Auto Attendant Script Example

This chapter describes how to configure an auto attendant (AA) script. It uses the sample script aa_sample1.aef, which is included with the Cisco Unity Express Script Editor, to illustrate basic procedures for configuring auto attendant scripts.

This document contains the following sections:

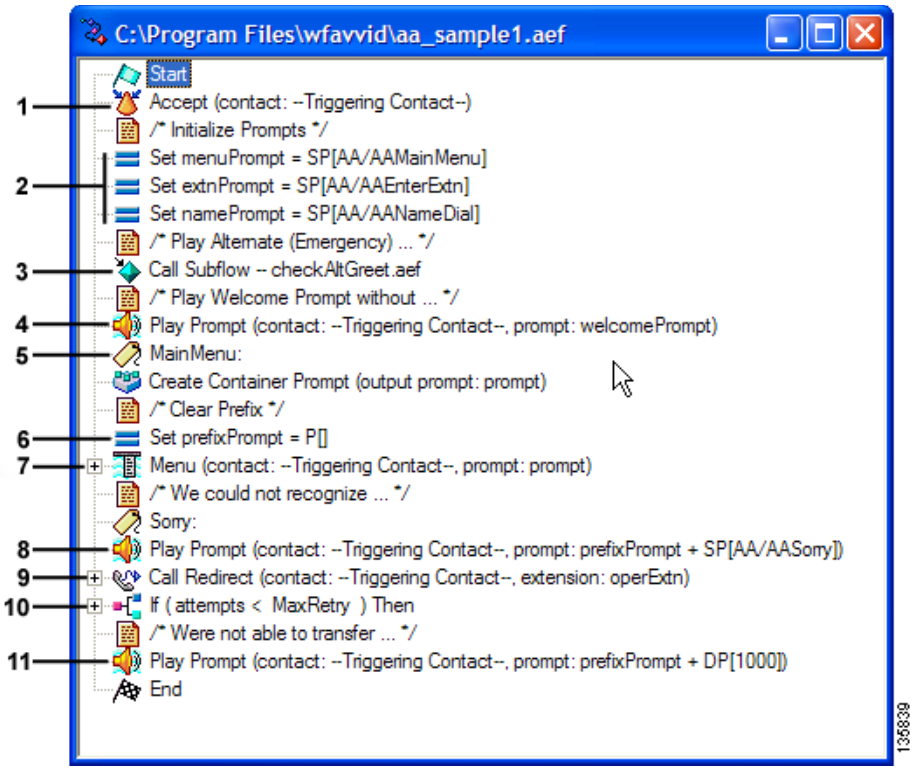
Sample Script Overview

The aa_sample1.aef file is a script that answers a call, asks for the name or extension of the person to whom the caller would like to be connected, and transfers the call.

Note You can modify the aa_sample1.aef file to create your own AA script. Make a backup copy of the aa_sample1.aef file before modifying it, so that you always have access to the original file.

Figure 9 shows the aa_sample1.aef script as it appears in the Design pane of the Cisco Unity Express Script Editor window.

Figure 9 aa_sample1.aef Script



The aa_sample1.aef script performs the following tasks:

1. The **Accept** step accepts the call.
2. The next three **Set** steps initialize, or clear, several variables: the main menu to play for the caller, the extension that will receive the call, and the name of the called person.
3. The **Call Subflow** step looks for an alternate emergency prompt and plays the prompt if required.
4. The **Play Prompt** step plays a welcome prompt, asking the caller to perform one of three actions:
   - Press “1” to enter an extension number.
   - Press “2” to enter the name of a person.

     If the caller chooses to spell a name, the script maps the letters entered against the available users defined in a specified directory and transfers the call to the primary extension of the user.

     If more than one match occurs, the script prompts the caller to choose the correct extension. If too many matches occur, the script prompts the caller to enter more characters. If no match occurs, the script prompts the caller to enter another name.
   - Press “0” to speak to an operator.

Note This **Welcome** prompt is a parameter, which means that the administrator can configure this prompt when provisioning an application with this script. (For more information on provisioning applications, refer to the *Cisco Unity Express CLI Administrator Guide* or the *Cisco Unity Express GUI Administrator Guide* for your system.

5. The **Main Menu** step is the beginning of a process that checks the caller's extension choice.
6. The **Set prefixPrompt** step initializes the beginning section of the prompt that the caller hears.
7. The **Menu** step contains a subprocess that checks the extension's status.
8. The **Play Prompt** step plays a message to the caller regarding the status of the extension. If the script receives a valid extension, it transfers the call.
   - If the destination is busy, the caller hears the system prompt, "The phone number you are trying to reach is currently busy."
   - If the destination is out of service, the caller hears the system prompt, "The phone number you are trying to reach is currently out of service."
9. The **Call Redirect** step sends the caller to the operator if the extension is not available.
10. The **If** step determines if the caller has reached the maximum number of tries to connect to a valid extension.
11. The **Play Prompt** step plays a message if the maximum number of tries has been reached without reaching a valid extension.

System Prompts

The aa_sample1.aef script uses system prompts stored as .wav files, which are installed automatically with the Cisco Unity Express software. These audio prompts include the following:

- AAMainMenu.wav—Provides a menu of choices: press 1 to enter an extension, press 2 to enter the first few characters of a user name, or press 0 to speak to an operator.
- AASorry.wav—States that the transfer was not successful.
- AABusyExtn.wav—States that the dialed extension is busy.
- AAInvalidExtn.wav—States that the entered extension is not a valid choice.
- AAExntOutofService.wav—States that the entered extension is no longer in service.
- AAWelcome.wav—Greets the caller.

  In the auto attendant application, you can configure the filename for the AAWelcome.wav prompt by selecting the **Voice Mail > Auto Attendant** menu option on the Cisco Unity Express GUI administration web interface. You can change the default welcome prompt to reference a custom prompt.

Note For custom scripts, you need to record your own prompts. You can either have them recorded professionally or you can use the AvT to record them in your own voice. For more information about the AvT, refer to the *Cisco Unity Express GUI Administrator Guide* or the *Cisco Unity Express CLI Administrator Guide* for your system.

Configuring the Sample Script

This chapter describes the steps necessary to configure the sample auto attendant script. Perform the following tasks:

Configure the Script Variables

Using the **Variable** pane of the Cisco Unity Express Script Editor, define the script variables as shown in Figure 10.

Figure 10 Variables Pane of the aa_sample1.aef Script

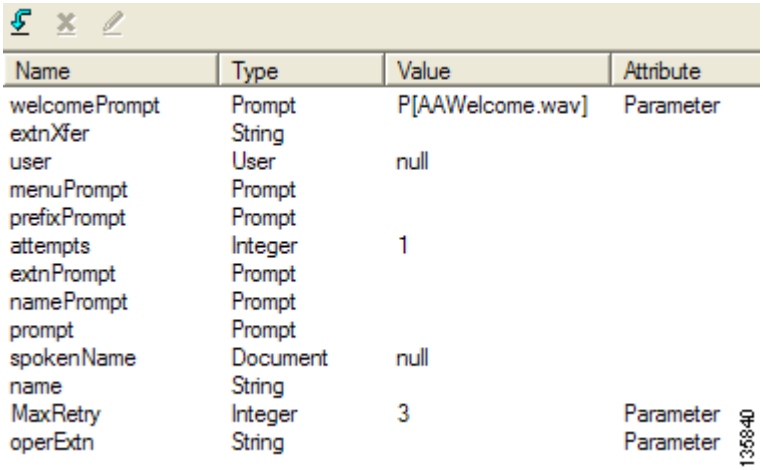


Table 4 describes the variables used in the aa_sample1.aef script.

Table 4 Variables in the aa_sample1.aef Script

Variable Name	Variable Type	Value	Function
welcomePrompt	Prompt	P[AA\Welcome.wav]	Greets the caller. Parameter. See page 57.
extnXfer	String	—	Stores the extension to which the caller is transferred. See page 40.
user	User	null	Identifies the user that the caller chooses with the Name To User step. See page 47.
menuPrompt	Prompt	—	This prompt presents the initial menu of options for calling by name or by extension. See page 33.
prefixPrompt	Prompt	—	Informs the caller of the status of the call. This value is dependent on many steps. See page 39.
attempts	Integer	1	Stores the number of times the script has attempted confirmation. See page 45.
extnPrompt	Prompt	—	Prompts the caller to enter the extension number. See page 39.

Table 4 Variables in the aa_sample1.aef Script (continued)

Variable Name	Variable Type	Value	Function
namePrompt	Prompt	—	Asks the caller to enter the name of the person the caller wants to reach. See page 37.
prompt	Prompt	—	Used for a variety of purposes throughout the script, such as playing a recorded status message, asking the caller for input, playing a menu of options, and so on. See page 37.
spokenName	Document	null	Stores the audio document of the spoken name of the person the caller is trying to reach. See page 49.
name	String	—	Stores the written name of the person the caller is trying to reach. See page 49.
MaxRetry	Integer	3	Stores the maximum retries a caller can make in this script before the script terminates the call. Parameter. See page 45.
operExtn	String	—	Stores the Operator extension the Call Redirect step uses to transfer the call to the operator. Parameter. See page 57.

Configure Steps in the Design Pane

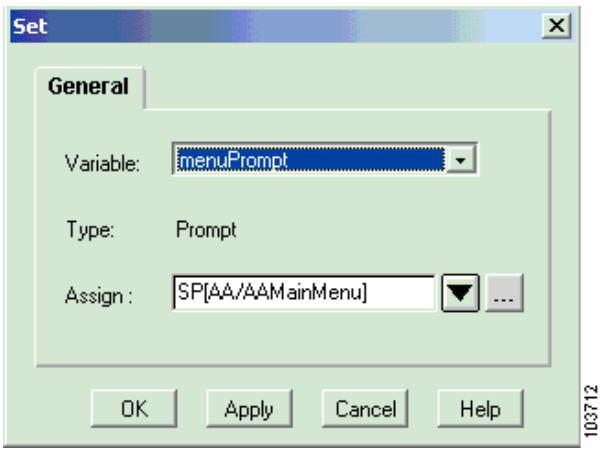
In the **Design** pane, perform the following tasks:

Step 1 Insert the **Start** step. Every script built in the **Design** pane of the Cisco Unity Express Script Editor window begins with a **Start** step, which needs no configuration and has no customizer window.

