Unity Express Custom Scripts Quick Start Guide

TAC Notice: What's Changing on TAC Web

Help us help you.

Please rate this document.

Excellent
Good
Average
Fair
Poor

This document solved my problem.

Yes
No
Just browsing

Suggestions for improvement:

(256 character limit)

Send

Contents

Introduction

This document provides information about the Cisco Unity Express (CUE) system. Specifically, this document offers a primer for the creation of a custom auto attendant application in the product.

For further details on Cisco Unity Express, refer to the Cisco Unity Express Script Editor Guide. This document is intended for administrators who need a basic guide in order to begin to use the Cisco Unity Express Editor and maintain scripts within the Cisco Unity Express product.

Note: Custom script support is not available through the regular Cisco Technical Support. For questions, comments, and help with a custom script, send an email to ask-cue-editor@external.cisco.com.

Prerequisites

Requirements

The reader needs to be familiar with Cisco Unity Express administration and configuration through the command-line interface (CLI) or GUI.

The Cisco Unity Express Script Editor (CUEEditor2.1.1.exe) and the sample script that this document uses (CUE-AA-S6-AASample.zip) are available from the Cisco Unity Express 2.1.1 Software Download Center.

Components Used

The information in this document is based on Cisco Unity Express version 2.1.1. The example contains steps that are only available in Cisco Unity Express 2.1.1. Although, many of the principles are the same for earlier releases. Steps that are specific to 2.1.1 are pointed out explicitly in this document.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, make sure that you understand the potential impact of any command.

Conventions

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Overview

The basic Cisco Unity Express product ships with a voicemail piece and a basic auto attendant (in Cisco Unity Express 2.1.1 and later, there are two auto attendant scripts). Many customers find the standard auto attendant too limited. Cisco Unity Express has a script editor which allows the creation of custom scripts. These custom scripts can be loaded into Cisco Unity Express and replace (or work in addition to) the regular auto attendant.

Warning: The example script shown here is for illustration purposes. It can be modified in any way. The goal of this paper is to illustrate the design and implementation process of a customized script. It is not intended to provide a finished production script. However, if you do choose to use this script, at a minimum, record each of the prompts. You can listen to the attached audio files in order to hear how they sound. But use the Administration via the Telephone (AVT)/Greetings Management System.

Note: At this time, it is not possible to obtain a copy of the actual system scripts that Cisco Unity Express uses. These scripts have steps that the regular script editor does not support. The system scripts as of Cisco Unity Express 2.1.1 are aa.aef, voicebrowser.aef, setmwi.aef, promptmgmt.aef, checkaltgreet.aef, xfermailbox.aef, and aasimple.aef.

The script editor is a tool that allows you to create scripts on a PC separate from Cisco Unity Express. Once created, the scripts are uploaded and configured on the Cisco Unity Express.

This list explains some basic terminology in order to help you understand the concepts in this document.

- **Step**—The basic building block for script creation. Each step is the most basic executable unit, such as an "if" statement, a "Goto", and so on.
- **Script**—One or more steps that are executed in sequence. A script is a file with an .aef extension.

- **Variable**—These are variables in a script. Variables can be of different types, such as Integer, Boolean, String, and so on.

- **Parameter**—This is a property of a variable so that the variable is exposed to the administrator through the Cisco Unity Express web interface. For example, if you have an OperatorExtension variable that you want to assign the value 1000 by default, it sometimes needs to be changed to 2000. In this case, the variable is exposed through the web interface so that the whole script does not have to be loaded into Cisco Unity Express again each time the value needs to be changed.

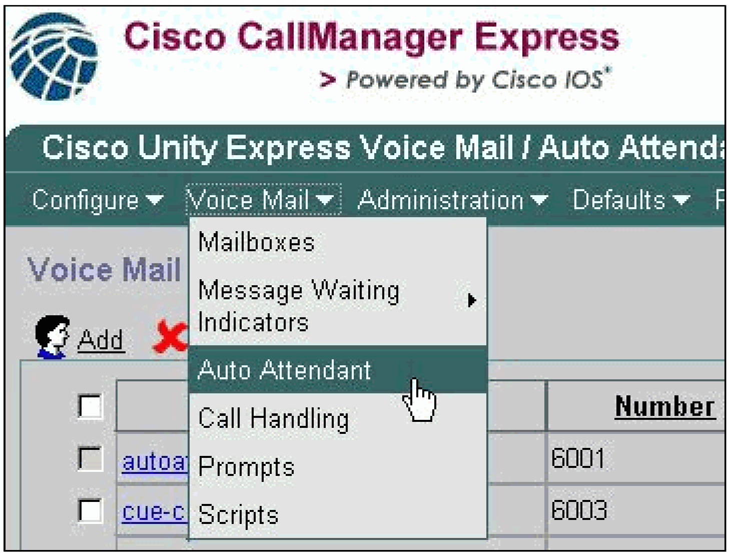
- **Prompt**—A .wav file that can be played. It is either uploaded manually into the Cisco Unity Express system or recorded through the Administration via the Telephone (AVT) system (prior to release 2.1.1 the AVT was called the Greetings Management System (GMS)). All user prompts that are uploaded are placed in the same directory. They are visible in the GUI through the **Voicemail > Prompts** menu item or in the CLI through the **show ccn prompts** command.

  In a script, user prompts are specified as P[<promptname>]. System prompts are pre-recorded and can be used. They are specified as SP[<promptname>]. Appendix 1 lists the available system prompts.

- **Application**—This is the script with all prompts and parameters filled in. By default, Cisco Unity Express ships with the Voicemail, the AVT system, and a simple Auto Attendant application which are all configured when you run the Initialization Wizard at the end of an install.

- **Trigger**—The trigger tells Cisco Unity Express that a particular application needs to be executed. For example, when you dial 1000, the phone system (Cisco CallManager or Cisco CallManager Express) routes the call to Cisco Unity Express. When Cisco Unity Express sees that a call is placed to the number 1000, it looks for a trigger for that extension. The respective application then launches. In other words, it knows whether a call to 1000 needs to go to voicemail, an auto-attendant, or something else. You can have multiple triggers to the same application. There are a number of triggers that are added by default, such as triggers to the voicemail, AVT, and canned auto attendant.

  When you understand the terminology, it makes the Cisco Unity Express web administration tool easier to understand.



From the Cisco Unity Express web page, the Auto Attendant option under the Voice Mail category refers to the Applications as defined in the terminology. Each entry contains the number associated with the application, the script it references, any parameters that are associated with the script, whether or not the application is enabled, and the number of associated ports.

Call Handling currently only contains the associated numbers for voicemail, the built-in Auto Attendant, and the AVT system. In Cisco Unity Express 2.1 and later, under **Administration > Call-in Numbers** there is an additional display of all Triggers, both JTAPI (for Cisco CallManager) and SIP (for Cisco CallManager Express or SRST mode). Currently, the only way to see if there are multiple triggers configured that point to the same application is with the **show ccn trigger** command issued in the CLI.

The Prompts option refers to the audio files (.wav) which exist on the system.

The Scripts selection refers to the .aef files that you can store on the system. Currently, you can upload up to four scripts in addition to the system scripts.

From the CLI, this information is readily available using the **show ccn** commands. This output represents portions of each command output:

```
cue-3745-44a> show ccn application
Name:                                   customaa
Description:                            customaa
Script:                                 customaa.aef
ID number:                              4
Enabled:                                yes
Maximum number of sessions:             8
OperatorExtension:                      205
MainMenu:                               MainMenu.wav
ClosedGreeting:                         ClosedGreeting.wav
InvalidExt:                             InvalidExt.wav
MaxRetries:                             3
MaxExtension:                           205
SorryGoodbye:                           SorryGoodbye.wav
EnterExtension:                         EnterExtension.wav
namePrompt:                             namePrompt.wav
MinExtension:                           200
...

cue-3745-44a> show ccn prompts
Name:                         MainMenu.wav
Language:                     de_DE
Last Modified Date:           Wed Dec 22 03:34:57 GMT+00:00 2004
Length in Bytes:              121978
Name:                         EnterExtension.wav
Language:                     de_DE
Last Modified Date:           Wed Dec 22 03:34:26 GMT+00:00 2004
Length in Bytes:              21338
Name:                         SorryGoodbye.wav
Language:                     de_DE
Last Modified Date:           Wed Dec 22 03:34:02 GMT+00:00 2004
Length in Bytes:              17658
Name:                         ClosedGreeting.wav
Language:                     de_DE
Last Modified Date:           Wed Dec 22 03:33:00 GMT+00:00 2004
Length in Bytes:              86138
Name:                         InvalidExt.wav
Language:                     de_DE
Last Modified Date:           Wed Dec 22 03:47:07 GMT+00:00 2004
Length in Bytes:              29818
```

```
Name:                         namePrompt.wav
Language:                     de_DE
Last Modified Date:           Wed Dec 22 03:40:28 GMT+00:00 2004
Length in Bytes:              22618
...

cue-3745-44a> show ccn trigger
Name:                         18955
Type:                         SIP
Application:                  customaa
Locale:                       systemDefault
Idle Timeout:                 10000
Enabled:                      yes
Maximum number of sessions:   8
...
```

The **show ccn application** output displays the application name, the script it refers to, whether or not it is enabled and how many simultaneous calls it can handle, along with all script parameters with their values.

