workflow templates and template properties and attributes.
- Deleting a Workflow Template—Describes how to delete a workflow template.
- Viewing the Output Of a Workflow—Describes how to view the output of a workflow.
- Aborting a Workflow—Describes how to abort a workflow that is running.
- Deleting a Workflow—Describes how to delete a workflow that is done or has been stopped.
- Viewing the Workflow History Using Cisco ANA EventVision—Describes how to view the workflow history using EventVision.
- Logging Out—Describes how to log out of Cisco ANA Manage.

## Starting Cisco ANA Manage

This section provides instructions for launching  Cisco ANA Manage. Cisco ANA Manage is password protected to ensure security. Before you start, make sure you know the username, password, and the Cisco ANA gateway IP address that is required.

To start Cisco ANA Manage:

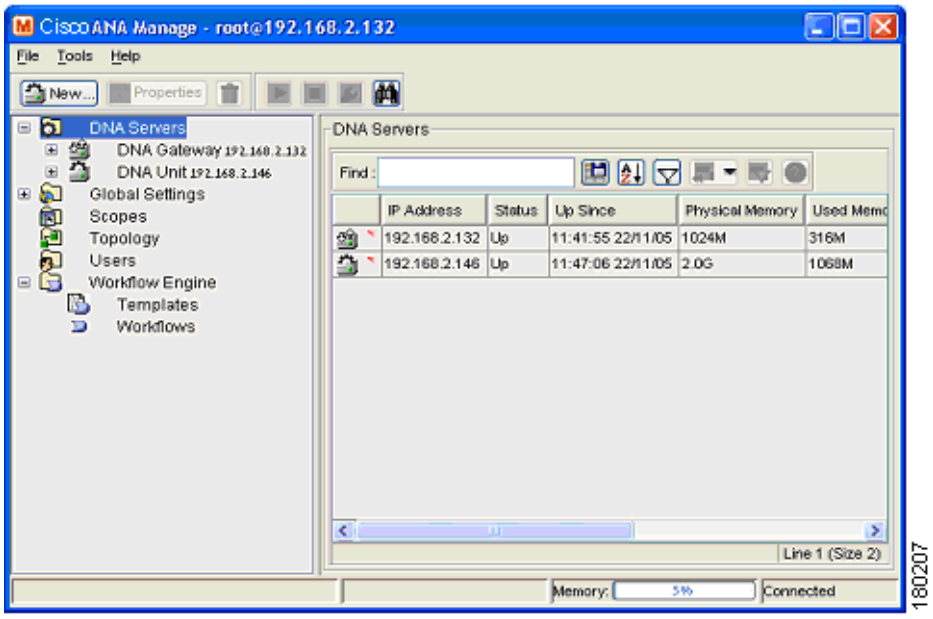
**Step 1** From the Start menu, choose the **Programs** folder, then **Cisco ANA/Cisco ANA Manage**. The Cisco ANA Manage Login dialog box is displayed.

**Step 2** Enter your username, password and host (Cisco ANA gateway IP address).

**Note** The gateway IP address that was used when the user last logged in is automatically displayed in the Host field.

**Step 3** Click **OK**. The Cisco ANA Manage window is displayed.

***Figure 4-1 Cisco ANA Manage Window***




The window is divided into two panes, as follows:

- The tree pane.
- The context pane.

**Note** For a detailed description of the Cisco ANA Manage application, refer to *Cisco Active Network Abstraction Administrator Guide*.

# Workflow Engine Branch

The Workflow Engine branch is used to manage templates and workflows and includes the following sub-branches:

- **Templates—**Displays a list of the workflow templates, and enables the user to view the properties of the workflow template. For a detailed description of Cisco ANA Manage, refer to *Cisco Active Network Abstraction Administrator Guide*.
- **Workflows—**Enables the user to manage the script in runtime by viewing and altering its current status. For a detailed description of Cisco ANA Manage, refer to *Cisco Active Network Abstraction Administrator Guide*.

# Viewing the List of Templates and Template Properties

Cisco ANA Manage enables the user to view a list of the workflow templates and template properties (attributes).

