Mitel Outbound Campaign IVR Version 4

User Guide

June 2021

NOTICE

The information contained in this document is believed to be accurate in all respects but is not warranted by Mitel Networks™ Corporation (MITEL®). Mitel makes no warranty of any kind with regards to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The information is subject to change without notice and should not be construed in any way as a commitment by Mitel or any of its affiliates or subsidiaries. Mitel and its affiliates and subsidiaries assume no responsibility for any errors or omissions in this document. Revisions of this document or new editions of it may be issued to incorporate such changes.

TRADEMARKS

Contents

1. INTRODUCTION

The Mitel Outbound Campaign Interactive Voice Response (OCIVR) application is a Microsoft Windows service application intended to be run 24*7 to handle making calls to a group of parties to provide an IVR interaction (the campaign). A campaign is configured to use a list of numbers stored in a comma separated value (CSV) input file. The OCIVR can be configured to make several waves or "cycles" of calls separated by a configurable delay between cycles. In the first cycle, all parties will be called. Each additional cycle will attempt to call any parties who failed to answer or who were busy in the previous cycles. For each call made, the OCIVR can write the results of the call to a CSV output file. In addition to the details read from the input file, the application can also write the time of the call (or call attempt), the call results, and the number of call cycles made to the party as well as data captured from the called parties.

The application allows several Campaigns to be configured and run on any OCIVR Server instance at any time on your Mitel system. A Campaign consists of several pieces of data including:

- Input CSV file.
- Optional Do Not Call CSV file.
- Optional Output CSV file.
- Campaign specific Prompt(s) and/or a campaign specific script file.
- If desired, a destination to transfer called parties to if they request it. For example, to speak with an agent.
- If a transfer is configured, an optional call property setting allowing calls to provide context to other applications.
- Number of Simultaneous Calls to make at the same time.
- No Answer Timeout (in seconds).
- Number of Calling Cycles to execute.
- Delay between Cycles (in minutes).

The OCIVR supports silence detection technology. This allows campaigns to wait for answering machines or humans to stop playing or speaking before starting to leave a message or commence attempting to interact with the called party.

The OCIVR also supports Text to Speech (TTS) technology using the Microsoft Server Text-to-Speech engine. This can be used to render the campaign's prompts from prompt texts as well as virtually any customer specific textual data. Using scripts, the OCIVR supports the mixing of pre-recorded and TTS prompts. See Appendix A: Configuring the Text to Speech Support for details on downloading and installing the TTS engine and voices.

Beginning with version 4.0.0, the application has been converted to a client/server architecture and consists of two components:

- OCIVR **Server**
  A Windows service which runs all the time providing the server functionality.
- OCIVR **Admin**
  A fat client which supports the configuration, control, and monitoring of campaigns. Starting with version 4.1.0, an additional command line utility is installed as part of the Admin installation:
- OCIVR **Command**
  A command line utility supports starting, stopping, pausing, and resuming a campaign.

2. LIMITATIONS

In general, customers should avoid creating campaigns that attempt to call more than around 50,000 parties. Doing so may cause potential problems, especially with regards to trying to use the Admin to monitor and/or control the campaign.

While this is not a hard and fast rule as it also depends on the number of fields for each party, it is a reasonable upper limit to use. Campaigns requiring calling more than this number of parties should be broken up into multiple campaigns that use the same script and run sequentially.

3. INSTALLATION

The OCIVR Server must be installed on a separate Mitel Distributed Voice Server (DVS). If it is installed on the HQ server, TAPI problems might occur when HQ becomes busy.

 **Note!** Multiple instances of OCIVR Server can be installed on more than one Mitel servers. Each OCIVR server instance is managed independently of every other OCIVR server instances.

The OCIVR Admin can be installed on any PC that can connect through the customer's local area network to the OCIVR server instances. It can be installed on more than one PC. Multiple administrators can monitor and configure any or all the OCIVR server instances simultaneously.

3.1 UNINSTALLING A PREVIOUS VERSION OF THE APPLICATION

If you are upgrading from a version of the OCIVR prior to version 4.x, you must first save all existing campaign configurations manually from the previous version and then recreate those campaigns in version 4.x after the software upgrade. This is because of the new client/server architecture of this application in version 4.x.

 **Note**! The settings and script features are 100% compatible between version 3.x and 4.x so you should be able to easily reproduce campaigns that you defined with version 3.x.

Therefore, before removing version 3.x and installing this version you need to run the current 3.x version to review and document the configuration settings of any and all campaigns you want to port to version 4.x, as well as secure copies of all campaign related files, specifically script files, prompt files and CSV input files.

If you are also upgrading your application topology to fully utilize the client/server architecture in version 4.x, you must also update the file references when you recreate the campaigns. See the Configure OCIVR Server Service Using OCIVR Admin section for details. When you are ready with the configuration details and files, you can then uninstall your current 3.x version of the application. To uninstall:

1. Run the Windows **Add/Remove** Programs tool located in the Control Panel and r
2. Remove the **"Mitel Outbound Campaign IVR"** application.
3. You must manually remove the existing installation folder, by default, at *C:\Program Files (x86)\Mitel\MitelOutbound Campaign IVR*. You can either delete the entire directory and its contents of log files and other support files or save it to a different location.

You are now ready to proceed with installing the new version 4.x of the application, starting with the Creating the Route Point section.

3.2 CREATING THE ROUTE POINT

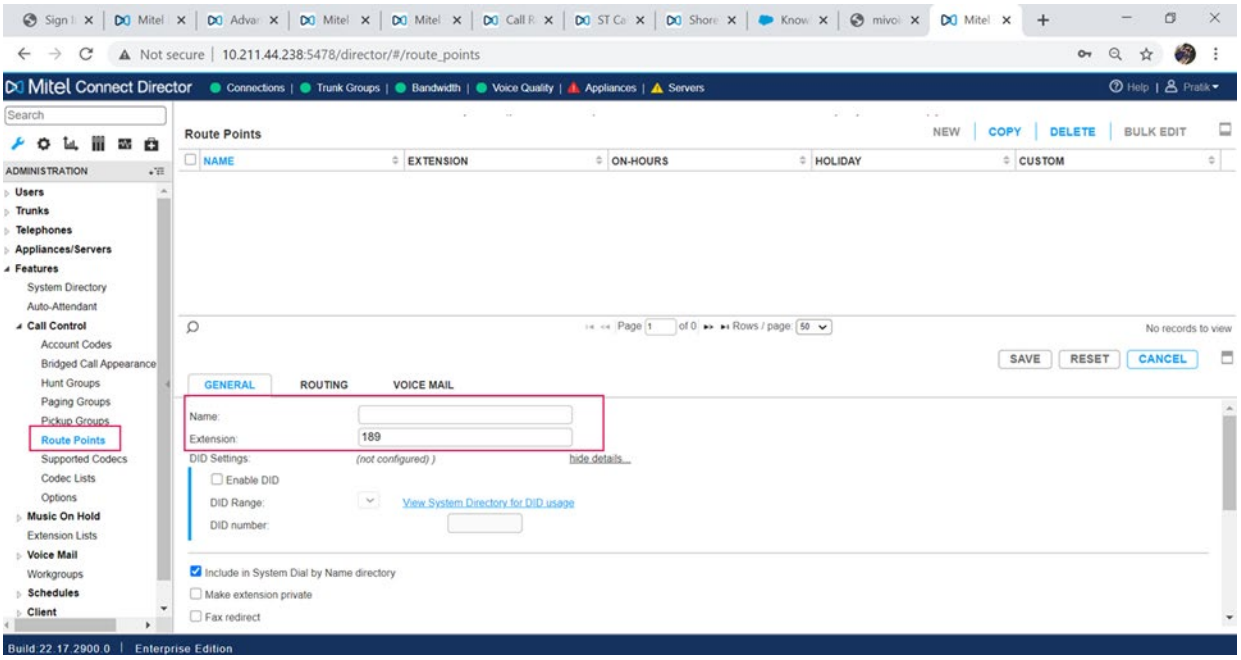
Each OCIVR server instance must be associated with a dedicated Route Point that is local to the Mitel Server where the OCIVR server instance is installed. Route Points are created in Connect Director **Administration > Call Control > Route Points > Add New**.

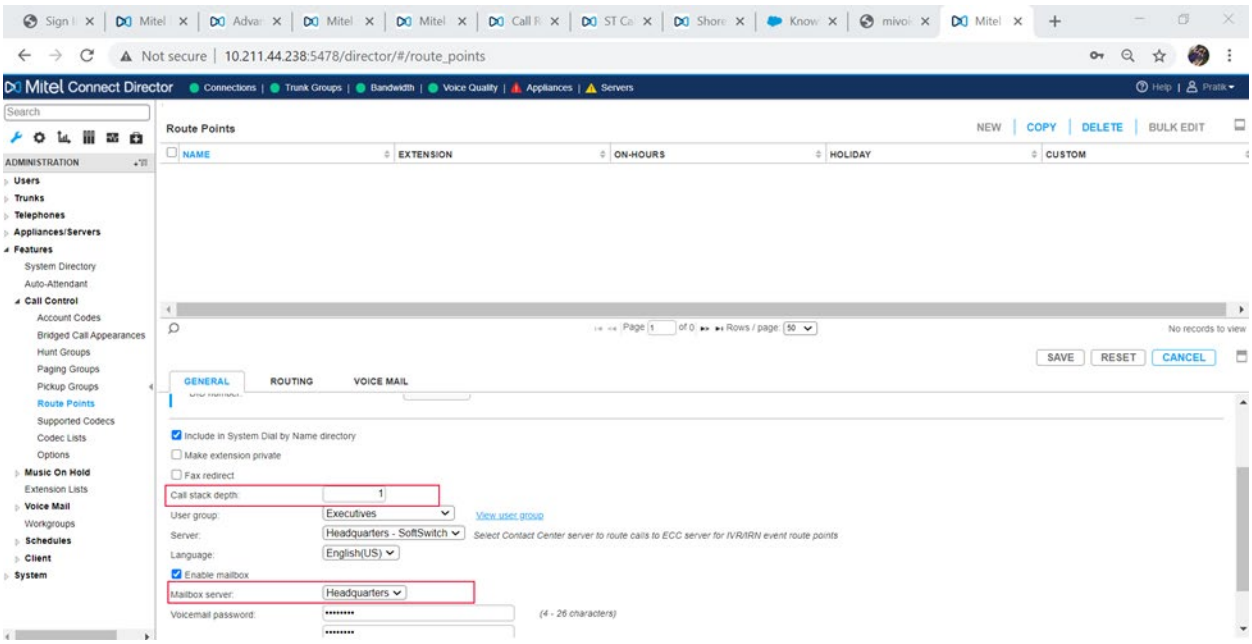
The Route Point should have a call stack size of at least the maximum number of outbound calls allowed for your campaigns. The default for a new campaign is 5 simultaneous calls but this can be easily changed when creating a new campaign or editing a campaign's settings. For example, if you expect to make up to 20 simultaneous calls you would want to set the call stack to at least 20. In addition, to support the admin feature which allows recording and playback of prompt files you will want to include an additional call stack item for each simultaneous OCIVR Admin user accessing the OCIVR server instance.

You must ensure that the Route Point Server is set to the Mitel server on which this application server instance is being installed. Also make sure that the User Group assigned to the Route Point can make external calls. You will probably want to disable the mailbox and forwarding of calls.

You will want to make note of the extension that you assign to the Route Point as you will be prompted for this when you install the application server instance although if you need to, you can easily change it later.

The figure below shows the important Route Point configuration fields within Connect Director:





3.2.1 INSTALL THE LICENSES

Before the OCIVR service can function, you must install the license using the Mitel Application License Server and the associated web-based Application License Admin. If you have purchased the OCIVR, then you should also have received a permanent license key. Alternatively, if you are trialing the OCIVR then you can install a trial license key good for 45 days and up to 5 simultaneous calls. The trial license key for the OCIVR is as follows:

```
Outbound Campaign IVR|5|Trial License| |0|3/25/2010
7:19:23|3/25/2010|5/9/2010|
|1|1440|zfT2tqFZhX2ZqZZAHA4XhQ|1.0|hOmqyWzX5nJ5cm8Vxe4jp
ryeKNwQdT nO6/Z/ipTLWk89TbWmjv2fzw==
```

Ensure that while installing the license that you copy and paste the complete license key starting with the words **“Outbound Campaign IVR”** through the double equal sign at the end of the key. While the trial license key allows up to 5 simultaneous calls, it will be automatically updated with the correct license count after you purchase the product and install the permanent license key.

3.3 CALCULATE LICENSE REQUIREMENTS

Licenses are shared by all the configured OCIVR server instance. Each OCIVR server instance requests several licenses based on its configured Maximum Calls settings. In theory, the sum of all OCIVR server instances. Maximum Calls setting should be less than or equal to the number of OCIVR licenses installed in the license server. If a given OCIVR server instance is configured for more licenses, then are available then it will get less than what it requested. After a server has retrieved its licenses, these licenses will always be provided to the server instance. To change the license count for an OCIVR server instance, it must be done through the Mitel Professional Services licensing admin.

3.4 CALCULATE SIMULTANEOUS CALL CAPACITY

Each instance of the OCIVR can place simultaneous calls to more rapidly complete calls to all the parties in a campaign. The number of simultaneous calls is controlled by the lesser of the configured Number of Simultaneous Calls for the campaign and the number of licenses retrieved by the OCIVR server instance which is normally the same as the server's configured Maximum Calls setting. For example, your OCIVR server instance might be configured with a count of 10 for the Maximum Calls setting but a given campaign on an OCIVR server instance could be configured to only place 5 simultaneous calls leaving 5 licenses free to run additional campaigns on the same OCIVR server. If multiple campaigns are run at the same time, then they will compete for the available licenses on the same server with the result that the number of calls actually made by a given campaign will vary over time as other campaign calls complete.

To calculate how many licenses you want to configure for the application, you must have an estimate for the three values for each campaign on all OCIVR server instance:

Parameter	Description
TOTAL_CALLS	The total number of parties you will be calling.
CALL_LENGTH	The average length of each call from the point where the call is made through when it disconnects.
TOTAL_TIME	The total amount of time that you would like the campaign to take before it completes calling all the parties.

To calculate the desired number of simultaneous calls or licenses the formula would be:

*LICENSES = (TOTAL_CALLS * CALL_LENGTH) / TOTAL_TIME*

So, with these numbers, one campaign can require 9 licenses as below:

Parameter	Required Value
TOTAL_CALLS	2 minutes.
CALL_LENGTH	1000
TOTAL_TIME	60 minutes

*LICENSES = (1000 * 2) / 60 = 2000 / 60 = 33.3 ≈ 34 Licenses*

Then, you must consider simultaneity among campaigns on all server instances and system resources to properly adjust or maybe simply sum up the license counts from simultaneous campaigns to have a good total license count for all campaigns on all your OCIVR server instances.

3.5 INSTALL THE OCIVR SERVER

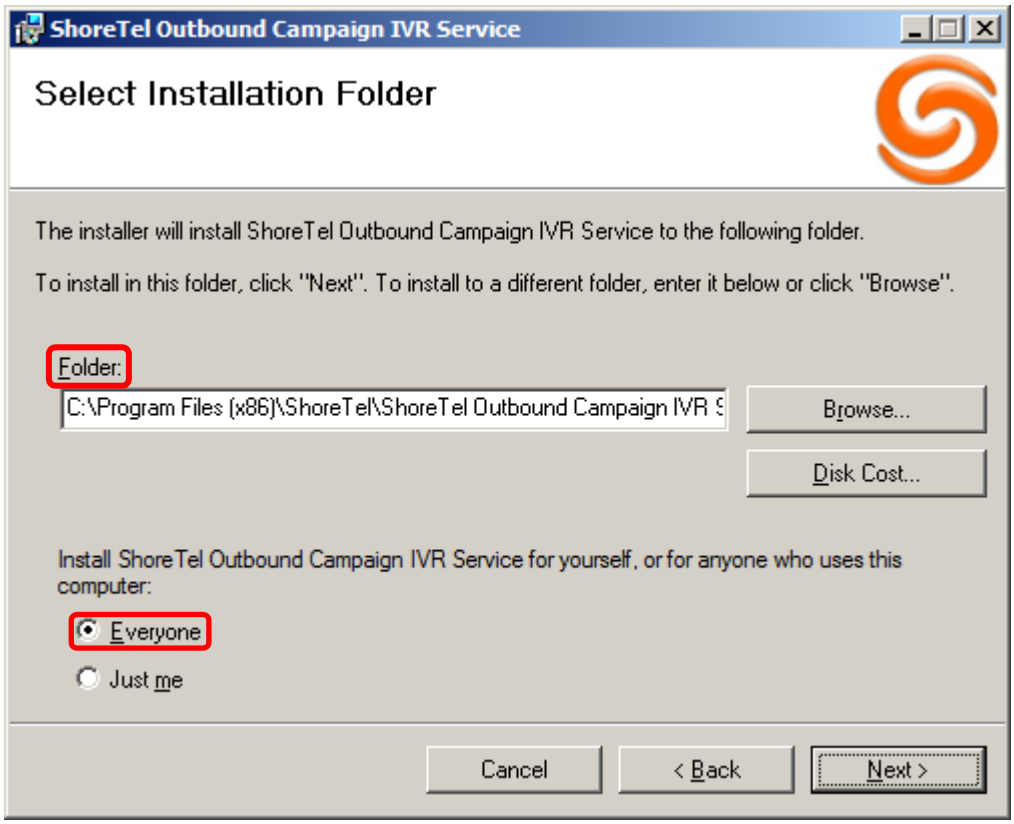
In addition to this documentation, Mitel supplies the setup executable and associated MSI file in the form of a zip file named *STPSOutboundCampaignIVRServiceSetupX.Y.Z.zip* in the application software package.

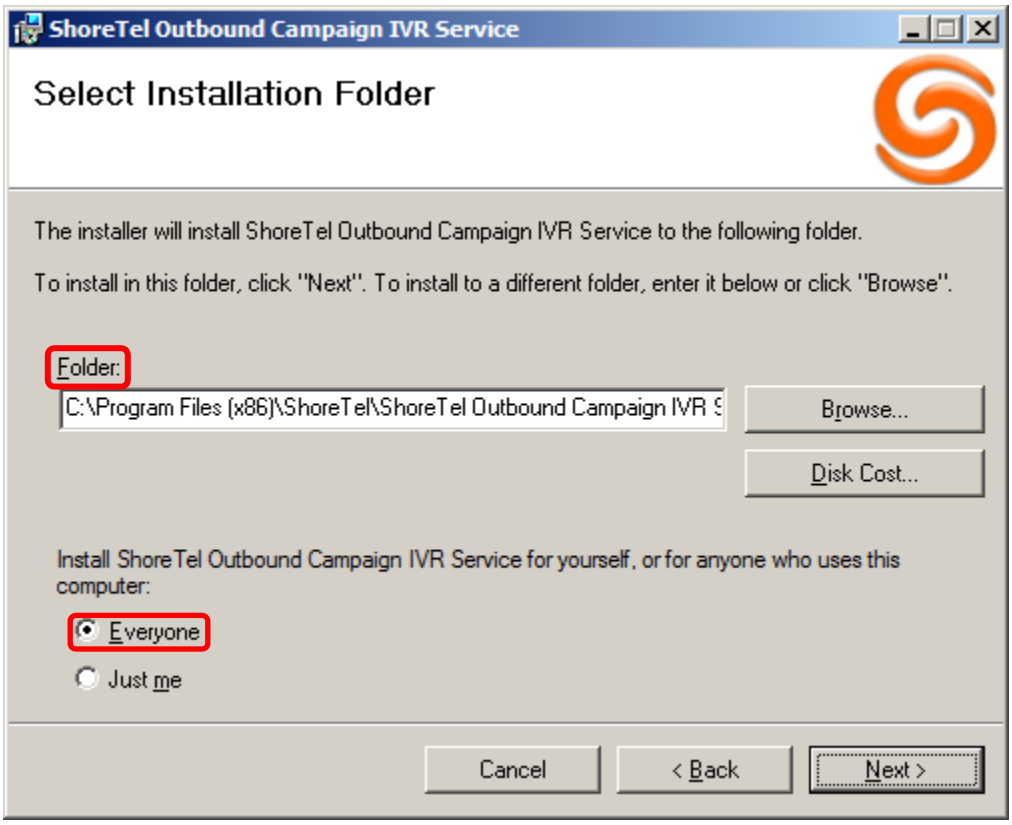
To install an instance of the application server on any Mitel server, follow these steps:

1. After unzipping *STPSOutboundCampaignIVRServiceSetupX.Y.Z.zip* to a folder, run the **STPSOutboundCampaignIVRServiceSetup.exe**.

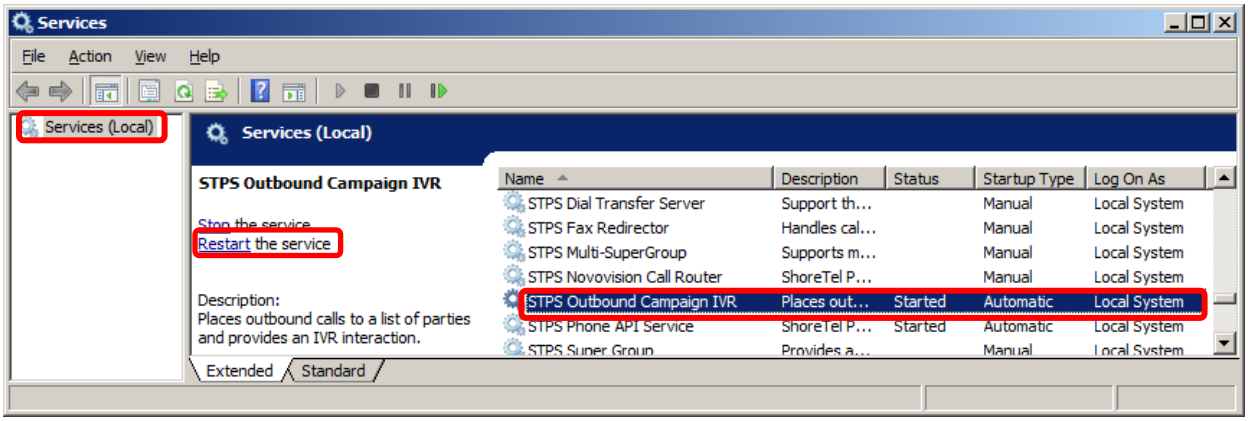
   Note! Do not run the setup executable directly from the zip file.