The **show ccn prompts** command lists all commands, their language, size and modified date.

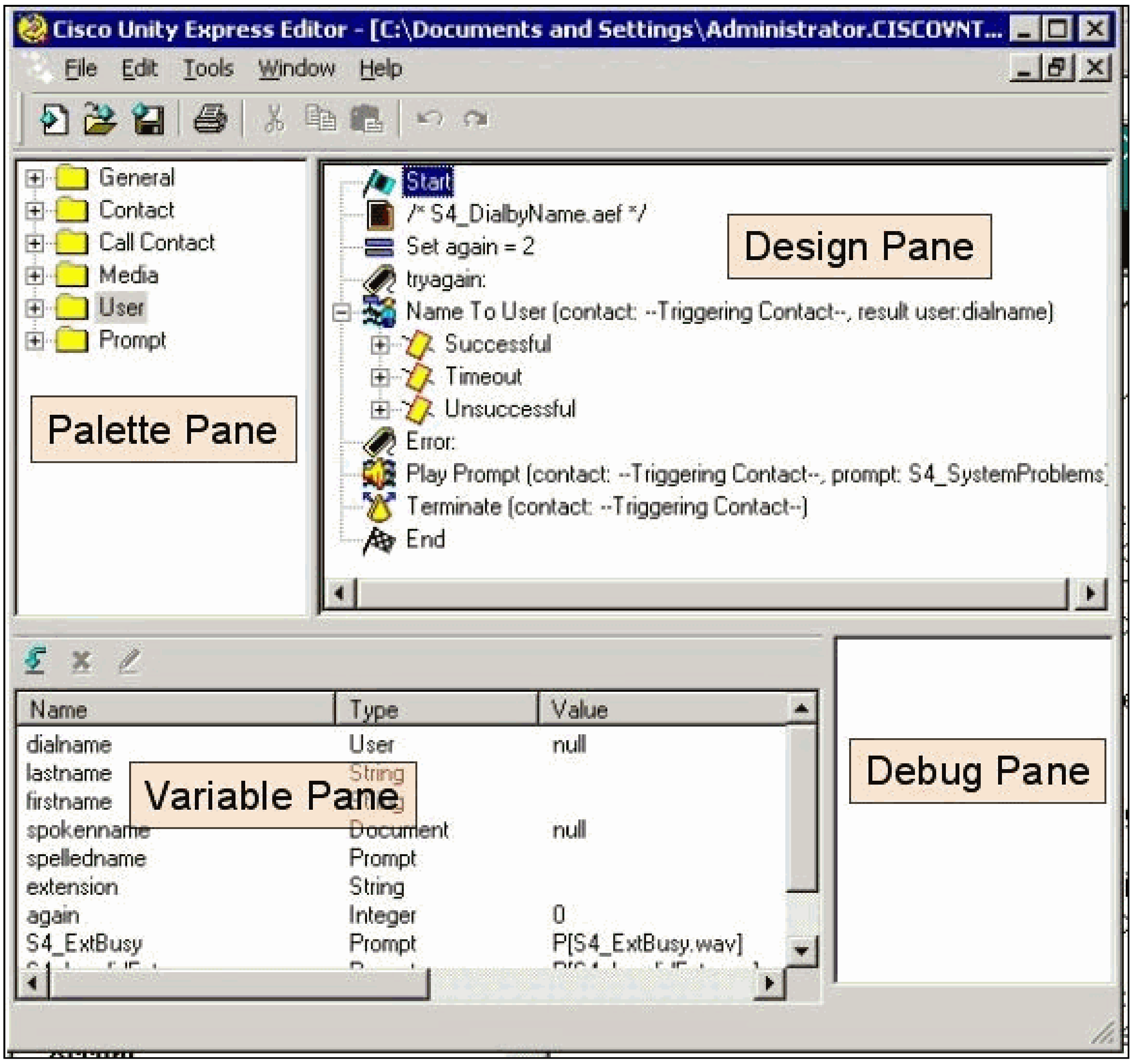
The **show ccn trigger** information gives the number ("Name") as well as the application it calls, the status (enabled/disabled), and the number of sessions.

Use the Cisco Unity Express Script Editor

Use the Script Editor (registered customers only) to create a new script. It must be installed on a separate PC.

At the same location are several sample scripts. These scripts help to learn how to do specific operations in a script. Cisco recommends you install the Editor and download a few sample scripts in order to become familiar with the way the Editor works.

Once the Script Editor launches, you can open a script as shown in this graphic:


Cisco Unity Express Editor - [C:\Documents and Settings\Administrator.CISCOVNT...
File Edit Tools Window Help
General
Contact
Call Contact
Media
User
Prompt
Palette Pane
Design Pane
Variable Pane
Debug Pane

```
Start
/* S4_DialbyName.aef */
Set again = 2
tryagain:
Name To User (contact: --Triggering Contact--, result user:dialname)
    Successful
    Timeout
    Unsuccessful
Error:
Play Prompt (contact: --Triggering Contact--, prompt: S4_SystemProblems
Terminate (contact: --Triggering Contact--)
End
```

Name	Type	Value
dialname	User	null
lastname	String	
firstname	String	
spokenname	Document	null
spelledname	Prompt	
extension	String	
again	Integer	0
S4_ExtBusy	Prompt	P[S4_ExtBusy.wav]

The Palette pane contains folders with each step, categorized based on their major function. From there, drag steps over to the Design pane in order to create a script. In the Variable pane, you can create and edit variables. After you drag a step to the Design pane, you can right-click on a step, select **Properties**, and customize the step and variables assigned to the step. Before you upload a script to Cisco Unity Express, you always need to validate it first through the **Tools > Validate** option. Any error output then displays in the Debug pane.

One of the most useful things to remember when you use the Editor is the Help function. You can either access it directly through the Help menu on the toolbar, or when you select a particular step property.

Create a Basic Script

You can use the Help menu in order to familiarize yourself with some of the steps. In the Script Editor, drag a few steps from the palette to the design pane and right-click on one and select **Properties**. Then click on the **Help** button and get help on that particular step.

This document describes the creation of a sample script. The document also discusses some of the common issues that people try to solve when they create a custom auto attendant. The goal here is to create a script that answers a call, checks to see if there is an emergency alternate greeting and if the current time is regular business hours or not. If it is closed, play a custom greeting and send the call to the operator. If it is open, play a menu that allows the caller to dial 1 for dial-by-name, 2 for dial-by-extension or 0 for an operator. The dial-by-extension option should only transfer the call to a specified range, not any number that you can enter.

Warning: The example this document uses is for illustration purposes only. While you are free to use whatever portions you like, it probably does not exactly match with the situation of your company. Especially in the areas of error handling, this script is of not much use.

Note: Aside from the sample scripts at www.cisco.com, the Cisco Unity Express CD contains a similar script as the one used here, called aa_sample1.aef. It is very similar to the regular, shipping Auto Attendant (prior to 2.1). If you plan to use the canned script as a starting point, use this script instead of the aa.aef file that is found when you install some versions of the Script Editor.

Integration Methods

Cisco Unity Express currently supports integration to Cisco CallManager or Cisco CallManager Express. Although the script itself does not change, the integration method can result in some slight differences. When you execute a Redirect step, which logically transfers the call from Cisco Unity Express to an external extension, this integration method is critical. When integrated with Cisco CallManager Express, which is the same for a Cisco CallManager integration currently in SRST mode, the signaling is done through SIP in the form of a BYE/Also message. What this means is that a redirect hangs up the call to Cisco Unity Express and tells the Cisco CallManager Express/SRST router to connect the caller to the extension specified in the "Also:" portion of the BYE message. If that extension is invalid or otherwise unreachable, the caller can be dropped. This is because there is no way for Cisco Unity Express to reclaim that call and be notified that the transfer fails. It is a true blind transfer. If the call redirects to an unregistered or busy or extension that does not answer, the regular call forwarding rules that apply to that directory number are in effect. Cisco Unity Express is no longer a part of the call. This also means that when you script Redirect steps, currently there is not much of a benefit when you add code to deal with busy/invalid/unsuccessful redirects, since the redirect itself is equivalent to a disconnect. There is no failure case possible.

For Cisco CallManager integrations, Cisco Unity Express communicates through Java Telephony Application Programming Interface (JTAPI). This allows busy/invalid/unsuccessful redirect steps to occur. Therefore, add code to the script in order to handle those situations. When all Cisco CallManagers become unreachable and the system falls into SRST mode, this uses the SIP signaling discussed earlier in this document. Therefore, all of the limitations described in this document apply.

Sample Callflow

Before you start with a production script, it is imperative to map out the complete call flow. For this example assume that all extensions are in the range 200-299 (the important point is that they all start with 2). For this example, you have something such as this procedure:

1. Answer the call.

2. Check if there is an alternate greeting. If there is, play it.

3. Check if it is a holiday. If so, play a holiday greeting and proceed to the main menu.

4. Check if it is during or after business hours and play the respective (open/closed) greeting. Proceed to the main menu.

