Cisco Unified Contact Center Express Editor Step Reference Guide, Release 11.0(1)

Cisco Unified Contact Center Express Scripting and Development Series: Volume 2

First Published: August 27, 2015

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

CONTENTS

Preface

This book is Volume 2 of the *Cisco Unified Contact Center Express (Cisco Unified CCX) Scripting and Development Series*. This series provides information about how to use the Cisco Unified Contact Center Express Editor to develop a wide variety of interactive scripts.

The *Cisco Unified Contact Center Express Scripting and Development Series* contains three volumes:

- *Volume 1, Getting Started with Scripts*, provides an overview of the Cisco Unified CCX and the Cisco Unified CCX Editor web interface.
- *Volume 2, Editor Step Reference* (this book), describes each individual step in the Cisco Unified CCX Editor palettes.
- *Volume 3, Expression Language Reference,* provides details on the working with the Cisco Unified CCX Expression Framework.

Note Your Cisco Unified CCX product might not contain all the steps described in this guide. For a description of the steps contained in individual Cisco Unified CCX products, see "Related Documentation" section on page iii.

Audience

Cisco Unified Contact Center Express Editor Step Reference Guide, Release 11.0(1) is written for application script developers who use the Cisco Unified CCX Editor to create and modify scripts. This guide targets developers who have the IP telephony knowledge required to create application scripts and who also have some background in programming or scripting. While experience with Java is not necessary, it will help developers fully utilize the capabilities of the Cisco Unified CCX.

Note For an overview of important Cisco Unified CCX Editor terms and concepts, including information on using the Cisco Unified CCX Editor interface and samples of script applications, see the *Cisco Unified CCX Scripting and Development Series: Volume 1, Getting Started with Scripts*.

Organization

This guide is organized by the palettes of the Cisco Unified CCX Editor, as shown in the following table.

Table 1 Step Descriptions for Each Palette

Chapter/ Section	Title	Description
Chapter 1	Cisco Unified CCX Editor Step Availability	Lists the Cisco Unified CCX Editor steps available according to Cisco Unified CCX license options.
Chapter 2	Cisco Unified CCX Editor Palette Step Descriptions	Describes the steps in each Step Palette according to the Palette organization.
Sections	Step Reference Descriptions	
1	General Steps, page 1	Describes the Cisco Unified CCX Editor General steps that provide basic programming functionality for scripts
2	Trigger Steps, page 23	Describes the Cisco Unified CCX Editor steps that manage triggers
3	Session Steps, page 26	Describes the Cisco Unified CCX Editor steps that manage sessions
4	Contact Steps, page 34	Describes the Cisco Unified CCX Editor steps that control contacts
5	Call Contact Steps, page 39	Describes the Cisco Unified CCX Editor steps that control calls
6	eMail Contact Steps, page 60	Describes the Cisco Unified CCX Editor steps used to send e-mails
7	HTTP Contact Steps, page 65	Describes the Cisco Unified CCX Editor steps used to receive HTTP requests and to send HTTP responses in web-enabled server applications
8	Media Steps, page 87	Describes the Cisco Unified CCX Editor steps used to process media interactions with users
9	User Steps, page 144	Describes the Cisco Unified CCX Editor steps used to authenticate, retrieve, and assign user attributes
10	Prompt Steps, page 149	Describes the Cisco Unified CCX Editor steps used to create prompts
11	Grammar Steps, page 162	Describes the Cisco Unified CCX Editor steps used to provide grammars to scripts
12	Document Steps, page 170	Describes the Cisco Unified CCX Editor steps used to handle a document object, such as files stored in a database or on an HTTP server
13	Database Steps, page 189	Describes the Cisco Unified CCX Editor steps used to read and write data to database tables
14	ACD Steps, page 198	Describes the Cisco Unified CCX Editor steps used to queue calls and connect calls to available resources

Table 1 Step Descriptions for Each Palette (continued)

Chapter/ Section	Title	Description
15	ICM Step, page 215	Describes the Cisco Unified CCX Editor steps used to provide queueing
16	Java Steps, page 217	Describes the Cisco Unified CCX Editor steps used to bridge scripts with existing Java objects
17	Context Service Steps, page 227	Describes the Cisco Unified CCX Editor steps used to bridge scripts with Context Service objects

Related Documentation

Refer to the following documents for further information about Cisco Unified CCX applications and products:

- *Cisco Unified Contact Center Express Scripting and Development Series: Volume 3,* Expression Language Reference
- *Cisco Unified Contact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts*
- Cisco Unified *Contact Center Express* Servicing and Troubleshooting Guide
- Cisco Unified *Contact Center Express* Administration Guide
- Cisco Unified *Contact Center Express* Installation and Upgrade Guide
- Cisco Unified Communications Manager Administration Guide
- Cisco Unified Communications Manager Extended Services Administrator Guide
- Cisco Unified Communications Manager System Guide
- Cisco Unified Communications Manager Enterprise Installation and Configuration Guide
- Cisco Unified Contact Center Express Gateway Deployment Guide
- Cisco Unified Contact Center Solution Reference Network Design (SRND) documents

Glossary

For the complete list of terms used in Cisco Unified CCX and Cisco Unified IP IVR, see

http://www.cisco.com/en/US/products/sw/custcosw/ps1846/prod_technical_reference_list.html

Conventions

This manual uses the following conventions.

Convention	Description
boldface font	**Boldface** font is used to indicate commands, such as user entries, keys, buttons, and folder and submenu names. For example: • Choose **Edit > Find**. • Click **Finish**.
italic font	*Italic* font is used to indicate the following: • To introduce a new term. Example: A *skill group* is a collection of agents who share similar skills. • For emphasis. Example: *Do not* use the numerical naming convention. • An argument for which you must supply values. Example: IF (*condition, true-value, false-value*) • A book title. Example: See the *Cisco Unified Contact Center Express Installation Guide*.
`window` font	`Window` font, such as Courier, is used for the following: • Text as it appears in code or information that the system displays. Example: `<html><title>Cisco Systems,Inc. </title></html>` • File names. Example: `tserver.properties`. • Directory paths. Example: `C:\Program Files\Adobe`
string	Nonquoted sets of characters (strings) appear in regular font. Do not use quotation marks around a string or the string will include the quotation marks.
[ ]	Optional elements appear in square brackets.
{ x | y | z }	Alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.

Convention	Description
< >	Angle brackets are used to indicate the following: • For arguments where the context does not allow italic, such as ASCII output. • A character string that the user enters but that does not appear on the window such as a password.
^	The key labeled Control is represented in screen displays by the symbol ^. For example, the screen instruction to hold down the Control key while you press the D key appears as ^D.

Obtaining Documentation, Obtaining Support, and Security Guidelines

For information on obtaining documentation, obtaining support, 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

Documentation Feedback

You can provide comments about this document by sending email to the following address:

ccbu_docfeedback@cisco.com

We appreciate your comments.

CHAPTER 1

Cisco Unified CCX Editor Step Availability

Table 1-1 lists the Cisco Unified CCX Editor step palettes available for each Cisco Unified CCX license option.

Note All the palettes listed in the following table are displayed in the Cisco Unified CCX Editor by default. However, the Cisco Unified CCX Engine enforces licensing at run time; if a script uses a step for which your system is not licensed, the Cisco Unified CCX Engine prevents the script from being loaded.

Table 1-1 Step Palette Availability with Cisco Unified CCX License Options

	Cisco Unified IP IVR	Cisco Unified CCX Standard	Cisco Unified CCX Enhanced	Cisco Unified CCX Premium
General[1]	X	X	X	X
Session	X	X	X	X
Contact	X	X	X	X
Call Contact	X	X	X	X
Email Contact	X			X
HTTP Contact	X			X
Media [2]	X	X	X	X
User	X	X	X	X
Prompt[3]	X	X	X	X
Grammar	X	X	X	X
Doc	X	X	X	X
DB	X			X
ACD		X[4]	X[5]	X
ICM	X			
Java[6]	X		X	X
Context Service	X	X	X	X

1. The ‘Get Reporting Statistic’ step is only available with the Cisco Unified CCX packages.
2. The ‘Voice Browser’ step is only available with the Cisco UnifiedIP IVR or Cisco Unified CCX Premium packages.
3. The ‘Create TTS Prompt’ step is only available with the Cisco UnifiedIP IVR or Cisco Unified CCX Premium packages.

4. The ‘Set Priority,’ and ‘CreateCSQSpokenNamePromptStep’ steps are not available with Cisco Unified CCX Standard
5. The ‘Set Priority,’ and ‘CreateCSQSpokenNamePromptStep’ steps are only available with Cisco Unified CCX Enhanced or Cisco Unified CCX Premium.
6. When the step in the Java palette is enabled, the Java functionality of the expression language is also enabled.

CHAPTER 2

Cisco Unified CCX Editor Palette Step Descriptions

This reference chapter section describes the Cisco Unified CCX Editor steps according to the Editor palette containing them:

- General Steps
- Trigger Steps
- Session Steps
- Contact Steps
- Call Contact Steps
- eMail Contact Steps
- HTTP Contact Steps
- Media Steps
- User Steps
- Prompt Steps
- Grammar Steps
- Document Steps
- Database Steps
- ACD Steps
- ICM Step
- Java Steps
- Context Service Steps

General Steps

The steps in the General palette of the Cisco Unified CCX Editor provide basic programming functionality for scripting.

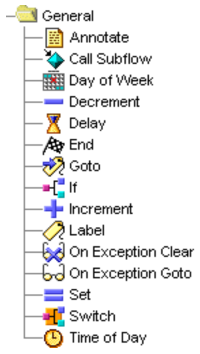
This section contains the following topics:

- Start Step
- Annotate Step

- Call Subflow Step
- Day of Week Step
- Decrement Step
- Delay Step
- Do Step
- Session Steps
- If Step
- Increment Step
- Label Step
- On Exception Clear Step
- On Exception Goto Step
- Set Step
- Switch Step
- Time of Day Step

The figure that follows shows the steps in the General palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

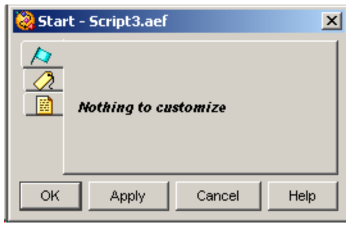
Figure 2-1 General Palette Steps



Start Step

The Start step, while not be in the General Palette, it appears automatically in the Design pane of the Cisco Unified CCX Editor window whenever you create a script. To create a script, from the Editor menu bar, select **File > New**.

Figure 2-2 Start Customizer Window



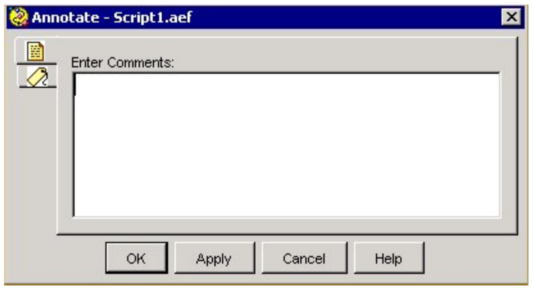
The Start step is not shown in any palette and you do not need to define any properties for it. However, Start provides a customizer window so you can use the Label and Annotate tabs to add a label or comment to the beginning of the script.

Annotate Step

Use the Annotate step to enter comments that explain the function of a script segment.

Note Effective with CRS 4.0(1), all CRS Editor customizer windows include a Comments tab you can use to add step-specific notes.

Figure 2-3 Annotate Customizer Window



To use the Annotate customizer window to annotate a script, enter your comments—inserting line breaks if the comments are lengthy—in the Enter Comments field, and then click **OK**.

The Annotate customizer window closes, and the comments you entered appear next to the Annotate icon in the Design pane of the Cisco Unified CCX Editor.

This step has no effect on script logic.

Call Subflow Step

Use the Call Subflow step to execute a subflow, which is analogous to a subroutine or module in structured programming.

You create the subflow with the Cisco Unified CCX Editor as an independent script that you can reuse in other scripts. You can also call subflows from within scripts that are themselves used as subflows.

If the script was originally loaded from disk, the subflow script must reside in the same folder as the script that calls it. If the script was loaded from the Repository, then the subflow must also reside in the Repository.

Note Use a subflow in a script (instead of a redirect) to transfer a call to another script.

Note When you debug a script that uses subflows, the Cisco Unified CCX Editor does not debug the subflows; it checks only the script you are debugging. If a subflow is invoked, it will be executed completely until the subflow returns back the control to the main flow being debugged. You must debug each subflow separately. The debug validation process does not detect mapping of mismatched types, but such a mapping causes a run-time exception when the script is loaded into the execution engine.

During run time, if an exception occurs within a subflow, and you do not handle the exception within the subflow, the exception is available to the parent script for processing. For more information about exceptions, see the "On Exception Goto Step" section on page 2-14.

The Call Subflow customizer window contains three tabs:

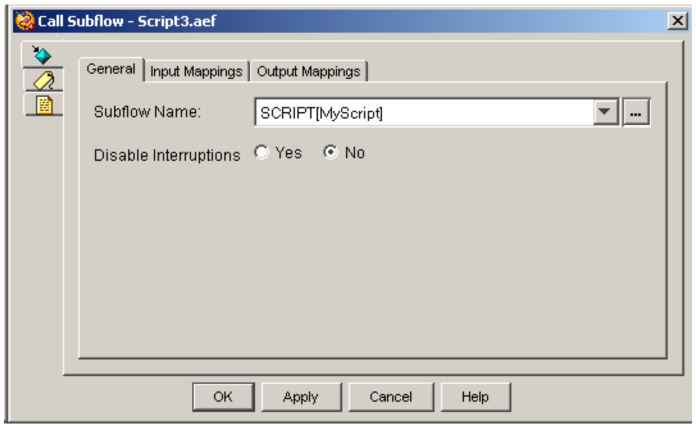
- General tab (Call Subflow step)
- Input Mappings tab (Call Subflow step)
- Output Mappings tab (Call Subflow step)

The following sections describe these tabs.

General tab (Call Subflow step)

Use the General tab of the Call Subflow customizer window to specify the file name of the subflow you want to call.

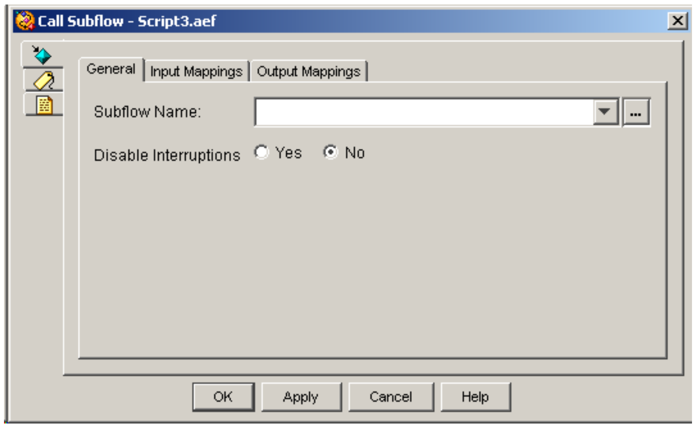
Figure 2-4 Call Subflow Customizer Window—General Tab



Subflow Name: SCRIPT[MyScript]

Disable Interruptions Yes [✓ No]

Figure 2-5 Call Subflow Customizer Window—General Tab



Subflow Name:
Disable Interruptions ( ) Yes [✓ No]

Table 2-1 describes the properties of the General tab of the Call Subflow customizer window:

Table 2-1 Call Subflow Properties—General Tab

Property	Description
Subflow Name	File name of the script that contains the subflow you want to call. All user script subflow names are displayed as SCRIPT[name] entries. For example: SCRIPT[myscript.aef] SCRIPT[myscript\MyScript.aef] **Note** To locate a subflow filename, open the Expression Editor by clicking ... . Then use the Script tab to browse. All relative subflow names are displayed as strings from the user repository if the edited script is loaded from the repository.
Disable Interruptions (radio buttons)	If Yes is selected, execution of the step can not be interrupted by external events.

Table 2-2 describes the properties of the General tab of the Call Subflow customizer window:

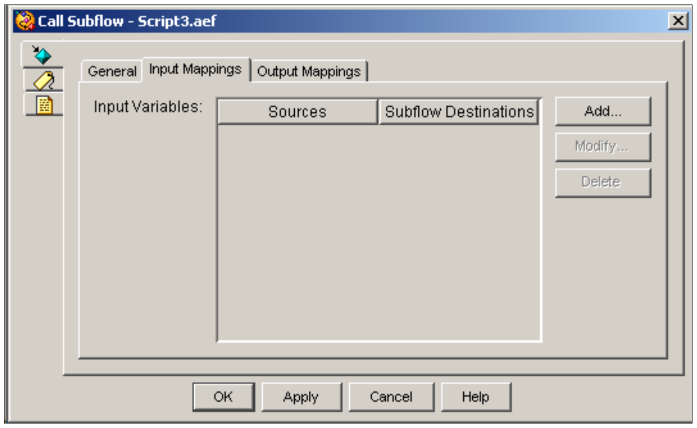
Table 2-2 Call Subflow Properties—General Tab

Property	Description
Subflow Name	File name of the script that contains the subflow you want to call. To locate a subflow filename, open the Expression Editor by clicking ... . Then use the Script tab to browse. All relative subflow names are displayed as strings from the user repository if the edited script is loaded from the repository.
Disable Interruptions (radio buttons)	If Yes is selected, execution of the step can not be interrupted by external events.

Input Mappings tab (Call Subflow step)

Use the Input Mappings tab of the Call Subflow customizer window to map variables or expressions from the main script to variables in the subflow you specified in the General tab of the Call Subflow customizer window.

Figure 2-6 Call Subflow Customizer Window—Input Mapping Tab



Note You must define variables in the map script before you can map them.

You can map variables only to variables of the same type. (For example, you can map a string variable in the main script only to a string variable in the subflow).

You can pass in any valid expression; for example, "4" or an expression like "counter + 3".

Cisco Unified CCX evaluates all expressions you specify before calling the subflow. Cisco Unified CCX then stores the results of the expressions in the specified subflow variable names; these variables must be of the same type as the expression.

Note Use the Output Mappings tab to specify how to return values from the subflow to the main flow. For more information, see the "Output Mappings tab (Call Subflow step)" section on page 2-7.

Table 2-3 describes the properties of the Input Mapping tab of the Call Subflow customizer window.

Table 2-3 Call Subflow Properties—Input Mappings Tab

Properties / Buttons	Description
Sources	Expressions to be evaluated before invoking the subflow; the results are stored in the subflow variable specified in the "Destination" field.
Subflow Destinations	Names of the subflow variables in which the result from the "Sources" field will be stored.

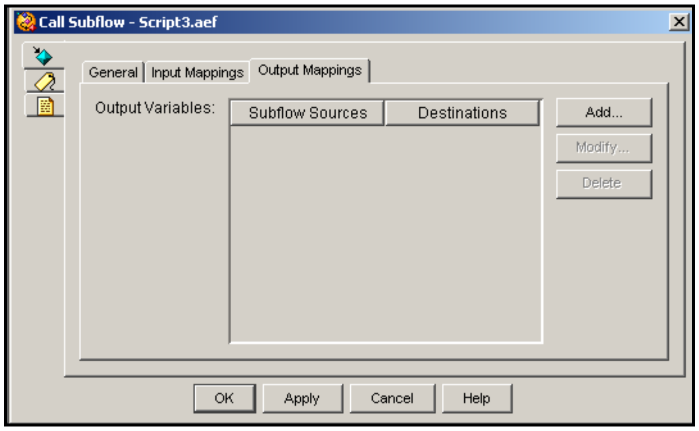
Table 2-3 Call Subflow Properties—Input Mappings Tab

Properties / Buttons	Description
Add / Modify (buttons)	Use these buttons to access the Add or Modify Input Variable dialog box where you can add or modify subflows. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove parameter mapping information, highlight a value in the list and click **Delete**.

Output Mappings tab (Call Subflow step)

Use the Output Mappings tab of the Call Subflow customizer window to map variables or expressions from the subflow you specified in the General tab of the Call Subflow customizer window to variables in the current script.

Figure 2-7 Call Subflow Customizer Window—Output Mappings Tab



Note You must define variables in the map script before you can map them.

Table 2-4 Call Subflow Properties—Output Parameter Mapping Tab

Properties / Buttons	Description
Subflow Sources	Names of the variables in the subflow which value at the end of executing the subfow must be returned to the main workflow variables specified in the “Destinations” field.
Destinations	Names of the variables in the main flow where the last value of the subflow variables specified in the “Sources” field will be stored when the subflow terminates executing. The “Destinations” subflow variable must be of the same type of the subflow variable specified in the “Subflow Sources” field or a Java base class of it.

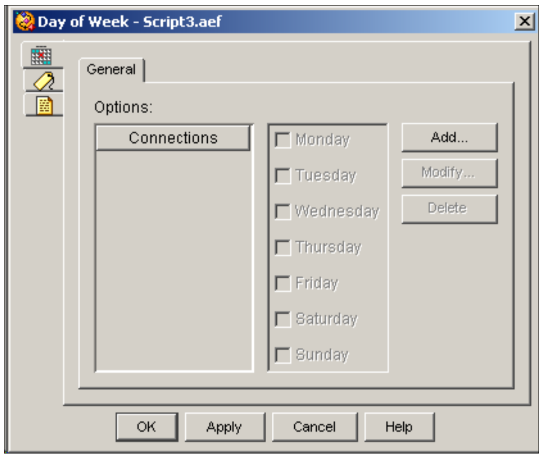
Table 2-4 Call Subflow Properties—Output Parameter Mapping Tab (continued)

Properties / Buttons	Description
Add / Modify (buttons)	Use these buttons to access the Add or Modify Output Variable dialog box where you can add or modify subflows. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove parameter mapping information, highlight a value in the list and click **Delete**.

Day of Week Step

Use the Day of Week step to cause the script to branch to different connection output branches depending on the current day of the week.

Figure 2-8 Day of Week Customizer Window



Steps that you add following a specific connection branch will execute if the application server system clock indicates that the day of the week matches one of the days associated with that connection.

You must cover all days with output branches and you must assign each day its own connection(s). If a day is not assigned to at least one output branch, the Cisco Unified CCX Editor displays a warning dialog box when you close the Day of Week customizer window.

Note Selecting all connections at a time shows, in the middle panel, all days currently selected

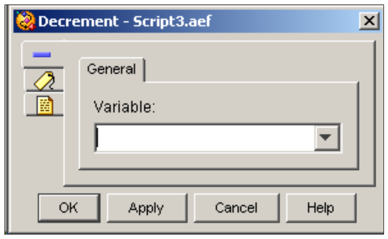
Table 2-5 describes the properties of the Day of Week customizer window.

Table 2-5 Day of Week Properties

Properties / Buttons	Description
Time Zone	You can configure the Time zone that you want the current step to use. **Note** List of time zones can be viewed from the Time zone tab in the Expression editor.
Connections	Output branches that execute depending on specified day of week **Note** Specified through the Connection Name field in the Add Connection dialog box.
(Days check boxes)	Days of the week for each connection branch
Add / Modify (buttons)	Use these buttons to access the Connection Name dialog box. Use the dialog to specify a Connection Name. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove a Connection, highlight a value in the list and click **Delete**.

Decrement Step

Use the Decrement step to decrease the value of a chosen numeric variable by one. This step is a specialized version of the Set step of the General palette, which you use to assign any value to a variable.

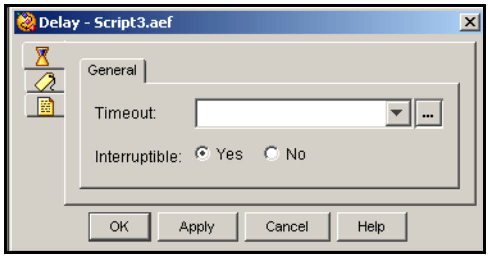


To use the Decrement customizer window to decrease the chosen numeric variable by one, choose the desired variable from the Variable drop-down menu, and then click **OK**.

Delay Step

Use the Delay step to pause the processing of a script for a specified number of seconds.

Figure 2-9 Delay Customizer Window



Interruptible: [✓ Yes] No

Table 2-6 describes the properties of the Delay customizer window

:

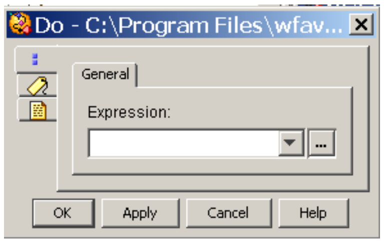
Table 2-6 Delay Properties

Property	Description
Timeout	Variable or expression indicating length of time in seconds for the delay. Enter delay time in seconds or enter an expression.
Interruptible (radio buttons)	If Yes is clicked, the delay is interruptible by external events.

Do Step

Use the Do step to execute an expression. It takes an expression as a parameter which is evaluated on execution. There is no return value.

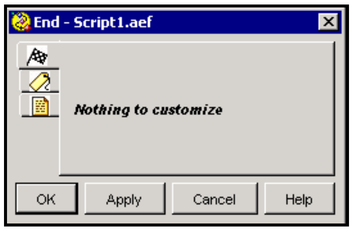
Figure 2-10 Do Step Window



End Step

Use the End step at the end of a script to complete processing and free all allocated resources.

Figure 2-11 End Customizer Window



You can also use the End step at the end of a branch of logic in a script. Any contact (call, HTTP request, or e-mail) still active by the time this step is executed will automatically be aborted and processed by the system default logic.

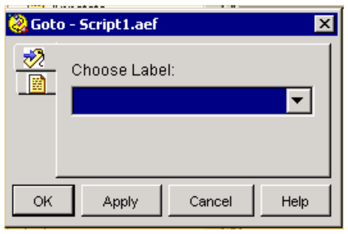
Although you do not need to define any properties for the End step, it provides customizer window so you can use the Label and Annotate tabs to add a label or comment to the step.

Goto Step

Use the Goto step to cause the script logic to branch to a specified Label within the script.

Note The Goto step customizer window does not contain a Label tab; you cannot specify the Goto step as a Label.

Figure 2-12 Goto Customizer Window



Note You must create a specific Label before you can customize the Goto step to branch to that label.

To use the Goto customizer window to cause the script logic to branch to a specific Label, choose that Label from the Select a Label drop-down menu, and then click **OK**.

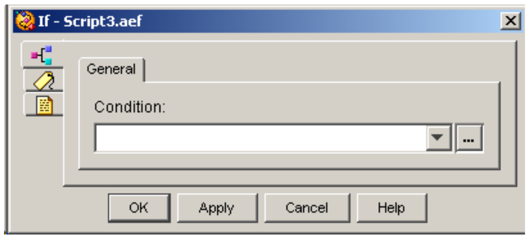
Caution Never insert a Goto step in the **middle** of a Select Resource step flow. (Doing so can cause an active agent to enter a Reserved state that can only be exited by logging out of the system.)

The Goto customizer window closes, and the label name appears next to the Goto step icon in the Design pane of the Cisco Unified CCX Editor.

If Step

Use the If step to cause the script to go to one of two branches based on the evaluation of a specified Boolean expression.

Figure 2-13 If Customizer Window



The If step automatically adds two output branches, True and False:

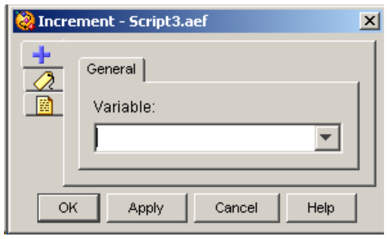
- True—Steps following this output branch execute if the expression is true.
- False—Steps following this output branch execute if the expression is false.

To use the If customizer window to create True and False output branches, either enter an expression in the text field or click the **Expression Editor** (**...**) button to enter an expression, and then click **OK**.

Increment Step

Use the Increment step to increase the value of a chosen numeric variable by one. This step is a specialized version of the Set step of the General palette, which you use to assign any value to a variable.

Figure 2-14 Increment Customizer Window

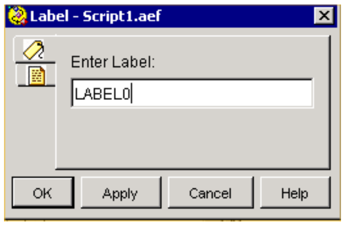


To use the Increment customizer window to increase the value of a specific Integer variable by one, choose that Integer type variable from the Variable drop-down menu, and then click **OK**.

Label Step

Use the Label step to insert a label into a script to serve as a target for a Goto or OnExceptionGoto step within the same script.

Figure 2-15 Label Customizer Window



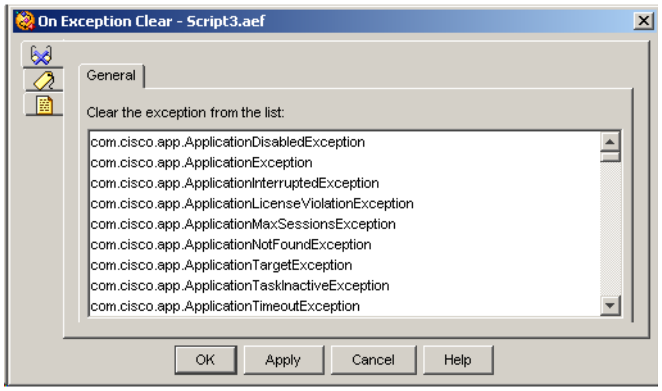
Enter Label:
LABEL0

To use the Label customizer window to insert a label into a script, enter a name in the Enter Label Name text field, and then click **OK**.

On Exception Clear Step

Use the On Exception Clear step to remove an exception set by a previous On Exception Goto step.

Figure 2-16 On Exception Clear Customizer Window



Clear the exception from the list:

```
com.cisco.app.ApplicationDisabledException
com.cisco.app.ApplicationException
com.cisco.app.ApplicationInterruptedException
com.cisco.app.ApplicationLicenseViolationException
com.cisco.app.ApplicationMaxSessionsException
com.cisco.app.ApplicationNotFoundException
com.cisco.app.ApplicationTargetException
com.cisco.app.ApplicationTaskInactiveException
com.cisco.app.ApplicationTimeoutException
```

Typically, this step is used in the following sequence:

1. An On Exception Goto step directs the script to a Label.
2. Scripting is placed after the Label to handle the exception.
3. An On Exception Clear step is then used to clear the exception.

You may also use this step when you no longer need to handle the selected exception within the script.

To use the On Exception Clear customizer window to clear an exception, select the specific exception from the list box, and then click **OK**.

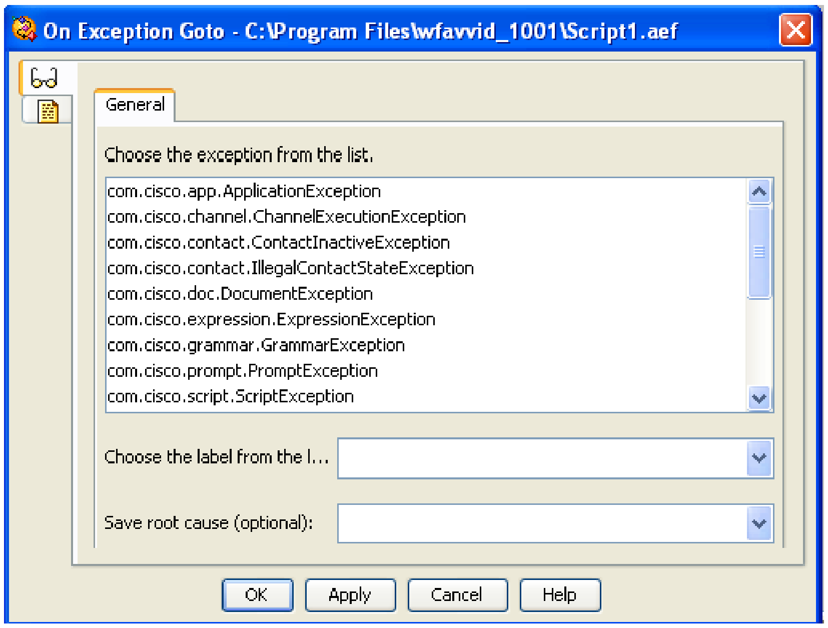
On Exception Goto Step

Use the On Exception Goto step to catch unhanded exceptions that may occur during script execution and allow a graceful exit from the situation.

Each Editor step may throw one or more of the following exceptions upon certain error conditions during script execution.

- `com.cisco.app.ApplicationException`
- `com.cisco.channel.ChannelExecutionException`
- `com.cisco.expression.ExpressionException`
- `com.cisco.grammar.GrammarException`
- `com.cisco.prompt.PromptExceptionste`
- `com.cisco.script.ScriptException`
- `com.cisco.user.UserException`
- `com.cisco.doc.DocumentException`
- `com.cisco.wf.steps.ivr.WFReportingStepException`
- `com.cisco.wf.subsystems.obj.WFInterruptedException`
- `com.cisco.wf.subsystems.obj.WFNoDayOfWeekFoundException`
- `com.cisco.wfapi.expression.WFEvaluationException`
- `com.cisco.wfapi.WFClassInvocationException`
- `com.cisco.wfapi.WFClassRemoteCreationException`
- `com.cisco.wfapi.WFWorkflowCompletedExecutionException`

Figure 2-17 On Exception Goto customizer window


On Exception Goto - C:\Program Files\wfavvid_1001\Script1.aef
General
OK
Apply
Cancel
Help

Choose the exception from the list.

com.cisco.app.ApplicationException
com.cisco.channel.ChannelExecutionException
com.cisco.contact.ContactInactiveException
com.cisco.contact.IllegalContactStateException
com.cisco.doc.DocumentException
com.cisco.expression.ExpressionException
com.cisco.grammar.GrammarException
com.cisco.prompt.PromptException
com.cisco.script.ScriptException

Choose the label from the l...

Save root cause (optional):

The following table describes the properties of the On Exception Goto customizer window:

Table 2-7 On Exception Goto properties

Property	Description
Choose the exception from the list	Exception that triggers the execution of the step.
Choose the label from the list	Label to which the script will branch.
Save root cause (optional)	Cause of the exception, which saved in an exception object. The object type must correspond to the type of exception being caught or to a base class of that exception. If it does not, no warning is generated at design time, but an error will result at run time.

Note To check which child/parent exception occurred, you must check the engine log file that corresponds to the time of the exception in the Cisco Unified CCX Engine logs that you obtained using the Unified RTMT tool.

If you want the script flow to handle the child exception separately, add it as an additional On Exception Goto step after the step that catches the parent exception in the Editor Script workflow.

For example:

- If you want the script flow to handle `com.cisco.contact.ContactInactiveException,` use it after its parent exception, `ApplicationException`. If not, the `ApplicationException` handles the exception and moves the script flow to ApplicationException label.

- To handle `com.cisco.app.ApplicationDisabledException`, use its parent exception, `com.cisco.app.ApplicationException` in the On Exception Goto Step.

Parent Exceptions and corresponding Child Exceptions

The list below describes all the parent exceptions and their corresponding child exceptions. Use it to find the parent of particular child exception thrown.

- `com.cisco.app.ApplicationException`

  The ApplicationException class defines all exceptions that can be used to interrupt an application task. It is base class for the following exceptions and handles these exceptions if it is used in On Exception Goto Step.

Note If both parent and child exceptions are handled in the On Exception Goto step, for example, if the child exception (`com.cisco.prompt.PromptException`) needs to be handled after the parent exception, the latest handler takes preference in the hierarchy to prevent the exception from being missed. This handling is contrary to normal Java exception hierarchy handling (that is, a base class exception must be handled last, after all child class exceptions).

  - `com.cisco.app.ApplicationDisabledException`

    The ApplicationDisabledException class indicates that a task is requested for an application that is currently disabled.
  - `com.cisco.app.ApplicationInterruptedException`

    The ApplicationInterruptedException class indicates that the task thread was interrupted using an InterruptedException.
  - `com.cisco.app.ApplicationMaxSessionsException`

    The ApplicationMaxSessionsException class indicates that a task is requested for an application that has reached its configured maximum number of concurrent sessions.
  - `com.cisco.app.ApplicationNotFoundException`

    The ApplicationNotFoundException class indicates that the configuration of an application is no longer defined.
  - `com.cisco.app.ApplicationTargetException`

    The ApplicationTargetException class extends the ApplicationException. It is a checked exception which wraps an exception thrown by a task when that exception is not an instance of the ApplicationException.
  - `com.cisco.app.ApplicationTaskInactiveException`

    The ApplicationTaskInactiveException class indicates that a task is either no longer active or has ended before the end of an operation.
  - `com.cisco.app.ApplicationTimeoutException`

    The ApplicationTimeoutException class indicates that a task cannot be invoked in the maximum time requested.
  - `com.cisco.app.impl.AppInterruptedException`

    AppInterruptedException is caused due to an interruptible action or step.
  - `com.cisco.app.InvalidApplicationException`

The Invalid ApplicationException class indicates that the configuration of an application is no longer valid. This parent exception is thrown by the following child exception: `com.cisco.app.ApplicationLicenseViolationException.`

- `com.cisco.channel.ChannelExecutionException`

  The Channel Execution Exception class indicates that the channel of a contact or task fails to execute the requested operation. This is the parent class for following exceptions:

  - `com.cisco.channel.ChannelAbortedException`
  - `com.cisco.channel.ChannelInactiveException`
  - `com.cisco.channel.ChannelLicenseViolationException`
  - `com.cisco.channel.ChannelTimeoutException`
  - `com.cisco.channel.ChannelUnsupportedException`
  - `com.cisco.channel.DuplicateChannelException`
  - `com.cisco.channel.IllegalGroupStateException`
  - `com.cisco.contact.ContactInactiveException`
  - `com.cisco.contact.IllegalContactStateException`
  - `com.cisco.dialog.DialogASRException`
  - `com.cisco.wf.dialog.InvalidAudioFormatException`
  - `com.cisco.channel.MissingChannelException`

  The Contact Inactive exception belongs to this group of exceptions. It is one of the most common exceptions that occurs in Unified CCX. This exception is thrown when a script attempts to execute a step that is dependent on an active contact, after the contact is terminated.

- `com.cisco.expression.ExpressionException`

  The Expression objects throw the Expression Exception class when they are unable to parse or evaluate the expression.

  The following child exceptions throw this parent exception:

  - `com.cisco.expression.ExpressionAuthorizationException`
  - `com.cisco.expression.ExpressionEvaluationException`
  - `com.cisco.expression.ExpressionLexicalException`
  - `com.cisco.expression.ExpressionLicenseViolationException`
  - `com.cisco.expression.ExpressionParsingException,com`
  - `cisco.expression.ExpressionRemoteException`
  - `com.cisco.expression.ExpressionSemanticException`

- `com.cisco.grammar.GrammarException`

  The Grammar objects throw Grammar Exception class when they are unable to resolve themselves. This class is a base class for the following exceptions:

  - `com.cisco.grammar.CircularGrammarException`
  - `com.cisco.grammar.GrammarLexicalException`
  - `com.cisco.grammar.GrammarMatchingException`
  - `com.cisco.grammar.GrammarParsingException`
  - `com.cisco.grammar.GrammarSyntaxException`
  - `com.cisco.grammar.UndefinedGrammarException`
  - `com.cisco.grammar.UnknownGrammarException`
  - `com.cisco.grammar.UnresolvedRuleException`

   - `com.cisco.grammar.UnsupportedGrammarException`
   - `com.cisco.grammar.UnsupportedPartialMatchingException`
   - `com.cisco.wf.subsystems.ged125.ParameterGrammarException`
- `com.cisco.prompt.PromptException`

  The Playable or PromptQueue objects throw PromptException class when they are unable to append prompts to a prompt queue. This class is a base class for the following exeptions:
   - `com.cisco.prompt.CircularPromptException`
   - `com.cisco.prompt.InvalidPromptArgumentException`
   - `com.cisco.prompt.PromptLexicalException`
   - `com.cisco.prompt.PromptParsingException`
   - `com.cisco.prompt.TTSDuplicateProviderException`
   - `com.cisco.prompt.TTSPromptProviderException`
   - `com.cisco.prompt.UndefinedPromptException`
   - `com.cisco.prompt.UndefinedPromptGeneratorException`
   - `com.cisco.prompt.UnknownPromptException`
   - `com.cisco.prompt.UnsupportedPromptException`
   - `com.cisco.prompt.UnsupportedTTSPromptException`
   - `com.cisco.wf.subsystems.ged125.UnsupportedMediaProtocolPromptException`
   - `com.cisco.wf.subsystems.ged125.UnsupportedMediaTypePromptException`
- `com.cisco.script.ScriptException`

  The ScriptException class is thrown when a script operations fails. This class is a base class exception for the following exceptions:
   - `com.cisco.script.InvalidScriptException`
   - `com.cisco.script.ScriptIncompatibleTypeException`
   - `com.cisco.script.ScriptIOException`
   - `com.cisco.script.ScriptPreprocessException`
   - `com.cisco.script.ScriptNotFoundException`
- `com.cisco.user.UserException`

  The UserException class represents the base class of all exceptions that can be thrown by user operations.
- `com.cisco.doc.DocumentException`

  DocumentException defines all exceptions that can be generated when you work with documents steps. It is the parent class for the following exceptions:
   - `com.cisco.doc.DocumentIOException,`
   - `com.cisco.doc.DocumentNotFoundException`
- `com.cisco.wf.steps.ivr.WFReportingStepException`

  WFReportingStepException occurs if a selected reporting object does not exist while executing Get Reporting Statistic Step.
- `com.cisco.wf.subsystems.obj.WFInterruptedException`

  WFInterruptedException indicates that an operation was interrupted.
- `com.cisco.wf.subsystems.obj.WFNoDayOfWeekFoundException`

WFNoDayOfWeekFoundException class exception is thrown if the engine could not find a range of days that contain the current day while executing DayOfWeekStep.

- `com.cisco.wfapi.expression.WFEvaluationException`

  WFEvaluationException exception is thrown if a run-time error occurs while evaluating the expression.
- `com.cisco.wfapi.WFClassInvocationException`

  WFClassInvocationException exception class is thrown if Class Invocation error occurs.
- `com.cisco.wfapi.WFClassRemoteCreationException`

  WFClassRemoteCreationException exception class occurs during the execution of Create Remote Java Object step.

Note Although the Create Remote Java Object step is no longer exposed in the Editor step palette, legacy scripts that were migrated to later releases may still include the step, so this exception can still be thrown.

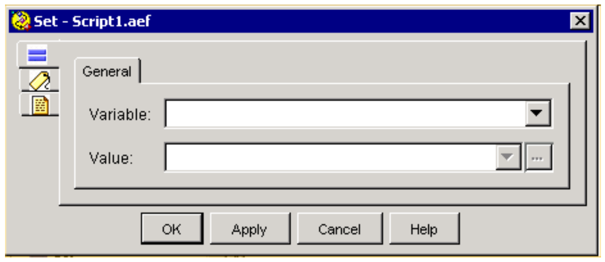
- `com.cisco.wfapi.WFWorkflowCompletedExecutionException`

  WFWorkflowCompletedExecutionException class indicates a normal exit. If an exception occurs in the middle of execution, the engine checks that the exception is an instance of WFWorkflowCompletedExecutionException and it ends the task.

Set Step

Use the Set step to change the value of a variable.

Figure 2-18 Set Customizer Window



The Set step supports type casting (with possible loss of precision) from any Number data type (Integer, Float, Long, Double, BigInteger, BigDecimal) to any other Number data type.

You can also use the Set step to convert a String variable to any Number data type. For String conversions, the system replaces all “*” characters with a decimal point (“.”) before performing the conversion.

The Set step also supports the following conversions:

- Time expressions to string with selection of the type of encoding between Short, Medium, Long and Full format.
- Date expressions to string with selection of the type of encoding between Short Date, Medium Date, Long Date, Full Date, Short Time, Medium Time, Long Time and Full Time format.
- Any other type to a string

- String expression to a Prompt by using the string to represent the user prompt name.
- A string expression to a Date by parsing the string
- A string expression to a Time by parsing the string
- A string expression to a Boolean by accepting yes, true, y, t, or 1 in any case as true values and no, false, n, f, 0 as false values.
- A document expression into a prompt by using the document content as if it was an audio document
- A prompt expression into a document by processing the prompt and storing its content into the specified variable of type document.

Note For more information on type casting and variables, see *Cisco Unified CCX Scripting and Development Series: Volume 3, Expression Language Reference.*

Table 2-8 describes the properties of the Set customizer window:

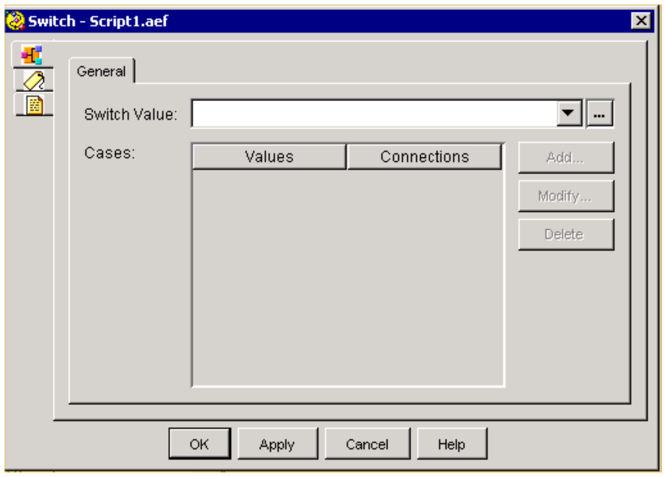
Table 2-8 Set Properties

Property	Description
Variable	Variable for which the value will be set.
Value	Value for the specified variable.

Switch Step

Use the Switch step to cause the program logic to branch to one of a number of cases based on the evaluation of a specified expression.

Figure 2-19 Switch Customizer Window



A *case* is a method for providing script logic based on the value of a variable at a point in time. You can assign one case for each value. The Switch step lets you define any number of case output branches. You can then create a separate script logic for each branch.

Note Use a subflow in a script (instead of a redirect) to transfer a call to another script.

The Switch step supports switching based on the following variables:

- Integer—Comparison of integers
- String—Comparison of string variables (case insensitive)
- Language—Comparison of language variables

Note The language comparison will attempt the parent language before falling back to default. For example, a match for L[en_US_Judy] will first attempt to match from the following list: {L[en_US_Judy], L[en_US], L[en]}, and then return "Default" if no item in the list matched.

The type of switching is automatically determined by the type of the specified expression.

Steps that you add following a specific case output branch execute if the integer, string, or language expression you specify for that case is equal to the global expression defined in the Switch Expression field.

The Default branch of the step allows you to handle cases where none of the branches matches the expression.

Table 2-9 describes the properties of the Switch customizer window:

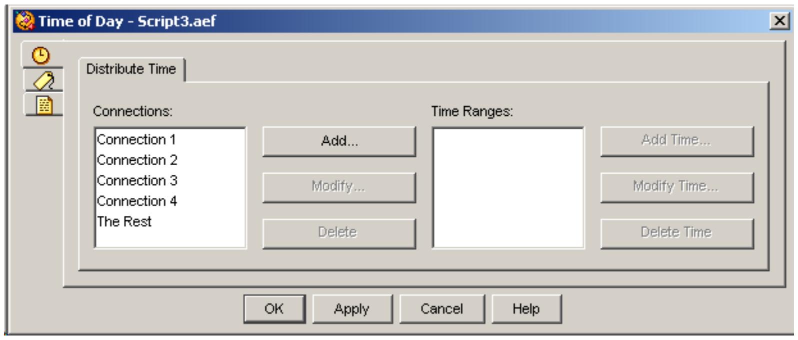
Table 2-9 Switch Properties

Properties / Buttons	Description
Switch Value	Variable or expression to be executed.
Values	The list of switch values Language case for the output branch containing script logic specific to one possible variable value.
Connections	Output label targets to which the script branches when the variable equals the specific values.
Add / Modify (buttons)	Use these buttons to access the Switch Case and Label dialog box. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove a Switch Case, highlight a value in the list and click **Delete**.

Time of Day Step

Use the Time of Day step to cause the script to branch to different connection branches depending on the current time of day.

Figure 2-20 Time of Day Customizer Window



Steps that you add following a specific output branch will execute if the Cisco Unified CCX Engine clock indicates that the time of day matches the time associated with that connection.

You associate each output branch with a specified range of time.

During run time, if the current time falls out of the configured time range, the script follows the Rest output branch of the Time of Day step.

Note When you debug this step on a remote workstation, the Cisco Unified CCX Editor uses the time zone of the Cisco Unified CCX server where the script is being debugged.

Table 2-10 describes the properties of the Time of Day customizer window.

Table 2-10 Time of Day Properties

Property	Description
Time Zone	You can configure the Time zone that you want the current step to use. **Note** List of time zones can be viewed from the Time zone tab in the Expression editor.
Connections	Output branches that execute depending on specified time of day. **Note** Specified through the Connection Name field in the Add Connection Name dialog box.
(Connection) Add / Modify (buttons)	Use these buttons to access the Connection name dialog box. Use the dialog to specify a Connection Name. When done, click **OK**.
Delete (button)	To remove a Connection Name, highlight a value in the list and click **Delete**.
Time Ranges	Time ranges for each connection branch.

Table 2-10 Time of Day Properties

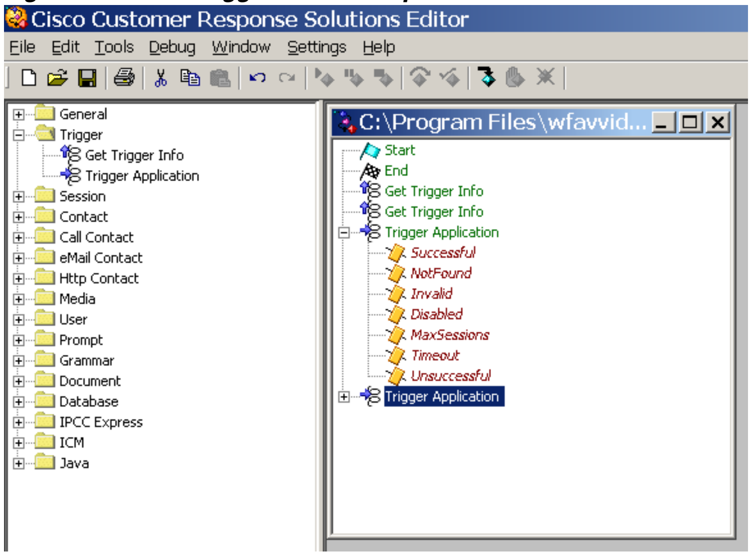
Property	Description
Add Time / Modify Time (buttons)	Use these buttons to access the Range of Hours dialog box. Use the dialog to specify a Start Time and End Time from the lists.
Delete Time (button)	To remove a Time, highlight a value in the list and click **Delete**.

Trigger Steps

The figure that follows shows the steps in the Trigger palette as they appear in the Palette pane of the Cisco Unified CCX Editor. This section contains the following topics:

- Get Trigger Info Step
- Trigger Application Step

Figure 2-21 Trigger Palette Steps

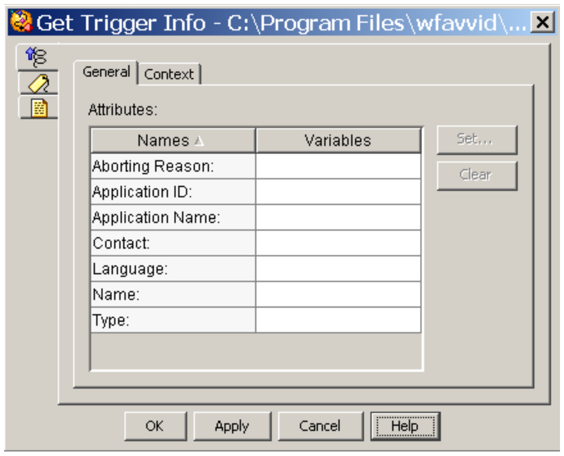


Get Trigger Info Step

Use the Get Trigger Info step to retrieve a reference to the triggering contact and store it in a script variable. The trigger contact is the call or HTTP request the system received that triggered the script.

Storing a triggering contact in a variable lets you then reference that contact directly or pass it into a subflow.

Figure 2-22 Get Trigger Info Customizer Window - General Tab


Get Trigger Info - C:\Program Files\wfavvid\...

Attributes:

Names	Variables
Aborting Reason:	
Application ID:	
Application Name:	
Contact:	
Language:	
Name:	
Type:	

Table 2-11 describes the properties of the Get Trigger Info customizer window.

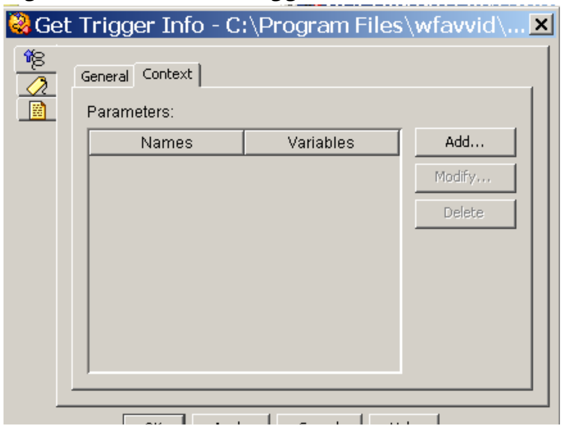
Table 2-11 Get Trigger Info Attributes and Values

Attributes / Buttons		Description
Attributes		Attributes and values of trigger information types.
	Aborting Reason	When the Get Trigger Info step is in a default script, specifying an Aborted Reason variable lets you access the exception reason generated when the normal script aborted. The exception reason can provide valuable information about why there was a script failure. For example, the value com.cisco.app.ApplicationFailoverException indicates that the Cisco Unified CCX Engine where the main script was running stepped down as the Master. Knowing this, you can then restructure your normal script to compensate for this happening in the future by transferring the call back to the DN where it was received so it can be automatically transferred to the new Cisco Unified CCX Engine master. **Note** The Aborted Reason variable must be of the same type as the expected exception or a base class.
	Contact	Contact object that triggered this script. If no contact triggered this script (for example, when a designer is debugging a script), the returned value is null.
	Trigger Name	Unique name of the trigger used when configuring it in the Cisco Unified CCX AppAdmin. This is the CTI route point # for JTAPI Triggers and the URL for HTTP triggers.
	Trigger Type	This is the type of the trigger: Possible types are Cisco Http Trigger, Cisco JTAPI Trigger, and Remote Debugging Trigger.
	Application Name	The unique name of the application associated with the trigger.
	Application ID	The application identifier configured with the associated application.
	Language	The language configured with the triggering.

Table 2-11 Get Trigger Info Attributes and Values

Attributes / Buttons		Description
Buttons		
	Set	To set a variable, select an Attribute name and click **Set**. Choose a variable from the Variable drop-down list and then click **OK**; the variable name appears in the Variable column next to the attribute you selected.
	Clear	To remove Attribute information, highlight a value in the list and click **Clear**.

Figure 2-23 Get Trigger Info Customizer Window - Contetxt Tab

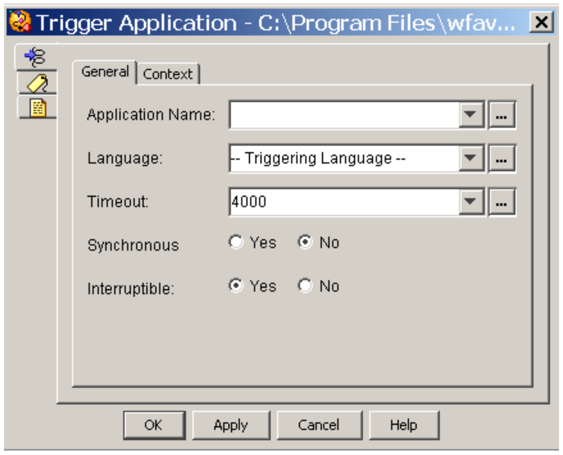


You can use the Context tab to get the trigger parameters for use in other steps in the script.

Trigger Application Step

Use the Trigger Application step to trigger a specific application.

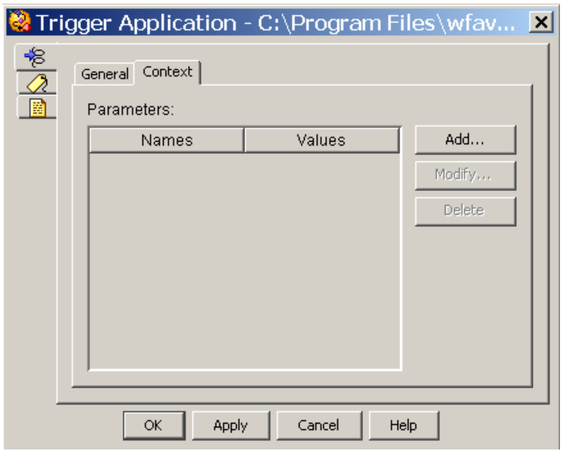
Figure 2-24 Trigger Application Window - General Tab



Application Name:
Language: -- Triggering Language --
Timeout: 4000
Synchronous Yes [✓ No]
Interruptible: [✓ Yes] No

Table 2-12 describes the General Tab properties of the Trigger Application window

Figure 2-25 Trigger Application Window - Context Tab



Context parameters are variables or expressions that are stored inside the trigger. These are retrievable in the script that runs for the application using the Get Trigger Info Step and the context parameter tab.

Session Steps

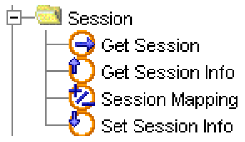
The steps in the Session palette of the Cisco Unified CCX Editor provide script designers with an easy way to associate information with a call contact, an e-mail contact, or a HTTP contact as the contact moves through the system. Session steps provide functionality similar to that of an in-memory database or a shopping cart on the web.

This section contains the following topics:

- Session Overview
- Get Session Step
- Get Session Info Step
- Session Mapping Step
- Set Session Info Step

The figure that follows shows the steps in the Session palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Figure 2-26 Session Palette Steps



Note For an example of a script that demonstrates Session steps, see Chapter 14, "Designing Cisco Unified CCX Scripts," in the *Cisco Unified Cotnact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts*.

Session Overview

The system automatically associates a contact with a session when the contact is received (inbound) or initiated (outbound). You can use the Get Session step to create sessions manually; you may want to do so when you want to use sessions for HTTP or e-mail contacts.

Customer information stored in a session object can persist for a specified length of time after the contact ends and can be made available to a subsequent contact. This feature can save customers the need to re-enter information such as credit card account digits.

You can store any type of information in these session objects and use the Set Session Info and Get Session Info steps to retrieve it.

Examples of session events include

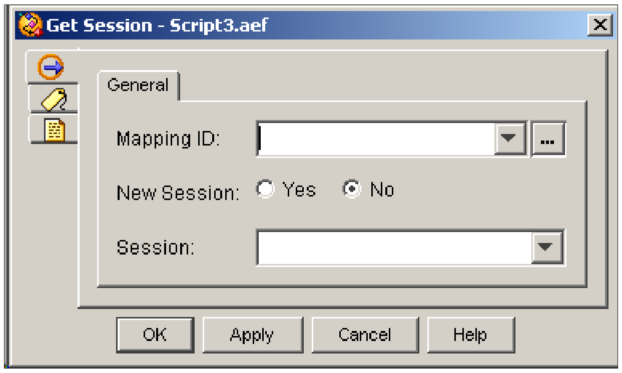
- A call contact connected to an agent
- A call contact redirected to another agent or back to an Interactive Voice Response (IVR) application.
- A call contact redirected from one application to another application on the same IVR server

Note A session resides on one Cisco Unified CCX server only; it does not follow a contact that is directed to a different Cisco Unified CCX server.

Get Session Step

Use the Get Session step to get session information based on a mapping identifier.

Figure 2-27 Get Session Customizer Window



Mapping ID:
New Session: Yes [✓ No]
Session:

You can also use the Get Session step to create a new session and automatically associate a specified mapping identifier with the new session.

If no session exists with the given mapping identifier, the system creates a new one only if the New Session flag is set to Yes. Otherwise, the system returns null.

Note The system automatically deletes a newly created session if the session becomes idle for 30 minutes (or for whatever value has been defined for a session timeout on the System Parameters page of the Cisco Unified CCX Administration web interface.)

Table 2-13 describes the properties of the Get Session customizer window.

Table 2-13 Get Session Customizer Window Descriptions

Properties / Buttons	Description
Mapping ID	Mapping identifier variable for the desired session
New Session	Radio button. If Yes, indicates that a new session should be created if none can be found in the system pool of sessions
Session	Variable name where the session will be returned (or null)

Get Session Info Step

Use the Get Session Info step to retrieve the properties associated with a session and make them available to the script by storing their values in specified variables.