Step 2 Insert the **Accept** step. This step accepts the default contact; no configuration is necessary for this step.

Step 3 Insert the **Menu Prompt** step. This step sets the value of **menuPrompt** to SP[AA/AAMainMenu], which is the system prompt for playing the main menu, as shown in Figure 11.

Figure 11 Set Customizer Window—Configured General Tab

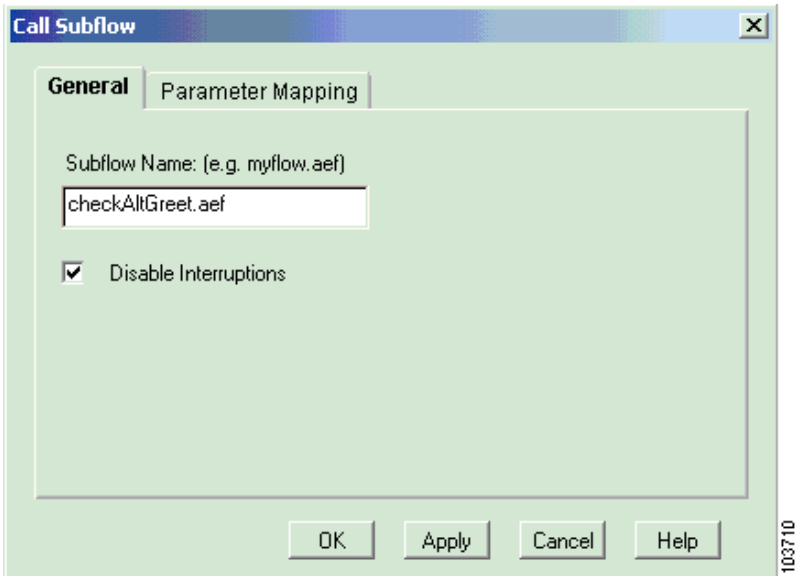


Step 4 Insert the **Extension Prompt** step. This step sets the value of **extnPrompt** to SP[AA/AAEnterExtn], which is the system prompt that asks the caller to enter the extension number.

Step 5 Insert the **Name Prompt** step. This step sets the value of **namePrompt** to SP[AA/AANameDial], which is the system prompt that asks the caller to enter the name of the called person.

Step 6 Insert the **Call Subflow** step. The **General** tab of the **Call Subflow** step sets the subflow name to **checkAltGreet.aef**, as shown in Figure 12.

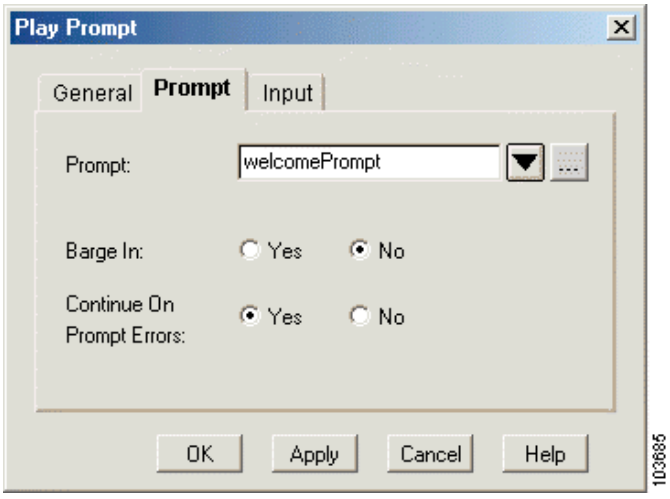
Figure 12 Call Subflow Customizer Window—Configured General Tab



The **Subflow Name**, **checkAltGreet.aef**, checks if an alternate greeting is enabled. If it is, then the subflow plays the alternate greeting and returns to the aa_sample1.aef script. Check **Disable interruptions** so that other script steps cannot interrupt the **Call Subflow** process.

Step 7 Insert the play prompt step. This step plays the **Welcome** prompt, as shown in Figure 13.

Figure 13 Play Prompt Customizer Window—Configured Prompt Tab



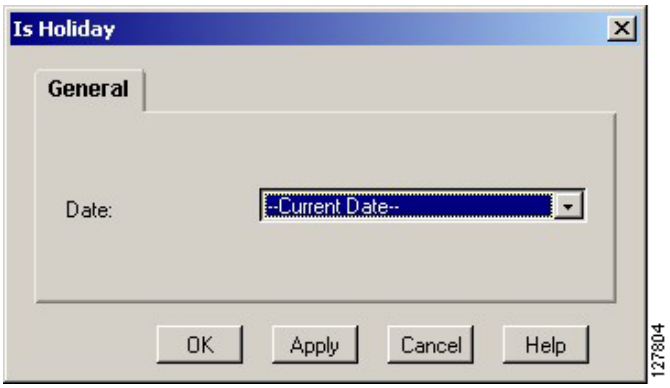
Configure the fields in the tabs of the Play Prompt customizer window as follows:

- **General** tab

  - **Contact**—Triggering contact: The contact that triggered the script remains the contact for this step.
  - **Interruptible**—No: No external events can interrupt the playback of the prompt.
- **Prompt** tab
  - **Prompt**—welcomePrompt: This prompt plays back to greet the caller.
  - **Barge In**—No: The caller must listen to the whole prompt before responding.
  - **Continue on Prompt Errors**—Yes: If a prompt error occurs, the script continues to play the next prompt, or, if this is the last prompt in the sequence, the script waits for caller input.
- **Input** tab
  - **Flush Input Buffer**— Yes: This step erases previous input.

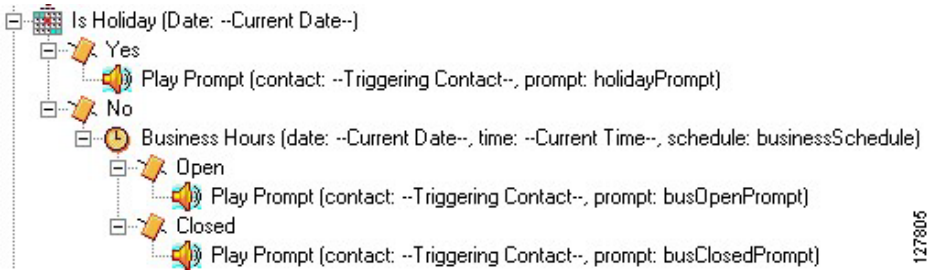
Step 8 Insert the **Is Holiday** step. By default, this step checks if the current day is a holiday, as shown in Figure 14, so no configuration is necessary for this step.

Figure 14 Is Holiday Customizer Window—Configured General Tab



The **Is Holiday** step contains two output branches: **Yes** and **No**, as shown in Figure 15.

Figure 15 Is Holiday Step Output Branches

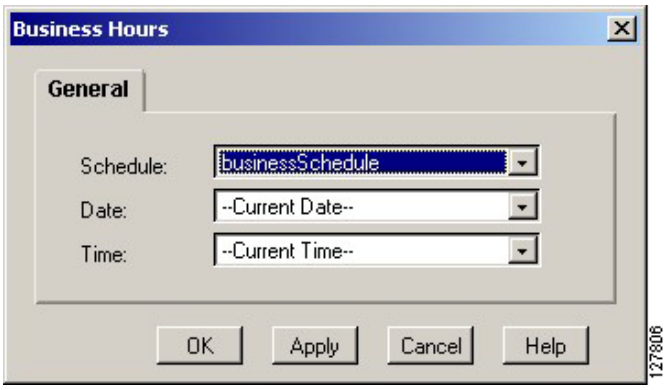


If the current day is a holiday, the script executes the **Yes** output branch. The script plays the holiday prompt and does not check the business hours. The script skips the **No** output branch and executes the MainMenu **Label** step.

If the current day is not a holiday, the script executes the **No** output branch. The script determines if the business is currently open or not, using the **Business Hours** step.

Step 9 Insert the **Business Hours** step under the **No** output branch of the **Is Holiday** step. This step determines if the business is currently open or not, as specified by the Business Hours Schedule configured for this step, as shown in Figure

Figure 16 Business Hours Customizer Window—Configured General Tab

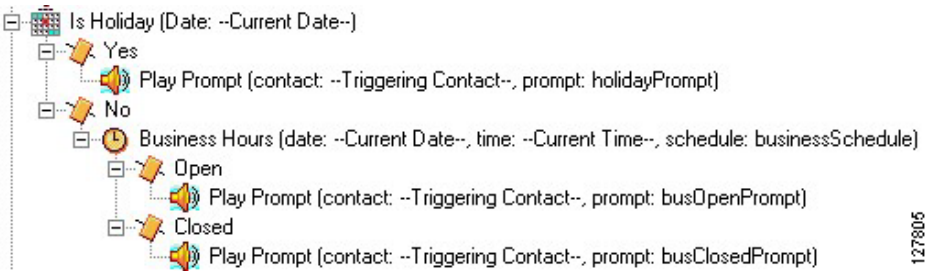


Configure the fields in the customizer window as follows:

- **Schedule**—Name of the Business Hours Schedule file. Use one of the schedules you created or customized using the Cisco Unity Express GUI options or CLI commands.
- **Date**—No configuration is necessary. By default, the script uses the current date.
- **Time**—No configuration is necessary. By default, the script uses the current time

The **Business Hours** step contains two output branches: **Open** and **Closed**, as shown in Figure 17.

Figure 17 Business Hours Step Output Branches



If the business is currently open, the script executes the **Open** output branch and plays the business open prompt. By default, the prompt AABusinessOpen.wav is empty. If the business is currently closed, the script executes the **Closed** output branch and plays the business closed prompt.

After executing either step, the script executes the MainMenu **Label** step.