5. The main menu needs to play a prompt and allow users to enter an extension at any time if

they know it. They are allowed to enter 1 if they want to look up a user in the directory, or 0 if they want to reach the operator. Since all valid user extensions start with 2, entering 2 branches to the dial-by-extension section.

6. The dial-by-name looks up users and transfer to them. The user is transferred to the operator if zero is entered.

7. The dial-by-extension option immediately attempts to collect two more digits. If less are entered, it goes back to the main menu. If two are collected, it attempts to look up the user. The user is transferred to that extension if the user exits, otherwise the user goes to the main menu.

8. The transfer to the operator function transfers the call. If it fails for some reason, it goes back to the main menu.

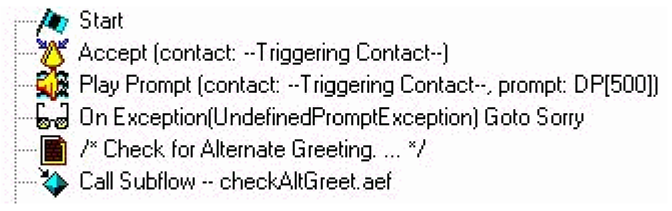
9. An additional error handling sequence for various problems (such as multiple unsuccessful transfers) which tells the user to try again later and hangs up.

A lot can be done for error handling. For a real implementation, it probably helps to map out the complete call flow (this includes error handling, and every user interaction or choice) in a flowchart.

Sample Script

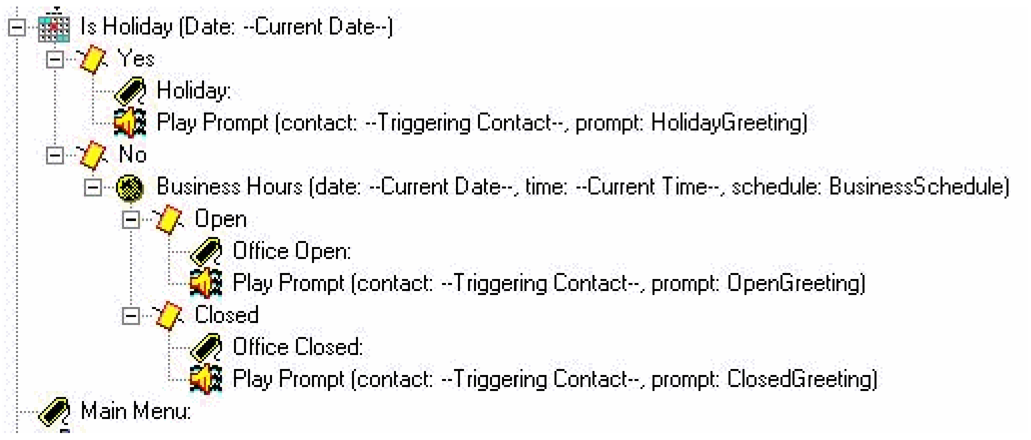
Start

Enter the script in the Script Editor.



```
Start
Accept (contact: --Triggering Contact--)
Play Prompt (contact: --Triggering Contact--, prompt: DP[500])
On Exception(UndefinedPromptException) Goto Sorry
/* Check for Alternate Greeting. ... */
Call Subflow -- checkAltGreet.aef
```

All scripts begin with a Start step. The first thing to do is Accept the call. After that, play a prompt (DP[500]) which is a 0.5 second pause. In some situations, audio cut-through to the public switched telephone network (PSTN) does not happen quickly. Play something before and "real" greeting in order to avoid any perceived clipping to the greeting. Next is an On Exception Goto step. What this does is look for any exception of type "UndefinedPromptException" at the time of the execution of the script. This particular exception occurs when the step tries to play a prompt that does not exist. This only occurs if the step that plays the prompt has the "Continue On PromptErrors" property set to No. If set to Yes, no prompt is played and no exception is thrown. The use of this On Exception Goto step makes sure that if a critical prompt that is administrable by the administrator (such as the main menu), is erased, you can branch to a section where you can branch to the "Sorry" label in order to tell the user to try back later. After that, call a subflow, checkAltGreet.aef. This is a system script, which is always there. It checks for the existence of the AltGreeting.wav file. If it exists, it plays it. If it does not, the subflow returns. This is handy since you can use it in conjunction with the GMS (now called AVT system) in order to play an emergency greeting of some kind before the regular message. For more information on this feature, refer to Configuring and Using the Greeting Management System and Emergency Alternate Greeting for Unity Express. There is no way to tell if the alternate greeting actually plays (or checks for the existence of a file in the local repository). You do not have to use this in the start of a script. Based on your application, you can use the emergency alternate greeting functionality in order to record holiday prompts, for example. Instead of calling this subflow, call it only if it is a holiday. This helps the site administrators to record holiday greetings through the Telephony User interface without the need to access the system through a browser or the CLI (to rename prompts or to assign them to script variables).

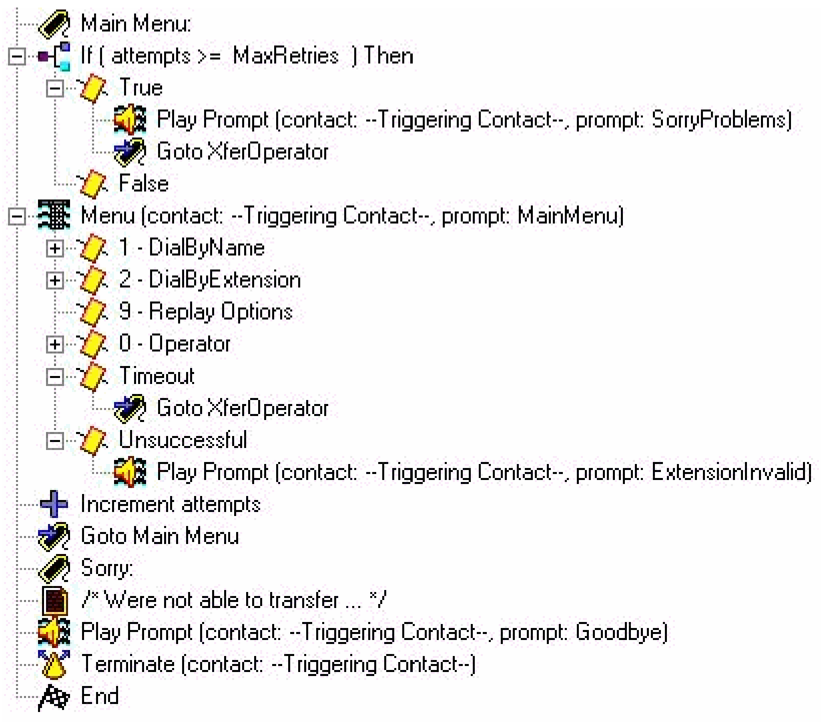


```
Is Holiday (Date: --Current Date--)
    Yes
        Holiday:
        Play Prompt (contact: --Triggering Contact--, prompt: HolidayGreeting)
    No
        Business Hours (date: --Current Date--, time: --Current Time--, schedule: BusinessSchedule)
            Open
                Office Open:
                Play Prompt (contact: --Triggering Contact--, prompt: OpenGreeting)
            Closed
                Office Closed:
                Play Prompt (contact: --Triggering Contact--, prompt: ClosedGreeting)
Main Menu:
```

Decide which opening prompt to play. If it is a holiday, play the holiday greeting. If it is during business hours (which are configurable through schedules in the Cisco Unity Express web administrator), then play the opening greeting (which can be something as simple as "thanks for calling Cisco Systems"), else play the closed greeting. Regardless of which one you play, you also then go straight to the main menu.

Note: The "Is Holiday" and "Business Hours" steps are specific to 2.1.1 and later. For earlier versions of Cisco Unity Express, the only option is to use the "Time of Day" step. This does not allow you to play different greetings based on the day or date. It is based only on the time.

Main Menu



```
Main Menu:
If ( attempts >=  MaxRetries  ) Then
    True
        Play Prompt (contact: --Triggering Contact--, prompt: SorryProblems)
        Goto XferOperator
    False
Menu (contact: --Triggering Contact--, prompt: MainMenu)
    1 - DialByName
    2 - DialByExtension
    9 - Replay Options
    0 - Operator
    Timeout
        Goto XferOperator
    Unsuccessful
        Play Prompt (contact: --Triggering Contact--, prompt: ExtensionInvalid)
Increment attempts
Goto Main Menu
Sorry:
/* Were not able to transfer ... */
Play Prompt (contact: --Triggering Contact--, prompt: Goodbye)
Terminate (contact: --Triggering Contact--)
End
```

Enter the Main Menu section. Since much of the error handling later sends the caller back to the Main Menu, a counter is added. The counter "attempts" are compared to "MaxRetries", another variable that is configurable by the user (3 by default). If that counter is reached, apologize and transfer the caller to the operator ("Sorry you are having problems. Please hold for an operator.").