2. At the initial welcome screen, select **Next**. Then set and/or verify the installation folder and follow the screen by clicking **Next** several more times to complete the installation:





3. The OCIVR Server service is now installed. You can verify that the service has been installed by running the Windows Services application and looking for the service named **STPS Outbound Campaign IVR** as shown here:



4. The application starts running automatically and you will be able to perform the initial configuration after you install the separate OCIVR Admin program.

Note! By default, newly installed OCIVR server instance expects the OCIVR Admin clients to connect to it through TCP/IP Port 37733. However, if you need to change this port for this OCIVR server instance, then do the following:

5. Locate the file named "**app.config**" in the OCIVR Server installation folder, by default at *C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR Service* on the hosting Mitel server.
6. Edit the file using Notepad and locate the endpoint address that starts with ***net.tcp://localhost:37733…*** and change **37733** to the port value you want to use.
7. Restart the OCIVR Service to have the change take effect.

Note! If additional OCIVR services are required on other Mitel servers, repeat this installation process to install each additional OCIVR server instance on a separate Mitel server.

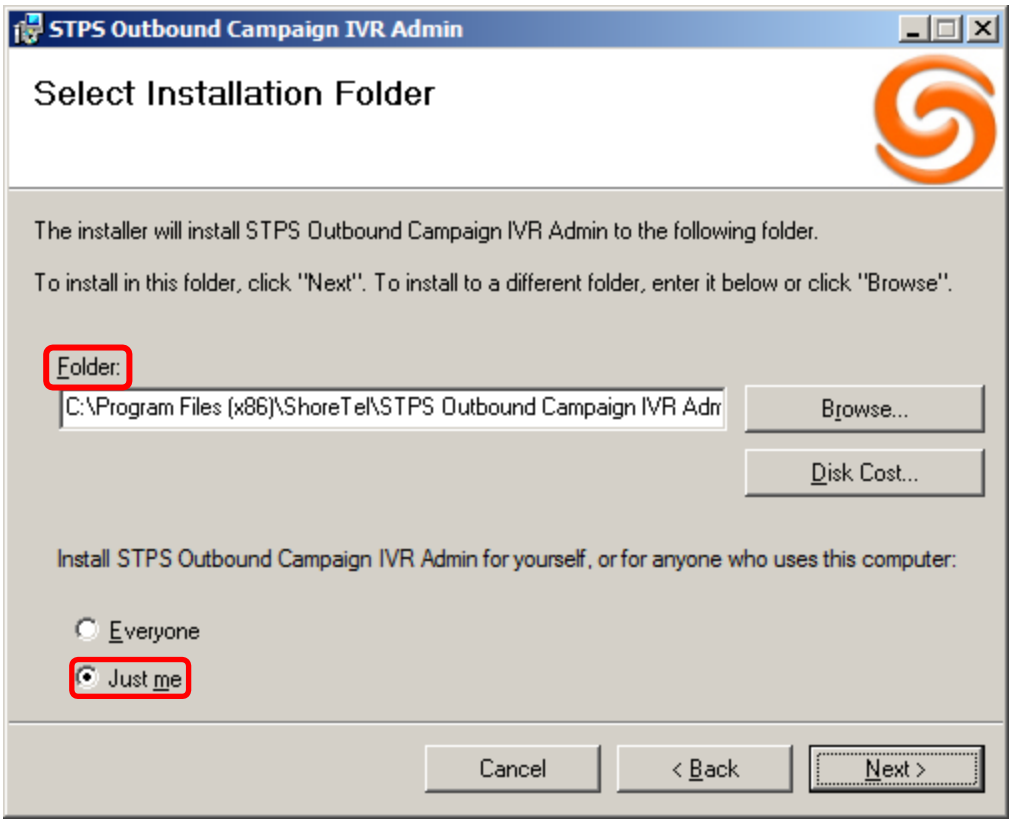
3.6 INSTALL THE OCIVR ADMIN

Mitel supplies the setup executable and associated MSI file in the form of a zip file named *STPSOutboundCampaignIVRAdminSetupX.Y.Z.zip* in the application software package.

To install the application admin on any Windows PC on your local area network, follow these steps:

1. After unzipping the setup files to a folder, run ***STPSOutbound CampaignIVRAdminSetup.exe***. Do not run *STPSOutboundCampaignIVRAdminSetup.exe* directly from the zip file.

   Note! Do not run *STPSOutboundCampaignIVRAdminSetup.exe* directly from the zip file.
2. At the initial welcome screen, select **Next**. Then set and/or verify the installation folder and follow the screen by clicking **Next** several times to complete the installation:



3. The Admin application is now installed. A shortcut labeled **STPSOutboundCampaignIVRAdmin** appears on your desktop as well as an installation folder at, by default, *C:\Program Files (x86)\Mitel\STPS Outbound Campaign IVR Admin* on the hosting Windows machine:

4. You can now run the OCIVR Admin to connect to and configure any of the OCIVR Service instance(s) over their TCP/IP ports on your Mitel system.
5. In addition to the admin, a separate command line utility is installed which allows starting, stopping, pausing, and resuming a campaign.

Note!

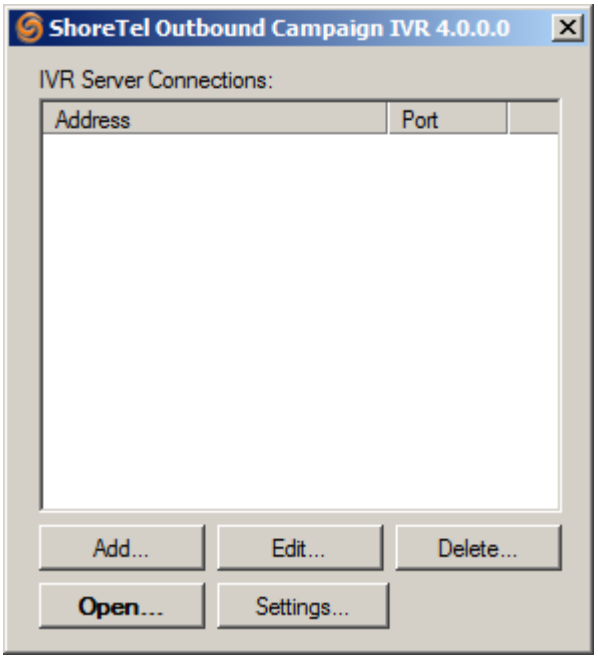
- You can now run the OCIVR Admin to connect to and configure any of the OCIVR Service instance(s) over their TCP/IP ports on your Mitel system.
- In addition to the admin, a separate command line utility is installed which allows starting, stopping, pausing, and resuming a campaign.

- If multiple copies of the OCIVR Admin and/or Command line utility are needed, repeat this process to install each additional copy on any Windows machine on your local area network. Multiple copies of OCIVR Admins can access OCIVR server instances simultaneously.

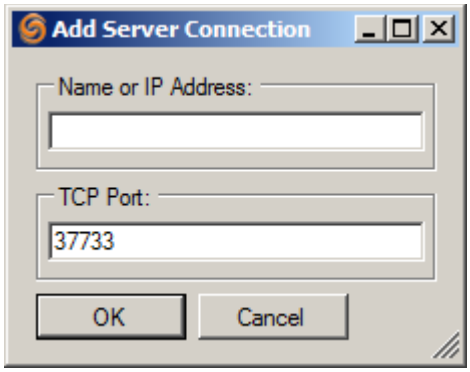
4. CONFIGURE THE OCIVR SERVER USING THE OCIVR ADMIN

The OCIVR service runs around the clock. To configure an instance of the service for the first time:

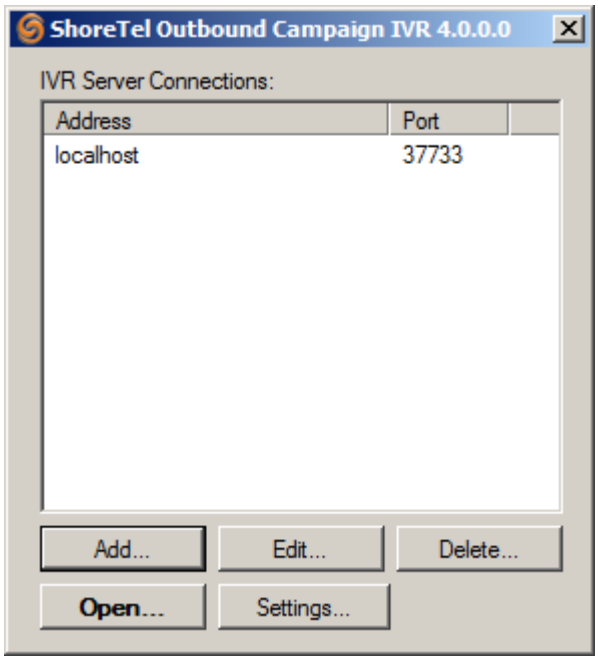
1. You must launch the OCIVR Admin program, create a connection to Mitel server hosting the OCIVR service instance, and then use that connection to access and modify the server instance's settings.
2. To launch the OCIVR Admin, either double click the **STPSOutboundCampaignIVRAdmin** desktop icon or click at **Start > Programs > Mitel group STPSOutboundCampaignIVRAdmin** shortcut icon to open the main OCIVR Admin screen which lists or will list all OCIVR Server Connections for this OCIVR Admin:



3. When first used, the list of OCIVR Server Connections is empty. In this page, you must add new connections, or the server instances you want to work with. You must create one connection to each Mitel server hosting one OCIVR server instance. To add a new OCIVR Server Connection, click **Add** to open the **Add Server Connection** dialog:



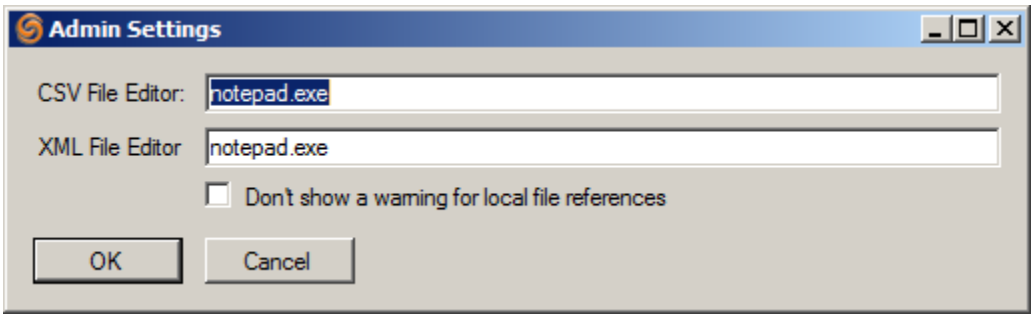
4. Enter the name or IP address of the Mitel server where the OCIVR service instance is or will be hosted. You can use the name “localhost” to connect to the service instance only if the OCIVR Admin and the server instance are running on the same Mitel server. The TCP Port setting should normally be left as the default value of 37733. If you have changed the TCP Port used by a given OCIVR service instance, you must use a different port value to connect the Admin to that server instance. The below figure shows the updated main OCIVR Admin screen after we have added a local connection to the service instance running on the same Mitel server:



In addition to creating a new Server Connection using the **Add** option, you can also delete a connection by selecting it and clicking the **Delete** option. You can change the OCIVR Server Connection name/IP address or TCP port of an existing Server Connection by selecting the Server Connection and clicking the **Edit** option.

Clicking the **Open** option or double clicking the Server Connection name opens the **Server Configuration** dialog for the OCIVR service instance with its IP address and TCP port shown in the title bar. OCIVR Admins use this dialog to construct and configure campaigns and launch campaigns with real-time control and status display. The OCIVR Server Configuration dialog is detailed in the Admin Server User Interface section below.

Clicking the **Settings** option displays the Admin Settings dialog:



These settings are specific to this admin user and for this OCIVR Server instance only:

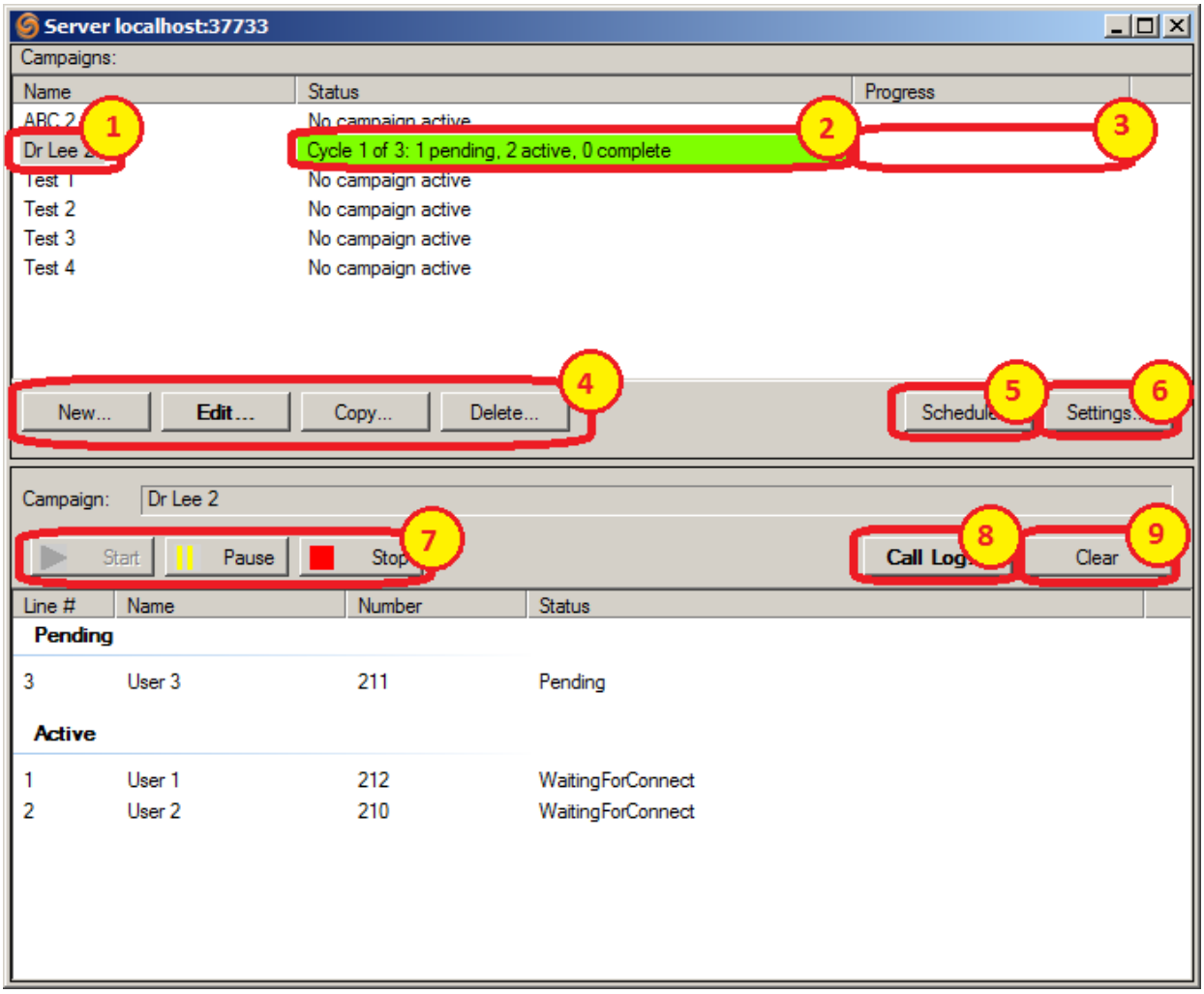
Field	Description
CSV File Editor	CSV File Editor is to define the file editor for Input and Output CSV files. *Notepad.exe* is the default setting for CSV File Editor to use the simple text editor built into Windows. When creating or editing a campaign (see Creating and Editing CSV Input File), you will notice an **Edit** option next to the Input and Output CSV file entry fields. Clicking the **Edit** option will open this editor and pass the CSV file name to the program. If this field is undefined, the **Edit** option will still function but the registered Windows Application for the file extension will be used. For example, if Microsoft Office is installed, the registered application to handle files of the *.CSV* type is Microsoft Excel.
XML File Editor	**XML File Editor** is selected to define the file editor for XML files. *Notepad.exe* is the default setting for XML File Editor to use the simple text editor built into Windows. When creating or editing a campaign using script (see Developing Scripts section below) you will notice an **Edit** option next to the **Script** radio button in the middle of the **Campaign Configuration** dialog box. Clicking the **Edit** option will open this editor and pass the XML script's file name to the program. If this field is undefined, the **Edit** options will still function, but the registered Windows Application for the XML file extension will be used.
Don't show a warning for local file references	This setting is to control if to repress the warning that drive relative paths are used when setting the CSV files, Script, or Prompt file paths. Checking this option will cause this warning message to be repressed.  **Note!** any file reference must be server relative. Normally, anytime you set a path to a drive relative location, you will receive a warning message when you go to save the settings. This option allows you to disable this warning if you either have this single OCIVR Admin locally on the Mitel server with the server instance or you know where the resources are located on the Mitel server.

The application always allows reference a file on the *C:* drive of the server containing, for example, the input CSV file with or without the warning, it is the OCIVR Admin's responsibility to ensure the correct file references for OCIVR files on the Mitel system. Using drive relative paths you may run into situations that you are not able to edit the Input CSV file using the admin programs **Edit** features or you are not able to use the file open dialogs to set the location of the CSV, Script and/or prompt files. The preferred approach is to always reference UNC names (names in the format **\\server\sharename\folder\file**) rather than

server drive relative paths. Then, if the admin user(s) have access to the same shares, they can directly locate and reference the various campaign resource files (**CSV**, **Script**, and **Prompts**).

4.1 ADMIN SERVER USER INTERFACE

When you double click a Server Connection entry or select a Server Connection and select **Open** on the main OCIVR Admin dialog, the **Server Configuration** dialog opens for the OCIVR server instance in the title bar:



The Server Configuration dialog has two principle areas: A top pane and a bottom pane.

The top pane is the **Campaign Pane** which shows the currently defined campaigns. The bottom pane is the **Call Pane** which shows the overall call status associated with the selected campaign (if any).

Each of the numbered highlighted areas is described here:

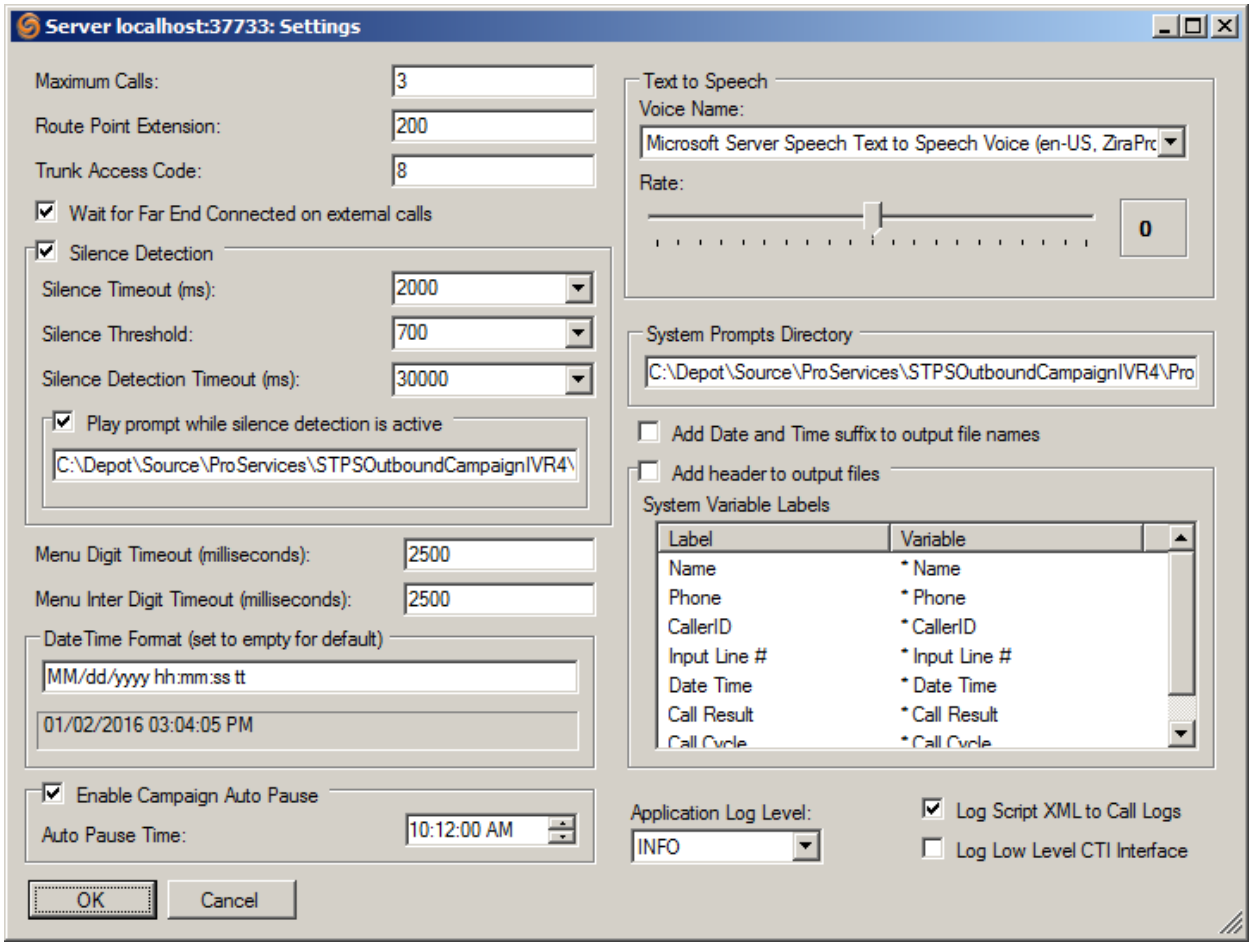
1. Shows the campaign name of the selected campaign
2. Shows the color-coded status overview of this campaign for its current run. If the campaign has never been run since the service was (re)started or if the status of the latest run has been cleared with the Clear option (see **#9**) then it shows "**No campaign active**".
3. Shows the campaign progress either by the percentage of calls that have been made relative to the total

number of calls or by the percentage of cycles relative to the total cycles if the campaign is pausing between cycles.