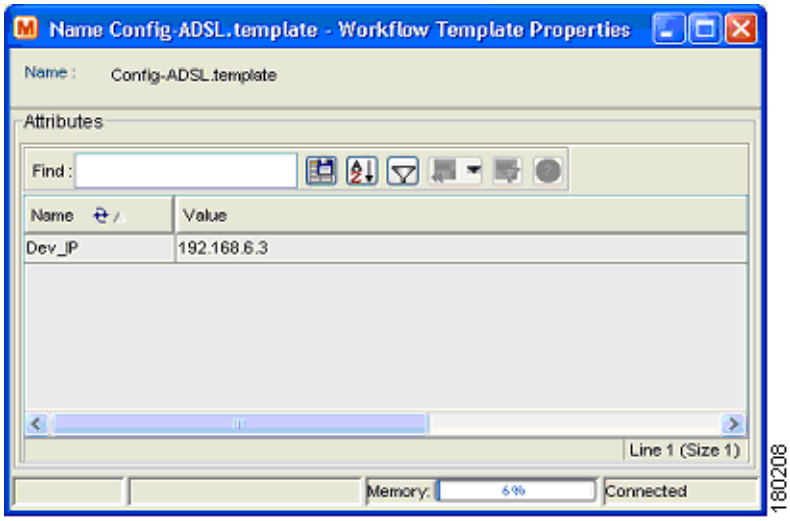
To view the list of templates and template properties:

**Step 1** Choose the Templates sub-branch. The list of workflow templates is displayed in the table.

**Step 2** Choose the required workflow template in the table.

**Step 3** Right-click to display the menu, and choose **Properties**. The Workflow Template Properties dialog box is displayed with the properties and attributes of the required workflow template.

*Figure 4-2 Workflow Template Properties Dialog Box*




The name of the selected template is displayed in the header and at the top of the dialog box. The following properties are displayed in the table of the Workflow Template Properties dialog box:

- **Name—**The attribute names defined for the tasks included in the workflow, as defined in the Task Properties dialog box using the Workflow Editor.
- **Value—**The values defined for the tasks included in the workflow, as defined in the Task Properties dialog box using the Workflow Editor.

**Step 4** Click in the top right-hand corner to close the Template Properties dialog box.

# Deleting a Workflow Template

Cisco ANA Manage enables the user to delete a workflow template.

To delete a workflow template:

**Step 1** Choose the Templates sub-branch. The list of workflow templates is displayed in the table.

**Step 2** Choose the workflow template that you want to delete in the table.

**Step 3** Right-click on the required workflow template to display the menu, and choose **Delete**, or on the toolbar, click **Delete**. A warning message is displayed.

**Step 4** Click **Yes**. The selected workflow template is deleted, and is no longer displayed in the table.

# Viewing the Output Of a Workflow

The user can view the output of a workflow whether it is running, done or aborted.

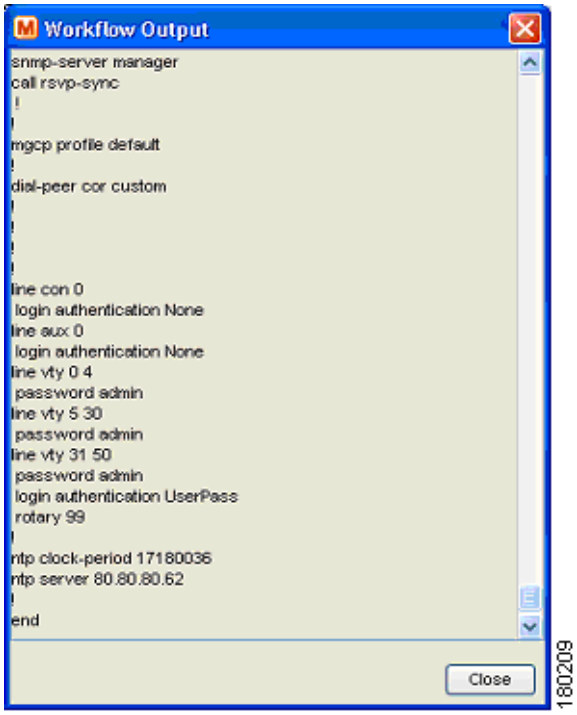
To view the output of a workflow:

**Step 1** Choose the Workflows sub-branch. The list of workflows is displayed in the table.

**Step 2** Choose the required workflow in the table.

**Step 3** Right-click to display the menu, and choose **Show Output**. The Output dialog box is displayed.

***Figure 4-3 Output Dialog Box***



The Output dialog box displays the output and details of the workflow.

**Step 4** Click **Close**. The Output dialog box is closed.

# Aborting a Workflow

The user can abort a workflow that is running. In addition, if any task in the workflow aborts, the workflow itself will abort.

When a workflow aborts a workflow rollback will occur:

- Workflow rollback causes the activation scripts that have already been run (by Execute BQL task) to rollback.
- The rollback of an activation script is the execution of the rollback section of the script, as defined in the Command Builder.
- The order of the scripts rollback is the reverse of their execution order.

**Note** Gateway commands do not support rollback.

Rollback can be disabled for specific BQL tasks by setting the RollbackEnabled value task attribute to false in the respective BQL task. This is useful for a BQL task executing a script which does not have an appropriate rollback, or a BQL task executing a Gateway command.

**Note** A workflow instance can only be aborted when it is running. It cannot be aborted when the process is done.

To abort a running workflow:

**Step 1** Choose the Workflows sub-branch. The list of workflows is displayed in the table.

**Step 2** Right-click on the required workflow, and choose **Abort**. A warning message is displayed.

**Step 3** Click **Yes**. The workflow is stopped, and the state of the workflow changes to **Aborted** in the table of the Workflows sub-branch.

# Deleting a Workflow

The user can delete a workflow from the Workflows sub-branch in the Cisco ANA Manage window. The workflow is deleted from the database.

To delete a workflow:

**Step 1** Choose the Workflows sub-branch. The list of workflows is displayed in the table.

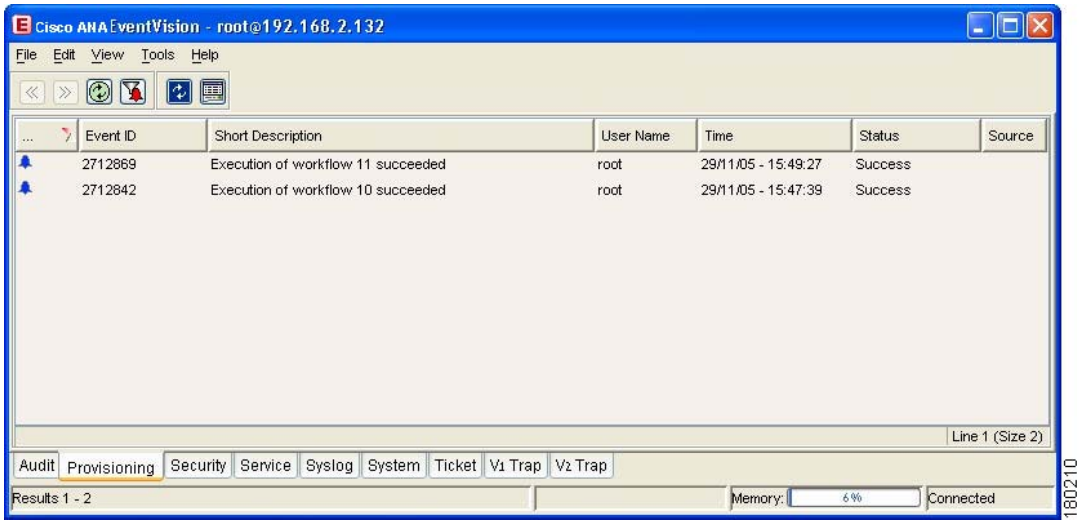
**Step 2** Right-click on the required workflow to display the menu, and choose **Delete**, or on the toolbar, click **Delete**. A warning message is displayed.