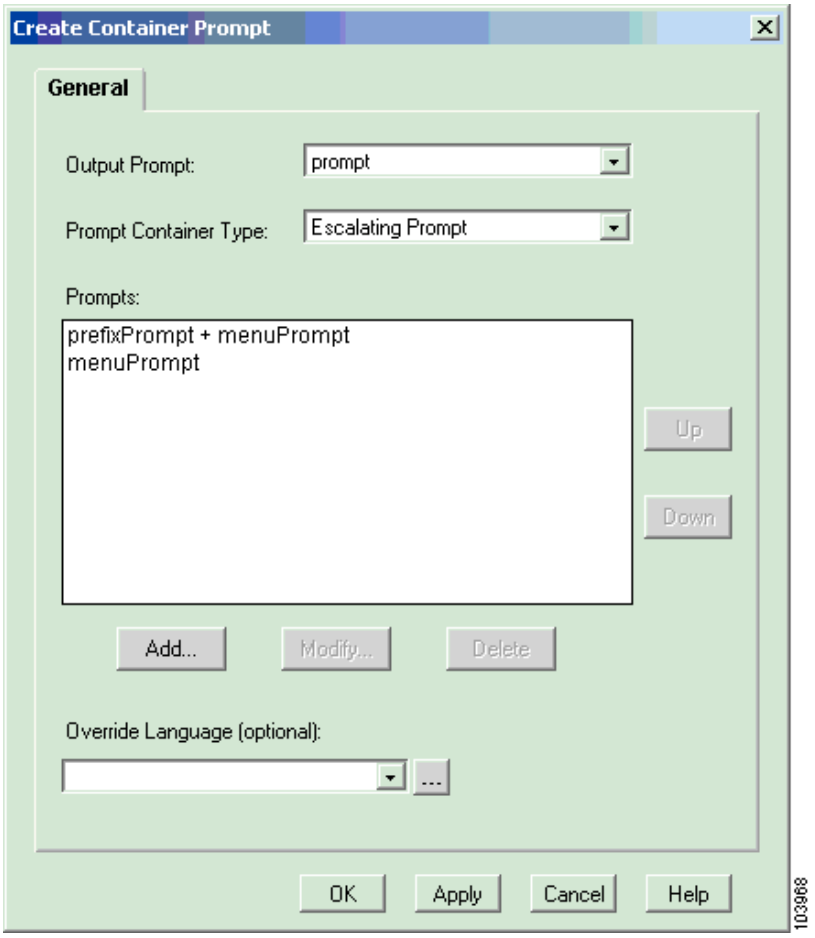
Step 10 Insert the **Label** step. This step creates a target for the script. This target is used later in the script when an output branch reaches a timeout, unsuccessful, or otherwise dead-end position and the script returns the caller to the **Label** step to try again.The **Label** step is named **MainMenu**.

Configure the Main Menu Step

Perform the following tasks to configure the **Main Menu Label** step that you just created:

Step 1 Create the **Container Prompt** step. This step, as shown in Figure 18, creates an escalating prompt called **prompt**, which combines **menuPrompt** (the first prompt created in the scriptStep 3) with a new prompt called **prefixPrompt**. The **prefixPrompt** variable initializes with no value, but as the caller loops through the application and fails to be connected to a destination, this variable holds an error message to be played back to the caller. The prompt is an escalating prompt so that the error message plays only on the first attempt of the subsequent **Menu** step, which uses the prompt created by this step.

Figure 18 Create Container Prompt—Configured General Tab



Configure the fields in the tabs of the **Create Container Prompt** window as follows:

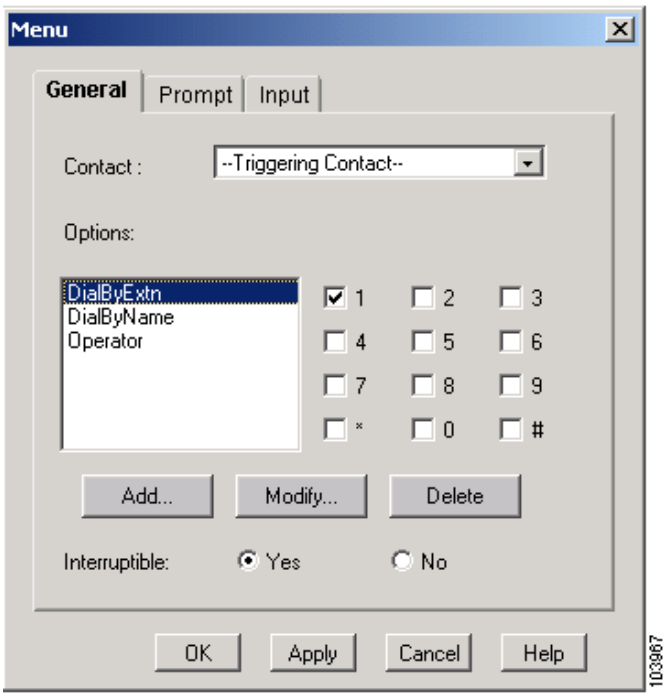
- **Output Prompt**—prompt: This prompt results from this **Create Container Prompt** step.
- **Prompt Container Type**—escalating: This step creates an escalating prompt.
- **Prompts List Box**
  - **prefixPrompt + menuPrompt**
  - **menuPrompt**

  This specifies the prompt phrases that are played if the **Media** step uses more than one attempt at eliciting a valid response from the caller.

Step 2 Insert the **Prefix Prompt Set** step. This step sets the value of **prefixPrompt** to P[], which means it is empty. When callers are returned to the **MainMenu** Label step to listen to menu options again, this **Set** step clears **prefixPrompt** of values that the script may have previously assigned to it. **Call Redirect** steps often return callers to the **MainMenu** Label. See Figure 28 on page 45 for an example.

Step 3 Insert the **Menu** step. The **Menu** step receives either speech or digits in response to prompts, as shown in Figure 19. The script procedures configured under the **Menu** step transfer the call to the proper extension if the script receives valid input from the caller.

Figure 19 Menu Customizer Window—Configured General Tab



The **Menu** customizer window contains the following values:

- **General** tab
  - **Contact**—Triggering Contact: The contact that triggered this script remains the contact for this step.
  - **Options**—DialByExtn, DialByName, Operator: The list of options the menu offers to the caller. The tags map to the output points to determine the execution of branching output paths. **DialByExtn** is mapped to dial keypad 1. **DialByName** is mapped to dial keypad 2. **Operator** is mapped to dial keypad 0.
  - **Interruptible**—Yes: External events can interrupt the execution of this step.
- **Prompt** tab
  - **Prompt**—Prompt: The step plays this prompt back to the caller.
  - **Barge In**—Yes: The caller can respond without first having to listen to the playback of the entire prompt.
  - **Continue on Prompt Errors**—Yes: If a prompt error occurs, the script continues to play the next prompt, or, if this is the last prompt in the sequence, the script waits for caller input.

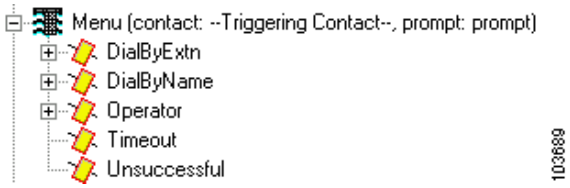
- **Input** tab
  - **Timeout** (in sec)—5: After playing all prompts, the script waits 5 seconds for initial input from the caller before re-attempting with a timeout error, or, if this was the last attempt, the script executes the Timeout output branch.
  - **Maximum Retries**—5: The script will retry to receive input 5 times before sending the script to the Unsuccessful output branch.
  - **Flush Input Buffer**—No: The script saves previous input.

Configure the Menu Step Output Branches

The **Menu** step contains two built-in output branches: **Timeout** and **Unsuccessful**. The **Timeout** and **Unsuccessful** output branches need no scripting. If the script reaches either of these branches, it proceeds to the next step on the same level as the **Menu** step, the second **Play Prompt** step (see "Configure the Play Prompt Step" on page 56).

You must configure three output branches, as shown in Figure 20. These branches correspond to the three choices **menuPrompt** gives the caller: dial by extension, dial by name, or dial the operator.

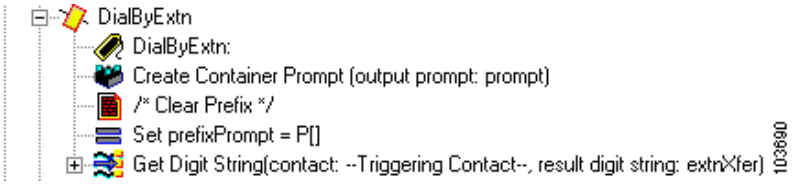
Figure 20 Menu Step Output Branches



Perform the following tasks to configure the **Menu** step output branches:

Step 1 Configure the **DialByExtn Output Branch**. If the caller chooses menu option "1" (presses an extension number) when given the option by the **Menu** step, the script executes the **DialByExtn** output branch. The **DialByExtn** output branch of the **Menu** step receives the extension number provided by the caller, as shown in Figure 21

Figure 21 DialByExtn Scripting



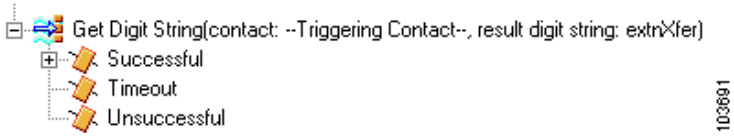
The **DialByExtn** output branch contains the following steps:

- **Label** step (DialByExtn): Creates a target **DialByExtn**.
- **Create Container Prompt** step: Creates a concatenated container prompt, consisting of **prefixPrompt** (see page 37) and **extnPrompt**, a preset prompt that prompts the caller to enter the extension number (with a possible error message when looping back if there is an error connecting to the destination).

- **Set** step: Sets the value of **prefixPrompt** to P[], which clears **prefixPrompt** of any values that the script may have previously assigned.
- **Get Digit String** step: Receives the digits entered by the caller in response to **prompt**, stores them in a result digit string variable named **extnXfer**, and then attempts to transfer the call.