4. These controls allow a new campaign to be created (**New**) or the selected campaign to be edited (**Edit**), a copy made (**Copy**) or deleted (**Delete**). As a short cut, you can double click an entry in the Campaign Pane to edit it. See the New/Edit/Copy/Delete Campaign section below for details.
5. The **Schedule** option allows access to the Schedule Configuration dialog for all campaigns on this OCIVR server instance. This configuration allows campaigns to be run automatically on their respective weekly schedules rather than on demand. See the Schedule section below in Sample IVR Flow chapter on the Campaign Schedule feature.
6. The **Settings** option provides access to campaign settings of this server instance which apply to all campaigns on this server instance. For details, see the Application Settings section below on the server instance Settings feature.
7. These options at the top of the **Call Pane** allow a user to start the selected campaign in the Campaign Pane (**Start)**, pause an active campaign (**Pause**) or stop an active campaign (**Stop**).
8. The **Call Log** option opens the Call Log form which logs details of the currently selected call of the selected campaign. The call log is useful when testing a new campaign run and especially when fixing problems with the message script feature. As a shortcut the user can double click an entry in the **Call Pane** to open the Call Log form and view the call's details. This Call Log function is independent of the Application Log Level setting in the **Application Settings** dialog.
9. The **Clear** option allows you to remove the selected campaign's call details from the Call Pane. Starting a new campaign run will also clear the details of the previous run of the campaign from Call Pane. But all call details of all campaign runs remains on server *machine's C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR Service\Logs* folder if default installation location is used.

4.2 APPLICATION SETTINGS

Clicking the **Settings** option (see point 6 in Admin Server User Interface section) displays the **Application Settings** dialog for the hosting OCIVR server instance with its name/IP address and TCP port in the title bar. The **Application Settings** apply to all campaigns defined on this server instance:



This configuration screen allows modification to several application wide settings for this OCIVR server instance. The settings are as follows:

Field	Description
Maximum Calls	Sets the number of licenses that this OCIVR server instance should request. This sets the maximum number of simultaneous calls that this server instance can make. The license count is checked during campaign run(s), if the server instance is unable to retrieve any license then it will display a popup error.
Route Point Extension	Sets the extension of the Route Point used by this server instance exclusively for this application. The extension can either be set automatically at the time of installation or you will need to set it to a valid Route Point's extension.
Trunk Access Code	This setting specifies the Trunk Access Code for the OCIVR server instance. Upon making the outbound campaign calls, the system will use this Trunk Access Code if the number is 10 digits or longer. If the number is less than 10 digits, then it is assumed to be an

	internal number and is dialed without this Trunk Access Code.
Wait for Far End Connected on external calls	This setting controls if the application on this server instance should expect a trunk signal when the far end is connected. The setting should be enabled if the above Trunk Access Code routes through PRI trunks. The setting should be disabled if the above Trunk Access Code routes through analog trunks because analog trunks do not provide far end connection events. With PRI trunks, enabling this setting allows the application on this server instance to know if it has truly achieved a connected state. By default, this setting is enabled. You can test if your trunks provide this far end connected notification by creating a simple campaign which will play a message upon answer and providing a test number in the Input CSV file. Start the campaign with this setting enabled causing this application to call the test number. Answer the call on the test number and check if the OCIVR application considers the call connected by virtue of playing (or not playing) the configured message when you answer. Subsequently if you want to see the far end connected notification event, you can examine the Call Log for the call in Call Pane. If the far end connected event is provided, it will be logged there.
Silence Detection	Enabling **Silence Detection** allows the OCIVR to wait for the called party to stop talking by detecting a configurable length and quietness of a silence period before starting to play a message or script. This allows superior connection to answering machines while still working well when the call is answered by a person. When Silence Detection is enabled it takes place at the point when the outbound trunk connects or, if **Wait for Far End Connected on external calls** is enabled, when the far end connects. After **Silence Detection** starts, the campaign pauses until either the Silence Detection is successfully concluded or if the silence period cannot be detected then eventually the detection will timeout per Silence Detection Timeout and the message or script will play.
Silence Timeout (ms)	This field specifies how long silence should be detected on the call before a silence period is considered to have been achieved. If calls are placed with analog trunks where silence detection must start before the far end connects, then a good time is around 4 seconds (4000 milliseconds.) Because typical ring back is every 3.5 seconds this provides a margin of safety that prevents false positives. If calls are placed on PRI trunks with far end answer detection and **Wait for Far End Connected on external calls** enabled, then 2 seconds (2000 milliseconds) can be acceptable. The

	lower the value used, the faster the OCIVR concludes silence detection, therefore the more likely to think it has waited out the silence period when really it just experienced a pause in message audio from, perhaps, a voice mail system.
Silence Threshold	This is an audio level indication used by the OCIVR to decide if a given 100 millisecond packet of audio data should be considered silent. The higher the value the more tolerant the OCIVR is to background or noisy line conditions but too high a value could result in quiet speakers, tones or messages being mistaken for silence. The default value is 700. This number represents an arbitrary linear scale. To tune silence detection, you must follow these rules: • If the OCIVR messages are delivered before the called party had stopped “talking” then the threshold is too high. Try setting it 100 lower and try again. • If the OCIVR messages are not delivered or there is a long silence before the message starts, then the threshold is too low. Try setting it 100 higher and try again.
Silence Detection Timeout (ms)	This controls how long the OCIVR will try to conclude Silence Detection before giving up. This prevents the application from becoming stuck if it encounters a trunk with high noise levels or an answering machine with an old noisy tape even while recording.
Play Prompt when Silence Detection is active	This checkbox controls if the OCIVR server instance should play a prompt during Silence Detection. Normally when Silence Detection is active, the application will play a tone or prompt to the called party. This serves two purposes. First, if the campaign call is answered by a person, the prompt keep the person from speaking so Silence Detection can be concluded earlier, and second if the campaign call is answered by an answering machine it leaves some audio to be recorded rather than dead air. The wave file containing the tone or prompt to play is indicated in the edit field under the checkbox.
Menu Digit Timeout (milliseconds)	This value specifies in milliseconds how long the OCIVR application on this server instance waits for the first digit input for campaign prompts soliciting digit input(s). At several points of a campaign, an OCIVR flow or script can be configured to capture input from the called party. This setting controls the maximum number of milliseconds (thousandths of a second) the application on this server instance waits for the digit input from the end of the prompt till the first digit is received. Upon timeout, the application on the server

	instance can repeat the prompt, disconnect the call, or play a different prompt (see examples in Sample IVR Flow section covering IVR flows.) By default, this parameter is set to 2500 milliseconds or 2.5 seconds.
Menu Inter Digit Timeout (milliseconds)	This setting controls the maximum number of milliseconds (thousandths of a second) the application on this server instance waits between digit inputs from the called party. This is only used by the application when handling scripts which capture multiple digits in response to a prompt. You might prefer to give called parties more time to respond to the prompt with the first digit input but slightly less for subsequent digits. By default, this parameter is set to 2500 milliseconds or 2.5 seconds. Date Time Format When writing the "Date Time" variable to the output CSV file the server will default to the normal Windows ".net" format for the string. However, users may prefer to have their dates formatted differently from the default For example, to show leading zeros in dates and times, this format string could be used: "**MM/dd/yyyy hh:mm:ss tt**" Users can experiment by typing in format strings and observing the date time shown below the edit field for the sample datetime of January 1st, 2016 at 3:04:05 PM. For example, the format string above would result in this display: "**01/02/2016 03:04:05 PM**" There are a few web sites that you can search for ".NET DateTime Format" that provide details on datetime format strings. For example, both of these sites provide helpful details (set to empty for default): https://msdn.microsoft.com/en-us/library/az4se3k1(v=vs.110).aspx https://msdn.microsoft.com/en-us/library/8kb3ddd4(v=vs.110).aspx http://www.csharp-examples.net/string-format-datetime/
Enable Campaign Auto Pause	If checked, then the system will automatically pause all running campaigns at the configured Auto Pause Time. Using the Schedule feature, campaigns can be individually paused or stopped but this setting applies to all campaigns.
Auto Pause Time	This control allows the time of day that the auto pause of all running campaigns should occur. This time can only be set of the **Enable Campaign Auto Pause** is checked.

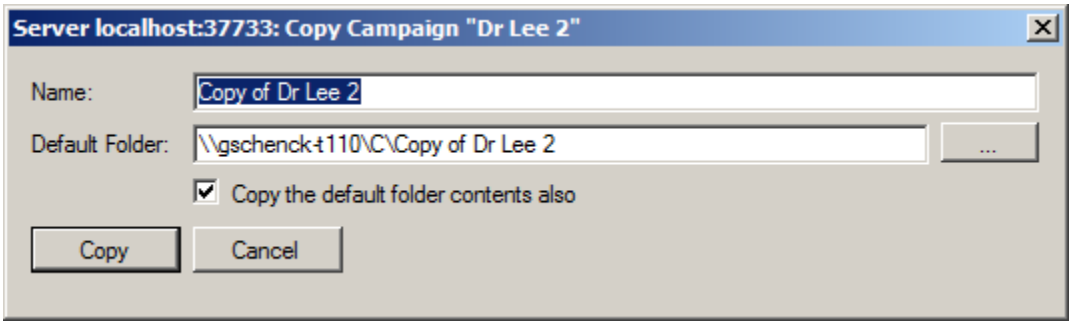
Text to Speech Voice Name	This combo box allows you to select the specific voice that the server should use when converting text to speech. For voices to show, the Microsoft Server Text to Speech engine being must be installed along with one or more voices. See Appendix A: Configuring the Text to Speech Support for additional details.
Text to Speech Voice Rate	This slider controls the relative speed of the audio generated from text using the Microsoft Speech API Text to Speech engine. It ranges from -10 to +10 as shown to the right of the slider. Note that scripts can also control the rate of the generated audio by using the ttsstartstyle and ttsendstyle node types (see Scripts section below). Rates set in the script are relative to the overall speed set with the slider.
System Prompts Directory	This field stores the default system prompt file location reference for all campaigns on this server instance. Campaign scripts may use a few pre-recorded prompt files containing speech segments such as numbers, letters and so on. If desired, you can customize and re-record the prompt files, store them in a new location, and update this field to point to the new prompt files rather than using the default prompt files. This is explained in more detail in the Scripts section on scripts.
Add Date and Time suffix to output file name	This checkbox controls whether the application on this server instance will automatically make every campaign output file name unique by appending a suffix of start date and time to the optional output file's name for every campaign run. By default, the optional campaign output file is named after the Optional Output CSV File provided in the Campaign Configuration dialog. This means if the same campaign is to run again with this checkbox unchecked, the current results will be replaced with the new results. The format of the Date and Time suffix **is _YYYY-MM-DD_HH-MM-SS**. So, if this option was enabled on the server instance when a campaign started at exactly 2 PM on January 14, 2010 and the Optional Output CSV File name was set with "Output.csv", then the output file for this specific campaign run will be "**Output_2010-01-14_14-00-00.csv**".
Add header to output files	This check box controls whether the application on this server instance will insert a default or customized header as the first line of the output file for all campaigns on this server instance. For example, if a campaign's Optional Output CSV File defines the output fields with **Name**, **Phone**, and **Result** and this checkbox is checked, then the default header line for this campaign output file would be:

	Name, **Phone**, **Result** The header line can be customized per server instance as detailed below under **System variable Labels**.
System Variable Labels	This shows the header line labels to be used for the system variables for all campaigns on the server instance if the Add header to output files option is enabled. The default is to use system variables without the asterisks and spaces in the header line. Admin users can customize the labels by selecting the label and then clicking it again to enter the edit mode to rename the label. Press **Enter** to save the change or press **Escape** to cancel it.
Application Log Level	This setting sets the OCIVR Log Level for this server instance to control the volume and the depth of information written into log files located in the OCIVR installation folder which by default is at *C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR Service\Logs*. The application provides a rich user interface including a feature to view the details of a given call (Call Log) in Call Pane of the Server Configuration dialog. In addition, the application writes log files to the Logs subdirectory of the application's directory. The volume of information written to the logs is controlled by this setting ranging from Off (no logging) through DEBUG (full logging.) Generally, the customer does not need to access this log unless working with a Mitel professional.
Log Script XML to Call Log	This setting controls the OCIVR script XML logging activity. If this setting is enabled, then the XML for each script statement executed will be shown in the call log. This setting is independent of Application Log Level setting.
Log Low Level CTI Interface	This setting controls the low-level CTI interface logging activity. Low level CTI Interface logging is a seperate log feature in addition to the Application Log. If the Log Low Level CTI Interface is enabled, it provides details of the inner workings of the Computer Telephony Interface (CTI). This setting is independent of Application Log Level and should only be enabled if requested by Mitel professionals.

4.3 NEW/EDIT/COPY/DELETE CAMPAIGN

The **Edit**, **Copy**, and **Delete** option on Server Configuration dialog page are enabled whenever the selected campaign is not running. The **New** option is always available in the dialog.

1. To create a new campaign, click the **New** option. See the Configuring Campaigns section below on configuring a campaign.
2. To edit an existing campaign, select it in the Campaign Pane and click **Edit**. See the Configuring Campaigns section below on configuring a campaign.
   To copy an existing campaign, select it and click **Copy** to open the following dialog:



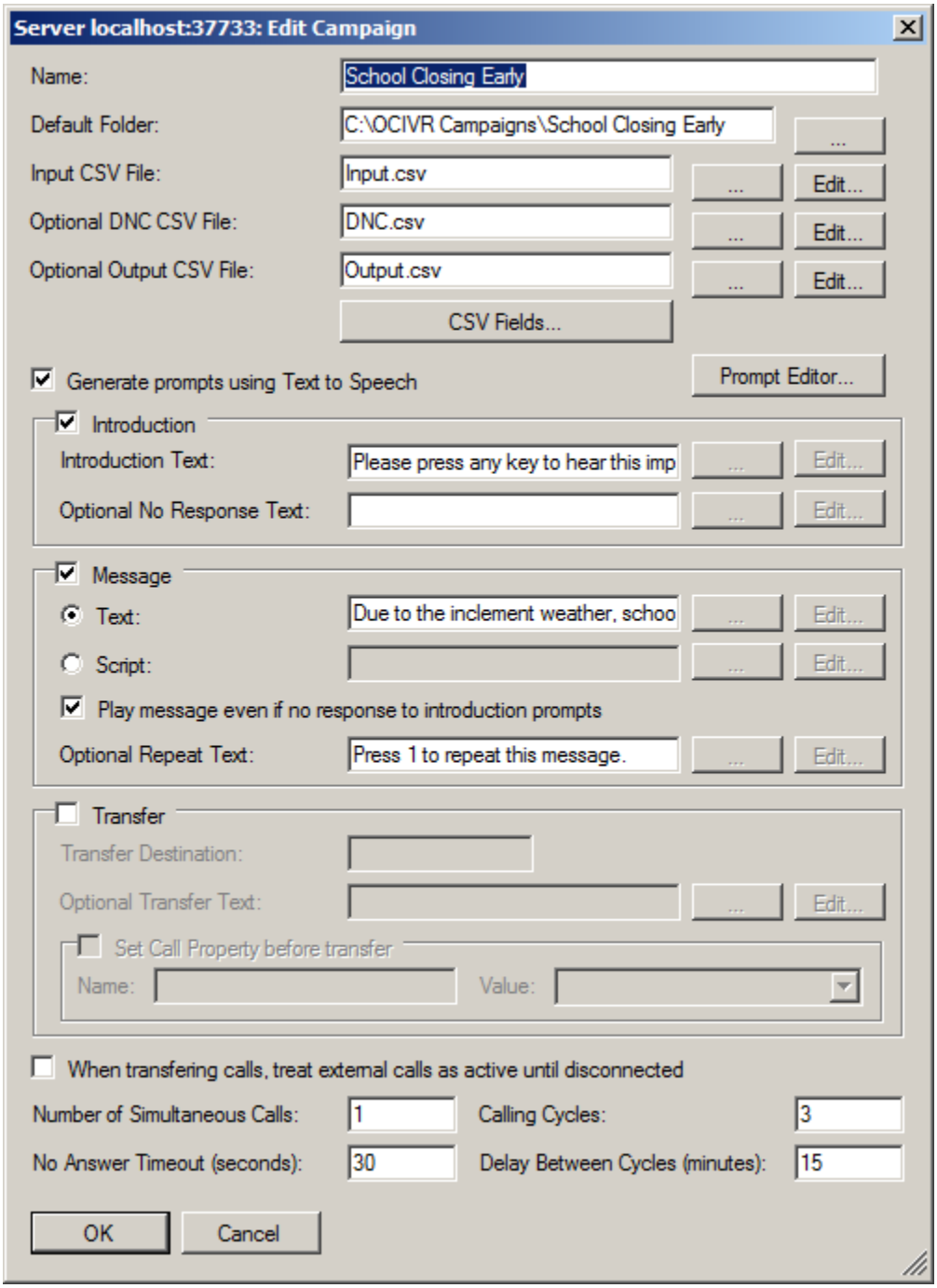
   By default, the new copy will be named as **"Copy of"** the existing campaign. If the existing campaign has a default folder, then the new campaign will have a default folder named as **Copy of** the existing default folder which is placed under the existing folder as a new subfolder. You must set these names and file structures as appropriate for the new campaign. Finally, if you want to copy any resource files from the existing campaign, then ensure that the **Copy the default folder contents also** checkbox is selected.

3. To delete a campaign, click **Delete**.

Note! Any campaign's resources (Input CSV file, script file, and prompts) will never be automatically deleted by the OCIVR application. You must manually delete a campaign's resources by using Windows Explorer or any appropriate tool.

4.4 CONFIGURING CAMPAIGNS

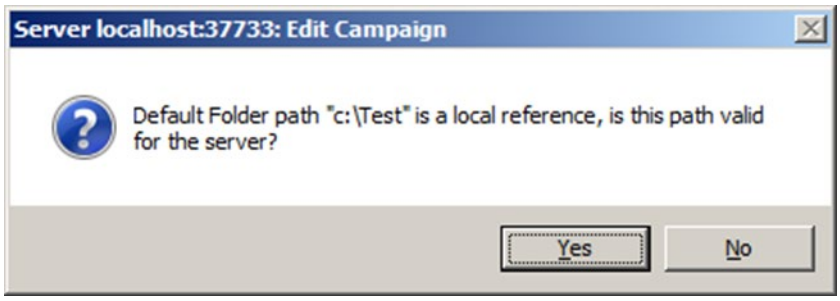
Selecting **New** or **Edit** opens the **Campaign Configuration** dialog:

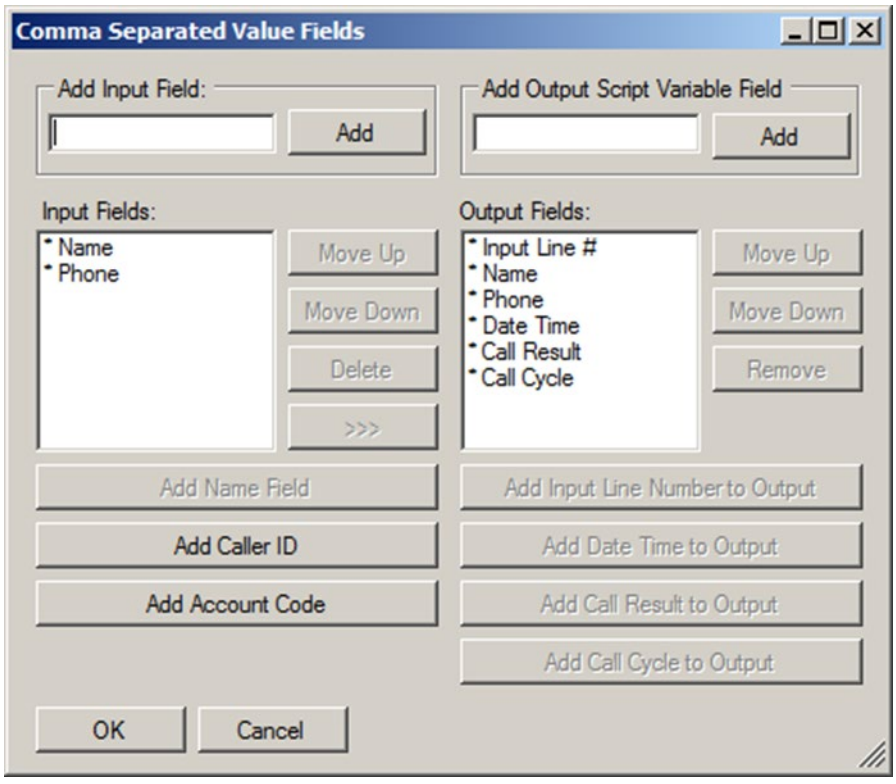


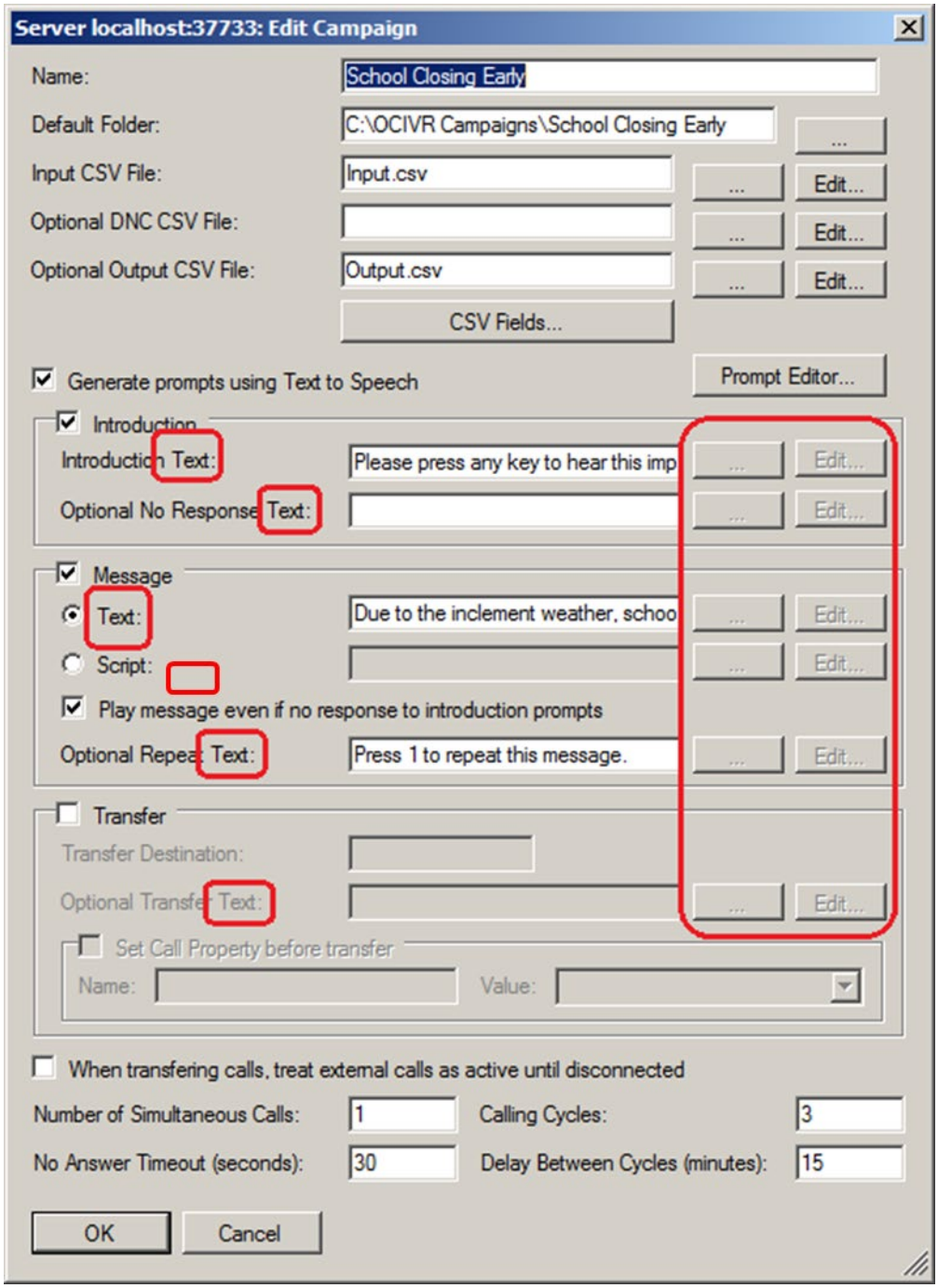
The title of the dialog indicates either **New Campaign** or **Edit Campaign** in addition to the name/IP address of the server instance and its TCP port.

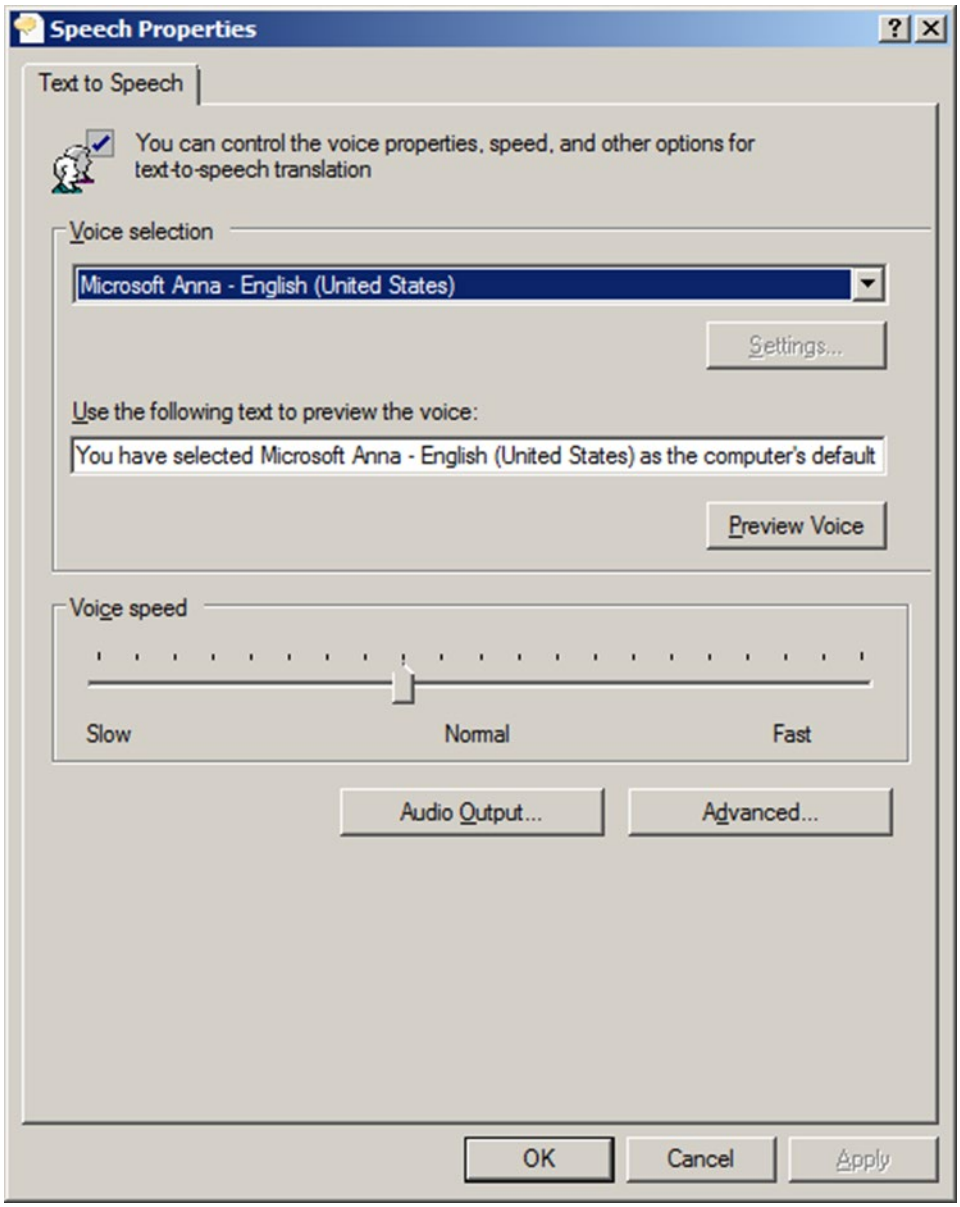
The fields that define a campaign are as follows:

Field	Description

Name	This is the name of the campaign. You can give the campaign any name you want, and it will show up in the first field in the Campaign Pane. Campaign names must be unique per server instance.
Default Folder	If set, then all the campaign resources which don't provide a rooted path will be treated as storage locations relative to this folder. In general, you must add the campaign's resources in a single folder structure and use UNC based relative paths to reference them. This allows the campaign contents to be easily copied and referenced. Setting the campaign Default Folder to a non-UNC based location will cause the following dialog to appear when the campaign settings are to be saved. The dialog is to remind you to double-check on the resource location and reference:  You will also see this dialog if any of the other resource references are not relative paths or UNC based references. If you are running the single OCIVR Admin to access the local OCIVR server instance on the server PC, then you must disable this warning message through the **Admin Settings** dialog accessed from the main OCIVR Admin dialog for the specific OCIVR Server Connections.
Input CSV File	Input CSV file stores the call number list and other input data for the campaign. The application places calls to each number in this CSV file. See the CSV Fields Dialog section below on information regarding creating and editing the CSV Input File. You can either directly enter the path and file name of the input file in the edit box, or, click at the ellipsis option (**...**) next to **Edit** to display a Windows file open dialog, select the directory, and the file name. Note that  **Note!** The system does not require that the file exists until you try to start the campaign. This allows you to create or modify a campaign and then, later, create the actual input file. If you click **Edit**, then either the configured CSV files editor (as set in the Admin Settings dialog) or the registered Windows application to handle the file extension will start. By default, the input file is assumed to contain two fields per line, the customer name, and their phone number. However, by using the **CSV Fields** dialog you can customize the fields and their orders for the input file.
Optional DNC CSV File	This is an optional parameter. The **Optional DNC CSV File** (DNC stands for Do-Not-Call) is used to provide a list of phone numbers which should not be called by the campaign. You must not include those numbers in the Input CSV file that you will not want to call. However, using the DNC file allows a separate list of numbers to be maintained and possibly used by multiple campaigns. This file is read by the OCIVR when a campaign is started or resumed following a pause. This allows do not call numbers to be added after a campaign has started by pausing the campaign, updating the DNC file, and then resuming the campaign. You can either directly enter the path and file name of the input file in the edit box, or click at the ellipsis

	option (**...**) next to the **Edit** to display a Windows file open dialog, select the directory, and the file name. **Note!** The system does not require that the file exists until you try to start or resume the campaign. If you click **Edit**, then either the configured CSV files editor (as set in the Admin Settings dialog) or the registered Windows application to handle the file extension will start. By default, the input file is assumed to contain two fields per line, the customer name, and their phone number.
Optional Output CSV File	This is an optional parameter. The **Optional Output CSV File** field is to store the campaign run results. If the parameter is left blank, the application will not write any output file for this campaign. If it is set, the application can write the results of a campaign run to this file. See the CSV Fields Dialog section below on the Output CSV File for details of what is written and when. For example, in the Input CSV file, you can either enter the full file path in the edit field or click the ellipsis option (**...**) to use the Windows file open dialog to select the directory and file name. The Output CSV File is editable, however, it is only used as an easy way to view the results of a campaign run after it has completed. By default, the output file has the following fields written for each call made: The **Input line number**, **Name**, **Phone number**, **Data**, **the time of the call**, **and the results of the call**. However, using the **CSV Fields** dialog you can customize the fields by adding or removing fields from the input fields, or change their orders. In addition, if the campaign solicits digit input, the message script can also write out data collected from the called party in one or more output fields. See the Scripts section below for more details.
Comma Separated Value Fields	Clicking this option displays the CSV fields that are specific to this campaign. New campaigns start with these as the default CSV fields.  See the CSV Fields Dialog section below for more details.

Generate Prompts using Text to Speech	This checkbox controls if Text to Speech (TTS) is to be used for this campaign. Selecting this option causes the various prompt fields in this dialog to be treated as prompt text to be converted to audio using Text to Speech rather than pre-recorded prompt file names. TTS support requires that the Microsoft Server Text to Speech engine be installed along with at least one voice. See Appendix A: Configuring the Text to Speech Support for details. When checked, the labels for the prompt fields change from **Prompt** to **Text** and all their **Ellipsis** and **Edit** option are grayed out.  In addition, if this is checked and the campaign employs a script then the script defaults to a <ttson/> setting when started (see the Scripts section below). Note that for Text-to-Speech to function, the user must have the Microsoft Text to Speech (SAPI) engine installed.

	Note! For Text-to-Speech to function, the user must have the Microsoft Text to Speech (SAPI) engine installed. This can be verified by checking in the **Start** > **Control Panel** > **Text to Speech** applet as shown here: 
Prompt Editor	Clicking at this option opens the **Edit Prompts** dialog. The **Edit Prompts** dialog allows prompt files to be played and recorded directly from the application. This option is functional regardless of the Generate prompts using **Text to Speech** option status. This feature is detailed under **Edit Prompts Dialog** in the Creating and Editing Prompt Files section below.
Introduction	This checkbox controls if the Introduction feature is to be included in the campaign flow. Introduction feature verifies if a campaign call is answered by a person by playing the Introduction Prompt to solicit any digit input from the called party. It is possible that an answering machine answered or, depending on the trunk type, the call is still ringing and has not been answered. Enabling this option and setting an

	Introduction prompt file or prompt text will cause the system to play a message that should ask the called user to press any key to accept the call. This message is repeated up to three times. If the called party does not respond with any digit press, to accept the call and if the **Optional No Response Prompt/Text** is set, then the **No Response Prompt** will be played before OCIVR considers the call Complete.
Message	This checkbox controls if the Message feature is to be included in the campaign flow. The Message feature is the core feature of a campaign, that is, playing the campaign Message Prompt to the called party. If **Message** is selected and **Generate prompts using Text to Speech** option is not enabled, you can enter the file path in the edit field or select the file through the Ellipsis (**…**) option like the Input and Output CSV files. You can also edit or review the current prompt file by clicking the **Edit** option. If Message is selected, and the **Generate prompts using Text to Speech** option is enabled, you can enter the prompt text directly in the edit field to be converted to an audio prompt upon running. There are two basic types of Messages that can be played: • **Prompt with single prompt file or prompt text** • **Script with multiple prompts** A prompt file is a single pre-recorded audio file. Prompt text is the text of the Message, it should be set if the **Generate Prompts using Text to Speech** is enabled. Using the prompt option of the Message feature allows a single pre-recorded or TTS-generated voice prompt to be played. This would be appropriate if the Message should be the same for all called parties, like a notification of an early dismissal at a school due to weather. A script is used to handle more complicated interactions such as: • Complex messages involving the stringing together of multiple prompts. • Playing data from the input file using concatenated prompts such as dates, times, currency amounts and called party specific prompts. • Capturing data from the called party and writing it to the output file as part of each called party record. The **Play Message even if no response to introduction prompts** checkbox can be enabled if the **Introduction** option is enabled and the **Optional No Response Prompt/Text** is not set in the Introduction area. Selecting this option will cause the OCIVR to play the Message even if the called party does not respond to the Introduction Prompts. This allows a message to be left hopefully on an answering machine. This option should not be used for a script which queries the called party for input as after three tries of the Introduction Prompts complete, any remaining prompts that might be played by the script will most likely be useless. The **Optional Repeat Prompt file or text**, if set, will play after the main campaign message completes to give the called party the opportunity to press **1** to hear the message again. The **Optional Repeat Prompt file or text message** option should tell the called party to press **1** to hear the message again. Note that the expected key '1' is hard-coded and can't be changed. All other digit input at this prompt will be ignored.  **Note!** The expected key **1** is hard-coded and cannot be changed. All other digit input at this prompt will be ignored.
Transfer	This checkbox controls whether the Transfer feature is included in the campaign flow so that the called party can be transferred to a representative following any message.

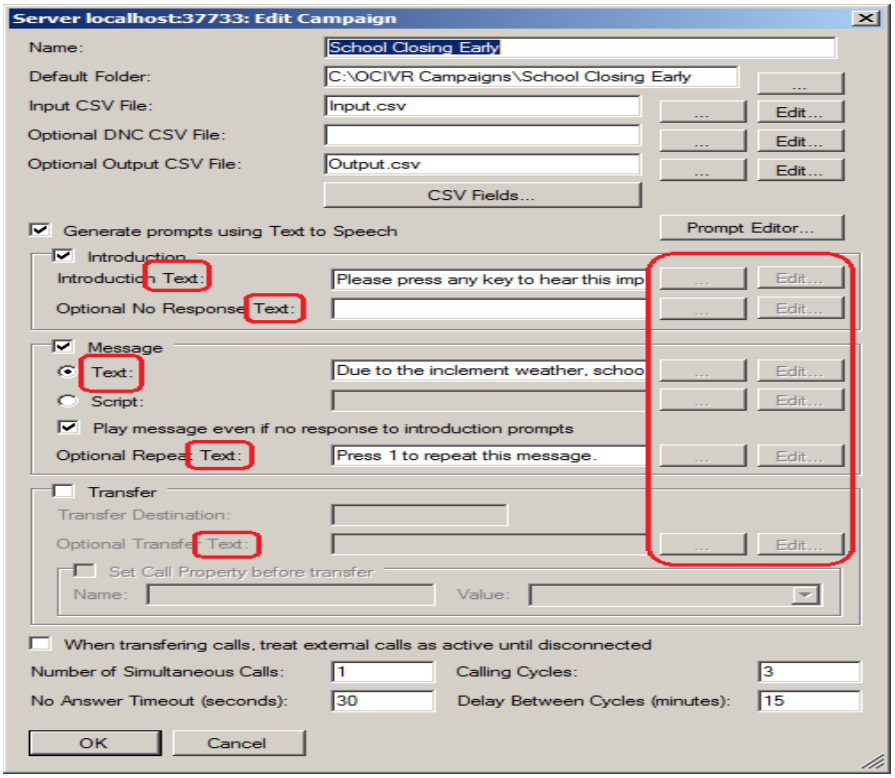
	Note! That while you can check both two options, **Message** or **Transfer**, you must check at least one. The **Transfer Destination** should be set to the extension to which called parties will be transferred as the campaign flows. Typically, this would be a Mitel Work Group but could be any valid extension such as an Auto Attendant menu, a Hunt Group, a Route Point, and so on. defined on your Mitel system. If the **Transfer** checkbox is checked and the **Optional Transfer Prompt** is set, the Optional Transfer Prompt is played from the prompt file or prompt text in the edit field to give the called party the option of being transferred. This Transfer Prompt message should ask the called party to press the asterisk (*) option on their phone to be transferred. If the called party does press the asterisk option, then they will be transferred to the configured Transfer Destination extension. **Note!** That the expected key * is hard-coded and can't be changed. All other digit keys are ignored. If no asterisk key is received, the application will retry by repeating the **Optional Transfer Prompt**. There are three tries in total before the call is disconnected. If the **Optional Transfer Prompt** is not set for the Transfer feature, then the transfer is automatic and the called party is not given the option. If the **When transferring calls, treat external calls as active until disconnected campaign** option is checked, then the OCIVR will continue to track the call after the transfer until it ultimately disconnects from the Mitel system. This will cause the OCIVR to hold off from making new call if it has reached the maximum number of allowed calls (see the next settings, Number of Simultaneous Calls.) This allows you to ensure that the OCIVR does not overwhelm the number of available agents or available trunks in those scenarios that include a transfer of the called party back into the Mitel system. If the **Set Call Property before transfer** option is selected, then just before the call is transferred, a call property with the specified name is set to the value of one of the specified input fields. This would be useful for OCIVR application to work effectively with other Mitel applications on your system, for example, setting the account number of the called party as a Call Property enables other Mitel client applications such as Mitel EasyPop to respond by (for example) opening the corresponding called party's record on the call in a web based application by forming up a URL containing the account number.
When transferring calls, treat external calls as active until disconnected	By default, when the OCIVR transfers a call it considers the call Complete and proceeds to make another call if allowed by the Number of Simultaneous Calls allowed and other OCIVR configurations. However, if this option is checked and the called party is an external party then the call will not be considered complete until it ultimately disconnects from the system. This will prevent the OCIVR from making a new call when the Number of Simultaneous Calls allowed has been reached. This option is very useful if the number of available agents or other system resources is limited and you want to guarantee agent or other resources availability at the time of the transfer.
Number of Simultaneous Calls	This field sets the maximum number of simultaneous calls allowed for the campaign. When OCIVR is running a campaign, it can place multiple calls at the same time to reduce the overall time it takes to call and interact with all the parties on the list. This setting controls the number of simultaneous calls it can make. For a new campaign this defaults to **5**.

	Note! The Route Point's call stack size must be equal or greater than this value for any campaign you run. See the Calculate Simultaneous Call Capacity section above on Licensing for a better understanding of how the OCIVR handles calling multiple parties for multiple simultaneous running campaigns across multiple OCIVR server instances.
No Answer Timeout (seconds)	This field sets the number of seconds the campaign waits for an Answer event for the campaign. When OCIVR calls a party, it needs the call to connect before it can start to deliver the first prompt. Depending on the application setting of Wait for Far End Connected on external calls for the OCIVR server instance (see Application Settings section above), the Answer is either a trunk level connection or a Telephone Company provided Far End connection state. If the outbound call fails to connect in either way within this period, then the call is completed with a **NoAnswer** result.
Calling Cycles	The OCIVR can make repeated attempts to call parties who are busy or who do not answer. See the Sample OCIVR Flow section below on OCIVR flows. The number of times or cycles that the IVR tries to call parties is controlled by this setting. If set to one, the OCIVR will call each party once and, when done with the call list, the campaign will be complete with one Calling Cycle. If set to a value greater than one, then after the first Calling Cycle, the OCIVR will pause for the **Delay Between Cycles** (see the next setting) and when that time expires it will then reinitiate calls to any **Busy** or **NoAnswer** parties from the previous cycle as the second Call Cycle. The campaign completes after all Calling Cycles specified are executed.
Delay Between Cycles (minutes)	This is the delay, in minutes, after one cycle completes and before the next cycle starts. It is only used if the number of Calling Cycles in the previous setting is greater than one.

4.5 CSV FIELDS DIALOG

This dialog is opened when editing a campaign by pressing the **CSV Fields** option. This allows campaigns to work with different input and output comma separated value formats.

This dialog allows you to define the content and order of fields on each line read from the input CSV file and written to the optional output CSV file. When a new campaign is created, the default fields are:



4.5.1 INPUT FIELD SETTINGS

For the example above, each line of the Input CSV file must contain at least two fields (additional fields are ignored):

- The first field contains the name of the party to be called
- The second field contains the corresponding phone number of the party to be called.