The Get Session Info customizer window contains two tabs:

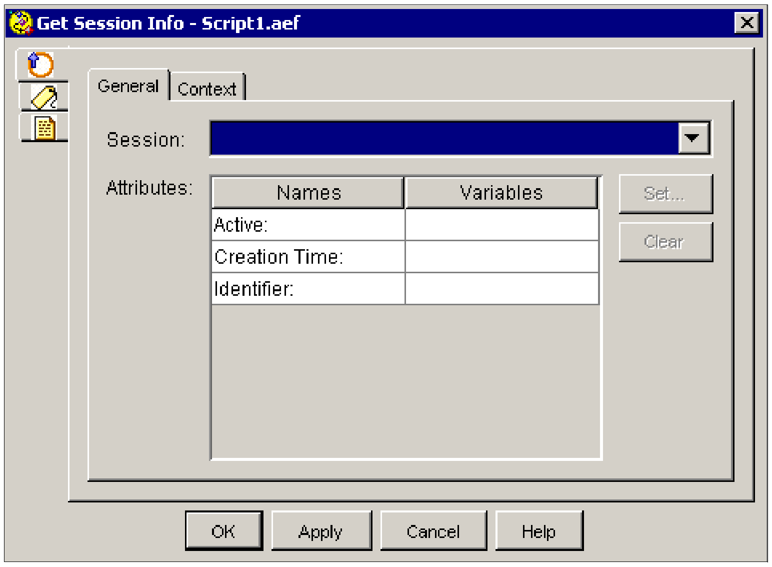
- General tab (Get Session Info step)
- Context tab (Get Session Info step)

The following sections describe these tabs.

General tab (Get Session Info step)

Use the General tab of the Get Session Info customizer window to get variable values from attributes.

Figure 2-28 Get Session Info Customizer Window—General Tab



Names	Variables
Active:	
Creation Time:	
Identifier:	

Table 2-14 describes the properties of the General tab of the Get Session Info customizer window.

Table 2-14 Get Session Info Properties—General Tab

Property / Buttons	Description
Session	Variable in which the session information is stored.
Attributes Names/Variables	Attributes and variables associated with the session. The following attributes are displayed in the Attribute column: • **Active**—Boolean flag that indicates if the session is still active; a session is no longer active when it is deleted by the system. • **Creation Time**—Date object that indicates the time the session was first created in the system. • **Identifier**—The unique identifier of the session as reported in CCDR.
Set (button)	To set Session information, select an Attribute name and click **Set**. Choose a variable from the Select Variable drop-down list and then click OK; the variable name appears in the Variable column next to the attribute you selected.
Clear (button)	To remove Session information, highlight a value in the list and click **Clear**.

Context tab (Get Session Info step)

Use the Context tab of the Get Session Info customizer window to add attributes associate variables and context parameters (name-value pairs). Context parameters are variables or expressions that are stored inside the trigger. These are retrievable in the script that runs for the application using the GetTriggerInfo step and the context parameter tab.

Figure 2-29 Get Session Info Customizer Window—Context Tab

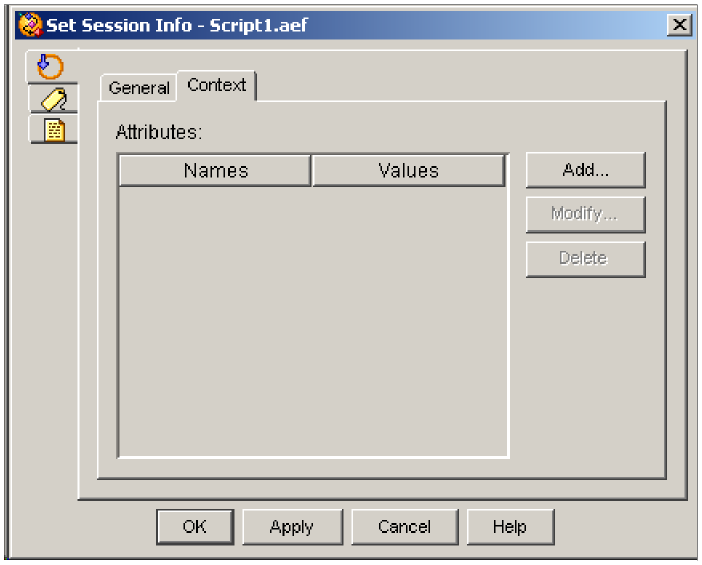


Table 2-15 describes the properties of the Context tab of the Get Session Info customizer window:

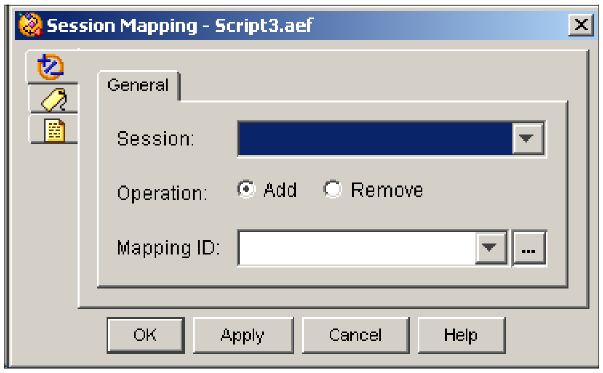
Table 2-15 Get Session Info Properties—Context Tab

Properties / Buttons	Description
Attributes Names	List of attributes for the specified session
Values	Values associated with the attributes
Set (button)	To set a value, select an Attribute name and click **Set**. The Get Set Active dialog box opens. Choose a value from the Select Variable drop-down list and then click **OK**; the variable name appears in the Variable column next to the attribute you selected.
Delete (button)	To remove Get Session information, highlight a value in the list and click **Delete**.

Session Mapping Step

Use the Session Mapping step to add or remove mapping identifiers associated with a particular session.

Figure 2-30 Session Mapping Customizer Window



Session:
Operation: [✓ Add] Remove
Mapping ID:

Note Never use a script to removed mapping identifiers that the system has automatically added to a session.

Table 2-16 describes the properties of the Session Mapping customizer window:

Table 2-16 Session Mapping Properties Session Mapping Customizer Window Descriptions

Properties / Buttons	Description
Session	Variable indicating the session for which you want to add or remove the mapping identifier.
Operation	Radio buttons. If Add, adds a mapping identifier. If Remove, removes a mapping identifier.
Mapping ID	Variable indicating the mapping identifier to be added or removed.

Set Session Info Step

Use the Set Session Info step to add or modify the context information for a session.

The Set Session Info customizer window contains two tabs:

- General tab (Set Session Info step)
- Context tab (Set Session Info step)

The following sections describe these tabs.

General tab (Set Session Info step)

Use the General tab of the Set Session Info customizer window to specify the Session variable for which you want to add or modify context information.

Figure 2-31 Set Session Info Customizer Window—General Tab

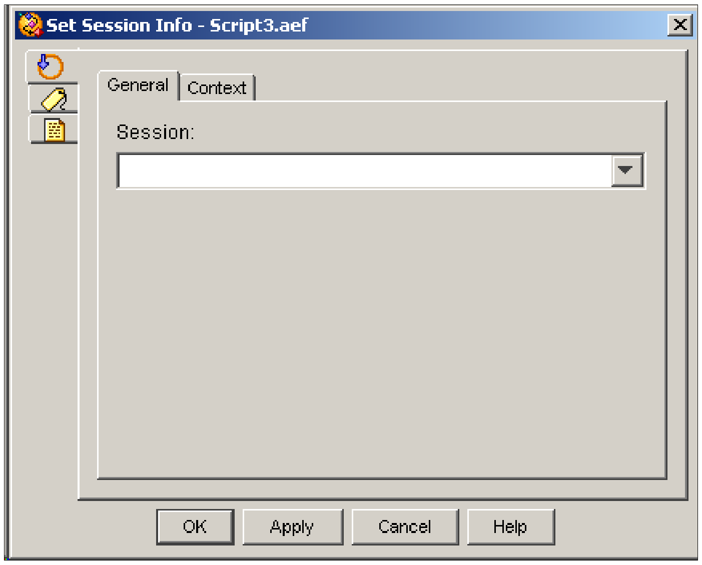


Table 2-17 describes the property of the General tab of the Set Session Info customizer window.

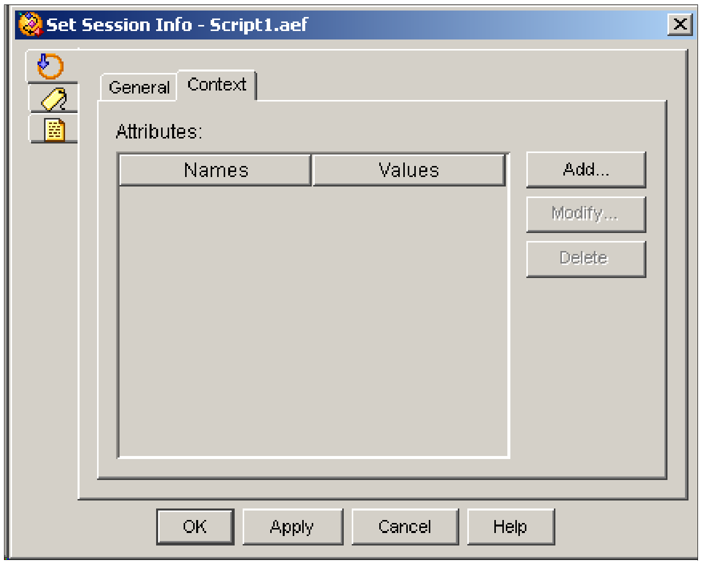
Table 2-17 Set Session Info Property—General TabSet Session Info Customizer Window Descriptions

Property	Description
Session	Variable indicating session for which you want to add or modify context information.

Context tab (Set Session Info step)

Use the Context tab of the Set Session Info customizer window to add or modify the value of attribute variables.

Figure 2-32 Set Session Info Customizer Window—Context Tab



As an example, you can use a Get Contact Info step to assign a Session attribute to a variable of type Session. Then use a Set Session Info step to use that Session variable and assign the desired value(s) to one or more of the context variables.These variables will be stored in the session object.

Table 2-18 describes the properties of the Context tab of the Set Session Info customizer window.

Table 2-18 Set Session Info Property—Context Tab Set Session Inf Customizer Window Descriptions

Property	Description
Attributes Names/Values	Attributes and associated values for the session identified in the General tab
Add / Modify (buttons)	Use these buttons to access the Set Context Attribute dialog box. Use the dialog to specify the following: • **Attribute**—Drop-down list of available context variables. • **Value**—Boolean expression or variable from the drop-down list, or an expression. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove the Attribute information, highlight a value in the list and click **Delete**.

Contact Steps

The steps in the Contact palette of the Cisco Unified CCX Editor provide designers with a way to control contacts.

A *contact* represents a specific interaction with a customer. The three types of contacts are

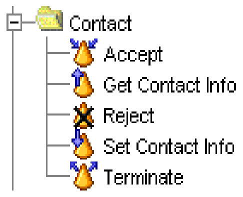
- telephone call
- e-mail message
- HTTP request

This section describes the following:

- Accept Step
- Get Contact Info Step
- Reject Step
- Set Contact Info Step
- Terminate Step

Figure 2-33 shows the steps in the Contact palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

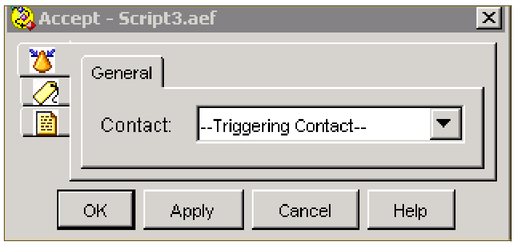
Figure 2-33 Contact Palette Steps



Accept Step

Use the Accept step to accept a particular contact. A contact can be a telephone call, an e-mail message, or an HTTP request.

Figure 2-34 Accept Customizer Window


Accept - Script3.aef

General

Contact: --Triggering Contact--

After the Start step, the Accept step is normally the first step in a Cisco script, triggered by an incoming contact.

If the contact is a call, the caller hears ringing until the script reaches this step. If the contact is an HTTP request or an e-mail message, this step has no effect.

Table 2-19 describes the property of the Accept customizer window.

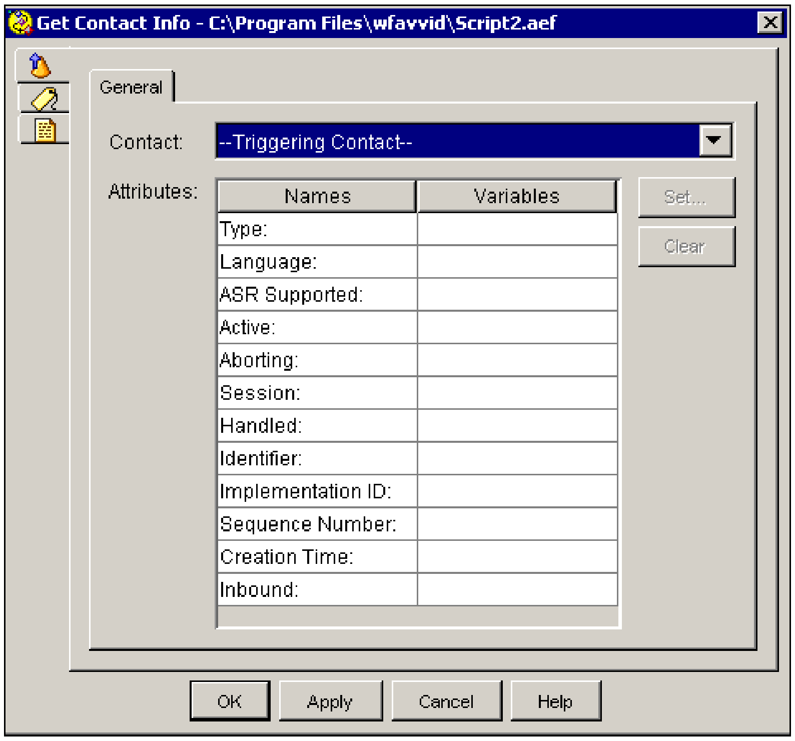
Table 2-19 Accept Property

Property	Description
Contact	Contact variable. Default is Triggering Contact (whichever contact triggers the execution of the script).

Get Contact Info Step

Use the Get Contact Info step to extract information from a particular type of object and store it in script variables to make this information about the contact available to subsequent steps in the script.

Figure 2-35 Get Contact Info Customizer Window



Get Contact Info - C:\Program Files\wfavvid\Script2.aef

General

Contact: --Triggering Contact--

Attributes:

Names	Variables
Type:	
Language:	
ASR Supported:	
Active:	
Aborting:	
Session:	
Handled:	
Identifier:	
Implementation ID:	
Sequence Number:	
Creation Time:	
Inbound:	

For example, you can define a conditional prompt to use the condition expression “asr” to determine which prompt to create based on whether or not Automatic Speech Recognition (ASR) is currently supported for the call.

Table 2-20 describes the properties of the Get Contact Info customizer window.

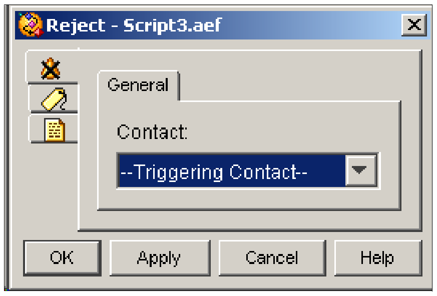
Table 2-20 Get Contact Info Properties

Property		Description
Contact		Contact variable for which you want to get information. Default is Triggering Contact, unless another contact is defined.
Attribute/Variable		Attributes and values of contact information types.
	Type	String representing the type of contact: call, e-mail, or HTTP.
	Language	Language object corresponding to the first language defined in the language context of the contact.
	ASR Supported	Boolean value indicating whether Automatic Speech Recognition is currently supported for the call.
	Active	Boolean value indicating whether the call is still active.
	Aborting	Boolean value indicating whether the call is being aborted.
	Session	A unique identifier for the contact session.
	Handled	Boolean value indicating whether the contact was previously marked as handled.
	Identifier	Integer value containing the contact identifier assigned by the system guaranteed to be unique among all contacts.
	Implementation ID	String value containing the implementation-specific identifier for the contact. This value is unique for a given contact type. For a Cisco JTAPI call contact, this value is equivalent to the global call identifier obtained by the Cisco Unified Communications Manager.
	Sequence Number	Integer value containing the sequence number of the contact assigned by the system if the contact is associated with a session. The value is -1 if the contact is not associated with a session. For every new contact associated with a session, the system increments the value by one.
	Creation Time	The time the contact was initiated.
	Inbound	Inbound or initiated (outbound).
Buttons		
	Set	To set a variable, select an Attribute name and click **Set**. Choose a variable from the Select Variable drop-down list and then click **OK**; the variable name appears in the Variable column next to the attribute you selected.
	Clear	To remove Attribute information, highlight a value in the list and click **Clear**.

Reject Step

Use the Reject step to reject a particular contact if it has not already been accepted.

Figure 2-36 Reject Customizer Window



Contact: --Triggering Contact--

The Reject step handles different types of calls in different ways:

- If the contact is an HTTP request, a FORBIDDEN (403) error is returned to the browser.
- If the contact is an e-mail message, the state of the e-mail is moved to a Rejected state, thus preventing the e-mail from being sent later.

Table 2-21 describes the property of the Reject customizer window.

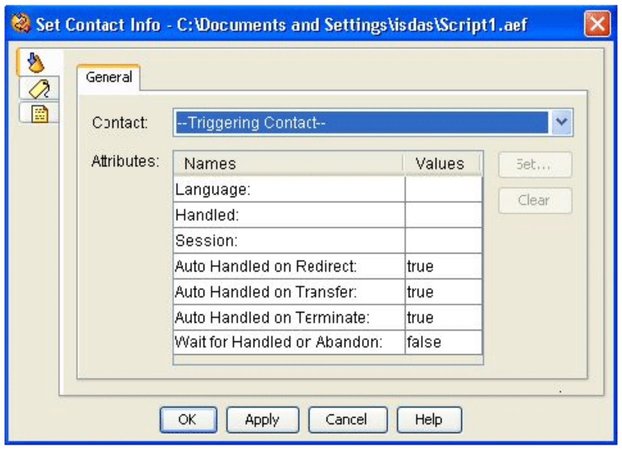
Table 2-21 Reject Property

Property	Description
Contact	Contact variable. Default is Triggering Contact, unless another contact is defined.

Set Contact Info Step

Use the Set Contact Info step to modify the context information associated with a contact. You can use this step at the begining of the script to mark the contact as handled in the following scenarios: redirect, transfer, terminate, and abandon. The default value for all of these attributes is true.

Figure 2-37 Set Contact Info Customizer Window


Set Contact Info - C:\Documents and Settings\isdas\Script1.aef
General
Set...
Clear
OK
Apply
Cancel
Help

Contact: --Triggering Contact--

Attributes:

Names	Values
Language:	
Handled:	
Session:	
Auto Handled on Redirect:	true
Auto Handled on Transfer:	true
Auto Handled on Terminate:	true
Wait for Handled or Abandon:	false

The Set Contact Info step often follows a Redirect step in the script, in order to mark the contact as Handled.

A contact can be marked Handled only while it is active. Once a contact becomes inactive (for example, after a successful transfer), the script has a maximum of 5 seconds to mark the contact as Handled; otherwise the mark will have no effect in reporting.

The system automatically marks a contact as Handled when the contact is connected to a Cisco Unified CCX or Intelligent Contact Management Enterprise (ICME) agent.

Note You cannot mark a contact as *unhandled*. Once a contact is reported as Handled, it will always be reported as such.

Table 2-22 describes the properties of the Set Contact Info customizer window.

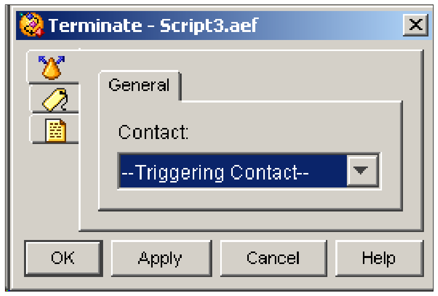
Table 2-22 Set Contact Info Properties

Property	Description
Contact	Contact variable for which you want to set information. Default is Triggering Contact, unless another contact is defined.
Attributes/Names and Values	Attribute names and values of contact information types. Valid choices are: • **Handled**—Final result of contact; this is important for reporting purposes • **Language**—Language that is used to retrieve prompts and/or grammars when interacting with the contact. • **Session**—Specified contacts can be associated with a different session object. • **Auto Handled on Redirect**—It indicates whether to mark the contact as handled in case of redirected call without waiting for it to be marked as handled from Set Contact Info Step. • **Auto Handled on Transfer**—It indicates whether to mark the contact as handled in case of transferred call without waiting for it to be marked as handled from Set Contact Info Step. • **Auto Handled on Terminate**—It indicates whether to mark the contact as handled in case of terminated call without waiting for it to be marked as handled from Set Contact Info Step. • **Wait for Handled on Abandon** —It indicates whether to wait for the handled event before posting "CONTACT_TERMINATED" event for the Contact.
Set (button)	Select an attribute and click **Set**. The following happens: • If you selected **Handled**, an X appears in the Value column. • If you select **Language** or **Session**, the Set Contact dialog box appears. Use the Select Variable drop-down menu or Expression Editor to choose a desired variable and click **OK**.
Clear (button)	To remove Attribute information, highlight a value in the list and click **Clear**.

Terminate Step

Use the Terminate step to terminate a contact.

Figure 2-38 Terminate Customizer Window



Contact: --Triggering Contact--

The Terminate step performs the following functions for the three Contact types:

- Call contact—The call is disconnected.
- HTTP contact—A NO_CONTENT (204) status message is returned to the browser.
- E-mail contact—The state of the e-mail is moved to a Terminated state, thus

Table 2-23 describes the property of the Terminate customizer window.

Table 2-23 Terminate Property

Property	Description
Contact	Contact variable that you want to be terminated. Default is Triggering Contact, unless another contact is defined.

Call Contact Steps

The steps in the Call Contact palette of the Cisco Unified CCX Editor provide script designers with a way to manage calls.

Note If you apply a Call Contact step to a contact that is not a call contact, a ChannelUnsupportedException results.

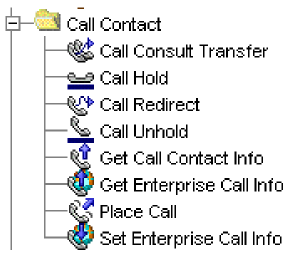
This section contains the following topics:

- Call Consult Transfer
- Call Hold Step
- Call Redirect Step
- Call Unhold Step
- Get Enterprise Call Info Step

- Get Call Contact Info Step
- Place Call Step
- Set Enterprise Call Info Step

Figure 2-39 shows the steps in the Call Contact palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Figure 2-39 Call Contact Palette Steps



Call Consult Transfer

Use the Call Consult Transfer step to perform a supervised transfer. The step can be used to outpulse DTMF digits once the destination answers. The transfer will be completed on the destination answering the call and DTMF outpulsing complete (if DTMF digits are specified).

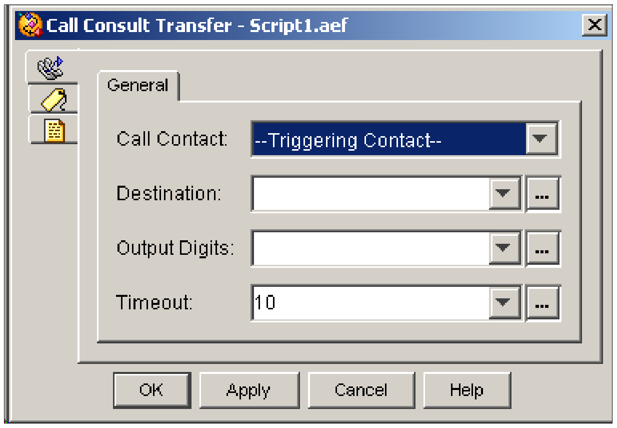
The Call Consult Transfer produces five output branches:

- Successful—The call is ringing at the specified extension.
- Busy—The specified extension is busy and the call cannot be transferred.
- Invalid—The specified extension does not exist.
- Timeout—When the call to the specified extension hits the RNA timer, the Call Consult Transfer step times out.
- Unsuccessful—The Call Consult Transfer step fails internally.

Note

- This step cannot be used to play any prompts to the destination.
- To support E.164 compliance, Unified CCX allows you to add "+" preceding to an agent extension or a route point directory number.

Figure 2-40 Call Consult Transfer Window


Call Consult Transfer - Script1.aef

General

Call Contact: --Triggering Contact--

Destination:

Output Digits:

Timeout: 10

Note While redirecting a call to an application one should use Call Redirect step and not Call Consult Transfer step.

Table 2-24 describes the properties of the Call Consult Transfer customizer window.

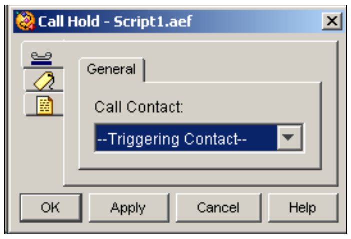
Table 2-24 Call Consult Transfer Properties

Properties	Description
Call Contact	Contact that you want to transfer. Default is Triggering Contact, unless another contact is defined.
Destination	Drop-down menu/Expression Editor string variable that stores the extension to which the call is to be transferred.
Output Digits	DTMF digits to outpulse to the destination after the destination answers and before the transfer is completed.
Timeout	Drop-down menu/Expression Editor Length of time, in seconds, to detect if the destination has answered the consult call.

Call Hold Step

Use the Call Hold step to put a call on hold.

Figure 2-41 Call Hold Customizer Window


Call Hold - Script1.aef

General

Call Contact: --Triggering Contact--

Executing this step if the call is already on hold has no impact on the call and produces no errors.

If a call is put on hold and then redirected or connected to an agent, the system automatically removed the call from hold before processing the connection request.

Note This step is available even if no media is associated with the call.

If the Computer Telephony Integration (CTI) ports of the Cisco Unified CCX server are configured for Music On Hold in the CTI ports in the Cisco Unified Communications Manager, then the User Hold Audio Source will be played to the caller. The audio source could be:

- A static .WAV file, such as a recorded voice message.
- A fixed audio source, such as a musical recording.
- A live audio source, such as a radio station.

Note While a call is on hold, media is temporarily disconnected from the Cisco Unified CCX server. If you design a script to play back prompts or wait for input from the caller while the call is on hold, the system will attempt, but fail, to play back the prompts, dropping all audio packets and timing out while waiting for caller input. To avoid this problem, you must include a Call Unhold step in the script to remove the call from hold in order to play a prompt or collect digits from the caller.

Table 2-25 describes the property of the Call Hold customizer window.

Table 2-25 Call Hold Property

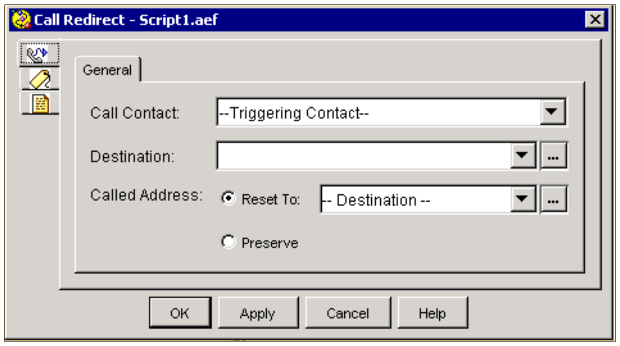
Properties / Buttons	Description
Call Contact	Contact that you want to put on hold. The default is Triggering Contact, unless another contact is defined.

Call Redirect Step

Use the Call Redirect step to redirect a call to another extension.

Note The Call Redirect step creates a new call which can result in double counting of an inbound call on real-time reports. Use the Select Resource Step if you do not want this to occur.

Figure 2-42 Call Redirect Customizer Window


Call Redirect - Script1.aef
General
Call Contact: --Triggering Contact--
Destination:
Called Address: (●) Reset To: -- Destination --
( ) Preserve

The Call Redirect step is often used in Interactive Voice Response (IVR) applications to transfer a call once a desired extension has been specified.

The Call Redirect step produces four output branches:

- Successful—The call is ringing at the specified extension.
- Busy—The specified extension is busy, and the call cannot be transferred.
- Invalid—The specified extension does not exist.
- Unsuccessful—The redirect step fails internally.

Place script steps after each of the different branches provided by this step to handle the possible outcomes of a redirected call.

Note

- The Call Redirect step updates Historical Data with a destination number only if the call is successful. In other cases (Invalid/Unsuccessful), the destination number is "Unknown. If the call redirection fails, then the destination number does not set in the contact call detail record.
- When transfering to another application, mark the call as handled using the Set Contact Info step. Not doing so will result in a delay of 5 seconds as additional internal events have to be handled for the call.
- The script moves only to the Busy branch of the Call Redirect step if the redirect destination is on the same Unified Communications Manager cluster. If the redirect destination is outside of the Unified Communications Manager cluster (over an ICT or a PSTN destination), the blind transfer aspect of the Call Redirect step does not allow the script to move to the Busy branch of the step. Instead, the Successful branch is used because the Unified Communications Manager cannot access call-state information after the blind transfer is completed to an off-cluster destination.
- To support E.164 compliance, Unified CCX allows you to add "+" preceding to an agent extension or a route point directory number.

Table 2-26 describes the properties of the Call Redirect customizer window.

Table 2-26 Call Redirect Properties

Properties / Buttons	Description
Call Contact	Contact that you want to redirect. The default is Triggering Contact, unless another contact is defined.
Destination	Variable that holds the extension where the call is to be redirected. (See Table 2-27 for supported extensions.)
Called Address	Select one of the following options: • **Reset To** radio button and drop-down list (default) • **Preserve** radio button If you select the **Reset To** option, you can select a value from the drop-down list. This value is reflected in the origCalledAddr column in Contact Call Detail Record (CCDR) report. If you select the **Preserve** option, the null value is reflected in the origCalledAddr column in CCDR report.

Table 2-27 describes the extensions supported by the Call Redirect step.

Table 2-27 Call Redirect—Supported Extensions

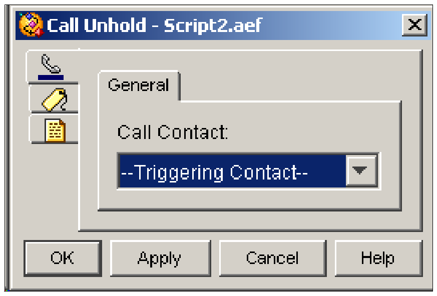
Extension	Description
Extensions starting with "#" or "*"	Extensions that trigger a network take-back and transfer where the specified string is outpulsed as is. The redirect is successful if a hang-up event occurs within a maximum of 5 seconds. **Note** You can use a comma (,) in the string to insert a pause of 1 second before the next digit is outpulsed.
Extensions ending with ".wav"	Extensions that trigger a network announcement type of redirect in which the system simulates a ring-back tone, then plays back the specified .wav file 4 times, and finally simulates a fastbusy tone. The redirect is successful if at any time the caller hangs up or the end of the fastbusy tone is reached, at which point the call is disconnected.
Extensions equal to "PROBLEMS"	Extensions that trigger a network announcement type of redirect with a system problem announcement. The redirect is successful if at any time the caller hangs up or the end of the audio is reached. The call will be reported as disconnected, not redirected.
Extensions equal to "BUSY", "RNA"[1], "FASTBUSY" or "DIALTONE"	The specified audio treatment is generated before the call is disconnected. The redirect is successful if at any time the caller hangs up or the end of the audio is reached. The call will be reported as disconnected, not redirected.

1. RNA = Ring No Answer

Call Unhold Step

Use the Call Unhold step to re-establish the connection with a call that you have previously put on hold by means of the Call Hold step.

Figure 2-43 Call Unhold Customizer Window



Call Contact: --Triggering Contact--

Note This step is important if you are designing a script that plays prompts or waits for input from the caller while the call is on hold. With using Call Unhold to re-establish the connection, the system will attempt, but fail, to play back the prompts, dropping all audio packets and timing out while waiting for caller input.

Table 2-28 describes the property of the Call Unhold customizer window.

Table 2-28 Call Unhold Property

Properties / Buttons	Description
Call Contact	Contact that you want to unhold. Default is Triggering Contact, unless another contact is defined.

Get Call Contact Info Step

Use the Get Call Contact Info step to access call-specific information and to store values in specified variables.

Note To support E.164 compliance, Unified CCX allows you to add "+" preceding to an agent extension or a route point directory number.

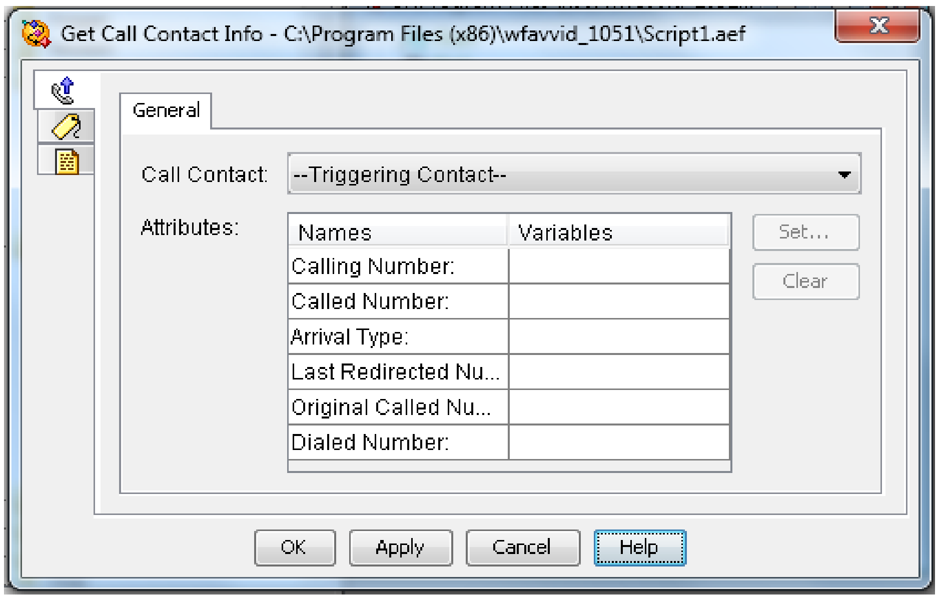
Figure 2-44 Get Call Contact Info Customizer Window

You can use this step to handle a call in a variety of ways depending on the source of the call and other properties associated with the session. For example, you can use this step with a Call Redirect step to transfer a call to another extension, or you can use this step with a Play Prompt step to play a voice prompt.

Table 2-29 describes the properties of the Get Call Contact Info customizer window.

Table 2-29 Get Call Contact Info Properties

Properties / Buttons	Description
Call Contact	Contact for which you want to get information. Default is Triggering Contact, unless another contact is defined.
Attributes (Names and Variables)	
Calling Number	The variable that stores the number of the originator of the call. If the call is an outbound call, this variable is the number dialed out.
Called Number	The variable that stores the number called by the calling party.
Arrival Type	The variable that holds the arrival type of the call. (See Table 2-30 for supported arrival types.)
Last Redirected Number	The number from which the last call diversion or transfer was invoked. This is the number at which the call was placed immediately before the current number.
Original Called Number	The number called from the perspective of the called party.


Get Call Contact Info - C:\Program Files (x86)\wfavvid_1051\Script1.aef
General
Set...
Clear
OK
Apply
Cancel
Help

Call Contact: --Triggering Contact--

Attributes:

Names	Variables
Calling Number:	
Called Number:	
Arrival Type:	
Last Redirected Nu...	
Original Called Nu...	
Dialed Number:	

Table 2-30 describes the arrival types of the Get Call Contact Info step.

Table 2-30

(Event) Arriv
UNKNOWN
DIRECT
REDIRECT
FORWARD_
FORWARD_
FORWARD_ R
TRANSFER
OUTBOUN
TIME_OF_I
DO_NOT_D
FOLLOW_M

Table 2-30 Get Call Contact Info—Arrival Types (continued)

(Event) Arrival Type	Description
OUT_OF_SERVICE	Call that was received because the originally called party was out of service.
AWAY	Call that was received because the originally called party was away.

Get Enterprise Call Info Step

Use the Get Enterprise Call Info step to get data from one part of your system and to transfer it to another part to store in local variables:

- If your system is a Cisco Unified IP IVR, then get data from Cisco Unified ICME and transfer it to Cisco Unified IP IVR.
- If your system is a Cisco Unified CCX, then get data from Cisco Finesse Desktop and transfer it to Cisco Unified CCX.
- If your system is a Cisco Unified CCX integrated with Cisco Unified ICME through the Cisco Unified Contact Center Gateway, then get data from Cisco Finesse Desktop and Cisco Unified ICME and transfer it to Cisco Unified CCX.

Note In a Cisco UnifiedIP IVR, this step requires the Cisco ICM software to receive the caller once the script completes, use this step only for Cisco Unified ICME VRU scripts and initial scripts, not for default scripts. (See the *Cisco Unified Contact Center Express Administration Guide*).

The Get Enterprise Call Info step description is divided into the following sections:

- General Tab (Get Enterprise Call Info Step)
- Expanded Call Variables Tab (Get Enterprise Call Info Step)

See also:

- Using Cisco Defined Call Variables
- Defining Expanded Call Variables
- System Default Expanded Call Variables
- Using the Parameter Separator

For further information on defining call variables in the Cisco Unified CCX Editor, see *Cisco Unified Contact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts.*

General Tab (Get Enterprise Call Info Step)

Use the General tab of the Get Enterprise Call Info customizer window to get data from enterprise call variables and to make it available to the script by storing the data in local variables.

The variables defined in the General tab are stored in the Cisco Finesse Desktop call record fields and will be written to the db_cra database. They can be found in the ContactDialDetail table and can be used in reporting. The General tab permits the use of Cisco predefined Enterprise Call Variables to be used both to populate CDR record fields and to populate the Cisco Finesse Desktop Data Fields using Layout Lists constructed in the Cisco Finesse Administration.

See the *Cisco Unified ICM Enterprise Software Installation and Configuration Guide* for how to configure call variables in Cisco Unified ICME software.

Figure 2-45 Get Enterprise Call Info Customizer Window, General Tab

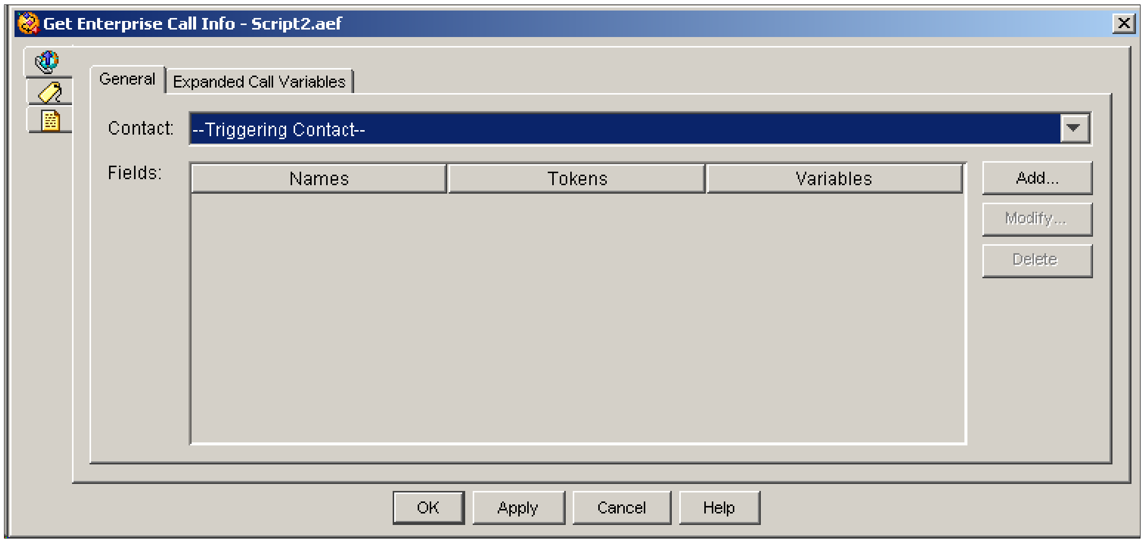


Table 2-31 describes the properties of the General tab of the Get Enterprise Call Info customizer window.

Table 2-31 Get Enterprise Call Info Properties, General Tab

Properties / Buttons	Description
Contact	The variable that stores the contact for which you want to get information. The default is the triggering contact.
Name (Field)	The name of the enterprise call variable from which you want to get data.
Token (Field)	The token (index) number of the enterprise call variable. Using a token, you can assign more than one value to a variable. If a call variable has a token, it can have a set of values similar to an array index. This is used only in a Cisco UnifiedICME environment. Do nothing if you do not want to use a token. If you want to use a token, select a number for the token. **Note** The default token separator is l. This can be modified on the system parameters page in the Cisco Unified CCX Application Administration window.

Table 2-31 Get Enterprise Call Info Properties, General Tab (continued)

Properties / Buttons	Description
Variable (Field)	The name of the local variable into which you want to put data.
Add / Modify (Button)	Use the Add button to assign the value of a local variable or expression (Value) to an ECC variable. The Add and Modify buttons access the Add Field and Modify dialog boxes. Use the dialog box to add or modify the mapping of enterprise call variables to local variables and click **OK** when finished. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (Button)	To remove the mapping of an enterprise call variable to a local variable, highlight the row containing the variable and click **Delete**.

Expanded Call Variables Tab (Get Enterprise Call Info Step)

Use the Expanded Call Variables tab of the Get Enterprise Call Info step to get data from enterprise Expanded Call Context (ECC) variables and to store the data in local variables.

Note

- In Cisco Unified IP IVR, you must first add enterprise ECC Variables in the Settings window of the Cisco Unified CCX Editor, before you can use them in the step.
- Unlike enterprise Call Variables (those defined in the General tab), the enterprise ECC variables (those defined in the Expanded All Variables tab) are not written to the db_cra database and cannot be used in reporting. The varaibles are stored in the Call Contact Detail table of the database.
- In Cisco Unified CCX, you do not have to predefine enterprise ECC variables in the settings window of the Cisco Unified CCX Editor. You can just open the Set Enterprise Call Info step, give the enterprise ECC variable a name, set its value to whatever you want, and save the step. If you predefine an enterprise ECC variable, then that variable can be accessed by all scripts. If you do not predefine an enterprise ECC variable, then you can use it only in the script containing the Set Enterprise Call Info step where you define it.

Every enterprise ECC variable must be defined on both sides of the system that gets and sends the variable data:

- In a Cisco UnifiedIP IVR, the enterprise ECC variable must be defined both in the Cisco UnifiedIP IVR and in the Cisco UnifiedICME software.
- In a Cisco Unified CCX system, the enterprise ECC variable must be defined both the Cisco Unified CCX script and in Cisco Finesse Administration.
- In a Cisco Unified CCX system integrated with Cisco UnifiedICME software through the IPCC gateway, the enterprise ECC variable must be defined in the Cisco Unified CCX Cisco Unified CCX script, in the Cisco Finesse Administration, and in Cisco UnifiedICME software.

Figure 2-46 Get Enterprise Call Info Customizer Window, Expanded Call Variables Tab

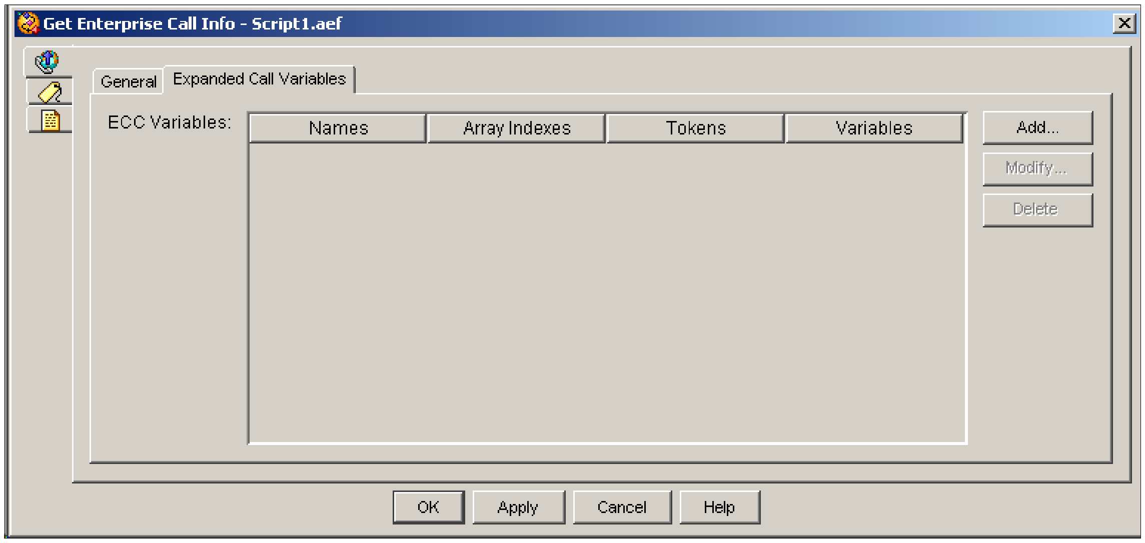


Table 2-33 describes the properties of the Expanded Call Variables tab of the Get Enterprise Call Info customizer window.

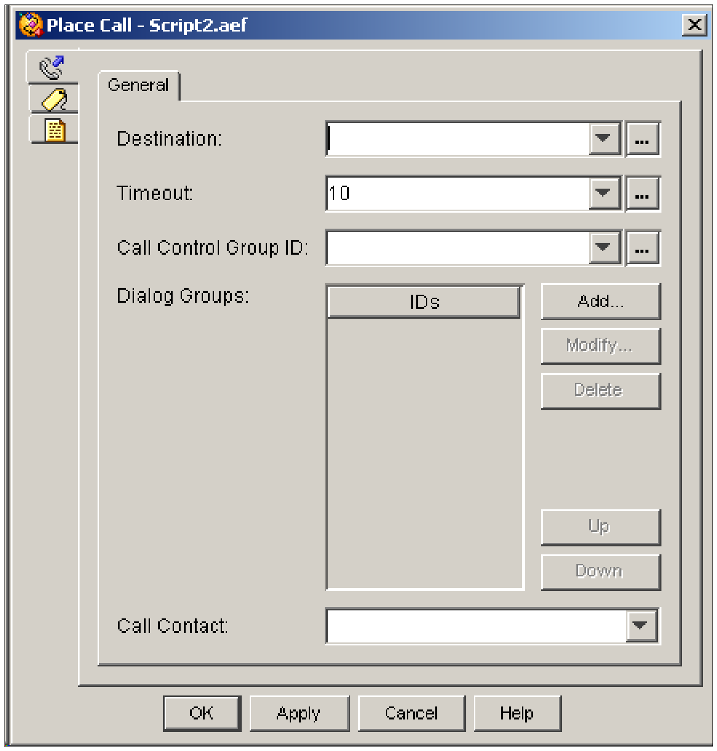
Table 2-32 Get Enterprise Call Info Properties, ECC Variables tab

Properties / Buttons	Description
Name (ECC variable)	The name of the enterprise ECC variable from which you want to get data. A user defined ECC variable should begin with "user."
Array Index (ECC variable)	If the enterprise ECC variable is of type array, the index of the array.
Token (ECC variable)	The token (index) number of the enterprise ECC variable. Using a token, you can assign more than one value to a variable. Only use in a Cisco UnifiedICME environment. Do nothing if you do not want to use a token. If you want to use a token, select a number for the token. **Note** The default token separator is |. This can be modified on the system parameters page in the Cisco Unified CCX Application Administration window.
Variable (Local Variable)	The name of the local variable in which you want to store data.
Add / Modify (Button)	Use the add or modify button to apply the value (Values) of a local variable to a user-defined Expanded Call Variable (Names). These buttons access the Add or Modify ECC Variable dialog box. Use that dialog to add or modify the mapping of enterprise ECC variables to local variables and click **OK** when finished. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (Button)	To remove the mapping of an enterprise ECC variable to a local variable, highlight the row containing the variable and click **Delete**.

Place Call Step

Use the Place Call step to place outbound calls.

Figure 2-47 Place Call Customizer Window



The Place Call step returns successfully if the call is answered in the configured time by a person, voice mail, an answering machine, or a fax machine.

The Place Call step has six output branches:

- Successful—The call is successfully made.
- NoAnswer—The call was attempted but the RNA Timeout limit was reached.
- Busy—The call was attempted but the line was busy.
- Invalid—The call was attempted but the extension was invalid.
- NoResource—The call was not attempted because no Resource was available to make the call.
- Unsuccessful—The call was attempted but failed because of an internal system error.

Note To support E.164 compliance, Unified CCX allows you to add "+" preceding to an agent extension or a route point directory number.

Table 2-33 describes the properties of the Place Call customizer window.

Table 2-33 Place Call Properties

Properties / Buttons	Description
Destination (Telephone Number)	Drop-down menu/Expression Editor string variable that stores the destination number of the outbound call.
Timeout (sec)	Drop-down menu/Expression Editor Length of time, in seconds, before a Ring No Answer condition stops the script from waiting for the remote side to answer and returns through the RingNoAnswer output branch.
Call Control Group ID	Variable that stores the identifying number of the call group with which the outbound call is associated. **Note** This identifier and the identifier for dialog groups will typically be configured as a parameter to the script. This variable is similar to the parameters defined during the configuration of a JTAPI[1] trigger.
Dialog Groups	This is an ordered list of dialog group identifiers where the first one has priority and the other dialog groups fall back in the specified order.
Call Contact	Variable that stores the call that is created when the step succeeds.

1. JTAPI = Java Telephony Application Program Interface

Set Enterprise Call Info Step

Use the Set Enterprise Call Info step to send data from one part of your system to another:

- If your system is a Cisco Unified IP IVR, then from Cisco Unified IP IVR to Cisco Unified ICME.
- If your system is a Cisco Unified CCX one, then from Cisco Unified CCX to Cisco Finesse Desktop.
- If your system is a Cisco Unified CCX one integrated with Cisco Unified ICME through the Cisco Unified CCGX, then from Cisco Unified CCX to Cisco Finesse Desktop and Cisco Unified ICME.

For an example of how to use the Set Enterprise Call Info Step in a script, see the chapter on “Designing Scripts for use with the Cisco Application Gateway” in the *Cisco Unified Contact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts*.

Note In a Cisco UnifiedIP IVR, because this step requires the Cisco Unified ICME software to receive the caller once the script completes, use this step only for Cisco Unified ICME VRU scripts and initial scripts, not for default scripts. (See the *Cisco Unified Contact Center Express Administration Guide*).

Note In a Cisco Unified CCX system:

- This step should be placed in the script before the call gets connected to an agent. This means the step in the script should be placed before the Select Resource Step or in the Selected/Queued branch of the Select Resource step.
- If call data is set after the call gets connected to the agent, it will not be displayed in Cisco Finesse Desktop.
- To display a custom layout on Cisco Finesse Desktop:

a. Create a new layout in Cisco Finesse Administration by modifying the Finesse layout xml.
b. In the Expanded Call Variables tab of the Set Enterprise Call Info step, create an ECC variable called "user.layout" and set its value to the name of the custom layout created in (1).

You can set the following enterprise call variables:

- Call.CallerEnteredDigits
- Call.PeripheralVariable1 to Call.PeripheralVariable10
- Call.AccountNumber
- Expanded Call Context (ECC) Variables

Note When an enterprise call variable or an enterprise ECC variable is used multiple times in the step, the result will be indeterminate in the following cases:

- If an enterprise call variable is set multiple times with the same token.
- If an enterprise ECC variable of type scalar is set multiple times with the same token.
- If an array element of an enterprise ECC variable of type array is set multiple times with the same token.

The Set Enterprise Call Info customizer window contains two tabs:

- General Tab (Set Enterprise Call Info Step)
- Using Cisco Defined Call Variables
- Expanded Call Variables Tab (Set Enterprise Call Info Step)
- System Default Expanded Call Variables
- Using the Parameter Separator

The following sections describe these tabs.

For further information on defining call variables in the Cisco Unified CCX Editor and for a list of call variable types used in Cisco Unified CCX scripts, see *Cisco Unified Contact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts.*

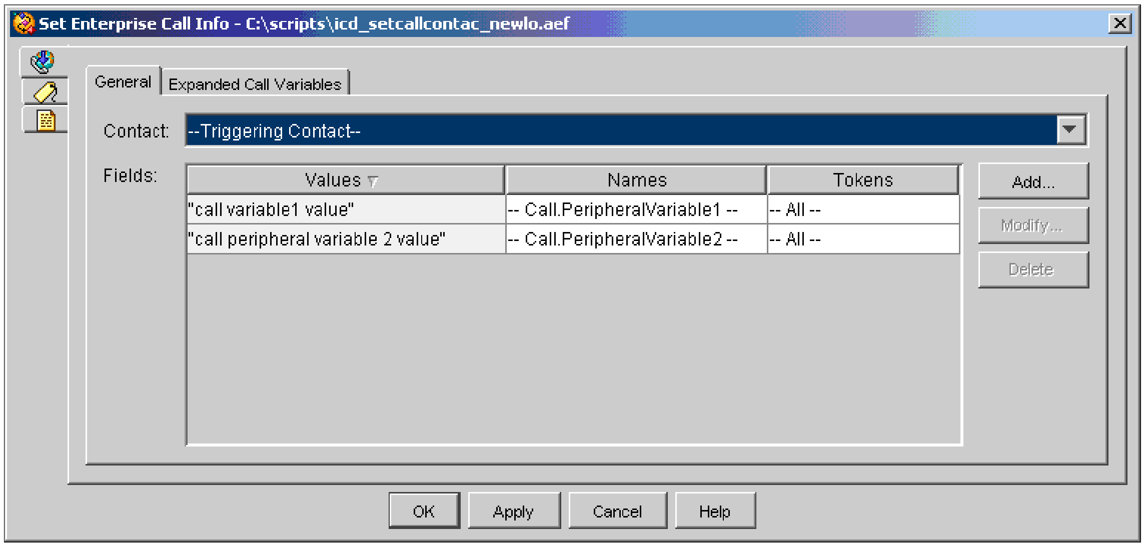
General Tab (Set Enterprise Call Info Step)

Use the General tab of the Set Enterprise Call Info step to set call data in predefined call variables. The variables defined in the General tab are stored in the Cisco Finesse Desktop call record fields. They can be found in the ContactDialDetail table and can be used in reporting.

Note You should not pass more than 40 characters in a call variable used in the Set Enterprise Call Info step since the database in which the call variables are stored limits the length of the call variables to 40 characters each.

Though the number of characters which you can pass in a custom call variable of the Set Enterprise Call Info Step is unlimited, if you include more than 40 characters, the extra characters will be lost when the variables are stored in the database and reports will not contain that additional information.

Figure 2-48 Set Enterprise Call Info Customizer Window, General Tab



Contact: --Triggering Contact--

Fields:

Values	Names	Tokens
"call variable1 value"	-- Call.PeripheralVariable1 --	-- All --
"call peripheral variable 2 value"	-- Call.PeripheralVariable2 --	-- All --

Table 2-34 describes the properties of the General tab of the Set Enterprise Call Info customizer window.

Table 2-34 Set Enterprise Call Info Properties, General Tab

Properties / Buttons	Description
Contact	The variable that stores the contact for which you want to set information. The default is the triggering contact.
Value (Field)	An expression or variable from the drop-down list, or an expression which will be the value to which you set the enterprise call variable
Name (Field)	The name of the enterprise call variable you want to set.
Token (Field)	The token (index) number of the enterprise call variable. Using a token, you can assign more than one value to a variable. If a call variable has a token, it can have a set of values similar to an array index. Do nothing if you do not want to use a token. If you want to use a token, select a number for the token. **Note** The default token separator is |. This can be modified on the system parameters page in the Cisco Unified CCX Application Administration window.

Table 2-34 Set Enterprise Call Info Properties, General Tab (continued)

Properties / Buttons	Description
Add / Modify (Button)	Use these buttons to access the Add Field dialog box. Use that dialog to add or modify the mapping of enterprise call variables to local call variables and click **OK** when finished. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (Button)	To remove the mapping of an enterprise call variable to a local call variable, highlight the row containing the variable and click **Delete**.

Using Cisco Defined Call Variables

In addition to the script variables that you can define in the Variable pane of the Cisco Unified CCX Editor window, you can use the following Cisco predefined strings as call variables in Cisco Unified CCX in the Get/Set Enterprise Call Info steps:

- VRU Script Name
- ConfigParam
- Call.CallingLineID
- Call.CallerEnteredDigits
- Call.PeripheralVariable1 to Call.PeripheralVariable10
- Call.AccountNumber

These call variables are written to the db_cra database.

Note These call variables are available from a list in the General tab of the Customization window only in the Get/Set Enterprise Call Info steps.

The Cisco Unified ICME Server, the Cisco Unified CCX system, and the Cisco Finesse Desktop support these call variables for passing data between themselves. You can pass data in these call variables between:

- Cisco Unified IP IVR and Cisco Unified ICME software, if you have a Cisco Unified IP IVR.
- Cisco Unified CCX and Cisco Finesse Desktop, if you have a Cisco Unified CCX system.
- Cisco Unified CCX and Cisco Finesse Desktop and Cisco Unified ICME software, if your system is a Cisco Unified CCX one integrated with Cisco Unified ICME software through the IPCC Gateway.

Expanded Call Variables Tab (Set Enterprise Call Info Step)

If you need more call variables than those predefined in the General tab, use expanded call context (ECC) variables.

Use the Expanded Call Variables tab of the Set Enterprise Call Info step to set data in enterprise ECC variables.

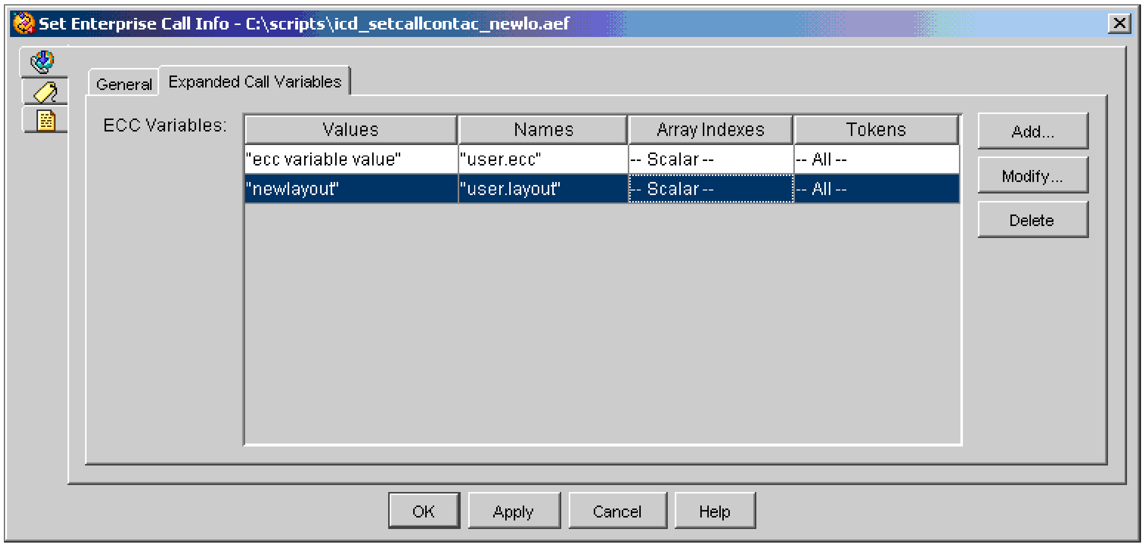
Note

- In Cisco UnifiedIP IVR= , you must first add the enterprise ECC variables in the Settings window of the Cisco Unified CCX Editor, before you can use them in the step (see Defining Expanded Call Variables).
- Unlike enterprise Call Variables (those defined in the General tab), the enterprise ECC variables (those defined in the Expanded All Variables tab) are not written to the db_cra database and cannot be used in reporting.
- In Cisco Unified CCX, you do not have to predefine enterprise ECC variables in the settings window of the Cisco Unified CCX Editor. You can just open the Set Enterprise Call Info step, give the enterprise ECC variable a name, set its value to whatever you want, and save the step. If you predefine an enterprise ECC variable, then that variable can be accessed by all scripts. If you do not predefine an enterprise ECC variable, then you can use it only in the script containing the Set Enterprise Call Info step where you define it.