**Step 3** Click **Yes**. The selected workflow is deleted, and is no longer displayed in the table.

# Viewing the Workflow History Using Cisco ANA EventVision

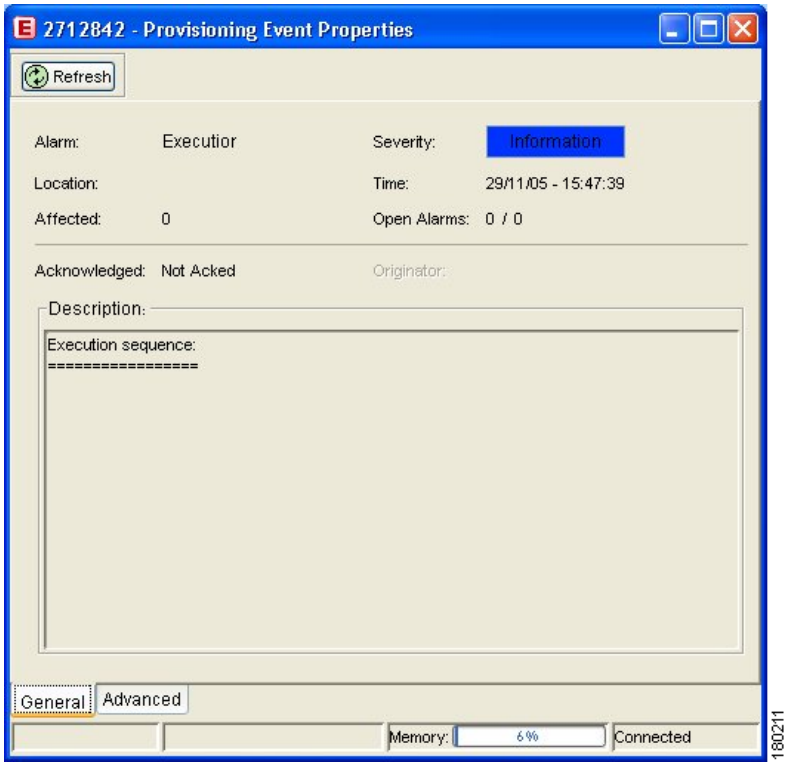
Event Vision enables the user to view the workflow history, including when workflows have been completed, their status, and the command invoked, as shown in the example.

*Figure 4-4 Cisco ANA EventVision - Provisioning Tab*



The user can also view the properties of the event, as shown in the example below.

***Figure 4-5 Properties Of The Event***




The Description area of the Provisioning Event Properties dialog box displays the details of the execution of the workflow, including all the scripts, script rollback (if it occurred), and log messages.

# Logging Out

When the user has finished working with Cisco ANA Manage and logs out of the application, any changes that were made are automatically saved.

To log out of Cisco ANA Manage:

**Step 1** From the File menu, choose **Exit**,

or

Click in the top right-hand corner to close the Cisco ANA Manage window.

Cisco ANA Manage is closed.

CHAPTER 5

# Customizing the Workflow Editor

This chapter explains how to extend the Workflow Engine with custom tasks and Workflow Editor callbacks.

# Extending the Workflow Engine With Custom Tasks and Workflow Editor Callbacks

This section describes customizing tasks and Workflow Editor callbacks.

## Coding

### Custom Tasks

The procedure for developing custom tasks is explained in the Dralasoft documentation.

For example, here is the definition of a simple task class:

```
package samples;

import com.dralasoft.workflow.Key;
import com.dralasoft.workflow.SynchronousTask;

public class CustomTask1 extends SynchronousTask {
    public CustomTask1(Key _key) {
        super(_key);
    }

    public void perform() {
        System.err.println("Hello from CustomTask1");
    }
}
```

### Custom Task Panel Factories

A TaskPanelFactory implementation class can also be assigned to each custom task class. This allows for creating custom property sheets for the custom class. This factory class should implement the interface *com.dralasoft.gui.common.ext.TaskPanelFactory.* A simple way to do this is to extend *com.dralasoft.gui.common.ext.DefaultPanelFactory* and override some of its methods.