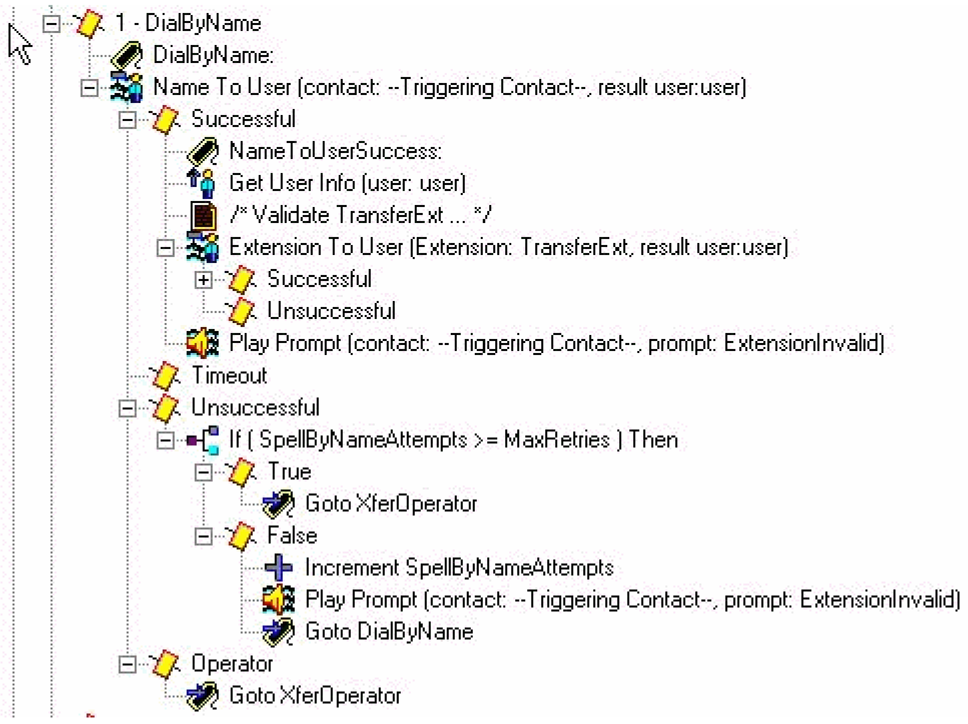
The Menu step plays a basic prompt ("If you know your party's extension please dial it any time. For spell-by-name press 1, for an operator press 0, to repeat these options, press 9."). In this case, since you play your own recorded prompt, you change the Maximum Retries parameter on this step to 0. This is done everywhere you have an option to do this. The reason is that if it is something other than 0, you hear the system prompt "are you still there?" each time the expiry timer is reached and it branches to the Timeout branch. The Unsuccessful branch is reached when someone presses anything other than 1, 2, 9, or 0. In this case, you play a prompt "The extension entered is invalid" and drop out of the step where you increment the attempts counter and go back to the Main Menu.

Note: In order to to ensure that the audio stops as soon as you enter a digit on Media steps (such as Menu or Play Prompt), you must check the **Barge In** field in the Prompt tab of the step. The Interruptible field (on the General tab) is used for other step interruptions that are currently not applicable to Cisco Unity Express.

A single error handling section is added. The "Sorry" section plays a message ("We are unable to transfer the call at this time. Please try again later. Goodbye."). The call then disconnects.

You can always use a system greeting instead of these custom variables. System prompts always have the SP[] notation while user prompts use the P[] notation. You cannot assign a system prompt to a prompt variable when you define it (in the variable pane since those are by definition user values). Once defined, in the script you can assign a system prompt to a variable through the Set step (or any of the Create Prompt steps).

Dial By Name



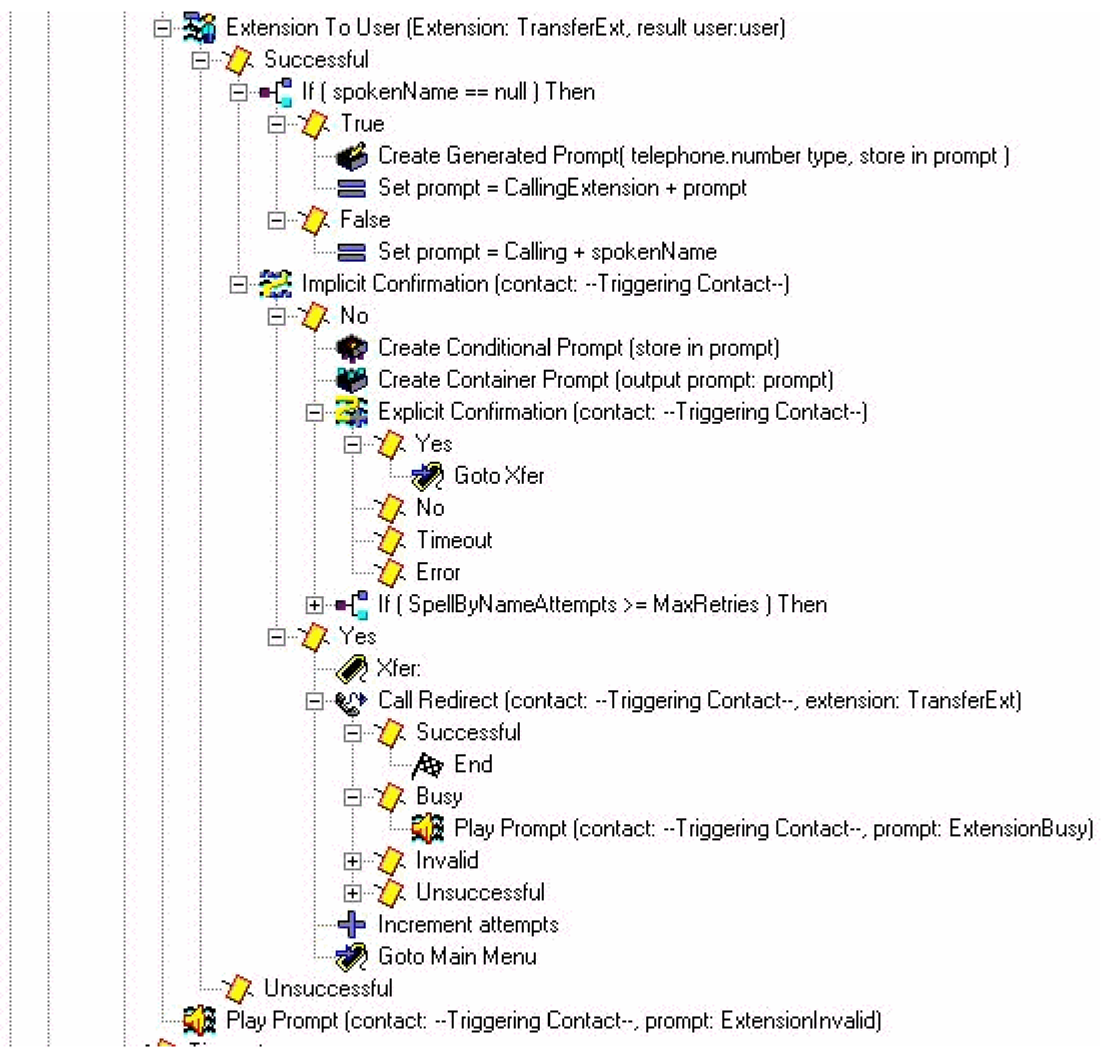
```
1 - DialByName
    DialByName:
    Name To User (contact: --Triggering Contact--, result user:user)
        Successful
            NameToUserSuccess:
            Get User Info (user: user)
            /* Validate TransferExt ... */
            Extension To User (Extension: TransferExt, result user:user)
                Successful
                Unsuccessful
            Play Prompt (contact: --Triggering Contact--, prompt: ExtensionInvalid)
        Timeout
        Unsuccessful
            If ( SpellByNameAttempts >= MaxRetries ) Then
                True
                    Goto XferOperator
                False
                    Increment SpellByNameAttempts
                    Play Prompt (contact: --Triggering Contact--, prompt: ExtensionInvalid)
                    Goto DialByName
        Operator
            Goto XferOperator
```

One way to do Dial By Name is to download the sample (S4_DialByName) from Cisco.com, load it as a separate script into Cisco Unity Express and then add a Call Subflow step to invoke it. In this case the step is put directly into the script and is discussed in detail. The NameToUser Step only allows you to configure one of its prompts ("Spell the last name of the person you want to call, followed by the first name for letter Q press 7, and for letter Z press 9"). It is not possible to eliminate all of the system prompts. In order to minimize them, you first need to disable the Cancel key (default *). If it is present, ,the system adds a message "to start over, press star" at the end of the recorded prompt. Second, change Maximum Retries to zero, so that the "please try again" and "are you still there" messages cannot occur. Always keep track of the number of tries with a separate counter variable that you add. There is still one system prompt that cannot be eliminated. It is the prompt for multiple matches ("More than one name was found. Select from the following..."). The Name To User step always ends as Successful, Timeout, Unsuccessful, or the Operator (if selected in the step properties). Except for the Successful and Operator branch, you drop out of the Menu step. This means that you increment the attempts counter and go back to the Main Menu. Based on the situation, you can have a separate retry counter and add a step to go back to the DialByName label so that it repeats the action that is already selected.