Every enterprise ECC variable must be defined on both sides of the system that sends and receives the variable data:

- In a Cisco Unified IP IVR, the enterprise ECC variable must be defined both in Cisco Unified IP IVR and in Cisco Unified ICME software.
- In a Cisco Unified CCX system, the enterprise ECC variable must be defined both the Cisco Unified CCX script and in Cisco Finesse Administration.
- In a Cisco Unified CCX system integrated with Cisco Unified ICME software through the IPCC gateway, the enterprise ECC variable must be defined in the Cisco Unified CCX script, in the Cisco Finesse Administration, and in Cisco Unified ICME software.

Figure 2-49 Set Enterprise Call Info Customizer Window, Expanded Call Variables Tab



Values	Names	Array Indexes	Tokens
"ecc variable value"	"user.ecc"	-- Scalar --	-- All --
"newlayout"	"user.layout"	-- Scalar --	-- All --

Table 2-35 describes the properties of the Expanded Call Variables tab of the Set Enterprise Call Info customizer window.

Table 2-35 Set Enterprise Call Info Properties, Expanded Call Variables tab

Properties / Buttons	Description
Value (ECC variable)	An expression or variable from the drop-down list, or an expression which will be the value to which you set the enterprise ECC variable.
Name (ECC variable)	The name of the enterprise ECC variable you want to set. A user defined ECC variable should begin with "user."
Array Index (ECC variable)	If the enterprise ECC variable is of type array, the index of the array.
Token (ECC variable)	The token (index) number of the enterprise ECC variable. Using a token, you can assign more than one value to a variable. Do nothing if you do not want to use a token. If you want to use a token, select a number for the token. **Note** The default token separator is |. This can be modified on the system parameters page in the Cisco Unified CCX Application Administration window.
Add / Modify (Button)	Use these buttons to access the Add ECC Variable dialog box. Use that dialog to add or modify the mapping of enterprise ECC variables to local variables and click **OK** when finished. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (Button)	To remove the mapping of an enterprise ECC variable to a local variable, highlight the row containing the ECC variable and click **Delete**.

Defining Expanded Call Variables

Expanded Call Context (ECC) are data fields used by all applications in the Cisco Unified CCX Cluster. There can be as many as 200 user-defined fields defined in the Field List (index numbers 0-199) of expanded call variables. These field values do not appear in the ContactCallDetail records as there are no fields reserved for them.

The Cisco Unified CCX system, and the Cisco Finesse Desktop pass ECC variables to each other. However, you must separately define ECC variables on both sides of the system that sends and receives the variable data.

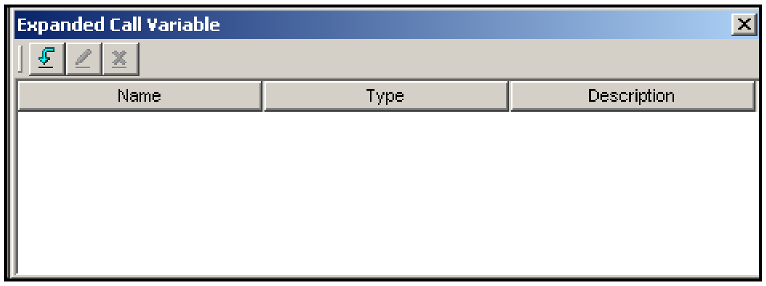
To define an ECC variable in the Cisco Unified CCX Editor, do the following.

Procedure

Step 1 From the Cisco Unified CCX Editor menu bar, choose **Settings > Expanded Call Variables.**

The Expanded Call Variables window appears.

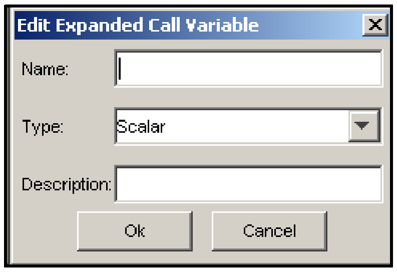
Figure 2-50 Expanded Call Variables Window



Step 2 In the tool bar, click the **Add New Variable (down arrow)** icon.

The Edit Expanded Call Variable dialog box appears.

Figure 2-51 Edit Expanded Call Variable Dialog Box



Step 3 In the Name text field, enter the ECC variable name as defined in the Cisco Unified ICME configuration (or the Cisco Finesse Administration using Cisco Unified CCX).

Step 4 In the Type drop-down menu, choose the type of expanded call variable (scalar or array).

Step 5 In the Description text field, enter a description of the variable.

Step 6 Click **OK**.

The Edit Expanded Call Variable dialog box closes, and the variable name, type, and description appear under their respective columns in the Cisco UnifiedICME Expanded Call Variables window (or the Cisco Finesse Administration using Cisco Unified CCX).

Step 7 Click the dialog box's **Close (X)** button.

The Expanded Call Variables window closes, and the ECC variable is now available to your script. It will be listed in the Enterprise Call Info step's drop-down list as ---name--, where *name* is the value you entered in the Variable Name field.

To define am ECC variable in the Cisco Finesse Administration, select **Enterprise Data > Field List**. When creating a new Field, a unique Index number (Index) between 0-199 is assigned. The Field Name (Field Name) is the name of the field called by the layout list and is case sensitive. The Display Name (Display Name) is the name (Field) that will show on the agent desktop with the value of the field.

Note For information on configuring Cisco Unified ICME software for Cisco Unified CCX and for configuring ECC variables in Cisco Unified ICME software, see the "*Cisco Unified Contact Center Express Administration Guide*" and the *Cisco Unified ICM Enterprise Installation and Configuration Guide* .

System Default Expanded Call Variables

Table 2-36 below describes the system default expanded call variables.

Table 2-36 System Expanded Call Variables

Applications	Field	Variable Name	Length	Description
Cisco UnifiedICME software (Only used with Cisco UnifiedIP IVR if configured on the Cisco Unified CCX Administration web site.) Cisco Unified CCX	Task ID	user.task.id	19	Task ID that handles the current call.
	Media ID	user.media.id	14	Media ID that handles the current call.
	Last Redirected Address	user.last.redirected.address	40	Transient part ID of the call.
	Arrival Type	user.connect.type	17	Arrival Type of the call.
	Session Handled	user.session.handled	5	Boolean flag that the Cisco Cisco UnifiedICME or a Set Contact Info step with the Handled flag sets to indicate whether the session is handled. **Note** In Cisco Unified CCX , the flag is automatically set whenever the call is connected to an agent.
	VRU Script Name	user.connect.script.name	40	Script name to run on the PreConnect feature.
	Config Param	user.connect.script.config	40	Parameters for the VRU scripts on the PreConnect feature.
	Note These variables are only available for VRU scripts.			
	Layout	user.layout	40	Defines the layout on the Cisco Finesse Desktop.

Using the Parameter Separator

You can send multiple values—or tokens—within one variable, so you can avoid using many variables at the same time.

The parameter separator is a character that defines the boundary between the different tokens in one variable. The token numbering begins with 0.

Note You specify this separator on the Cisco Unified CCX Administrator's System Parameters web page.

For example, if the parameter separator is “|”, you can send an expanded call variable as

```
true|4|4/3/2000
```

where the value of token number 0 is “true”, token number 1 is “4”, and token number 2 is “4/3/2000”.

The following variables can have multiple tokens separated by the parameter separator:

- Peripheral (1-10)
- Expanded Call (ECC)
- ConfigParam
- VRU Script Name

Note The VRU Script Name variable holds the name of a VRU script to run when the Cisco Unified CCX system receives a Run Script request from Cisco Cisco Unified ICME software. The Cisco Unified CCX system reads only the first token (token number 0) as VRU Script Name; the rest of the tokens are passed on (with the script name - token #0) to be used as regular variable tokens.

The Get/Set Enterprise Call Info steps of the Call Contact palette in the Cisco Unified CCX Editor allow you to set and read different tokens in the variables passed between the Cisco Unified CCX system and the Unified ICM Enterprise Server.

eMail Contact Steps

The steps in the eMail Contact palette of the Cisco UnifiedContact Center Express Editor provide script designers with a way to create and send e-mail messages and embedded attachments. You can send e-mail to regular e-mail accounts, to e-mail-accessible text and numeric pagers, and to service providers who convert e-mail into a format acceptable to fax machines.

The kind of e-mail messages and attachments you can send depends on the target device. Numeric pagers, for example, will accept only digits.

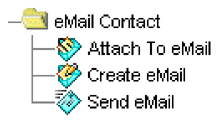
This section contains the following topics:

- Using the Step Editor to send eMail Messages
- Attach To eMail step
- Create eMail Step
- Send eMail Step

Figure 2-52 shows the steps in the eMail Contact palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Note The eMail Contact palette steps are runnable in your Cisco Unified CCX scripts only if you have purchased the Cisco Unified IP IVR or Cisco Unified CCX Premium license options.

Figure 2-52 eMail Contact Palette Steps



Note If you apply any eMail Contact step to a contact that is not an e-mail contact, a ChannelUnsupportedException results.

Using the Step Editor to send eMail Messages

The following sequence is the typical sequence used to create and send an e-mail message:

1. Obtain the client’s e-mail address by using, for example, the Name To User step or a database lookup.

If the address is a numeric address, you can obtain it through dual tone multifrequency (DTMF) input. See the "Name To User Step" section on page 2-112 for more information.

2. Use the Create eMail step to create the e-mail message.
3. Use the Attach to eMail step to attach any required or requested documents to the e-mail message.

   You can create a menu of documents that a client can use to attach selected documents to an e-mail. If the documents are stored in a database, first use the Database steps to retrieve the documents (see "Do not use hard-coded string for the output variables. For output variables such as Response, Status Code, and Status Detail, declare the variables first and then associate the declared variables to the output variables." section on page 2-188
4. Use the Send eMail step to send the e-mail.

   If you are sending a document to a fax machine, you must send it to a service provider that can convert the attachment to a format that fax machines can handle.

   Unless you specify otherwise, the e-mail server sends the e-mail message from the account defined in the eMail Subsystem Configuration window of the Cisco Unified CCX Administration web interface. You can use the Send eMail step to customize the outgoing e-mail account.

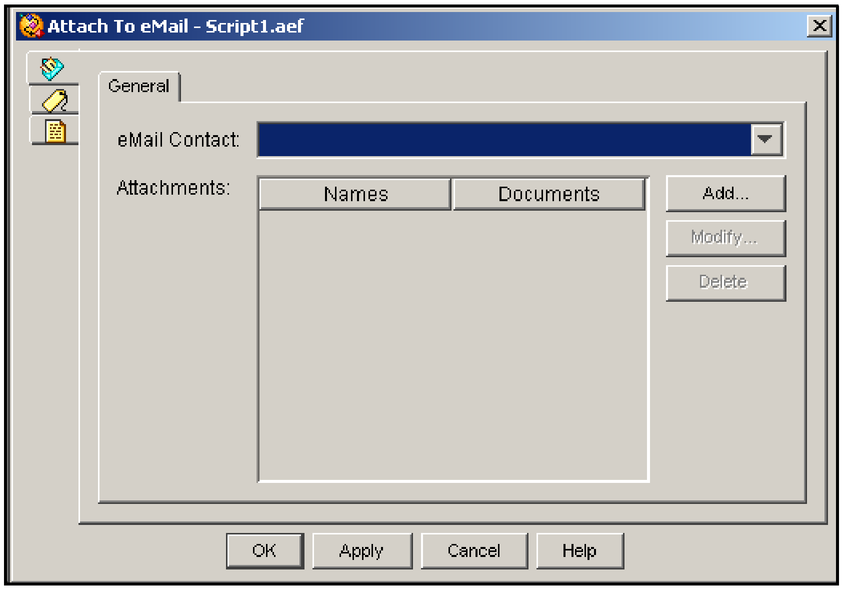
   If you set the sending e-mail account to require acknowledgments, you can check the e-mail account to determine whether the e-mail message was successfully sent and received.

Attach To eMail step

Use the Attach To eMail step to attach a document to an e-mail.

Note Before you use an Attach To eMail step, you must use a Create eMail step to create the e-mail message.

Figure 2-53 Attach To eMail Customizer Window



Note If the document to be attached resides in a database, you must use the Database steps to retrieve the document from the database before attaching it. (See "Do not use hard-coded string for the output variables. For output variables such as Response, Status Code, and Status Detail, declare the variables first and then associate the declared variables to the output variables." section on page 2-188)

Note You can use the Keyword Transform Document step to modify documents before attaching them. For instructions on using the Keyword Transform Document step, see the "Keyword Transform Document Step" section on page 2-180.

Assign variables before you use the Attach to eMail step, using the Create eMail step. (See the "Create eMail Step" section on page 2-63.)

The variable you use for the attachment can be a string or document variable, described as follows:

- String—Use a string variable if you do not need to manipulate the document using the Document steps (for example, the Keyword Transform Document step).
- Document—Use a document variable if you also use the Document steps to manipulate the attachment.

The Attach To eMail step has two branches:

- Successful—The document was attached to the e-mail object.
- Failed—The document could not be attached to the e-mail object.

Table 2-37 describes the properties of the Attach To eMail customizer window.

Table 2-37 Attach To eMail Properties

Property	Description
eMail Contact	Contact variable that specifies the e-mail message to which you want to attach the document
Attachments	Information identifying the document you want to attach to the e-mail message
Name	Local variable
Document	Data Type Name for Attachment

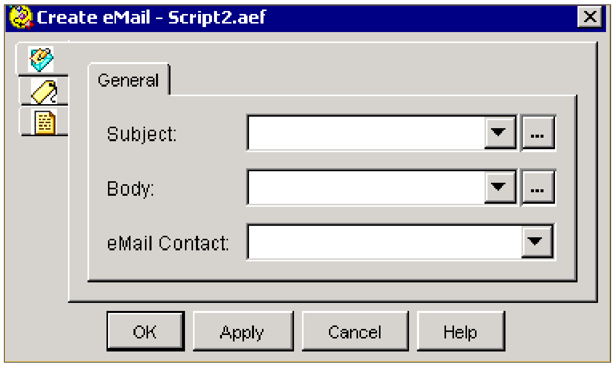
Table 2-37 Attach To eMail Properties (continued)

Property	Description
Add / Modify (buttons)	Use these buttons to access the Attachment dialog box. Use the dialog to specify the following: • **Local Variable**—Choose a variable from the drop-down list that identifies the document you want to attach to the e-mail. • **Data Type**—Display only. Appears automatically in the Data Type box after you specify the Local Variable. • **Name for Attachment**—Specify a filename, including its extension. **Note** Most e-mail readers place the Name for Attachment text near an icon that represents the attachment. The extension you provide as part of the document name indicates to the user the type of document you attached. For example, ".doc" normally indicates that a document was created with Microsoft Word, ".wav" is an audio document, and so on. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove a Local Variable, highlight a value in the list and click **Delete**.

Create eMail Step

Use the Create eMail step to create an e-mail message.

Figure 2-54 Create eMail Customizer Window



In the Create eMail step, you generate the subject line and body of the e-mail message. After you create the e-mail, you can attach documents to it (with the Attach To eMail step) and then send it (with the Send eMail step).

Table 2-38 describes the properties of the Create eMail customizer window.

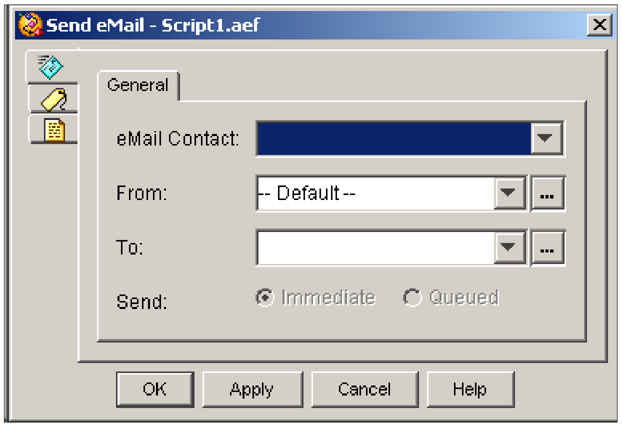
Table 2-38 Create eMail Properties

Property	Description
Subject	Variable or expression that you want to use for the subject line of the message.
Body	Variable or expression that you want to use for the body of the e-mail message
eMail Contact	Variable that identifies the email. **Note** You can manage multiple e-mail messages in the same script by saving each message in a different variable.

Send eMail Step

Use the Send eMail step to send an e-mail message you have created with the Create eMail step.

Figure 2-55 Send eMail Customizer Window



eMail Contact:
From: -- Default --
To:
Send: [✓ Immediate] Queued

The Send eMail step has two branches:

- Successful—The e-mail message was sent.
- Failed—The e-mail message could not be sent.

When a script reaches the Send eMail step, it immediately sends the e-mail message to the e-mail server, and keeps the client waiting until the message is accepted by the e-mail server. If the server is unavailable because of server or network problems, the client must wait until the transaction times out.

Table 2-39 describes the properties of the Send eMail customizer window.

Table 2-39 Send eMail Properties

Property	Description
eMail Contact	Contact variable created in the Create eMail step that identifies the e-mail you want to send.
From	Account from which you are sending the e-mail. Valid options are: • Default. Uses the account configured for the e-mail subsystem on the Cisco Unified CCX Administration web interface. • Variable name or expression.
To	Variable or string expression that identifies the recipient of the e-mail. **Note** Typically, this expression is based on either user input or a database lookup of the e-mail address. Make sure the e-mail address is fully qualified, that is, that it is in the form of account@domainname.
Send	(Radio buttons) **Immediate**. Causes an e-mail to be sent as soon as the step executes. **Queued**. Queues the e-mail rather than sending it immediately.

HTTP Contact Steps

The steps in the Http Contact palette of the Cisco Unified CCX Editor provide script designers with a way to enable scripts to receive HTTP requests and send HTTP responses in web-enabled server applications.

Note Use steps in the Document palette to compose the server response. For more information, see Document Steps.

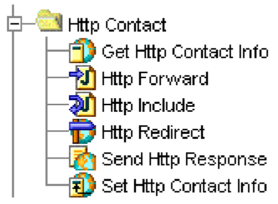
This section contains the following topics:

- Get Http Contact Info Step
- Http Forward Step
- Http Include Step
- Http Redirect Step
- Send Http Response Step
- Set Http Contact Info Step

Figure 2-56 shows the steps in the Http Contact palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Note The HTTP Contact palette steps are runnable in your Cisco Unified CCX scripts only if you have purchased the Cisco UnifiedIP IVR or Cisco Unified CCX Premium license options.

Figure 2-56 Http Contact Palette Steps



Note If you apply any of Http Contact step to a contact that is not an HTTP contact, a ChannelUnsupportedException results.

Get Http Contact Info Step

Use the Get Http Contact Info step to map parameters from an HTTP request to locally defined variables.

The Get Http Contact Info step gets URL parameters, HTTP headers, cookies, or Common Gateway Interface (CGI) environment variables. This information is stored in variables you define using the Edit Variables window.

The Get Http Contact Info customizer window contains the following five tabs:

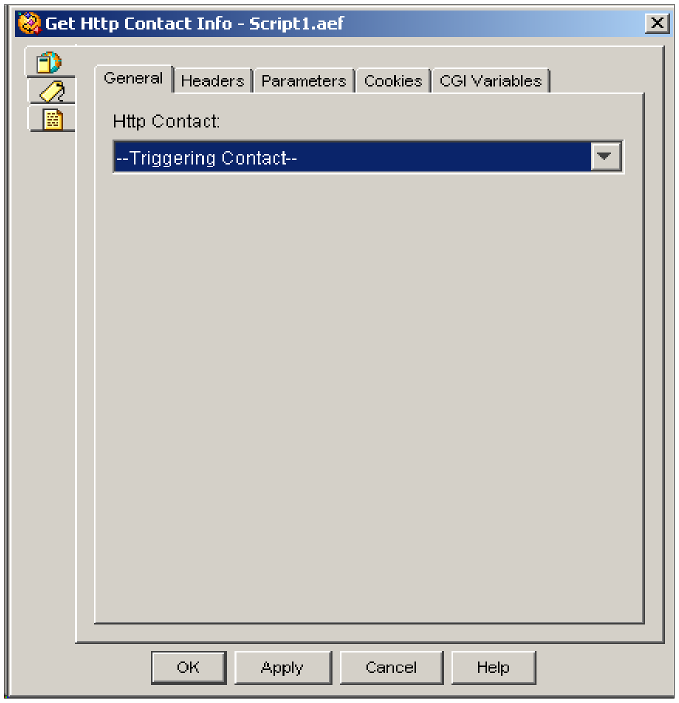
- General tab (Get Http Contact Info step)
- Headers tab (Get Http Contact Info step)
- Parameters tab (Get Http Contact Info step)
- Cookies tab (Get Http Contact Info step)
- CGI Variables tab (Get Http Contact Info step)

The following sections describe these tabs.

General tab (Get Http Contact Info step)

Use the General tab of the Get Http Contact Info customizer window to get trigger contact information.

Figure 2-57 Get Http Contact Info Customizer Window—General Tab



Http Contact: --Triggering Contact--

Table 2-40 describes the property of the General tab of the Get Http Contact Info customizer window.

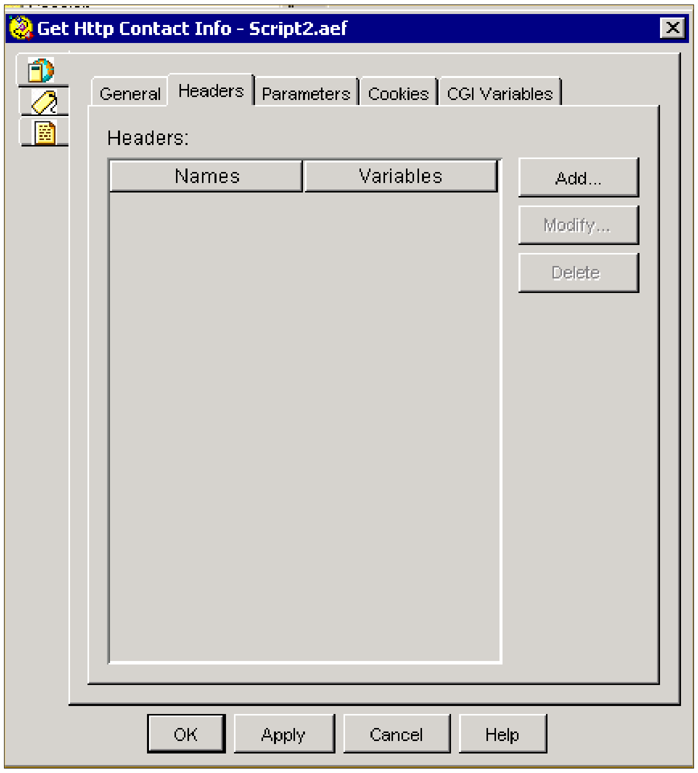
Table 2-40 Get Http Contact Info Property

Property	Description
Http Contact	Contact variable that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.

Headers tab (Get Http Contact Info step)

Use the Headers tab of the Get Http Contact Info customizer window to display the HTTP headers that you have mapped to local variables.

Figure 2-58 Get Http Contact Info Customizer Window—Headers Tab



HTTP headers contain general information such as the type of browser or the version of HTTP used. Each header provides one value, which is identified by the header name.

As an example, you may use information from HTTP headers in advanced scripts to customize the behavior of your script for different HTTP versions or for different browser types.

HTTP provides four types of headers:

- General—Used by both servers and clients (browsers).
- Server—Used only by servers.
- Request—Used only by clients (browsers). Common HTTP Request Headers include:
  - **Accept**. Preferred media type.
  - **Authorization**. Client user name and password.
  - **From**. E-mail address of the client.
  - **Host**. Host name and port number of the server receiving the original request.
  - **Referrer**. URL of the source document.
  - **User-Agent**. Browser type.
- Entity—Used by servers and by clients using POST or PUT methods.

Note For detailed information about these or other headers, refer to any HTTP reference guide.

Table 2-41 describes the properties of the Headers tab of the Get Http Contact Info customizer window.

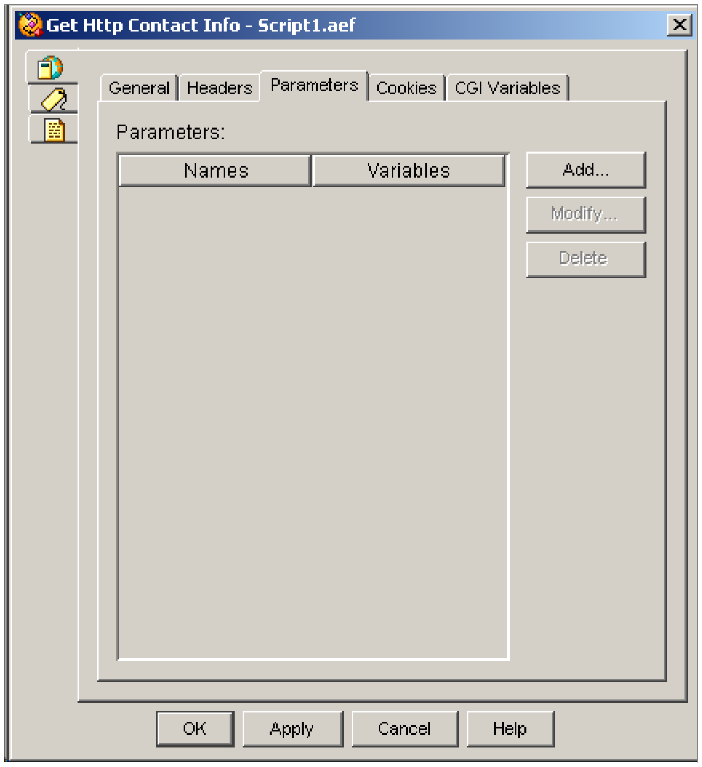
Table 2-41 Get Http Contact Info—Headers Tab Properties

Property	Description
Headers (Names and Variables)	The names of the headers and the corresponding variables that map to the headers.
Add / Modify (buttons)	Use these buttons to access the Header dialog box. Use the dialog to specify the following: • **Header**—Name for the HTTP header. • **Variable**—A string variable to be mapped to the parameter. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove HTTP Header information, highlight a value in the list and click **Delete**.

Parameters tab (Get Http Contact Info step)

Use the Parameters tab of the Get Http Contact Info customizer window to map the parameters for your script to variables you have defined in the Cisco Unified CCX Editor.

Figure 2-59 Get Http Contact Info Customizer Window—Parameters Tab



When you fill in an HTML form, the values are typically passed as parameters to the web server. The Get Http Contact Info step reads the values of these parameters from the HTTP request and updates the current values of the local variables in your application.

The Get Http Contact Info step reads values from both the GET and POST methods.

Table 2-42 describes the properties of the Parameters tab of the Get Http Contact Info customizer window.

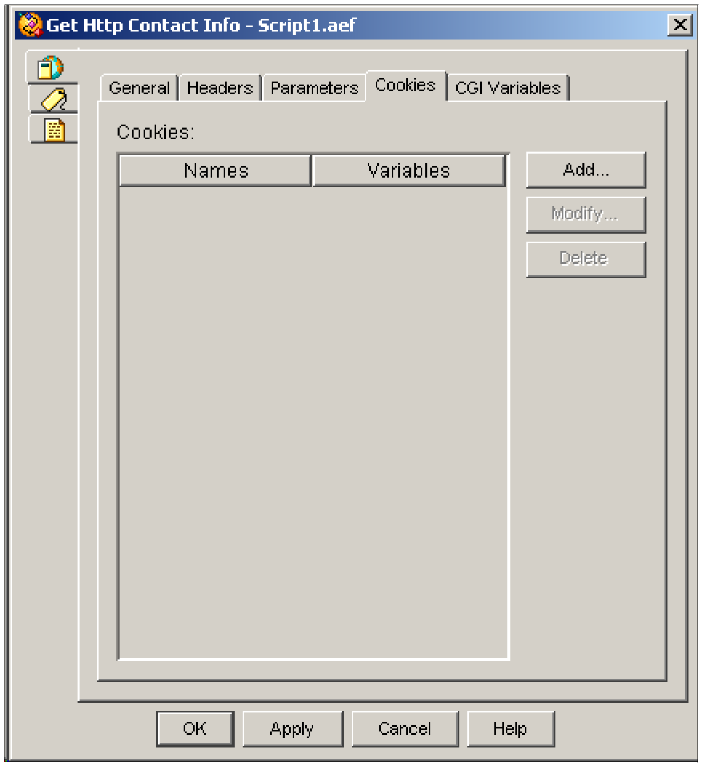
Table 2-42 Get Http Contact Info—Parameter Tab Properties

Properties / Buttons	Description
Parameters (Names and Variables)	The names of the parameters and the corresponding variables that map to the parameters.
Add / Modify (buttons)	Use these buttons to access the Parameters dialog box. Use the dialog to specify the following: • **Parameter**—Name for the Http Contact. • **Variable**—A string variable to be mapped to the parameter. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove Parameter information, highlight a value in the list and click **Delete**.

Cookies tab (Get Http Contact Info step)

Use the Cookies tab of the Get Http Contact Info customizer window to map information from a local variable to a cookie.

Figure 2-60 Get Http Contact Info Customizer Window—Cookies Tab



A *cookie* is information maintained by the browser that is typically sent by an HTTP server.

The information in cookies can improve performance and convenience when a user repeatedly accesses the same web page. Most cookies store authentication or identifying information. Once the server authenticates a browser, it can send authentication credentials or other user identifiers to the browser cookie. The user can then access the web page without further authentication or identification.

Another use for cookies is to store a mapping identifier to a Session object so you can, on subsequent requests, retrieve the original Session object associated with the previous HTTP request and re-associate it to the new Http Contact.

Note The use of cookies for authentication may present a security risk if a user leaves an authenticated browser unattended.

Table 2-43 describes the property of the Cookies tab of the Get Http Contact Info customizer window.

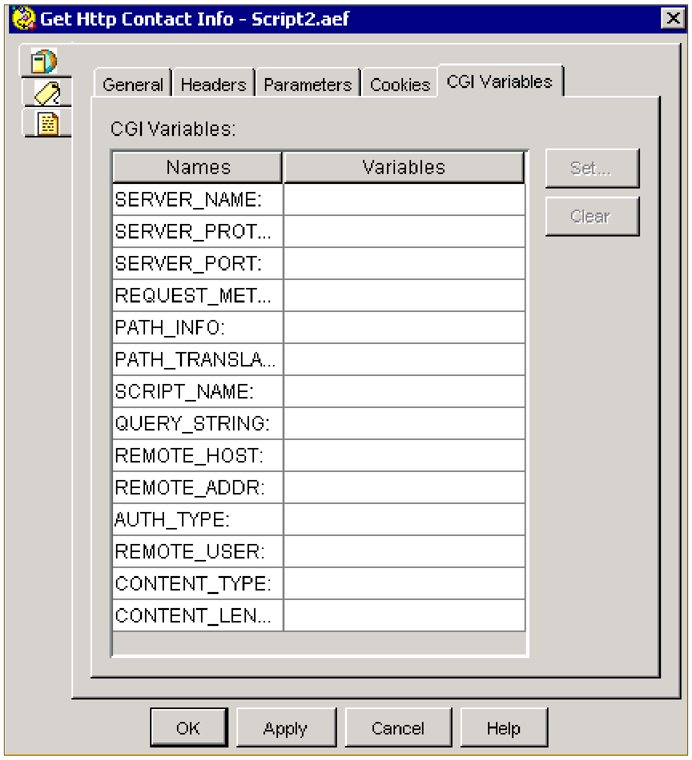
Table 2-43 Get Http Contact Info—Cookies Tab Property

Property	Description
Cookies (Names and Variables)	The names of cookies and the variables to which they are mapped.
Add / Modify (buttons)	Use these buttons to access the Cookie dialog box. Use the dialog to specify the following: • **Name**—Name for the cookie. • **Variable**—A string variable to be mapped to the cookie. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove the Cookie information, highlight a value in the list and click **Delete**.

CGI Variables tab (Get Http Contact Info step)

Use the CGI (Common Gateway Interface) Variables tab of the Get Http Contact Info customizer window to map information from CGI environment variables to local variables.

Figure 2-61 Get Http Contact Info Customizer Window—CGI Variables Tab



CGI Variables:

Names	Variables
SERVER_NAME:	
SERVER_PROT...	
SERVER_PORT:	
REQUEST_MET...	
PATH_INFO:	
PATH_TRANSLA...	
SCRIPT_NAME:	
QUERY_STRING:	
REMOTE_HOST:	
REMOTE_ADDR:	
AUTH_TYPE:	
REMOTE_USER:	
CONTENT_TYPE:	
CONTENT_LEN...	

Table 2-44 describes the properties of the Environment tab of the Get Http Contact Info customizer.

Table 2-44 Get Http Contact Info—Environment Tab Properties

Property	Description
Name / Variable	The names of CGI Environment variables and corresponding local variables to which they are mapped. **Note** See Table 2-45 for descriptions of each of the Environment variables.
Set (button)	To set a variable, select a Name and click **Set**. Choose a value from the Variable drop-down list and then click **OK**; the variable name appears in the Variable column next to the Name you selected.
Delete (button)	To remove Environment information, highlight a value in the list and click **Delete**.
Property	**Description**

Table 2-45 describes each Environment variable.

Table 2-45 Http Contact Environment Variables

Name	Description
AUTH_TYPE	The protocol-specific authentication method used to validate the user when the server supports user authentication and the script requires authentication.
CONTENT_LENGTH	The content length of the data as specified by the client.
CONTENT_TYPE	The content type of the data for queries such as HTTP GET and HTTP POST that have attached information.
PATH_INFO	Extra path information as given by the client. Scripts can be accessed by a virtual pathname, followed by extra information at the end of the path. The extra information is sent as PATH_INFO. The server decodes this information if it comes from a URL before it is passed to the script.
PATH_TRANSLATED	Translated version of PATH_INFO, with any associated virtual-to-physical mapping.
QUERY_STRING	The information that follows the question mark (?) in the URL that references this script. This information is the query information. Do not decode it. Always set this variable when there is query information, regardless of command line decoding.
REMOTE_ADDR	The IP address of the remote host making the request.
REMOTE_HOST	The hostname making the request. If the server does not have this information, it will instead set REMOTE_ADDR.
REMOTE_USER	The authenticated user name when the server supports user authentication and the script requires authentication.
REQUEST_METHOD	The method of the request that was made, such as GET or POST.

Table 2-45 Http Contact Environment Variables (continued)

Name	Description
SCRIPT_NAME	A virtual path to the script being executed, used for self-referencing URLs.
SERVER_NAME	The server’s hostname, DNS[1] alias, or IP address as it would appear in a self-referencing URL.
SERVER_PORT	The TCP port number of the request.
SERVER_PROTOCOL	The name and revision of the information protocol of the request, in the format: protocol/revision.

1. DNS = Domain Name Service

Http Forward Step

Use the Http Forward step to forward an HTTP request to an internal URI deployed under the document repository \default\webapps\ROOT (the default web applications folder) or \default\webapps\<your web apps> folder. The URI can be a JSP page, which would permit dynamic content generation, or any other resource like an HTML page for static content generation.

Internal URI paths are defined starting at the \default\webapps\ROOT folder. For example, a JSP page stored as:

- \default\webapps\ROOT\hello.jsp would be referenced using the URI path **hello.jsp**.
- \default\webapps\ROOT\help\help.jsp would be referenced using the URI path **\help\help.jsp**.

It is also possible to store war files under \default\webapps\ folder which would be automatically expanded by the internal web server and define new web applications that would be referenced using the URI path **<war file name>\<resource filename>**.

Use the HTTP Forward step to use Java Server Pages to generate dynamic output to a browser.

The HTTP Forward step lets you define local variables that you can pass to the JSP. When the HTTP Forward step executes, these variables are passed as parameters to the JSP. The JSP uses these parameters to format the output that is sent to the user.

You can use the HTTP Forward step instead of the Keyword Transform Document step (see the “Keyword Transform Document Step” section on page 2-180) if you are already using JSP or if you need the full capabilities of the Java programming language.

A script should use only one of the following steps from the Http Contact palette to respond to an HTTP request:

- Http Redirect
- Http Forward
- Http Include
- Send Http Response

If you use the Http Forward step, you should not use either a Http Redirect or Send Response step in the same sequence within a script, because using any one of these three steps moves the Http Contact into a final state when the response is returned back to the browser. The script cannot attempt to send another response back because one has already been sent. If, however, a script has some conditional logic, you can use one of these three steps in one condition and another in another condition.

Note You can also use the Reject and Terminate steps from the Contact palette as responses to an HTTP request. For more information, see "Contact Steps" section on page 2-34.

With JSP, you can access the parameter value by creating a parameter bean and by retrieving the parameter passed by the HTTP Forward step. The following example, which you create using a text editor, shows how to access and retrieve the parameter *name*.

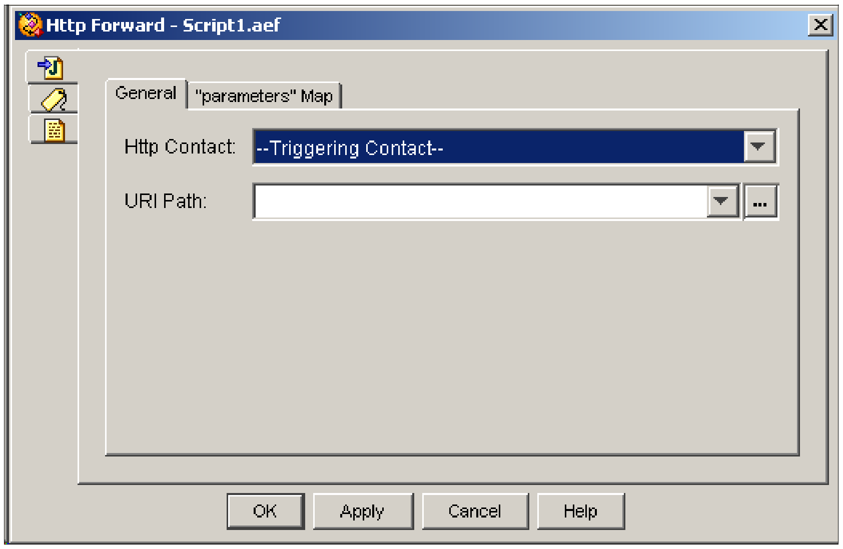
Note This file would be uploaded to the Document repository's default\webapps\ROOT folder.

Example 2-1 Sample JSP file (hello.jsp) for Accessing a Parameter

```
<html>
<body>

<!-- Create the "parameters" bean -->
<jsp:useBean id="parameters" type="java.util.Dictionary" scope="request"/>

<!-- Retrieve the parameter "name" -->
Hello <%= parameters.get("name") %>

</body>
</html>
```

Figure 2-62 shows the customizer window, General tab for the HTTP Forward step.

Figure 2-62 HTTP Forward Customizer Window—General tab



Http Contact: --Triggering Contact--
URI Path:

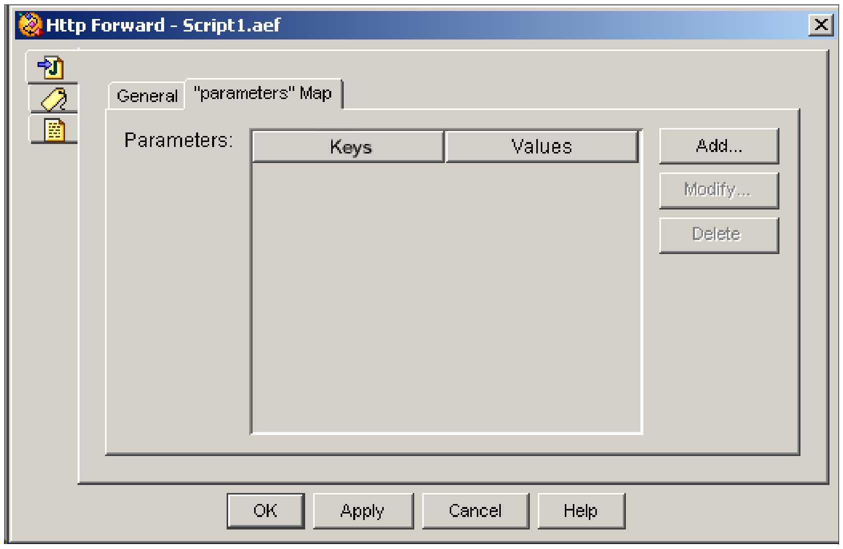
Table 2-46 describes the properties of the HTTP Forward customizer window—General tab.

Table 2-46 HTTP Forward Properties—General tab

Properties / Buttons	Description
Http Contact	Contact variable that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
URI Path	Relative URI path from either the ROOT or user-installed web applications. **Note** Web applications are installed in the Document repository under the folder \default\webapps\.

Figure 2-63 shows the Http Forward Customizer Window “Parameters” Map tab.

Figure 2-63 HTTP Forward Customizer Window—”Parameters” Map tab



Http Include Step

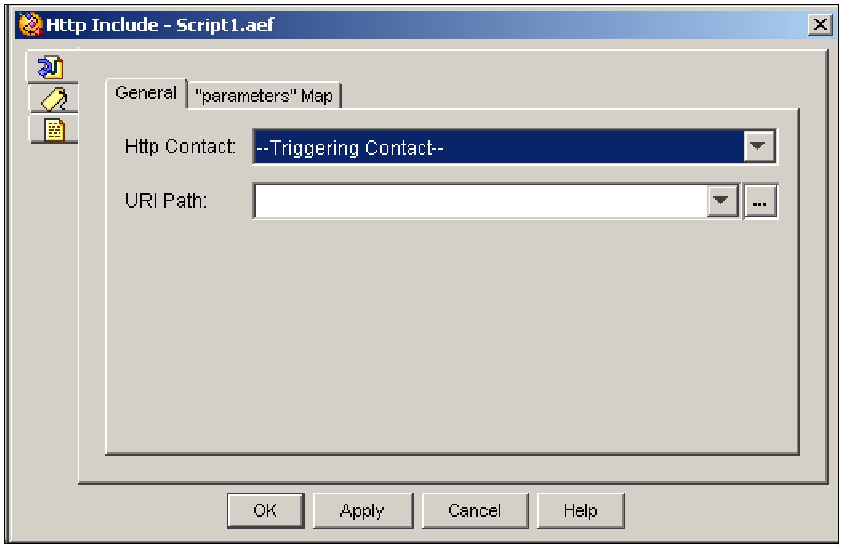
Use the Http Include step to forward an HTTP request to an internal URI deployed under the document repository \default\webapps\ROOT (the default web applications folder) or \default\webapps\<your web apps> folder. The URI can be a JSP page, which would permit dynamic content generation, or any other resource like an HTML page for static content generation.

Note The HTTP Include step is the same as the HTTP Forward step except that the HTTP response is not committed and the control is returned back to the script to continue appending to the HTTP response.

For further details, see the description of the Http Forward Step.

Figure 2-64 shows the customizer window for the Http Include step.

Figure 2-64 Http Include Customizer Window—General Tab



Http Contact: --Triggering Contact--
URI Path:

Table 2-48 describes the General properties of the Http Include customizer window.

Table 2-48 Http Include Properties—General Tab

Property	**Description**
Http Contact	Contact variable that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
URI Path	Relative URI path from either the ROOT or user-installed web applications. **Note** Web applications are installed in the Document repository under the folder \default\webapps\.

Figure 2-65 Http Include Customizer Window—"Parameters" Map tab

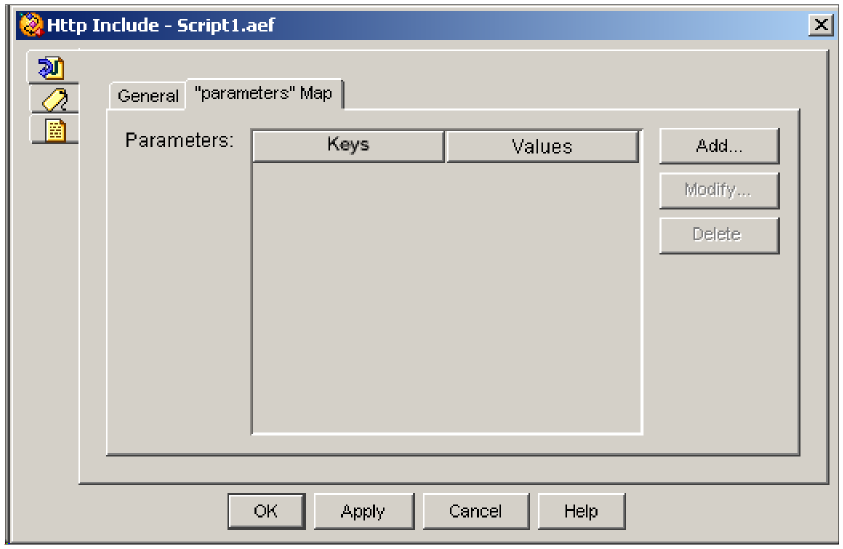


Table 2-49 Http Include Properties—Parameters Map tab

Property	Description
Parameters (Keys and Values)	Keywords in the JSP form and variables to be mapped to the keywords in the JSP form. The JSP page expects to receive this information in the form of a java.util.Hashtable stored in the **parameters** HTTP Request Header. The keys of the hash table will correspond to the keywords and the value is the variable's value at the time the step is executed.
Add / Modify (buttons)	Use these buttons to access the Parameter dialog box. Use the dialog to specify the following: • **Keyword**—Name for the Keyword. • **Variable**—A variable to be mapped to the Keyword. When done, click **OK**.
Delete (button)	To remove JSP information, highlight a value in the list and click **Delete**.

Http Redirect Step

Use the Http Redirect step to redirect the browser to go to a specified URL instead of responding directly to an HTTP request.

You can, for example, use this step to redirect the browser to a different server.

A script should use only one of the following steps from the Http Contact palette to respond to an HTTP request:

- Http Redirect
- HTTP Forward

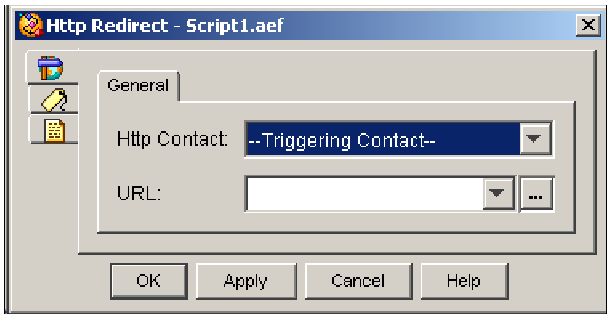
- Send Http Response

If you use the Http Redirect step, you should not use either a HTTP Forward or Send Response step in the same sequence within a script, because using any one of these three steps moves the Http Contact into a final state when the response is returned back to the browser. The script cannot attempt to send another response back because one has already been sent. If, however, a script has some conditional logic, you can use one of these three steps in one condition and another in another condition.

Note You can also use the Reject and Terminate steps from the Contact palette as responses to an HTTP request. For more information, see "Contact Steps" section on page 2-34.

Figure 2-66 shows the customizer window for the Http Redirect step.

Figure 2-66 Http Redirect Customizer Window



Http Contact: --Triggering Contact--
URL:

Table 2-50 describes the properties of the Http Redirect customizer window.

Table 2-50 Http Redirect Properties

Property	Description
Http Contact	Contact that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
URL	Variable containing—or an expression resolving to—a URL to which the browser is redirected.

Send Http Response Step

Use the Send Http Response step to send a document to a browser.

Note You should upload all documents to the Document repository and reference them using the expression language command DOC[filename]. Storing documents in the repository allows them to be synchronized across all Cisco Unified CCX servers in the cluster and included in server backups.

The document, which is stored in a document variable, must be ready to be sent. You may, for example, place the Keyword Transform Document step before this step in your script to update the document with dynamic information before it is sent to a user.

A script should use only one of the following steps from the Http Contact palette to respond to an HTTP request:

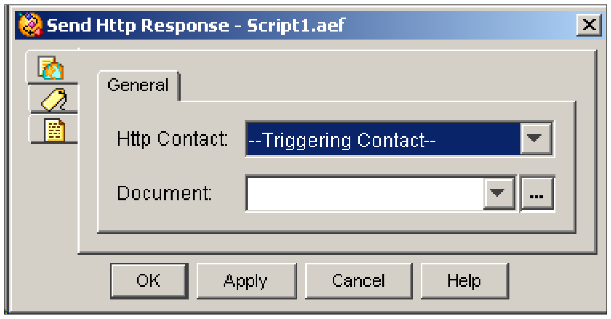
- Http Redirect
- Http Forward
- Send Http Response

If you use the Send Http Response step, you should not use either a Http Redirect or Http Forward step in the same sequence within a script, because using any one of these three steps moves the Http Contact into a final state when the response is returned back to the browser. The script cannot attempt to send another response back because one has already been sent. If, however, a script has some conditional logic, you can use one of these three steps in one condition and another in another condition.

Note You can also use the Reject and Terminate steps from the Contact palette as responses to an HTTP request. For more information, see "Contact Steps" section on page 2-34.

Figure 2-67 shows the customizer window for the Send Http Response step.

Figure 2-67 Send Response Customizer Window



Http Contact: --Triggering Contact--
Document:

Table 2-51 describes the properties of the Send Http Response customizer window.

Table 2-51 Send Http Response Properties

Property	Description
Http Contact	Contact variable that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
Document	Variable or expression indicating the location of the document to be sent.

Set Http Contact Info Step

Use the Set Http Contact Info step to set the values of HTTP headers and cookies in an HTTP response.

Note Use this step in advanced scripts. You need specialized knowledge of HTTP to make this step useful.

The Set Http Contact Info customizer window contains the following three tabs:

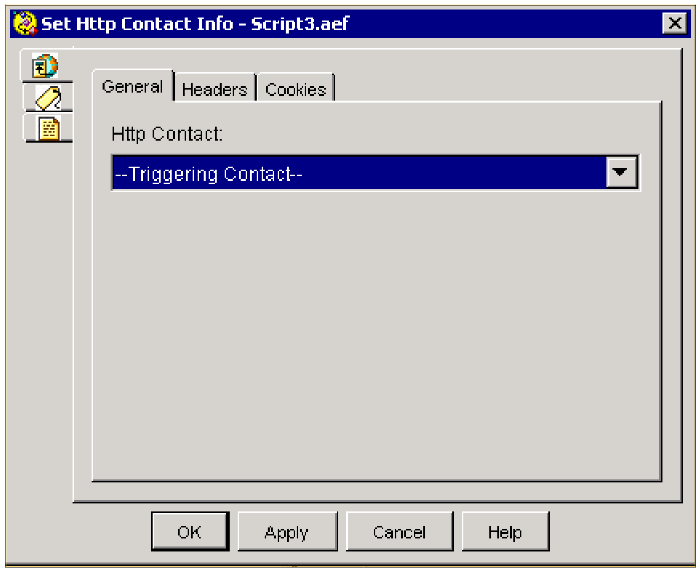
- General tab (Set Http Contact step)
- Headers tab (Set Http Contact step)
- Cookies tab (Set Http Contact step)

The following sections describe these tabs.

General tab (Set Http Contact step)

Use the General tab of the Set Http Contact Step customizer window to specify the Contact variable that triggers the execution of the step.

Figure 2-68 Set Http Contact Customizer Window—General Tab



Http Contact:
--Triggering Contact--

Table 2-52 describes the property of the General tab of the Set Http Contact Info customizer window.

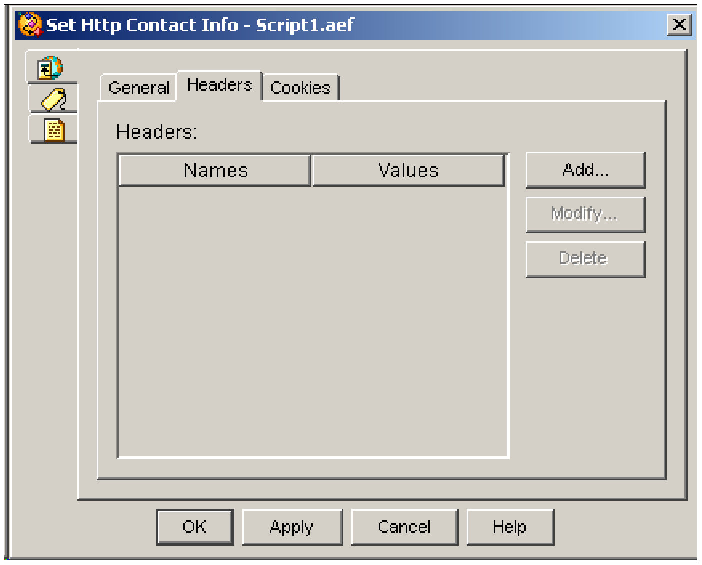
Table 2-52 Set Http Contact Info Property

Property	Description
Http Contact	Contact variable that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.

Headers tab (Set Http Contact step)

Use the Headers tab of the Set Http Contact Info step to map each HTTP header to a local variable or a valid expression from which the header value will be obtained when the step executes.

Figure 2-69 Set Http Contact Customizer Window—Headers Tab



HTTP headers contain general information such as the type of browser or the version of HTTP used. When you design a browser script, you can add information to the HTTP header to identify the HTTP version or browser type you are using.

You can also, for example, use this tab to control how long the browser maintains a cached copy of a document, by setting a value for the Expires header.

Table 2-53 describes the properties of the Headers tab of the Set Http Contact Info customizer window.

Table 2-53 Set Http Contact Info—Headers Tab Properties

Properties / Buttons	Description
Headers (Names and Values)	The Names of the headers and the corresponding variables that map to the headers.
Add / Modify (buttons)	Use these buttons to access the Header dialog box. Use the dialog to specify the following: • **Header**—Name for the HTTP header. • **Variable**—A variable or expression to be mapped to the HTTP header When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove Header information, highlight a value in the list and click **Delete**.

Cookies tab (Set Http Contact step)

Use the Cookies tab of the Set Http Contact Info customizer window to map each cookie to a local variable from which the cookie value will be obtained when the step executes.

A cookie is information maintained by the browser that is sent by the HTTP server in response to an HTTP request.

Figure 2-70 Set HTTP Contact Customizer Window—Cookies Tab

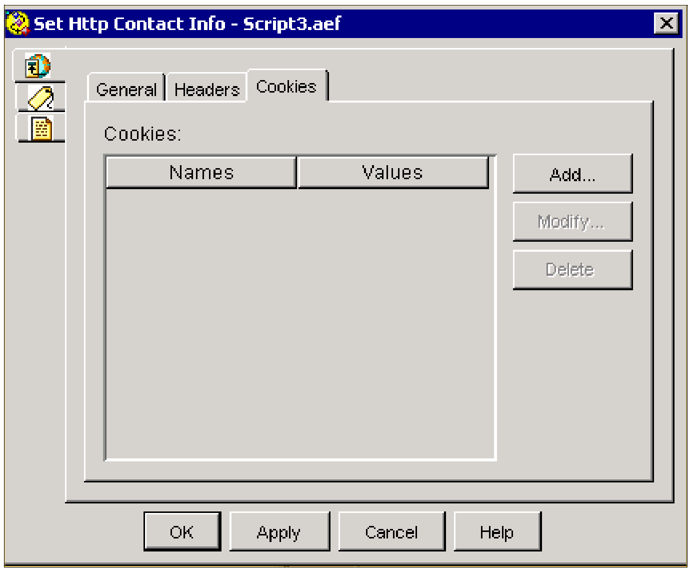


Table 2-54 describes the property of the Cookies tab of the Set Http Contact Info customizer window.

Table 2-54 Set Http Contact Info—Cookies Tab Property

Property	Description
Cookies (Names and Values)	The names of the cookies and the corresponding variables to which they are mapped.
Add / Modify (buttons)	Use these buttons to access the Cookie dialog box. Use the dialog to specify the Cookie attributes and click OK. **Note** For complete details about the Cookie Attributes, see Table 2-55. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove Cookie information, highlight a value in the list and click **Delete**.

Table 2-55 describes the attributes of the Cookie dialog box.

Table 2-55 Cookie Dialog Box Attributes

Cookie Attributes	Description
Name	Alphanumeric string that identifies the cookie. Excluded characters include the semicolon, comma, and space. Names beginning with $ are reserved and cannot be used by scripts. Each cookie begins with a NAME = VALUE pair.
Value	Variable to map to the cookie identified in Name. Each cookie begins with a NAME = VALUE pair.
Comment	Variable containing an alphanumeric string that identifies the purpose of the cookie to the user. The user may use this information to decide whether to establish or continue a session with this cookie.
Domain	Variable that contains the domain of hosts that can set a cookie. The domain must contain at least two periods. The default value is the host name of the server that generated the cookie response.
Max Age	Variable that stores the number of seconds before the cookie expires. A positive value represents the number of seconds before the cookie expires. A negative value means the cookie will be deleted when the web browser closes. A value of 0 causes the cookie to be deleted immediately.
Path	Variable that specifies the subset of URLs in a domain for which the cookie is valid. The default path is the path of the document described by the header that contains the cookie.
Secure	Variable that specifies the secure cookie, so that it will be transmitted only to SSL[1] servers.
Version	Variable that contains the decimal integer that identifies the version of the state management specification to which the cookie conforms.

1. SSL = Secure Sockets Layer

Media Steps

The steps in the Media palette of the Cisco Unified CCX Editor provide script designers with a way to process media interactions with callers.

Media interactions can include playing or recording prompts and acquiring Dual Tone Multi-Frequency (DTMF) or speech input.

Note When developing scripts that prompt callers, remember to accommodate the needs of hearing-impaired customers. To make your application fully accessible to these callers, set up scripts to interact with devices such as a Telecommunications Relay Service (TRS) or Telecommunications Device for the Deaf (TTY/TDD).

All media steps work with either DTMF or Automatic Speech Recognition (ASR) input with the exception of the Voice Browser step, which works only with ASR.

Note Before using an ASR or TTS script as a Cisco Unified CCX application, you should first validate the script against the MRCP ASR and TTS vendor's list of supported capabilities.

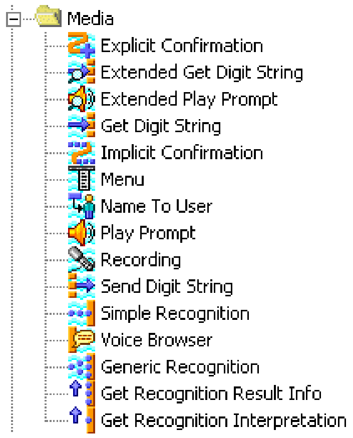
This section contains the following topics:

- Explicit Confirmation Step
- Extended Get Digit String Step
- Extended Play Prompt Step
- Get Digit String Step
- Implicit Confirmation Step
- Menu Step
- Name To User Step
- Play Prompt Step
- Recording Step
- Send Digit String Step
- Simple Recognition Step
- Voice Browser Step
- Generic Recognition Step
- Get Recognition Result Info Step
- Get Recognition Interpretation Step
- User Steps

Figure 2-71 shows the steps in the Media palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Note The Name to User and Voice Browser steps are only available for the Cisco UnifiedIP IVR or Cisco Unified CCX Premium license options.

Figure 2-71 Media Palette Steps



Note If you apply any of these Media steps to a contact that is not associated with a Media channel (also called a *dialog* channel), a ChannelUnsupportedException results.

If You Are Upgrading Scripts From a Previous Cisco Unified CCX Release

If you are upgrading from Cisco Unified CCX 2.x and are using queuing scripts to interface with Cisco UnifiedICME software, please note that all media steps must be set to "interruptible" if you want the VRU script to be interruptible by the Cisco UnifiedICME software when an agent becomes available. If not, then make sure to configure the VRU script as not interruptible in the Cisco UnifiedICME system.

In Cisco Unified CCX 2.x, **No** was the default setting for Media steps interruptibility. Scripts upgraded from 2.x will retain this setting, so you must manually change the settings to Yes so that the interface with Cisco UnifiedICME software can work properly.

Note If you are upgrading from Cisco Unified CCX 3.x, you do not need to edit your queuing scripts, as the default interruptibility setting for Media steps in that release is Yes.

Explicit Confirmation Step

Use the Explicit Confirmation step to confirm an explicit response to a prompt.

The Explicit Confirmation step is defined with a default grammar that accepts the following:

- **DTMF**—Use 1 for yes and 2 for no.
- **ASR**—Use typical yes or no grammar in proper language.

You can override the default grammar with a caller-defined grammar. (For more information about requirements for defining grammars, see Grammar Steps.)

Note Tags for grammars are case-sensitive.