The **Get Digit String** has three output branches: **Successful**, **Timeout**, and **Unsuccessful**, as shown in Figure 22.

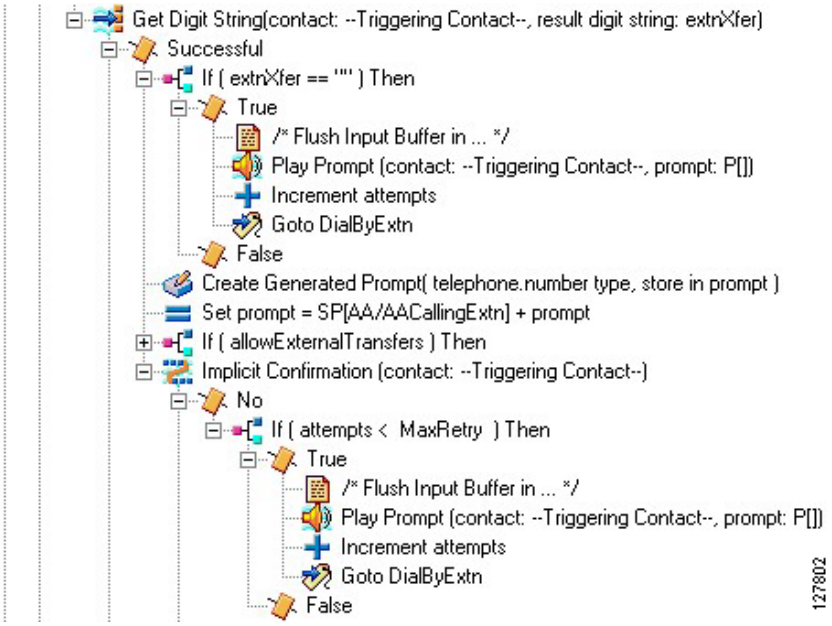
Figure 22 Get Digit String Output Branches



The following are the three output branches of the **Get Digit String** step:

- **Timeout Output** branch: If the **Get Digit String** step does not receive input before reaching the timeout limit, the script executes the **Timeout** output branch, and the script skips the rest of the output branches of the **Menu** step and proceeds to the second **Play Prompt** step (see page 56).
- **Unsuccessful Output** branch: If the **Get Digit String** step is unsuccessful in receiving valid input, the script executes the **Unsuccessful** output branch, and the script skips the rest of the output branches of the **Menu** step and proceeds to the second **Play Prompt** step (see page 56).
- **Successful Output** branch: If the **Get Digit String** step successfully receives caller input, the script executes the **Successful** output branch. The **Successful** output branch transfers the call, as shown in Figure 23.

Figure 23 Get Digit String—Successful Branch Scripting



Note See Figure 28 for the diagram of the **Implicit Confirmation Yes** branch steps.

Step 2 Configure the **If** step for the get digit string successful branch. The **If** step checks if the extension entered (**extnXfer**) is empty or not by evaluating the expression (**extnXfer == “”**). The If step has two output branches:

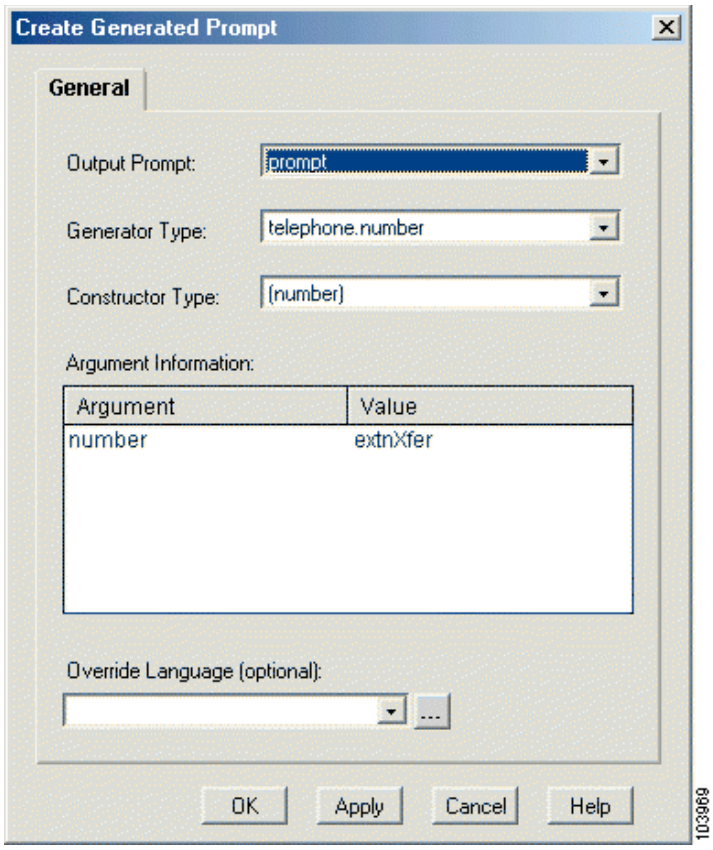
- **True** output branch: If the **If** step determines that the extension entered is empty, the script executes the **True** output branch. The **True** output branch increments the number of attempts by one and gives the caller another try. The **True** output branch of the **If** step contains three functional steps:
  - **Play Prompt** Step: The **Play Prompt s**tep plays an empty prompt, P[]**,** to flush the DTMF buffer of any digits that the script may have accumulated as part of the previous **Get Digit String** step.

Note The **Play Prompt** step has an empty prompt to enable the script to return immediately from this step after flushing out the buffer.

  - **Increment** step: The **Increment** step increases the number of attempts until the maximum number of retries is reached.
  - **Goto** step: The **Goto** step returns the caller to the beginning of the **DialByExtn** Label step at the beginning of the **DialByExtn** output branch in order to give the caller more attempts to input the proper extension.
- **False** output branch: If the **If** step determines that the caller has entered an extension, the script executes the **False** output branch. The **False** output branch of the **If** step proceeds to the next step (**Create Generated Prompt** step).

Step 3 **Create Generated Prompt** step for the get digit string successful branch. This step, as shown in Figure 24, creates a prompt to play back to the caller the digits received, in order to confirm the caller input before transferring the call.

Figure 24 Configured Create Generated Prompt Customizer Window



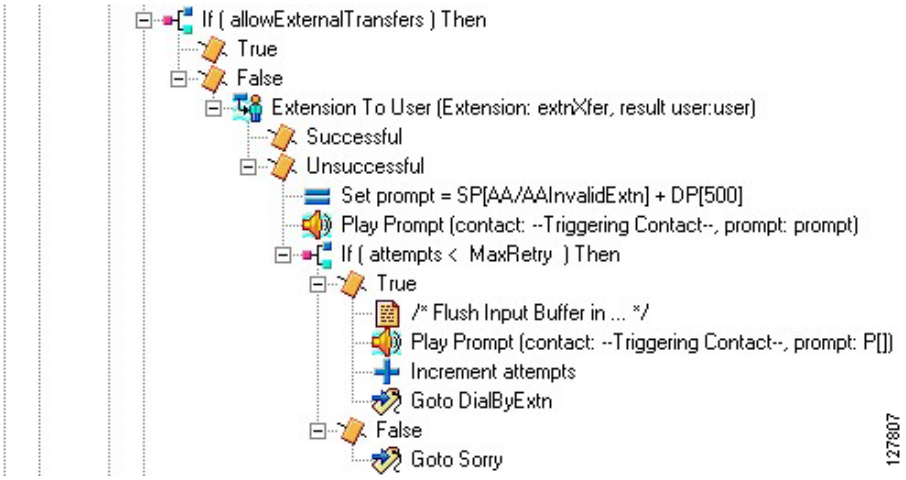
The Create Generated Prompt customizer window contains the following values:

- **Output Prompt**—prompt: Stores the value that results from this step.
- **Generator Type**— telephone.number: Generator type. (See page 114.)
- **Constructor Type**—number: Constructor type. (See page 114.)
- Argument Information list box—**extnXfer**: The **extnXfer** variable stores the results of the number constructor.

Step 4 Configure the **If** step to check if transfer to external numbers (**allowExternalTransfers**) is allowed.

The **If** step has two output branches: **True** and **False**, as shown in Figure 25.

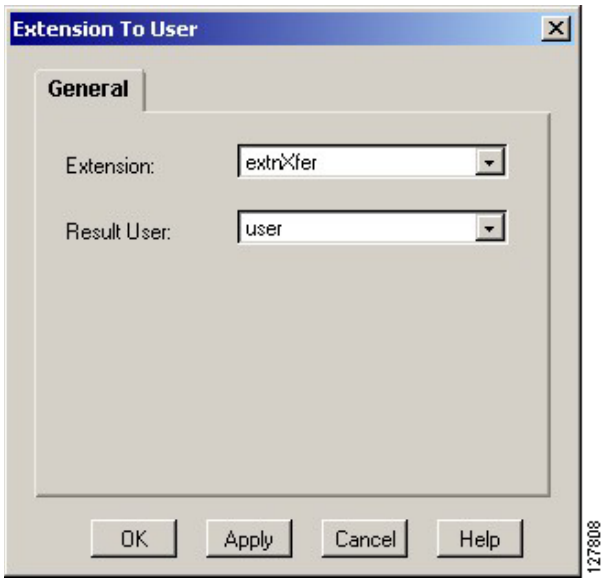
Figure 25 If Step Output Branches



- If transfer to external numbers is allowed, the script executes the **True** output branch. No configuration is required under this branch, and the script continues to the **Implicit Confirmation** step.
- If transfer to external numbers is not allowed, the script executes the **False** output branch. This branch uses the **Extension To User** step to determine if the number entered by the caller is a valid extension of a local user.

Step 5 Configure the **Extension To User** step under the **False** output branch of the **If** step, as shown in Figure 26. The **Extension To User** step allows the script to find a user based on the extension entered by the caller.