Note: One defect in release 2.1.1 that required consideration is Cisco bug ID CSCeg81385 (registered customers only) . When Maximum Retries on the NameToUser step is set to zero or one, you must use a termination character. Without the termination character, it does not go to the "Successful" branch of this step. Instead it always goes to "Timeout", even if there is a match.

If the user does not enter an extension that can be mapped to a name, the Unsuccessful branch checks a new counter (SpellByNameAttempts) against MaxRetries. Assume that the same maximum number of retries to enter the main menu needs to apply to the spell-by-name function. You can add a separate variable if these need to be different.

A successful NameToUser step means that at least one digit is entered and the last name of a single user as defined in Cisco Unity Express is selected (the Timeout branch is only reached if no digits at all are entered). In order to be sure that the extension is unique, check that you can map the extension back to the user. This can be accomplished by the ExtensionToUser step. This step is added in Cisco Unity Express 2.1.1. Previously, there was no way, given an extension, to make sure that it mapped to a real user in the voicemail system. The best you could do was to set up some variables to compare against, such as with the addition of a step "If (TransferExt < MinExtension) || (TransferExt > MaxExtension)..." or something similar.

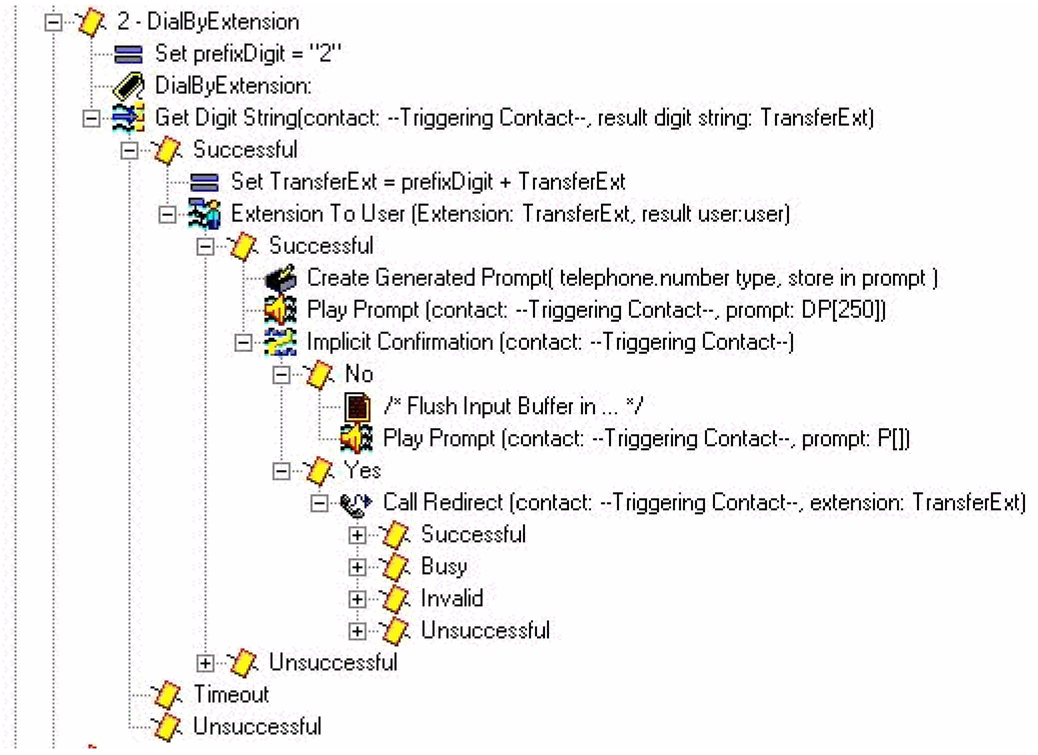


```
Extension To User (Extension: TransferExt, result user:user)
    Successful
        If ( spokenName == null ) Then
            True
                Create Generated Prompt( telephone.number type, store in prompt )
                Set prompt = CallingExtension + prompt
            False
                Set prompt = Calling + spokenName
        Implicit Confirmation (contact: --Triggering Contact--)
            No
                Create Conditional Prompt (store in prompt)
                Create Container Prompt (output prompt: prompt)
                Explicit Confirmation (contact: --Triggering Contact--)
                    Yes
                        Goto Xfer
                    No
                    Timeout
                    Error
                If ( SpellByNameAttempts >= MaxRetries ) Then
            Yes
                Xfer:
                Call Redirect (contact: --Triggering Contact--, extension: TransferExt)
                    Successful
                        End
                    Busy
                        Play Prompt (contact: --Triggering Contact--, prompt: ExtensionBusy)
                    Invalid
                    Unsuccessful
                Increment attempts
                Goto Main Menu
    Unsuccessful
Play Prompt (contact: --Triggering Contact--, prompt: ExtensionInvalid)
```

Here you see the script after you map the extension back to the user. If it does not then play a prompt that the extension is invalid. Usually there is no extension for some reason. In the case of Cisco CallManager Express, this can mean that you hang up if you try to transfer the call. This can cause slight confusion since the caller already selected a user. However, this is clearly already an error condition that you want to know about anyway. If mapping the extension to a user is successful, first check to see if they have a recorded name (the spokenName variable is filled in when you look up that user). Then set the prompt variable to **Calling** plus the spokenName. Otherwise, set it to **Calling Extension** plus the spelled extension of the user. You can also spell out the name (play S[name] in the prompt). There is no recorded name to play. Any system spelling, such as spelling out a name or an extension, is always done in the system's voice and cannot be customized. The Implicit Confirmation step plays the prompt that you just created. If the user does not type anything, it goes ahead and enters the 'Yes' branch and redirects the call. If there is a problem, in the case of Cisco CallManager integration, it plays a greeting and the caller is sent back to the Main Menu. If the caller enters something, generate another prompt. This time begin the prompt with either the spokenName, or a spelled version of the name of the user (Create Conditional Prompt step). Follow this by a pause and a prompt "if this is the name of the person you are calling, press 1". Now the user can either press 1 for yes, 2, *, or wait. The system either sends them back to the spell-by-name section or to the operator. This depends on how many times they have gone through the system.

Dial By Extension

The dial by extension section already has the first digit ("2" in this example) collected. Set the prefixDigit variable to **2**. You can easily change this, based on the leading digit (or if there is more than one possible leading digit). No leading digit can be the same as another option in the menu (such as 1, which you use for dial-by-name in this example). The Get Digit String also collects a fixed number of digits (two in this case). Therefore, the total number of digits needs to be fixed. If the valid extensions begin with 2 or 3, then you can add a "Set prefixDigit = "3"" step to option 3 of the Menu step followed by a Goto DialByExtension step.

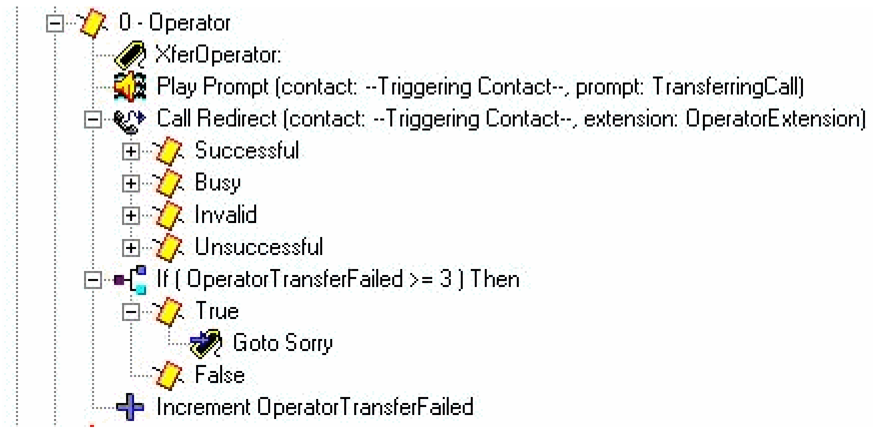


```
2 - DialByExtension
    Set prefixDigit = "2"
    DialByExtension:
    Get Digit String(contact: --Triggering Contact--, result digit string: TransferExt)
        Successful
            Set TransferExt = prefixDigit + TransferExt
            Extension To User (Extension: TransferExt, result user:user)
                Successful
                    Create Generated Prompt( telephone.number type, store in prompt )
                    Play Prompt (contact: --Triggering Contact--, prompt: DP[250])
                    Implicit Confirmation (contact: --Triggering Contact--)
                        No
                            /* Flush Input Buffer in ... */
                            Play Prompt (contact: --Triggering Contact--, prompt: P[])
                        Yes
                            Call Redirect (contact: --Triggering Contact--, extension: TransferExt)
                                Successful
                                Busy
                                Invalid
                                Unsuccessful
                Unsuccessful
        Timeout
        Unsuccessful
```

Since you have three-digit extensions in this environment, the Get Digit String step Input length parameter is 2. There is no terminating character and no cancel key. The maximum number of retries is 0. If a user does not enter a valid extension, the script increments the number of attempts and goes back to the main menu. The step does not play any audio (Prompt is P[]) since the user is already in the middle of dialing digits.