The customizer window of the Explicit Confirmation step contains three tabs:

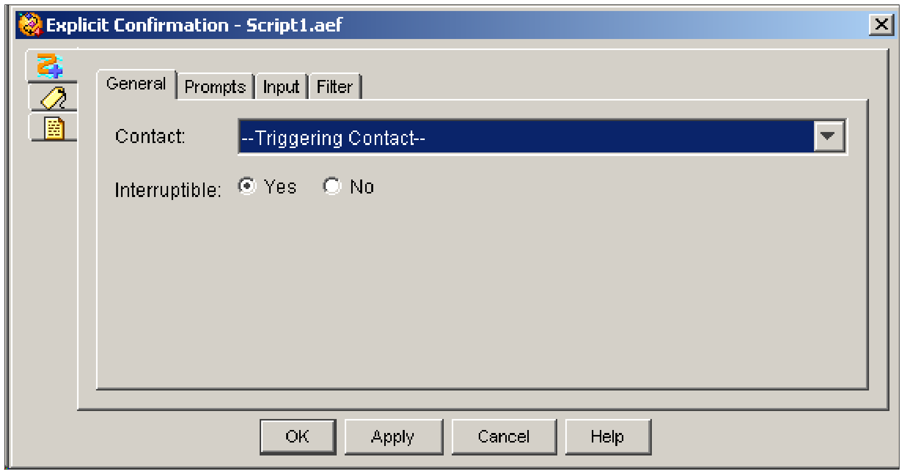
- General tab (Explicit Confirmation step)
- Prompts tab (Explicit Confirmation step)
- Input tab (Explicit Confirmation step)
- Filter tab (Explicit Confirmation step)

The following sections describe these tabs.

General tab (Explicit Confirmation step)

Use the General tab of the Explicit Confirmation customizer window to select the contact on which to perform the confirmation, and to set the Interruptible option.

Figure 2-72 Explicit Confirmation Customizer Window—General Tab



Contact: --Triggering Contact--

Interruptible: [✓ Yes] No

Table 2-56 describes the properties of the General tab of the Explicit Confirmation customizer window.

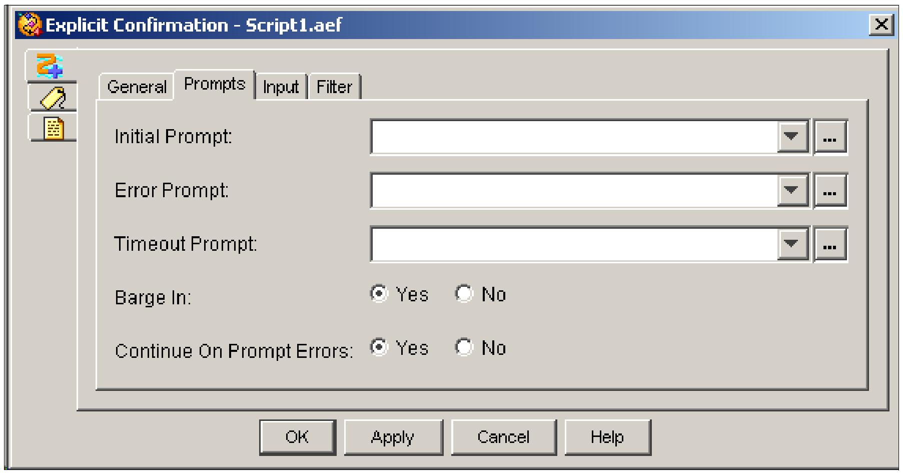
Table 2-56 Explicit Confirmation Properties—General Tab

Properties / Buttons	Description
Contact	Contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified.
Interruptible	**Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.

Prompts tab (Explicit Confirmation step)

Use the Prompts tab of the Explicit Confirmation customizer window to specify initial, error, and timeout prompts and to set Barge In and Continue On Prompt Errors options.

Figure 2-73 Explicit Confirmation Customizer Window—Prompts Tab



Initial Prompt:
Error Prompt:
Timeout Prompt:
Barge In: [✓ Yes] No
Continue On Prompt Errors: [✓ Yes] No

Table 2-57 describes the properties of the Prompts tab of the Explicit Confirmation customizer window.

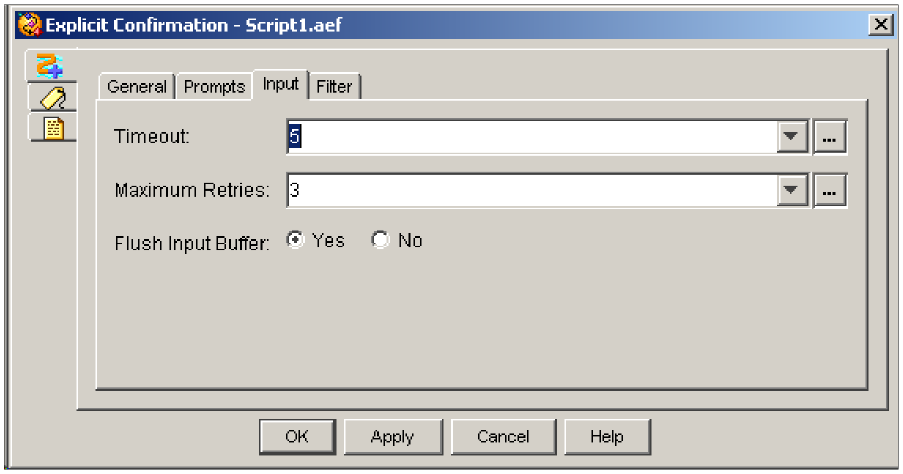
Table 2-57 Explicit Confirmation Properties—Prompts Tab

Properties / Buttons	Description
Initial Prompt	Variable or expression indicating the first prompt to be played back.
Error Prompt	Variable or expression indicating the prompt to be played in the event of an input error. (Optional; leave blank to use the system default.)
Timeout Prompt	Variable or expression indicating the prompt to be played in the event of a timeout. (Optional; leave blank to use the system default.)
Barge In	**Yes**—The caller can interrupt the prompt. **No**—The prompt must complete playback before the caller can respond.
Continue on Prompt Errors	**Yes**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **No**—An exception results, which can then be handled in the script.

Input tab (Explicit Confirmation step)

Use the Input tab of the Explicit Confirmation customizer window to set timeout duration, maximum number of retries, and Flush Input Buffer options. Also use the Input tab to specify a grammar expression.

Figure 2-74 Explicit Confirmation Customizer Window—Input Tab



Timeout: 5

Maximum Retries: 3

Flush Input Buffer: [✓ Yes] No

Table 2-58 describes the properties of the Input tab of the Explicit Confirmation customizer window.

Table 2-58 Explicit Confirmation Properties—Input Tab

Properties / Buttons	Description
Timeout (in sec)	Amount of time that the step waits for a response before timing out.
Maximum Retries	Number of times a new entry can be entered after a timeout or invalid key.
Flush Input Buffer	**Yes**—The system erases previously entered input before capturing caller input. **No**—The system does not erase previously entered input before capturing caller input.

Filter tab (Explicit Confirmation step)

Use the Filter tab of the Explicit Confirmation customizer window to specify an optional grammar expression to be used for recognizing Yes or No.

Figure 2-75 Explicit Confirmation Customizer Window—Filter Tab

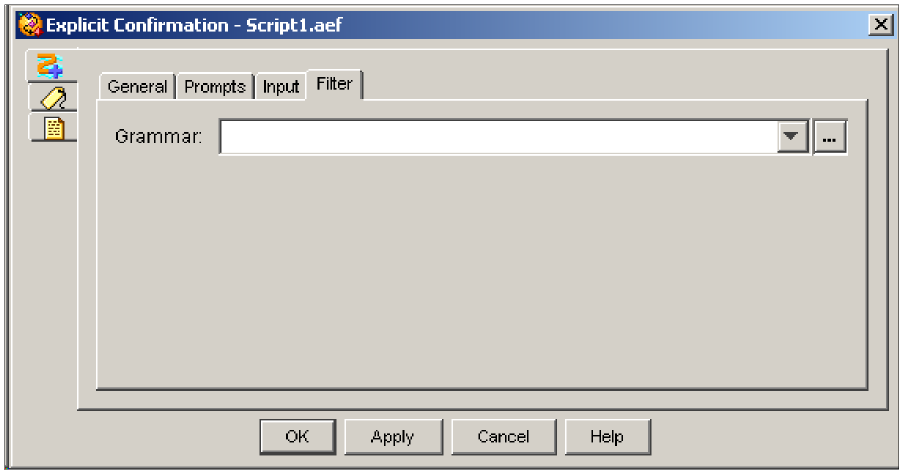


Table 2-59 describes the properties of the Filter tab of the Explicit Confirmation customizer window.

Table 2-59 Explicit Confirmation Properties—Filter Tab

Properties / Buttons	Description
Grammar	Variable or expression indicating the optional grammar expression to be used for recognizing Yes or No. If supplied, the grammar will override the system default grammar.

Extended Get Digit String Step

Use the **Extended** Get Digit String step to capture either spoken or DTMF entries based on information defined at run time. If you are using ASR, digits can be spoken in the particular language of the call.

This step waits for input until the caller does one of the following:

- Presses the terminating key (DTMF only)
- Exhausts the maximum number of retries
- Enters the maximum number of keys (DTMF only)
- Does not respond before the timeout length is reached

Note When any previous escalating prompt in the script enters the Extended Get Digit String step, it is reset to the first prompt in its list.

The Extended Get Digit String step behaves exactly like the Get Digit String step, with two exceptions. First, with the Extended Get Digit String step, you can enter a Boolean expression for the Interruptible and Clear DTMF Buffer on Retry fields. Second, although the same limits apply to both steps, most of the Extended Get Digit String properties can be defined using variables or grammar expressions that can be changed while the script is running.

The terminator key and the cancel key support Character objects as well as String objects. When a string is passed as the terminator or cancel key, only the first character of the string is used. The special string "none" can be used to specify that either no terminator key or no cancel key is required.

The Extended Get Digit String step produces the following output branches:

- Successful—Input was valid.
- Timeout—After the retry limit was reached, the last try timed out.
- Unsuccessful—After the retry limit was reached, an invalid key was pressed.

Note When an error output branch is reached, all collected digits are returned and stored in a specified input variable.

The customizer window of the Extended Get Digit step contains four tabs:

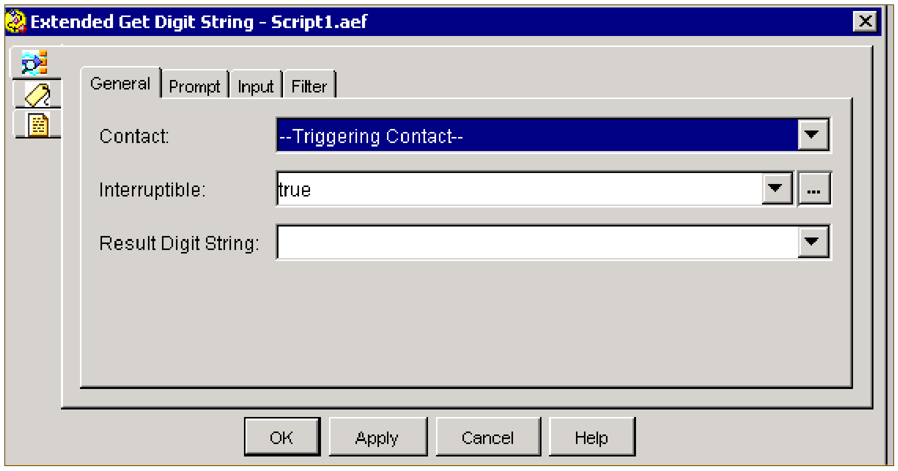
- General tab (Extended Get Digit String step)
- Prompt tab (Extended Get Digit String step)
- Input tab (Extended Get Digit String step
- Filter tab (Extended Get Digit String step

The following sections describe these tabs.

General tab (Extended Get Digit String step)

Use the General tab of the Extended Get Digit String customizer window to choose a contact on which to perform the Get Digit String operation, specify the variable that will hold the resulting digit string, and set the Interruptible option.

Figure 2-76 Extended Get Digit String Customizer Window—General Tab



Contact: --Triggering Contact--

Interruptible: true

Result Digit String:

Table 2-60 describes the properties of the General tab of the Extended Get Digit String customizer window.

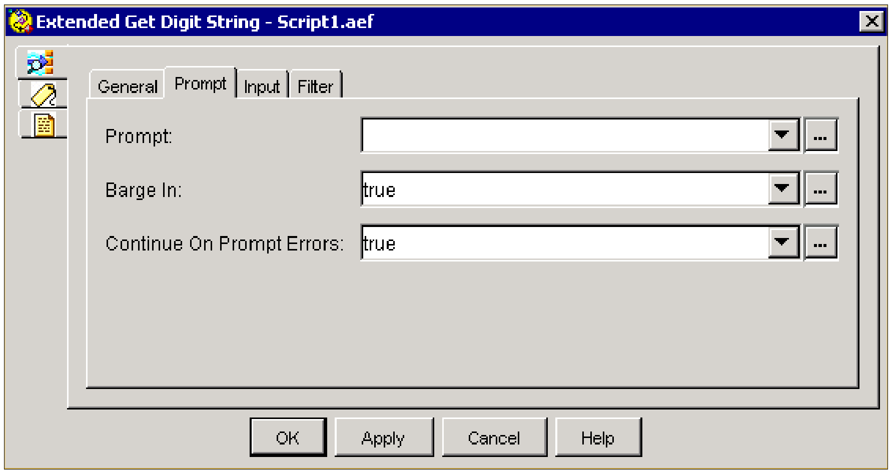
Table 2-60 Extended Get Digit String—General Tab Properties

Properties / Buttons	Description
Contact	Contact variable that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified.
Interruptible	A Boolean variable—or an expression resolving to a Boolean variable—indicating whether the step is interruptible: **True**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **False**—The step must complete before any other process can execute.
Result Digit String	Variable that stores the resulting digit string.

Prompt tab (Extended Get Digit String step)

Use the Prompt tab of the Extended Get Digit String customizer window to choose the prompt input variable and set Barge In and Continue On Prompt Error options.

Figure 2-77 Extended Get Digit String Customizer Window—Prompt Tab



Prompt:
Barge In: true
Continue On Prompt Errors: true

Table 2-61 describes the properties of the Prompt tab of the Extended Get Digit String customizer window.

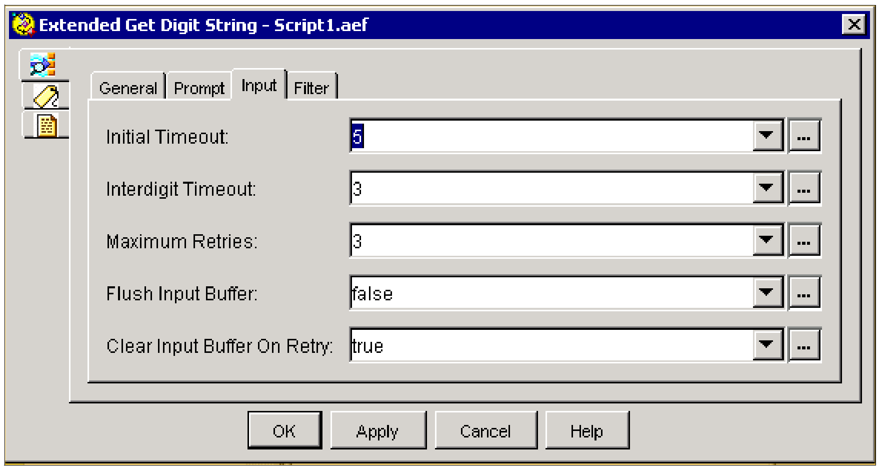
Table 2-61 Extended Get Digit String—Prompt Tab Properties

Properties / Buttons	Description
Prompt	Variable or expression indicating the prompt used to prompt the caller to input a digit string.
Barge In	A Boolean variable—or an expression resolving to a Boolean variable—indicating whether the step is interruptible. **True**—The caller can interrupt the prompt. **False**—The prompt must complete playback before the caller can respond.
Continue on Prompt Error	A Boolean variable—or an expression resolving to a Boolean variable—indicating what the step should do on prompt error. **True**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **False**—An exception results, which can then be handled in the script.

Input tab (Extended Get Digit String step

Use the Input tab of the Extended Get Digit String customizer window to set input properties.

Figure 2-78 Extended Get Digit String Customizer Window—Input Tab



Initial Timeout: 5
Interdigit Timeout: 3
Maximum Retries: 3
Flush Input Buffer: false
Clear Input Buffer On Retry: true

Table 2-62 describes the properties of the Input Tab of the Extended Get Digit String customizer window.

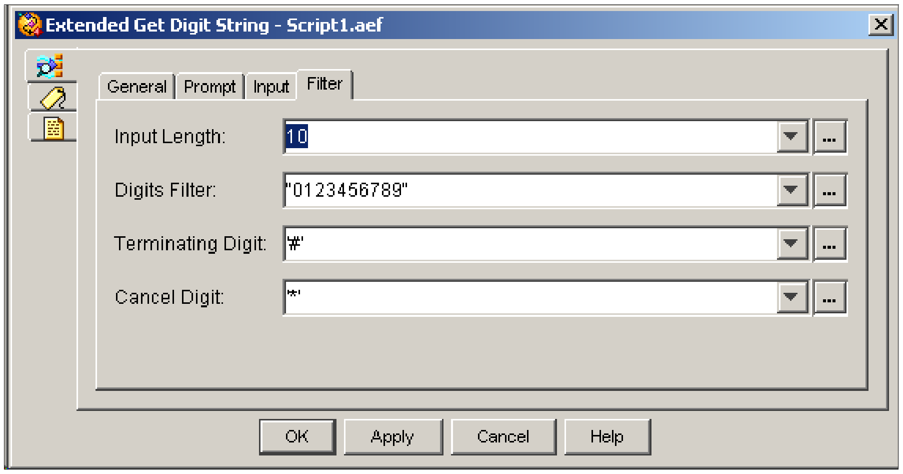
Table 2-62 Extended Get Digit String—Input Tab Properties

Properties / Buttons	Description
Input Length	Variable or expression indicating the maximum number of digits or characters. When this limit is reached, the script uses input already entered.
Initial Timeout (sec)	Variable or expression indicating the amount of time the system will wait for initial input from the caller.
Interdigit Timeout (sec)	Variable or expression indicating the amount of time that the system will wait for the caller to enter the next digit, after receiving initial input from the caller.
Maximum Retries	Variable or expression indicating the number of times the entry can be started over after a timeout or an invalid key. A “0” value means that no retry is allowed; in this case, the script must handle the retry scenario.
Flush Input Buffer	A Boolean variable—or an expression resolving to a Boolean variable—indicating whether the system should flush the input buffer. **True**—The system erases previously entered input before capturing caller input. **False**—The system does not erase previously entered input before capturing caller input.
Clear Input Buffer on Retry	A Boolean variable—or an expression resolving to a Boolean variable—indicating whether the system should clear the DTMF buffer. **True**—The script clears the DTMF buffer before each retry. **False**—The script does not clear the DTMF buffer before each retry.

Filter tab (Extended Get Digit String step

Use the Filter tab of the Extended Get Digit String customizer window to set the DTMF Controls: the input filter values, the terminating key, and the cancel key.

Figure 2-79 Extended Get Digit String Customizer Window—Filter Tab



Input Length: 10
Digits Filter: "0123456789"
Terminating Digit: '#'
Cancel Digit: '*'

Table 2-63 describes the properties of the Filter tab Extended Get Digit String customizer window.

Table 2-63 Extended Get Digit String Properties—Filter Tab

Properties / Buttons	Description
Input Length	Variable indicating the maximum number of digits or characters. When this limit is reached, the step stops accumulating digits and returns.
Digits Filter	Variable or expression indicating the valid DTMF keys that can be entered in the DTMF keypad (excluding the terminating and cancel keys).
Terminating Digit	Variable or expression indicating the key the caller uses to signal the end of caller input. The terminating key overrides the Maximum Input Length to terminate input. Leaving this field empty or setting it to "none" or "n" means that no terminating key exists. **Note** This field supports character expressions and strings. For string use only the first character of the string.
Cancel Digit	Variable or expression indicating the key the caller uses to cancel input to start over. Leaving this field empty or setting it to "none" or "n" means that no cancel key exists. **Note** This field supports character expressions and strings. For string use only the first character of the string.

Extended Play Prompt Step

Use the Extended Play Prompt step to play prompts back to the caller.

The Extended Play Prompt step behaves exactly like the Play Prompt step, with the exception that you can use the Expression Editor to assign values to the Interruptible, Barge In, Continue On Prompt Errors, and Flush Input Buffer options for greater script flexibility.

Note When any previous escalating prompt in the script enters the Extended Play Prompt step, it is reset to the first prompt in its list.

The customizer window of the Extended Play Prompt step contains three tabs:

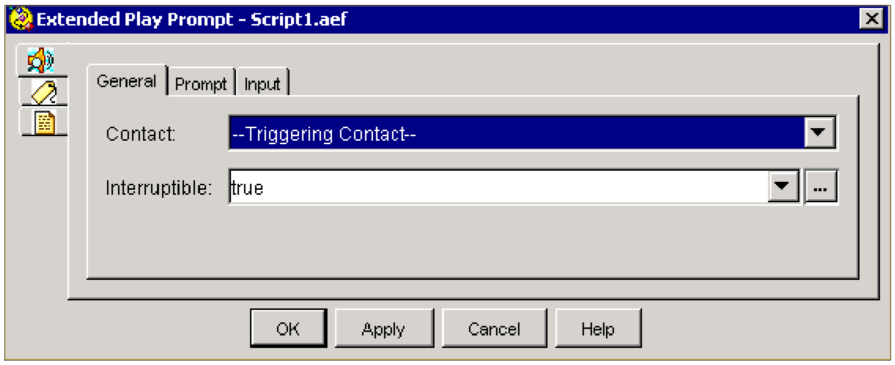
- General tab (Extended Play Prompt step)
- Prompt tab (Extended Play Prompt step)
- Input tab (Extended Play Prompt step)

These tabs are described below.

General tab (Extended Play Prompt step)

Use the General tab of the Extended Play Prompt customizer window to specify the triggering contact and set the Interruptible option.

Figure 2-80 Extended Play Prompt Customizer Window—General Tab



Contact: --Triggering Contact--

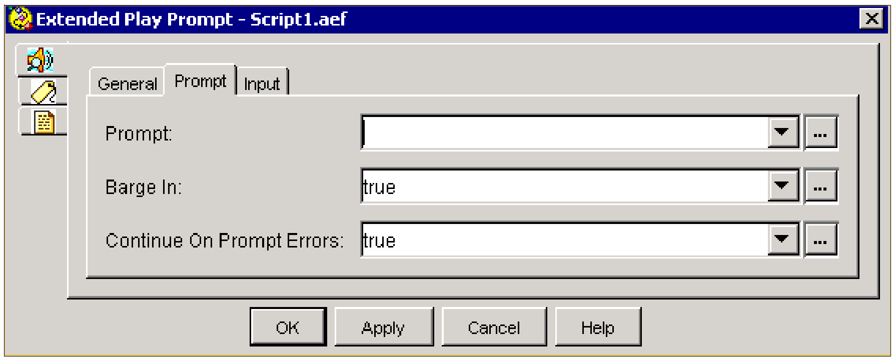
Interruptible: true

Table 2-64 describes the properties of the General tab of the Extended Play Prompt customizer window.

Prompt tab (Extended Play Prompt step)

Use the Prompt tab of the Extended Play Prompt customizer window to select the Prompt variable and set the Barge In and Continue on Prompt Error options.

Figure 2-81 Extended Play Prompt Customizer Window—Prompt Tab


Extended Play Prompt - Script1.aef

Prompt:
Barge In: true
Continue On Prompt Errors: true

Table 2-65 describes the properties of the Prompt tab of the Extended Play Prompt customizer window.

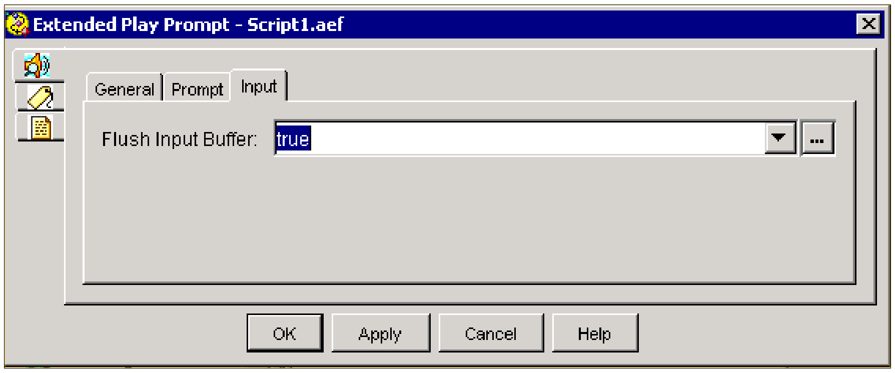
Table 2-65 Extended Play Prompt—Prompt Tab Properties

Properties / Buttons	Description
Prompt	A variable or an expression indicating which prompt is to be played.
Barge In	A Boolean variable—or an expression resolving to a Boolean variable—indicating whether the step is interruptible. **True**—The caller can interrupt the prompt. **False**—The prompt must complete playback before the caller can respond.
Continue on Prompt Error	A Boolean variable—or an expression resolving to a Boolean variable—indicating whether the step should continue on prompt error. **True**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **False**—An exception results, which can then be handled in the script.

Input tab (Extended Play Prompt step)

Use the Input tab of the Extended Play Prompt customizer window to configure the Flush Input Buffer option.

Figure 2-82 Extended Play Prompt Customizer Window—Input Tab


Extended Play Prompt - Script1.aef

Flush Input Buffer: true

Table 2-66 describes the property of the Input tab of the Extended Play Prompt customizer window.

Table 2-66 Extended Play Prompt—Input Tab

Properties / Buttons	Description
Flush Input Buffer	A Boolean variable—or an expression resolving to a Boolean variable—indicating whether the step should flush the input buffer. **True**—The system erases previously entered input before capturing caller input. **False**—The system does not erase previously entered input before capturing caller input.

Get Digit String Step

Use the Get Digit String step to capture either a DTMF or spoken digit string from the caller in response to a prompt. When using ASR, the prompt used should ask the caller to "press or say" the digit string, because this step automatically supports spoken digits as well as DTMF digits when executed with ASR.

The Get Digit String step waits for input until the caller does one of the following:

- Presses the terminating key (DTMF only)
- Exhausts the maximum number of retries
- Enters the maximum number of keys (DTMF only)
- Does not respond before the timeout length is reached

In ASR mode, the step will take an entire spoken digit string but will return the digit result only up the specified Input Length.

Users must speak digits sequentially; for example, 1 2 3 4 must be spoken "one-two-three-four" instead of "twelve thirty-four".

Note When any previous escalating prompt in the script enters the Get Digit String step, it is reset to the first prompt in its list.

The Get Digit String step provides three output branches:

- Successful—Input was valid.
- Timeout—After the retry limit was reached, the last try timed out.
- Unsuccessful—After the retry limit was reached, an invalid key was pressed or an invalid value was spoken.

Note In the event of an error, the accumulated digits are returned and saved in the specified variable before the script exits through the unsuccessful or timeout output branches.

The customizer window of the Get Digit String step contains four tabs:

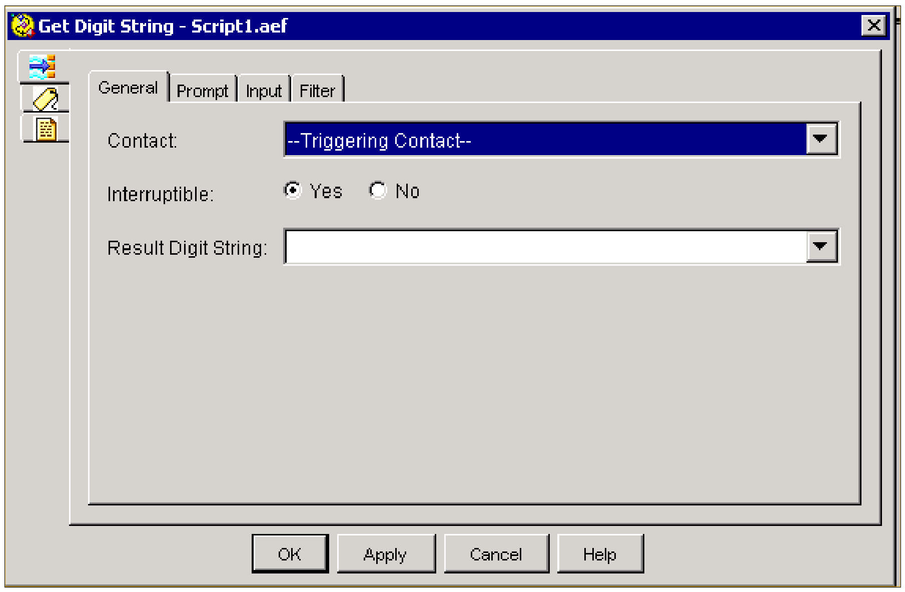
- General tab (Get Digit String step)l
- Prompt tab (Get Digit String)
- Input tab (Get Digit String)
- Filter tab (Get Digit String)

The following sections describe these tabs.

General tab (Get Digit String step)

Use the General tab of the Get Digit String step to choose the contact, specify the variable that will store the digit string, and specify whether or not the step is interruptible by external events.

Figure 2-83 Get Digit String Customizer Window—General Tab



Contact: --Triggering Contact--

Interruptible: [✓ Yes] No

Result Digit String:

Table 2-67 describes the properties of the General tab of the Get Digit String customizer window.

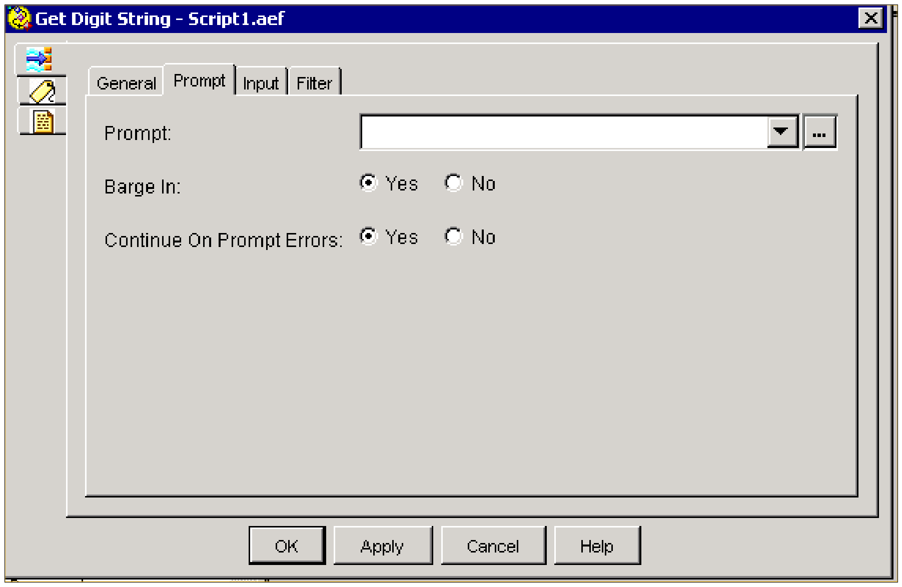
Table 2-67 Get Digit String Properties—General Tab

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
Interruptible	Radio button. **Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.
Result Digit String	Variable indicating the name of the variable that stores the digits that the caller enters.

Prompt tab (Get Digit String)

Use the Prompt tab of the Get Digit String customizer window to specify a prompt, and to set Barge In and Continue on Prompt Errors options.

Figure 2-84 Get Digit String Customizer Window—Prompt Tab



Prompt:

Barge In: [✓ Yes] No

Continue On Prompt Errors: [✓ Yes] No

Table 2-68 describes the properties of the Prompt tab of the Get Digit String customizer window.

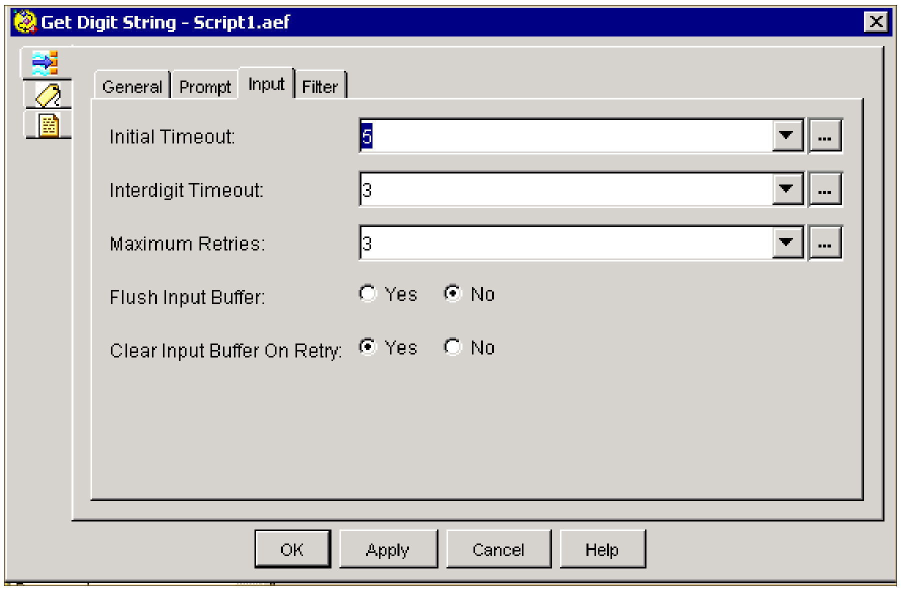
Table 2-68 Get Digit String Properties—Prompt Tab

Properties / Buttons	Description
Prompt	Variable or expression indicating the prompt to be played back.
Barge In	Radio button. **Yes**—The caller can interrupt the prompt. **No**—The prompt must complete playback before the caller can respond.
Continue on Prompt Errors	Radio button. **Yes**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **No**—An exception results, which can then be handled in the script.

Input tab (Get Digit String)

Use the Input tab of the Get Digit String customizer window to set conditions for receiving caller input.

Figure 2-85 Get Digit String Customizer Window—Input Tab



Initial Timeout: 5

Interdigit Timeout: 3

Maximum Retries: 3

Flush Input Buffer: Yes [✓ No]

Clear Input Buffer On Retry: [✓ Yes] No

Table 2-69 describes the properties of the Input tab of the Get Digit String customizer window.

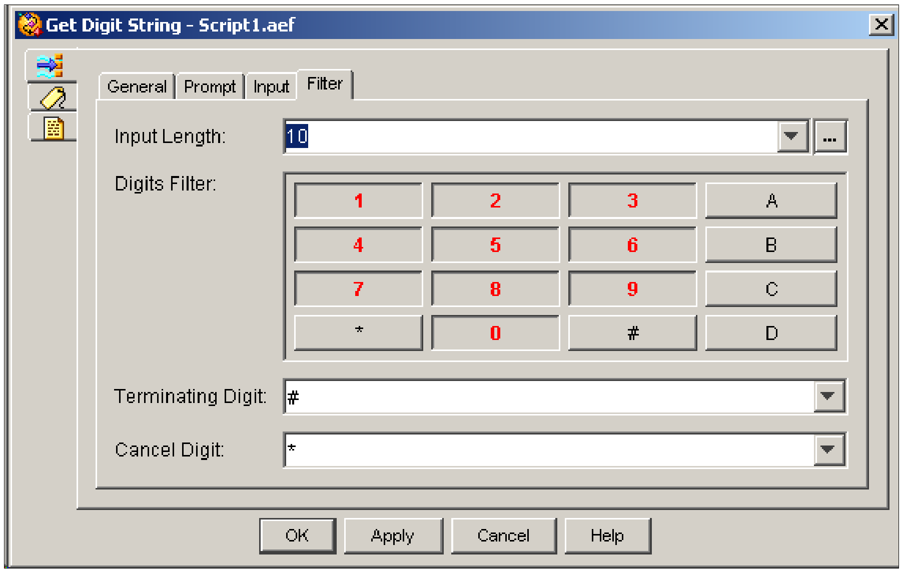
Table 2-69 Get Digit String Properties—Input Tab

Properties / Buttons	Description
Initial timeout (in sec)	Variable indicating the amount of time the system waits for initial input from the caller.
Interdigit timeout (in sec)	Variable indicating the amount of time that the system waits for the caller to enter the next digit, after receiving initial input from the caller (DTMF).
Maximum Retries	Variable indicating the number of times a new entry can be entered after a timeout or invalid key. After the maximum number of retries is reached, the step continues on the Timeout or Unsuccessful output branch, depending on whether the last try timed out or an invalid key was entered. On retry due to an invalid key, a system prompt plays if none is specified. **Note** A “0” value means that no retry is allowed; in this case, the script must handle the retry scenario.
Flush Input Buffer	Radio button. **Yes**—The system erases previously entered input before capturing caller input. **No**—The system does not erase previously entered input before capturing caller input.
Clear Input Buffer	Radio button. **Yes**—The step clears the DTMF buffer before each retry. **No**—The step does not clear the DTMF buffer before each retry.

Filter tab (Get Digit String)

Use the Filter tab of the Get Digit String step to specify digits that can be accepted from the caller.

Figure 2-86 Get Digit String Customizer Window—Filter Tab


Get Digit String - Script1.aef

Input Length: 10

Digits Filter: 1 2 3 A 4 5 6 B 7 8 9 C * 0 # D

Terminating Digit: #

Cancel Digit: *

Table 2-70 describes the property of the Filter tab of the Get Digit String customizer window.

Table 2-70 Get Digit String Properties—Filter Tab

Properties / Buttons	Description
Input Length	Variable indicating the maximum number of digits or characters. When this limit is reached, the step stops accumulating digits and returns.
Digits filter	Specifies the digits that you want to accept from the caller (excluding the terminating and cancel keys). If the caller enters digits that you do not choose, the system plays an error prompt for the caller and retries the Input step until the maximum numbers of retries is reached. At that time, the Unsuccessful output branch executes.
Terminating Digit	Variable indicating the key used to indicate the end of caller input (DTMF only). The terminating key overrides the Maximum Input Length to terminate input.
Cancel Digit	Variable indicating the key the caller presses to start over.

The Get Digit String customizer window closes, and the name of the triggering contact and the result digit string variable appear next to the Get Digit String step icon in the Design pane of the Cisco Unified CCX Editor.

Implicit Confirmation Step

Use the Implicit Confirmation step to confirm an action without having to ask a question. This step is typically used in speech-enabled Solutions.

A prompt explaining the action to be taken is played back and the system waits a configured number of seconds for input from the caller. If the caller presses any DTMF digits or speaks anything before the configured timeout, the confirmation is considered to have failed, and an Explicit Confirmation step should be used.

For DTMF media, the caller can fail the confirmation only by entering DTMF digits.

Note When any previous escalating prompt in the script enters the Implicit Confirmation step, it is reset to the first prompt in its list.

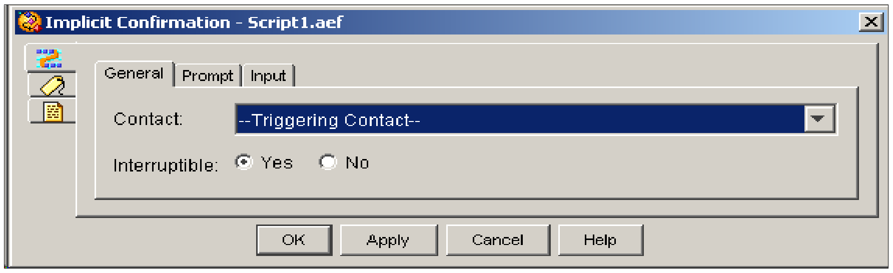
When a valid string of digits is received, a prompt plays the extension that will be dialed, based on the caller's input. The Implicit Confirmation step is usually configured to give the caller two seconds after hearing the prompt to decline confirmation before timeout.

Under the No output branch of the Implicit Confirmation step, an If step tracks the number of times the confirmation is attempted before the script moves to a subsequent step.

If the extension played back to the caller is accurate and the caller makes no effort to stop the operation, the Yes output branch executes, and a Call Redirect step attempts to connect the caller to the desired extension.

Figure 2-87 shows the customizer window for the Implicit Confirmation step.

Figure 2-87 Implicit Confirmation Customizer Window—General Tab



Contact: --Triggering Contact--

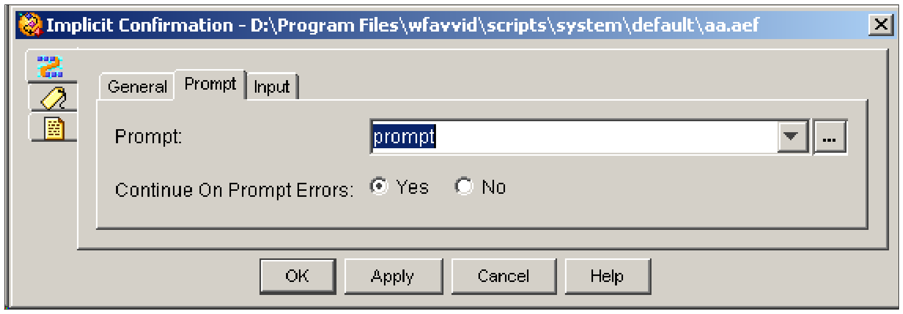
Interruptible: [✓ Yes] No

Table 2-71 describes the properties of the General tab of the Implicit Confirmation customizer window.

Table 2-71 Implicit Confirmation General Tab Properties

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified.
Interruptible	Radio button. **Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.

Figure 2-88 Implicit Confirmation Customizer Window—Prompt Tab


Implicit Confirmation - D:\Program Files\wfavvid\scripts\system\default\aa.aef
General
Prompt
Input
OK
Apply
Cancel
Help

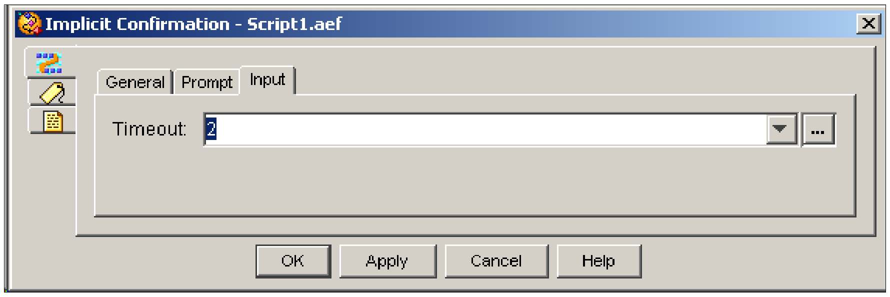
Prompt: prompt

Continue On Prompt Errors: [✓ Yes] No

Table 2-72 Implicit Confirmation Prompt Tab Properties

Properties / Buttons	Description
Prompt	Variable or expression indicating the prompt played to the caller.
Continue on Prompt Errors	Radio button. **Yes**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **No**—An exception results, which can then be handled in the script.

Figure 2-89 Implicit Confirmation Customizer Window—Input Tab



Timeout: 2

Table 2-73 Implicit Confirmation Input Tab Properties

Properties / Buttons	Description
Timeout	Variable indicating the number of seconds without a caller response before confirmation is considered successful. (Usual value is 2 seconds.)

Menu Step

Use the Menu step to provide a menu from which callers can choose a series of options. The Menu step receives a single digit entered or spoken by a caller and maps this entry to a series of option output branches. The system executes the steps that you add after each of these option output branches.

Note When any previous escalating prompt in the script enters the Menu step, it is reset to the first prompt in its list.

Although the Menu step combines the functionality of a Get Digit String step and a Switch step, it allows the caller to enter only one digit.

By default, the Menu step has the following output branches:

- Output 1
- Output 2
- Output 3
- Timeout
- Unsuccessful

You can add more output branches in the General tab of the Menu customizer window.

The Menu step retries for a timeout, an invalid digit entry (a digit that is not associated with any connections), or invalid spoken input. If the maximum number of retries is reached, the Menu step takes either the Timeout or Unsuccessful connection, depending on the reason for the latest failure.

The customizer window of the Menu step contains three tabs:

- General tab (Menu step)
- Prompt tab (Menu step)
- Input tab (Menu step)
- Filter tab (Menu step)

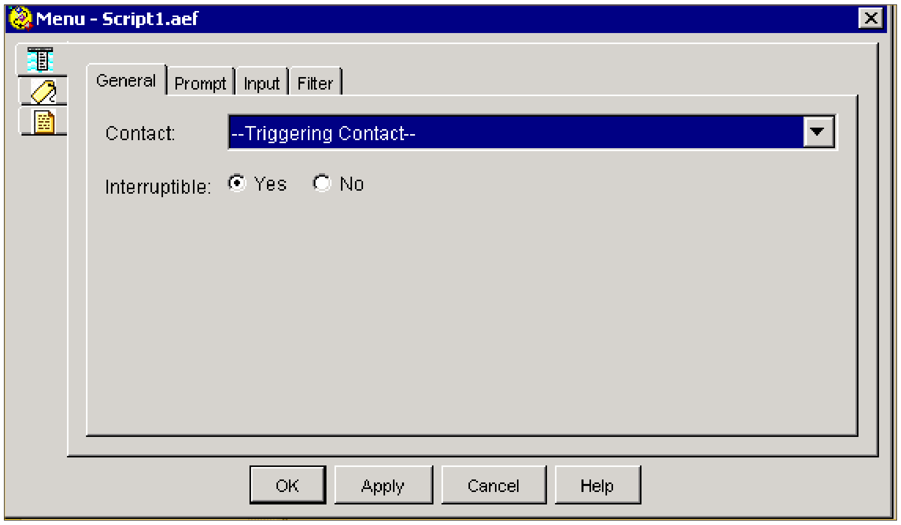
The following sections describe these tabs.

General tab (Menu step)

Use the General tab of the Menu customizer window to associate digits (typically entered by the caller from a telephone keypad or spoken in response to a speech-enabled prompt) with an output branch label.

You can associate multiple inputs with a single output branch label, but you can associate only one output branch label with a given input.

Figure 2-90 ***Menu Customizer Window—General Tab***



Contact: --Triggering Contact--

Interruptible: [✓ Yes] No

Table 2-74 describes the properties of the General tab of the Menu customizer window.

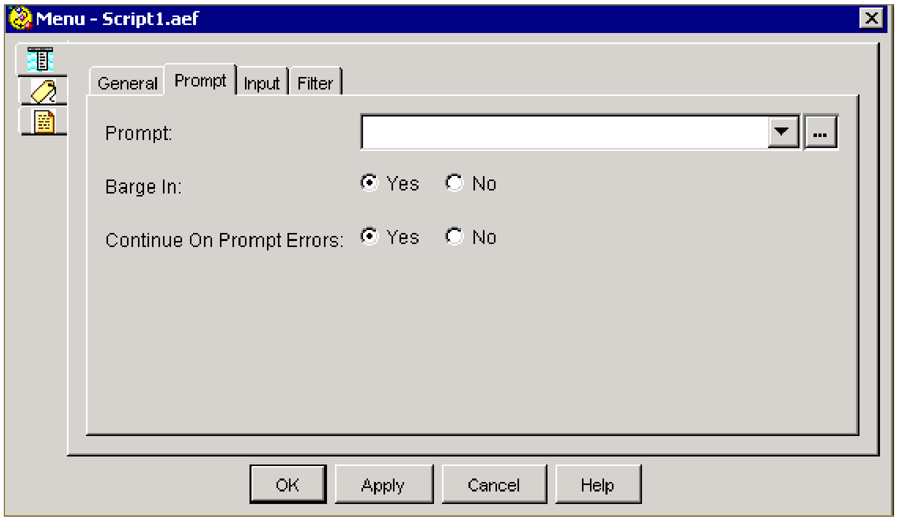
Table 2-74 ***Menu Properties—General Tab***

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified.
Interruptible (radio buttons)	Radio button. **Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.

Prompt tab (Menu step)

Use the Prompt tab of the Menu customizer window to choose the prompt to be played back, and to set the Barge In and Continue on Prompt Errors options.

Figure 2-91 Menu Customizer Window—Prompt Tab


Menu - Script1.aef

General | Prompt | Input | Filter

Prompt:

Barge In: [✓ Yes] No

Continue On Prompt Errors: [✓ Yes] No

Table 2-75 describes the properties of the Prompt tab of the Menu customizer window.

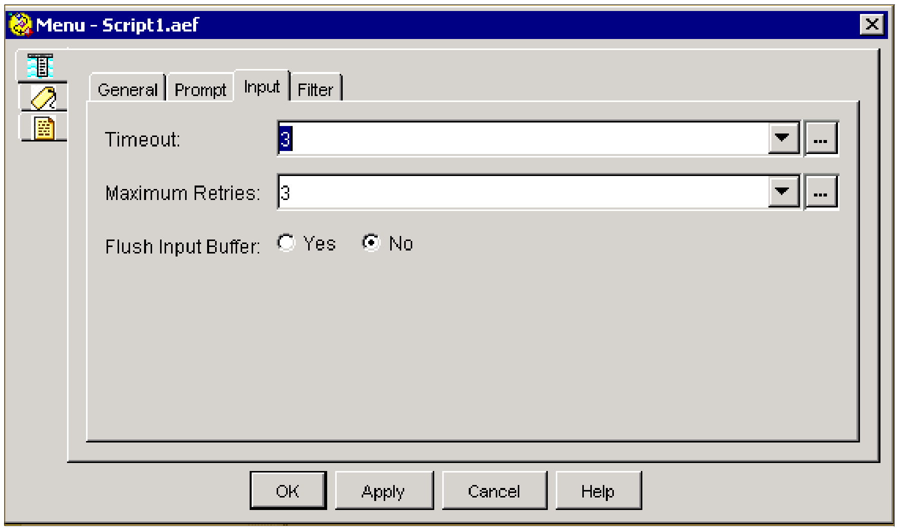
Table 2-75 Menu Properties—Prompt Tab

Properties / Buttons	Description
Prompt	Variable or expression indicating the prompt to be played back to caller.
Barge In	Radio button. **Yes**—The caller can interrupt the prompt. **No**—The prompt must complete playback before the caller can respond.
Continue on Prompt Errors	Radio button. **Yes**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **No**—An exception results, which can then be handled in the script.

Input tab (Menu step)

Use the Input tab of the Menu customizer window to set the timeout setting, maximum number of retries, and Flush Input Buffer options.

Figure 2-92 Menu Customizer Window—Input Tab



Timeout: 3
Maximum Retries: 3
Flush Input Buffer: Yes [✓ No]

Table 2-76 describes the properties of the Input tab of the Menu customizer window.

Table 2-76 Menu Properties—Input Tab

Properties / Buttons	Description
Timeout	Variable or expression indicating the amount of time the system waits for input from the caller. When this timer expires, the system either replays the prompt or plays the system prompt that asks if the caller is still there.
Maximum Retries	Variable indicating the number of times the entry can be restarted after a timeout or invalid input response. After the maximum number of retries is reached, the Menu step continues on the Timeout or Unsuccessful output branches depending on whether the last try timed out or an invalid input response was entered. A "0" value means that no retry is allowed; in this case, the script must handle the retry scenario.
Flush Input Buffer	Radio button. **Yes**—The system erases previously entered input before capturing caller input. **No**—The system does not erase previously entered input before capturing caller input.

Filter tab (Menu step)

Use the Filter tab of the Menu customizer window to set options. Selecting all connections at a time will show all DTMF digits currently selected throughout all connections in the middle pane.

Figure 2-93 Menu Customizer Window—Filter Tab

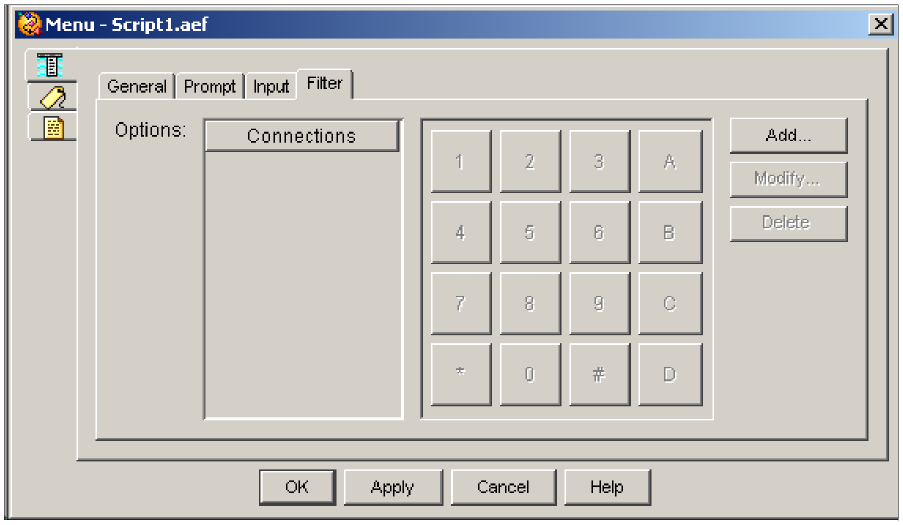


Table 2-77 describes the properties of the Filter tab of the Menu customizer window.

Table 2-77 Menu Properties—Filter Tab

Properties / Buttons	Description
Options (list and key pad)	Use the Connections list box and the number key pad to map an option name to a digit.
Add / Modify (buttons)	Use these buttons to add or modify a Connection Name for the option. When done, click **OK**.
Delete (button)	To remove an Output Option Name, highlight an option in the list and click **Delete**.

Name To User Step

Note The Name to User step is available in the Cisco UnifiedIP IVR, Cisco Unified CCX Premium, and Cisco Unified CCX Enhanced license packages.

The Name To User step is used in the following ways:

- To prompt a caller for the name of the person being called (using either DTMF or speech), and then to compare the name entered by the caller with names stored in a directory.
- To automatically transfer a caller to the extension of the person being called.

Note The Name to User Step supports only the 26-character English alphabet. As a result, the Cisco Automated Attendant Dial-by-Name feature and other Solutions that use this step accept English characters only when using CMT(Cisco Media Terminated).

- To assign a value to a variable that can later be queried using the Get User Info step, in order to retrieve information such as the extension, e-mail address, and spoken name of the caller.

Note When any previous escalating prompt in a script enters the Name To User step, it is reset to the first prompt in its list.

Features of the Name to User Step:

- Using a CMT channel, the Name To User step receives DTMF input from a caller with the following numeric keypad mapping:
  - 2 = ABC
  - 3 = DEF
  - 4 = GHI
  - 5 = JKL
  - 6 = MNO
  - 7 = PQRS
  - 8 = TUV
  - 9 = WXYZ
- Using the information from this step, a script creates a subsequent prompt that plays the prerecorded name of the caller if it exists. If no recording exists, the script spells the caller name.

Note The Name To User step is limited to spelling back (over CMT media names with ASCII-only characters, which may be a limitation under some international conditions.

- Using an ASR channel, the Name To User step receives the spoken input from the caller. If configured to do so, the caller can speak a nickname.

The Name To User step produces the following output branches:

- Successful—A successful match is made between the input from the caller and a name in the directory.
- Timeout—The step has reached the maximum number of retries (as configured in the customizer) without receiving input from the caller.
- Unsuccessful—The input from the caller does not match a name in the directory.
- Operator—The operator's extension was entered.

Note The Operator output branch appears under the Name To User step in the script only if Yes is selected for the Operator option in the General tab of the Name To User customizer window.

Note If the Name To User step matches the caller input with a single user defined in the Cisco Unified Communications Manager, that result is returned immediately without requiring the caller to confirm the selection.

If you are upgrading your system to Cisco Unified CCX 4.x and later, please note that this *might* be a change in functionality for your script. In Cisco Unified CCX versions prior to 3.x, the Name To User step *would* prompt the caller to confirm the selection.

The customizer window of the Name To User step contains three tabs:

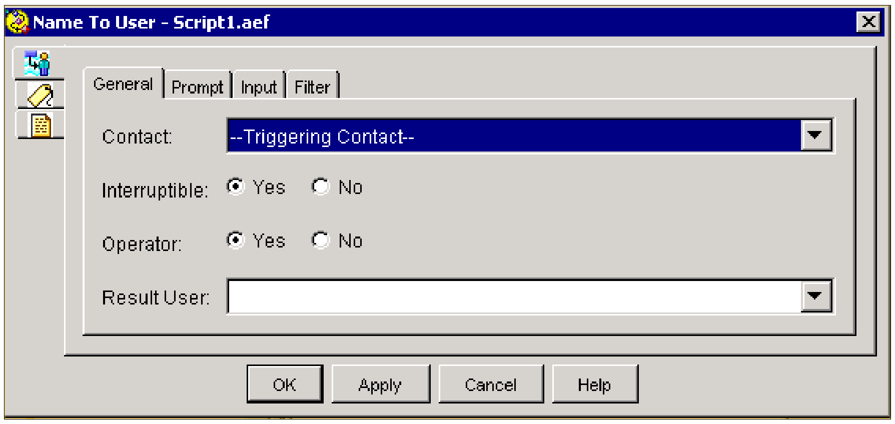
- General tab (Name to User step)
- Prompt tab (Name to User step)
- Input tab (Name to User step)
- Filter tab (Name to User step)

The following sections describe these tabs.

General tab (Name to User step)

Use the General tab to specify the Result User variable and to set other properties for the Name To User step.

Figure 2-94 Name To User Customizer Window—General Tab



Contact: --Triggering Contact--
Interruptible: [✓ Yes] No
Operator: [✓ Yes] No
Result User:

Table 2-78 describes the properties of the General tab of the Name To User customizer window.

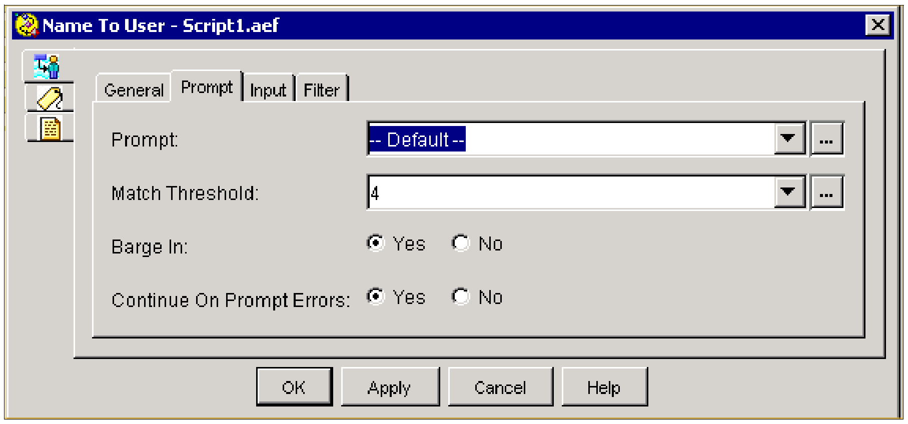
Table 2-78 Name To User Properties—General Tab

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified.
Interruptible	Radio button. **Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.
Operator	Radio button. **Yes**—The caller has the option to connect to an operator by pressing "0" or saying "Operator" in the language set for the contact. **No**—The caller is not offered the option to connect to an operator.
Result User	Variable that stores a user object representing the caller selected.

Prompt tab (Name to User step)

Use the Prompt tab to specify prompts to be played back by the Name To User step, and to set the Barge In and Continue on Prompt Errors options.

Figure 2-95 Name To User Customizer Window—Prompt Tab



Prompt: -- Default --
Match Threshold: 4
Barge In: [✓ Yes] No
Continue On Prompt Errors: [✓ Yes] No

Table 2-79 describes the properties of the Prompt tab of the Name To User customizer window.

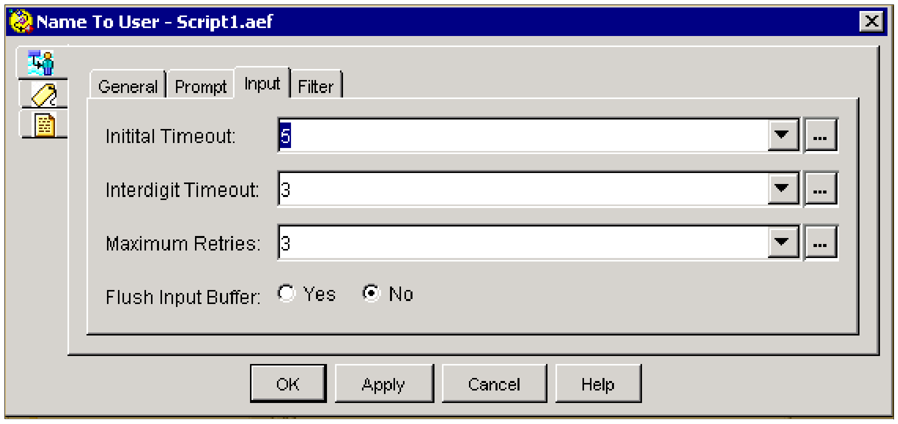
Table 2-79 Name To User Properties—Prompt Tab

Properties / Buttons	Description
Prompt	Variable or expression indicating the prompt to be played back to the caller. • Default prompt— System prompt bundled with the media. "Spell the last name followed by the first name" is included with DTMF media. "Say the name" is included for ASR channel. • Customized prompt—Prompt created by the script designer. • No prompt—No prompt is played.
Match Threshold	
Barge In	Radio button. **Yes**—The caller can interrupt the prompt. **No**—The prompt must complete playback before the caller can respond.
Continue on Prompt Errors	Radio button. **Yes**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **No**—An exception results, which can then be handled in the script.