Each line written to the Output file (if the **Optional Output CSV File** is set) contains 5 fields:

- The first field contains the line number of the corresponding entry read from the input file.
- The second field contains a copy of the called party Name read from the input file.
- The third field contains a copy of the called party Phone Number read from input file.
- The fourth field contains the date and time of the call made to the party.

To execute a campaign, you must define the field in the Input CSV file which contains the phone number to be called. Additional fields can be defined and would have to be if the phone number is not the first field to represent the order of fields relative to the phone number. If this is confusing, see the Example CSV File Configuration section below for more details. The fifth field will contain the result of the call.

CSV fields are of two types, system fields and user fields. The system fields begin with an asterisk and are entered using the buttons in the dialog. The system fields are divided into two groups, those associated with the **Input CSV file** and those with the **Output CSV file**. Input fields can also be copied to the Output CSV file. This is accomplished using the **>>>** option (**Copy** option) and allows copying a reference to a given field from the **Input Fields** to the **Output Fields**.

The system fields which are or can be read from the input file are:

Field	Description

* Phone	The **Phone** field is required. It is the only system field that cannot be removed. If you try to remove it, you will get a message reminding you that you must have at least a phone number field.
* Name	The **Name** field is given special treatment by the system by appearing in the Call Log section in Call Pane. It is typically the name of the party associated with the phone number.
* CallerID	If the **Caller ID** Field is included and defined in the input file then when the application dials a party's phone number it presents the call to the called party with this value as the caller ID reported by the telephone company instead of the caller ID reported by the system. Note that for the Caller ID setting to work, the trunks supplied by the telephone company must support caller ID. You must test with a few calls to known numbers to determine if the application can explicitly set the caller ID.
* AccountCode	If the **AccountCode** field is included and defined in the input file then when the application connects to the trunk it will generate the account code as DTMF digits in order to pass a customer specific account code an external account code service.  **Note!** This option does not work with the built in Mitel Account Code service.

In addition to the system fields you can also define user fields in the CSV files. Adding input fields serve several purposes:

Field	Description
Space fillers	If the phone number is not the first field in your input files, then must define additional fields which precede the phone number.
Fields to write to output	If you want to write a field read from the input file to the output file, then you must define its order among input fields so you can then place a copy in the output file.
Fields to be used by scripts	Scripts can use input fields as variables. For example, an appointment reminder script could use an input field named **Schedule Date** assumed to contain the date of an appointment to play to a called party. Using fields allows scripts to provide custom interactions based on their values at the time the Campaign is run.

4.5.2 OUTPUT FIELD SETTINGS

On the output side, you can also add your own fields. These fields can be set by a message script based on called party inputs in response to prompts. This allows campaigns such as surveys to be constructed. This feature is explained in more detail in the sections on scripting.

The system fields which can be written to the output file are:

Field	Description
* Input Line #	The line number of the Input CSV File entry which provides the details for this call.
* Date Time	The date and time that the call is placed by the system.
* Call Result	The specific result of this call. See Sample Campaigns section below on campaigns which describes the different possible results and their denotations.
* Call Cycle	The cycle that the call connected on. If the call never connected, then this value is set to the total call attempts made.

4.5.3 EXAMPLE CSV FILE CONFIGURATION

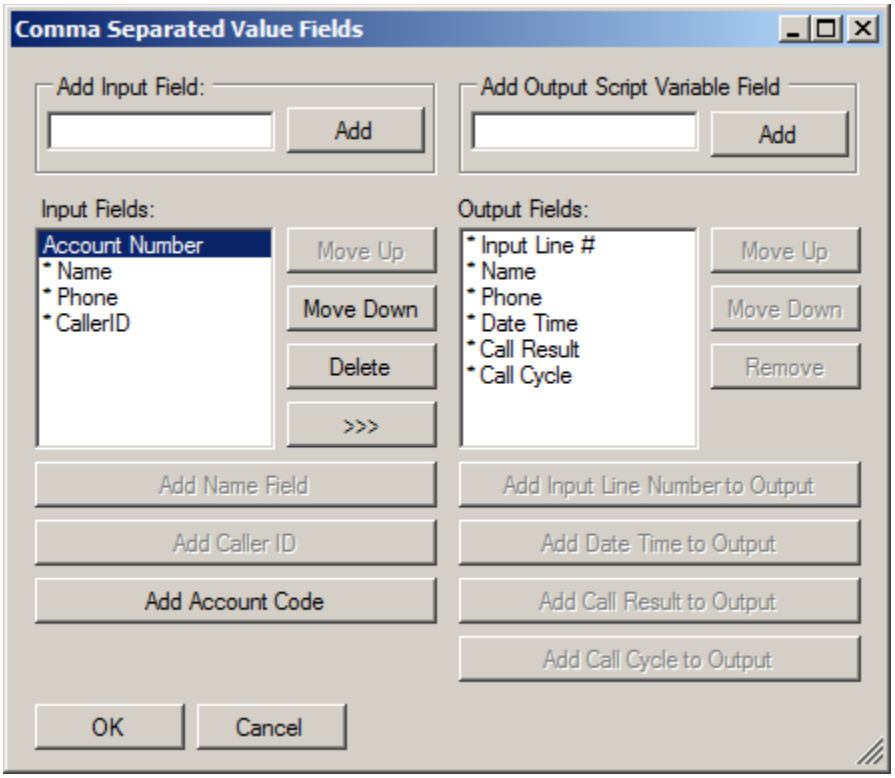
As an example of how to configure the settings for CSV fields to read from an Input CSV file with 4 fields in this order:

- **Account Number**
- **Name**
- **Phone Number**
- **Caller ID**

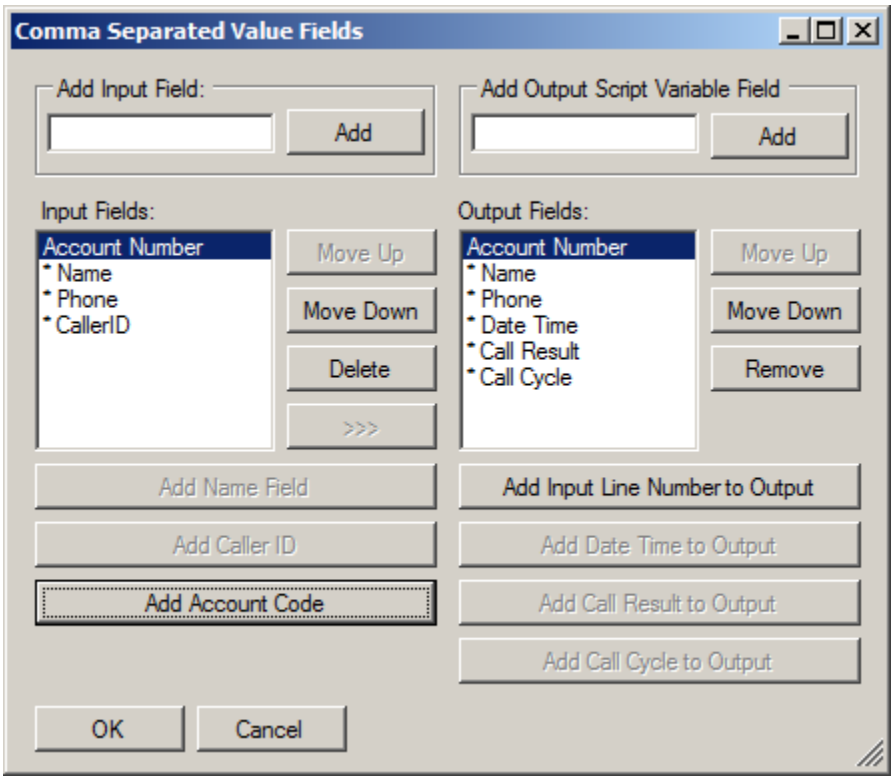
Based on this format, this is an example of the input file contents:

```
23, Maura Higgins, 8005551212, 8002341000
45, Grant Schenck, 2035551313, 8002341000
89, James Benton, 4085551414, 8002342000
```

You must add the system Caller ID field and the user **Account Number** field to **Input Fields** and then reorder the fields to match this list:



For example, assume you want to write the Account Number read from the input file to the output file rather than the input line number and you also want the Name and Phone number to be swapped. To do this, you must select **Input Line #** in the right-hand **Output Fields** list and click the **Remove** option. Then, with the **Account Number** selected in the **Input Fields**, click the **>>>** option to copy it to the **Output Fields**. Using the **Move Up** and **Move Down"**options next to the **Output Fields** to reorder the fields as needed. This shows the updated settings:



4.6 CREATING AND EDITING CSV INPUT FILE

The application reads an Input CSV file containing a number of commas separated data items or fields per line. By default, the application expects each line of the input file to contain two fields:

Field	Description
Name	Name of the party to be called.
Phone	Phone number to be called. It should not have any trunk access code as the application will automatically supply that from the OCIVR settings per OCIVR server instance. Note that you can use this application to call internal parties. As long as the number is less than 10 digits long then it is assumed to be an internal party and no trunk access code is pre-pended. **Note!** You can use this application to call internal parties. If the number is less than 10 digits long, then it is assumed to be an internal party and no trunk access code is pre-pended.

An example of a small input file is as follows:

```
Maura Higgins, 8005551212
Grant Schenck, 2035551313
James Benton, 4085551414
```

If a given line of the input file does not contain enough fields as configured in the CSV Fields dialog, then the entry is skipped, and a warning message will be written in the application log file.

Fields may contain the comma character if the field as a whole is enclosed in quotes. For example, if a name field contains last name, comma, first name then a valid input file might be:

"Higgins, Maura", 8005551212

"Schenck, Grant", 2035551313

"Benton, James", 4085551414

You can edit the CSV file using Microsoft Notepad, Microsoft Excel, or any text editor of your choice. You can create and modify the file outside of the OCIVR application and make sure to set the campaign's **Input CSV File** field to the appropriate file path or you can directly edit/create the CSV file using the **Edit** option next to the **Input CSV File** field of the **Campaign Configuration** dialog. If you use Excel or other editor to edit the file, ensure to save the file as a CSV text file. Excel defaults to saving files in native Excel format.

You can have any number of campaigns point to and use the same Input CSV file.

4.7 CREATING AND EDITING PROMPT FILES

Usually, at a minimum, the application plays one prompt file, the Message prompt. In addition, depending on configuration, several additional prompt files can also be played. These include the **Introduction**, **No Response**, and **Repeat** and **Transfer** prompts as well as script prompts. Of these prompts, three must instruct the called party which key to use to respond. Specifically:

- The **Introduction** prompt should ask the called party to press any key to accept the call.
- The **Repeat** prompt should tell the called party to press the one (**1**) key to repeat the message.
- The **Transfer** prompt should tell the called party to press the asterisk (*****) key to be transferred.

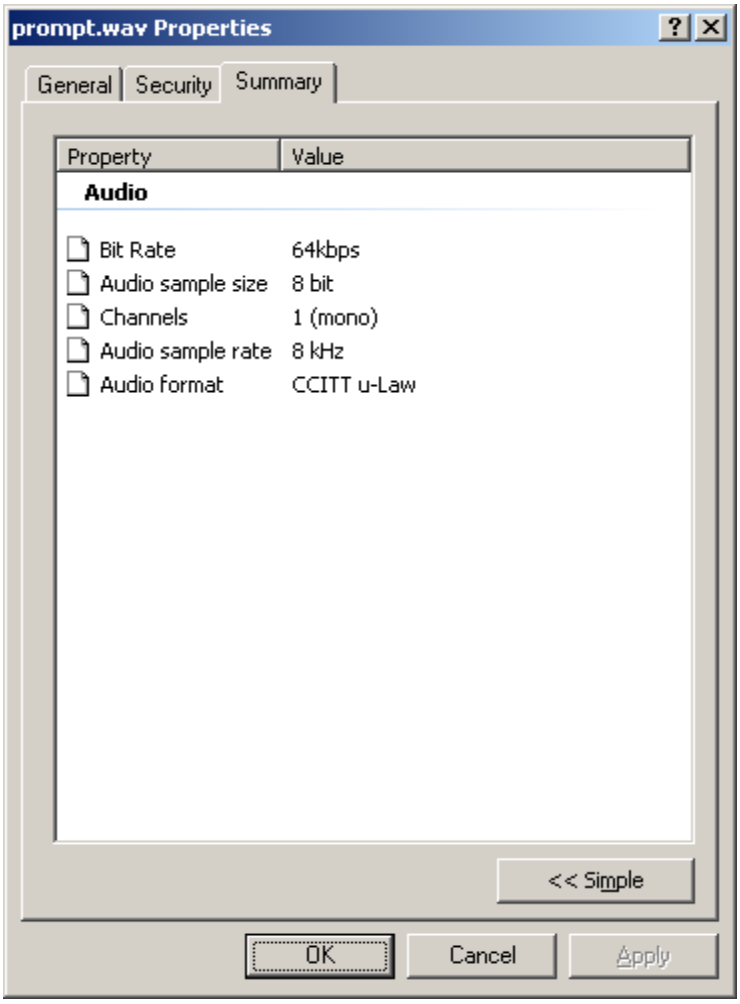
In addition to the campaign specific prompts, there are several pre-recorded system prompts supplied with the application to support playing data values in audio form. For example, there are prompts for spoken digits, dates, and times and others. Customers might want to record their own replacements for these prompts to provide fidelity between these system prompts and any campaign specific custom prompts they may record.

Prompt files must be recorded with a specific wave format. Specifically:

Field	Value
Bit Rate	64 Kbps
Sample Size	8 Bit
Sample Rate	8 Khz
Format	CCITT u_law

Note! It is critical that only this file format be used for prompt files. Attempting to use any other format is not supported and may require a server restart!

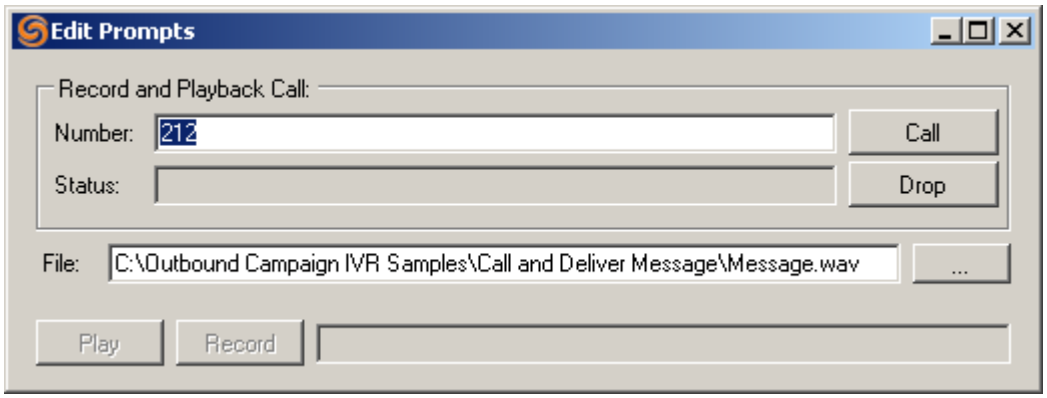
The screenshot below shows the file properties of an acceptable prompt file:



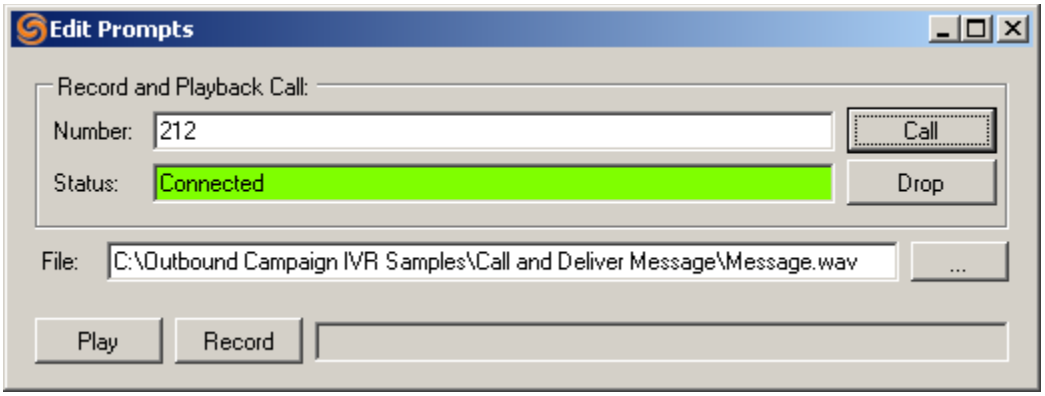
You can use a third-party application and a microphone and sound card to record prompts or you can record prompts directly from a phone using the application's build-in **Edit Prompts** dialog. This is accessed by clicking the **Edit** option next to any of the prompt fields in the **Campaign Configuration** dialog.

4.7.1 EDIT PROMPTS DIALOG

The **Edit Prompts** dialog can be opened by clicking at **Edit** option at **Campaign Configuration** dialog. The **Edit Prompts** dialog allows prompts to be played and recorded directly from the application. **Edit Prompts** dialog works on prompt files only, the function is not used to edit prompt text when prompts are generated using Text to Speech. For example, to record the **Message Prompt** click the **Edit** option next to **Prompt** field in Message area. The following dialog displays:



For the first time, you must enter the extension of a phone to which you have access. Alternatively, you can enter an external number (do not enter a trunk access code as this will be supplied automatically.) Then click the **Call** option and the phone should ring and if it is an internal call to a Mitel IP phone, it auto-connects. When the call connects the dialog should indicate that you are now connected:



You can drop the call by clicking the **Drop** option or simply hang up your phone.

You can now play or record the selected file (the path is shown in the File: field.) For example, if you click the **Play** option you should hear the prompt being played. You can view playback status and you can **Stop** or **Pause** the playback as needed.

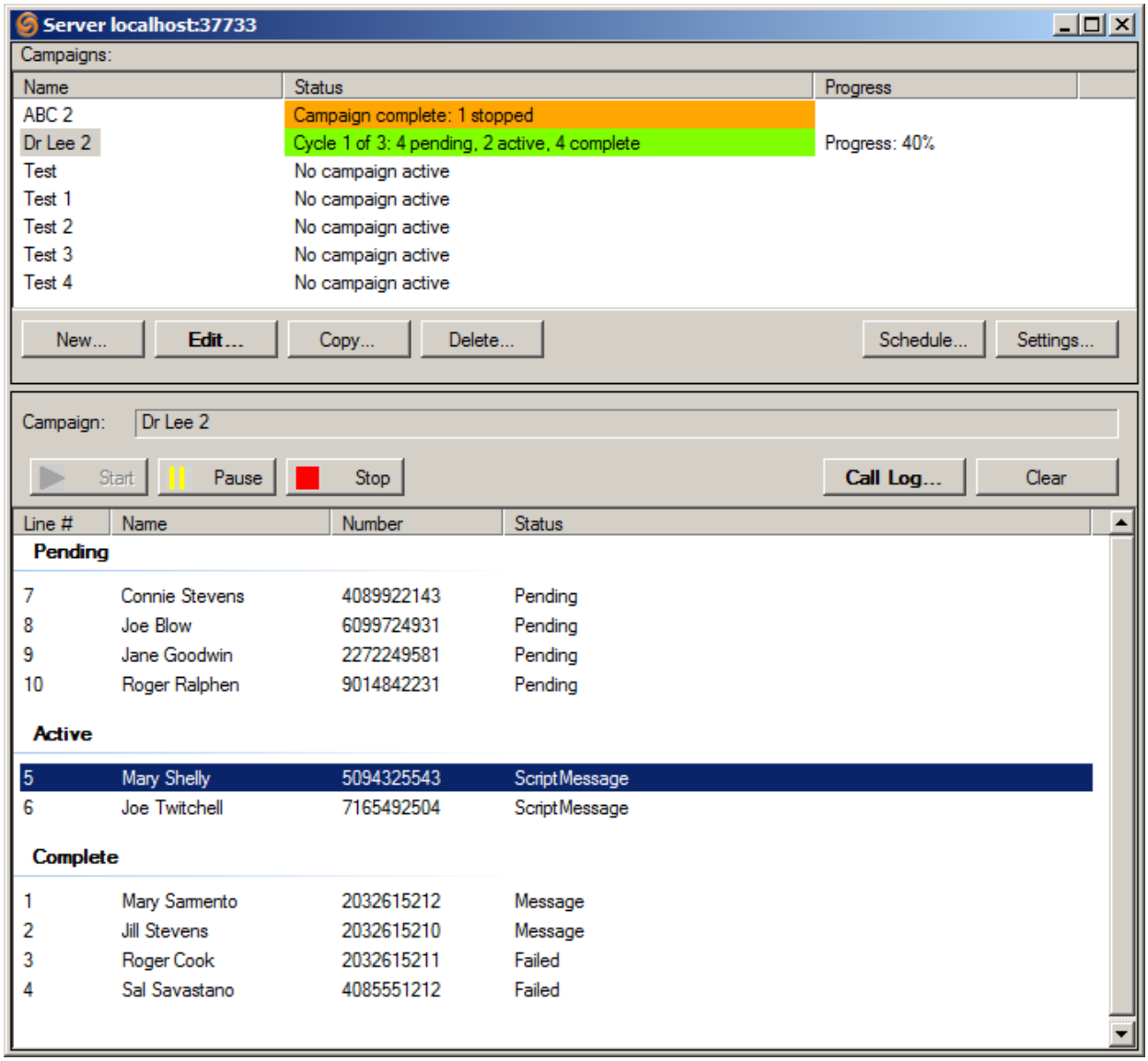
When you have recorded and reviewed a prompt to your satisfaction you can click the **Edit** option next to a prompt in the Campaign Configuration dialog to edit a different prompt. You can also use the ellipsis option (**...**) to directly select a different file. You can even close the dialog and open another campaign to work on its prompts without disconnecting the call.

4.8 RUNNING A CAMPAIGN

After you create a campaign and saved it, the campaign will appear in the Campaign Pane at the top of the **Server Configuration** dialog screen. To start the campaign, click **Start** in the Call Pane. Before the OCIVR places any calls, it first verifies that both the campaign specific Input CSV file and prompt files or prompt texts exist. If an Output CSV file is specified, then it also verifies that the file's directory exists. If there is an existing file with the same name as the Output CSV File is specified, the existing file is deleted. The OCIVR reads all the input from the input file, queues up all entries as pending calls to be made, and starts placing calls up the **Number of Simultaneous Calls** setting in the **Campaign Configuration** dialog of the campaign.