## Workflow Editor Callbacks Class

It is possible to develop a class that implements the workflow editor callbacks. This class should implement the interface *com.sheer.client.workflowstudio.IWorkflowEditorCallbacks*, which includes the following methods:

```
/**
* Called before the deploy template action
* @param templateName
* @return true if deploy template action should proceed
*/
public boolean preDeployTemplate(String templateName);

/**
* Called after the deploy template action
* @param templateName
*/
public void postDeployTemplate(String templateName);

/**
* Called before the delete template action
* @param templateName
* @return true if delete template action should proceed
*/
public boolean preDeleteTemplate(String templateName);

/**
* Called after the delete template action
* @param templateName
*/
public void postDeleteTemplate(String templateName);

/**
* @param templateName
* @return true if this template should be displayed, false if not
*/
public boolean shouldDisplayTemplate(String templateName);
```

# Packaging For Deployment

Classes and resources must be packaged into JARs to be deployed on the server. A JAR can contain multiple tasks and an optional workflow editor callbacks implementation.

In addition to the class files, each JAR must contain a descriptor file named *extension-config.xml*. This file contains the XML block or blocks that define the tasks' appearance in the task palette, the tasks' custom panel factories, and, optionally, the workflow editor callbacks implementation.

## Custom Tasks

In addition to the custom tasks class files, task icons should be included in the JAR.

Task icons should measure 16 by 16 pixels and must be placed in the JAR in the subdirectory, com/dralasoft/gui/common/images/16x16.

## Custom Task Panel Factories

The full class name of the task panel factory should be added in an element called task-panel-factory-class inside the custom-task element. If this element were not specified, *com.dralasoft.gui.common.ext.DefaultPanelFactory* would be used for the custom task.

## Workflow Editor Callbacks Class

Adding an editor-callbacks-class element to the extension-config element configures a class that implements the workflow editor callbacks. This element should be added to the extension-config.xml file in only one of the JAR files (if the element is present in more then one descriptor, one element's value would be used arbitrarily).

Here is an example of the contents of a task descriptor file that defines two task types, one of which has a task panel factory, and an editor callback implementation:

```
<extension-config>
    <editor-callbacks-class>
        samples.EditorCallbacksImpl
    </editor-callbacks-class>

    <custom-task>
        <class-name>samples.CustomTask1</class-name>
        <label>Custom Task 1</label>
        <icon>task1.png</icon>
        <tooltip>Custom Task 1</tooltip>
        <menu-display>true</menu-display>
        <toolbar-display>true</toolbar-display>
        <task-panel-factory-class>
            com.sheer.client.workflowstudio.TestTaskPanelFactory
        </task-panel-factory-class>
    </custom-task>

    <custom-task>
        <class-name>samples.CustomTask2</class-name>
        <label>Custom Task 2</label>
        <icon>task2.png</icon>
        <tooltip>Custom Task 2</tooltip>
        <menu-display>true</menu-display>
        <toolbar-display>true</toolbar-display>
    </custom-task>
</extension-config>
```

# Deploying

The procedure below describes deploying JARs:

**Step 1** On the Cisco ANA gateway, copy the JARs to the following directory:

```
/export/home/sheer/dralasoft_extensions
```

**Note** Create the Cisco ANA gateway if necessary.

**Step 2** Run the script:

```
/export/home/sheer/Main/scripts/
installDralasoftExtensions.pl
```

**Note** Run this script every time you want to add or remove a JAR, or replace an existing JAR with a new version

The script installs and uninstalls JARs so that the set of installed JARs is the same as the contents of the directory /export/home/sheer/dralasoft_extensions. It is therefore recommended that you keep your extension JARs here even after they have been deployed.

**Note** The script also restarts the Workflow engine, so it is best to note if there are currently running workflows in the engine.

**Step 3** When the script is done, run the Workflow Editor again. The deployed JARs are automatically downloaded, and the new tasks are included in the task toolbar.