Input tab (Name to User step)

Use the Input tab to configure various input properties for the Name To User step.

Figure 2-96 Name To User Customizer Window—Input Tab



Initital Timeout: 5
Interdigit Timeout: 3
Maximum Retries: 3
Flush Input Buffer: Yes [✓ No]

Table 2-80 describes the properties of the Input tab of the Name To User customizer window.

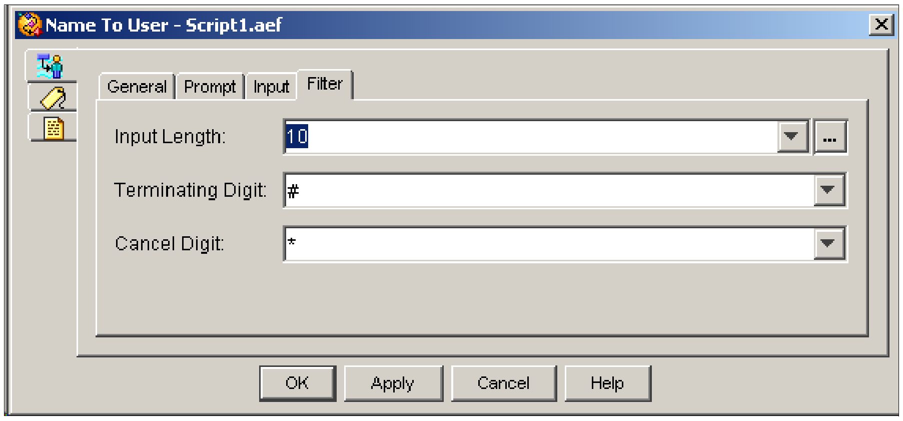
Table 2-80 Name To User Properties—Input Tab

Properties / Buttons	Description
Initial Timeout (in sec)	Variable indicating the amount of time, in seconds, that the system waits for initial input from the caller.
Interdigit Timeout (in sec)	Variable indicating the amount of time that the system waits for the caller to enter the next digit, after receiving initial input from the caller. **Note** This property does not apply to ASR channels.
Maximum Retries	Variable indicating the number of times the step attempts to receive valid input. A “0” value means that no retry is allowed; in this case, the script must handle the retry scenario.
Flush Input Buffer	Radio button. **Yes**—The system erases previously entered input before capturing caller input. **No**—The system does not erase previously entered input before capturing caller input.

Filter tab (Name to User step)

Use the Filter tab to configure various filter properties for the Name To User step.

Figure 2-97 Name To User Customizer Window—Filter Tab



Input Length: 10
Terminating Digit: #
Cancel Digit: *

Table 2-81 describes the properties of the Filter tab of the Name To User customizer window.

Table 2-81 Name To User Properties—Filter Tab

Properties / Buttons	Description
Input Length	Variable indicating the minimum number of digits required before automatically checking for a caller match. This property applies only to CMT channels.

Table 2-81 Name To User Properties—Filter Tab (continued)

Properties / Buttons	Description
Terminating Digit	Variable indicating the key used to indicate the end of caller input.
Cancel Digit	Variable indicating the key the caller presses to start over. **Note** The Cancel key works only until the number of maximum retries is reached.

Play Prompt Step

Use the Play Prompt step to play back specified prompts to the caller.

Note When any previous escalating prompt in the script enters the Play Prompt step, it is reset to the first prompt in its list.

The customizer window of the Play Prompt step contains three tabs:

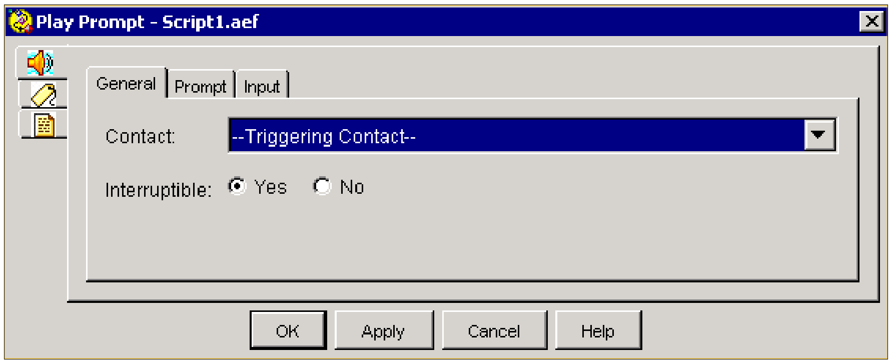
- General tab (Play Prompt step)
- Prompt tab (Play Prompt step)
- Input tab (Play Prompt step)

The following sections describe these tabs.

General tab (Play Prompt step)

Use the General tab to identify the contact and to set the Interruptible option.

Figure 2-98 Play Prompt Customizer Window—General Tab



Contact: --Triggering Contact--

Interruptible: [✓ Yes] No

Table 2-82 describes the properties of the General tab of the Play Prompt customizer window.

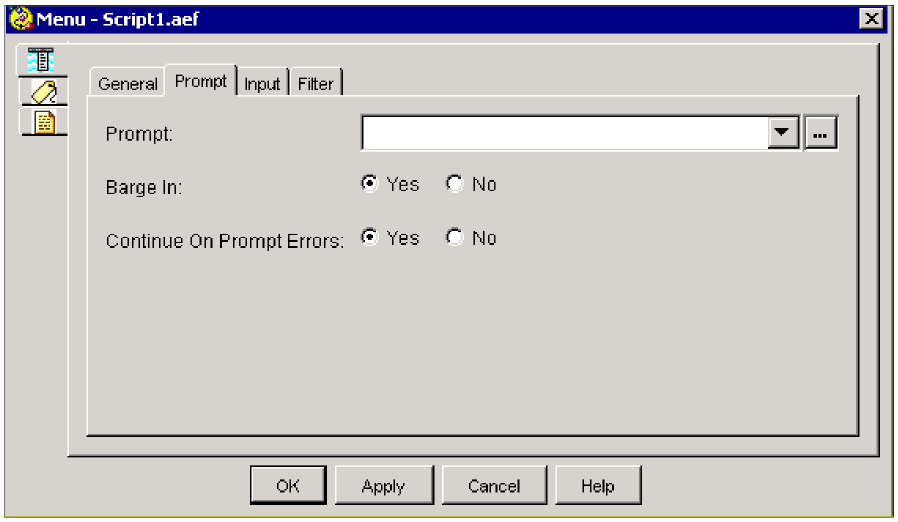
Table 2-82 Play Prompt Properties—General Tab

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
Interruptible	Radio button. **Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.

Prompt tab (Play Prompt step)

Use the Prompt tab of the Play Prompt customizer window to specify the prompt to be played back, and to set the Barge In and Continue on Prompt Errors options.

Figure 2-99 1Play Prompt Customizer Window—Prompt Tab



Prompt:
Barge In: [✓ Yes] No
Continue On Prompt Errors: [✓ Yes] No

Table 2-83 describes the properties of the Prompt tab of the Play Prompt customizer window.

Table 2-83 Play Prompt Properties—Prompt Tab

Properties / Buttons	Description
Prompt	Variable or expression indicating which prompt is to be played.

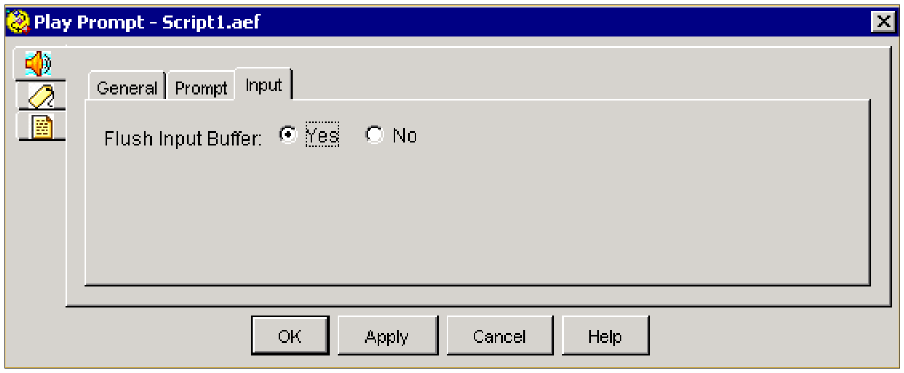
Table 2-83 Play Prompt Properties—Prompt Tab (continued)

Properties / Buttons	Description
Barge In	Radio button. **Yes**—The caller can interrupt the prompt. **No**—The prompt must complete playback before the caller can respond.
Continue on Prompt Errors	Radio button. **Yes**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **No**—An exception results, which can then be handled in the script.

Input tab (Play Prompt step)

Use the Input tab of the Play Prompt step to specify whether or not to erase previously entered input before capturing caller input.

Figure 2-100 Play Prompt Customizer Window—Input Tab



Flush Input Buffer: Yes [✓ Yes] No

Table 2-84 describes the property of the Input tab of the Play Prompt customizer window.

Table 2-84 Play Prompt —Input Tab

Properties / Buttons	Description
Flush Input Buffer	Radio button. **Yes**—The system erases previously entered input before capturing caller input. **No**—The system does not erase previously entered input before capturing caller input.

Recording Step

Use the Recording step to record audio input from the caller and return it as a Document object that can later be uploaded as a spoken name (or uploaded directly into the Document repository to be made available to all Cisco Unified CCX servers in the cluster), saved to disk or to a database, or e-mailed.

All recordings are defined with a RIFF (Resource Interchange File Format) header of type WAVE and encoded using G711 u-law and G729 format.

Note When any previous escalating prompt in the script enters the Recording step, it is reset to the first prompt in its list.

The customizer window of the Recording step contains three tabs:

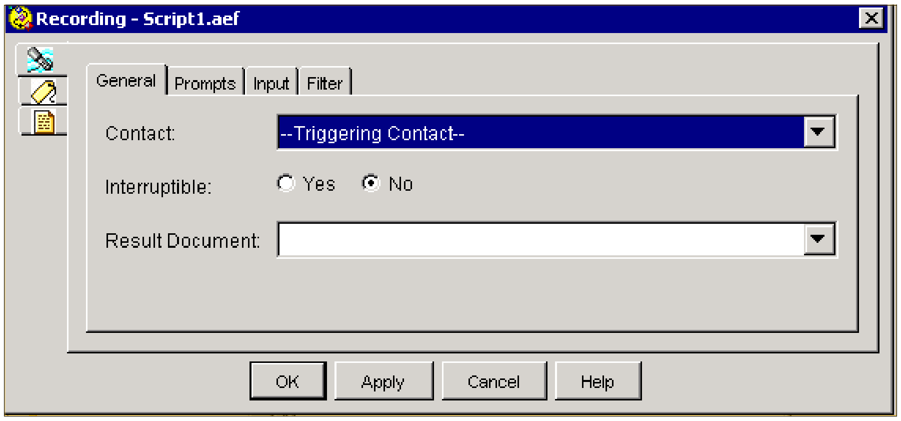
- General tab (Recording step)
- Prompts tab (Recording step)
- Input tab (Recording step)
- Filter tab (Recording step)

The following sections describe these tabs.

General tab (Recording step)

Use the General tab of the Recording step to label the recorded document, set the recording duration and media type, and set the Interruptible option.

Figure 2-101 Recording Customizer Window—General Tab



Contact: --Triggering Contact--
Interruptible: Yes [✓ No]
Result Document:

Table 2-85 describes the properties of the General tab of the Recording customizer window.

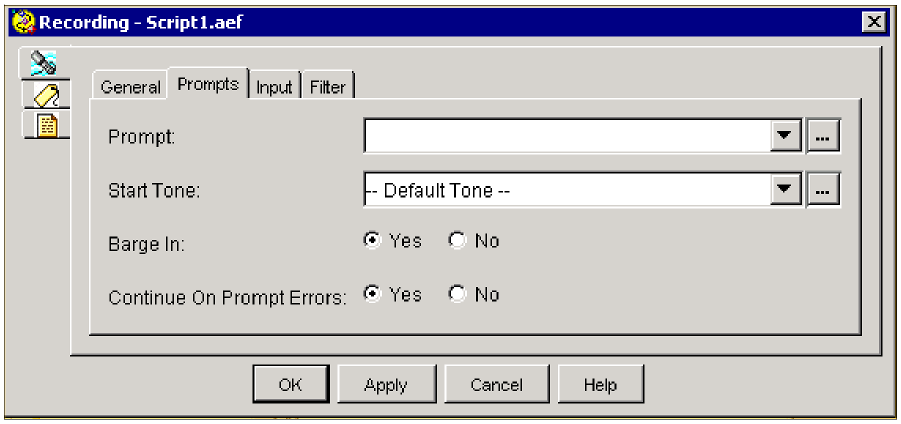
Table 2-85 Recording Properties—General Tab

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified.
Interruptible	Radio button. **Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.
Result Document	Variable indicating where the resulting audio document is saved.

Prompts tab (Recording step)

Use the Prompts tab of the Recording step to specify a prompt name, a start tone, a barge in option, and a continue on prompt errors option.

Figure 2-102 Recording Customizer Window—Prompt Tab



Prompt:
Start Tone: -- Default Tone --
Barge In: [✓ Yes] No
Continue On Prompt Errors: [✓ Yes] No

Table 2-86 describes the properties of the Prompts tab of the Recording customizer window.

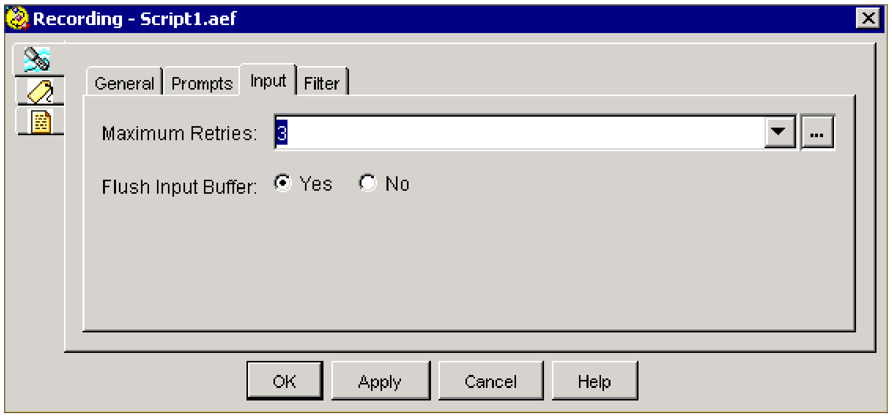
Table 2-86 Recording Properties—Prompt Tab

Properties / Buttons	Description
Prompt	Variable indicating the prompt to be played back to the callers: • Customized Prompt—Prompt created by the designer. • No Prompt—No prompt is played. **Note** If you select Customized Prompt, use the List of Prompts drop-down list or the Expression Editor to specify the prompt to be played.
Start Tone	Tone that indicates the recording is about to begin. Options from the drop-down list: • Default Prompt—System prompt providing a default start tone for playback. • Customized Prompt—Start tone created by the designer. • No Prompt—No start tone is played.
Barge In	Radio button. **Yes**—The caller can interrupt the prompt. **No**—The prompt must complete playback before the caller can respond. **Note** The start tone is not interruptible and will always be played back if specified, even if the main prompt is interrupted.
Continue on Prompt Errors	Radio button. **Yes**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **No**—An exception results, which can then be handled in the script.

Input tab (Recording step)

Use the Input tab of the Recording customizer window to set terminating and cancel keys, the maximum number of retries, and the Flush Input Buffer option.

Figure 2-103 Recording Customizer Window—Input Tab



Maximum Retries: 3

Flush Input Buffer: [✓ Yes] No

Table 2-87 describes the properties of the Input tab of the Recording customizer window.

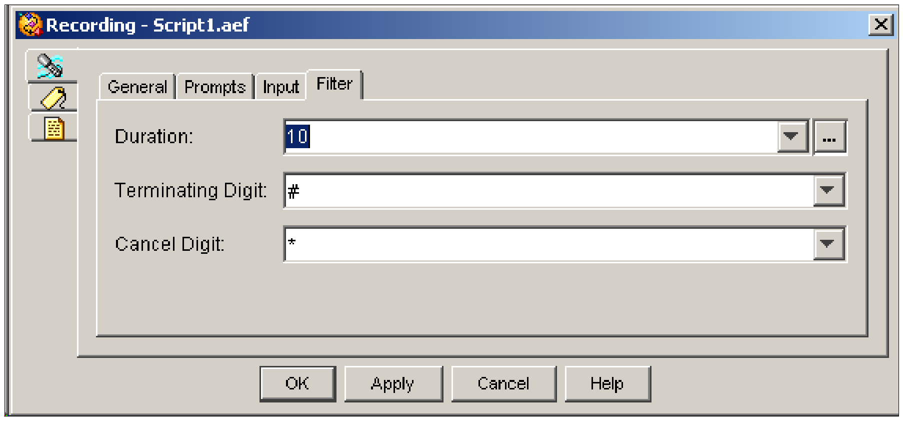
Table 2-87 Recording Properties—Input Tab

Properties / Buttons	Description
Maximum Retries	Variable indicating the number of times the recording can be re-attempted after the cancel key has been detected. After the maximum number of retries is reached, the step continues on the Unsuccessful output branch. On retry due to an invalid key, a system prompt plays if none is specified. A “0” value means that no retry is allowed; in this case, the script must handle the retry scenario.
Flush Input Buffer	Radio button. **Yes**—The system erases previously entered input before capturing caller input. **No**—The system does not erase previously entered input before capturing caller input.

Filter tab (Recording step)

Use the Filter tab of the Recording customizer window to set the duration, terminating digit, and cancel digit.

Figure 2-104 Recording Customizer Window—Filter Tab


Recording - Script1.aef

Duration: 10

Terminating Digit: #

Cancel Digit: *

Table 2-88 describes the properties of the Filter tab of the Recording customizer window.

Table 2-88 Recording Properties—Filter Tab

Properties / Buttons	Description
Duration	Variable or expression indicating the maximum length of time allotted for recording.
Terminating Digit	Variable indicating the key used to indicate the end of caller input.
Cancel Digit	Variable indicating the key the caller presses to start over.

Send Digit String Step

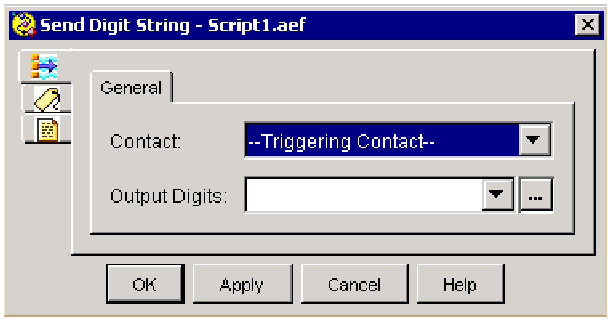
Use the Send Digit String step to out-pulse back to the caller a specified set of DTMF digits.

The DTMF digits are sent out-of-band and out-pulsed in-band by the gateways.

Note You can insert a “,” (comma) in the sequence of DTMF digits to instruct the script to insert a 1-second pause before continuing to out-pulse the remaining DTMF digits. You can use multiple commas to increase the length of the pause.

Figure 2-105 shows the Send Digit String customizer window.

Figure 2-105 Send Digit String Customizer Window



Contact: --Triggering Contact--

Output Digits:

Table 2-89 describes the properties of the Send Digit String customizer window.

Table 2-89 Send Digit String Properties

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is Triggering Contact, unless another contact is specified.
Output Digits	Variable or expression indicating the sequence of DTMF digits to be sent.

Simple Recognition Step

Use the Simple Recognition step to allow the caller to choose options from a menu.

Note When any previous escalating prompt in the script enters the Simple Recognition step, it is reset to the first prompt in its list.

When used over an ASR channel, the Simple Recognition step provides voice-enabled word and digit menus. When used over a CMT channel, the step ignores DTMF digits specified in the grammar.

For example, a prompt may say "Please select from 'Stocks,' 'Sports,' and 'Music,' or press or say 'star' to speak with a representative." In this case, the script can recognize the spoken words "stocks," "sports," "music," and "star," and can also recognize the DTMF key "*".

Note For optimal ASR functionality, use VoiceXML interpretation provided by the Voice Browser step.

The Simple Recognition step is similar to the Menu step, with an additional capability that allows the script designer to utilize caller-defined grammar for matching caller input.

The Simple Recognition step uses a configurable grammar variable that stores the words or digits that the script can recognize. (See Grammar Steps for information on creating grammar variables.)

Note Tags for grammars are case-sensitive.

Note When you use the Simple Recognition step with ASR, the number of output points defined in the step is subject to the license agreement of the ASR ports purchased. If you purchased a limited-use ASR license for the ASR ports, you can use no more than 40 output points in the Simple Recognition step. If you require more then 40 different choices, you must purchase a full-use ASR license.

The customizer window of the Simple Recognition step contains three tabs:

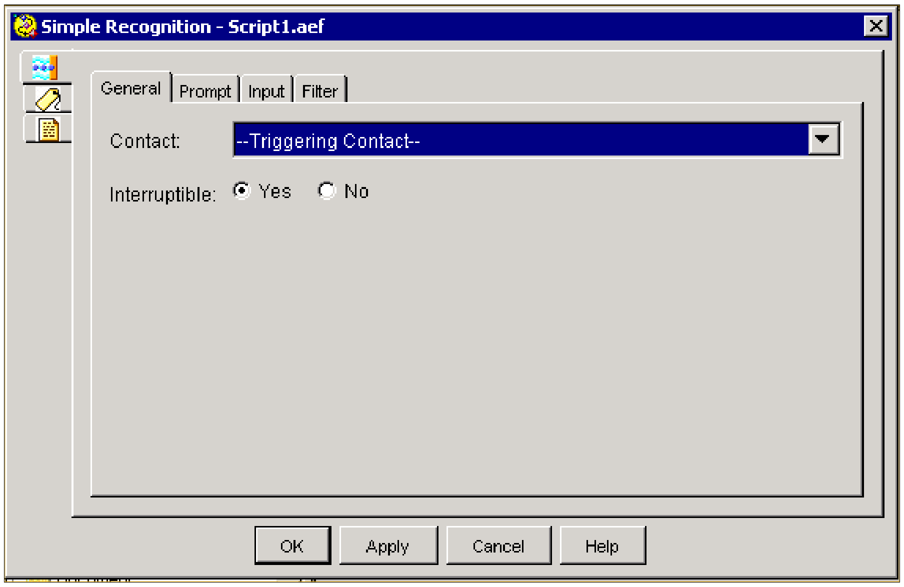
- General tab (Simple Recognition step)
- Prompt tab (Simple Recognition step)
- Input tab (Simple Recognition step)
- Filter tab (Simple Recognition step)

The following sections describe these tabs.

General tab (Simple Recognition step)

Use the General tab of the Simple Recognition customizer window to set the contact, grammar, output points and tags, and the interruptible option.

Figure 2-106 Simple Recognition Customizer Window—General Tab



Contact: --Triggering Contact--

Interruptible: [✓ Yes] No

Table 2-90 describes the properties of the General tab of the Simple Recognition customizer window.

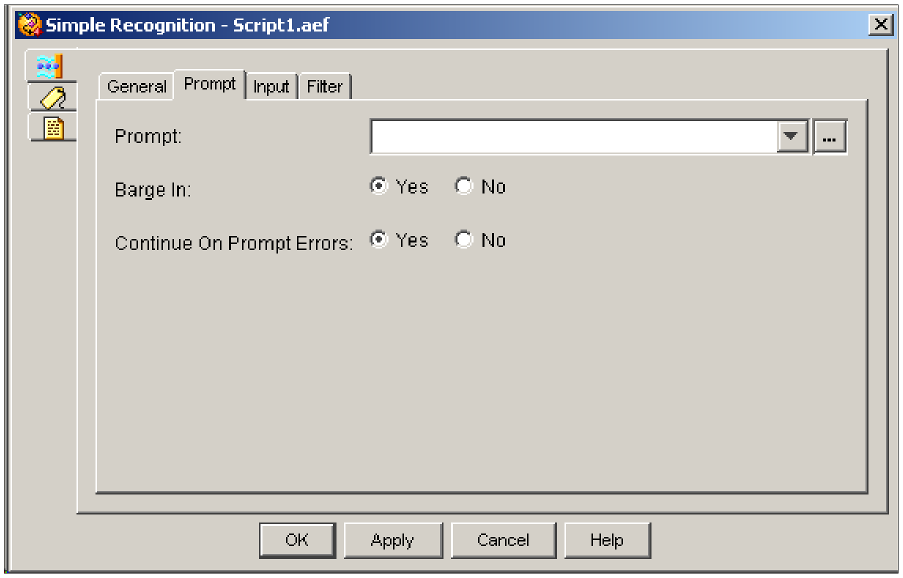
Table 2-90 Simple Recognition Properties—General Tab

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified.
Interruptible	Radio button. **Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.

Prompt tab (Simple Recognition step)

Use the Prompt tab of the Simple Recognition step to specify the prompt to be played back.

Figure 2-107 Simple Recognition Customizer Window—Prompt Tab



Prompt:
Barge In: [✓ Yes] No
Continue On Prompt Errors: [✓ Yes] No

Table 2-91 describes the properties of the Prompt tab of the Simple Recognition customizer window.

Table 2-91 Simple Recognition Properties—Prompt Tab

Properties / Buttons	Description
Prompt	Variable or expression indicating prompt to be played back to caller.

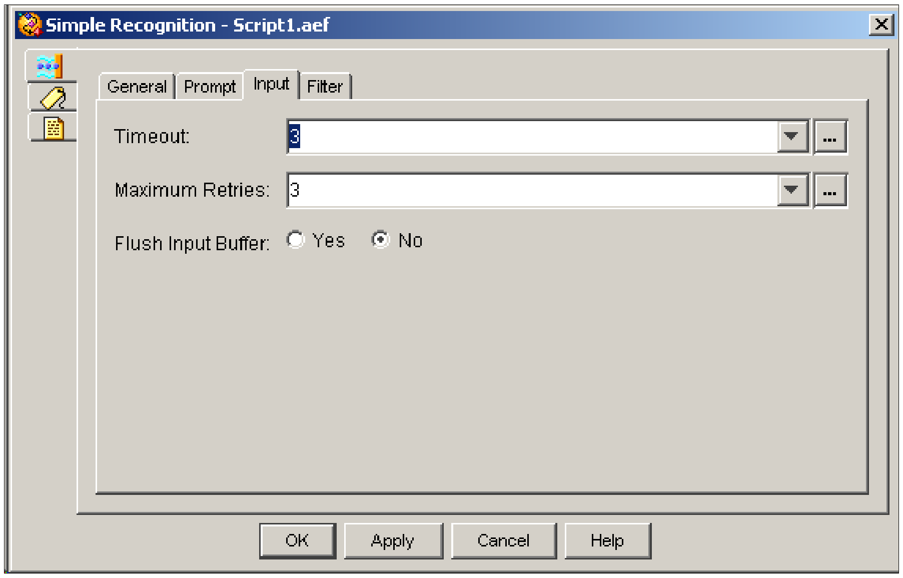
Table 2-91 Simple Recognition Properties—Prompt Tab (continued)

Properties / Buttons	Description
Barge In	Radio button. **Yes**—The caller can interrupt the prompt. **No**—The prompt must complete playback before the caller can respond.
Continue on Prompt Errors	Radio button. **Yes**—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. **No**—An exception results, which can then be handled in the script.

Input tab (Simple Recognition step)

Use the Input tab of the Simple Recognition step to set the timeout setting, maximum number of retries, and the Flush Input Buffer option.

Figure 2-108 Simple Recognition Customizer Window—Input Tab


Simple Recognition - Script1.aef

Timeout: 3
Maximum Retries: 3
Flush Input Buffer: Yes [✓ No]

Table 2-92 describes the properties of the Input tab of the Simple Recognition customizer window.

Table 2-92 Simple Recognition Properties—Input Tab

Properties / Buttons	Description
Timeout	Variable or expression indicating the amount of time in seconds that the system waits for input from the caller.
Maximum Retries	Variable indicating the number of times a new entry can be entered after a timeout or invalid key. After the maximum number of retries is reached, the step continues on the Timeout or Unsuccessful output branch, depending on whether the last try timed out or an invalid key was entered. On retry due to an invalid key, a system prompt plays if none is specified. A "0" value means that no retry is allowed; in this case, the script must handle the retry scenario.
Flush Input Buffer	Radio button. **Yes**—The system erases previously entered input before capturing caller input. **No**—The system does not erase previously entered input before capturing caller input.

Filter tab (Simple Recognition step)

Use the Filter tab of the Simple Recognition step to set the timeout setting, maximum number of retries, and the Flush Input Buffer option.

Figure 2-109 Simple Recognition Customizer Window—Filter Tab

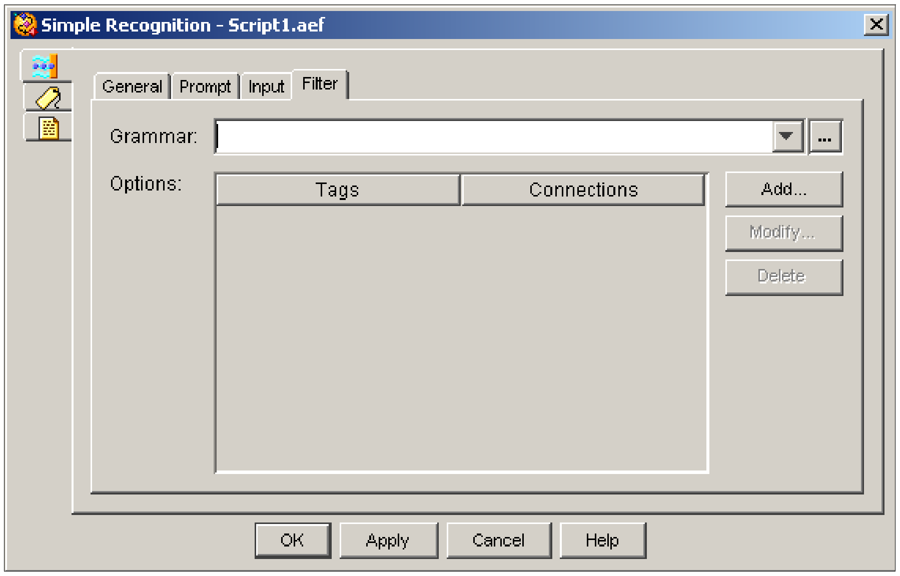


Table 2-93 describes the properties of the Filter tab of the Simple Recognition customizer window.

Table 2-93 Simple Recognition Properties—Filter Tab

Properties / Buttons	Description
Grammar	Variable or expression indicating the grammar to be used in the Simple Recognition step.
Options (Tags and Connections)	List of options in the menu offered to the caller. Tags are mapped to output connections to determine execution of branching output paths.
Add / Modify (buttons)	Use these buttons to access the Add or Modify Option dialog box. Use the dialog to specify the following: • Output Connection—A name for the option. • Tag—Value mapped to the Output Connection that determines branching. **Note** Tag values entered must correspond to the tag values defined in the grammar When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.

Voice Browser Step

Use the Voice Browser step to allow callers to access and interpret VoiceXML-enabled web pages.

Note The Voice Browser step supports:

DTMF—for all Cisco Unified CCX packages.
That is, it accepts pre-recorded .wav files and input from a touch-tone phone.

Only DTMF —for the Cisco Unified CCX Standard, and the Cisco Unified CCX Enhanced packages.

DTMF, MRCP ASR, and MRCP TTS —for the Cisco UnifiedIP IVR or the Cisco Unified CCX Premium packages, only.

The customizer window of the Voice Browser step contains three tabs:

- General tab (Voice Browser step)
- <exit> Attributes tab (Voice Browser step)
- Prompt tab (Voice Browser step)

The following sections describe these tabs.

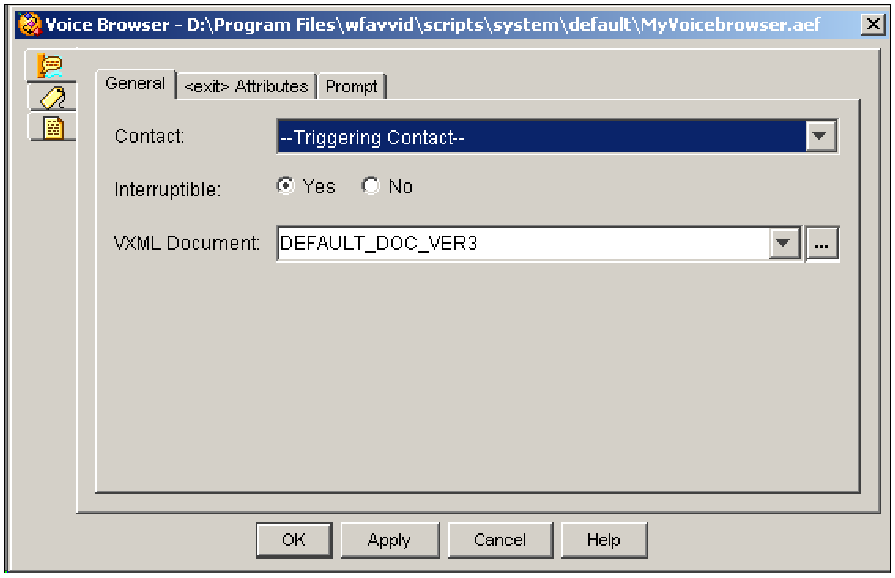
Note The Voice Browser step works with either an ASR channel or a CMT channel. If you invoke it with a contact not associated with one of these channels, a ChannelUnsupportedException error results.

General tab (Voice Browser step)

Use the General tab of the Voice Browser workflow editor step to select a Document variable that contains the name of what to open. This document variable points the browser toward the specific URL from which you can invoke the Voice Browser and run it.

Note You need to create the Document variable first using the Create URL Document step before you can specify the variable name in the Voice Browser step.

Figure 2-110 Voice Browser Customizer Window—General Tab


Voice Browser - D:\Program Files\wfavvid\scripts\system\default\MyVoicebrowser.aef

Contact: --Triggering Contact--

Interruptible: [✓ Yes] No

VXML Document: DEFAULT_DOC_VER3

Table 2-94 describes the properties of the General tab of the Voice Browser customizer window.

Table 2-94 Voice Browser Properties—General Tab

Properties / Buttons	Description
Contact	Variable indicating the contact that triggers the execution of the step. Default is the Triggering Contact, unless another contact is specified.
Interruptible	Radio button. **Yes**—An external event (such as an agent becoming available or a caller hanging up) can interrupt the step. **No**—The step must complete before any other process can execute.
VXML Document	Variable or expression indicating the URL that points to the VoiceXML-enabled web pages. In addition to the URL, request string parameters are put into the URL document and passed through the Document object. **Note** If the VoiceXML application uses a grammar file with more than one rule, be aware that the Voice Browser starts recognition using the **last** rule in the file. Remember to take this into consideration when designing your VoiceXML script.

<exit> Attributes tab (Voice Browser step)

Use the <exit> Attributes tab of the Voice Browser step to return parameter information to the script.

These parameters correspond to the exit parameters in the VXML script. Inside VoiceXML scripts are elements you can specify as exit parameters that are handled by the <exit> Attributes tab in the Cisco Unified CCX script editor. You must specify (add) these attributes in the Voice Browser <exit> Attributes tab if you want to use the corresponding VoiceXML script Exit parameters in a Cisco Unified CCX script.

Figure 2-111 Voice Browser Customizer Window—<exit> Attributes Tab

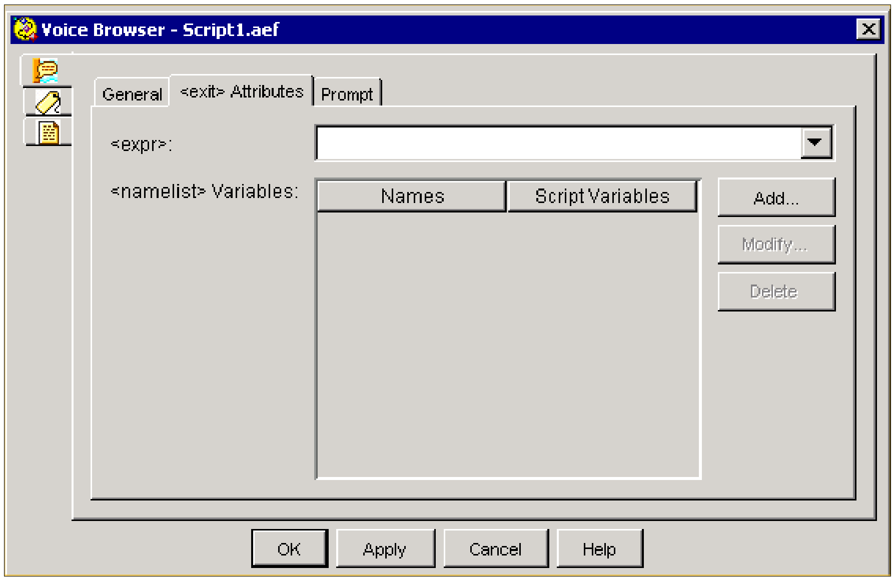


Table 2-95 describes the properties of the <exit> Attributes tab of the Voice Browser customizer window.

Table 2-95 Voice Browser Properties—<exit> Attributes Tab

Properties / Buttons	Description
<expr>	The variable that stores the expression of the parameter.
<namelist> Variables (Names and Script Variables)	The names of the parameters with their corresponding script variables.

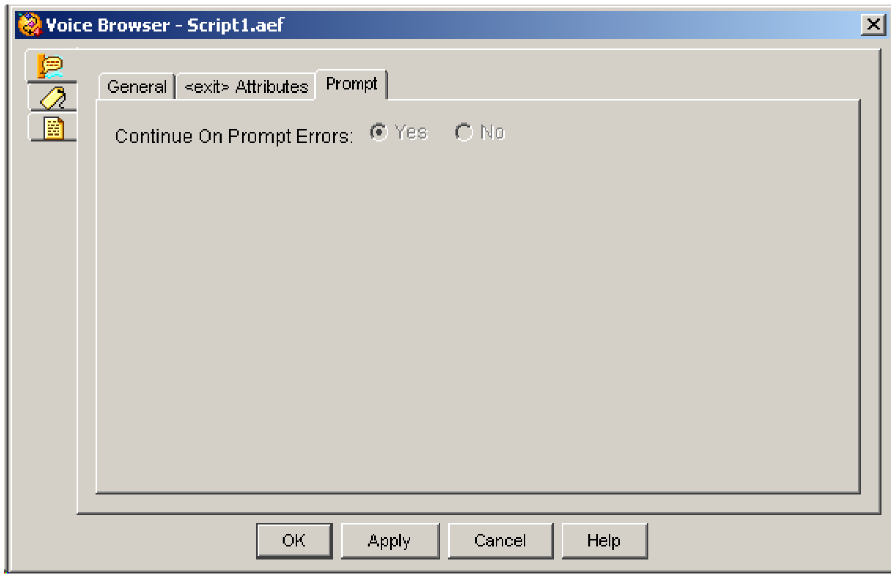
Table 2-95 *Voice Browser Properties—<exit> Attributes Tab*

Properties / Buttons	Description
Add / Modify (buttons)	Use these buttons to access the Add or Modify <namelist> Variable dialog box. Use the dialog to specify the following: • **Name**—The name of the parameter. • **Script Variable**—The variable that stores the parameter value in the script. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove parameter information, highlight a value in the list and click **Delete**.

Prompt tab (Voice Browser step)

Use the Prompt tab of the Voice Browser step to enable or disable the continue on prompt errors option.

Figure 2-112 *Voice Browser Customizer Window—Prompt Tab*



Continue On Prompt Errors: [✓ Yes] No

Table 2-96 describes the properties of the Prompt tab of the Voice Browser customizer window.

Table 2-96 *Voice Browser Properties—Prompt Tab*

Properties / Buttons	Description
Continue on Prompt Errors	This property is disabled for the Voice Browser step. **Note** A similar functionality is available through the VoiceXML application.

Generic Recognition Step

This section contains the following topics:

- About the Generic Recogntion Steps
- How the Generic Recognition Set of Steps Work
- The Customizer Window of the Generic Recognition Step

For an example script showing how the generic recognition steps work, see the *Cisco Unified Contact Center Express Scripting and Development Series, Volume 1: Getting Started with Scripts*.

About the Generic Recogntion Steps

The Generic Recognition step is used when something more complex than a selection menu is needed in a speech dialog. The Generic Recognition step allows the application designer to use an arbitrarily complex Speech Recognition Grammar. These grammars can be used to build mixed initiative dialogs. They can be ambiguous. They can assign meaning to the results returned using semantic interpretation. They can allow multiple results to be returned.

The result returned by a recognition using this step is stored in an opaque data object that is associated with a name that is assigned in this step. The information in this result object can be extracted using two other steps: Get Recognition Result Info Step and Get Recognition Interpretation Step. These two steps along with the Generic Recognition Step are designed to work together. The way the result data is stored and how it is extracted is explained below.

How the Generic Recognition Set of Steps Work

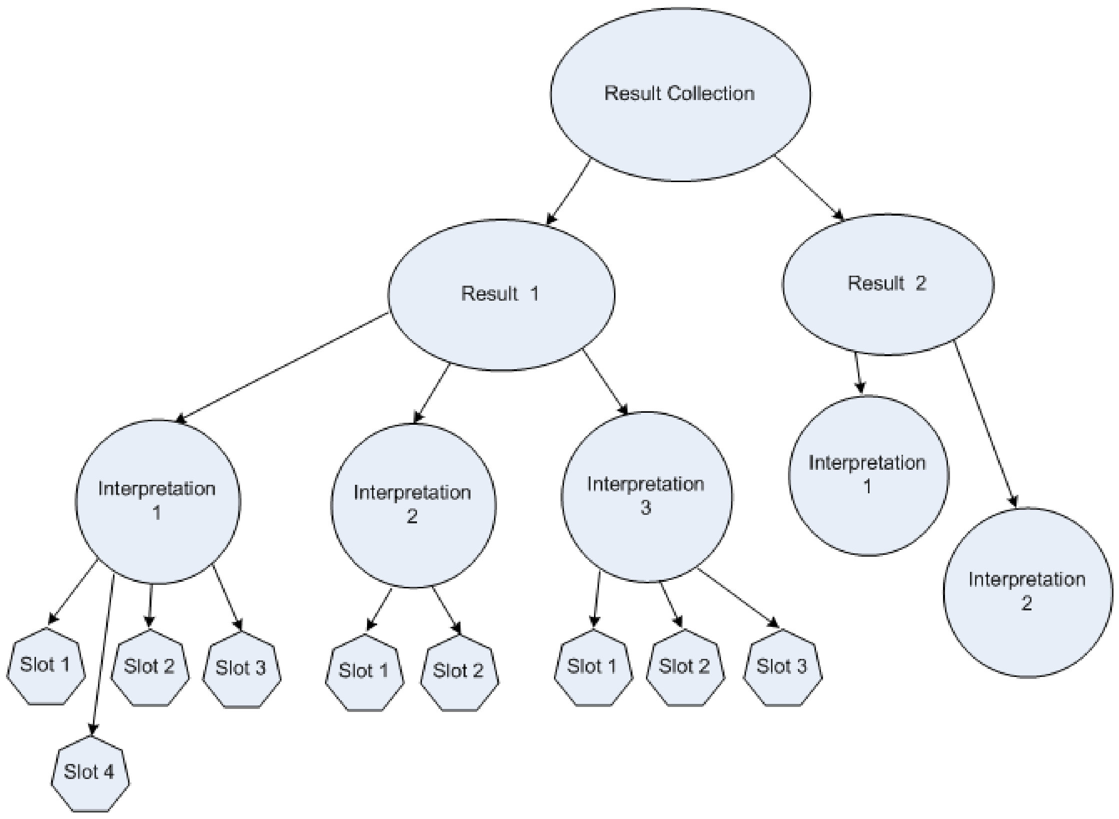
The Generic Recognition set of steps gives the script writer access to more control over speech recognition capability than the Simple Recognition step.

The input part of the Generic Recognition step uses a general purpose grammar: The Generic Recognition step performs recognition based on the rules of the grammar specified in the step.

1. The Generic Recognition step uses the grammar to recognize the utterance provided by a user. It puts the result of this recognition into a result collection. The step assigns a name to the result collection. This name is used by the other steps to access this result collection.
2. The Get Recognition Interpretation step is used to select a specific result from the result collection.
3. The Get Recognition Interpretation step is used to select a specific interpretation within a specific result. The selected interpretation may have one or more slots as specified in the grammar. These slots contain information associated with this particular interpretation that give meaning to the recognition. For instance, if the recognized string was the name of a person from a company directory, there may be a slot in the interpretation called "extension" that contains the actual extension for that person. Another slot might contain the name of that person.

   Slots are optional. Not all interpretations in the same result have to have the same slots (Although it is best to design the grammar so that slots are returned in a consistent manner).

Figure 2-113 General Structure of the Result Collection



The Result Collection returned by the Generic Recognition Step can contain one or more Result. Each result differs from the other results in that their utterance and confidence level will not be the same. This will typically happen when two or more phrases in the grammar sound very similar but are in fact different words.

This could happen, for example, if the grammar contained separate recognition phrases of, say, "*fog*", "*dog*" and "*frog*".

If the speaker says "*fog*," the recognizer may actually also match the other two phrases as well. The difference is that the confidence level for the others will be lower than the confidence level for "*fog*". The confidence level for "*fog*" will probably be above 90 because it is a very close match to the word "*fog*" in the grammar. Note that "*fog*" will be the utterance for this result.

The result for "*frog*" may have a confidence level somewhere around 50 or 60. This is not a good match, but it will probably exceed the normal confidence threshold.

Finally the result for "*dog*" will probably have a confidence level much less than 50. It may not actually be returned in the result collection at all because the default confidence threshold is usually around 50.

Within each result there may be one or more interpretations. Multiple interpretations are possible when an ambiguous grammar is used. An ambiguous grammar is one where there is more than one path through the grammar for the same utterance. This is not necessarily a bad thing.

Consider a company directory. There might be several people in the directory with the same name (say, "*John Smith*"). For proper recognition, each of these people must have its own place in the recognition grammar. Because there is more than one way to recognize "*John Smith*" in this grammar, we say that is "*ambiguous*".

Therefore, when a speaker says "*John Smith*", there will be multiple interpretations returned within the result for "*John Smith*". Since they will all have the same confidence level and utterance, they are in the same result.

Of course, in order to take advantage of multiple interpretations, there must be a way to disambiguate them. This can be done using slots. For instance, each user in the directory will typically have a unique user Id. This user id can be returned in a slot. The slot can be called "*userId*". Additional information for a user can be returned in other slots. For instance there could be a slot for the user's extension number. There may be another one to return the user's E-mail address.

This information could be used to, say, generate a disambiguation prompt so that the caller can choose which user they really want to call.

These steps are also designed to iterate through each result in the result collection and through each interpretation in each result. This is enabled by the fact that the Generic Recognition Step returns the number of results in the result collection. The Generic Recognition Result Info step in turn returns the number of interpretations within a selected result.

This information can be used along with the Go To step and the Conditional step to walk through all the interpretations in all the results.

Note There is no way to "*introspect*" the set of slots returned in an interpretation. The script must be designed along with the grammar so that it knows what slots are going to be filled in by the grammar for each interpretation.

The Customizer Window of the Generic Recognition Step

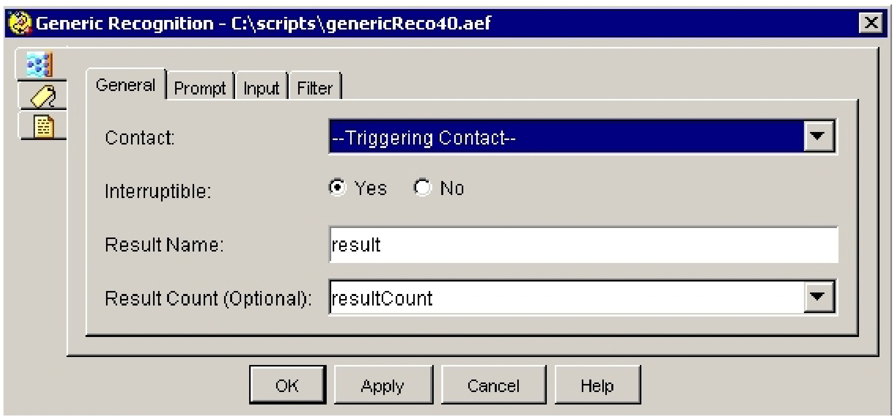
The customizer window of the Generic Recognition step contains four tabs:

- General tab (Generic Recognition step)
- Prompt tab (Generic Recognition step)
- Input tab (Generic Recognition step)
- Filter tab (Generic Recognition step)

General tab (Generic Recognition step)

Use the General tab of the Generic Recognition step to set the contact, the interruptible option, and the recognition result information.

Figure 2-114 Generic Recognition Customizer Window—General Tab



Contact: --Triggering Contact--
Interruptible: [✓ Yes] No
Result Name: result
Result Count (Optional): resultCount

Table 2-97 describes the properties of the General tab of the Generic Recognition customizer window.

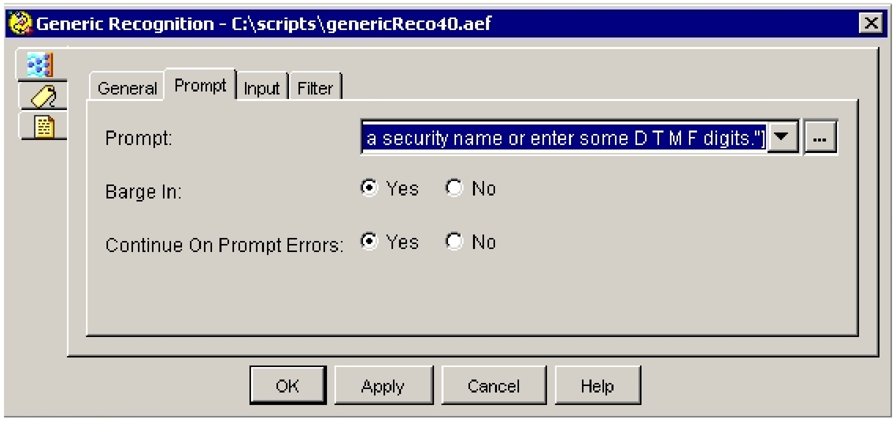
Table 2-97 Generic Recognition Properties—General Tab

Field name	Type	Description
Contact	Contact	The contact that is used by the step. The default contact is the Triggering Contact, unless another contact is specified.
Interruptible	Boolean	Yes-An external event (such as an agent becoming available or a caller hanging up) can interrupt this step. No-The step must complete before any other process can execute.
Result name	String literal	A user-specified name that identifies the recognition result. A temporary private variable is created for the duration of the call to hold the recognition result. The same result name is used in the "Get Recognition Result Info" and "Get Recognition Interpretation" steps to access the recognition data.
Result count	Integer	The number of results in the recognition result is returned in the variable specified in this field. This is optional because a user may only wish to determine if recognition was successful or not.

Prompt tab (Generic Recognition step)

Use the Prompt tab of the Generic Recognition step to configure the prompt for this step.

Figure 2-115 Generic Recognition Customizer Window—Prompt Tab


Generic Recognition - C:\scripts\genericReco40.aef

Prompt: a security name or enter some D T M F digits."]

Barge In: [✓ Yes] No

Continue On Prompt Errors: [✓ Yes] No

Table 2-98 describes the properties of the Prompt tab of the Generic Recognition customizer window.

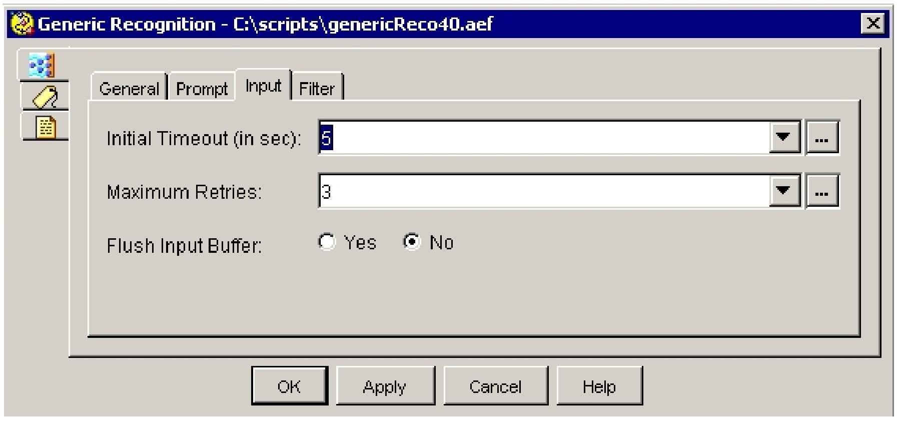
Table 2-98 Generic Recognition Properties—Prompt Tab

Properties	Description
Prompt	Variable or expression indicating the prompt to be played.
Barge In	Yes—The caller can interrupt the prompt. No—The prompt must complete playback before the caller can respond.
Continue on Prompt Errors	Yes—The step continues with the next prompt in the list if a prompt error occurs, or, if this prompt was the last in the list, the step waits for input from the caller. No—An exception results, which can then be handled in the script.

Input tab (Generic Recognition step)

Use the Input tab of the Generic Recognition step to control how long it waits for input.

Figure 2-116 Generic Recognition Customizer Window—Input Tab


Generic Recognition - C:\scripts\genericReco40.aef

General | Prompt | Input | Filter

Initial Timeout (in sec): 5

Maximum Retries: 3

Flush Input Buffer: Yes [✓ No]

Table 2-99 describes the properties of the Input tab of the Generic Recognition customizer window.

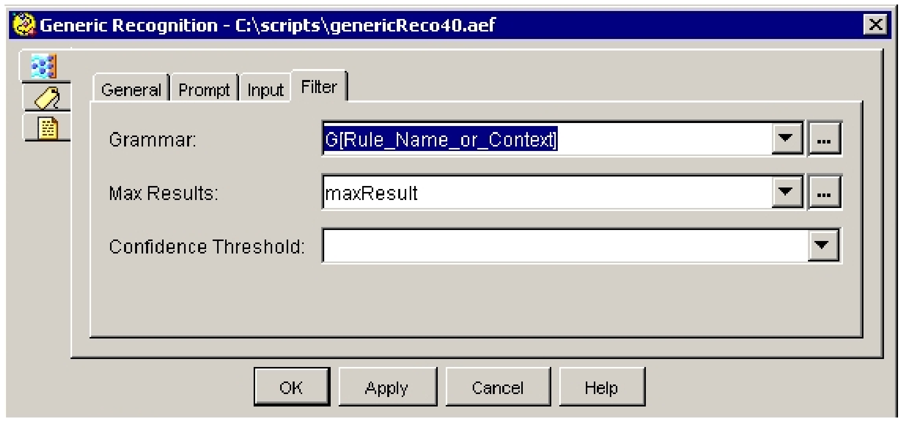
Table 2-99 Generic Recognition Properties—Input Tab

Properties	Description
Initial Timeout	Amount of time that the step waits for a response before timing out. Enter delay time in seconds or enter an expression.
Maximum Retries	Number of times a new entry can be entered after a timeout or invalid key.
Flush Input Buffer	Yes—The system erases previously entered DTMF input before capturing caller input. No—The system does not erase previously entered input before capturing caller input.

Filter tab (Generic Recognition step)

Use the Filter tab of the Generic Recognition step to specify the recognition grammar and to control what recognition results are returned.

Figure 2-117 Generic Recognition Customizer Window—Filter Tab


Generic Recognition - C:\scripts\genericReco40.aef

General | Prompt | Input | Filter

Grammar: G[Rule_Name_or_Context]
Max Results: maxResult
Confidence Threshold:

Table 2-100 describes the properties of the Filter tab of the Generic Recognition customizer window.

Table 2-100 Generic Recognition Properties—Filter Tab

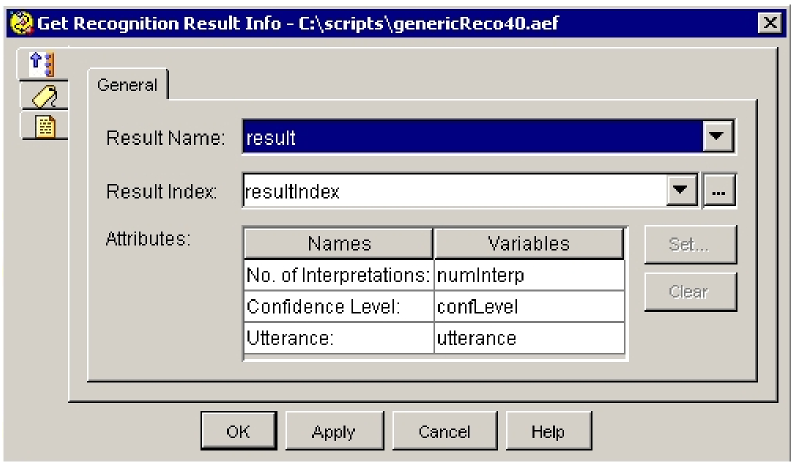
Field name	Type	Description
Grammar	Grammar	Identifies the grammar that should be used for this recognition. Enter a grammar expression. It should resolve to an SRGS grammar or SRGS grammar file.
Max results	Integer	Specifies the maximum number of results that can be returned by this step. For example, if the recognition process generates 5 results and this value is set to 3, only the best 3 results will be returned.
Confidence Threshold	Integer	Specifies the minimum confidence level for acceptable results. All results returned will have a threshold greater than or equal to this value. The allowable range is from 0-100. To use the platform default confidence threshold, simply specify an empty field.

Get Recognition Result Info Step

Use the Get Recognition Result Info step to:

- Extract the Results from the recognition performed in the Generic Recognition step. There may be more than one result. The Result Index is used to select the desired result. The range is 0 (zero) to one less than the number of results. Each result will contain one or more interpretations.
- Extract the number of interpretations for a result.
- Extract the confidence level for each result.
- Extract the string which represents the actual utterance that was recognized for each result.

Figure 2-118 Get Recognition Result Info Customizer Window—General Tab



Result Name: result

Result Index: resultIndex

Attributes:

Names	Variables
No. of Interpretations:	numInterp
Confidence Level:	confLevel
Utterance:	utterance

Table 2-101 describes the properties of the General tab of the Get Recognition Result Info customizer window.

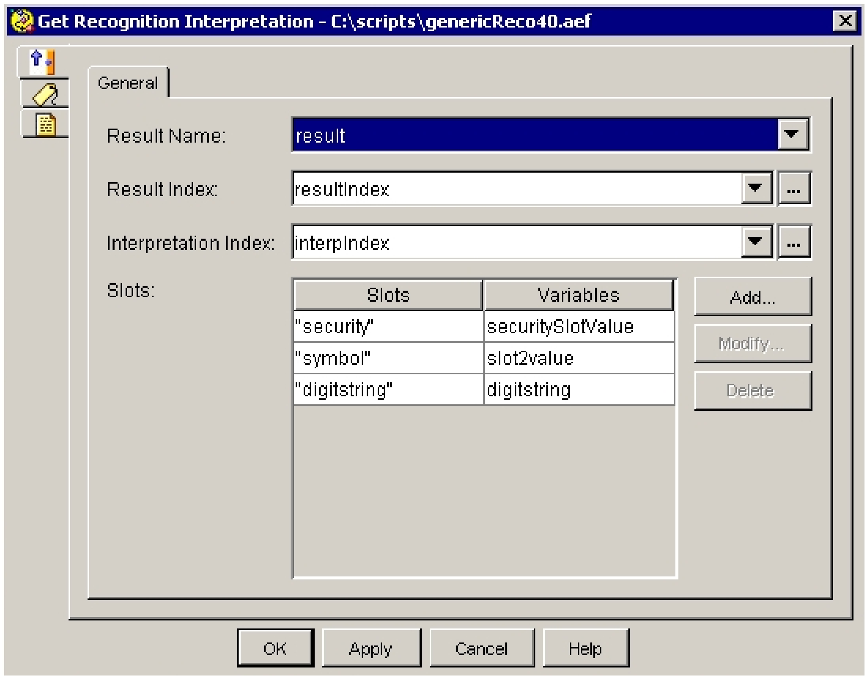
Table 2-101 Get Recognition Result Info Customizer Window—General Tab

Field name	Type	Description
Result Name	String literal	A name that identifies the recognition result collection from which to access result information. This is the same name that was specified in the Generic Recognition step.
Result Index	Integer	A number that selects which result to access from the collection of result objects returned by the Generic Recognition step. This value should be less than the Result count value returned in the Generic Recognition step
Attributes		This table is used to specify variables to which to extract the desired values of the selected result.
	Integer	The number of Interpretations: This will receive the total number of interpretations in the selected result.
	Integer	The Confidence Level: This will receive the actual confidence level of the selected result.
	String	The Utterance: This receives the actual spoken text of the selected result.