When a campaign is running, all the parties found in the Input CSV file will be shown in the **Call Pane**. They will be organized into three sections or groups. At the top will be the Pending calls. In the middle the **Active** calls and at the bottom the **Complete** calls. When a campaign completes, all the calls should be in the last group, the Complete group.

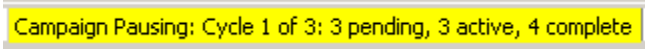
The following screenshot shows a campaign in process with 10 calls and the **Number of Simultaneous Calls** set to **3**:



4.8.1 PAUSING A CAMPAIGN

After a campaign is running, it can be paused. To pause the campaign, click the **Pause** option.

If a campaign is being paused, any active calls will be completed but no pending calls will be initiated. While the campaign is being paused the status will show that the campaign is pausing, and the number of active calls will get updated as they get completed:



After all active calls are completed, the campaign is paused, and the status will change to show that it is paused:

If a campaign is paused while the OCIVR is waiting to start the next cycle, then the next cycle won't start until the campaign is resumed.

Note! That the time to track the **Delay Between Cycles (minutes)** will still advance so that if for example the campaign is paused with 10 minutes to go before the next cycle and is then resumed 20 minutes later, it will immediately start initiating new calls.

4.8.2 STOPPING A CAMPAIGN

A running campaign can be stopped. To stop the campaign, press the **Stop** option. The **Stop** function, unlike **Pause**, causes active calls to be immediately disconnected regardless of their states. All calls that were pending or active are completed with a status of Stopped. If there are more cycles that could have run, they are skipped.

4.8.3 CALL STATUS

The **Status** column of the **Call Pane** shows details of the calls. Depending on the group the call is shown under, different values apply:

4.8.3.1.1 Pending

The only status for calls in the Pending group is Pending.

4.8.3.1.2 Active

Calls in the Active group are currently active. The status of active calls gives an indication of the positions of the calls within the OCIVR flow. Valid values include:

Value	Description
MakingCall	The OCIVR has started placing a call to the party and is waiting for the call request to be accepted by the system.
WaitingForConnect	The OCIVR has reached the far end trunk or destination and is now waiting for the trunk level connect or the called party to answer the call.
WaitingForFarEndAnswer	The OCIVR has connected to a trunk and is waiting for the far end to answer.
SilenceDetection	The OCIVR has connected to the far end and is now detecting silence on the line.
PromptIntroOne	The call has been connected and now the OCIVR is playing the Introduction Prompt for the first time.
PromptIntroTwo	The party did not respond to the first Introduction Prompt within the Menu Digit Timeout, The OCIVR is playing the Introduction Prompt for the second time.

PromptIntroThree	The party did not respond to the second Introduction Prompt within the Menu Digit Timeout, and the OCIVR is now playing the Introduction Prompt for the third and final time.
PromptNoResponse	After playing three Introduction Prompts, the called party still did not respond within Menu Digit Timeout period. As a result, and if the **Optional No Response Prompt/Text** is set, then the OCIVR is now playing the **Optional No Response** prompt.
PromptCampaignMessage	The called party pressed any digit in response to the Introduction Prompt, or, the called party pressed the digit '1' in response to the **Optional Repeat Prompt**, or, as the very first prompt of the campaign, the OCIVR is now playing the campaign-specific Message Prompt.
PromptRepeatTransferOne	Following the campaign-specific Message Prompt, if Transfer feature is included and the Optional Transfer Prompt/Text is set, the OCIVR is now playing the Optional Transfer Prompt for the first time instructing the party to hang up and/or press asterisk (**'*'**) digit to get transferred, for example, to talk with an agent.
PromptRepeatTransferTwo	The called party did not respond to the first **Optional Transfer** prompt within the Menu Digit Timeout and the OCIVR is now playing the prompt for the second time.
PromptRepeatTransferThree	Again, the party did not respond to the second **Optional Transfer** prompt within the Menu Digit Timeout and the OCIVR is now playing the prompt for the third and final time.
Transferring	The called party has pressed the asterisk ('*') digit in response to the **Optional Transfer** prompt to indicate that he or she did want to get transferred, for example, to speak with an agent and the OCIVR has submitted the transfer request to the system and is waiting for the transfer to occur.

4.8.3.1.3 Complete

The status shown for calls in the Complete group represents final call results for the Calling Cycle. The status will, for the most part for single Calling Cycle campaigns will be the same as the result written to the Optional Output CSV file with two important exceptions:

- If the Status is **Busy** or **No Answer** and there are additional cycles still to run as part of the current campaign, then this (interim) result is not written to the Output CSV file. Only once the Campaign completes or is stopped, any calls with a result status of Busy or No Answer will be written. Busy and No Answer are provisional results which are not saved until all call attempts (Calling Cycles) are complete or stopped.
- If the campaign is stopped, any **Active** and **Pending** calls will show as **Complete** with a status of **Stopped**. Calls with a status of **Stopped** will not be written to the **Optional Output CSV** file although calls which completed in a previous cycle with **Busy** or **No Answer** will be.

Valid call status values are shown below. The value shown in parentheses following the status string is the string written to the **Optional Output CSV** file:

Value	Description
Failed	This indicates a failure trying to make the call or a disconnection while prompting the user to respond with a digit press to hear the initial message. This may indicate an invalid number.

NoAnswer	The call failed to achieve a connected state within the campaign's No Answer Timeout. Note that the OCIVR setting Wait for Far End Connected on external calls for the OCIVR Server instance will affect when and if the call is considered connected.
Busy	The called party was busy.
NoResponse	The called party answered but did not respond. This is likely due to connecting to an answering machine. Note that the system will not attempt to recall parties of No Response in subsequent cycles, if any, to avoid the (undesirable) result of leaving the same message - repeatedly on the called party's answering machine.
Disconnected	The caller disconnected either due to a network issue or because the called party hung up during the Introduction Prompt or during the Message Prompt. If the called party disconnects any time before the end of the Message, the result will be Disconnected vs. Message.
Message	The called party answered and pressed a digit if/when asked if he or she wanted to hear the Message Prompt. The called party then listened to whole Message all the way to end and responded completely for any prompts that are part of the Message or scripts.
DoNotCall	The party's phone number matches an entry in the campaign's Do Not Call CSV file, so the party was not called.
Transfer	After hearing the Message or as part of a script, the called party was transferred to another number or, to a voice mail box.

4.9 CALL CAMPAIGN STATE PROGRAM

A given call progresses through a few states based primarily on the application and campaign settings. All calls start in the Pending state and progress from there:

Value	Description
All Calls	
Pending	Call is queued to be made.
MakingCall	Call is being made.
WaitingForConnect	The called party was busy.
Wait for Far End Connected on external calls application setting for the server instance is enabled	
WaitingForFarEndAnswer	Call was an external call and connected to a trunk but has not yet connected to the far end.
Silence Detection application setting for the server instance is enabled	
SilenceDetection	Call has connected to the far end and now we are waiting for silence on the line.
Introduction setting for the campaign is enabled	

PromptIntroOne	We are playing the Introduction Prompt for the first time.
PromptIntroTwo	We are playing the Introduction Prompt for the second time.
PromptIntroThree	We are playing the Introduction Prompt for the third time.
Introduction and Optional No Response Prompt settings for the campaign are enabled	
PromptNoResponse	We are playing the No Response prompt.
Message setting for the campaign is enabled with the Prompt/Text option	
PromptMessage	We are playing the Message prompt file or prompt text.
Message setting for the campaign is enabled with the Script option	
ScriptMessage	We are executing the script.
Message setting is enabled with the Optional Repeat Prompt" set and/or Transfer setting is enabled with the Optional Transfer Prompt set for the campaign	
PromptRepeatTransferOne	Playing the Optional Repeat and/or Optional Transfer Prompt for the first time.
PromptRepeatTransferTwo	Playing the Optional Repeat and/or Optional Transfer Prompt for the second time.
PromptRepeatTransferThree	Playing the Optional Repeat and/or Optional Transfer Prompt for the third time.
Transfer setting for the campaign is enabled	
Transferring	Call is being transferred
Transfer setting and When transferring calls, treat external calls as active until disconnected for the campaign are both enabled	
WaitingForDisconnect	Call has been transferred and we are waiting for the trunk end of the call to disconnect.
All Calls	
Complete	Call has completed

4.10 SCRIPTS

The core of most campaigns is the Message. Many campaigns will work fine using the built-in Introduction and Message prompts if you are playing a simple message such as notification about an early closing at a facility or a special offer for customers and so on. However, there are several examples where a simple prompt message does not suffice. These include:

- Playing prompts with unique prompt files per called party
- Playing multiple prompts strung together as a message
- Playing prompts interspersed with customer specific variable data read from the input file like dates, times, currency amounts and so on
- Playing prompts and capturing digit responses and saving to the output CSV file

- Capturing and manipulating data from the called party
- Branching to present different menus or options
- Creation, setting and testing of input, output, and script local variables

Scripts are XML formatted files with certain pre-defined support element types. A script file has this basic framework:

```
<?xml version="1.0" encoding="utf-8"?>
<script directory="C:\Outbound Campaign IVR Samples\Call and
Deliver Appointment Reminder Script">
  <play>
    <file>"Reminder.wav"</file>
    <datetime>Appointment</datetime>
    <file>"With.wav"</file>
    <file>Doctor</file>
   </play>
 </script>
```

The first line is the standard xml header:

```
<?xml version="1.0" encoding="utf-8"?>
```

This is followed by a single script element:

```
<script directory="C:\Outbound Campaign IVR Samples\Call and
Deliver Appointment Reminder Script">

...
</script>
```

The script element has a single optional attribute:

- Directory

Directory specifies the default location in which to look for prompt files referenced by the script. If directory is not set, then the campaign's default directory will be used. In addition, prompts referenced by the script can always be provided with rooted paths in which case the script's or campaign's directory is irrelevant.

The script element contains one or more sub elements. In the example above we have a single play element that again has several sub elements. These include play sub elements segments for playing files, dates, times, currency, and other types of data. The play element itself also supports several attributes. The play element, the attributes and the sub elements are detailed below along with the other script commands.

4.11 SCRIPTS VARIABLES

Variables are named storage locations which store strings of characters. Scripts can test and set variables. There are four types of variables:

Field	Description
CSV Input Fields	Each input field referenced in the campaign's CSV Field dialog can be used as a script variable. You can set the value of an input field which would override the value passed as part of the call's data to, for example, provide a default value if the call left this field empty.
CSV Output Fields	Each output field referenced in the campaign's CSV Field dialog can be used as a script variable. All output fields which are not also input fields are initialized to an empty string.
CSV Input/Output Fields	A CSV input field can also be an output field. In this case, the value read from the input file will be set for the output value unless changed by the script.
Script Variables	In addition to variables representing input and output CSV file fields, script writers can also create variables purely for use within a script using the XML define element. These could be used to hold temporary data, perhaps queried from the called party with a play command and then tested to set an internal state. For example, a script variable could be used to track a user wanting to open an urgent vs. routine trouble ticket.

4.12 SCRIPTS LABELS

Scripts can change their own execution path based on the data read from the Input CSV file and/or data entered by the called party in response to prompts. Any points in the script which you want to allow to be branched to must have an XML label element defined (see below). A label is given a unique ID and this ID value is referenced by other xml elements when branching.

4.13 SCRIPTS COMMANDS

Scripts consist of one or more elements collected under the script element. Some script commands such as set or switch execute quickly and others, such as play or transfer, take time to complete. All script commands are in lower case. The two main types of references made by the application are to variables and labels. The scripts we show use the convention of showing labels as all uppercase with optional underscores separating words, variables as mixed case without spaces between words and the XML is all lowercase.

comments (<! ----- >)

XML style comments can be inserted anywhere in the script.

4.13.1 EXAMPLE:

```
<!-- Ask the caller if they accept the changes -->
```

Value	Description
define	The define element is used to create a script variable for the script to use as needed. User input can be captured in a script variable using the play element and script variables can be compared using the switch element. The define element has several forms but at a minimum you must provide the variable attribute to define the name of the variable being created.

	Example: • `<define var="RepairQuality"/>` Creates a variable named RepairQuality and initializes it to an empty string. • `<define var="RepairQuality">"High"</define>` Creates a variable named RepairQuality and initializes it to the string "High". • `<define var="RepairQuality">PreviousRepairQuality</define>` Creates a variable named RepairQuality and initializes it to the contents of the variable (input, output or script) named **PreviousRepairQuality**.
exit	Normally a script completes when the last element in the script is executed. However, the exit element can be used to terminate the script from anywhere. **Example:** <exit/> Exits the script.
label	Defines a label which can be branched to by goto, play, switch, transfer and transfervm. The name of the label is set through the required id attribute. During execution, if a script attempts to branch to a label which does not exist then script will fail at run time. By convention labels are typically defined in upper case. **Example:** *<label id="SURVEY_RESULTS"/>* Creates a label named SURVEY_RESULTS which can be branched to by other elements.
Goto	Branches script execution to the specified label.  **Note!** The label can be a literal name in quotes or a variable name. In the latter case, the branch is made to the label matching the contents of the specified variable. If the variable is not found or the contents of the variable is not a known label, then the script will fail. **Example:** • *<goto>"SURVEY_RESULTS"</goto>* Branches to the label with an id of *SURVEY_RESULT*. • <goto>customerPriority<goto> Branches to the label whose id matches the contents of the variable named CustomerPriority at run time. If the CustomerPriority contains **HIGH**, then the script would branch to the label with the id of **HIGH**.
ttson and ttsoff	Elements ttson and ttsoff control whether the script will use prompt file concatenation to speak out numbers, dates, currency, etc. or if it will use the Windows TTS engine to generate a prompt from the text. Please note that when a script starts running, it

	will first default to the “Generate Prompts using Text to Speech” setting for the campaign. So, for example, if the campaign’s setting “Generate Prompts using Text to Speech” is checked then the script will start as if <ttson> has been executed.
	Example:
	`<ttson/>`
	`<play>`
	`<date>”12/19/2012”</date>`
	`</play>`
	`<ttsoff/>`
	`<play>`
	`<date>”12/19/2012”</date>`
	`</play>`
	Plays the date using Text to Speech and then plays the same date using concatenated pre-recorded prompt files.

ttsvoice	Element ttsvoice allows a specific TTS voice to be selected as the voice to be used. Normally if TTS is used, then the voice used is the one configured in the server’s settings. However, within a script the voice can be changed dynamically by using the ttsvoice element. The ttsvoice element expects either a string which should match the end of an installed voice’s name or a variable containing the end of a valid installed voice name. **Example:** • *<ttsvoice>”Microsoft Server Speech Text to Speech Voice (en-GB, Hazel)”</ttsvoice* *<play>* *<tts>”This is a test”</tts>* *</play>* Plays the “This is a test” using the English Great Britain voice “Hazel”. • *<define var="Zirapro">”(en-US, ZiraPro)”</define>* *<ttsvoice>Zirapro</ttsvoice>* *<play>* *<tts>”This is a test”</tts>* *</play>* Plays the “This is a test” using the US English voice “ZiraPro”. Note how we only have to provide the end of the voice name, i.e., (en-US, ZiraPro). Also note that in this case we’re passing in the voice as a variable.  **Note!** You must provide only the end of the voice name, that is en-US, or ZiraPro. Also, in this case in the voice as a variable.

play	The play element is the primary script element used to play prompts and other audio information to called parties and to capture digit press results. Through attributes, it allows automatic setting of variable contents and/or branching to labels based on customer entered digits as well as the direct capture of customer entered digits. There are several play attributes. Play attributes control how the play command works overall while the individual play sub-elements control the content of the audio played. By default, the play will use the **Menu Digit Timeout** and **Menu Inter Digit Timeout** specified in the application settings for the server instance. However, these values can be overridden through the play element's timeout and interdigittimeout attributes. Similarly, if a play is requesting input and the called party fails to respond or fails to respond adequately, then normally the play will repeat twice for a total of three times before treating the call as a no response call. However, the number of times the prompt automatically repeats can be changed with the repeat attribute. To directly capture the digits entered, you can specify the name of a variable to receive the results with the result attribute. Typically, you will be capturing either a single digit, in which case you will use the exitdigits attribute to set the digits that end the play and return or you would set the maxdigits attribute to set the number of digits that should be capture before the play command completes.  **Note!** When you use maxdigits, as soon as the caller enters the first digit, the playing stops, but the play command waits for the additional input before completing and returning to the script. If you want to set a variable with a result based on the entered digits or no response using the various on*set attributes, then set that variable name with the var attribute. You can use the various on* attributes to directly branch to a label based on an entered digit or no response.
timeout	This sets the timeout, in milliseconds, after the play finishes that the play either pauses or waits for input. If the timeout is not set, it defaults to the system **Menu Digit Timeout** setting for the server instance if the play is capturing input (exitdigits and/or maxdigits set). If the play does not wait for input, the timeout defaults to zero. Setting this attribute allows a different timeout value to be used.
interdigittimeout	If the play is capturing multiple digits of input by virtue of having the maxdigits attribute set, then this controls the maximum time allowed after the first digit that the play waits for each additional digit of input. If not set it defaults to the system Menu Inter Digit Timeout setting for the server instance.
exitdigits	If just any of the on* or on*set attributes are set or exitdigits is set, then the OCIVR script will try to capture a single digit. The result attribute can be set if you want to save the digit to a variable. If just exitdigit is set, then this is returned as the result. If both exitdigits and maxdigits are set, then the result will be either all the digits entered up to the maxdigits or the digits entered before the exit digit. When using exitdigits with maxdigits, the exit digits would typically be "#" and/or '*'.
maxdigits	The **maxdigits** attribute allows the capture of multiple digits. For example, to capture a 5-digit account number you would set the maxdigits to 5. As another example, if you want to capture an account number up to 5 digits optionally terminated with the pound ('#') sign you would set maxdigits to 5 and exitdigits to "#". timeout and interdigittimeout are also used in conjunction with maxdigits.
repeat	The repeat attribute changes the number of times the play repeats before the called party is unresponsive. The play defaults to two repeats (that is the prompt plays for a total of three times if the called party does not respond) but can be changed through this attribute. If the play is not capturing input (no exitdigits or maxdigits set) then the

	play defaults to zero repeat. If set to zero, then the prompt will play only once before completing.
result	The result attribute specifies the name of a variable where the digits entered by the called party will be placed. For example, if you had an output field variable called **AccountNumber,** you canthen prompt a called party to enter an account number and in the play command, in addition to specifying the maxdigits to 5, you would set the result to field named **AccountNumber**.
var	The var attribute specifies the name of a variable where the results of any on*set attributes will be placed and must be set if any are used. For example, if they were setting variable called QualityResult based on a user entering **1** for High, **2** for Medium, and **3** for Low then they could use the **on1set, on2set,** and **on3set** attributes combined with the **var="QualityResult**".
on1set, on2set..., on0set, onpoundset, onasteriskset and onnoresponseset	There are 13 versions of this attributes and any number may be used in a single play command depending on the requirements. They work in conjunction with the var attribute to allow setting a variable to a specific string based on the user entering a specific digit, pound, asterisk or not responding.
on1, on2..., on0, onpound, onasterisk, and onnoresponse	There are 13 versions of this attributes and any number may be used in a single play command depending on the requirements. These attributes allow the direct branching to the specified label based on the user entering a specific digit, pound, asterisk or not responding.

.