Figure 26 Extension To User Window—Configured General Tab



Configure the fields in the customizer window as follows:

- **Extension**—The extension entered by the caller. Use the variable **extnXfer** returned by the **Get Digit String** step.

- **Result User**—The user variable that stores the User object that maps to the extension entered by the caller.

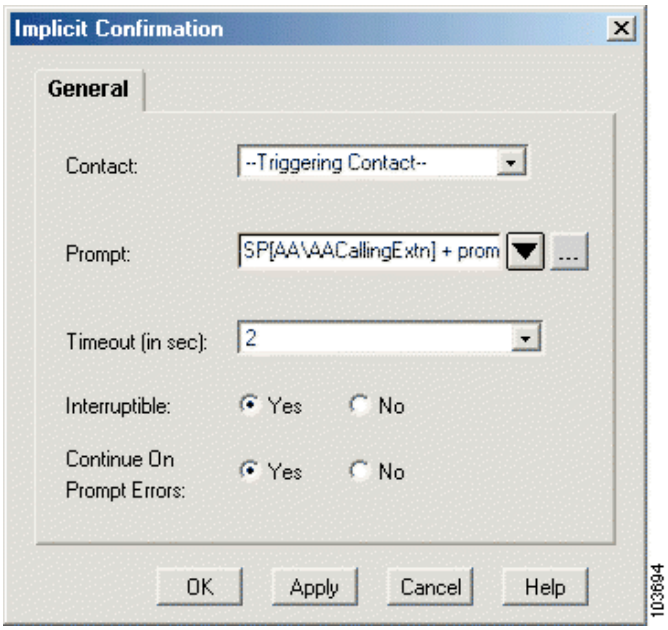
The **Extension To User** step contains two output branches: **Successful** and **Unsuccessful**, as shown in Figure 25.

- If the **Extension To User** step finds a user with the matching extension in the local directory, the script executes the **Successful** output branch. The script continues to the **Implicit Confirmation** step.
- If the **Extension To User** step does not find a matching extension, the script executes the **Unsuccessful** output branch. The script plays a prompt indicating that an invalid extension has been entered.

  The script then uses an **If** step to try again until the maximum number of retries is reached. If the maximum number of retries has not been reached, the script uses the **Goto** step to return the caller to the beginning of the **DialByExtn** output branch to give the caller more attempts to input the correct extension. If the maximum number of retries has been reached, the script uses the **Goto** step to take the caller to the **Sorry Labe**l step at the end of the script.

Step 6 Create **Implicit Confirmation** step for the **Get Digit String Successful** branch. This step confirms the extension entered without requiring more input from the caller, as shown in Figure 27.

Figure 27 Configured Implicit Confirmation Customizer Window



The Implicit Confirmation customizer window contains the following values:

- **Contact**—Triggering Contact: The contact that triggered the script remains the contact for this step.
- **Prompt**—SP[AA/AACallingExtn] + prompt: The system prompt and the generated prompt indicate the specified extension the script plays back to the caller.
- **Timeout** (in sec)— 2: The caller has 2 seconds to stop the transfer before the script accepts the confirmation and transfers the call.
- **Interruptible**—Yes: External events are allowed to interrupt the execution of this step.

- **Continue on Prompt Errors**—Yes: In the event of a prompt error, the script will play the next prompt in the sequence, or if this is the last prompt, will wait for caller input.

Configure the Implicit Confirmation Step

The **Implicit Confirmation** step has two output branches:

- **No** output branch: If the caller interrupts the **Implicit Confirmation** step and does not give confirmation, the script executes the **No** output branch. The **No** output branch of the **Implicit Confirmation** step uses an **If** step to try again, until the maximum number of retries is reached. The **If** step determines whether or not the maximum number of retries has been reached by evaluating the expression **attempts < MaxRetry**. The expression determines if the number of attempts, as stored in the **Attempts** variable, is less than the maximum number of retries, as stored in the **MaxRetry** variable.

  The **If** step has two output branches:

  – **True** output branch: If the **If** step determines that the maximum number of retries has not been reached, the script executes the **True** output branch. The **True** output branch increments the number of retries by one and gives the caller another try. The **True** output branch of the **If** step contains three functional steps:

  1. **Play Prompt** step: The **Play Prompt** step plays an empty prompt, P[] to flush the DTMF buffer of any digits that the script may have accumulated as part of the previous Implicit Confirmation step.

  Note The **Play Prompt** step has an empty prompt to enable the script to return immediately from this step after flushing out the buffer.

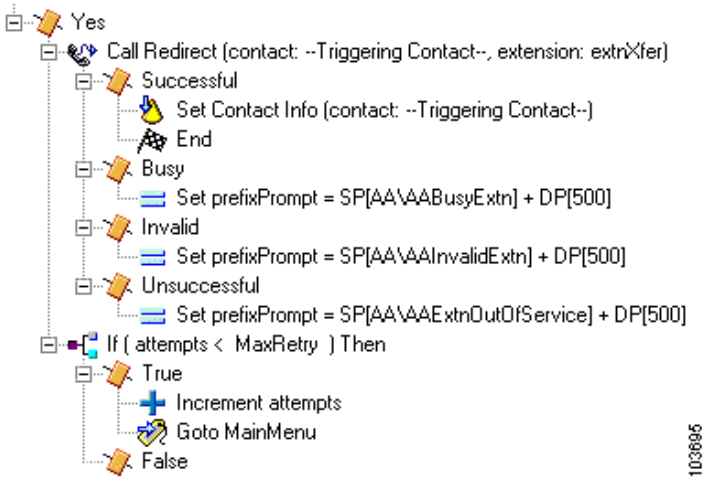
  2. **Increment** step: The **Increment** step increases the number of attempts until the maximum number of retries is reached.
  3. **Goto** step: The **Goto** step returns the caller to the beginning of the **DialByExtn Label** step at the beginning of the **DialByExtn** output branch in order to give the caller more attempts to input the proper extension.

  – **False** output branch: If the **If** step determines that the maximum number or retries has been reached, the script executes the **False** output branch.The **False** output branch of the **If** step skips the rest of the steps under the **Menu** step and proceeds to the second **Play Prompt** step (see page 56).

- **Yes** output branch: If the **Implicit Confirmation** step successfully confirms the extension, the script executes the **Yes** output branch. The **Yes** output branch of the **Implicit Confirmation** step transfers the call.The **Yes** output branch contains the following steps:

  – **Call Redirect** step for the **Yes** output branch: As in the other two main output branches of the **Menu** step (**DialByName** and **Operator**), the **DialByExtn** output branch contains the **Call Redirect** step, which attempts to transfer the call, in this case to the desired extension number. The **Call Redirect** step has four output branches, as shown in Figure 28:

Figure 28 Call Redirect Output Branch Scripting



1. **Successful** output branch for the call redirect step: If the **Call Redirect** step successfully transfers the call, the script executes the **Successful** output branch. The **Successful** output branch of the **Call Redirect** step marks the contact as **Handled** and ends the script. The **Successful** output branch of the **Call Redirect** step contains two steps. The **Set Contact Info** step marks the call as **Handled**. The **End** step ends this branch of the script.
2. **Busy** output branch for the **Call Redirect** step: If the **Call Redirect** step registers the destination extension as busy, the script executes the **Busy** output branch. The **Busy** output branch of the **Call Redirect** step sets the value of the **prefixPrompt** variable to inform the caller that the extension was busy. The **Busy** output branch of the **Call Redirect** step contains the **Set** step. The **Set** step, as shown in Figure 28, sets the value of **prefixPrompt** to contain a system prompt that plays back a message to the caller that the extension is busy.
3. **Invalid** output branch for the call redirect step: If the **Call Redirect** step registers the destination extension as invalid, the script executes the **Invalid** output branch. The **Invalid** output branch of the **Call Redirect** step sets the value of the **prefixPrompt** variable to inform the caller that the extension was invalid.The **Invalid** output branch of the **Call Redirect** step contains the **Set** step. The **Set** step, as shown in Figure 28, sets the value of **prefixPrompt** to contain a system prompt that plays back a message to the caller that the extension is busy.
4. **Unsuccessful** output branch for the call redirect step: If the **Call Redirect** step registers the destination extension as out of service, the script executes the **Unsuccessful** output branch. The **Unsuccessful** output branch of the **Call Redirect** step sets the value of the **prefixPrompt** variable to inform the caller that the extension was out of service. The **Unsuccessful** output branch of the **Call Redirect** step contains the **Set** step. The **Set** step, as shown in Figure 28, sets the value of **prefixPrompt** to contain a system prompt that plays back a message to the caller that the extension is busy.

– **If** step for the **Yes** output branch: The **If** step allows the script to determine whether or not the maximum number of retries has been reached by evaluating the expression **attempts < MaxRetry**. This expression determines if the number of attempts, as stored in the attempts variable, is less than the maximum number of retries, as stored in the MaxRetry variable. The If step has two output branches, **True** and **False**:

  True: If the **If** step determines that the maximum number of retries has not been reached, the script executes the **True** output branch. The **True** output branch of the **If** step allows the caller to keep returning to the **MainMenu** label until it reaches the maximum number of retries.

The **True** output branch contains two steps. The Increment step increases the numbers or retries by 1. The **Goto** step sends the caller back to the **MainMenu** label step to provide the caller another opportunity to enter an extension.

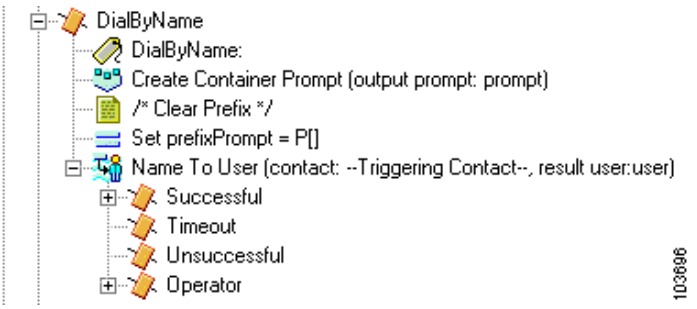
False: If the **If** step determines that the maximum number of retries has been reached, the script executes the **False** output branch.The **False** output branch of the **If** step proceeds to the second **Play Prompt** step (see page 56).