Get Recognition Interpretation Step

Use the Get Recognition Interpretation step to extract the Interpretation from a specific result. This is done by using the Result and Interpretation indices to select the appropriate Interpretation from the recognition result.

Figure 2-119 Get Recognition Interpretation Customizer Window—General Tab


Get Recognition Interpretation - C:\scripts\genericReco40.aef

General

Result Name: result

Result Index: resultIndex

Interpretation Index: interpIndex

Slots:

Slots	Variables
"security"	securitySlotValue
"symbol"	slot2value
"digitstring"	digitstring

Table 2-102 describes the properties of the General tab of the Get Recognition Interpretation step customizer window.

Table 2-102 Properties—General Tab

Field name	Type	Description
Result Name	String literal	A Name that identifies the recognition result collection from which to access result information. This is the same name that was specified in the Generic Recognition step.
Result Index	Integer	A number that selects which result to access from the collection of result objects returned by the Generic Recognition step. This value should be less than the Result Count value returned in the Generic Recognition step.
Interpretation Index	Integer	A number that selects which interpretation within the selected result to access. This value should be less than the Number of Interpretations returned from the Get Recognition Result Info step.
Slots	Slots/Variables	Attributes specified in the grammar for interpretations. Each interpretation can have zero or more slots. A slot is something into which you can put meaning. For example, a UserID, or an E-Mail address, or a phone number. Each slot is associated with a script string variable into which the value of the slot is extracted.

User Steps

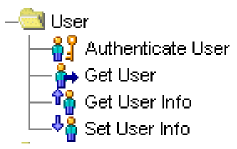
The steps in the User palette of the Cisco Unified CCX Editor provide designers with a way to authenticate, retrieve, and assign user attributes.

This section contains the following topics:

- Authenticate User Step
- Get User Step
- Get User Info Step
- Set User Info Step

Figure 2-120 shows the steps in the User palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Figure 2-120 User Palette Steps

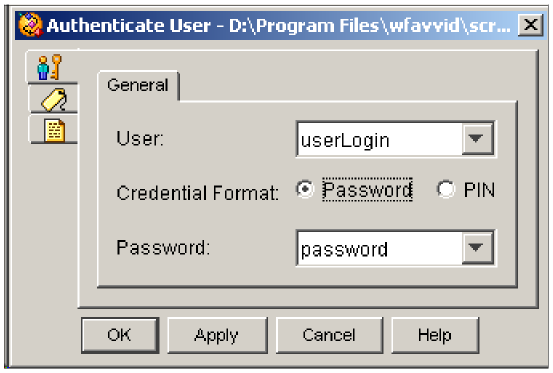


Authenticate User Step

Use the Authenticate User step to authenticate user identification.

Note This step is required in a script before you can use the Upload Prompt, Upload Grammar, or Upload Document step in that script. Only an authenticated user can upload a prompt, grammar, or document.

Figure 2-121 Authenticate User Customizer Window


Authenticate User - D:\Program Files\wfavvid\scr...

General

User: userLogin

Credential Format: [✓ Password] PIN

Password: password

You can, for example, use this step as part of a login (and logout) service that performs the necessary administrative updates when users authenticate themselves at a phone.

The Authenticate User step uses the information provided in the customizer (a user object representing the user and corresponding password or Personal Identification Number (PIN) to authenticate a user by comparing the information supplied with the information entered for this user in the Cisco Unified CCX Administration web interface.

The Authenticate User step has two output branches:

- Success
- Fail

Table 2-103 describes the properties of the Authenticate User customizer window.

Table 2-103 Authenticate User Properties

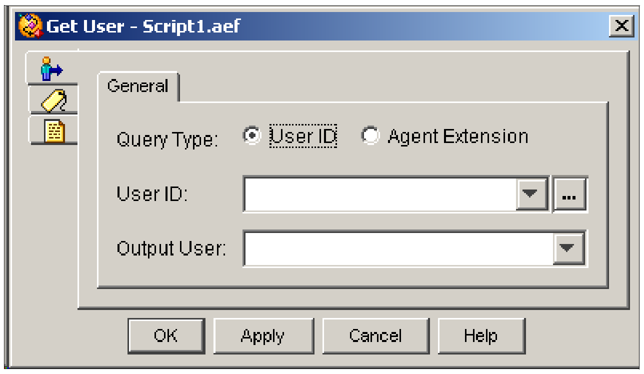
Property	Description
User	Variable storing the user object.
Credential Format (radio button)	Credential Format—Check boxes indicates whether you are specifying a password or PIN variable. **Note** If using a PIN, the step attempts to authenticate using the PIN entered in the user's Cisco Unified CCX Administration web interface instead of the alphanumeric password. If using a Password, the application attempts to authenticate the user against the user's actual alphanumeric password.
Pin (field)	A variable containing a user PIN.

Get User Step

Use the Get User step to create a variable based on a given user ID or agent extension.

Note To support E.164 compliance, Unified CCX allows you to add "+" preceding to an agent extension or a route point directory number.

Figure 2-122 Get User Customizer Window



Query Type: [✓ User ID] Agent Extension
User ID:
Output User:

The output of the user variable can be passed into the Select Resource Step that is required for Agent Based Routing or other steps that take a user variable as an input.

The Get User Step has two output branches:

- Successful—If the step can uniquely identify a user, based on the user input, it returns through Successful branch.
- Unsuccessful—If the step identifies more than one user or encounters any error, it goes through Unsuccessful branch.

Table 2-104 describes the properties of the Get User customizer window.

Table 2-104 Get User Properties

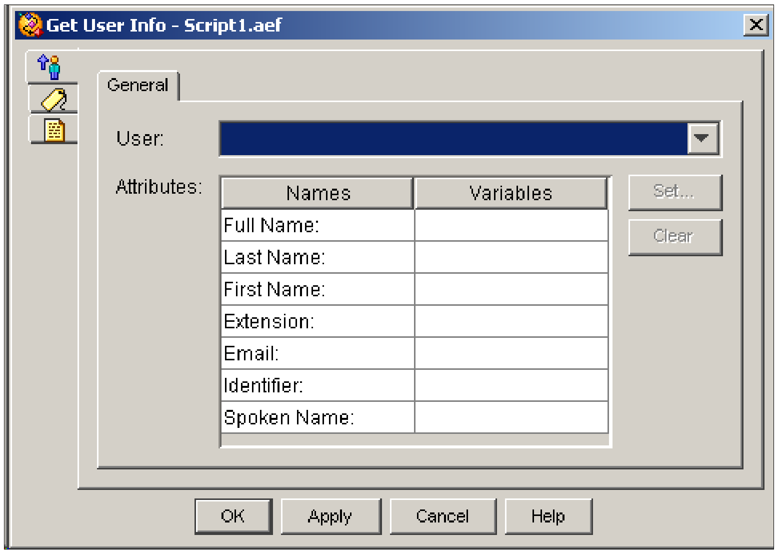
Properties / Buttons	Description
Query Type	Variable indicating the retrieval method. One of the following: • User ID • Agent Extension **Note** If this field is set to Agent Extension, the User ID field is renamed to Agent Extension.
User ID	Variable or expression identifying the User ID of a user or Agent Extension of a Cisco Unified CCX user.
Output User	Variable represents the User Extension returned by the system.

Get User Info Step

Use the Get User Info step to make user attributes available to the script.

Note To support E.164 compliance, Unified CCX allows you to add "+" preceding to an agent extension or a route point directory number.

Figure 2-123 Get User Info Customizer Window


Get User Info - Script1.aef

General

User:

Attributes:

Names	Variables
Full Name:	
Last Name:	
First Name:	
Extension:	
Email:	
Identifier:	
Spoken Name:	

Table 2-105 describes the properties of the Get User Info customizer window.

Table 2-105 Get User Info Properties

Property	Description
User	The variable that identifies the user for whom you want to get information.
Attributes (Names and Variables)	The attribute names with the associated variables for the user. **Note** For complete details about the available Get User attributes, see Table 2-106.
Set (button)	To set a variable, select an Attribute name and click **Set**. Choose a variable from the Select Variable drop-down list and then click **OK**; the variable name appears in the Variable column next to the attribute you selected.
Clear (button)	To remove User information, highlight a value in the list and click **Clear**.

Table 2-106 describes the attributes that can be retrieved by using the Get User Info step.

Table 2-106 Get User Info Attributes

Attribute	Description
E-mail	String representing the e-mail ID for this user. The user ID field is currently returned.
Extension	String representing the primary extension selected in the User pages of the Cisco Unified CCX Administration web interface.
First Name	String for the first name of the user.
Full Name	String for the full name of the user as configured in the Cisco Unified CCX Administration web interface.

Table 2-106 Get User Info Attributes (continued)

Attribute	Description
Last Name	String for the last name of the user.
Spoken Name	Document object representing the recorded name of the user.
Identifier (User ID)	The user ID of the user as configured in the Cisco Unified Communications Manager.

Set User Info Step

Use the Set User Info step to assign values to user attributes.

Figure 2-124 Set User Info Customizer Window

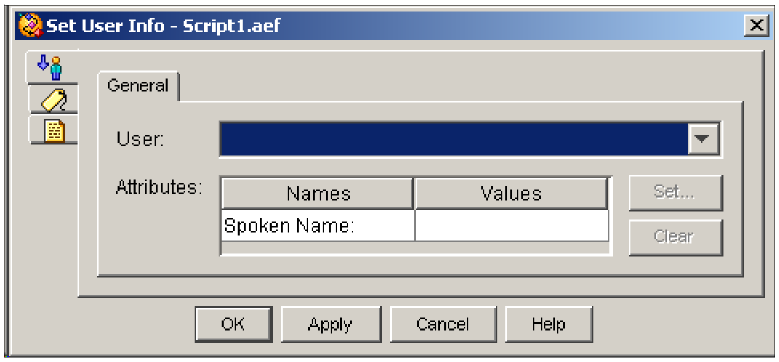


Table 2-107 describes the properties of the Set User Info customizer window.

Table 2-107 Set User Info Properties

Property	Description
User	The user variable that identifies the user for which you want to set information.
Attributes (Names and Values)	The attribute names with the assigned values for the user. **Note** The Set User Info step supports only the Spoken Name attribute, for which the expected value must be a Document object that represents a recorded prompt.
Set (button)	To set a variable, select an Attribute name and click **Set**. Choose a variable from the Select Variable drop-down list and then click **OK**; the variable name appears in the Variable column next to the attribute you selected.
Clear (button)	To remove User information, highlight a value in the list and click **Clear**.

Prompt Steps

The steps in the Prompt palette of the Cisco Unified CCX Editor provide script designers with a way to create intelligent prompts.

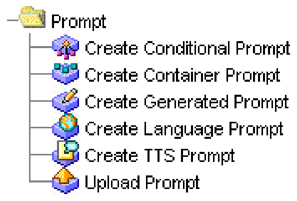
This section contains the following topics:

- Create Conditional Prompt
- Create Container Prompt
- Create Generated Prompt
- Create Language Prompt
- Create TTS Prompt
- Upload Prompt Step

Figure 2-125 shows the steps in the Prompt palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Note The Create TTS Prompt step is only available for the Cisco UnifiedIP IVR or Cisco Unified CCX Premium license options.

Figure 2-125 Prompt Palette Steps



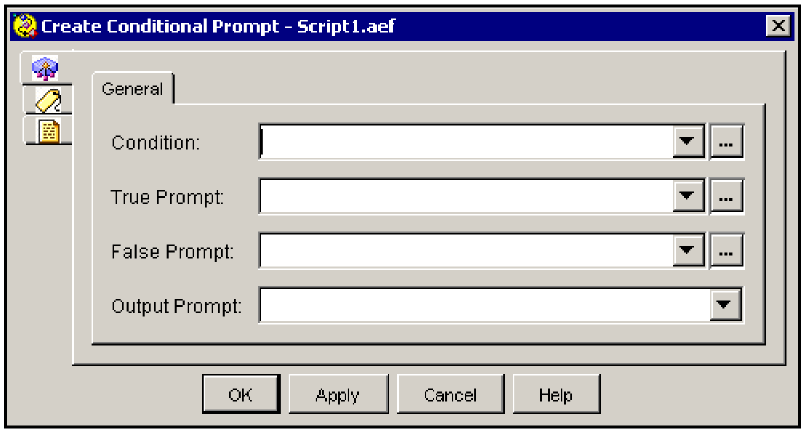
Create Conditional Prompt

Use the Create Conditional Prompt step to create a prompt based on the result of evaluating a specified Boolean expression.

Note Optionally, you can use the Expression Language and specify a conditional prompt: <*condition*> ? <*true prompt*> : <*false prompt*>

where <*condition*> is a boolean expression, <*true prompt*> is a prompt expression to play out if the boolean expression is true, and <*false prompt*> is a prompt expression to play out if the boolean expression is false.

Figure 2-126 Create Conditional Prompt Customizer Window



The prompts passed are evaluated immediately as prompt objects, but they are not resolved until the time of playback. This means that if the values of any variables entered as part of the expression change between the time this prompt was created and the time the prompt is played back, then the new value of the variable is used to evaluate the conditional expression.

As an example, you can write a script that offers callers the choice between pressing digits or speaking a number. The conditional prompt provides prompts for both situations and lets the caller's choice determine which prompt the script plays in the given case.

Table 2-108 describes the properties of the Create Conditional Prompt customizer window.

Table 2-108 Create Conditional Prompt Properties

Property	Description
Condition	A Boolean variable—or a Boolean expression—indicating whether the script uses to decide which one of the two prompts to play back.
True Prompt	Variable or expression indicating the prompt to be used if the expression is True.
False Prompt	Variable or expression indicating the prompt to be used if the expression is False.
Output Prompt	Variable that stores the prompt that results from the Create Conditional Prompt step.

Create Container Prompt

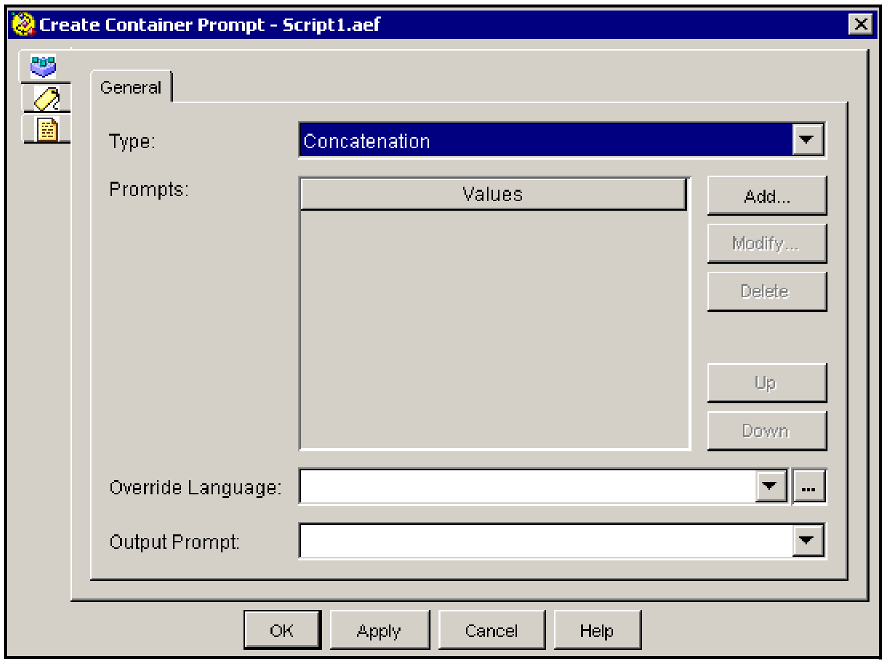
Use the Create Container Prompt step to combine multiple prompts into one larger prompt.

Note Optionally, you can use the Expression Language to specify container prompts.
where <*prompt 1*>, <*prompt 2*> and <*prompt 3*> are prompt expressions:

- <*prompt 1*> + <*prompt 2*> + <*prompt 3*> would be a Concatenated prompt

- *<prompt 1>* || *<prompt 2>* || *<prompt 3>* would be an Escalating prompt
- *<prompt 1>* % *<weight1>* || *<prompt 2>* % *<weight2>* ||
  <prompt 3> % *<weight3>* would be a Random prompt.

Figure 2-127 Create Container Prompt Customizer Window



Create Container Prompt - Script1.aef

General

Type: Concatenation

Prompts:

Override Language:

Output Prompt:

You can create three types of container prompts:

- **Concatenated Prompt**—Contains a list of prompt phrases that are played back in a specific sequence.

  For example, for a prompt of "Your checking account balance is one hundred and sixty-eight dollars", you can create a concatenated prompt that (1) begins with a user prompt "Your"; (2) continues with a conditional prompt that specifies a condition such as <accountType == "check">, and plays "checking account" if the condition is True or "savings account" if the condition is False; and (3) ends with the balance amount.

- **Escalating Prompt**—Provides an initial question prompt with a minimal amount of information at first, and then adds additional prompt phrases if no response is given.

  For example, for a prompt that provides the caller with more information as needed, you can create an escalating prompt that, when passed to a media step such as the Get Digit String step, begins by playing the first concise prompt inside the escalating prompt, such as "What is your account number?"

  If the step fails to collect the account number due to the caller's failure to provide it, a second prompt plays, such as "Please provide your account number by either pressing the account number using your touch tone phone followed by the pound key, or simply speaking out the account number digits."

- **Random Prompt**—Plays back a series of promotional or informational messages in a random order while a caller is waiting for an available agent.

The Table 2-109 describes the properties of the Create Container Prompt customizer window.

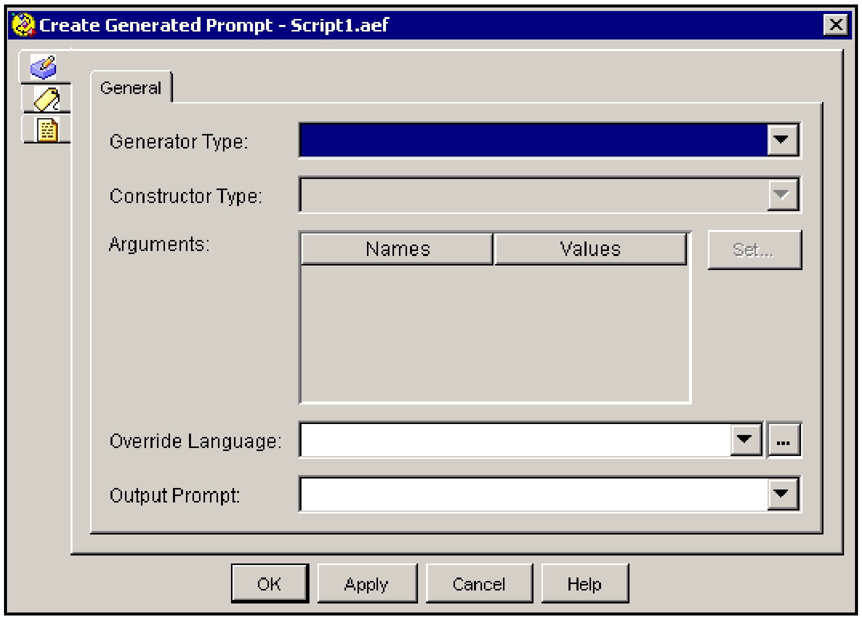
Table 2-109 Create Container Prompt Properties

Properties / Buttons	Description
Type	Concatenated, Escalating, or Random prompt.
Prompts	List of prompts to be combined into the container prompt **Note** For Concatenated and Escalating prompts, use the **Up** and **Down** buttons to determine the order of playback of the prompts in the concatenated prompt. For Random prompts, use the **Up** and **Down** buttons to make an individual prompt play more or less often than other prompts.
Override Language (optional)	Variable or expression indicating language in which the prompts are played back.
Output Prompt	Script variable that holds the combined prompt generated by the Create Container Prompt step
Add / Modify (buttons)	Use these buttons to access the Add Prompt dialog box. Use the dialog to specify the following: • **Prompt**—Variable or expression containing the prompt. • **Weight**—(Appears only when the prompt field is set to Random.) Represent the priority of the prompt in the sequence. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove the Prompt information, highlight a value in the list and click **Delete**.

Create Generated Prompt

Use the Create Generated Prompt step to create prompt phrases from intermediate variables whose values are dynamically determined based on run-time script information.

Figure 2-128 Create Generated Prompt Customizer Window



For example, you can create the prompt phrase of “account balance is one hundred and sixty-eight dollars” by querying the database of account balances at a particular point in the script and using a currency generator to generate the number.

Note The Create Generated Prompt step accepts only the 4-digit year format only. A 3-digit date format is not accepted.

Note If the Generate Prompt step encounters an invalid time, it outputs 4:00 P.M. Specify a valid time between 0000 and 2400.

Table 2-110 describes the properties of the Create Generated Prompt customizer window.

Table 2-110 Create Generated Prompt Properties

Property	Description
Generator Type	Variable indicating the type of information generated. **Note** See Create Generated Prompt Step Generator Types, for information about supported generator types.
Constructor Type	Variable indicating the constructor type that corresponds to the generator type. **Note** When you choose the constructor type, the constructors automatically appear in the Argument column of the Argument list box.
Arguments (Names and Values)	Names and their values.
Override Language (Optional)	Language in which the prompt is played back. Used only if the resulting prompt is played in a different language than the one defined by the contact in which that prompt is played back.
Output Prompt	Variable indicating where the prompt object resulting from this step is stored.
Set (button)	To set a variable, select an Argument name and click **Set**. The Variable dialog box appears. Select a variable or expression indicating variable that holds the value for the argument and then click OK; the name or the argument and its value appear in the Argument Information list box. **Note** You must define all arguments listed with some value or define them as null; you cannot leave them blank.

Create Generated Prompt Step Generator Types

The Create Generated Prompt step supports the following 12 generator types:

- Number
- Character
- Spelling
- Date
- Time
- Ordinal
- Currency
- Country
- Language
- Telephone Number
- Credit Card Number
- Credit Card Expiration Date

The following sections describe these supported generator types.

Number

The Number generator type supports the following constructors:

- (Number number)
- (String number)
- (Number number, Number gender)
- (String number, Number gender)
- (Number number, Boolean play.full)
- (String number, Boolean play.full)
- (Number number, Boolean play.full, Number gender)
- (Number number, Boolean play.full, Number gender)

The three parameters are:

- **Number**—Any Number object (for example; Integer, Long, Float, Double, BigInteger, BigDecimal) or String object defining the number to be played back.
- **Gender**—When the number must be played back in a specific gender context, this parameter specifies the context. Valid values are 0 for neutral, 1 for male, and 2 for female.

Note If the language associated with the call does not behave differently based on gender, this parameter is ignored.

- **Play.full**—Plays the number in full format if this optional Boolean argument is true or omitted. (For example, "709" is played as "Seven Hundred and Nine".) Otherwise, the number plays in brief format. (For example, "709" is played as "Seven Oh Nine".)

Note If the number is played in full format, the maximum number supported is +/- 999,999,999,999.

Character

The Character generator type supports the following constructors:

- (Character character)
- Character character, Boolean play_all)

The two parameters are:

- **Character**—The character object to be played back.
- **Play_all**—Optional Boolean flag indicating whether to play spaces, punctuation, and other special characters normally instead of playing them as silence (ranging from 250ms to 500ms).

Spelling

The Spelling generator type supports the following constructors:

- (String string)
- (String string, Boolean punctuation)

- (Object object)
- (Object object, Boolean punctuation)

The three parameters are:

- **String**—String object to be played back.
- **Object**—Object for which the string representation returned by the `String.valueOf()` method should be spelled out.
- **Punctuation**—An optional Boolean flag indicating whether to play spaces, punctuations, and special characters normally or as silences.

Note Punctuation default behavior in the Spelling generator is different from Play-all default behavior in the Character generator.

Date

The Date generator type supports the following constructors:

- (Date date)
- (Date date, Boolean skip.current.year)
- (Number year)
- (Number year, Number month)
- (Number year, Number month, Boolean skip.current.year)
- (Number year, Number month, Number day)
- (Number year, Number month, Number day, Boolean skip.current.year)

The five parameters are:

- **Date**—Any Date object from which to extract the date to be played back.
- **Skip.current.year**—If set to true, the year does not play back if it is the same as the current year.
- **Year**—The year of the date to be played back. This year must be specified in full (for example, 2005).

Note Any number given is played, so it is the responsibility of the caller to ensure that the specified year is valid.

- **Month**—The month of the date to be played back. Valid values range from 1 to 12, where 1 represents January and 12 represents December.
- **Day**—The day of the date to be played back. Valid values range from 1 to 31 and are validated at run time based on the specified month and year.

Time

The Time generator type supports the following constructors:

- (Time)
- (Hours, Minutes)

The three parameters are:

- **Time**—Any Date or Time object representing the time to be played back. Time can also be defined as a Number object (Integer, Float, Long, and so forth) that specifies the time to be played, from 0 to 2359. (For example, a number such as 1234 is played as “12 34 PM.”) If the value specified is greater than 2359, then Time is considered to be the number of milliseconds since the standard base time known as “the epoch,” namely January 1, 1970, 00:00:00 GMT.
- **Hours**—Number object that specifies the hour to be played.
- **Minutes**—Number object that specifies the minutes to be played.

Ordinal

The Ordinal generator type supports the following constructors:

- (Number number)
- (String number)
- (Number number, Number gender)
- (String number, Number gender)

The two parameters are:

- **Number**—Any Number or String object defining the ordinal number to be played back. The supported range is from 1 to 999999.
- **Gender**—When the ordinal number must be played back in a specific gender context, this parameter specifies this context. Valid values are 0 for neutral, 1 for male, and 2 for female.

Note If the language associated with the call does not behave differently based on gender, then this parameter is ignored.

Currency

The Currency generator type supports the following constructors:

- (Currency designator)
- (Number amount)
- (Number amount, Currency currency)
- (Number dollar, Number cent)
- (Number dollar, Number cent, Currency currency)
- (Number amount, Boolean colloquial)
- (Number amount, Boolean colloquial, Currency currency)
- (Number dollar, Number cent, Boolean colloquial, Currency currency)

The six parameters are:

- **Designator**—The designator of a currency to play back. (For example, “USD” is played back as “U.S. Dollar”.)
- **Amount**—The currency amount to be played back in the system configured default currency or in the specified currency.

- **Dollar**—Number object representing the amount of currency unit to be played. Only the integer part of the number is played. The fractional part, if any, is ignored.
- **Cent**—Number object representing the currency subdivision to be played. Only the integer part of the number is played. The fractional part, if any, is ignored.

Note If the number specified exceeds the maximum value allowed for the subdivisions, the excess is added properly to the number of currency unit. For example, specifying “5 dollars and 233” cents results in “7 dollars and 33 cents”.

- **Colloquial**—An optional Boolean flag, which specifies whether to use colloquial currencies’ representations (for example, “Dollars” instead of “US Dollars”). If omitted, the currency amount is played in colloquial format.

Note This field does not apply for currency first denomination. For example 1 dollar is termed as 1 US Dollar or un dolar americano etc based on the language.

- **Currency**—The currency in which the amount should be played back. If not specified, the system default configured currency is played back.

Country

The Country generator type supports only one constructor: (Language language). The parameter “language” is a Language object from which to get the language to be played back. (For example, en_US is played back as “United States English”.)

Language

The Language generator type supports only one constructor: (Language language). The parameter “language” is a Language object from which to get the country to be played back. (For example, en_US is played back as “United States English”.)

Telephone Number

The Telephone Number generator type supports only one constructor: (String number). The parameter “number” is a String object specifying the telephone number to be played out as a sequence of digits.

The character is replaced with 250 ms of silence if the string contains any of the following characters: “ - ( ) `.` Otherwise, the string is automatically formatted.

Automatic formatting of the string inserts 250 ms of silence between sections of digits. These sections follow the following rule: “XXX-XXX-XXX-XXXX” unless there are exactly five digits in the string, in which case the string is considered to be a single section of five digits.

An “x” character is played back as “Extension”. DTMF digits (“`ABCD0123456789#*`”) are played back normally.

A string of the form “`*xx`” where `x` is a Dual Tone Multi-Frequency (DTMF) digit (“`0123456789`”) is played back as “star `xx`” (for example,“`*69`” is played back as “star sixty-nine”).

Credit Card Number

The Credit Card Number generator type supports only one constructor: (String number). The parameter "number" is a String object specifying the credit card number to be played out as a sequence of digits.

If the specified credit card number includes "-", then it is played as is, replacing the "-" character with 250 ms of silence; otherwise the number is automatically separated into sections of four digits and played back with 250 ms of silence inserted between sections.

Credit Card Expiration Date

The Credit Card Expiration Date generator type supports the following constructors:

- (Number year, Number month, Number day)
- (Number year, Number month)

The parameters are identical to the following Generated Date constructors:

- If day is 0 or omitted—GeneratedDate (year, month, true)
- All other cases—GeneratedDate (year, month, day, true)

Create Language Prompt

Use the Create Language Prompt step to input a set of prompts corresponding to different languages.

You can use the Create Language Prompt step to adapt concatenating prompts to the sentence structures of different languages.

For example, a normal grammar sequence for an English sentence is Subject + Verb + Object. For a Japanese sentence, it is Subject + Object + Verb.

The selection of the prompt is based on the standard search for a matching language. For example, assuming a language context of {L[fr_FR}, L[en_GB]}, the search returns the first prompt defined for the following languages: L[fr_FR], L[fr], L[en_GB], L[en], and finally L[ ].

Table 2-111 describes the properties of the Create Language Prompt customizer window.

Table 2-111 Create Language Prompt Properties

Properties / Buttons	Description
Options (Languages and Prompts	Prompt expression names or prompt expressions and all the languages that have been entered.
Output Prompt	Script variable where the prompt that results from the Create Language Prompt step is stored

Table 2-111 Create Language Prompt Properties (continued)

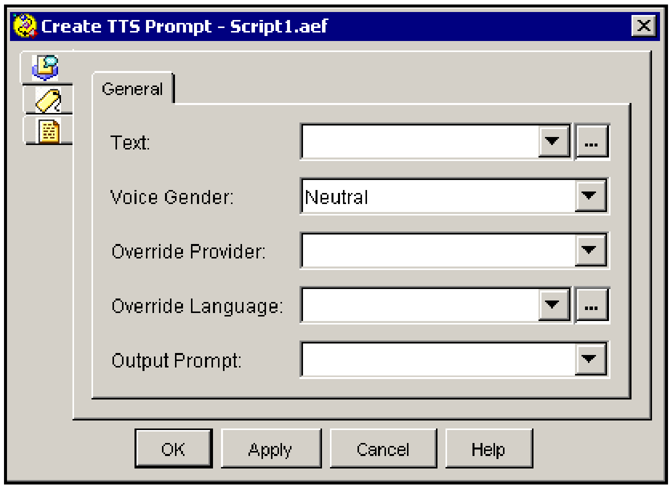
Properties / Buttons	Description
Add / Modify (buttons)	Use these buttons to access the Add Prompt dialog box. Use the dialog to specify the following: • **Language**—Variable or expression containing the language. • **Prompt**—Variable or expression containing the value for the prompt. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove the Prompt information, highlight a value in the list and click **Delete**.

Create TTS Prompt

Use the Create Text-to-Speech (TTS) Prompt step to play back the text from a string or document expression as speech, using one of the installed TTS servers.

Note The Create TTS Prompt step is only available for the Cisco UnifiedIP IVR or Cisco Unified CCX Premium license options.

Figure 2-129 Create TTS Prompt Customizer Window


Create TTS Prompt - Script1.aef

General

Text:

Voice Gender: Neutral

Override Provider:

Override Language:

Output Prompt:

By default, the language of the TTS Prompt is determined by the language in which the prompt is going to be played back. You have the option of overriding this default and specifying another language. However, you should keep this field up to date with the language of the text passed in to ensure proper interpretation of the text.

Note If the script cannot find the Override Language (or the language defined for the contact if Override Language is undefined) from the initialized TTS server list, this TTS request will be denied.

Table 2-112 describes the properties of the TTS Prompt customizer window.

Table 2-112 Create TTS Prompt Properties

Properties / Buttons	Description
Text	Variable or expression indicating the location of the string or document expression to be played.
Voice Gender	Variable indicating the gender of the simulated voice to be used for the prompt. Voice gender can be male, female, or default, if supported by the TTS provider. If not supported, the system automatically falls back to a supported voice gender.
Override Provider	(optional) Variable or expression indicating a different TTS provider to be used where the prompt is played back instead of the provider defined for the contact.
Override Language	(optional) Variable or expression indicating a different language to be used where the prompt is played back instead of the language defined for the contact. **Note** To ensure proper conversion, keep this field current with the language of the text passed in.
Output Prompt	Variable that stores the prompt that results from this step.

Upload Prompt Step

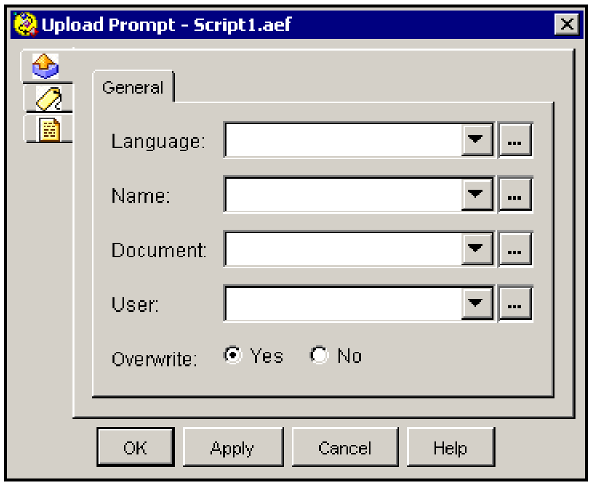
Use the Upload Prompt step to add a user prompt to the Prompt repository. Uploading a prompt makes it accessible to all the Cisco Unified CCX servers in the cluster and allows it to be backed up with all other repository data.

For example, you can upload a recorded announcement to the prompt repository that replaces a "Message of Day" announcement.

Note The Upload Prompt step will not work unless the user specified in the step is an authenticated user. Use the Authenticate User step before the Upload Prompt step to authenticate the user.

Warning **Although the prompt repository is meant to hold prompts used by scripts, do not use the Upload Prompt functionality to store all types of prompts. There is no provision for a delete prompt operation from a script. Also, the system will not behave properly if this step is over-utilized. This means that this step has not been designed to be extensively used by all calls coming into the system. Rather, this step is meant for updating prompts, like the "message-of-the-day" prompt, in the repository database from time to time.**

Figure 2-130 Upload Prompt Customizer Window



Language:
Name:
Document:
User:
Overwrite: [✓ Yes] No

Table 2-113 describes the properties of the Upload Prompt customizer window.

Table 2-113 Upload Prompt Properties

Property	Description
Language	Variable or expression indicating the prompt language you want to upload to.
Name	Variable or expression indicating the name of the prompt you want to upload. (This is relative to the language folder in the repository.)
Document	Variable or expression containing the prompt to be uploaded to the repository.
User	The authenticated user generating and uploading prompt.
Overwrite (radio buttons)	Select one of the following: • **Yes**—Select this option to overwrite the current data. • **No**—Select this option to retain the current data.

Grammar Steps

The steps in the Grammar palette of the Cisco Unified CCX Editor provide script designers with a way to specify a set of all possible spoken phrases and/or Dual Tone Multi-Frequency (DTMF) digits to be recognized by Cisco Unified CCX Solutions and acted on during run time.

Note If you specify a Grammar tag with incorrect case in a Grammar step, the script goes to the Unsuccessful branch when it reaches this step. The Validate function does not detect this case mismatch. Ensure the proper case is used in every instance to avoid this issue.

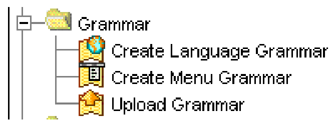
This section contains the following topics:

- Create Language Grammar Step
- Create Menu Grammar Step

- Upload Grammar Step
- Using Grammar Formats and Rules

Figure 2-131 shows the steps in the Grammar palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

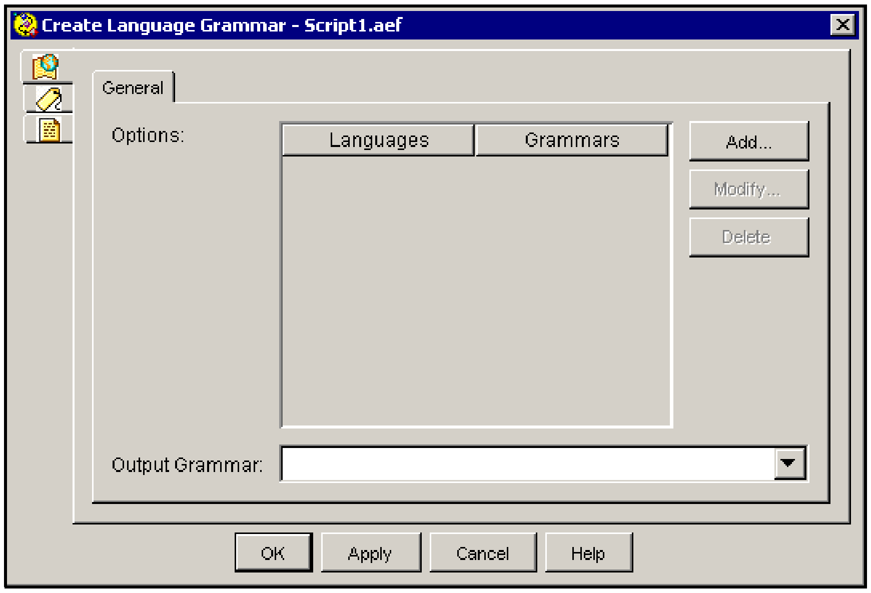
Figure 2-131 Grammar Palette Steps



Create Language Grammar Step

Use the Create Language Grammar step to select a set of grammars based on the language context of the call.

Figure 2-132 Create Language Grammar Customizer Window



Grammar selection is based on the standard search for a matching language. For example, assuming a language context of {L[fr_FR], L[en_GB]}, the search would return the first grammar defined for the following languages:

- L[fr_FR]
- L[fr]
- L[en_GB]
- L[en]
- L[ ]

Table 2-114 describes the properties of the Create Language Grammar step.

Table 2-114 Create Language Grammar Properties

Properties / Buttons	Description
Options (Languages and Grammars)	Languages with their corresponding Grammar expressions. This grammar can be the result of the Create Menu Grammar step or any grammar expression previously stored in a Grammar variable.
Output Grammar	Variable storing the language grammar information.

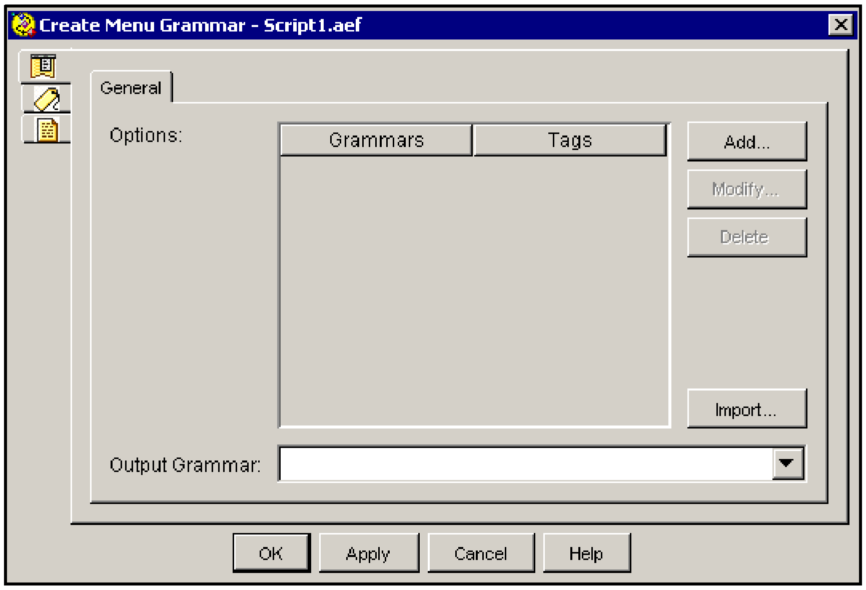
Table 2-114 Create Language Grammar Properties (continued)

Properties / Buttons	Description
Add / Modify (buttons)	Use these buttons to access the Add or Modify Option dialog box. Use the dialog to specify the following: • **Language**—Variable or expression that stores the language name for the grammar. **Note** If you select the Expression button in the Add Option dialog box, the Define Language dialog box appears. Either select the desired language from the Language display list, or enter the name of the desired language in the User Defined text field.(Click **OK** to close the dialog.) • **Grammar**—Variable that stores the grammar information for the language When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove a language supported for the grammar, highlight a value in the list and click **Delete**.

Create Menu Grammar Step

Use the Create Menu Grammar step to create spoken word and/or DTMF menus in which the user makes a single choice from multiple options.

Figure 2-133 Create Menu Grammar Customizer Window



Each step holds the equivalent phrases or digits for the menu in corresponding languages. For each Create Menu Grammar step you create, you must assign a script variable of type grammar as the output variable.

By default, the actual language in which any grammar is played is determined by the language associated with the contact. When you define multiple grammars to be available, the script chooses a single grammar based on the language of the contact.

Table 2-115 describes the properties of the Create Menu Grammar step.

Table 2-115 Create Menu Grammar Properties

Properties / Buttons	Description
Options (Grammars and Tags)	Phrases and/or digits with their corresponding tags identifying the grammars.
Output Grammar	Variable that stores the menu grammar information.
Add / Modify (buttons)	Use these buttons to access the Add or Modify Option dialog box. Use the dialog to specify the following: • **Grammar**—Variable or expression that stores the grammar information. • **Tag**—A tag name for the grammar. **Note** Tags for grammars are case-sensitive. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove a phrase from the grammar, highlight a value in the list and click **Delete**.

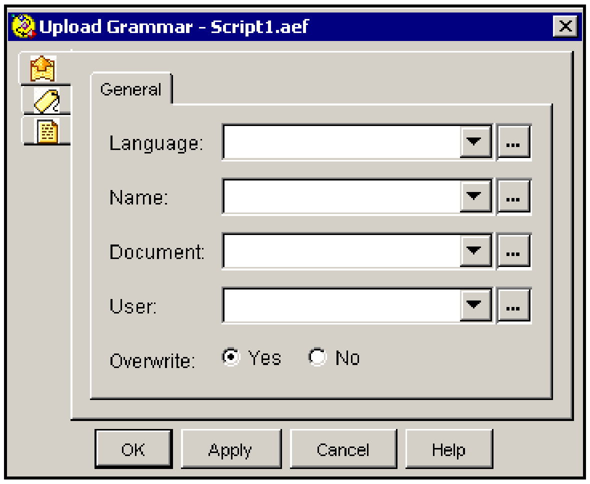
Upload Grammar Step

Use the Upload Grammar step to add a user grammar to the repository database. Uploading a grammar makes it accessible to all the Cisco Unified CCX servers in the cluster and allows it to be backed up with all other repository data.

Note The Upload Grammar step will not work unless the user specified in the step is an authenticated user. Use the Authenticate User step before the Upload Grammar step to authenticate the user.

Warning **Although the repository database is meant to hold grammars used by scripts, do not use the Upload Grammar functionality to store all types of grammars. There is no provision for a delete grammar operation from a script. Also, the system will not behave properly if this step is over-utilized. This means that this step has not been designed to be extensively used by all calls coming into the system. Rather, this step is meant for updating grammars in the repository database from time to time.**

Figure 2-134 Upload Grammar Customizer Window



Overwrite: (● Yes) ○ No

Table 2-116 describes the properties of the Upload Grammar customizer window.

Table 2-116 Upload Grammar Properties

Properties / Buttons	Description
Language	Variable or expression indicating the language you want to upload to.
Name	Variable or expression indicating the name of the grammar you want to upload. (This is relative to the language folder in the repository.)
Document	Variable or expression containing the grammar to be uploaded to the repository.
User	The authenticated user generating and uploading the grammar.
Overwrite (radio buttons)	Select one of the following: • **Yes**—Select this option to overwrite the current data. • **No**—Select this option to retain the current data.

Using Grammar Formats and Rules

The following sections describe the following:

- Using the SRGS Grammar Format
- Using GSL File Grammar Format (Deprecated)
- Using the Digit File Grammar Format

Using the SRGS Grammar Format

SRGS (Speech Recognition Grammar Specification) is a W3C standard. The following URL links to the current version of that standard: http://www.w3.org/TR/speech-grammar/.

Note The following are two example SRGS grammars. Each goes in a separate file. That is because an SRGS grammar can be either Voice mode or DTMF mode, but not both. Since voice is the default mode, it is does not need to be specified explicitly. Note, however, that it is possible to have both of these grammars active at the same time during recognition with a compound grammar. For further information on this topic, see "Compound Grammar" in the *Cisco Unified Contact Center Express Scripting and Development Series: Volume 1, Getting Started with Scripts.*

Figure 2-135 Two Example SRGS Grammars

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<grammar xml:lang="en-US" version="1.0"
         xmlns="http://www.w3.org/2001/06/grammar"
        root="main">
    <rule id="main" scope="public">
        <one-of>
            <item>
               hi
                <tag>tag="hi"</tag>
            </item>
            <item>
               joy
                <tag>tag="lg"</tag>
            </item>
        </one-of>
    </rule>
</grammar>

<?xml version="1.0" encoding="ISO-8859-1"?>
<grammar xml:lang="en-US" version="1.0"
         xmlns="http://www.w3.org/2001/06/grammar"
         root="main"
         mode="dtmf">
    <rule id="main" scope="public">
        <one-of>
            <item>
                *
                <tag>tag="bye"</tag>
            </item>
            <item>
                4
                <tag>tag="4"</tag>
            </item>
        </one-of>
    </rule>
</grammar>
```

Nuance does not follow the W3C SRGS specification exactly. The SRGS specification allows vendor specific tag formats and Nuance uses their own specific tag formats. Nuance also requires a slightly different placement of the <tag> element. The following is a copy of the preceding two grammars for use with Nuance:

Figure 2-136 Two Example SRGS Grammars from Nuance

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<grammar xml:lang="en-US" version="1.0"
xmlns="http://www.w3.org/2001/06/grammar"
         tag-format="nuance"
         root="main">
         root="main">
    <rule id="main" scope="public">
```

```
            <one-of>
                <item>
                    hi
                </item>
                <tag><![CDATA[<tag hi>]]></tag>
                <item>
                    joy
                </item>
                <tag><![CDATA[<tag lg>]]></tag>
            </one-of>
        </rule>
    </grammar>
    <?xml version="1.0" encoding="ISO-8859-1"?>
    <grammar xml:lang="en-US" version="1.0"
             xmlns="http://www.w3.org/2001/06/grammar"
             tag-format="nuance"
             root="main"
             mode="dtmf">
        <rule id="main" scope="public">
            <one-of>
                <item>
                    *
    </item>
                <tag><![CDATA[<tag bye>]]></tag>
                <item>
                    4
                </item>
                <tag><![CDATA[<tag 4>]]></tag>
            </one-of>
        </rule>
    </grammar>
```

Note By specifying tag-format in the SRGS header, Nuance programmers are able to use their Nuance proprietary syntax for representing semantic interpretation in the <tag> element. Note also that this Nuance syntax requires the use of CDATA to enclose the text in the <tag> element because the syntax uses syntactic elements that are also used by XML (that is < >). CDATA tells the XML parser to ignore these syntactic elements until it sees the end of the CDATA enclosure ( ]]>). This is described in the Extensible Markup Language (XML) specification at the following URL: http://www.w3.org/TR/REC-xml/. In particular, see the section: http://www.w3.org/TR/REC-xml/#sec-cdata-sect.

Using GSL File Grammar Format (Deprecated)

The Cisco Unified CCX Engine uses a language called Nuance Grammar Specification Language (GSL).

Table 2-117 describes sample expressions you can use to specify grammars in GSL.

Table 2-117 Grammar Expressions

Expression	Description
[ ]	Only one of the words in the list must match.
( )	All words in the list must match.
?	The following word or phrase is optional.
{ }	Slot value pair

The following are examples of GSL language statements:

- Digits [one two three four five six seven eight nine zero]
- HowMany [I [want need] Digits [pencil pencils]]

  In this example, the caller can say "I want" or "I need", followed by a number from zero to nine, and then either "pencil" or "pencils".

Using the Digit File Grammar Format

The Digit File Grammar Format (".digit") is based on a Java Properties File, in which keys are defined as "dtmf-x", where "x" is from the set "0123456789*#ABCD" or one of "star" or "pound", where values are the corresponding tag to be returned when a key is pressed or recognized.

You can also use an optional entry defined as "word=true" to specify that the word representation of each DTMF digit should be automatically included during a recognition.

For example:

```
word=true
dtmf-star=bye
dtmf-4=4
```

Document Steps

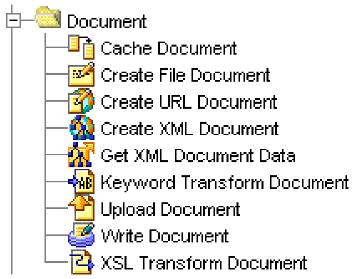
The steps in the Document palette of the Cisco Unified CCX Editor provide script designers with a way to handle various kinds of documents.

This section contains the following topics:

- Document Step Types
- Cache Document Step
- Create File Document Step
- Create URL Document Step
- Create XML Document Step
- Get XML Document Data Step
- Keyword Transform Document Step
- Write Document Step
- XSL Transform Document Step
- Upload Document Step
- Make REST Call

The figure that follows shows the steps in the Document palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Figure 2-137 Document Palette Steps



Document Step Types

The Document palette provides three types of Document steps, which are usually used in the following order: source, transformation, and output steps.

- Source steps—Used to define a document object using different parameters, such as a URL or a file name. Three Document steps are Source steps:
  - Create File Document
  - Create URL Document
  - XSL Transform Document

  Source steps define the parameters for a document variable but do not read data at the time they execute. You can use the Cache step to cause the system to obtain and store the document defined by a Source step. Otherwise, the document is read when passed to a transformation or output step.

  Note Other source steps used for documents are DB Get (a Database step), Recording (a Media step), Get User Info (a User step), and Get Session Info when retrieving document objects stored in the session (a Session step). For information on these steps, see the appropriate chapters in this guide.

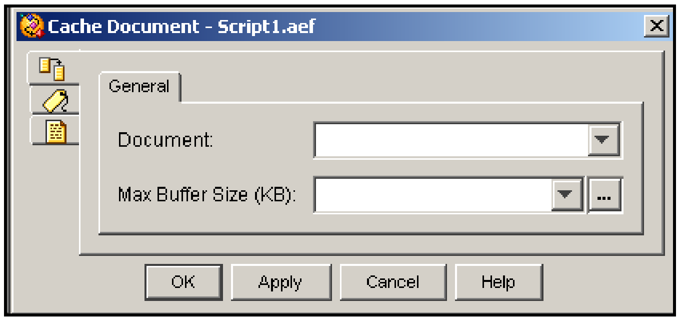
- Transformation steps—Used to process an input document and store the results in an output document. The output document can use the same variable as the input document. Three Document steps are Transformation steps:
  - Create XML Document
  - Cache Document
  - Keyword Transform Document
- Output steps—Used to direct a document to an output destination, such as a URL or file system. Two Document steps are Output steps:
  - Write Document
  - Get XML Document Data

Note Other steps can act as output steps for documents. You can use any Media or Prompt step to accept a document as input representing a prompt to be played back. You can use the Set User Info step (a User step) to upload a document as the spoken name for the user. You can use the Attach To eMail step (an eMail step) to attach a document to an e-mail message. You can use the DB Write step (a Database step) to upload a document to a database. You can use the Send Response step (an Http Contact step) to send a document as a response to an HTTP request. You can use the Set Session Info step (a Session step) to store a document in the session context.

Cache Document Step

Use the Cache Document step to perform an input/output (I/O) operation (such as reading a file or making an HTTP request) and cache the resulting document in the memory buffer.

Figure 2-138 Cache Document Customizer Window



Note Because the Cache Document step can use a great deal of memory, you should use it with caution.

The I/O operation is specified by the document defined in a preceding step (such as Create File Document or Create URL Document) or by a document expression that contains a hard-coded document. When the Create File Document or Create URL Document step executes, it creates the document variable but does not send the URL request or access the file system.

The actual I/O operation occurs when another step (such as Send Response or Write Document) references the document. The Cache Document step allows you to complete the I/O operation before executing any subsequent steps.

Example 2-2 shows script pseudocode demonstrating how you can use the Cache Document step.

Example 2-2 Using the Cache Document Step

```
doc=Create URL Document(“http://mybank.money.com/debit?amount=500”)
doc=Cache Document(doc)
. . .
SendResponse(doc)
```

In Example 2-2, the Cache step makes the HTTP request to mybank.money.com. Without the Cache step, the I/O does not occur until the Send Response step executes.

Table 2-118 describes the properties of the Cache Document customizer window.

Table 2-118 Cache Document Properties

Property	Description
Document	The document you want to obtain and cache from the Document variable drop-down list.
Max Buffer Size (KB)	The maximum buffer size that you want to use. Assign this value carefully because the buffer size you assign may affect system performance. The Maximum Buffer Size can be set to 0 to request everything to be cached in memory. If the document is bigger than the specified size, it will be truncated.

To use the Cache Document customizer window to cache a document in the memory buffer, perform the following procedure:

Procedure

Step 1 From the Document drop-down menu, choose the variable that stores the document value.

Step 2 In the Maximum Buffer Size (KB) field, enter a value directly or take one of the following actions:

- From the Maximum Buffer Size (KB) drop-down menu, choose the variable that stores the maximum buffer size value.
- Click the **Expression Editor** (**...**) button and enter an expression that specifies the maximum buffer size value.

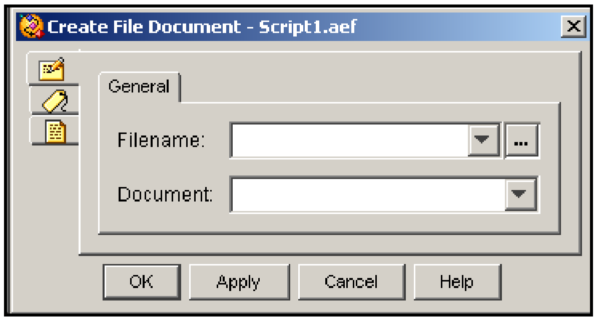
Step 3 Click **OK**.

The Cache Document customizer window closes, and the maximum buffer size variable appears next to the Cache Document step icon in the Design pane of the Cisco Unified CCX Editor.

Create File Document Step

Use the Create File Document step to define a document variable by referencing the file name you supply.

Figure 2-139 Create File Document Customizer Window



The Create File Document step does not read the file. The script reads the file when the document variable is used by another step, such as the Write Document step or the Create XML Document step.

Note The Create File Document step (a Document step) is exactly equivalent in functionality to using the FILE[filename] expression form for a Document.

Table 2-119 describes the properties of the Create File Document customizer window.

Table 2-119 Create File Document Properties

Property	Description
Filename	Path and file name for the document
Document	Document variable that represents the specified document

To use the Create File Document customizer window to define a document variable, perform the following procedure.

Procedure

Step 1 In the Filename field, enter a value directly or take one of the following actions:

- From the Filename drop-down menu, choose the variable that stores the filename value.
- Click the **Expression Editor** (**...**) button and enter an expression that specifies the filename value.

Step 2 In the Document text field, enter the name of the document variable that stores the document value.

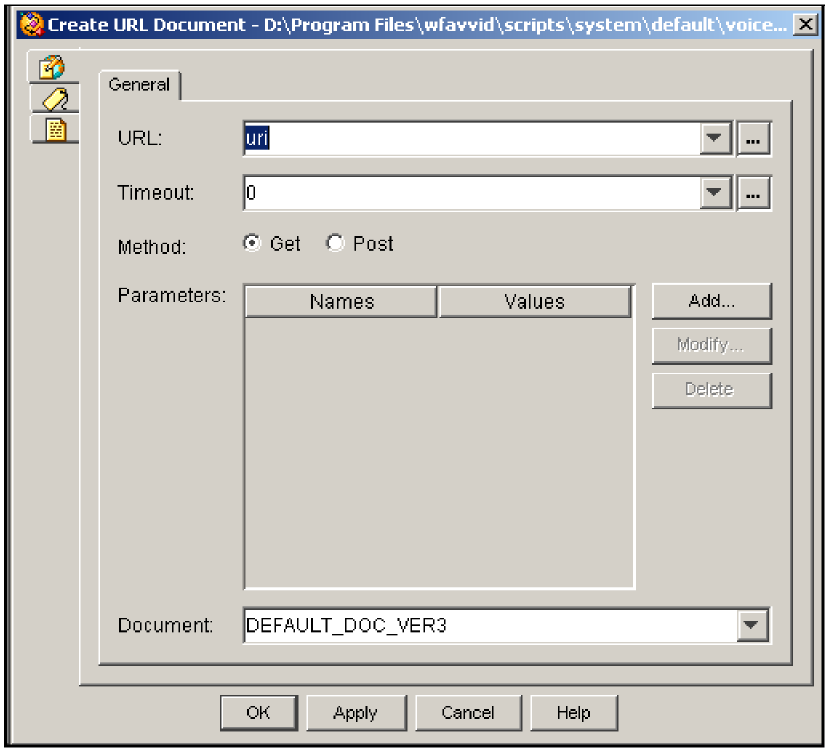
Step 3 Click **OK**.

The Create File Document customizer window closes.

Create URL Document Step

The Create URL Document step defines a document variable by referencing a URL you supply.

Figure 2-140 Create URL Document Customizer Window


Create URL Document - D:\Program Files\wfavvid\scripts\system\default\voice...

General

URL: uri

Timeout: 0

Method: [✓ Get] Post

Parameters:

Names	Values

Document: DEFAULT_DOC_VER3

The Create URL Document step does not issue the HTTP request. The actual request occurs when the document is used by another step, such as the Send Response step or the HTTP Forward step.

For a GET HTTP request, parameters are usually appended to a URL in an HTTP request to provide data required to execute the request. For example, the following HTTP request is made when the value of the **total** parameter is set to 500:

```
http://mybank.money.com/debit?amount=500
```

For POST, the parameters are passed in the body of the HTTP request as if entered in an HTML form.

Table 2-120 describes the properties of the Create URL Document customizer window.

Table 2-120 Create URL Document Properties

Property	Description
URL	URL for the document.
Timeout	Variable indicating the amount of time in milliseconds that the system waits before time out.
Method	Method to use if the URL represents an HTTP request. • The GET method appends parameters to the URL. (This step is equivalent to using the document expression form: URL[url?name=value,name=value].) • The POST method includes the parameters as if they were entered in an HTML form.

Table 2-120 Create URL Document Properties

Property	Description
Parameters (Names and Values)	Name and value pairs that form a parameter string to send to the web server. The URL-encoded parameter string allows special characters.
Document	Variable in which the resulting document object is stored.

To use the Create URL Document customizer window to define a document variable, do the following procedure:

Procedure

Step 1 In the URL field, enter a value directly or take one of the following actions:

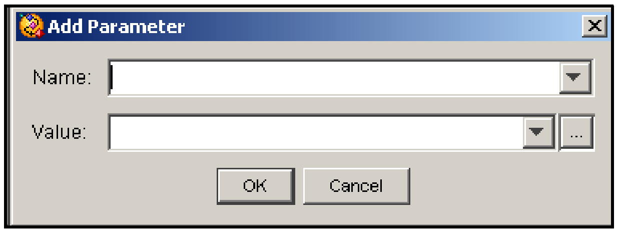
- From the URL drop-down menu, choose the variable that stores the URL value.
- Click the **Expression Editor** (**...**) button and enter a relative or absolute URL.