When two digits are successfully collected, prepend the prefixDigit ("2" in this case) that is already dialed (which allows you to branch from the Menu step). The TransferExt that results then passes to the Extension To User step, which verifies if this is a valid user. If it is, then create a prompt with the spelled out extension. In the Implicit Confirmation step play the "calling extension" prompt followed by the extension. Look up the user and play the spoken name, as you did in the DialByName. However, in this case it is not necessary. If no digit is dialed, assume that this is the extension the caller wants to reach, and redirect the call. Otherwise cycle back to the Main menu. The purpose of the Play Prompt DP[250] step is to purge any other input, because the Get Digit String step does not have any termination character configured. The idea is that you want this to work if someone dials extension "200" or "200#". If the # is entered while the Implicit Confirmation step runs, then it goes to the No portion of that step. In some cases, it is probably not necessary to have an Implicit Confirmation step at all, but rather to redirect the call right away. For illustration purposes, these steps are left in.

Operator Transfer



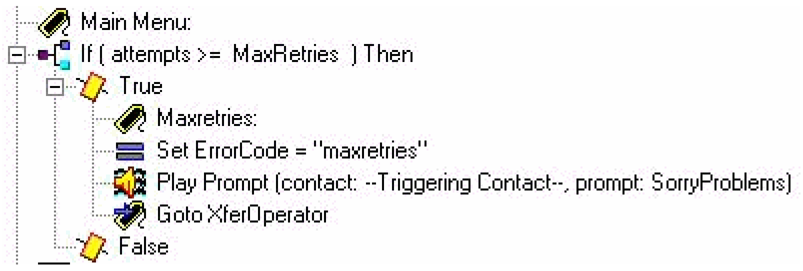
```
0 - Operator
    XferOperator:
    Play Prompt (contact: --Triggering Contact--, prompt: TransferringCall)
    Call Redirect (contact: --Triggering Contact--, extension: OperatorExtension)
        Successful
        Busy
        Invalid
        Unsuccessful
    If ( OperatorTransferFailed >= 3 ) Then
        True
            Goto Sorry
        False
    Increment OperatorTransferFailed
```

Play a prompt "Transferring call" then redirect it to the configured operator extension. If it fails for some reason, you have a separate counter which you increment and go back to the main menu. The reason for this is that you want to have some method to give the operator some time to get off the phone or re-register so that you can successfully try to transfer to them again. After a number of attempts (hard-coded at 3 in this example), you branch to the "Sorry" label, which drops the call.

Advanced Error Handling

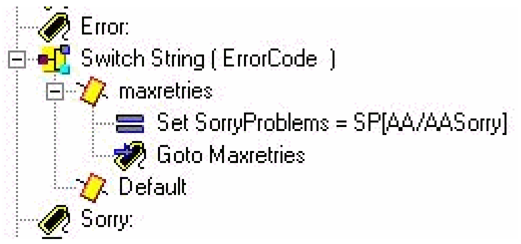
There are many places where scripts can fail. One of the more common areas in a well-designed script is when user prompts are missing. When the system attempts to play a prompt which does not exist, a number of things can occur. First, if the step that plays the prompt has "Continue On Prompt Errors" set to Yes. It moves on to the next step and does not play the prompt. If Continue On Prompt Errors is set to No, an exception is generated. Most of the time, this is the "UndefindedPromptException", which means that the prompt is not found or is never configured. When you have an On Exception Goto step, it can allow you to catch this exception and branch to a section of code that can either play a system message, or another form of audio instead. Since the On Exception Goto step applies to all audio that plays in the entire script (you cannot define a separate On Exception Goto step for each prompt you play), the best way to handle this is to set an error code variable immediately before you play the prompt. This way, you can branch somewhere, take an action based on that error code, and then possibly branch back.

For example, you can modify the script with this:



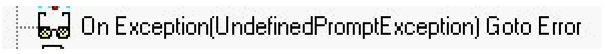
```
Main Menu:
If ( attempts >=  MaxRetries  ) Then
    True
        Maxretries:
        Set ErrorCode = "maxretries"
        Play Prompt (contact: --Triggering Contact--, prompt: SorryProblems)
        Goto XferOperator
    False
```

You have added the "Maxretries" label. This is used to get back to the same place. Set an ErrorCode variable to **Maxretries**. In the Play Prompt step, make sure that Continue On Prompt Errors is set to **No**. Now add a section that can handle this. See if the ErrorCode matches "Maxretries", then set the SorryProblems variable (which you attempted to play earlier) to the system message **Sorry you are having trouble**. **Please stay on the line and some will be with you shortly**. Then go to the Maxretries label so that it can play.



```
Error:
Switch String ( ErrorCode )
    maxretries
        Set SorryProblems = SP[AA/AASorry]
        Goto Maxretries
    Default
Sorry:
```

The last step is to change the On Exception Goto message to look for the UndefinedPromptException error:



```
On Exception(UndefinedPromptException) Goto Error
```

This step can be anywhere in the script. However, it is usually put at or near the beginning. There are a number of steps which can play prompts.. The system prompts are listed at the bottom of this document.

This method can be used instead of calling the checkAltGreet.aef subflow. You need to play the AltGreeting.wav file. If there is an error, handle it. The benefit is that you know when it does or does not play.

Although less common, these are the other prompt exceptions:

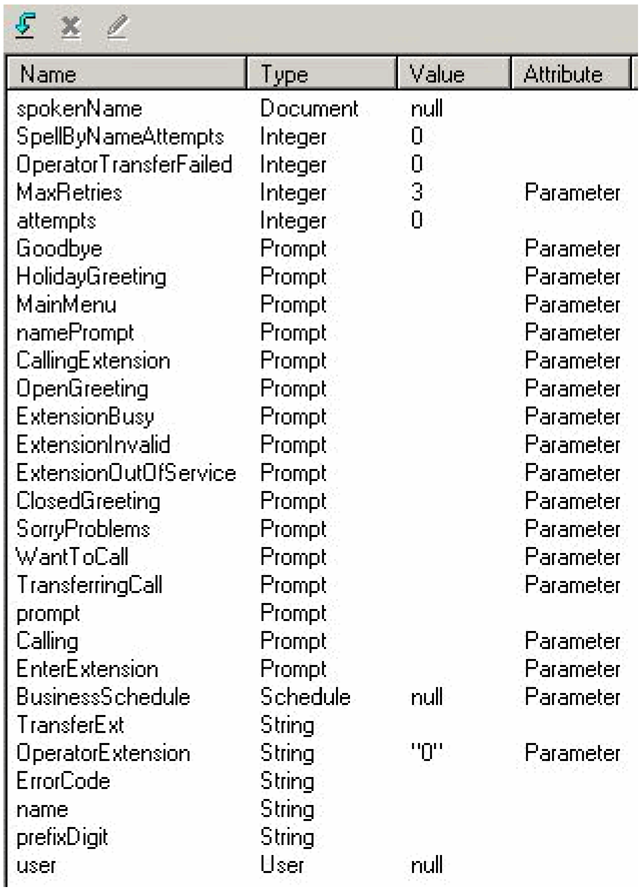
- UndefinedPromptException
- PromptException
- UndefinedPromptGenerator
- InvalidPromptArgumentException
- UnsupportedPromptExpression

Transfer a Call Directly to Voicemail

It is not currently possible to redirect the call straight to a voicemail box. It must be sent to a number which forwards to voicemail. This can be a dummy number in Cisco CallManager Express (or even in Cisco CallManager) which is configured to call-forward all set to the voicemail pilot number. Use the Cisco Unity Express number/e.164 number settings in order to match that number to a subscriber or group. Transfer a Caller Directly into a Unity Express Mailbox explains how this can be done.

Script Variables

You must create all variables used in a script before you can use them. If you use one and then later delete it, select **Tools > Validate** in the Script Editor in order to find it. For this sample script, these are the variables that are used. Notice the Name, Type, Value (which is the value of the variable initially) and Attribute. The Parameter attribute means that the variable is exposed to the administrator through the web page. Anything that you want the Cisco Unity Express administrator to customize needs to have the Parameter attribute.



Name	Type	Value	Attribute
spokenName	Document	null	
SpellByNameAttempts	Integer	0	
OperatorTransferFailed	Integer	0	
MaxRetries	Integer	3	Parameter
attempts	Integer	0	
Goodbye	Prompt		Parameter
HolidayGreeting	Prompt		Parameter
MainMenu	Prompt		Parameter
namePrompt	Prompt		Parameter
CallingExtension	Prompt		Parameter
OpenGreeting	Prompt		Parameter
ExtensionBusy	Prompt		Parameter
ExtensionInvalid	Prompt		Parameter
ExtensionOutOfService	Prompt		Parameter
ClosedGreeting	Prompt		Parameter
SorryProblems	Prompt		Parameter
WantToCall	Prompt		Parameter
TransferringCall	Prompt		Parameter
prompt	Prompt		
Calling	Prompt		Parameter
EnterExtension	Prompt		Parameter
BusinessSchedule	Schedule	null	Parameter
TransferExt	String		
OperatorExtension	String	"0"	Parameter
ErrorCode	String		
name	String		
prefixDigit	String		
user	User	null	

If you do not want to record all of these custom prompts, you can edit each one and uncheck the **Parameter** attribute so that it is not exposed via the web page. In order to make sure the system still works, add several Set steps to the start of the script. For example, set 'Calling' to 'SP [AA/AACalling]'. Not all of the prompts shown here have corresponding system scripts. In most cases, it makes sense to record them yourself. If the prompts do not change, you can record them once, make sure they exist on the Cisco Unity Express System with a given file name, and then remove the Parameter attribute from the variable. This way, the prompt plays a fixed file name. But, the variable is not exposed via the web interface where an administrator can accidentally change it.