Configure the Menu Step Output Branches (continued)

Perform the following tasks to continue configuration of the **Menu** step output branches:

Step 1 Configure the **DialByName** output branch. If the caller chooses menu option "2" (presses to enter the name of a person) when given the option by the **Menu** step, the script executes the **DialByName** output branch. The **DialByName** output branch, as shown in Figure 29, receives the name of the person the caller desires to reach.

Figure 29 DialByName Output Branch Scripting



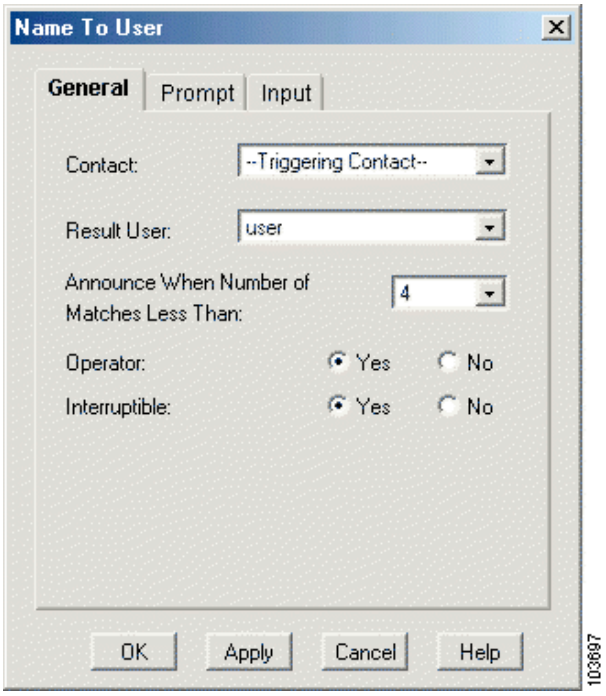
Step 2 Configure the **Label** step (**DialByName**). The **Label** step is named **DialByName** to provide a target for the script so that the caller has more opportunities, if necessary, to enter a name successfully.

Step 3 Configure the create **Container Prompt** step. The **Create Container Prompt** step creates a prompt that asks the caller to enter the name of the desired person.

Step 4 Configure the **Set** Step. Configure the **Set** step to clear the value of the **prefixPrompt** variable so that it can be assigned by subsequent steps.

Step 5 Configure the **Name To User** step. The **Name To User** step, as shown in Figure 30, allows the caller to find a user based on DTMF digits input from the caller.

Figure 30 Name To User Customizer Window—Configured General Tab



The **Name To User** customizer window contains the following values:

- **General** tab
  - **Contact**—Triggering Contact: The contact that triggered the script remains the contact for this step.
  - **Result User**—user: The user variable stores the user object that maps to the selection of the caller.
  - **Announce When Number of Matches Less Than**—4: If the number of matches is less than 4, the script prompts the caller to choose the correct entry from the list of matches. If the number of matches is greater than or equal to 4, the script prompts the caller to enter additional letters to reduce the number of matches.
  - **Operator**—Yes: The script gives the caller the option to connect to an operator by pressing "0".
  - **Interruptible**—Yes: External events can interrupt the playback of the prompt.
- Prompt tab
  - **Prompt**—Customize Prompt: The script uses a customized prompt.
  - **Prompt**—prompt: The prompt variable stores the custom prompt the script plays back to the caller.
  - **Barge In**—Yes: The caller can respond without first having to listen to the playback of the entire prompt.
  - **Continue On Prompt Errors**—Yes: If a prompt error occurs, the script continues to play the next prompt, or, if this prompt is the last in the sequence, the script waits for caller input.
- **Input** tab

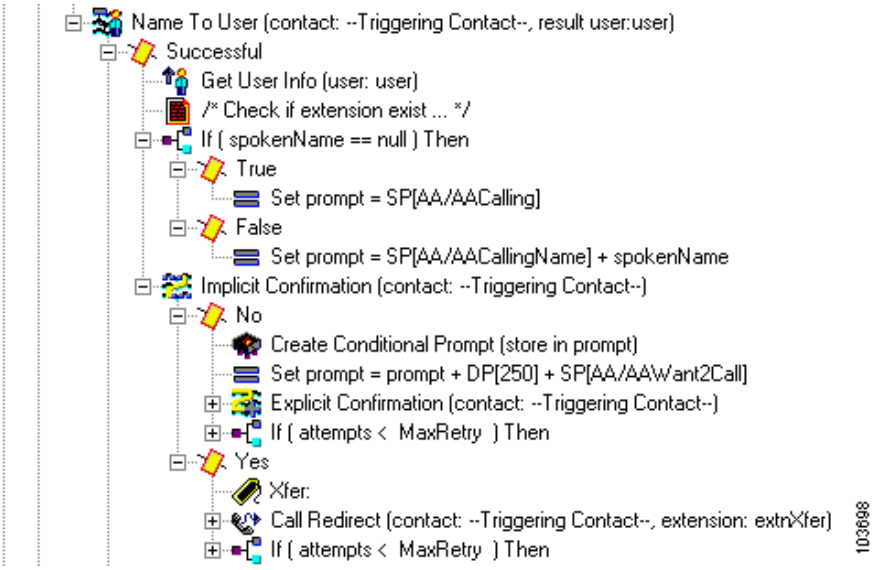
- **Input Length**—30: Specifies that the script automatically triggers a lookup when the caller enters 30 digits.
- **Terminating Key**—#: The terminating key is "#".
- **Cancel Key**—*: The cancel key is "*".
- **Maximum Retries**—5: The maximum number of retries is 5.
- **Initial Timeout** (in sec)—5: The step times out if the script receives no input within 5 seconds after playing back the prompt.
- **Interdigit Timeout** (in sec)—3: The step times out if the script receives no input between digits for 3 seconds.
- **Flush Input Buffer**—No: The script saves input previously entered by the caller.

The **Name To User** step has four output branches: **Successful**, **Timeout**, **Unsuccessful**, and **Operator**. The **Timeout** and **Unsuccessful** output branches need no scripting. If the step times out, the script proceeds to the second **Play Prompt** step (see page 56). If an invalid entry is made after 5 attempts, the script also proceeds to the second **Play Prompt** step (see page 56).

Perform the following tasks to configure the output branches for the **Name to User** step:

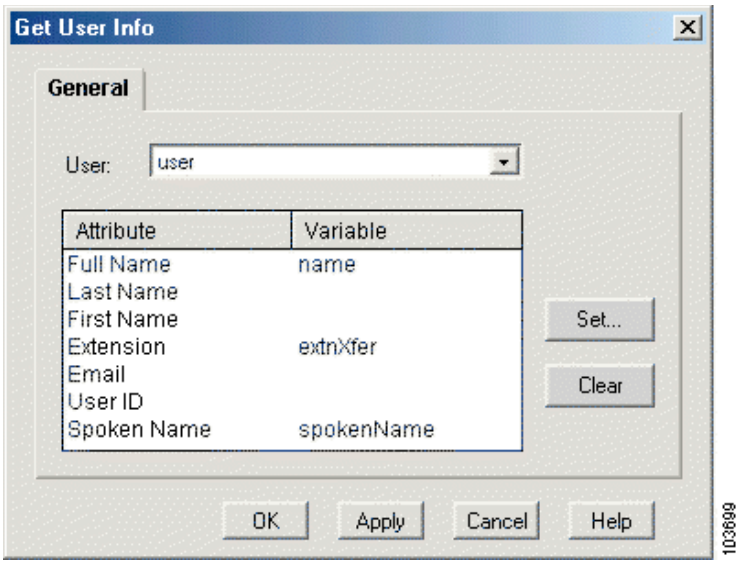
Step 1 Configure the **Successful** output branch for the **Name to User** step. The **Successful** output branch of the **Name To User** step, as shown in Figure 31, receives confirmation of the name from the caller and to transfer the call. The steps under this branch are similar to the steps under the **Successful** output branch of the **Get Digit String** step above (See page 40): the script requests confirmation and redirects the call to the desired extension.

Figure 31 Name To User—Successful Output Branch Scripting



Step 2 Configure the **Get User Info** step for the **Name to User** successful output branch. The **Get User Info** step, as shown in Figure 32, makes user attributes available to the script.

Figure 32 Configured Get User Info Customizer Window



The **Get User Info** customizer window contains the following values:

- **User**—user: Specifies user as the variable that holds a handle to the user information selected by the Name To User step.
- **Attribute/Variable** text box
  - **Full Name**—name
  - **Extension**—extnXfer
  - **Spoken Name**—spokenName

Step 3 Configure the **if** step for the **Name to User** successful output branch. The **If** step creates a prompt based on whether or not a recording of the spoken name of the person whose extension is being called is available. The **If** step evaluates the Boolean expression **spokenName==null**. This expression determines if the value of the **Document** variable **spokenName** is equal to null.

Step 4 Configure the **True** output branch for the **Name to User Successful If** step. If the **If** step determines that a recording of the spoken name of the person whose extension is being called is not available, the script executes the **True** output branch. The **True** output branch of the **If** step instructs prompt to play the system prompt **SP[AA/AACalling]**, which does not play the name of the person being called.

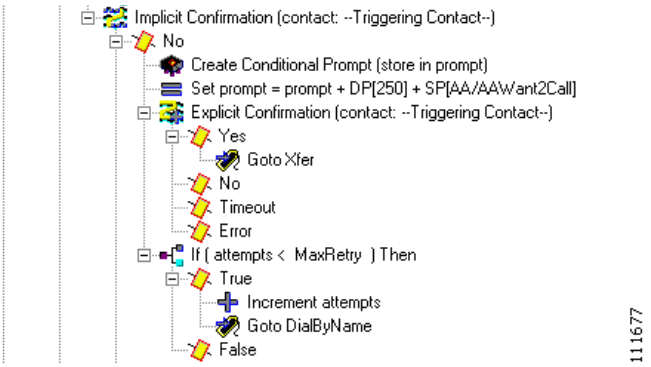
Step 5 Configure the **False** output branch for the **Name to User Successful If** step. If the **If** step determines that a recording of the spoken name of the person whose extension is being called is available, the script executes the **False** output branch. The **False** output branch of the **If** step instructs prompt to play the system prompt **SP[AA/AACallingName],** which is then followed by the spoken name.