Step 2 In the Timeout field, enter the time in milliseconds.

Step 3 In the Method field, click the **Get** radio button to append parameters to the URL, or click the **Post** radio button to include parameters as if they were entered in an HTML form.

Step 4 To map a parameter to a local variable, click **Add.**

The Parameter dialog box appears.



Step 5 In the Name text field, enter the name of the parameter you want to define.

Step 6 From the Value drop-down menu, choose the variable that stores the parameter value.

Step 7 Click **OK**.

The Parameter dialog box closes, and the names of the parameter and variable appear in the Name and Value columns of the Create URL Document window.

Step 8 Repeat Steps 4 to 7 as needed to add parameters and variables as desired.

(If you want to modify an existing parameter, click **Modify**. The Parameter dialog box appears again. Follow the same procedure you used to add parameters.)

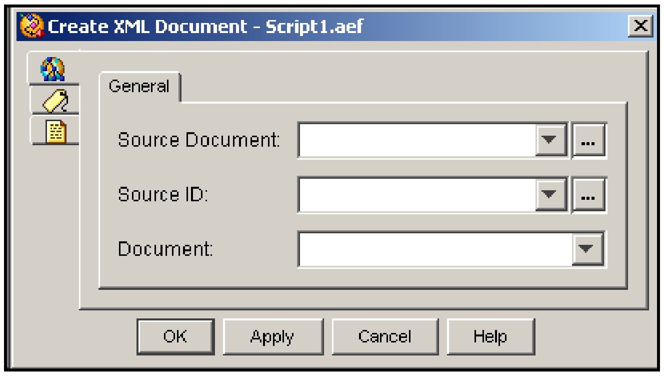
Step 9 Click **OK**.

The Create URL Document customizer window closes.

Create XML Document Step

Use the Create XML Document step to create a logical document that maps a document to another document variable (where the document has already been pre-parsed as an XML document and is ready to be accessed by the Get XML Document Data step).

Figure 2-141 Create XML Document Customizer Window



Use this step before the Get XML Document Data step to obtain data from a document formatted using the Extensible Markup Language (XML).

Table 2-121 describes the properties of the Create XML Document customizer window.

Table 2-121 Create XML Document Properties

Property	Description
Source Document	Variable containing the source document from which you want to create an XML document.
Source ID	(Optional) Source identifier (URL) to be used when parsing the document. If the document is a URL document, this field is not required; otherwise, you can supply it if known.
Document	Document that contains the resulting XML document. You can use the same variable that you used for the Source Document field if you do not need to use the original document again with subsequent steps. The variable assigned to the Result Document is used as the input to the Get XML Document Data step.

Note Note: When creating a non-English XML file, you must accurately set the character encoding. XML uses Unicode (UTF-8) by default, but you can use other encoding methods. For example, many Western European language text editors use ISO-8859-1 (latin-1) encoding by default. In this case, you must set the encoding attribute of the XML declaration, as in the example below:
<?xml version="1.0" encoding="ISO-8859-1"?>"

To use the Create XML Document customizer window to map a document to another document variable, do the following procedure:

Procedure

Step 1 From the Source Document drop-down menu, choose the variable that stores the source document.

Step 2 In the Source ID (Optional) field, enter a value directly or take one of the following actions:

- From the Source ID (Optional) drop-down menu, choose the variable that stores the ID value of the source document.
- Click the **Expression Editor** (**...**) button and enter an expression that specifies the ID value of the source document.

Step 3 From the Result Document drop-down menu, choose the variable that stores the resulting XML document.

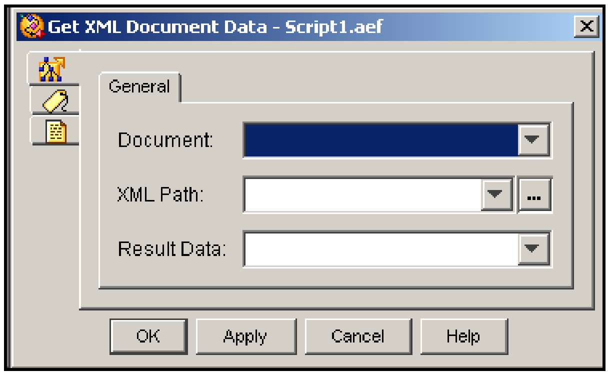
Step 4 Click **OK**.

The Create XML Document customizer window closes, and the name of the source document, the Source ID variable, and the result document appear next to the Create XML Document step icon in the Design pane of the Cisco Unified CCX Editor.

Get XML Document Data Step

Use the Get XML Document Data step after the Create XML Document step to obtain data from a document formatted with XML.

Figure 2-142 Get XML Document Data Customizer Window



Note You can use the Get XML Document Data step only on a document that was returned by the Create XML Document step. If you attempt to use it on any other type of document, errors will occur.

Example 2-3 shows a few lines from a typical XML file, created with a text editor, which formats dynamic stock price data.

Example 2-3 A Sample XML File

```
<?xml version="1.0" standalone="yes"?>
<STOCKLIST>
  <STOCK symbol="MSFT" error="0">
```

```
    <HIGH>58.0625</HIGH>
    <PCT_CHANGE>0.67114094</PCT_CHANGE>
    <LOW>55.1875</LOW>
    <LAST>56.25</LAST>
    <CHANGE>0.375</CHANGE>
    <VOLUME>31,973,600</VOLUME>
    <REC_STATUS>0</REC_STATUS>
    <DATE>02/21/2001</DATE>
    <TIME>15:52</TIME>
  </STOCK>
</STOCKLIST>
```

To extract data from an XML file, you must identify the exact XML path, as in Example 2-4:

Example 2-4 Typical XML Path

```
/descendant::STOCKLIST
/child::STOCK[attribute::symbol='CSCO']
/child::LAST
```

The XML path is composed of location steps separated by a forward slash (/). In the example above, the XML path has three location steps. Each location step is composed of three parts:

- Axis—Relation to the context (for example, descendant or child).
- Node test—Field name (for example, STOCK).
- Predicates (optional)—Properties and values within a field (for example, [attribute::symbol='CSCO']).

Note You must execute a separate Get XML Document Data step for each field in the XML file that you want to use.

Table 2-122 describes the properties of the Get XML Document Data customizer window.

Table 2-122 GET XML Document Data Properties

Property	Description
Document	Document variable created as the Result Document in a preceding Create XML Document step.
XML Path	XML path that defines a specific field or value in an XML document.
Result Data	Document variable that contains the XML data.

To use the Get XML Document Data customizer window to specify a document variable that will contain XML data, do the following procedure:

Procedure

Step 1 From the Document drop-down menu, choose the variable that stores the source document.

Step 2 In the XML Path field, enter a value directly or take one of the following actions:

- From the XML Path drop-down menu, choose the variable that stores the XML Path value of the source document.

- Click the **Expression Editor** (**...**) button and enter an expression that specifies the XML Path value of the source document.

Step 3 From the Result Data drop-down menu, choose the variable that stores the resulting XML document.

Note Make sure that the Document variable you choose here matches the one you chose for the Result Document in a preceding Create XML Document step.

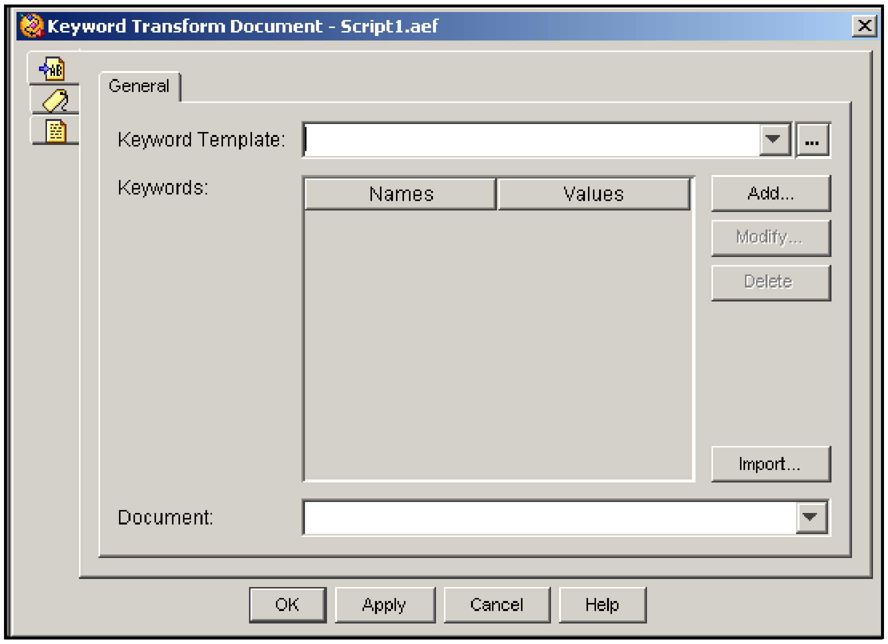
Step 4 Click **OK**.

The Get XML Document Data customizer window closes, and the names of the Result Data, Document, and XML Path variables appear next to the Get XML Document Data step icon in the Design pane of the Cisco Unified CCX Editor.

Keyword Transform Document Step

Use the Keyword Transform Document step to replace keywords in a predefined template with values contained in local variables, in order to display dynamic data to users.

Figure 2-143 Keyword Transform Document Customizer Window



You map these keywords to variables. When the script runs, the current values of the local variables replace the keywords.

You can load keyword definitions from a source document, or you can add each keyword manually.

Table 2-123 describes the properties of the Keyword Transform Document customizer window.

Table 2-123 Keyword Transform Document Properties

Property	Description
Keyword Template	An Expression that identifies a source template document.
Keywords (Names and Values)	Names of keywords with the value mapped to each in the source document.
Document	Variable that stores the resulting document.

To use the Keyword Transform Document customizer window to map keywords to local variables, do the following procedure:

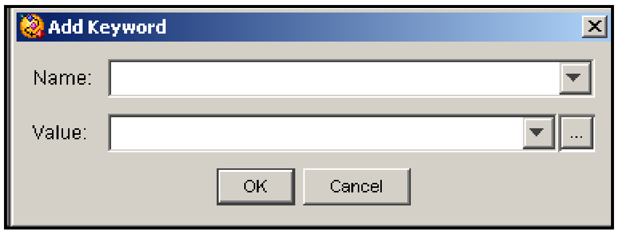
Procedure

Step 1 From the Keyword Template drop-down menu, choose the variable that stores the source template document or enter the expression.

Step 2 Take one of the following actions:

- To define a keyword manually, click **Add.**

  The Add Keyword dialog box appears.

  – In the Name field, enter an alphanumeric string to uniquely identify the purpose of the keyword.

    For example, if you are using a keyword to display the results of a database query, you might define a keyword named “QueryResult”.

  – In the Value drop-down menu, choose the variable you want to map to the keyword.

  – Click **OK**.

    The Add Keyword dialog box closes, and the keyword template name and value appear in their respective columns in the display list of the Keyword Transform Document customizer window.

  – Repeat this procedure as needed to map all keywords to local variables.

- To load keyword definitions from a file, click **Import**.

  The Importing Keyword Template dialog box appears.

– In the Importing Keyword Template dialog box, browse to the file that contains the keywords you want to import, and then click **Open**.

  Keywords must be flagged by percentage signs in the source document (%keyword%).

  The Importing Keyword Template dialog box closes, and any words in the document that are enclosed by percentage signs are displayed in the Keyword column of the Keyword Transform Document customizer window.

(If you want to map imported keywords to local variables, select the keyword in the list box, and click **Modify**. The Keyword Mapping dialog box appears again. Follow the same procedure you used to add variables.)

Step 3 From the Document drop-down menu, choose the variable that stores the resulting document.

Step 4 Click **OK**.

The Keyword Transform Document customizer window closes, and the names of the Document and Source Document appear next to the Keyword Transform Document step icon in the Design pane of the Cisco Unified CCX Editor.

Upload Document Step

Use the Upload Document step to add a document to the repository database. Uploading a document makes it accessible to all the Cisco Unified CCX servers in the cluster and allows it to be backed up with all other repository data.

Note The Upload Document step will not work unless the user specified in the step is an authenticated user. Use the Authenticate User step before the Upload Document step to authenticate the user.

Warning **Although the repository database is meant to hold documents used by scripts, do not use the Upload Document functionality to store all types of documents. There is no provision for a delete document operation from a script. Also, the system will not behave properly if this step is over-utilized. This means that this step has not been designed to be extensively used by all calls coming into the system. Rather, the step is meant for updating documents in the repository database from time to time.**

Figure 2-144 Upload Document Customizer Window

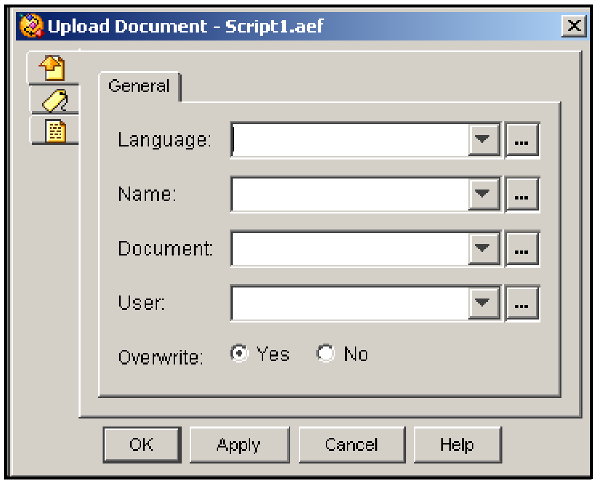


Table 2-124 describes the properties of the Upload Document customizer window.

Table 2-124 Upload Document Properties

Property	Description
Language	Variable or expression indicating the language you want to upload to.
Name	Variable or expression indicating the name of the document you want to upload. (This is relative to the language folder in the repository.)
Document	The variable or expression to be uploaded to the repository.
User	The authenticated user generating and uploading the document.
Overwrite (radio buttons)	Select one of the following: • **Yes**—Select this option to overwrite the current data. • **No**—Select this option to retain the current data.

Write Document Step

Use the Write Document step to write the document to disk on the Cisco Unified CCX server.

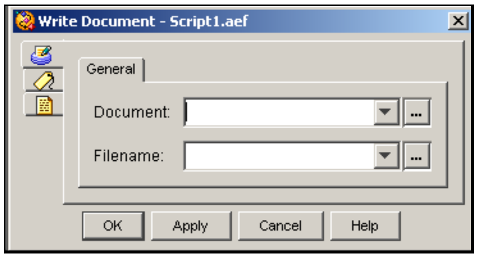
Note You can write to the Unified CCX customer folder only, which can be retrieved using the following syntax:

```
System.getProperty("uccx.customer.dir")
```

For more information on how to retrieve information from the customer folder using CLI commands, refer to the *Command Line Interface Reference Guide for Cisco Unified CCX and Cisco Unified IP IVR* available here: http://www.cisco.com/en/US/products/sw/custcosw/ps1846/products_installation_and_configuration_guides_list.html

Warning **Note that this customer folder is not backed up by Unified CCX. Also, the purging capability is not supported for this folder.**

Figure 2-145 Write Document Customizer Window



For example, you can save a recorded announcement to disk that replaces a “Message of Day” announcement.

Table 2-125 describes the properties of the Write Document customizer window.

Table 2-125 Write Document Properties

Property	Description
Document	Variable that stores the resulting document
Filename	Name of the file that contains the information to be saved

To use the Write Document customizer window to specify the filename and Document variable, do the following procedure:

Procedure

Step 1 From the Document drop-down menu, choose the variable that stores the resulting document.

Step 2 In the Filename field, enter a value directly or take one of the following actions:

- From the Filename drop-down menu, choose the variable that stores the filename value.
- Click the **Expression Editor** (**...**) button and enter an expression that specifies the filename value.

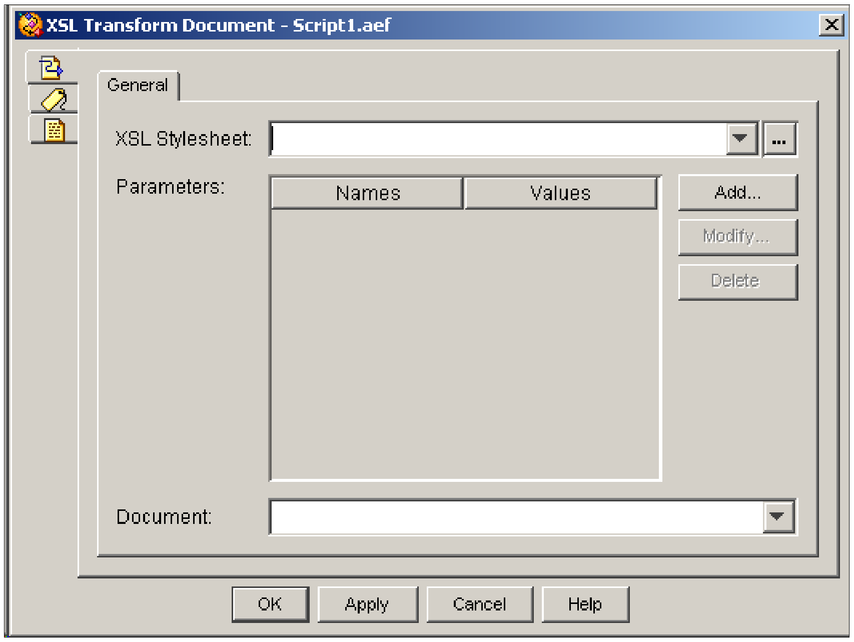
Step 3 Click **OK**.

The Write Document customizer window closes, and the name of the Filename variable appears next to the Write Document step icon in the Design pane of the Cisco Unified CCX Editor.

XSL Transform Document Step

Use the XSL Transform Document step to apply eXtensible Style sheet Language (XSL) transformation to produce an output document.

Figure 2-146 XSL Transform Document Customizer Window



XSL transformation is performed by an XSL processor provided by the Cisco Unified CCX Engine. The XSL processor performs the transformation using an XSL Style sheet, which contains formatting and other processing instructions.

When the XSL Transform Document step executes, it converts the specified variables into an internal XML document. This internal XML document is passed to the XSL processor in conjunction with an XSL Style sheet to form the resulting document.

Example 2-5 shows the structure of an internal XML representation.