Upload a Script to Cisco Unity Express

Usually, the first step to add a new application and script to Cisco Unity Express is to upload the prompts. This is important because the prompts have to exist on the system in order for you to configure them in the script parameters. Otherwise, you have to leave the parameters blank, upload the prompts, and then go back and edit the script parameters. For additional information on how to record and upload prompts, refer to Administration via the Telephone (AVT)/Greetings Management System.

Once you have your script saved and are ready to try it out, you first need to upload it to the Cisco Unity Express. The easiest way is to use the GUI, since that does not require an FTP server to load the script files from.

Complete these steps:

1. Log in to the Cisco Unity Express GUI with an account that has administrative rights.

2. Select **Voice Mail > Auto Attendant**.

3. Click **Add**.

4. Next to the "Selected Automated Attendant Script", click the **Upload** button.

5. Click **Browse**.

6. Find the script file, select it and press **OK**.

7. If you want to change the destination file name, do so. Otherwise press **Upload**.

8. Enter an Application Name. This can be anything in lower case and is used to refer to the application associated with this script file.

9. Click **Next**.

10. The page now shows all the variables for the script that are marked as parameters. This means that they can be configured on this page. If you have not uploaded the prompt files yet, then you can go through this first and then fill in the information later.

11. Click **Next** when you finish entering script parameters.

12. Make sure that the application is marked as **Enabled**, configure the maximum number of concurrent calls (default is the maximum number allowed by the license) and enter a call-in number.

    This is the number that is called for which this particular application needs to be invoked (trigger). This can be your main auto-attendant number, or a temporary one. You can also skip this and fill it in later. Without a call-in number, you cannot test or launch a script.

    Note: For Cisco CallManager integrations with SRST support, you need to enter multiple call-in numbers. Prior to Cisco Unity Express 2.1.1, this information had to be entered through the CLI. In Cisco Unity Express 2.1, multiple call-in numbers (triggers) can be entered from the **Administration > Call-in Numbers** screen.

13. Click **Finished**.

Once you upload the script and configure the application, you have to make sure you can route calls to the application. This means that either Cisco CallManager Express or Cisco CallManager must be configured. With CallManager Express, this involves the configuration of a dial-peer that points to Cisco Unity Express that matches the pattern (trigger number or the Call-in number) configured in Cisco Unity Express. In Cisco CallManager, a CTI Route Point with this number must be added. Do not forget that you must associate this CTI Route Point with the JTAPI user that Cisco Unity Express logs in as. This configuration can possibly require a reboot of Cisco Unity Express. Once you add the Route Point and associate with the user, you can change the script as much as you want without the need to reboot.

When you update a script, if no variable parameters are added or changed, you can go into the Cisco Unity Express editor, click **Voice Mail > Scripts**, Upload, and select the script. It is overwritten with all other configured parameters that remain the same.

When you create prompts, the easiest thing to do is to use the AVT system. The file names it records have date/time stamps included. But in Cisco Unit Express 2.1.1 and later, it is possible to use the CLI **ccn rename prompt** command. Previously, you had to download the prompt, re-upload it with a new name, then delete the original .wav file. For example:

```
cue-3745-44a> show ccn prompts

...
Name:                           UserPrompt_01032005170038.wav
Language:                       en_US
Last Modified Date:             Mon Jan 03 17:00:38 EST 2005
Length in Bytes:                35098

cue-3745-44a> ccn rename prompt UserPrompt_01032005170038.wav
MyPrompt.wav language en_US
Warning! Any existing ccn applications that use this prompt will
stop working until they are manually updated with the new prompt name.
Continue? (y/n) y
cue-3745-44a>
```

Troubleshoot

When you troubleshoot a script, you need to call in and test various scenarios, both regular user input, as well as timeout and anticipated error conditions.

Before you upload a script, always access the **Tools > Validate** entry in the Script Editor and make sure that you get the "Validation succeeded" message. If not, all errors display in the debug pane. You can double-click them and it shows you where the error condition exists in the script.

In most instances, the default traces are more than sufficient for troubleshooting problems. In order to simplify this, leave the default traces on and utilize the filtering capability in order to display what you need.

Before any test, issue the **clear trace** command. This clears the memory buffer of trace messages so that the amount of information filtered only includes the data after the command is issued.

For script execution, the most important item to filter on is EXECUTING_STEP. This shows all steps as they are run. For example:

```
cue-3745-44a> clear trace

cue-3745-44a>   /*** NOW THE TEST CALL IS MADE  ***/

cue-3745-44a> show trace buffer long  | include EXECUTING_STEP

3119 12/28 17:05:33.955 ACCN APMG 0 EXECUTING_STEP:Executing a step:
Application=App[name=customaa,type=Cisco Script Application,id=4,
desc=customaa,enabled=true,max=8,valid=true,optional=
[cfgVars=[Lcom.cisco.wfapi.util.WFNameValuePair;@14efaa66,script=customaa21.
Task id=17,000,000,007,Step id=0,Step Class=com.cisco.wfframework.steps.core
StepStart,Step Description=Start
...
```

In this example output some of the redundant information is removed in the middle and cropped the beginning of the date/time stamp. Most of the time, the most important information is at the end of each line.

Note: Many of these lines of output are brought down to a second and third line due to spatial reasons.

```
5:33.956 Step id=529,Step Class=com.cisco.wfframework.steps.core.StepComment
Step Description=/* Assumptions: User extensions ... */
5:33.957 Step id=3,Step Class=com.cisco.wf.steps.ivr.AcceptStep,Step
```

```
Description=Accept (contact: --Triggering Contact--)
5:33.977 Step id=129,Step Class=com.cisco.wf.steps.ivr.OutputStep,Step
Description=Play Prompt (contact: --Triggering Contact--, prompt: DP[500])
5:34.461 Step id=2,190,Step Class=com.cisco.wfframework.steps.core
.StepOnExceptionGoto,Step Description=On Exception(UndefinedPromptException)
Goto Error
5:34.463 Step id=1,649,Step Class=com.cisco.wfframework.steps.core
.StepComment,Step Description=/* Check for Alternate Greeting. ... */
5:34.464 Step id=1,650,Step Class=com.cisco.wfframework.steps.core
.StepCallSubflow,Step Description=Call Subflow -- checkAltGreet.aef
5:34.467 Step id=0,Step Class=com.cisco.wfframework.steps.core.StepStart,
Step Description=Start
5:34.468 Step id=78,Step Class=com.cisco.wf.steps.ivr.GetContactInfoStep,
Step Description=Get Contact Info (contact: --Triggering Contact--)
5:34.469 Step id=79,Step Class=com.cisco.wfframework.steps.core
.StepCreateHost,Step Description=aType = new com.cisco
.aesop.AltGreetType(language)
5:34.473 Step id=56,Step Class=com.cisco.wfframework.steps.core
.StepHost,Step Description=pExist = aType.isEnabled()
5:34.477 Step id=5,Step Class=com.cisco.wfframework.steps.core
.StepIf,Step Description=If ( pExist == true
5:34.478 Step id=3,Step Class=com.cisco.wfframework.steps.core
.StepEnd,Step Description=End
5:34.480 Step id=510,Step Class=com.cisco.wf.steps.ivr.HolidayStep,
Step Description=Is Holiday (Date: --Current Date--)
5:34.487 Step id=512,Step Class=com.cisco.wf.steps.ivr.BusinessHoursStep,
Step Description=Business Hours (date: --Current Date--,
time: --Current Time--, schedule: BusinessSchedule)
5:34.527 Step id=1,659,Step Class=com.cisco.wfframework.steps.core
.StepLabel,Step Description=Office Open:
5:34.529 Step id=1,660,Step Class=com.cisco.wf.steps.ivr.OutputStep,
Step Description=Play Prompt (contact:
--Triggering Contact--, prompt: OpenGreeting)
5:35.722 Step id=1,669,Step Class=com.cisco.wfframework.steps.core
.StepLabel,Step Description=Main Menu:
5:35.723 Step id=732,Step Class=com.cisco.wfframework.steps.core
.StepIf,Step Description=If ( attempts >=  MaxRetries  ) Then
5:35.724 Step id=2,195,Step Class=com.cisco.wfframework.steps.core
.StepAssign,Step Description=Set ErrorCode = "mainmenu"
5:35.726 Step id=2,259,Step Class=com.cisco.wf.steps.ivr.MenuStep,
Step Description=Menu (contact: --Triggering Contact--, prompt: MainMenu)
5:35.730 Step id=2,294,Step Class=com.cisco.wf.steps.ivr.ParseInputStep,
Step Description=Get Digit String(contact:
--Triggering Contact--, result digit string: TransferExt)
5:36.197 Step id=2,295,Step Class=com.cisco.wfframework.steps.core
.StepAssign,Step Description=Set TransferExt = "2" + TransferExt
5:36.198 Step id=2,296,Step Class=com.cisco.wf.steps.ivr.ExtensionToAddressS
Step Description=Extension To User (Extension: TransferExt, result user:user
5:36.214 Step id=2,297,Step Class=com.cisco.prompt.steps.CreateGeneratedProm
Step Description=Create Generated Prompt( telephone.number type, store in pr
5:36.229 Step id=2,298,Step Class=com.cisco.wf.steps.ivr.ImplicitConfirmStep
Step Description=Implicit Confirmation (contact: --Triggering Contact--)
5:41.971 Step id=2,301,Step Class=com.cisco.wf.steps.ivr.RedirectStep,
Step Description=Call Redirect
(contact: --Triggering Contact--, extension: TransferExt)
```

Now you can see each step as it executes. You cannot see how expressions evaluate, nor can you see any user input. If the script in front of you, you can follow where the script branches, with the knowledge of the digits you entered (in this case, the user dials in, dials an extension, and is transferred).