Step 6 Configure the **Implicit Confirmation** step for the **Name to User** successful output branch. The **Implicit Confirmation** step confirms the name entered without demanding more input from the caller. The **Implicit Confirmation** step performs in a similar way to the **DialByExtn** section above. (See page 44.)

The **Implicit Confirmation** step has two output branches (see Figure 31.). To configure the output branches for the **Implicit Confirmation** step, perform the following tasks:

Step 1 Configure the **No** output branch for the **Implicit Confirmation** step. If the **Implicit Confirmation** step does not successfully confirm the choice of the caller, the script executes the **No** output branch. The **No** output branch of the **If** step, as shown in Figure 33, creates a prompt that provides the caller an opportunity to explicitly confirm the choice.

Figure 33 Name To User—No Output Branch of Implicit Confirmation Step

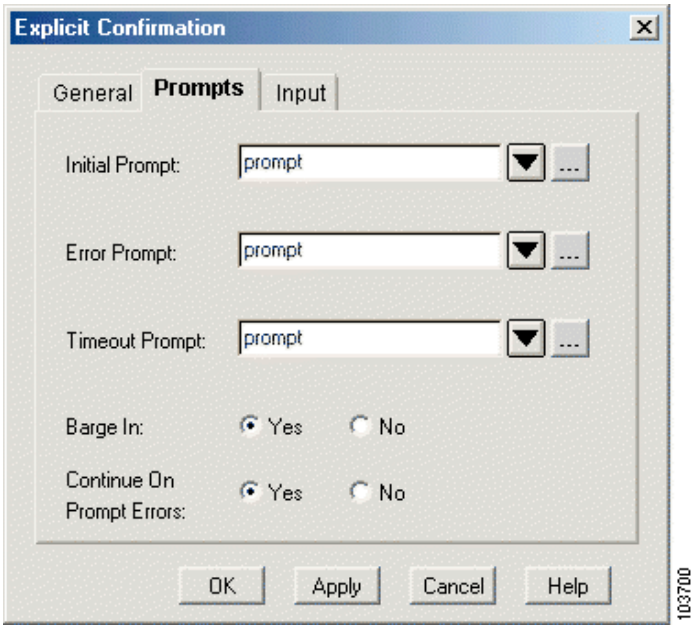


Step 2 Create the **Conditional Prompt** step for the **No** branch. The **Create Conditional Prompt** step creates a prompt based on whether or not the variable **spokenName** is null. The **spokenName** variable is not null if a spoken name exists for the selected user in the directory.

Step 3 Create the **Set** step for the **No** branch. The **Set** step appends the prompt created by the **Create Conditional Prompt** step with the system prompt **SP[AA/AAWantToCall]**.

Step 4 Create the **Explicit Confirmation** step for the **No** branch. The **Explicit Confirmation** step makes an explicit confirmation of the name of the desired person, as shown in Figure 34.

Figure 34 Explicit Confirmation Customizer Window—Configured Prompt Tab



The **Explicit Confirmation** customizer window contains the following values:

- **General** tab
  - **Contact**—Triggering Contact: The contact that triggered the script remains the contact for this step.
  - **Interruptible**—Yes: External events can interrupt the playback of the prompt.
- **Prompt** tab
  - **Initial Prompt**—prompt: The prompt variable stores the first prompt.
  - **Error Prompt**—prompt: The prompt variable plays in the event of an input error.
  - **Timeout Prompt**—prompt: The prompt variable plays if the timeout limit is reached.
  - **Barge In**—Yes: The caller can respond without first having to listen to the playback of the entire prompt.
  - **Continue on Prompt Errors**—Yes: If a prompt error occurs, the script continues to play the next prompt, or, if this is the last prompt in the sequence, the script waits for caller input.
- **Input** tab
  - **Timeout (in sec)**—5: After playing all prompts, the script waits 5 seconds for initial input from the caller before re-attempting with a timeout error, or, if this was the last attempt, the script executes the Timeout output branch.
  - **Maximum Retries**—3: The script will attempt a maximum of 3 retries to receive confirmation before executing the Unsuccessful output branch.
  - **Flush Input Buffer**—Yes: The step erases previous input.
  - **Grammar**—grammar: Leave blank.

The **Explicit Confirmation** step has four output branches: **Yes**, **No**, **Timeout**, and **Error**.

Configure the **Yes** output branch for the **Explicit Confirmation** step. If the **Explicit Confirmation** step successfully receives confirmation from the caller, the script executes the **Yes** output branch. The **Yes** output branch of the **Explicit Confirmation** step directs the script to the **Xfer Label** step under the **Yes** output branch of the **Implicit Confirmation** step (see page 53), which contains the steps necessary to redirect the call to the desired extension.

Step 5 Configure the **No** output branch for the **Explicit Confirmation** step. The **No** output branch contains the **If** Step. The **If** step determines whether or not the maximum number of retries has been reached by evaluating the expression **attempts < MaxRetry**. This expression determines if the number of attempts (as stored by the attempts variable) is less than the maximum retries value stored in the **MaxRetry** variable. If the **If** step determines that the maximum number of retries has not been reached, the script executes the **True** output branch. The **True** output branch of the **If** step provides the caller with another opportunity to enter the name of the desired person.

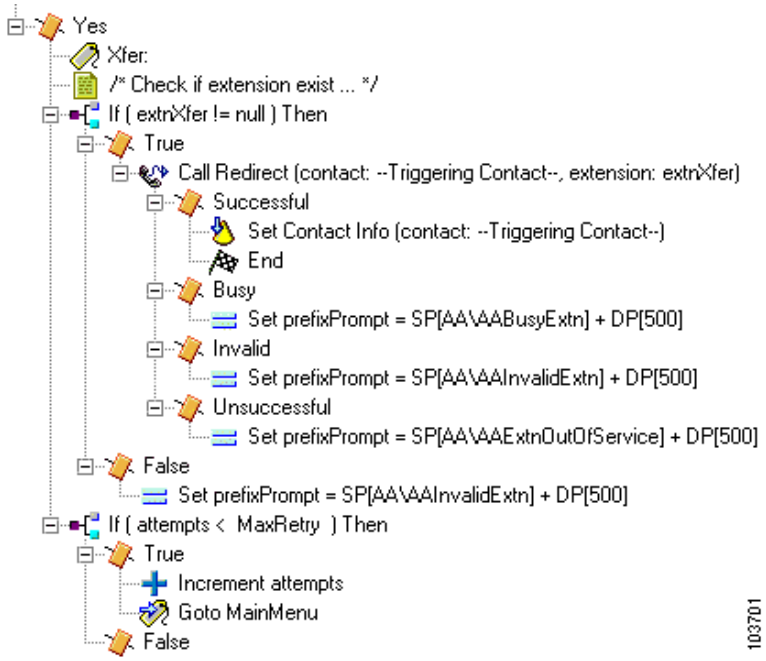
Step 6 Configure the **Increment** step of the **True** output branch. The **Increment** step increases the value of the attempts variable by 1.

Step 7 Configure the **Goto** step of the True output branch. The **Goto** step returns the caller to the beginning of the **DialByName Label** step at the beginning of the **DialByName** output branch in order to give the caller more attempts to input the proper name.

As stated in Step 5 above, the **If** step for the **No** output branch determines if the maximum number of retries has not been reached. If the number has been reached, the script executes the **False** output branch. The script proceeds to the second **Play Prompt** step (see page 56).

After completing configuration for the **No** output branch of the **Implicit Confirmation** step, you must configure the **Yes** output branch. The **Yes** output branch of the **Implicit Confirmation** step redirects the call to the desired extension, as shown in Figure 35.

Figure 35 Name To User—Yes Output Branch of Implicit Confirmation Step



To configure the **Yes** output branch of the **Implicit Confirmation** step, perform the following tasks:

Step 1 Create a **Label** step for the **Implicit Confirmation Yes** branch. The **Label** step has the value **Xfer** that provides a target for the **Yes** output branch of the **Explicit Confirmation** step above (see page 52).

Step 2 Create the first **If** step for the **Implicit Confirmation Yes** branch. The first **If** step directs the script based on whether or not the desired extension exists by evaluating the expression **extnXfer != null**. The expression determines if the value of the **extnXfer** variable (which stores the extension number) is not null.

Step 3 Configure the **False** output branch for the first **If** step. If the **If** step finds no extension for the selected user, the script executes the **False** output branch, which has one step, the **Set** step. The **Set** step sets the value of a prompt that will be played back to inform the caller that the extension was invalid.

Step 4 Configure the **True** output branch for the first **If** step. If the **If** step evaluates the desired extension as valid, the script executes the **True** output branch. The **True** output branch of the **If** step transfers the call using the **Call Redirect** step.

As in the other two main output branches of the **Menu** step (**DialByExtn** and **Operator**), the **DialByName** output branch contains the **Call Redirect** step, which attempts to transfer the call, in this case to the desired extension number. To configure the **Call Redirect** step, perform the following tasks:

Step 1 Configure the **Successful** output branch for the **Call Redirect** step. The **Successful** output branch of the **Call Redirect** step uses the **Set Contact Info** step to mark the contact as **Handled** and the **End** step to end the script.

Step 2 Configure the **Busy** output branch for the **Call Redirect** step. The **Busy** output branch of the **Call Redirect** step uses the **Set** step to set the value of the **prefixPrompt** variable to contain a system prompt that will play back a message to the caller that the extension is busy when the script proceeds to the final **Play Prompt** step (see page 57).