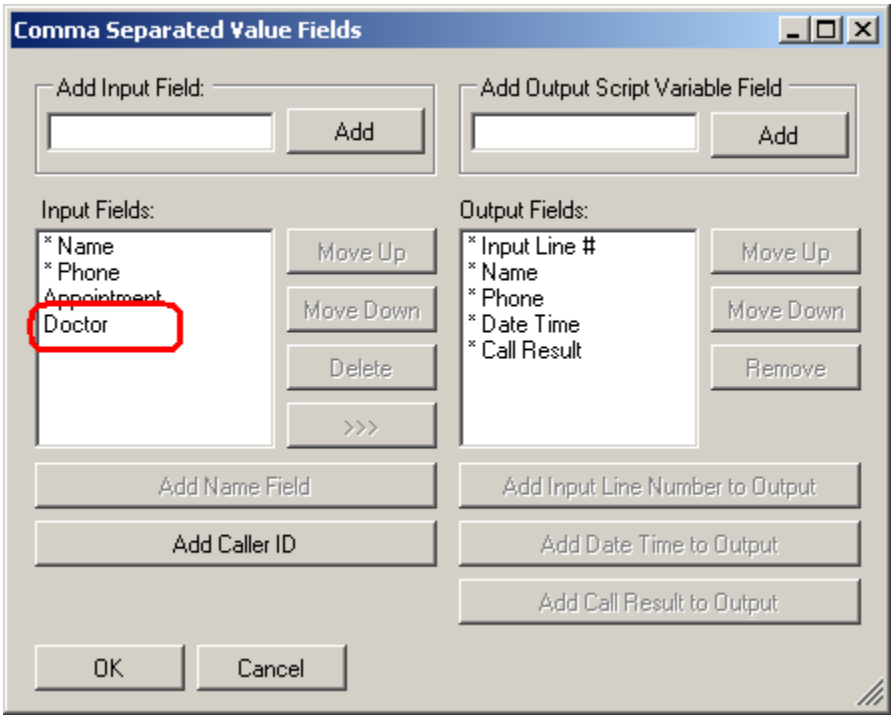
There are various element types that can be part of the play element. All the element types except those related to modifying how TTS audio is generated accept either a literal or a variable. For example, a play file element could look like this:

```
<file>"With.wav"</file>
```

or like this:

```
<file>Doctor</file>
```

The first is an example of a literal value. That is, this element would result in the script playing the contents of a wave file named **With.wav**. The second example references a CSV input field named Doctor (or a script variable or an output field) and uses the customer-specific value as the name of a wave file to play. As a reminder, wave files are in the script-specific prompt directory. For this second case to work, the campaign would have to have either an input or output CSV field variable or script variable with the specified name. For example, it might be an input CSV field as shown below:



The specific play sub-element types are:

Value	Description
file	Plays audio using the contents of a single prompt file. **Example:** • *<file>"Greeting"</file>* Plays the contents of the audio file named **Greeting.wav** located in either the scripts directory or the System Prompt Directory. • *<file>Greeting</file>* Treat the contents of the variable named "Greeting" as the name of an audio file and play it.
tts	Plays audio by converting the text to speech. **Example:** • *<tts>"This is ShoreTel Calling"</tts>* Plays "This is Mitel Calling" by converting the contents of the string to an audio prompt using Text to Speech (TTS). • *<tts>Greeting</tts>* Treat the contents of the variable named "Greeting" as a text string and play using TTS.
ttsstartparagraph/ttsendparagraph	Brackets several plays that represent a paragraph to guide the TTS engine. Usually not needed but can be used if more control is desired.
ttsstartsentance/ttsendsentance	Brackets several plays that represent a sentence to guide the TTS engine. Usually not needed but can be used if more control is desired.
ttsstartstyle/ttsendstyle	Brackets several plays that use TTS to modify how the audio is generated with regards to emphasis and/or rate (speed) and/or volume. Styles can be nested so a second style can be started to apply to a portion of text and when ended, the previous style will apply. Three attributes are supported. At least one must be used if needed, all three can be used in the same ttsstartstyle element. Each attribute has several valid values. They are: • emphasis= "none" | "reduced" | "moderate" | "strong" • rate="extraslow" | "slow" | "medium" | "fast" | "extrafast" • volume= "default" | "silent" | "extrasoft" | "soft" | "medium" | "loud" | "extraloud" **Examples:** • *<play>* *<tts>"Are you"</tts>* *<ttsstartstyle emphasis="strong" />* *<tts>"sure?"</tts>* *<ttsendstyle/>*

	</play> Plays **Are you sure**? where the word “sure” is spoken with a strong emphasis. • *<ttson/>* *<play>* *<tts>”Your balance is”</tts>* *<ttsstartstyle rate=”slow” volume=”loud” />* *<currency>Balance</currency>* *<ttsendstyle/>* *</play>* Plays **Your balance is** normally followed by the balance (presumably one of the CSV input fields) slowly and loud.
ttsbreak	Using the break attribute allows a pause to be inserted into the audio stream. Valid values for the attribute are: *break=“none”* | *“extrasmall”* | *“small”* | *“medium”* | *“large”* | *“extralarge”* **Example:** • *<play>* *<tts>”Hello, this is ShoreTel calling.”</tts>* *<ttsbreak break=”small” />* *<tts>”Please press 1 to speak with someone about your account.”</tts>* *</play>* Plays “**Hello, this is Mitel calling**” followed by a short pause then “**Please press 1 to speak with someone about your account**”.
systemfiles	Plays one or more system prompts. System prompts are initially located in the Prompts subdirectory of the application's installation directory, by default, at *C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR Service* on the hosting server. But it can be configured to a different directory by updating the application's **System Prompts Directory** setting at the Application Settings dialog of the server instance. Example: • *<play>* *<systemfiles>”YOUENTERED”</systemfiles>* *<tts>Amount</tts>* *</play>* Plays “**You entered**” followed by the contents of the Amount variable rendered using TTS.

currency	Plays the literal text or variable contents as a currency amount. For example, with an input of "123.45" the called party would hear "One Hundred Twenty-Three Dollars and Forty-Five cents."
decimal	Plays the literal text or variable contents as a decimal number. For example, with an input of "123.45" the called party would hear "One Hundred Twenty-Three Point four five."
int	Plays the literal text or variable contents as an integer number. For example, with an input of "123" the called party would hear "One Hundred Twenty-Three".
datemonthyear	Plays the literal text or variable contents as a month and year. For example, given an input of "12/19/57" the called party would hear "December, Nineteen Fifty-Seven."
date	Plays the literal text or variable contents as day, month, and year. For example, given an input of "12/19/1957" the called party would hear "December Nineteenth, Nineteen Fifty-Seven."
dateandday	Plays the literal text or variable contents as a day of the week, day, month and year. For example, given an input of "10/2/2009" the called party would hear "Friday, October second, Two Thousand and Nine."
time	Plays the literal text or variable contents as a time. For example, given an input of "12:34 PM" the called party would hear "Twelve Thirty-Four PM."
datetime	Plays the literal text or variable contents as a day, month, year and time. For example, given an input of "12/19/1957 12:34 PM" the called party would hear "December, Nineteenth, Nineteen Fifty-Seven at Twelve Thirty-Four PM."
Timedate	Plays the literal text or variable contents as a time, day, month, and year. For example, given an input of "12/19/1957 12:34 PM" the called party would hear "Twelve Thirty-Four PM on December, Nineteenth, Nineteen Fifty-Seven."
year	Plays the year part of the literal text or variable contents interpreted as a date. For example, given an input of "12/19/1957 12:34: PM" the called party would hear "Nineteen Fifty-seven."
month	Plays the month part of the literal text or variable contents interpreted as a date. For example, given an input of "12/19/1957 12:34 PM" the called party would hear "December."
day	Plays the day part of the literal text or variable contents interpreted as a date. For example, given an input of "12/19/1957 12:34 PM" the called party would hear "Nineteenth."
dayoftheweek	Plays the day part of the literal text or variable contents interpreted as a date. For example, given an input of "10/2/2009" the called party would hear "Friday."
timespan	Plays the literal text or variable contents as a time span. For example, given an input of "12:34:56" the called party would hear "Twelve Hours, Thirty-Four Minutes and Fifty-Six Seconds".

characters	Plays the literal text or variable contents as individual characters. For example, given an input of "www.shoretel.com" the called party would hear "W-W-W dot M-I-T- E-L dot C-O-M."
telephone	Only for North American (10 digit) phone numbers. Plays “Area code” + the number’s digits.

Except for files and TTS related play element types, all other play elements use either a set of prerecorded prompt files stored in the System Prompts Directory (see the Application Settings section) or generate the prompts from prompt text using TTS. When the application is first installed, the various system prompts are placed in the "Prompts" folder of the application's installation directory. The customer may wish to record replacements for these “factory” prompts so that they blend seamlessly with any campaign specific prompts that they record themselves. In this case, the recommended procedure is to make a copy of the system prompts and place them in a new location, then update the application settings to point to this new location and finally, on an as needed basis, record replacements for each of the prompts.

Value	Description
setproperty	Allows setting a Mitel call property to a specific value or the contents of a variable. Mitel call properties are short named pieces of data which can be attached to a call. Then, if the call is transferred to another extension the call properties travel with the call and are available to other applications. A typical use of Mitel call properties is to facilitate integration with some kind of agent desktop application. For example, a calling campaign can mark calls transferred to a Mitel Workgroup with the account number of the customer. This way when a Workgroup agent gets the call, a desktop client application can access the call, extract the account number property from the call and respond by opening the appropriate customer page for display to the agent to provide context for the call. The *setproperty* element has a required attribute, name, which identifies the name of the property being set. **Examples:** • *<setproperty name="_ST_ACCOUNT_NUMBER"/>* This clears the Mitel call property named ***_ST_ACCOUNT_NUMBER*** by setting it to an empty string. • *<setproperty name="_ST_ACCOUNT_NUMBER">"12345"</setproperty>* This sets the Mitel call property named **_ST_ACCOUNT_NUMBER** to the string "12345". • <setproperty name="_ST_ACCOUNT_NUMBER">AccountNumber</setproperty> This sets the Mitel call property named **_ST_ACCOUNT_NUMBER** to the contents of the variable AccountNumber.
switch	The switch element allows a variable to be compared to one or more values for the script to branch to the label associated with the matching value. The variable to compare is set by the required var attribute. The comparison case(s) are set via the case element. In addition, there is also an optional

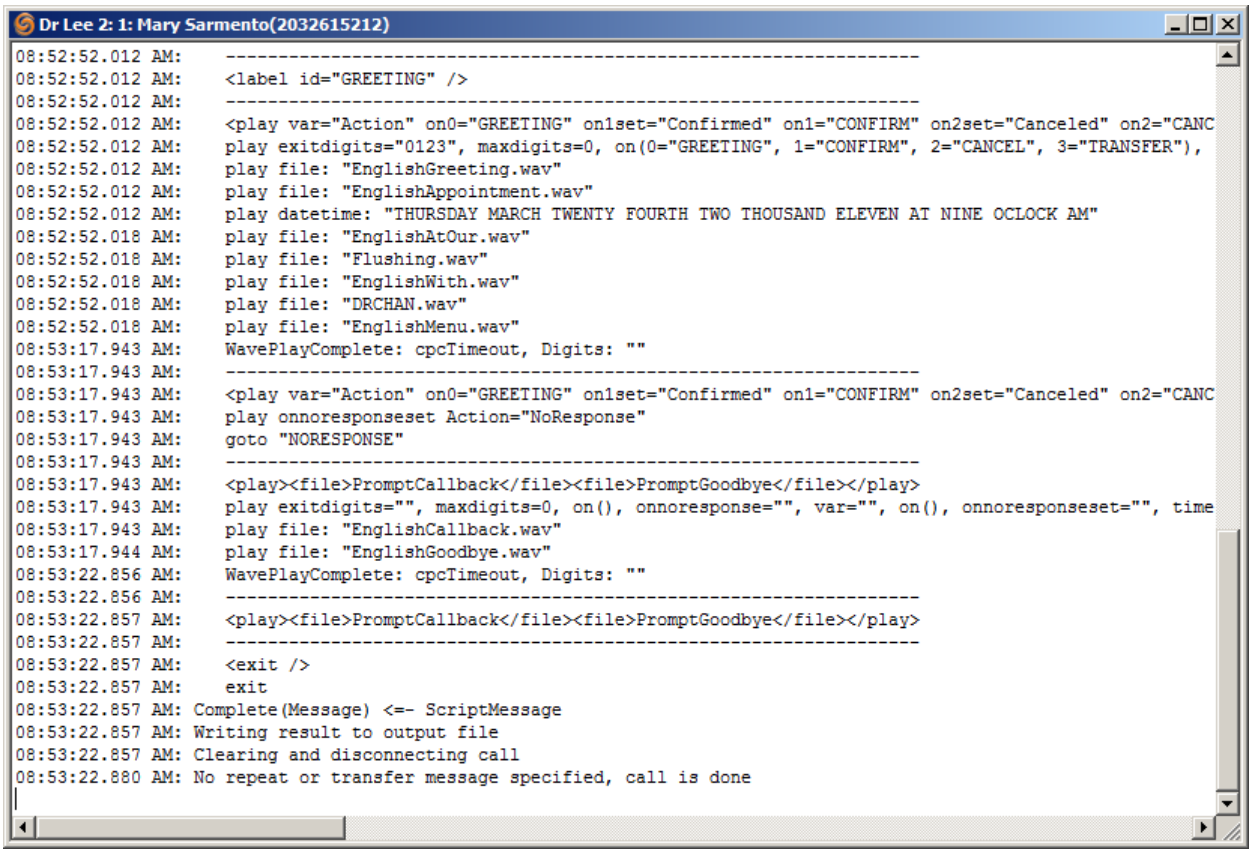
	default element to handle the scenario where none of the cases match the variable contents. **Examples:** • *<switch var="InsuranceRisk">* *<case value="High">HIGH_RISK_CUSTOMERS</case>* *<case value="Medium">MEDIUM_RISK_CUSTOMERS</case>* *<case value="Low">LOW_RISK_CUSTOMERS</case>* *</switch>* This compares the contents of the variable named InsuranceRisk to three different values, "**High**", "**Medium**" and "**Low**" and branches to either, the HIGH_RISK_CUSTOMERS, MEDIUM_RISK_CUSTOMERS, or LOW_RISK_CUSTOMERS label based on the value. If none of the values match then the script continues at the next statement. • *<switch var="RiskOk">* *<case value="True">RISK_OK</case>* *</switch>* This compares the contents of the variable named RiskOk to the value, "True" and if it matches then it branches to the label RISK_OK; if it does not match, it branches to the label RISK_NOT_OK.
Transfer	This transfers the called party to the number passed as either a string the contents of a variable. If the transfer is successful, the call result is set to "Transfer". If the request fails then the call is disconnected unless the optional attribute, "onfailure" is set. If it is set, then the script will branch to the specified onfailure label.  **Note!** That the transfer can succeed even if the destination is not the intended destination. **Examples:** • *<transfer>"212"</transfer>* This transfers the call to extension 212. • *<transfer>HandlingAgent</transfer>* This transfers the call to the contents of the variable named HandlingAgent. For example, if HandlingAgent contained "212" then this statement would be equivalent to the previous example.
transfervm	This transfers the caller to the Mitel voice mail box passed as either a string literal or the contents of a variable. If the transfer is successful, the call result is set to "**Transfer**". If the request fails then the call is disconnected unless the optional attribute, "**onfailure**" is set. If it is set, then the script will branch to the specified onfailure label.

	Note! The transfer will succeed even if the mailbox is not a valid mailbox. **Examples:** • *<transfer>"212"</transfer>* This transfers the call to mailbox 212. • *<transfer>MessageBox</transfer>* This transfers the call to the contents of the variable named MessageBox. For example, if MessageBox contained "212" then this statement would be equivalent to the previous example.

4.14 CALL LOG WINDOW

In addition to the status information shown in the **Call Pane**, additional details about the calls can be seen by opening the Call Log. The Call Log can be opened at any time and left open. It will display the details of any call selected in the Call Pane including pending, active, and completed calls. If the Call Pane is not already open, you can select a call and open the **Call Pane** by double-clicking a **call**.

The figure below shows the Call Log with details of call:



The level of detail may be a bit overwhelming, but it can be helpful in the case of failed calls, for example, to see why the system considered the call to have failed.

4.15 OPTIONAL OUTPUT CSV FILE

An important feature of the application is the output file that it can optionally generate. In the Campaign Configuration dialog settings, the Optional Output CSV File field can be left blank in which case no Output CSV file is written. Assuming it is set to a valid file name, the Output CSV file will contain details of the calls as defined by the CSV Fields settings. When initially installed, the default CSV Fields will cause five fields to be written to the Output CSV file:

- **Input Line Number**
- **Name**
- **Number**
- **Date and Time**
- **Call Result**
- **Call Cycle**

The possible Call Result for OCIVR campaign calls were defined as part of Call Status in the Running a Campaign section above.

The following shows a few sample entries:

1, Sal Savatano, 211, 3/6/2009 2:45:19 PM, Transfer

2, Joe Swin, 212, 3/6/2009 2:45:19 PM, Transfer

3, Joe Herve, 213, 3/6/2009 2:45:21 PM, NoAnswer

5. USING THE COMMAND LINE UTILITY (OCIVR COMMAND EXE)

Installed as part of the admin setup is a command line utility which can be run on demand or through a script to automate starting, stopping, pausing, or resuming a campaign. Like the Admin utility, the command line utility can be run from any Windows PC which can connect to the OCIVR Server(s).

Assuming the admin was installed to the default location on the user's PC, the utility would be located at this location on a 64-bit PC:

C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR Admin\OCIVR_Command.exe

The OCIVR_Command.exe requires 4 parameters be entered on the command line. They are:

Field	Description
Server	The server can be a machine name or IP address. If running the *OCIVR_Command.exe* on the same PC where the server is installed, then it can be set to **"localhost**".
Port	The port is the port configured for the server to expect the admin to connect on. The default value is **37733** unless changed.
Campaign Name	The name of the campaign to control. The campaign should have been previously created and configured using the Admin program. If the campaign name contains any spaces, then the **Campaign Name** must be enclosed in quotes.
Action	The action being requested. There are four actions supported; **Start**, **Stop**, **Pause**, and **Resume**.

The *OCIVR_Command.exe* will return a result which can be checked, for example, from a script that is executing the command. If the command is accepted, then the result will be zero. Is there is a missing parameter or any other error then the returned result will be -1.

In all cases, the command will also display either the parameters and an error message or will display a message indicating that the action has been accepted.

Note! That the request is handled immediately and while a zero return indicates the request was accepted, it does not indicate that, for example, the campaign has completed. To check the status the admin can be run, and the campaign inspected.

5.1 OCIVR COMMAND EXE EXAMPLES

This shows some examples of using the command line utility:

- Displaying the parameters:

```
C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR
Admin>ocivr_command OCIVR_Command: [server] [port] [campaign] [action]
```

```
where port is usually 37733 and action can be start, stop, pause or
resume

OCIVR_Command: For example:

OCIVR_Command: localhost 37733 "Appointment Reminder" start
```

- Starting a campaign:

```
C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR
Admin>ocivr_command localhost 37733 "Appointment Reminder" start

OCIVR_Command: Using settings Server: localhost, Port: 37733,
Campaign: Appointment Reminder, Action: start
```

- Stopping a campaign:

```
C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR
Admin>ocivr_command localhost 37733 "Appointment Reminder" stop

OCIVR_Command: Using settings Server: localhost, Port: 37733,
Campaign: Appointment Reminder, Action: stop
```

- Error example, stopping a campaign which isn't running:

```
C:\Program Files (x86)\Mitel\Mitel Outbound Campaign IVR
Admin>ocivr_command localhost 37733 "Appointment Reminder" stop

OCIVR_Command: Using settings Server: localhost, Port: 37733,
Campaign: Appointment Reminder, Action: stop

Campaign Appointment Reminder isn't runn
```

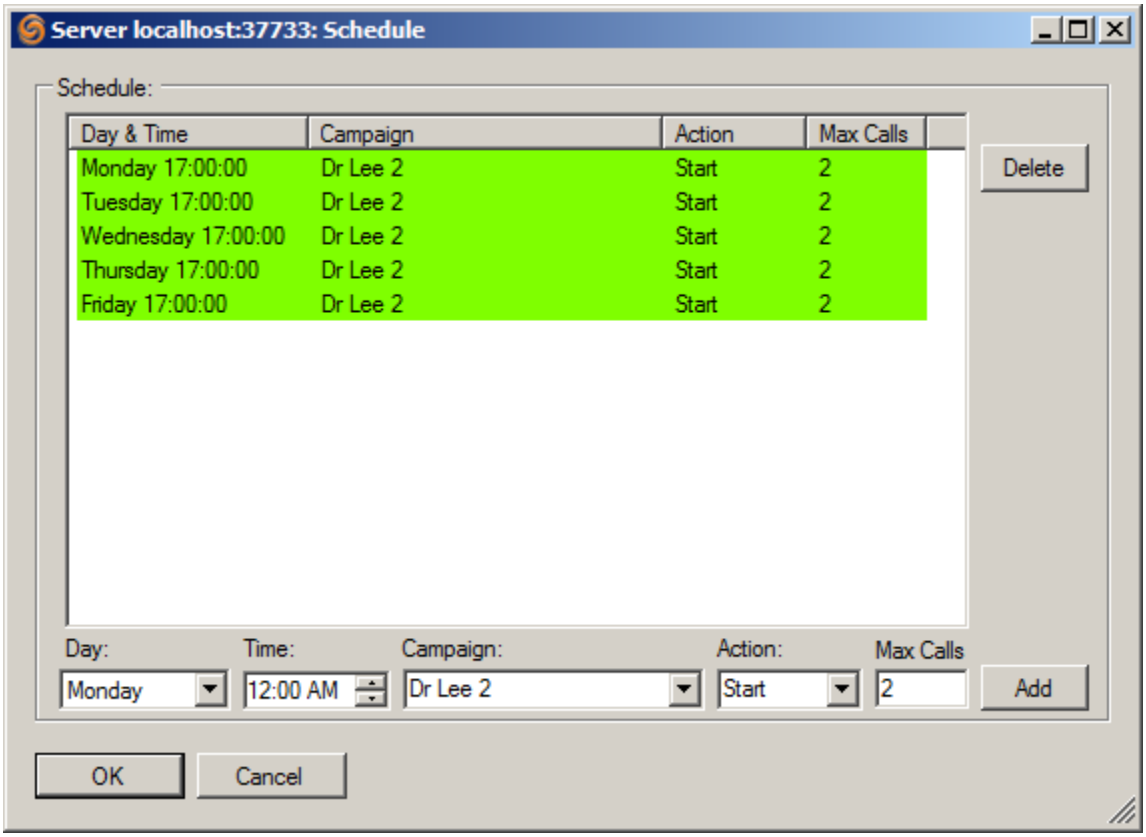
6. SAMPLE IVR FLOW

The user experience when being called by the OCIVR in support of a campaign constructed with the Campaign Configuration dialog by using Prompt or Text feature is as follows:

- The called party's phone rings. The call must be answered by the called party or device within the No Answer Timeout for the campaign otherwise the result is No Answer.
- After the call connects, if the **Introduction** option is selected, then the called party or device that answers hears the campaign's Introduction Prompt. This prompt will request that the called party or device press any key to accept the call, for example: **If you would like to hear an important message from Mitel Professional Services, please press any key now**.
    - If the called party fails to press a key, the prompt repeats two more times. If the called party still fails to respond after 3 tries
    - If the **Optional No Response Prompt** is selected, then the optional No Response prompt, defined for the campaign is played. For example, **If you want more information on this call please contact Mitel Professional Services at 408-331-3331**. After the No Response Prompt completes the call is disconnected. If the **Optional No Response Prompt** is not set, the call is disconnected.
    - If the **Play message even if no response to introduction prompts** is selected, then the Message Prompt is played. As normally a person would respond by pressing a digit on their phone to accept the call or simply disconnect by now, playing Message Prompt at this stage would presumably have the result of playing the Message on the called party's answering machine.
- If the **Introduction** option is not selected or if the called party presses a key in response to the Introduction Prompt, then the called party or device hears the campaign Message Prompt or script upon answering the call
- Following the campaign Message Prompt or script the call is either disconnected OR, if the **Optional Repeat Prompt** and/or "**Transfer**" is selected, the called party will hear the Optional Repeat or Transfer Prompt or both and be given the option to enter an appropriate key ('1' to repeat the Message Prompt and '*' to transfer.)
- If the called party fails to respond, the messages are repeated twice and if the called party still fails to respond, the call is disconnected.