Example 2-5 Structure of Internal XML Representation

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ExecutionState[
     <!ELEMENT ExecutionState (scriptName, OutputContext)>
     <!ELEMENT scriptName (#PCDATA)>
     <!ELEMENT OutputContext ANY>
]>
```

Table 2-126 describes each element in the representation.

Table 2-126 Elements of XML Internal Representation

Element	Function
ExecutionState Element	Result after the script is executed.
ScriptName Element	Name of the script being executed.
OutputContext	Variables added to the Output Parameters box within the XSL Transform Document customizer.

Table 2-127 describes the properties of the XSL Transform Document customizer window.

Table 2-127 XSL Transform Document Properties

Property	Description
XSL Style sheet	Style sheet used to form the result document.
Parameters (Names and Values)	Variables with their values that are passed to the XSL processor.
Document	Variable that stores the document resulting from this step.
Add / Modify (buttons)	Use these buttons to access the Add or Modify Parameter dialog box to add or change parameters. When done, click **OK**. **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove a parameter, highlight a value in the list and click **Delete**.

To use the XSL Transform Document customizer window to apply XSL transformation to produce an output document, do the following procedure:

Procedure

Step 1 In the XSL Style sheet text field, enter the name of the XSL style sheet.

Step 2 To add a parameter to the Parameters list to pass to the XSL processor, click **Add**. Then add the parameter in the Add Keyword dialog box. To modify a parameter, select that parameter and click **Modify**. Modify the parameter in the Modify Keyword dialog box.

Step 3 From the Result Document drop-down menu, choose the variable that stores the resulting document.

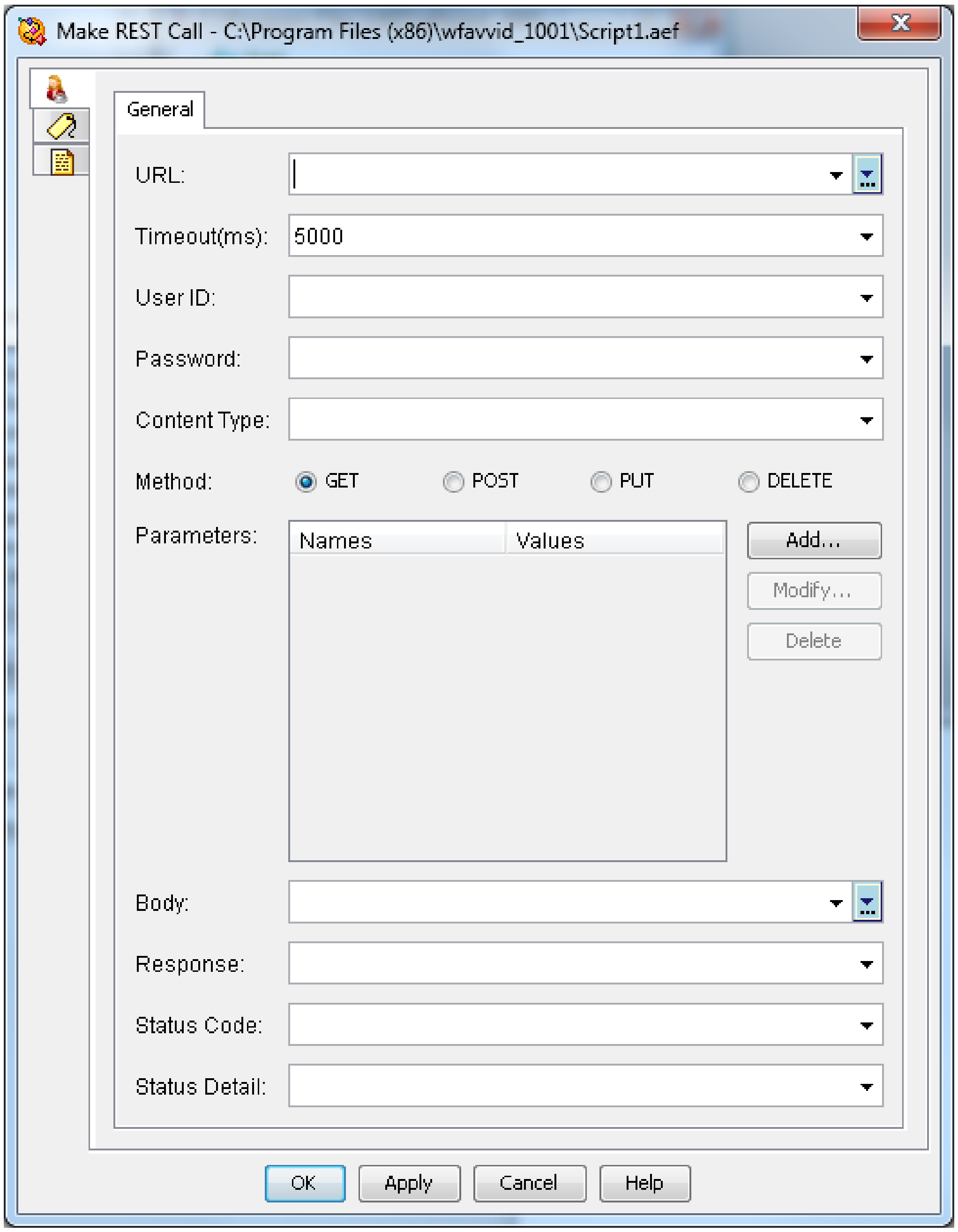
Step 4 Click **OK**.

The XSL Transform Document customizer window closes, and the name of the XSL style sheet and the result document appear next to the XSL Transform Document step icon in the Design pane of the Cisco Unified CCX Editor.

Make REST Call

Use the Make REST Call step to contact any of the REST Services exposed within Unified CCX or outside Unified CCX such as Cisco Unified Intelligence Center, Cisco MediaSense, or any REST APIs.

Figure 2-147 Make REST Call


Make REST Call - C:\Program Files (x86)\wfavvid_1001\Script1.aef
General
Names
Values
Add...
Modify...
Delete
OK
Apply
Cancel
Help

URL:
Timeout(ms): 5000
User ID:
Password:
Content Type:
Method: [✓ GET] POST PUT DELETE
Parameters:
Body:
Response:
Status Code:
Status Detail:

Table 2-128 Make REST Call Properties

Property	Description
URL	Variable or an expression resolving to the REST URL. Open the Expression Editor by clicking and then form the REST URL using the required variables (parameters).
Timeout (ms)	Use this timeout value for both connection timeout and read timeout. If 5000 milliseconds is specified as the timeout value, the step times out when a connection is not established within five seconds. It also times out when the connection is established but no response is obtained within five seconds. The default value is 5000 milliseconds.
User ID	Enter the username for the REST Service authentication or declare the username as variable and associate that to User ID from the drop-down list.
Password	The password is shown in plain text. Use a variable for the password and expose it as a script parameter so that changing the password is easy if required.
Content Type	REST step accepts any valid content type. The commonly used content types are Application/XML and Application/JSON.
Method	• The GET method lists all the instance of the object or gets details of the instances specified by instanceId. • The POST method creates a new instance of the object. • The PUT method modifies the instance specified by instanceId. • The DELETE method deletes the instance specified by the instanceId.
Parameters	Enter query parameters as key value pairs. These key value pairs are appended to the URL.
Body	Declare the body string as variable and associate that to Body from the drop-down list or directly hard code the string in the Body. **Note** It is applicable only to PUT and POST method.
Response	The REST response is converted to a string and is assigned to the associated string variable.
Status Code	Status code returned by REST API.
Status Detail	Status details returned by the REST API.

Do not use hard-coded string for the output variables. For output variables such as Response, Status Code, and Status Detail, declare the variables first and then associate the declared variables to the output variables.

For example, for Response, declare a variable called Response and then choose the variable Response from Response drop-down list.

If you access the external secure REST APIs, upload certificates to platform Tomcat's keystore. The engine Tomcat picks them up from platform Tomcat.

For information about uploading certificates, see *Cisco Unified Communications Operating System Administration Guide for Cisco Unified CCX and Cisco Unified IP IVR, Release 10.0(1)* at http://www.cisco.com/en/US/products/sw/custcosw/ps1846/products_installation_and_configuration_guides_list.html

To specify the query parameters for the URL, there are two ways:

- Construct the URL with query parameters using the Unified CCX Express Editor.
- Pass the query parameters as key value pairs in the Parameters table. These key value pairs are appended to the URL.

Database Steps

The steps in the Database palette of the Cisco Unified CCX Editor provide script designers with a way to read and write data and/or documents to database tables.

Note Before you use the Database steps, refer to the *Cisco Unified Contact Center Express Administration Guide* for information on how to set up a Data Source Name (DSN) and configure the Database subsystem.

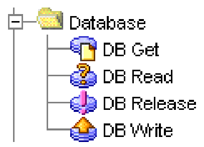
This section contains the following topics:

- DB Get Step
- DB Read Step
- DB Release Step
- DB Write Step

Figure 2-148 shows the steps in the Database palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Note The Database palette steps are runnable in your Cisco Unified CCX scripts only if you have purchased the Cisco Unified IP IVR or Cisco Unified CCX Premium license options.

Figure 2-148 Database Palette Steps



DB Get Step

Use the DB Get step to assign to specific variables the results of the Structured Query Language (SQL) query that you define in the DB Read step.

Note Before you can use a DB Get step, you must use a DB Read step to define the SQL statements and identify the target database.

Each time the script executes the DB Get step, the script retrieves one row of the results returned by the DB Read step and places them in the variables you assign. To move to the next row in a table, you must execute the DB Get step again.

The DB Get step has three output branches:

- Successful—The DB Get step successfully retrieved data.
- No Data—The DB Get step did not retrieve data.
- SQL Error—There was an error in the SQL command.

Note You cannot create Join queries that retrieve columns with the same name from different tables. (For example, you cannot create "select a.x, b.x from a,b" and "select a.x as y from a".) Similarly, you cannot make aliases of column names. The SQL statements SELECT count(*) and SELECT min(*) are not supported.

The DB Get customizer window contains two tabs:

- General (DB Get Step)
- Field Selection (DB Get Step)

The following sections describe these two tabs.

General (DB Get Step)

Use the General tab of the DB Get step to specify the DB Resource Name and Data Source Name.

Figure 2-149 DB Get Customizer Window—General Tab

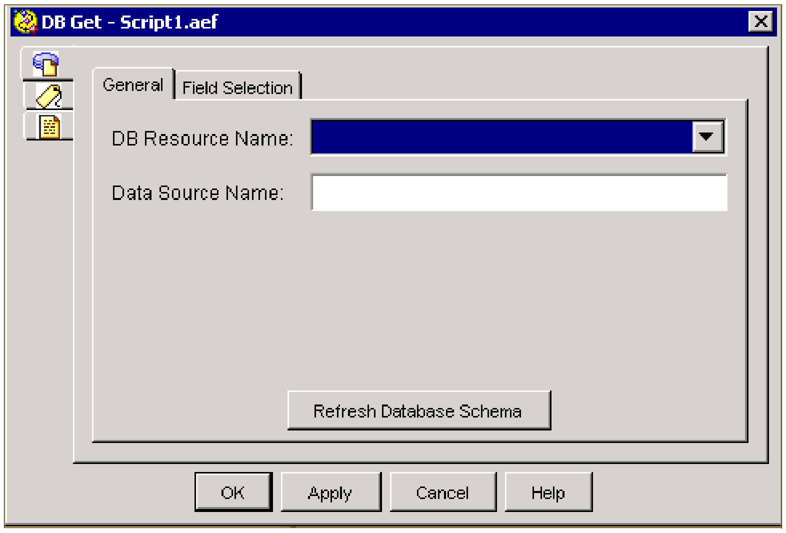


Table 2-129 describes the properties of the General tab of the DB Get customizer window.

Table 2-129 DB Get Properties—General Tab

Properties / Buttons	Description
DB Resource Name	Variable indicating the name of query defined by a DB Read step. You can choose a specific query when your script has multiple DB Read steps open at the same time.
Data Source Name	Name of the database selected on the Cisco Unified CCX Administration Database web page. For information about defining DSNs, see the *Cisco Unified Contact Center Express Administration Guide*.
Refresh Database Schema (button)	Click this button to update the Cisco Unified CCX Editor with the fields defined in the selected database.

Field Selection (DB Get Step)

Use the Field Selection tab of the DB Get customizer window to identify the fields within the selected query and to choose the variables in which you want to store the value for each field.

Figure 2-150 DB Get Customizer Window—Field Selection Tab

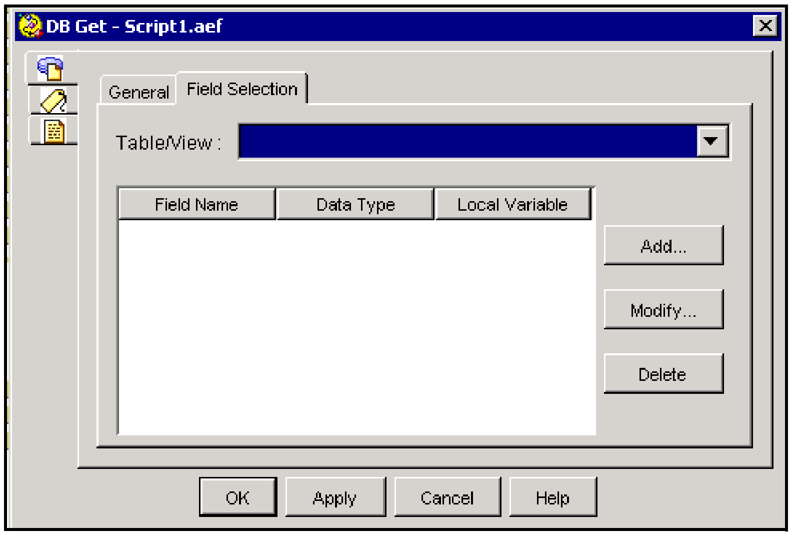


Table 2-130 describes the properties of the Field Selection tab of the DB Get customizer.

Table 2-130 DB Get Properties—Field Selection Tab

Properties / Buttons	Description
Table/View	Variable indicating the name of the table from the database selected on the General tab.
Field Name/Data Type/Local Variable	Field selection information: • **Field Name—**Name of the field in the selected database. • **Data Type—**(Display only) Data type of the variable. • **Local Variabke—**variable that stores the values of the associated field.
Add / Modify (buttons)	Use these buttons to access the Add Database Field dialog box. Use the dialog to add or modify settings and click **OK** when finished **Note** An entry can be directly edited in the table list by double clicking on it. To sort the entries, click on the column header.
Delete (button)	To remove database field information, highlight a value in the list and click **Delete**.

DB Read Step

Use the DB Read step to choose a database and enter SQL statements required to obtain the data you need for your script.

You can use a DB Get step after a DB Read step to assign the results of your query to specific variables.

Note The DB Read step is not interruptible by external events.

The DB Read step has three output branches:

- Successful—The DB Read step successfully executes.
- Connection Not Available—The DB Read step can not make a connection to the specified database.
- SQL Error—The DB Read step encounters a SQL command error.

The DB Read customizer window contains three tabs:

- General tab (DB Read step)
- Field Selection tab (DB Read step)

The following sections describe these tabs.

General tab (DB Read step)

Use the General tab of the DB Read customizer window to specify a DB Resource Name and Data Source Name.

Figure 2-151 DB Read Customizer Window—General Tab

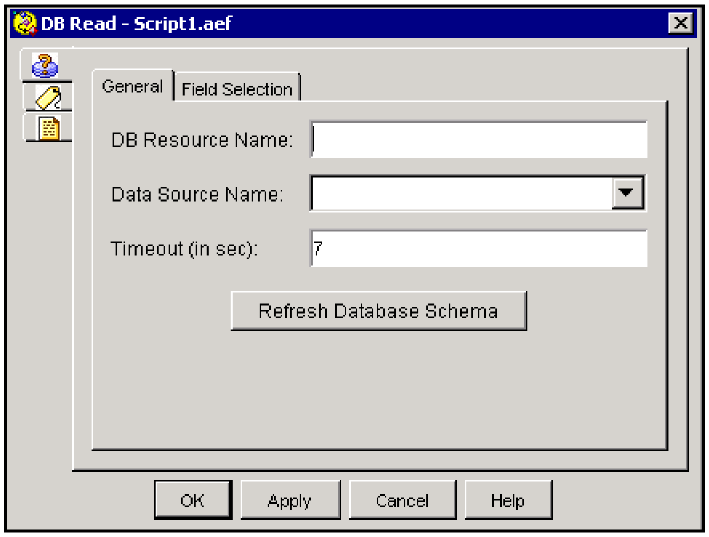


Table 2-131 describes the properties of the General tab of the DB Read customizer window.

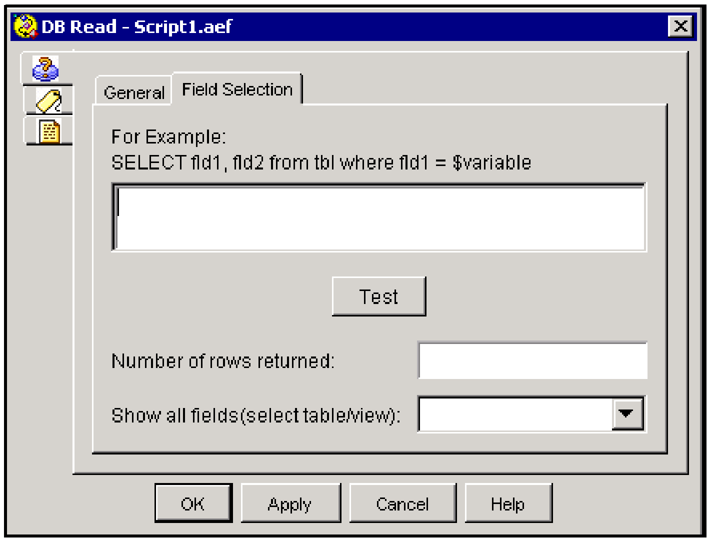
Table 2-131 DB Read Properties—General Tab

Properties / Buttons	Description
DB Resource Name	Name identifying this database query. **Note** The same DB Resource Name is used in the accompanying DB Get and DB Release steps.
Data Source Name	Variable indicating database you want to access. This list contains the DSNs you identified in the Cisco Unified CCX Administration Database web page. **Note** For information about defining DSNs, refer to the *Cisco Unified Contact Center Express Administration Guide*.
Timeout (in sec)	Interval that prevents your script from waiting indefinitely when the database is unavailable. If the value for the timeout interval is 0, an indefinite wait occurs. **Note** An indefinite wait may block the script from responding to events such as a remote disconnection or an agent becoming available.
Refresh Database Schema (button)	If the Data Source Name drop-down list is empty, click this button to download the schema from the Cisco Unified CCX engine.

Field Selection tab (DB Read step)

Use the Field Selection tab of the DB Read customizer window to define the SQL SELECT statement required to retrieve the data that you need from the specified database.

Figure 2-152 DB Read Customizer Window—Field Selection Tab



For Example:
SELECT fld1, fld2 from tbl where fld1 = $variable

Number of rows returned:

Show all fields(select table/view):

Table 2-132 describes the properties of the Field Selection tab of the DB Read customizer window.

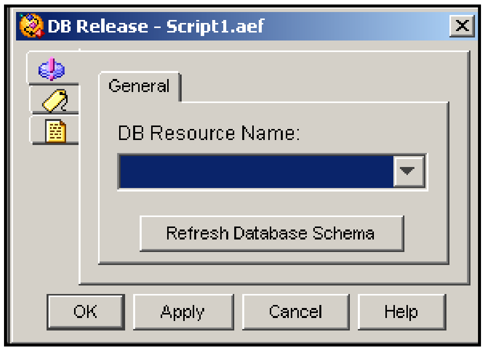
Table 2-132 DB Read Properties—SQL Tab

Properties / Buttons	Description
Enter SQL command to be executed	SQL command that you want to be executed, using standard SQL syntax. **Note** The SQL statements SELECT count(*) and SELECT min(*) are not supported.
Test (button)	Click this button to test your query and connectivity to the chosen database. **Note** To test the SQL statement, you must have a connection to an active Cisco Unified CCX Engine.
Number of rows returned	Display only. Populates when the **Test** button is clicked. **Note** Use this value to determine whether or not your query is working correctly.
Show All Fields (Select Table/View)	Displays fields defined in a particular table in this database.

DB Release Step

Use the DB Release step after a DB Get or DB Write step to close a SQL query and release the allocated resources.

Figure 2-153 DB Release Customizer Window



A released DB connection is returned to the connection pool and data is no longer associated with this connection.

Table 2-133 describes the properties of the DB Release customizer window.

Table 2-133 DB Release Properties

Properties / Buttons	Description
DB Resource Name	Name of the DB Resource for which you want to release resources.
Refresh Database Schema (button)	Click this button to update the DB Resource Name drop-down list.

DB Write Step

Use the DB Write step to select a database and enter SQL UPDATE, SQL DELETE, or SQL INSERT statements.

You can use this step to update an enterprise database. When you are finished with a database write operation, use the DB Release step to free the connection to the database server.

The DB Write step has three output branches:

- Successful—The DB Write step successfully enters the specified SQL statements.
- Connection Not Available—The DB Write step can not find a connection to the specified database.
- SQL Error—The DB Write step encounters a SQL command error.

The DB Write customizer window contains four tabs:

- General tab (DB Write step)
- SQL tab (DB Write step)
- Test tab (DB Write step)

The following sections describe these tabs.

General tab (DB Write step)

Use the General tab of the DB Write customizer window to identify the DB Resource Name variable and the database that you want to update.

Figure 2-154 DB Write Customizer Window—General Tab

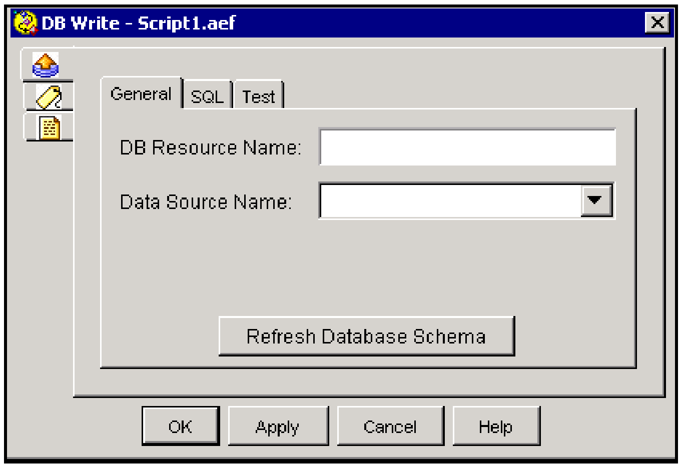


Table 2-134 describes the properties of the General tab of the DB Write customizer window.

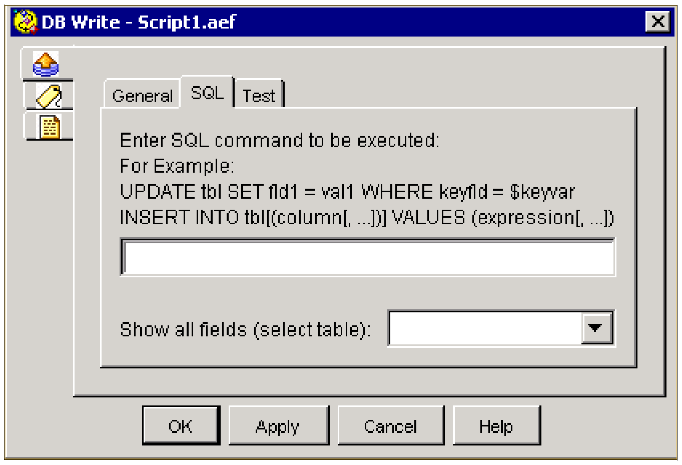
Table 2-134 DB Write Properties—General Tab

Properties / Buttons	Description
DB Resource Name	Name identifying this database query.
Data Source Name	Variable indicating the database you want to update when you are defining a new query.
Refresh Database Schema (button)	If the Data Source Name drop-down menu is empty, click this button that allows you to download the schema from the Cisco Unified CCX Engine.

SQL tab (DB Write step)

Use the SQL tab of the DB Write customizer window to define SQL UPDATE, SQL DELETE, or SQL INSERT statements that will update the chosen database.

Figure 2-155 DB Write Customizer Window—SQL Tab



Enter SQL command to be executed:
For Example:
UPDATE tbl SET fld1 = val1 WHERE keyfld = $keyvar
INSERT INTO tbl[(column[, ...])] VALUES (expression[, ...])

Show all fields (select table):

Table 2-135 describes the properties of the SQL tab of the DB Write customizer window.

Table 2-135 DB Write Properties—SQL Tab

Properties / Buttons	Description
Enter SQL command to be executed	SQL command that you want to be executed. **Note** Enter the SQL UPDATE, SQL DELETE, or SQL INSERT statements, using standard SQL syntax.
Show All Fields (Select Table)	Use this drop-down list to select the table from the database and display all fields.

Test tab (DB Write step)

Use the Test tab of the DB Write customizer window to test your query and connectivity to the chosen database.

Figure 2-156 DB Write Customizer Window—Test Tab

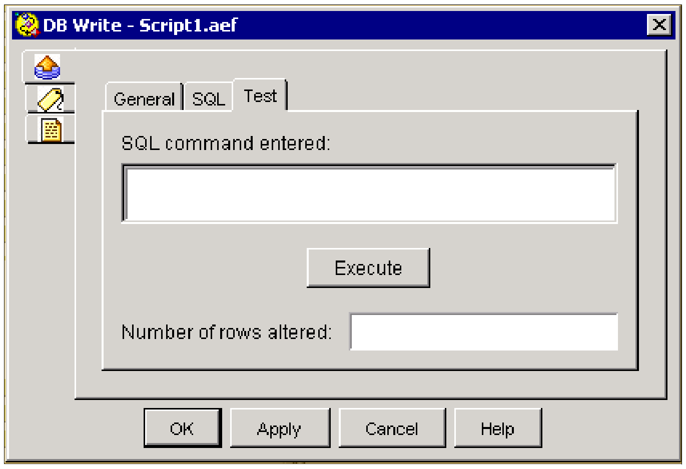


Table 2-136 describes the properties of the Test tab of the DB Write customizer window.

Table 2-136 DB Write Properties—Test Tab

Properties / Buttons	Description
SQL command entered	SQL command you want to test.
Execute (button)	Click this button to perform the test.
Number of rows altered	(Display only) Number of rows altered by the test. **Note** Observe the figure in the Number of Rows Altered text field. This number indicates whether or not your SQL statements are producing the expected results. Clicking the Execute button in this window does not update the database; this action does nothing but validate the SQL statement. You must be connected to an active Cisco Unified CCX server to perform this operation.

ACD Steps

The steps in the ACD palette of the Cisco Unified CCX Editor provide script designers with a way to queue calls and connect them to available resources.

These steps are only available in Cisco Unified CCX packages.

Note See the *Cisco Unified Contact Center Express Administration Guide* for configuration information about Cisco Unified CCX and Contact Service Queues (CSQ).

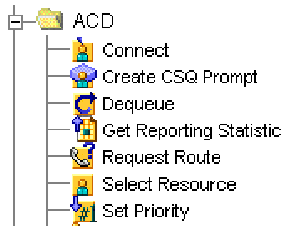
This section contains the following topics:

- Connect Step
- Create CSQ Prompt Step

- Dequeue Step
- Get Reporting Statistic Step
- Request Route Step
- Select Resource Step
- Set Priority Step
- ICM Step

Figure 2-157 shows the steps in the ACD palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

Figure 2-157 ACD Palette Steps



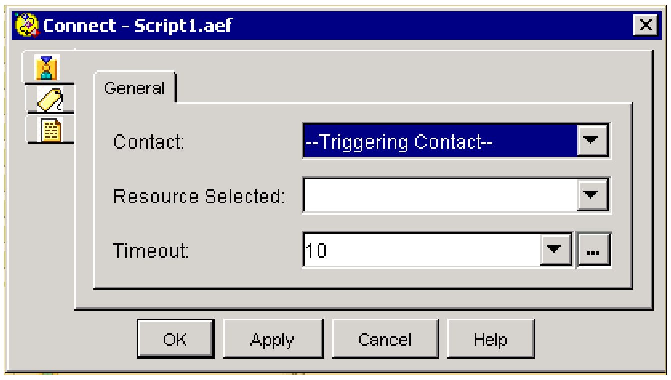
Note The functionality for the Set Priority and Create CSQ Prompt steps is specific to both the Cisco Unified CCX Enhanced and Premium products.

These steps do not appear if the product you have licensed is the Cisco Unified CCX Standard product. If you have not licensed any of the Cisco Unified CCX products (that is, Cisco UnifiedIP IVR licensed) this entire palette and its steps does not appear in your Cisco Unified CCX Editor.

Connect Step

Use the Connect step to connect a contact to the specified resource.

Figure 2-158 Connect Customizer Window



Contact: --Triggering Contact--

Resource Selected:

Timeout: 10

Note This step works only when the Connect option in the Select Resource step is set to **No**. A **No** setting in the Select Resource step means the resource is selected and reserved—but not *connected*—until the script executes a Connect step. (For more information, see the "Select Resource Step" section on page 2-213.

The Connect step has two output branches:

- **Connected**. The Connect step successfully connects the contact to the resource.
- **Failed**. The Connect step fails to connect the contact to the resource.

Table 2-137 describes the properties of the Connect customizer window.

Table 2-137 Connect Properties

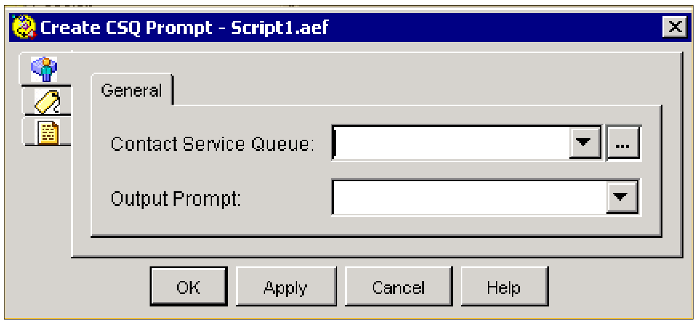
Properties / Buttons	Description
Contact	Variable containing the triggering contact information for this step.
Resource Selected	Variable identifying the resource assigned to handle the call.
Timeout	Variable or expression containing the length of time, in seconds, before the contact is retrieved back into the queue. This value must be lower than the Call Forward No Answer timeout in the Cisco Unified Communications Manager.

Create CSQ Prompt Step

Use the Create CSQ Prompt step to obtain the prompt that records the name of a CSQ. This prompt, in turn, can be passed to a Play Prompt step to play back the CSQ name.

Note The Create CSQ Prompt step is only available if you have licensed the Cisco Unified CCX *Enhanced* or *Premium* product package.

Figure 2-159 Create CSQ Prompt Customizer Window


Create CSQ Prompt - Script1.aef

Contact Service Queue:

Output Prompt:

Table 2-138 describes the properties of the Create CSQ Prompt customizer window.

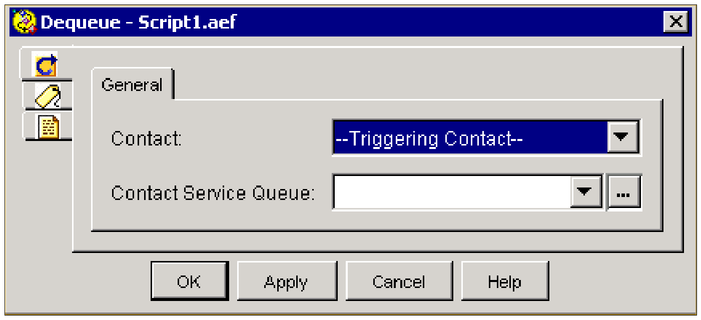
Table 2-138 Create CSQ Prompt Properties

Properties / Buttons	Description
Contact Service Queue	A unique identifier for the CSQ.
Output Prompt	The variable that stores the prompt object the records the name of a CSQ.

Dequeue Step

Use the Dequeue step to remove a call from one or more queues.

Figure 2-160 Dequeue Customizer Window


Dequeue - Script1.aef

Contact: --Triggering Contact--

Contact Service Queue:

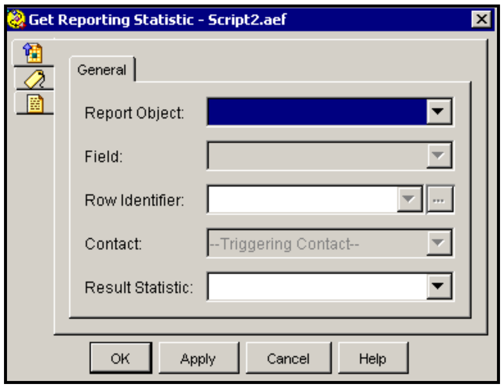
Table 2-139 describes the properties of the Dequeue customizer window.

Get Reporting Statistic Step

Use the Get Reporting Statistic step to access real-time information about agents, Contact Service Queues (CSQs), the overall Cisco Unified CCX system, and contacts.

Note The Get Reporting Statistic step is only available with Cisco Unified CCX packages.

Figure 2-161 Get Reporting Statistic Customizer Window


Get Reporting Statistic - Script2.aef

General

Report Object:
Field:
Row Identifier:
Contact: --Triggering Contact--
Result Statistic:

You can use this step to view the current state of the Cisco Unified CCX system or to report back to a caller such information as current wait duration, position in queue, and expected wait time.

The system bases the expected wait time calculation on the number of agents in reserved, talking, and work states for this CSQ, the call's position in the queue, and the average call duration for this CSQ. Average call duration of a CSQ is the average time agents spend in Reserved, Talking, and Work states while handling a call from this CSQ.

Table 2-140 describes the properties of the Get Reporting Statistic customizer window:

Table 2-140 Get Reporting Statistic Properties

Property	Description
Report Object	Type of report: • Outbound Campaign • Overall Cisco Unified CCX • Resource Cisco Unified CCX • CSQ Cisco Unified CCX
Field	Specific statistic to retrieve from the report. See the *Cisco Unified Contact Cenre Application Administration Guide* for how to run real-time reports. See the following tables for a list of available fields for each report: • Outbound Campaign Statistics report—Table 2-141 on page 2-204. • Overall Cisco Unified CCX Statistics report—Table 2-142 on page 2-206. • Resource Cisco Unified CCX Statistics report—Table 2-143 on page 2-209. • CSQ Cisco Unified CCX Statistics report—Table 2-144 on page 2-211. **Note** One of the options for this property that appears for the Overall Cisco Unified CCX and CSQ Cisco Unified CCX report options is **Work Resources**. "Work" state is not supported in Cisco Unified CCX Standard. If you have Cisco Unified CCX Standard and select either "overall Cisco Unified CCX" or "CSQ" for the Report Object, selecting "Work Resources" for the Field property returns "0" when retrieved by the workflow.
Row Identifier	**Note** Value used to identify the CSQ or Resource (only applicable to CSQ Cisco Unified CCX or Resource Cisco Unified CCX reports). Either choose the String variable that holds the value used to identify the CSQ or Resource, or click the **Expression Editor** (**...**).
Contact	Contact for which to get the statistic
Result Statistic	Script variable that will contain the resulting statistic value.

Note The statistic value is usually returned as an integer. The State option is an exception to this rule; it returns a string.

Table 2-141 describes the available fields for the Outbound Campaign Stats report.

Table 2-141 Available Fields, Outbound Campaign Stats

Row Heading	Description
Status	The current activation state of the campaign: • A "Running" status indicates an active campaign. • A "Stopped" status indicates an inactive campaign.
Preview	The total number of outbound calls currently previewed but that have not been accepted, rejected, or closed by the agents as part of this campaign.
Connected	The total number of Outbound calls currently connected to agents for this campaign.
Active	The total number of Outbound calls currently previewed by or connected to agents for this campaign. Active Calls = Previewed + Connected.
Timed-Out	The total number of outbound calls that timed out for this campaign. A call is considered timed out when it is presented to an agent and not accepted, rejected, or closed within the allocated time. These contacts will be dialed again. If a contact is timed out at multiple agents, this field is incremented each time the contact is timed out at each agent.
Offered	The total number of outbound calls offered for this campaign. A call is considered offered when it is presented to an agent as part of this campaign. A contact that is presented to an agent, skipped/rejected by that agent, and then presented to the same agent or to another agent is counted twice towards the number of calls offered. Offered = Accepted + Rejected + Closed + Timed-out.
Accepted	The total number of outbound calls accepted for this campaign. A call is considered accepted if an agent has clicked Accept when presented the call. A call that is presented to an agent, skipped/rejected by that agent, presented to another agent, and then accepted by that other agent is counted once towards the number of calls accepted.
Rejected	The number of outbound calls that were skipped or rejected by an agent as part of this campaign. This means that the agent selected 'Reject', 'Skip', or 'Cancel Reservation'. These contacts will be dialed again. If a contact is rejected by multiple agents, this field is incremented each time the contact is rejected. The number of Rejected is also incremented each time an agent drops the preview call while it is ringing at the customer's contact.
Closed	The number of Outbound contacts that were closed by agents as part of this campaign. This means that the agent selected 'Skip-Close' or 'Reject-close'. These contacts will not be dialed again.

Table 2-141 Available Fields, Outbound Campaign Stats (continued)

Row Heading	Description
Voice	The number of outbound calls that ended in successful customer contact for this campaign. This means that an agent accepted the call (by clicking Accept) AND selected a classification of 'Voice' or 'Do Not Call' for this contact.
Answering Machine	The number of outbound calls that connected to an answering machine for this campaign. This means that the agent accepted the call (by clicking Accept), got connected to the answering machine and selected the "Answering Machine" option from the contact Reclassfication dropdown. **Note** The agent can manually reclassify the contact as 'Answering Machine' while the customer contact is on the call or when the agent has gone into the Work state after the call.
Invalid Number	The number of outbound calls that were dialed to an invalid number for this campaign. This means that the agent accepted the call (by clicking Accept), got connected to the customer contact, and selected the "Invalid Number" option from the contact Reclassification dropdown. It also includes the number of outbound calls that failed at the network level. **Note** The agent can manually reclassify the contact as 'Invalid Number' while the customer contact is on the call or when the agent has gone into the Work state after the call.
Requested Callback	The number of contacts marked for callback for this campaign. This means that the agent accepted the call (by clicking Accept), got connected to the contact, the contact requested a callback, and the agent selected the 'CallBack'option. A call that is accepted by an agent, marked for callback, later presented to and accepted by another agent (at the callback time), and marked for callback again is counted twice towards the number of callback calls.
Avg Talk Duration	The average time in HH:MM:SS (hours, minutes, seconds) that agents spend talking on outbound calls for this campaign. The durations consider all calls that were Agent Accepted and classified as Voice. If a call is transferred or conferenced back to the route point, the campaign talk duration does not include the talk time of agents who handle the call after it came through the route point.
Longest Talk Duration	The longest time in HH:MM:SS (hours, minutes, seconds) that an agent has spent talking on an outbound call. The durations consider all calls that were Agent Accepted and classified as Voice.

Table 2-142 describes the available fields for the Overall Cisco Unified CCX Stats report.

Table 2-142 Available Fields, Overall Cisco Unified CCX

Row Heading	Description
CSQs	Number of CSQs currently configured. If a CSQ is added or removed, this statistic reflects that change.
Logged-in Resources	Number of resources currently logged in.
Talking Resources	Number of resources currently talking. **Note** This number includes resources in Talking, Work, and Reserved states.
Ready Resources	Number of resources currently ready.
Not Ready Resources	Number of CSQs currently configured. If a CSQ is added or removed, this statistic reflects that change.
Work Resources	Number of resources currently in Work.
Reserved Resources	Number of resources currently in reserved.
Total Contacts	Number of total contacts that have arrived since the statistics were last reset. This includes contacts that are waiting, contacts connected to a resource, and contacts that have disconnected. If a resource transfers to or conferences with a route point, this value increases.
Contacts Waiting	Number of contacts waiting to be connected to a resource. This column also displays how long the oldest contact in the queue has been waiting. **Note** A contact is shown as waiting until the call is *answered* by the agent. This means that, even if the phone is ringing at the agent, the contact will still show as waiting in RTR.
Oldest Contact in Queue	The length of time the oldest contact has waited in queue.
Contacts Handled	Number of contacts that have been handled by a resource.
Contacts Abandoned	Number of contacts that have arrived and disconnected before being connected to a resource.
Avg Talk Duration	Average duration (in seconds) that resources spend talking on contacts. Talk duration starts when a contact first connects to a resource and ends when the contact disconnects from the last resource to which it was connected.
Avg Wait Duration	Average wait time (in seconds). It begins when the contact enters the system and ends when the contact stops waiting. Wait duration does not include hold time. The time a contact spends on a CTI port prior to getting queued is included in this report.
Longest Talk Duration	Longest talk duration (in seconds) of a contact. Talk duration does not include hold time.
Longest Wait Duration	Longest wait (in seconds) for a contact to be connected to a resource. Wait duration does not include hold time.
Current Wait Duration	Wait duration of contact (in seconds) specified in contact field (that is, how long this contact has waited).

Table 2-142 Available Fields, Overall Cisco Unified CCX (continued)

Row Heading	Description
Preview	The total number of outbound calls currently previewed but that have not been accepted, rejected, or closed by the agents.
Connected	The total number of Outbound calls currently connected to agents. When an agent conferences in other agents, the call is counted once towards the total number of connected calls.
Active	The total number of Outbound calls currently previewed by or connected to agents. Active Calls = Previewed + Connected.
Timed-Out	The total number of outbound calls that timed out. A call is considered timed out when it is presented to an agent and not accepted, rejected, or closed within the allocated time. These contacts will be dialed again. If a contact is timed out at multiple agents, this field is incremented each time the contact is timed out at each agent.
Offered	The total number of outbound calls offered. A call is considered offered when it is presented to an agent as part of this campaign. A contact that is presented to an agent, skipped/rejected by that agent, and then presented to the same agent or to another agent is counted twice towards the number of calls offered. Offered = Accepted + Rejected + Closed + Timed-out.
Accepted	The total number of outbound calls accepted. A call is considered accepted if an agent has clicked Accept when presented the call. A call that is presented to an agent, skipped/rejected by that agent, presented to another agent, and then accepted by that other agent is counted once towards the number of calls accepted.
Rejected	The number of outbound calls that were skipped or rejected by an agent. This means that the agent selected 'Reject', 'Skip', or 'Cancel Reservation'. These contacts will be dialed again. If a contact is rejected by multiple agents, this field is incremented each time the contact is rejected. The number of Rejected is also incremented each time an agent drops the preview call while it is ringing at the customer's contact.
Closed	The number of Outbound contacts that were closed by agents. This means that the agent selected 'Skip-Close' or 'Reject-close'. These contacts will not be dialed again.
Voice	The number of outbound calls that ended in successful customer contact. This means that an agent accepted the call (by clicking Accept) AND selected a classification of 'Voice' or 'Do Not Call' for this contact.

Table 2-142 Available Fields, Overall Cisco Unified CCX (continued)

Row Heading	Description
Answering Machine	The number of outbound calls that connected to an answering machine. This means that the agent accepted the call (by clicking Accept), got connected to the answering machine and selected the "Answering Machine" option from the contact Reclassfication dropdown. **Note** The agent can manually reclassify the contact as 'Answering Machine' while the customer contact is on the call or when the agent has gone into the Work state after the call.
Invalid Number	The number of outbound calls that were dialed to an invalid number. This means that the agent accepted the call (by clicking Accept), got connected to the customer contact, and selected the "Invalid Number" option from the contact Reclassification dropdown. It also includes the number of outbound calls that failed at the network level. **Note** The agent can manually reclassify the contact as 'Invalid Number' while the customer contact is on the call or when the agent has gone into the Work state after the call.
Requested Callback	The number of contacts marked for callback. This means that the agent accepted the call (by clicking Accept), got connected to the contact, the contact requested a callback, and the agent selected the 'CallBack'option. A call that is accepted by an agent, marked for callback, later presented to and accepted by another agent (at the callback time), and marked for callback again is counted twice towards the number of callback calls.
Avg Talk Duration	The average time in HH:MM:SS (hours, minutes, seconds) that agents spend talking on outbound calls. The durations consider all calls that were Agent Accepted and classified as Voice. If an Outbound call is transferred or conferenced to a route point, this average outbound talk duration does not include the talk time of agents who handle the call after it came through the route point.
Longest Talk Duration	The longest time in HH:MM:SS (hours, minutes, seconds) that an agent has spent talking on an outbound call. The durations consider all calls that were Agent Accepted and classified as Voice.

Table 2-143 describes the available fields for the Resource Cisco Unified CCX Stats report:

Table 2-143 Available Fields, Resource Cisco Unified CCX Stats Report

Column Name	Description
State	Current state of the resource. **Note:** If you have pre-existing scripts that use Get Reporting Statistic steps—and these scripts were created with a previous version of Cisco Unified CCX software—please be aware that there is a change in step behavior in Version 4.0. Prior to Version 4.0, if a Get Reporting Statistic step had the **Report Object** field set to “Resource ICD” and the **Field** field set to “State”, the step returned one of the following agent state names: Available, Unavailable, and In-Session. In Version 4.0, a Get Reporting Statistic step with the same configuration returns one of the “new” agent state names: Ready, Not Ready, and Talking. This change can impact pre-Version 4.0 scripts where subsequent steps act on the results of Get Reporting Statistics to comparing agent state *strings*; in this case, the script will need to be modified to reflect the revised agent state names.
Duration in Same State	Length of time (in seconds) that the resource has remained in the current state.
Contacts Presented	Number of contacts that have been connected to this resource.
Contacts Handled	Number of contacts that have been handled by this resource.
Avg Talk Duration	Average time (in seconds) that this resource spends talking to contacts.
Avg Hold Duration	Average time (in seconds) that the resource keeps contacts on hold.
Longest Talk Duration	Longest time (in seconds) that this resource has spent talking to a contact.
Longest Hold Duration	Longest time (in seconds) that this resource has placed a call on hold.
Avg Handle Duration	The average time the agent spends handling a call, which includes talk time, hold time and work time. Avg Handle Duration = Avg Talk Duration + Avg Hold Duration + Avg Work Duration (time spent in Work state).
Outbound Offered	The total number of outbound calls offered. A call is considered offered when it is presented to an agent. A contact that is presented to an agent, skipped/rejected by that agent, and then presented to the same agent or to another agent is counted twice towards the number of calls offered. Offered = Accepted + Rejected + Closed + Timed-out.

Table 2-143 Available Fields, Resource Cisco Unified CCX Stats Report (continued)

Column Name	Description
Outbound Timed-Out	The total number of outbound calls that timed out. A call is considered timed out when it is presented to an agent and not accepted, rejected, or closed within the allocated time. These contacts will be dialed again. If a contact is timed out at multiple agents, this field is incremented each time the contact is timed out at each agent.
Outbound Accepted	Total number of outbound calls accepted by this resource. A call is considered accepted if an agent has clicked Accept when presented the call. A call that is presented to an agent, skipped/rejected by that agent, presented to another agent, and then accepted by that other agent is counted once towards the number of calls accepted. For transferred or conferenced outbound calls, the call is considered handled by the resource if it is answered by that resource
Outbound Rejected	The number of outbound calls that were skipped or rejected by this agent. This means that the agent selected Reject, Skip, or Cancel Reservation. These contacts will be dialed again. The number of Rejected is also incremented each time an agent drops the preview call while it is ringing at the customer's contact.
Outbound Closed	The number of Outbound contacts that were closed by agents. This means that the agent selected 'Skip-Close' or 'Reject-close'. These contacts will not be dialed again.
Outbound Voice	The number of outbound calls that ended in successful customer contact. This means that an agent accepted the call (by clicking Accept) AND selected a classification of 'Voice' or 'Do Not Call' for this contact.
Outbound Avg Talk Duration	The average time in HH:MM:SS (hours, minutes, seconds) that agents spend talking on outbound calls. The durations consider all calls that were Agent Accepted and classified as Voice. This talk duration includes talk time spent by a resource handling an outbound call that was transferred or conferenced to a route point.
Outbound Avg Hold Duration	The average time in HH:MM:SS (hours, minutes, seconds) that the Resource has spent holding the outbound calls among accepted calls. The durations consider all calls that were Agent Accepted and classified as Voice.
Outbound Longest Talk Duration	The longest time in HH:MM:SS (hours, minutes, seconds) that an agent has spent talking on an outbound call. The durations consider all calls that were Agent Accepted and classified as Voice.
Outbound Longest Hold Duration	The longest time in HH:MM:SS (hours, minutes, seconds) that the Resource has spent holding an outbound call among accepted calls. The durations consider all calls that were Agent Accepted and classified as Voice

Table 2-144 describes the available fields for the CSQ Cisco Unified CCX Stats report.

Table 2-144 Available Fields, CSQ Cisco Unified CCX Stats Report

Column Name	Description
Logged-In Resources Talking Resources/ Ready Resources/ Not Ready Resources/ Work Resources/ Reserved Resources	Number of resources who are logged in for this CSQ and who are in the talking, ready, not ready Work Resources, and Reserved Resources states. Values for these items are separated by colons. Values are displayed in the same order that the items appear in the column heading.
Total Contacts	Number of total contacts since the statistics were last reset for this CSQ.
Contacts Waiting	Number of contacts waiting to be connected to a resource in this CSQ.
Oldest Contact in Queue	The length of time that the oldest contact in the queue has been waiting.
Contacts Handled	Number of contacts that have been handled by this CSQ.
Contacts Abandoned	Number of contacts that have been abandoned by this CSQ.
Contacts Dequeued	Number of contacts that have been dequeued from this CSQ.
Avg Talk Duration	Average time (in seconds) that agents in this CSQ spend taking to contacts.
Avg Wait Duration	Average time (in seconds) that contacts have waited to be connected to a resource in this CSQ. Wait begins when the contact is queued and ends when the contact stops waiting. Wait duration does not include hold time. The time a contact spends on a CTI port prior to getting queued is not included in this wait time.
Longest Talk Duration	Longest time (in seconds) that agents in this CSQ spend talking to contacts.
Longest Wait Duration	Longest wait (in seconds) for a contact to be connected to a resource.
Expected Wait Time	The expected wait time of the contact specified in the contact field. That is, how long this contact will wait before it is connected to an agent.
Position in Queue	Position in queue of the contact specified in the contact field for the specified CSQ.
Current Wait Duration	Wait duration of contact (in seconds) specified in contact field (that is, how long this contact has waited).

Request Route Step

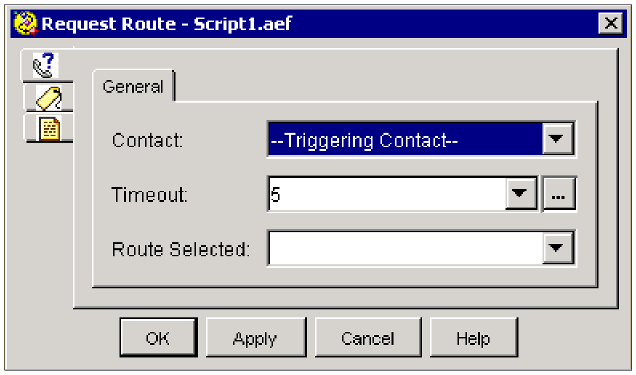
Use the Request Route step to request a calling routing location from Cisco UnifiedICME software. A Cisco Unified CCX script can then use that location to redirect a call.

Note This step can be used with the Cisco UnifiedIPCC Gateway. See the *Cisco Unified Contact Center Express Gateway Gateway Deployment Guide* for more information.

The Request Route step has two output branches:

- **Selected**. The Request Route step successfully returned a routing destination from Cisco UnifiedICME software.
- **Failed**. The Request Route step failed to return a routing destination from Cisco UnifiedICME software.

Figure 2-162 Request Route Customizer Window


Request Route - Script1.aef

General

Contact: --Triggering Contact--

Timeout: 5

Route Selected:

Table 2-145 describes the properties of the Request Route customizer window.

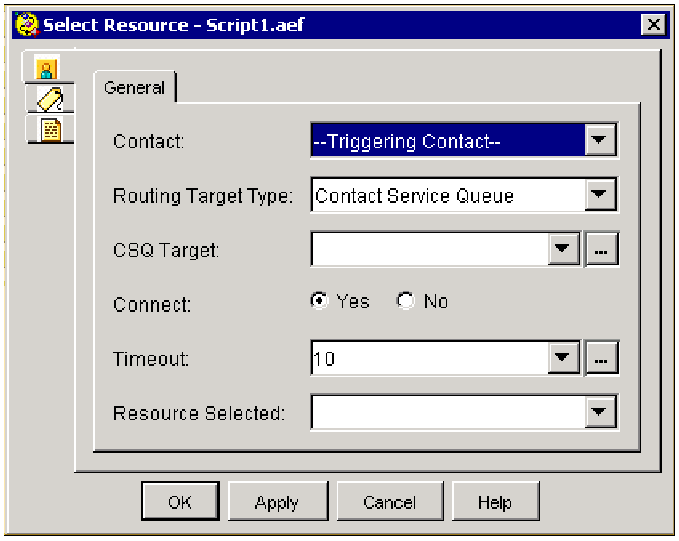
Table 2-145 Request Route Properties

Properties / Buttons	Description
Contact	Contact that is making the route request.
Timeout	Variable or expression indicating the amount of time (in seconds) that the system should wait for a reply from the Cisco UnifiedICME software
Route Selected	Variable indicating the post routing result expected from the Cisco UnifiedICME software.

Select Resource Step

Use the Select Resource step to queue a call to a specific set of agents and optionally to connect the call to the agent the system chooses.

Figure 2-163 Select Resource Customizer Window



Select Resource - Script1.aef

General

Contact: --Triggering Contact--

Routing Target Type: Contact Service Queue

CSQ Target:

Connect: [✓ Yes] No

Timeout: 10

Resource Selected:

Note If you change the Routing Target field setting to Resource, the CSQ Target field is renamed to Resource Target. A resource is a specific agent that is represented by the user variable specified in Resource Target field.

The Select Resource step offers a call to a Contact Service Queue (CSQ) or Resource:

- A **CSQ** contains an associated set of agents who are capable of handling a certain type of call. The incoming call will be handled by one of the available agents in the CSQ.

  An example of a Contact Service Queue is *sales department*, whose resources can be all the sales representative within your company.
- A **resource** is a specific agent and is indicated by user ID or agent extension.

The Select Resource step has different output branches, depending on the Routing Target Type and Connect settings:

- If the Routing Target Type setting is **CSQ** and Connect is:
  - **Yes**, the branches are Connected and Queued.
  - **No**, the branches are Selected and Queued.
- If the Routing Target Type setting is **Resource** and Connect is:
  - **Yes**, the branches are Connected and Failed.
  - **No**, the branches are Selected and Failed.

Note The Connect field allows you to specify whether the call will be connected to a CSQ/agent resource directly from Select Resource or use Select Resource in conjunction with the Connect Step. For more information, see Connect Step.

Table 2-146 describes the properties of the Select Resource customizer window.

Table 2-146 Select Resource Properties

Properties / Buttons	Description
Contact	Variable containing the triggering contact information for this step.
Routing Target Type	Variable indicating the routing method. One of the following: • **Contact Service Queue**—Call will be routed to an available agent in the specified CSQ. • **Resource**—Call will be routed to the specified agent. Select this option for Agent Based Routing feature. **Note** If you set this field to Resource, the CSQ Target field is renamed to Resource Target. Resource Routing Target Type is only available for Cisco Unified CCX Enhanced Edition. If you use Resource Routing Target Type with Cisco Unified CCX Standard edition, Cisco Unified CCX Engine will be unable to load the script.
CSQ Target	String variable or expression identifying the CSQ target
Connect	Option for the call to be connected to the specified Resource ID the instant the resource becomes available: • **Yes**—The call is automatically connected to the available resource as soon as it becomes available. • **No**—The resource is selected but not connected until additional script steps are executed. **Note** If you choose not to connect the call as soon as the resource becomes available, the Cisco Unified CCX Desktop of the chosen resource will be designated Reserved. Because the agent's phone will not ring until a Connect step executes in the script, a few seconds may elapse while other steps in the Selected output branch execute prior to the execution of the Connect step. During this extra time, the agent's state remains in Reserved state.
Timeout	Variable or expression containing the length of time, in seconds, before the contact is retrieved back into the queue. (Default is 10 seconds.) This value must be lower than the Call Forward No Answer timeout in the Cisco Unified Communications Manager.
Resource Target	User variable or expression identifying the target Resource.

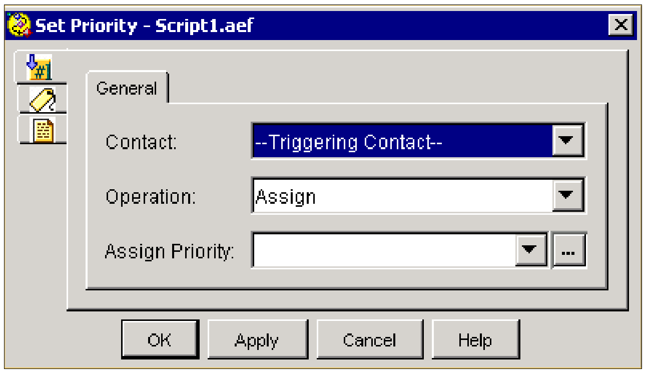
Caution Never insert a Goto step in the **middle** of a Select Resource step flow. (Doing so can cause an active agent to enter a Reserved state that can only be exited by logging out of the system.)

Set Priority Step

Use the Set Priority step to assign a call higher or lower priority in a queue.

Note The Set Priority step is only available if you have licensed the Cisco Unified CCX *Enhanced* or *Premium* product package.

Figure 2-164 Set Priority Customizer Window



Contact: --Triggering Contact--
Operation: Assign
Assign Priority:

Every contact has one priority for all CSQs for which it is queued. The priority of the contact can be set or changed at any time during the execution of the script. All calls have a default priority of 1.

Table 2-147 describes the properties of the Set Priority customizer window.

Table 2-147 Set Priority Properties

Properties / Buttons	Description
Contact	Variable identifying the contact for which to change the priority. The default is the triggering contact.
Operation	Assign, Increase, or Decrease
Assign Priority	Select a numerical priority from 1 (lowest) to 10 (highest) or specify an expression containing a number.

ICM Step

The step in the ICME palette of the Cisco Unified CCX Editor is used in conjunction with the Cisco UnifiedContact Center Enterprise solution.

Note Prior to the Cisco CRS 4.0(x) release, the ICM palette included the Get ICM Data and Set ICM Data steps. Beginning With the Cisco CRS 4.0(x) release, these steps were renamed Get Enterprise Call Info and Set Enterprise Call Info and moved to the Call Contact palette.

Cisco UnifiedIntelligent Contact Management Enterprise (ICME) software is an application that routes incoming calls across several geographically distributed call centers.

This section contains the following topic:

- Set ICM Result Step

Figure 2-165 shows the step in the ICM palette as it appears in the Palette pane of the Cisco Unified CCX Editor.

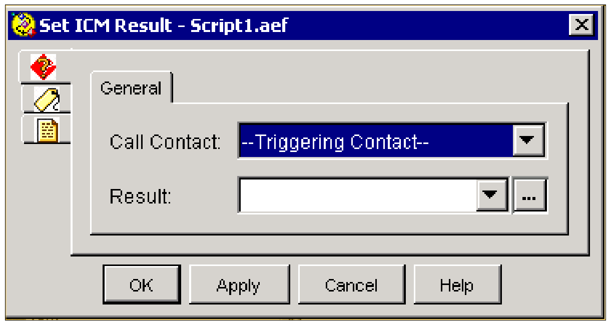
Figure 2-165 ICM Palette Step

Note For more information about Cisco Unified ICME software, ICM subsystem configuration, and Cisco Unified ICM VRU (Voice Response Unit) scripts, see the *Cisco Unified Contact Center Express Administration Guide*.

Set ICM Result Step

Use the Set ICM Result step to set the value of the VRU Script request message that Cisco Unified CCX sends to the Cisco UnifiedICME software when it completes a VRU script.

Figure 2-166 Set ICM Result Customizer Window


Set ICM Result - Script1.aef

General

Call Contact: --Triggering Contact--

Result:

The VRU Script result tells the Cisco UnifiedICME software whether the script ran successfully (true) or not (false).

If the script returns a false value, the Cisco UnifiedICME software runs the failure path of the Run VRU Script node in the Cisco Cisco UnifiedICME script.

By default, Cisco Unified CCX returns **true** to the Cisco UnifiedICME software.

Note Because the VRU Script result message must be sent from Cisco Unified CCX to the Cisco Cisco Unified ICME software, use the Set ICM Results step only for Cisco Unified ICME VRU scripts. For more information, see the *CiscoUnified Contact Center Express Administration Guide*.

Table 2-148 describes the properties of the Set ICM Result customizer window.

Table 2-148 Set ICM Result Properties

Properties / Buttons	Description
Call Contact	Variable for which you want to set the result. Default is Triggering Contact.
Result	Boolean variable—or expression that resolves to a Boolean variable—indicating the outcome of the execution of the Set ICM Result step. **True—**(Default) The Cisco Cisco UnifiedICME software will follow the success path of the Run VRU Script node in the Cisco Cisco UnifiedICME script. **False—**The Cisco Cisco UnifiedICME software will follow the failure path of the Run VRU Script node in the Cisco Cisco UnifiedICME script.

Java Steps

Note The Cisco Unified CCX Enhanced, and Cisco Unified CCX Premium licences include the Java license. The Cisco Unified CCX Standard license does not include the Java license. You need the Java license for full Java functionality. For further information, see the Cisco Unified CCX Expression Language Reference Guide.

Note When a step in the Java palette is enabled, the Java functionality of the Cisco Unified CCX Expression language is also enabled if your product license includes it.

Use the Cisco Unified CCX Expression Editor to create Java method calls, constructor calls, or attribute access and to replace the deprecated Java step functionality. For more information, see the *Cisco Unified Contact Center Express Scripting and Development Series: Volume 3, Expression Language Reference*.

This section contains the following topics:

- Create Remote Java Object Step
- Create Java Object Step
- Execute Java Method Step
- Set/Get Java Property Step

Note
- You should be experienced in Java programming to use Java Type variables and the steps in the JAVA palette of the Cisco Unified CCX Editor.
- The Create Java Object, Execute Java Method, and Set/Get Java Property steps are deprecated and, although no longer visible in the Cisco Unified CCX Editor palette, they are still visible in old scripts and are supported.

- Use the Java tool in the Cisco Unified CCX Expression Editor to enter Java functionality that the deprecated steps offered. The Expression Editor Java tool is easier to use and more functional than the deprecated steps. For more information, see the *Cisco Unified Contact Center Express Scripting and Development Series: Volume 3, Expression Language Reference*.
- To use full Java functionality, you need a Java license.

Java Licensing

- In Cisco Unified CCX 4.x, expressions are validated against installed licenses to make sure that they do not violate license agreements. This validation is performed by the Cisco Unified CCX Engine whenever a script is loaded or whenever a prompt template or grammar template is accessed and evaluated.
- For script expressions containing TTS or Java features to work during runtime, you must have either a Cisco UnifiedIP-IVR or a Cisco Unified CCX Premium license.
  An example of a TTS feature is a TTS prompt complex literal. A Java feature is a complex expression block, a Java-like statement, method, constructor invocation expression, or a field access expression.
- Any license violation will be recorded in the logs and prevent the scripts from being loaded in memory.

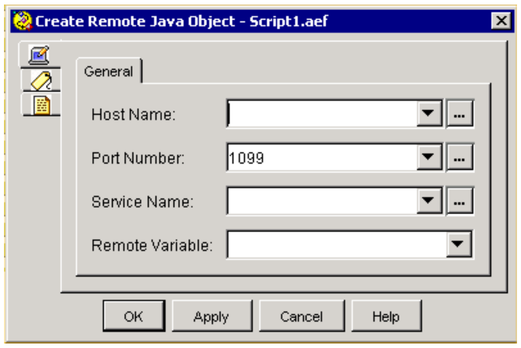
Figure 2-167 shows the step in the JAVA palette as it appears in the Palette pane of the Cisco Unified CCX Editor.

Figure 2-167 JAVA Palette Step

Create Remote Java Object Step

Use the Create Remote Java Object step to create a Java object representing an object hosted on a remote computer.

Figure 2-168 Create Remote Java Object Customizer Window



Host Name:	
Port Number:	1099
Service Name:	
Remote Variable:	

Once the Java object is created, you can then use the Execute Java Method step on the returned object to invoke methods on this remote object.

Table 2-149 describes the properties of the Create Remote Java Object customizer window.

Table 2-149 Create Remote Java Object Properties

Properties / Buttons	Description
Host Name	Variable or expression indicating the DNS[1] or WINS[2] host name of the remote host or the IP address.
Port Number	Variable or expression indicating the TCP port number of the RMI[3] registry on the remote host that you will use to connect to it. The default port number for RMI is 1099.
Service Name	Variable or expression indicating the service name under which the remote object was registered in the RMI registry.
Remote Variable	Variable that represents the remote Java object.

1. DNS = Domain Name Service
2. WINS = Windows Internet Naming Service
3. RMI = Remote Method Invocation

Deprecated Java Steps

The following steps are depreciated. That is, although these steps are no longer visible in the Cisco Unified CCX Editor palette, they are still visible in old scripts and are supported:

- Create Java Object
- Execute Java Method
- Set/Get Java Property

Note

- If it is a resulting script from 3.x, there is no upgrade path. The steps will appear in the 4.5 system, so you need to remove these steps manually and rebuild using the new 4.5 java step in the Cisco Unified CCX Editor.
- Use the Java tool in the Cisco Unified CCX Expression Editor to enter Java functionality that the deprecated steps offered. The Expression Editor Java tool is easier to use and more functional than the deprecated steps. For more information, see the *Cisco UnifiedContact Center Express Scripting and Development Series: Volume 3, Expression Language Reference*.

Create Java Object Step

Use the Create Java Object step to instantiate a variable or an object of a specified Java class. You can create object instances for any variables or Java classes that you defined in the Variable pane of the Cisco Unified CCX Editor or that are included in the definitions of the steps in your palette.

Note Any customer-defined classes you create should be uploaded to the Document repository under \default\classpath. In addition, you must use Cisco Unified CCX Administration to specify the classpath the system should use.

The Variable pane of the Cisco Unified CCX Editor supports 20 built-in data types:

- **int** (java.lang.Integer)—Whole numbers.
- **String** (java.lang.String)—Set of unicode characters.
- **boolean** (java.lang.Boolean)—True or False.
- **float** (java.lang.Float)—Decimal numbers.
- **Prompt** (com.cisco.prompt.Playable)—Prompt object created with prompt expressions or returned using prompt steps.
- **char** (java.lang.Character)—Characters, such as the letters in the alphabet.
- **Contact** (com.cisco.contact.Contact)—Represents a call, e-mail, or HTTP request.
- **Session** (com.cisco.session.Session)—Information that can be used to track contacts as they move through the system.
- **User** (com.cisco.user.User)—Information useful for user authentication.
- **Grammar** (com.cisco.grammar.Recognizable)—Words or Dual Tone Multi-Frequency (DTMF) digits that can be recognized by the Cisco Unified CCX script.
- **Document** (com.cisco.doc.Document)—Any type of document, such as a file, a URL, or a recording.
- **Currency** (com.cisco.util.Currency)—Monetary format.
- **Date** (java.util.Date)—Date information.
- **Language** (java.util.Locale)—Language information.
- **Time** (java.sql.Time)—Time information.
- **BigDecimal** (java.math.BigDecimal)—Large decimal number.
- **BigInteger** (java.math.BigInteger)—Large integer.
- **double** (java.lang.Double)—Expanded Float variable.
- **long** (java.lang.Long)—Expanded Integer variable.
- **byte** (javalang.Byte)—Byte information.
- **short** (javalang.Short)—Short information.
- **Iterator** (javalang.Iterator)—Iterator information.

Note You can only use Java types compatible with JDK 1.4.2_05 (Java 2).

The Create Java Object customizer window contains two tabs:

- Class Assignment Tab (Create Java Object Step)
- Class Information Tab (Create Java Object Step)

The following sections describe these tabs.

Class Assignment Tab (Create Java Object Step)

Use the Class Assignment tab of the Create Java Object customizer window to assign a class to the object.

Figure 2-169 Create Java Object Customizer Window—Class Assignment Tab

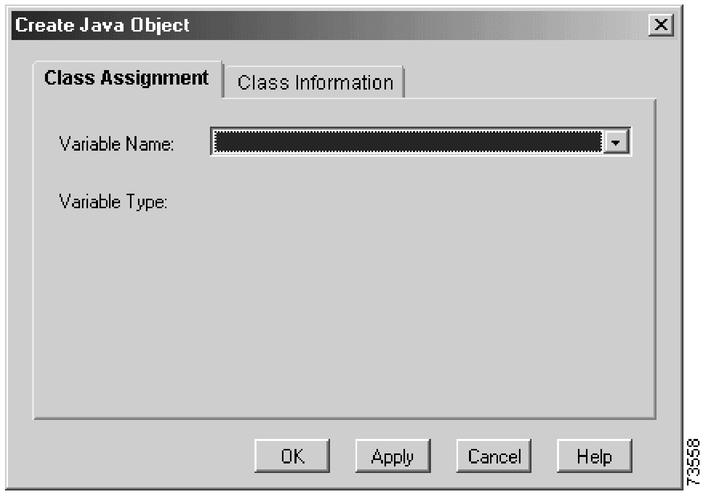


Table 2-150 describes the properties of the Class Assignment tab of the Create Java Object customizer window.

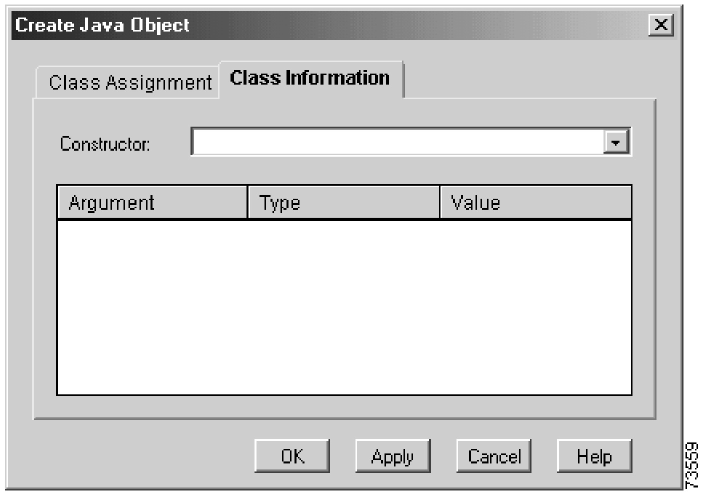
Table 2-150 Create Java Object Properties—Class Assignment Tab

Properties / Buttons	Description
Variable Name	Variable that represents a specific Java class.
Variable Type	(Display only) A list of Java classes that corresponds to the variable you select.

Class Information Tab (Create Java Object Step)

Use the Class Information tab of the Create Java Object customizer window to choose the constructor for your chosen Java class.

Figure 2-170 Create Java Object Customizer Window—Class Information Tab



A *constructor* is a function that allows you to instantiate an instance of this class.

Note You can only access constructors compatible with JDK 1.4.2_05 (Java 2).

If any of the parameters to be passed to a constructor is of type InputStream or Reader, the Create Java Object step allows you to select a Document object and automatically convert the document to an InputStream or Reader object.

If the type created is an instance of InputStream or Reader, the Create Java Object step allows you to assign the result to a Document variable by converting the result into the appropriate format. However, the resulting document will only be accessible once by a single output step (unless cached in memory using the Cache Document step).

Table 2-151 describes the properties of the Class Information Tab of the Create Java Object customizer window.

Table 2-151 Create Java Object Properties—Class Information Tab

Properties / Buttons	Description
Constructor	Drop-down menu from which you choose the applicable constructor for the specific Java class.
Argument/Type/Value	(Display only) A list of the arguments, types, and values for the chosen constructor.
Set (button)	To assign the value to the constructor, highlight the entry in the Argument list box and click **Set**. Choose a value from the Variable drop-down list—or use the Expressions Editor to specify a value—and then click **OK**; the value appears in the Value column next to the Argument you selected.

Execute Java Method Step

Use the Execute Java Method step to execute a specified static or non-static method of a Java class for a defined variable.

Note You can only use methods compatible with JDK 1.4.2_05 (Java 2),

The expression language now allows calling Java methods directly on objects, variables or other expressions in a form natural to Java developers.

For example, you could enter:

<var>.toString()

to invoke the to String() method of the object stored in the specified script variable. This is more useful for calling static methods, as they do not require a dummy variable of the same type. One can simply say:

<classname>.<method>(<args>).

The Execute Java Method customizer window contains two tabs:

- Explore Tab (Execute Java Method Step)
- Explore Class Information Tab (Execute Java Method Step)

The following sections describe these tabs.

Explore Tab (Execute Java Method Step)

Use the Explore tab of the Execute Java Method customizer window to select a variable for which you want to execute a specific Java method.

Figure 2-171 Execute Java Method Customizer Window—Explore Tab

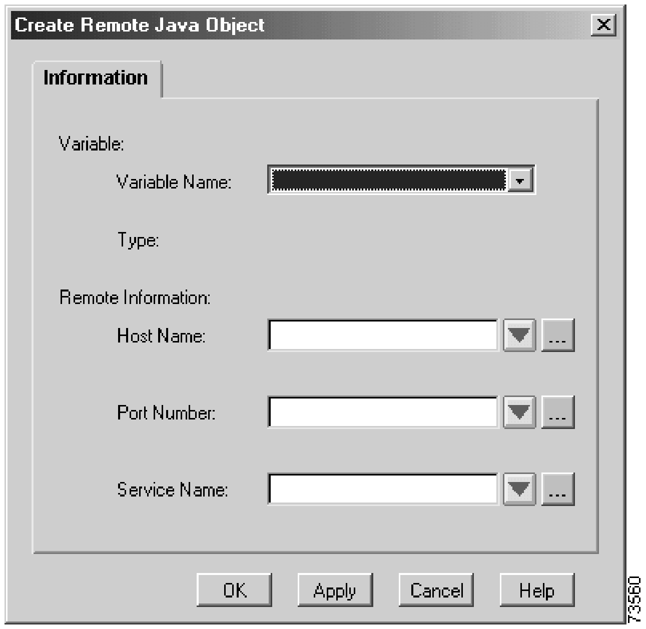


Table 2-152 describes the properties of the Explore tab of the Execute Java Method customizer window.

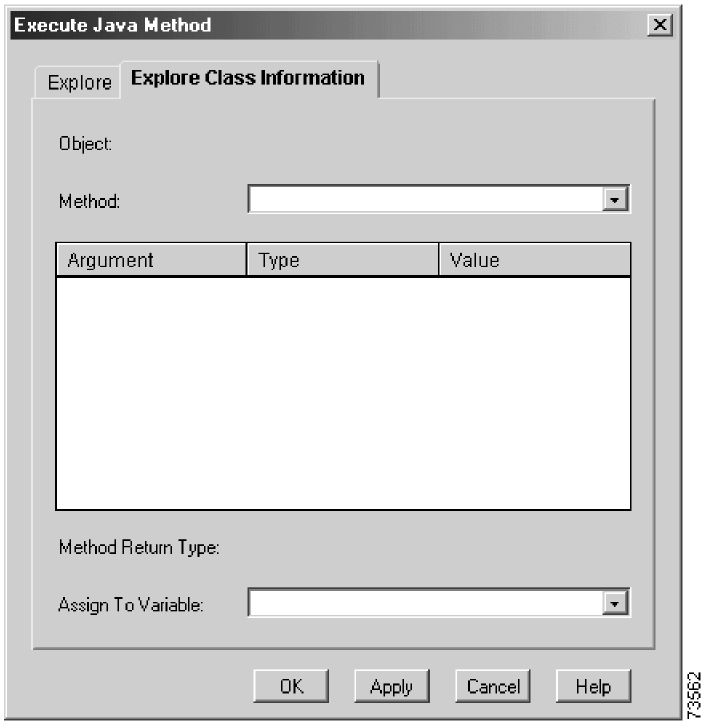
Table 2-152 Execute Java Method Properties—Explore Tab

Properties / Buttons	Description
Select Variable	Variable on which you want to invoke a method
Variable Name	(Display only) Name of the selected variable.
Variable Type	(Display only) Type of selected variable.
Browse the Class	Members defined for the class assigned to this variable. To view the different members of the class in the Browse the Class display box, click the plus sign (+) in this box.

Explore Class Information Tab (Execute Java Method Step)

Use the Explore Class Information tab of the Execute Java Method customizer window to choose a method to execute from the class selected in the Explore tab.

Figure 2-172 Execute Java Method Customizer Window—Explore Class Information Tab



If any of the parameters to be passed to a constructor is of type InputStream or Reader, the Execute Java Method step allows you to select a Document object and automatically convert the document to an InputStream or Reader object.

If the return type is an instance of InputStream or Reader, the Execute Java Method step allows you to assign the result to a Document variable by converting the result into the appropriate format.

Table 2-153 describes the properties of the Explore Class Information ab of the Execute Java Method customizer window.

Table 2-153 Execute Java Method Properties—Explore Class Information Tab

Properties / Buttons	**Description**
Object	Variable on which you want to invoke a method
Method	Name of the method to call. **Note** The list contains the methods available for the specified variable in Create Java Object steps. You cannot call methods that require unsupported data types, such as short and array.
Argument/Type/Value	List of input arguments, types, and values to the method for the specified Method.
Set (button)	To assign a value to an argument, highlight the entry in the Argument list box and click **Set**. Choose a value from the Variable drop-down list—or use the Expressions Editor to specify a value—and then click **OK**; the value appears in the Value column next to the Argument you selected.
Method Return Type	(Display only) Data type returned by the selected method.
Assign to Variable	Variable that stores the value returned by the method.

Set/Get Java Property Step

Use the Set/Get Java Property step to obtain and/or modify the value of a property defined in a variable.

Note Optionally, you can use the Expression Language and specify:
<obj or var> <attribute_name>

Note You can access or modify only properties defined since JDK 1.4.2_05 (Java 2).

If an attribute of a class of type InputStream or Reader is set, the Set/Get Java Property step accepts a parameter of type Document and automatically convert it to either an InputStream or a Reader.

If the attribute retrieved is an instance of InputStream or Reader, the Set/Get Java Property step allows you to assign the result to a Document variable by converting the result into the appropriate format. However, the resulting document will only be accessible once by a single output step (unless cached in memory using the Cache Document step).

The Set/Get Java Property customizer window contains two tabs:

- Source Tab (Set/Get Java Property Step)
- Destination Tab (Set/Get Java Property Step)

The following sections describe these tabs.

Source Tab (Set/Get Java Property Step)

Use the Source tab the Set/Get Java Property customizer window to select a source variable or field with the value you want to obtain.

Figure 2-173 Set/Get Java Property Customizer Window—Source Tab

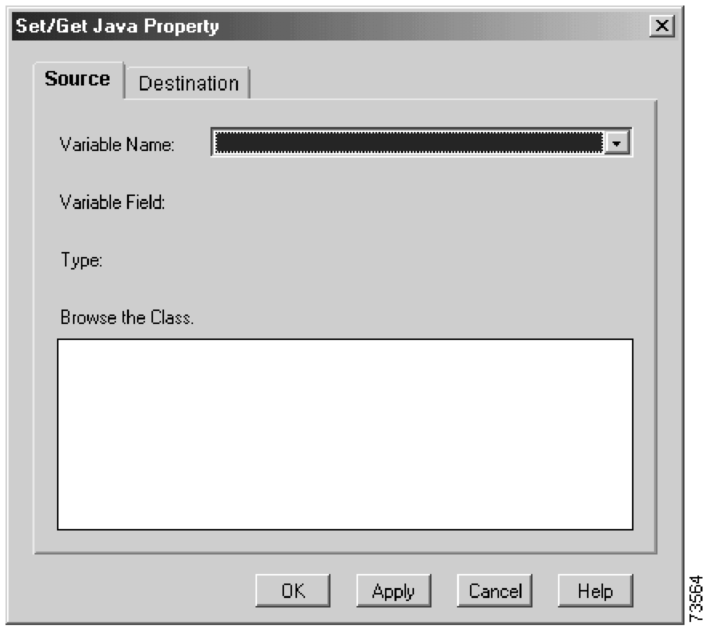


Table 2-154 describes the properties of the Source tab of the Set/Get Java Property customizer window.

Table 2-154 Set/Get Java Property Properties—Source Tab

Properties / Buttons	Description
Variable Name	Variable for which you want to obtain and/or modify the value of a Java property.
Variable Field	(Display only) Name of the selected variable.
Type	(Display only) Type of selected variable.
Browse the Class	Members defined for the class assigned to this variable. To view the different members of the class in the Browse the Class display box, click the plus sign (+) in this box.

Destination Tab (Set/Get Java Property Step)

Use the Destination tab of the Set/Get Java Property customizer window to select the variable or a field to receive the value of the source variable.

Figure 2-174 Set/Get Java Property Customizer Window—Destination Tab

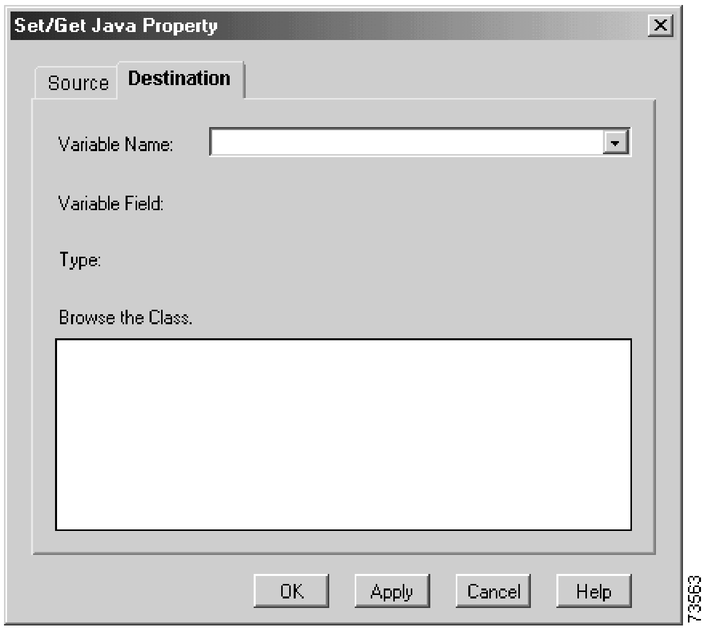


Table 2-155 describes the properties of the Destination ab of the Set/Get Java Property customizer window.

Table 2-155 Set/Get Java Property Properties—Destination Tab

Properties / Buttons	Description
Variable Name	Variable that stores the value of the specified Java property.
Variable Field	(Display only) Name of the selected variable.
Type	(Display only) Type of selected variable.
Browse the Class	Members defined for the class assigned to this variable. To view the different members of the class in the Browse the Class display box, click the plus sign (+) in this box.

Context Service Steps

Context Service is a cloud-based storage service that provides a repository for customer journey data. It enables Cisco Contact Center customers to deliver a seamless omnichannel experience with an out-of-the-box integration from Cisco Customer Collaboration products and APIs for 3rd party integration.

Note Reactive debugging in the production mode will expose all incoming data to the script debugger.

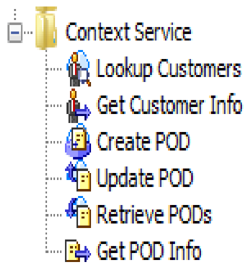
This section contains the following topics:

- Lookup Customers Step
- Get Customer Info Step
- Create POD Step

- Update POD Step
- Retrieve PODs Step
- Get POD Info Step

Figure 2-175 shows the steps in the Context Service palette as they appear in the Palette pane of the Cisco Unified CCX Editor.

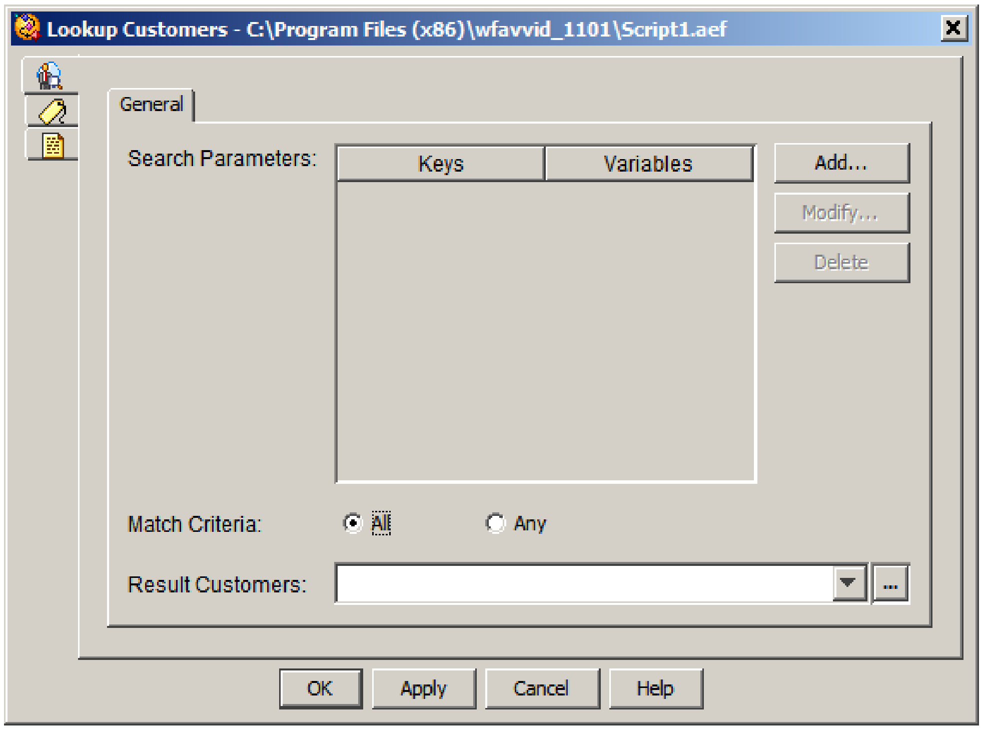
Figure 2-175 Context Service Palette Steps



Lookup Customers Step

Use the Lookup Customers step to lookup a particular customer.

Figure 2-176 Lookup Customers Customizer Window



Match Criteria: [✓ All] Any

Table 2-156 describes the properties of the Lookup Customers customizer window.

Table 2-156 Lookup Customers Properties

Property	Description
Search Parameters	There are two search parameters, Keys and Variables A **Key**is the field in the customer data that you want to search. A **Key** is of type **String** value. A **Variable** is a string variable that contains the actual key value that you want to search. You can have multiple key-variable pairs in the search parameters.
Match Criteria	To lookup a customer, select **All** or **Any** option to select the matching criteria. This controls how the search parameters are used to lookup customers. If **All** is selected only those customers which match all the specified search criteria are returned. If **Any** is selected, only those customers which match at least one of the specified search criteria are returned.
Result Customers	The array of customers matching the search criteria.

The Lookup Customers step has three possible outcomes:

- Success - when the lookup succeeds and zero or more customer objects are returned.
- Failure - when the lookup failed
- Timeout - when the connection to the context service cloud service times out. A script can retry this step after some time delay.

Get Customer Info Step

Use the Get Customer Info step to extract information from a Customer particular type of object and store it in script variables to make this information about the customer available to subsequent steps in the script.

Figure 2-177 Get Customer Info Customizer Window

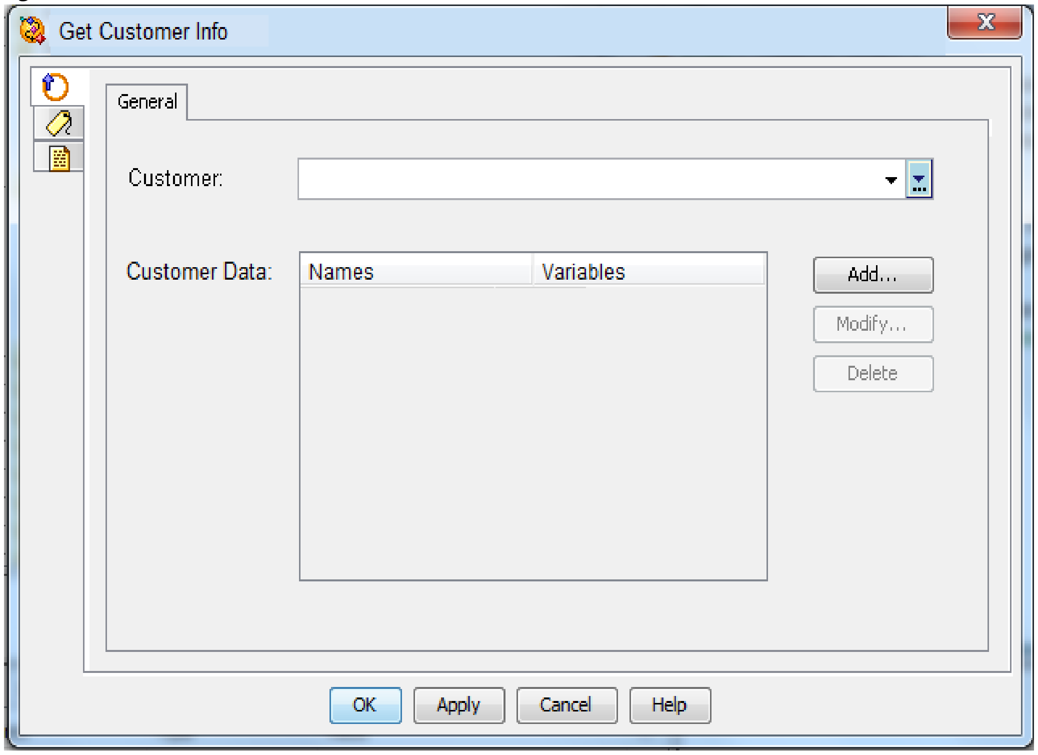


Table 2-157 describes the properties of the Get Customer Info customizer window.

Table 2-157 Get Customer Info Properties

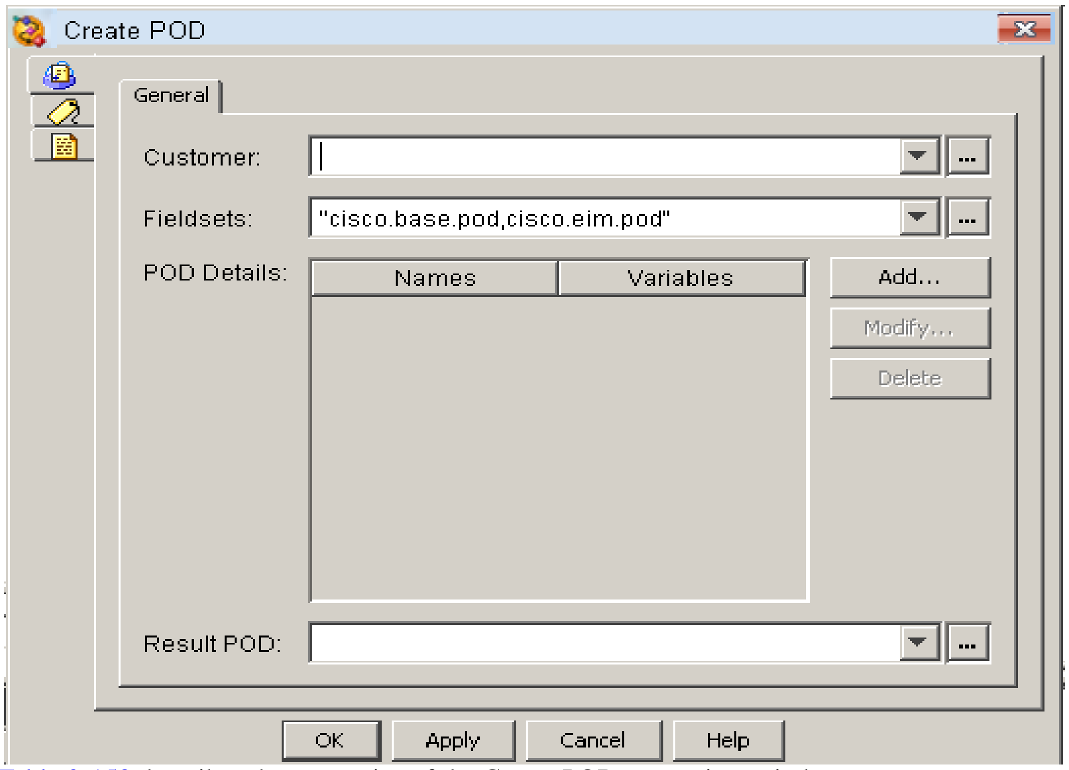
Property	Description
Customer	The Customer object from which to extract information.
Customer Details	Required information to be added for getting the customer info.
Add/Modify	Add a new Customer field and variable/Modify an existing Customer field and/or variable.
Delete	Deletes an existing Customer field and variable.

Create POD Step

Use the Create POD step to create a POD with the given information.

Note Every POD will have a default field, **Context_POD_Source_Phone**, associated with it. It will contain the customer phone number. Users have the privilege to override this field.

Figure 2-178 Create POD Customizer Window



Fieldsets: "cisco.base.pod,cisco.eim.pod"

Table 2-158 describes the properties of the Create POD customizer window.

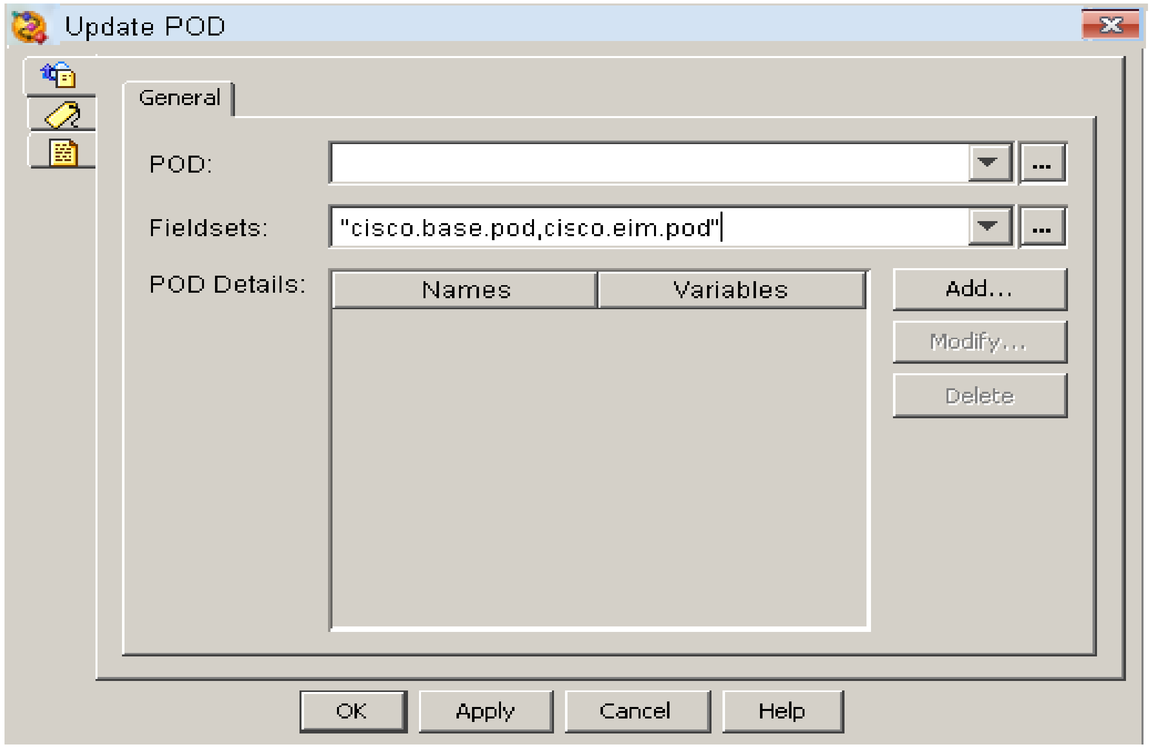
Table 2-158 Create POD Properties

Property	Description
POD	POD Object variable.
Customer	Customer to be associated for POD being created.
Fieldsets	These are Comma separated strings in the format aaa.bbb.ccc and define the valid POD names that will be specified in POD Details.
POD Details	Required information for creating a new POD.
Result POD	POD created.
Add/Modify	Add a new POD field and Value / Modify an existing POD field and/or value.
Delete	Deletes an existing POD field.

Update POD Step

Use the Update POD step to update the POD data, either by adding a new POD field/value, modifying an existing POD field and/or value, or deleting an existing POD field.

Figure 2-179 Update POD Customizer Window



Fieldsets: "cisco.base.pod,cisco.eim.pod"

Table 2-159 describes the properties of the Update POD customizer window.

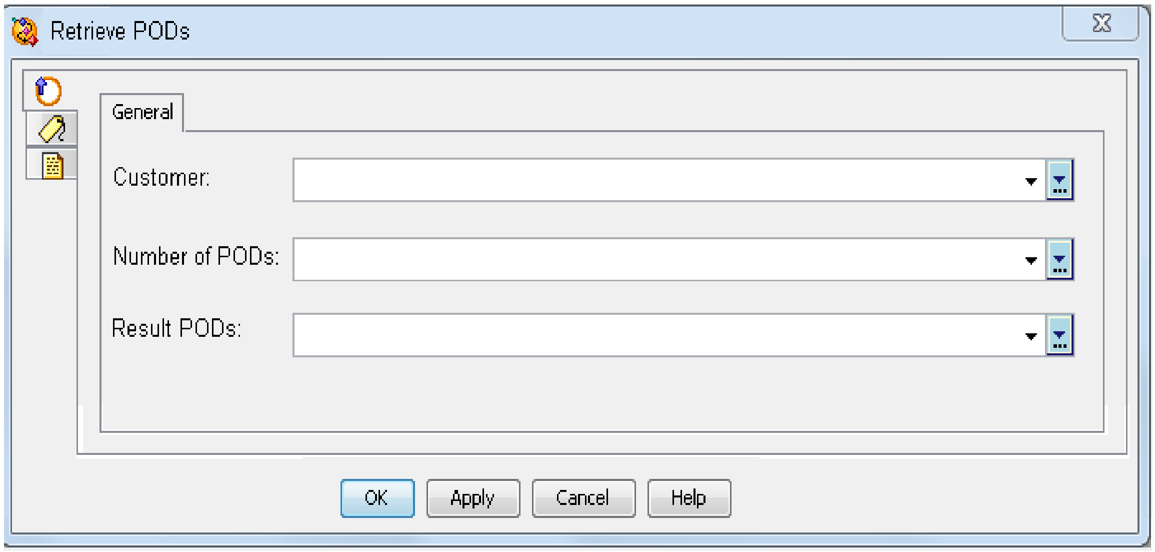
Table 2-159 Update POD Properties

Property	Description
POD	POD Object variable.
Fieldsets	These are Comma separated strings in the format aaa.bbb.ccc and define the valid POD names that will be specified in POD Details.
POD Details	The existing POD detail as Name Value pair.
Add/Modify	Add a new POD field and value/Modify an existing POD field and/or value.
Delete	Deletes an existing POD field.

Retrieve PODs Step

Use the Retrieve PODs step to retrieve POD detail.

Figure 2-180 Retrieve PODs Customizer Window



Customer:
Number of PODs:
Result PODs:

Table 2-160 describes the properties of the Retrieve PODs customizer window.

Table 2-160 Retrieve PODs Properties

Property	Description
POD	Customer variable.
Customer	Customer being associated with POD(s) retrieval.
Number of PODs	Required number of POD(s) to be entered for retrieval.
Result PODs	POD(s) retrieved.

Get POD Info Step

Use the Get POD Info step to get information about the POD.

Figure 2-181 Get POD Info Customizer Window

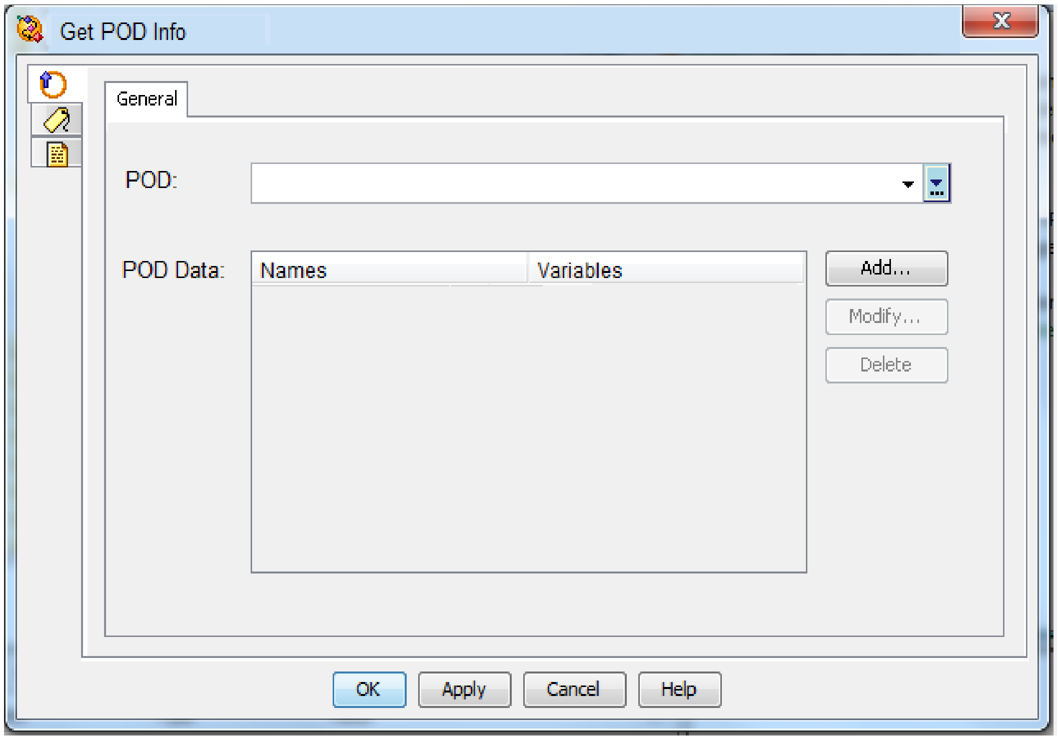


Table 2-161 describes the properties of the Get POD Info customizer window.

Table 2-161 Get POD Info Properties

Property	Description
POD	POD Object variable.
POD Details	The existing POD data as Name Value pair.
Add/Modify	Add a new POD field and variable/Modify an existing POD field and/or variable.
Delete	Deletes an existing POD field and variable.

INDEX