Use a filter in order to see the digits.

```
cue-3745-44a> show trace buffer long  | include "process digit"
3119 12/28 17:05:35.728 ACCN CMTS 0 process digit 2 tag=2
3119 12/28 17:05:35.943 ACCN CMTS 0 process digit 0
3119 12/28 17:05:36.195 ACCN CMTS 0 process digit 1
```

From this output you see that extension 201 is dialed.

If the versions of the CUE and the CUE script editor are both different, and you are trying to upload an AA script via GUI or CLI, the upload fails with the `Upload failed` error message. If both versions are different, then you need to recreate the script using the CUE editor of the version that is the same as your CUE. Or, upgrade the CUE to the version of the CUE script editor and run the script.

Appendix

Appendix 1 - System Prompts

This table shows system prompts that you can use.

Note: You cannot re-use all of the prompts that you hear in the Cisco Unity Express voicemail application in a custom script. Some of the prompts are only available via the voicemail script which you cannot customize via the Cisco Unity Express Script Editor.

System Prompt	Recorded Audio
SP[AA/AAWelcome]	"Welcome to Automated Attendant"
SP[AA/AAMainMenu]	"To enter the phone number of the person you are trying to reach press 1, To enter the name of the person you are trying to reach press 2, To transfer to operator press 0"
SP[AA/AAEnterExtn]	"Please enter the phone number and press the # key"
SP[AA/AACallingExtn]	"Calling extension"
SP[AA/AAPhoneReach]	"The phone number you are trying to reach"
SP [AA/AAOutOfServicePhone]	"Is currently out of service"
SP[AA/AANameDial]	"Spell the last name of the person you want to call followed by the first name. For letter Q, press 7, for Z press 9"
SP[AA/AACalling]	"Calling"
	"Sorry You are having trouble. Please stay on the

SP[AA/AASorry]	line and some will be with you shortly."
SP[AA/AAWant2Call]	"If this is the name of the person you are calling, press 1, to start over, press *"
SP[AA/still_there]	"Are you still there?"

Appendix 2 - Reset Default Traces

In order to reset the system (Cisco Unity Express 2.1.x and earlier) to default traces through the CLI, first disable all traces using the **no trace all** command, then paste this into the CLI:

```
trace ccn engine dbug
trace ccn libldap dbug
trace ccn subsystemappl dbug
trace ccn managerappl dbug
trace ccn managerchannel dbug
trace ccn subsystemjtapi dbug
trace ccn subsystemsip dbug
trace ccn stacksip dbug
trace ccn subsystemhttp dbug
trace ccn vbrowsercore dbug
trace ccn subsystemcmt dbug
trace ccn libmedia dbug
trace ccn managercontact dbug
trace ccn stepcall dbug
trace ccn stepmedia dbug
trace config-ccn sip-subsystem debug
trace config-ccn jtapi-subsystem debug
trace config-ccn sip-trigger debug
trace config-ccn jtapi-trigger debug
trace config-ccn http-trigger debug
trace config-ccn group debug
trace config-ccn application debug
trace config-ccn script debug
trace config-ccn prompt debug
trace config-ccn miscellaneous debug
trace voicemail database query
trace voicemail database results
trace voicemail database transaction
trace voicemail database connection
trace voicemail database execute
trace voicemail mailbox login
trace voicemail mailbox logout
trace voicemail mailbox send
trace voicemail mailbox save
trace voicemail mailbox receive
trace voicemail mailbox delete
trace voicemail message create
trace voicemail message dec
trace voicemail message delete
trace voicemail message get
trace voicemail message inc
trace webinterface initwizard inittrace ccn engine dbug
trace ccn libldap dbug
trace ccn subsystemappl dbug
trace ccn managerappl dbug
trace ccn managerchannel dbug
trace ccn subsystemjtapi dbug
```

```
        trace ccn subsystemsip dbug
        trace ccn stacksip dbug
        trace ccn subsystemhttp dbug
        trace ccn vbrowsercore dbug
        trace ccn subsystemcmt dbug
        trace ccn libmedia dbug
        trace ccn managercontact dbug
        trace ccn stepcall dbug
        trace ccn stepmedia dbug
        trace config-ccn sip-subsystem debug
        trace config-ccn jtapi-subsystem debug
        trace config-ccn sip-trigger debug
        trace config-ccn jtapi-trigger debug
        trace config-ccn http-trigger debug
        trace config-ccn group debug
        trace config-ccn application debug
        trace config-ccn script debug
        trace config-ccn prompt debug
        trace config-ccn miscellaneous debug
        trace voicemail database query
        trace voicemail database results
        trace voicemail database transaction
        trace voicemail database connection
        trace voicemail database execute
        trace voicemail mailbox login
        trace voicemail mailbox logout
        trace voicemail mailbox send
        trace voicemail mailbox save
        trace voicemail mailbox receive
        trace voicemail mailbox delete
        trace voicemail message create
        trace voicemail message dec
        trace voicemail message delete
        trace voicemail message get
        trace voicemail message inc
        trace webinterface initwizard init
```

NetPro Discussion Forums - Featured Conversations

Networking Professionals Connection is a forum for networking professionals to share questions, suggestions, and information about networking solutions, products, and technologies. The featured links are some of the most recent conversations available in this technology.

NetPro Discussion Forums - Featured Conversations for Voice
Service Providers: Voice over IP
7914 expansion module on CCME-the buttons are constantly RED - Feb 3, 2009 VIC-4FXS does not work in ISRs - Feb 3, 2009 Matching VoIP traffic - Feb 3, 2009 20 digits max - Feb 2, 2009 T.38 and FAX on CISCO gateways - Jan 30, 2009
Voice & Video: Voice over IP
Clarification needed LMS and OPS Manager on same Server - Feb 3, 2009 video conferencing ? - Feb 3, 2009 Unified Communication Manager - Feb 2, 2009 VideoConferenceManager - Feb 2, 2009 MeetMe time limit - Feb 2, 2009
Voice & Video: IP Telephony
cause code - Invalid call reference value - Feb 3, 2009

CUCM 6 call recording. Ask question - Feb 3, 2009 Multi tenancy dial plan sample - Feb 3, 2009 7965G patch upload - Feb 3, 2009 Incoming CALL - Feb 3, 2009
Voice & Video: IP Phone Services for End Users
Nokia E-Series - Feb 3, 2009 CCM 5.1(3d) - XML Parse Error [4] when accessing personal directory - Feb 3, 2009 Forced Authentication Codes - Feb 3, 2009 One 877 call get blocked - Feb 2, 2009 CME IP Phone Security via LSC - Feb 2, 2009
Voice & Video: Unified Communications
DUCS 1.2.5 - Feb 3, 2009 Conference Connection - Feb 3, 2009 MeetingPlace Express Phone View - Feb 3, 2009 CRA - Wrong trigger (HTTP) - Feb 3, 2009 Clarification needed LMS and OPS Manager on same Server - Feb 3, 2009
Voice & Video: IP Phone Services for Developers
How to cleanup display after service request - Feb 3, 2009 Get Phone Registration Status - Feb 2, 2009 Update password - Feb 2, 2009 Accessing CDR DB remotely - Jan 28, 2009 Ldap Provider - Jan 28, 2009
Voice & Video: General
Clarification needed LMS and OPS Manager on same Server - Feb 3, 2009 Webview - Busy Other Time - Feb 3, 2009 Faxserver: Interaction between Rightfax 9.3 and ATA 186, anyway for work? - Feb 2, 2009 Call Termination Cause Value = 47 - Feb 2, 2009 End User Configuration Questions - Feb 2, 2009

Related Information

- **Cisco Unity Express Release 1.1 Product Documentation**
- **Voice Technology Support**
- **Voice and Unified Communications Product Support**
- **Recommended Reading:Troubleshooting Cisco IP Telephony**
- **Technical Support - Cisco Systems**

Home | How to Buy | Login | Profile | Feedback | Site Map | Help

Contacts & Feedback | Help | Site Map