Step 3 Configure the **Invalid** output branch for the **Call Redirect** step. If the **Call Redirect** step registers the destination extension as invalid, the script executes the **Invalid** output branch. The **Invalid** output branch of the **Call Redirect** step uses the **Set** step to set the value of the **prefixPrompt** variable to contain a system prompt that will play back a message to the caller that the extension is invalid when the script proceeds to the final **Play Prompt** step (see page 57).

Step 4 Configure the **Unsuccessful** output branch for the **Call Redirect** step. If the **Call Redirect** step registers the destination extension as out of service, the script executes the **Unsuccessful** output branch. The **Unsuccessful** output branch of the **Call Redirect** step uses the **Set** step to set the value of the **prefixPrompt** variable to contain a system prompt that will play back a message to the caller that the extension is out of service when the script proceeds to the final **Play Prompt** step (see page 57).

After configuring the first **If** Step for the **Yes** output branch of the **Implicit Confirmation** step, you need to configure the second **If** step. Perform the following tasks:

Step 1 Create the second **If** step for the **Implicit Confirmation Yes** branch. The second **If** step directs the script based on whether or not the maximum number of retries has been reached by evaluating the expression **attempts < MaxRetry**. The expression determines if the number of attempts, as stored in the attempts variable, is less than the maximum number of retries allowed, as stored in the **MaxRetry** variable.

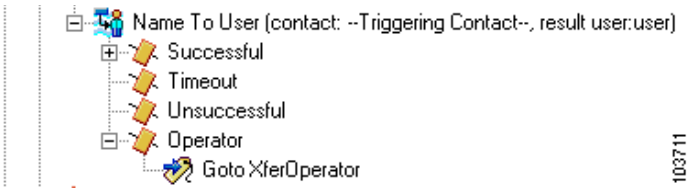
Step 2 Configure the **False** output branch for the **If** step. If the second **If** step finds that the maximum number of retries has been reached, the script executes the **False** output branch, and the script proceeds to the final **Play Prompt** step (see page 57).

Step 3 Configure the **True** output branch for the **If** step. If the **If** step finds that the maximum number of retries has not been reached, the script executes the **True** output branch. The **True** output branch of the **If** step provides more opportunities for the caller to successfully enter a name. The **Increment** step increases the value of the attempts variable by 1. The **Goto** step returns the caller to the beginning of the **DialByName Label** step at the beginning of the **DialByName** output branch of the **Get Digit String** step in order to give the caller more attempts to input the proper extension.

Configure the Name to User Step (continued)

To complete configuration of the output branches for the **Name to User** step, configure the Operator Output Branch for the Name to User Step. If the **Name to User** step receives caller input for transfer to the operator, the script executes the **Operator** output branch.The output branch transfers the call to an operator, as shown in Figure 36.

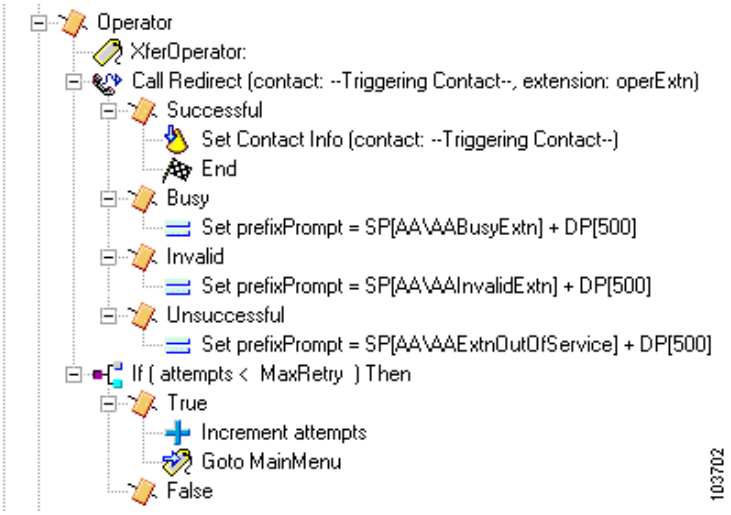
Figure 36 Name to User—Operator Output Branch



The **Operator** output branch uses the **Goto** step. to send the caller to the **Operator** output branch of the **Menu** step. If the caller chooses menu option "3" to speak to an operator when given the option by the **Menu** step, the script executes the **Operator** output branch.

The **Operator** output branch transfers the call to an operator, as shown in Figure 37.

Figure 37 Menu—Operator Output Branch



To configure the **Operator** output branch, perform the following tasks:

Step 1 Create the **Label** step (**Xfer Operator**) for the **Operator** branch. The **Label** step has the value **Xfer Operator** that provides a target for the **Goto** step.

Step 2 Create the **Call Redirect** step for the **Operator** branch. As in the other two main output branches of the **Menu** step (**DialByExtn** and **DialByName**), the **Operator** output branch contains the **Call Redirect** step, which attempts to transfer the call, in this case to the operator.

Step 3 Configure the **Successful** output branch for the **Call Redirect** step. The **Successful** output branch of the **Call Redirect** step uses the **Set Contact Info** step to mark the contact as **Handled** and the **End** step to end the script.

Step 4 Configure the **Busy** output branch for the **Call Redirect** step. The **Busy** output branch of the **Call Redirect** step uses the **Set** step to set the value of the **prefixPrompt** variable to contain a system prompt that will play back a message to the caller that the extension is busy when the script proceeds to the final **Play Prompt** step (see page 57).

Step 5 Configure the **Invalid Output b**ranch for the **Call Redirect s**tep. If the **Call Redirect** step registers the destination extension as invalid, the script executes the **Invalid** output branch. The **Invalid** output branch of the **Call Redirect** step uses the **Set** step to set the value of the **prefixPrompt** variable. This variable contains a system prompt that will play back a message to the caller that the extension is invalid when the script proceeds to the final **Play Prompt** step (see page 57).

Step 6 Configure the **Unsuccessful** output branch for the **Call Redirect** step. If the **Call Redirect** step registers the destination extension as out of service, the script executes the **Unsuccessful** output branch. The **Unsuccessful** output branch of the **Call Redirect** step uses the **Set** step to set the value of the **prefixPrompt** variable to contain a system prompt that will play back a message to the caller that the extension is out of service when the script proceeds to the final **Play Prompt** step (see page 57).

Step 7 Configure the **If** step for the operator branch. The **If** step allows the script to determine whether or not the maximum number of retries has been reached by evaluating the expression **attempts < MaxRetry**. The expression determines if the number of attempts, as stored in the attempts variable, is less than the maximum number of retries, as stored in the **MaxRetry** variable.

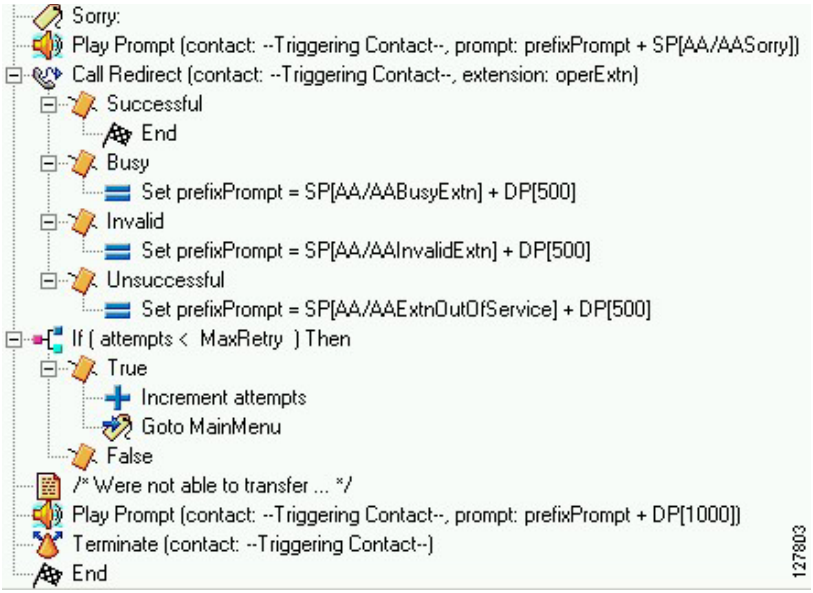
Step 8 Configure the **True** output branch for the **Operator** branch **If** step. If the **If** step determines that the maximum number of retries has not been reached, the script executes the **True** output branch. The **True** output branch of the **If** step allows the caller to keep returning to the **MainMenu** label until the system reaches the maximum number of retries.The **Increment** step increases the value of the attempts variable by 1. The **Goto** step sends the script back to the **MainMenu** label.

Step 9 Configure the **False** output branch for the **Operator** branch **If** step. If the **If** step determines that the maximum number of retries has been reached, the script executes the **False** output branch. The **False** output branch of the **If** step proceeds to the next step at the same level in the script as the **Menu** step, which is the **Play Prompt** step. (See Figure 9.)

Configure the Play Prompt Step

If none of the steps and branches under the **Menu** step succeed in transferring the call, the script proceeds to the **Play Prompt** step. The **Play Prompt** step, as shown in Figure 38, plays (as a last resort) a prompt informing the caller of the inability to transfer the call. The **Play Prompt** step plays **prefixPrompt + SP[AA/AASorry]**, which informs the caller of the reason that the attempted transfer was unsuccessful.

Figure 38 End of Script



Configure the Call Redirect Step

After the **Play Prompt** step informs the caller of the unsuccessful call transfer, the **Call Redirect** step attempts to redirect the call to an operator, using the extension number stored in the **operExtn** variable.

Configure the If Step

The **If** step determines whether or not the maximum number of retries has been reached. As in the previous examples, the **If** step and **Increment** step allow the caller the maximum number of retries. If the transfer is successful, the script ends.

Configure the Play Prompt Step

After the maximum number of retries is reached without successfully transferring the call to an operator, the **Play Prompt** step plays **prefixPrompt**, which explains why the transfer was unsuccessful.

Configure the Terminate Step

After playing the **prefixPrompt**, the **Terminate** step disconnects the call. This step accepts the default contact; no configuration is necessary for this step.

Insert the End Step

The **End** step ends the script and releases all system resources. The **End** step requires no configuration.