6.1 SCHEDULE

In addition to manually running a campaign by **Starting**, **Pausing**, and **Resuming** the campaign at the **Server Configuration** dialog, you can also configure a weekly schedule to automatically run a campaign. To access the server's schedule, click the **Schedule** option at the **Server Configuration** dialog on the server instance:

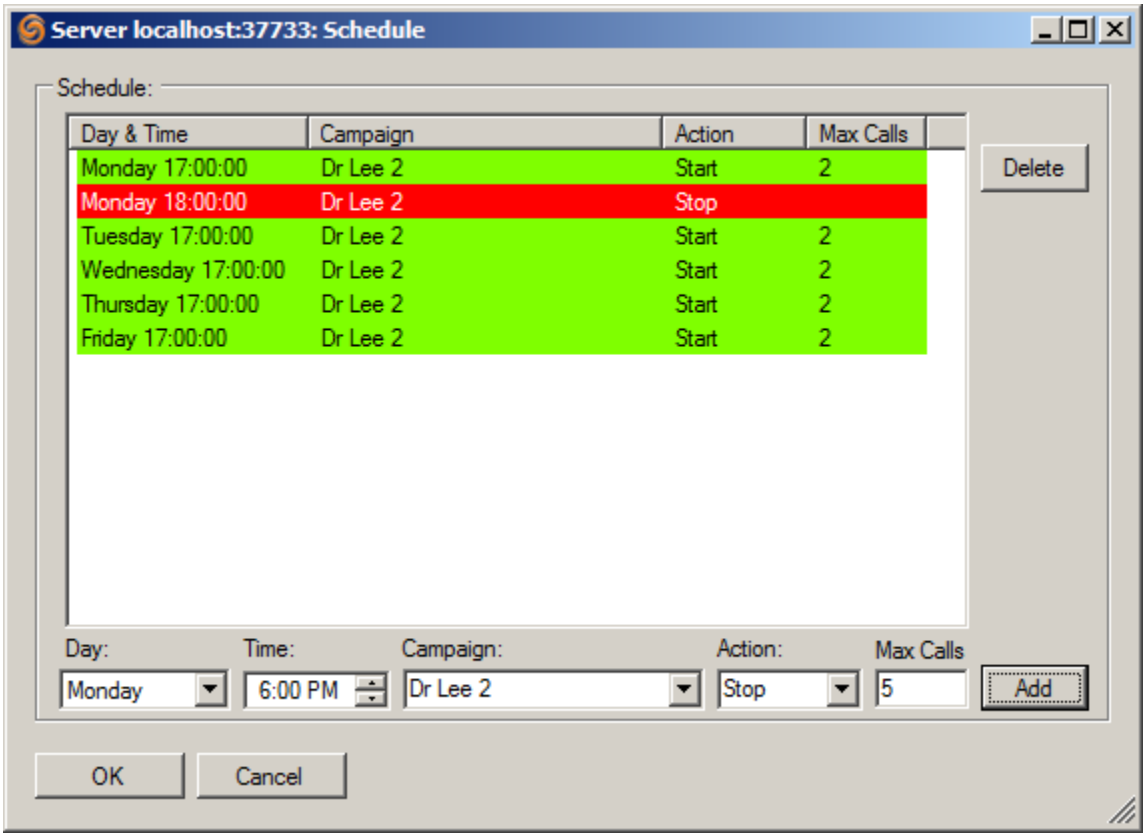


The current server's schedule will be displayed. Using this dialog, you can add automatic actions for campaigns including **Start**, **Stop**, **Pause,** and **Resume**. The above schedule shows the server instance configured to automatically run the campaign named "Dr Lee 2" every weekday at 5 PM. With no Stop action, this campaign will run until complete.

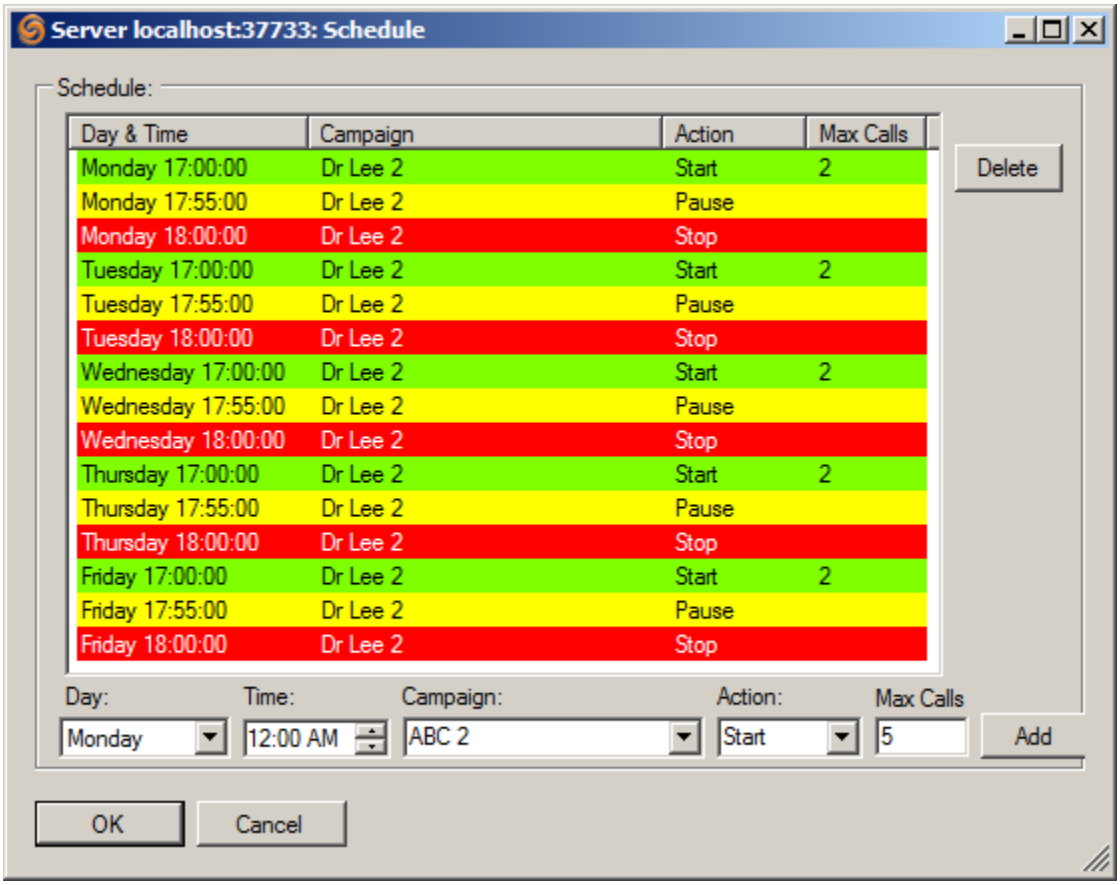
To add a schedule item, enter the schedule details and click **Add**. The details include the day of the week, the local time to the server instance (make sure to correctly select AM or PM), the name of the campaign that the schedule item applies to, the action (**Start, Stop**, **Pause**, and **Resume**) and the Max Calls. Max Calls is only relevant for the **Start** and **Resume** actions. This allows you to configure a campaign to use a different number of calls then the default Number of Simultaneous Calls at the Campaign Configuration dialog used when the campaign is manually started or resumed.

For example, to add a schedule item to automatically stop the "Dr Lee 2" campaign at 6 PM on Monday you would enter these details:

And then click the **Add** option resulting in this updated list of schedule items:

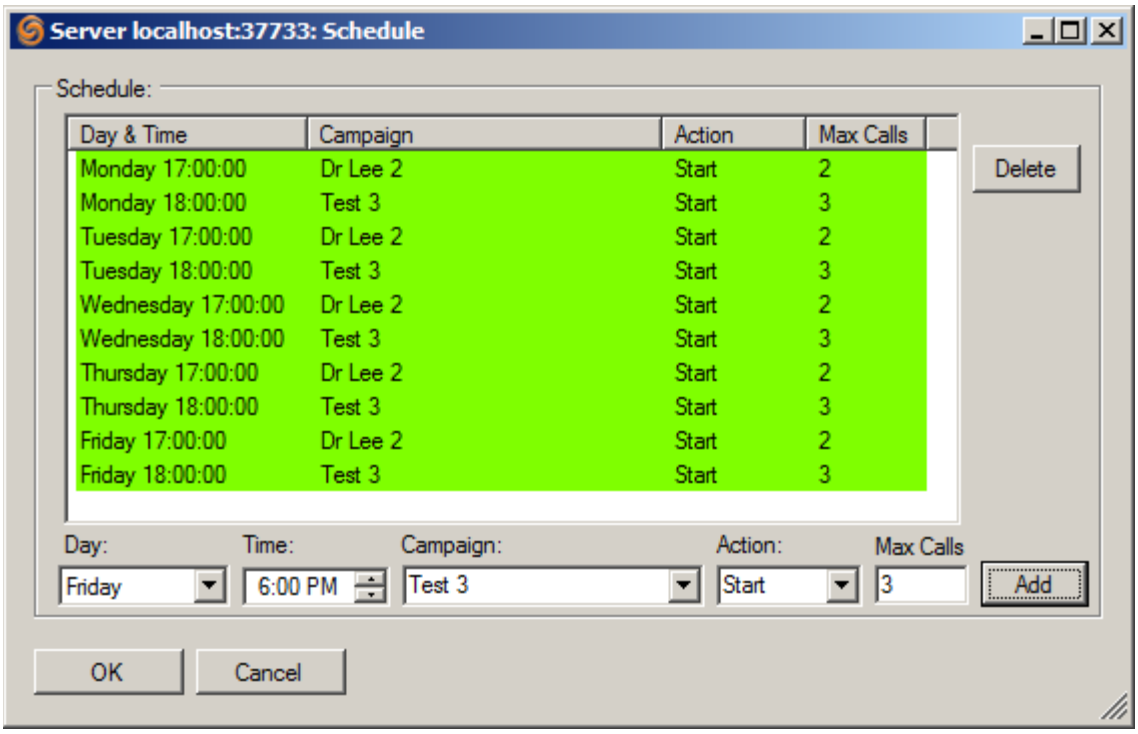


The figure below shows a more complex schedule:



In this case, the schedule is set to run the “Dr Lee 2” campaign starting at 5 PM on weekdays but then, at 5:55 PM, to automatically pause the campaign if it is still running. Then, at 6 PM it is configured to automatically stop the campaign. Pausing 5 minutes before the stop time allows active calls to finish normally and not be disconnected while the customer is still interacting with the campaign.

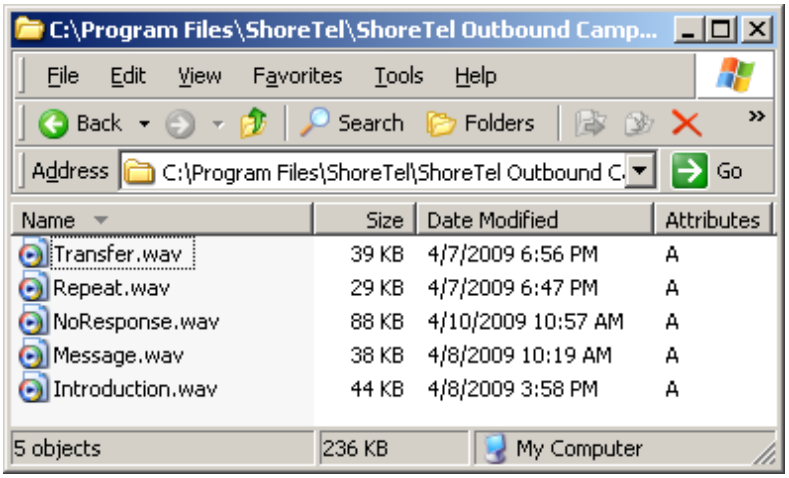
The figure below shows a simple scenario where two different campaigns are scheduled to be run:



Now the application is scheduled to run the “Dr Lee 2” campaign every weekday starting at 5 PM and the “Test 3” campaign every weekday starting at 6 PM.

6.2 SAMPLE PROMPTS

As examples of the recommended contents of the five prompt files, pre-recorded samples of the five prompts are installed in the Prompts subdirectory of the application's installation directory:



The audio format of these prompts is the same as that described above in the Creating and Editing the Prompt Files section.

You should not modify prompts while a campaign is running. Though it is fine to have the application loaded, there is a danger of having a call result in a failure if prompts are modified or removed while a campaign is running.

Following is the content for each of the five example prompts:

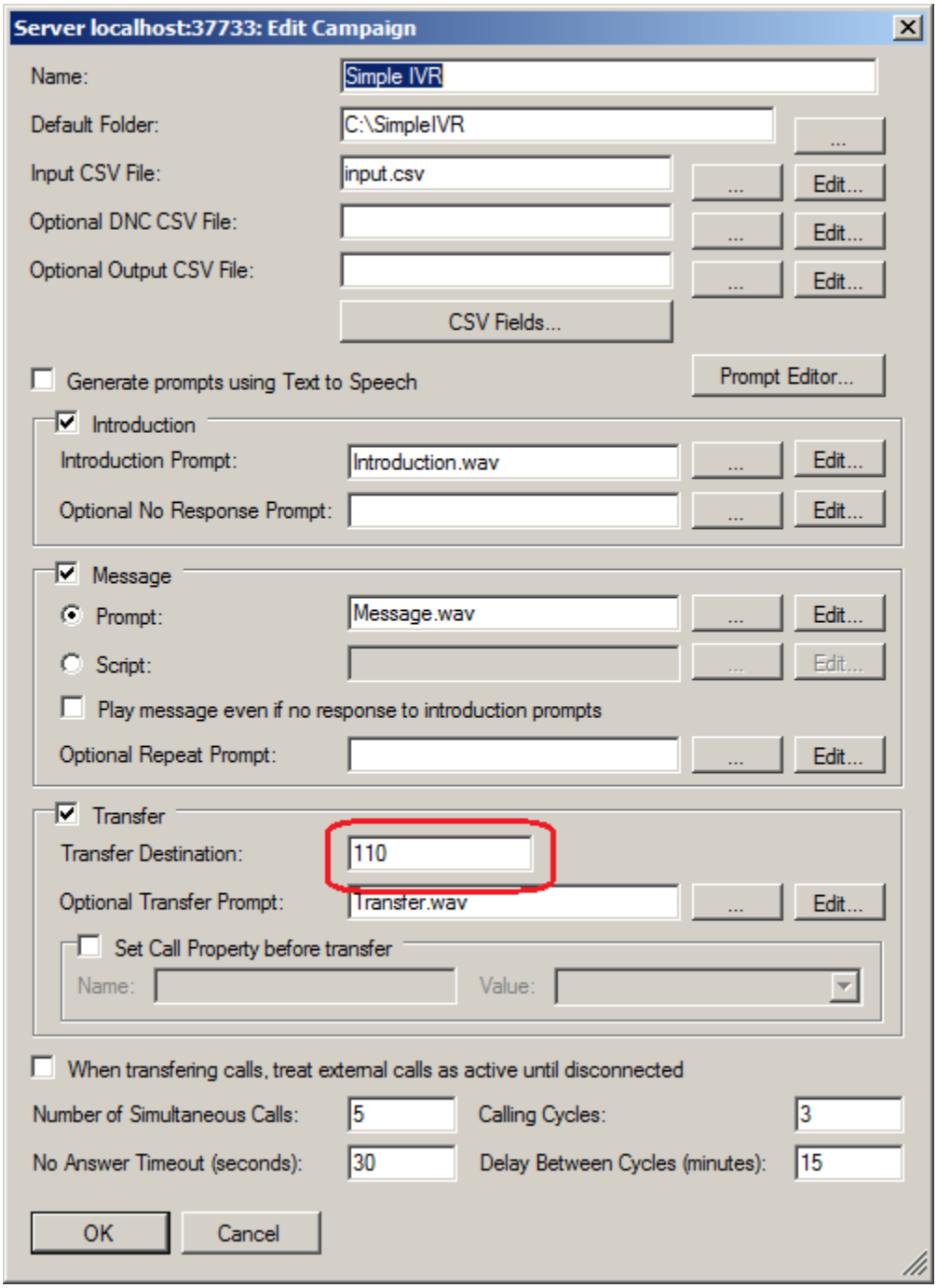
Prompt	Description
Introduction.wav	This is the introductory message played to the called party when they first answer which asks them to enter a key to hear the message: **If you would like to hear an important message from Mitel Professional Services, please press any key now.**
NoResponse.wav	If the called party fails to respond after the Introduction Prompt is played three times, they hear this message: **If you want more information regarding this call, please contact Mitel Professional Services at 408-331-3331**
Campaign Name	The name of the campaign to control. The campaign should have been previously created and configured using the Admin program. If the campaign name contains any spaces, then the **Campaign Name** must be enclosed in quotes.
Message.wav	This is the main message played to the called party. In most campaigns, it is the core message that is always required regardless of the campaign settings. The sample says: **Mitel Professional Services has a number of exciting applications available.**
Repeat.wav	If set, this message plays following the Message Prompt (assuming if there was an Introduction Prompt and that the called party accepted the call or there was no Introduction prompt.) This message must tell the user to press **1** to repeat the Message Prompt. The sample prompt says: **If you'd like to hear this message again, please press the one key.**
Transfer.wav	If enabled, this message gives the called party the opportunity to transfer to, for example, a representative by pressing the asterisk key. The sample prompt says: **If you'd like to speak with someone in the Mitel Professional Services department, please press the star key.**

7. EXAMPLE CAMPAIGNS

The following are some examples of the primary types of campaigns that can be created using this application. Many variations are possible through use of the various settings described throughout this document:

7.1 SAMPLE PROMPTS

The figure below provides an example of a very simple campaign that calls a party and as soon as the call connects, transfers the called party to a specific extension, for example a Mitel Workgroup:

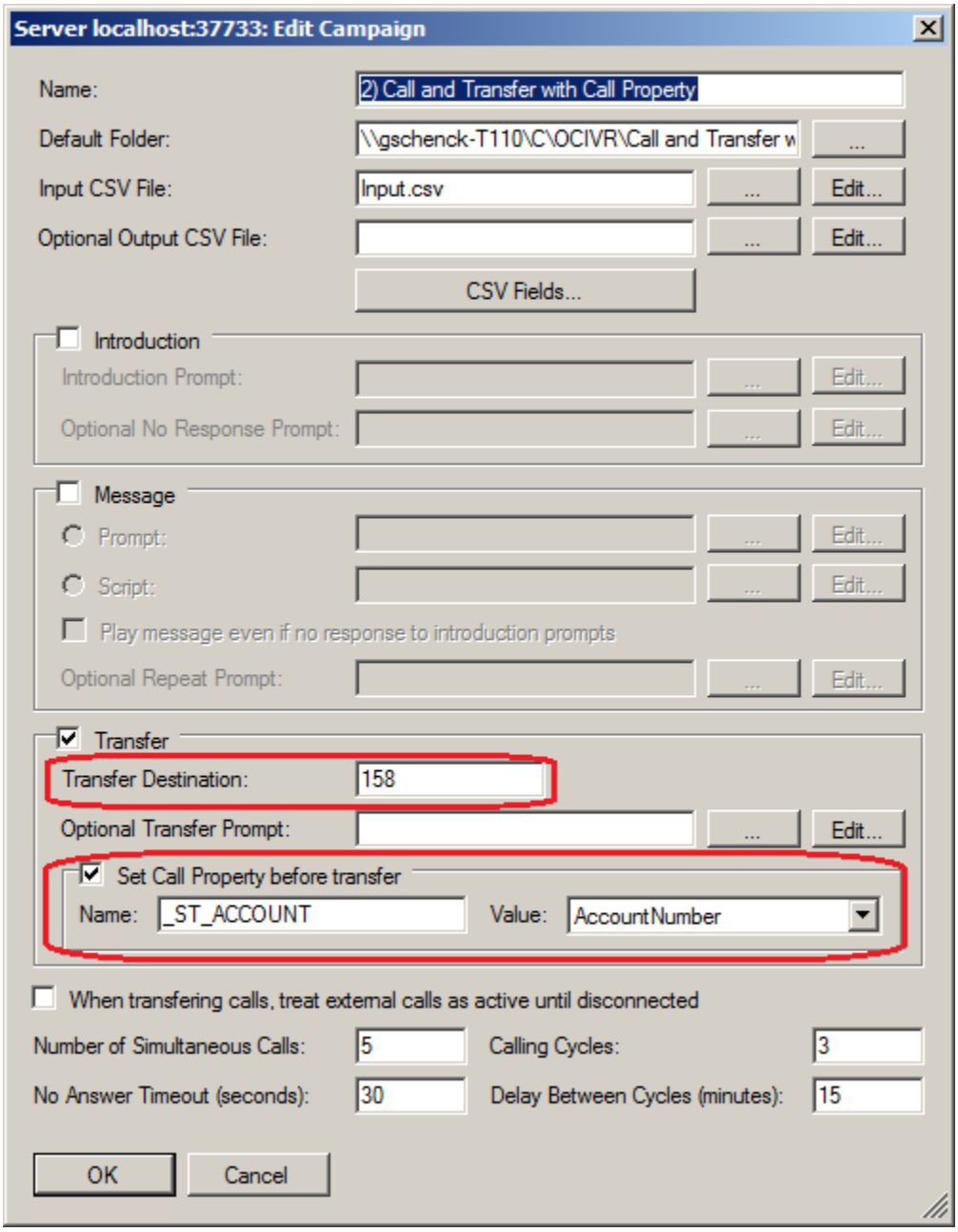


The only settings are specification of the default folder in UNC format where the campaign resources will be found (in this case, just the input file containing the list of parties to call), the name of the input file and

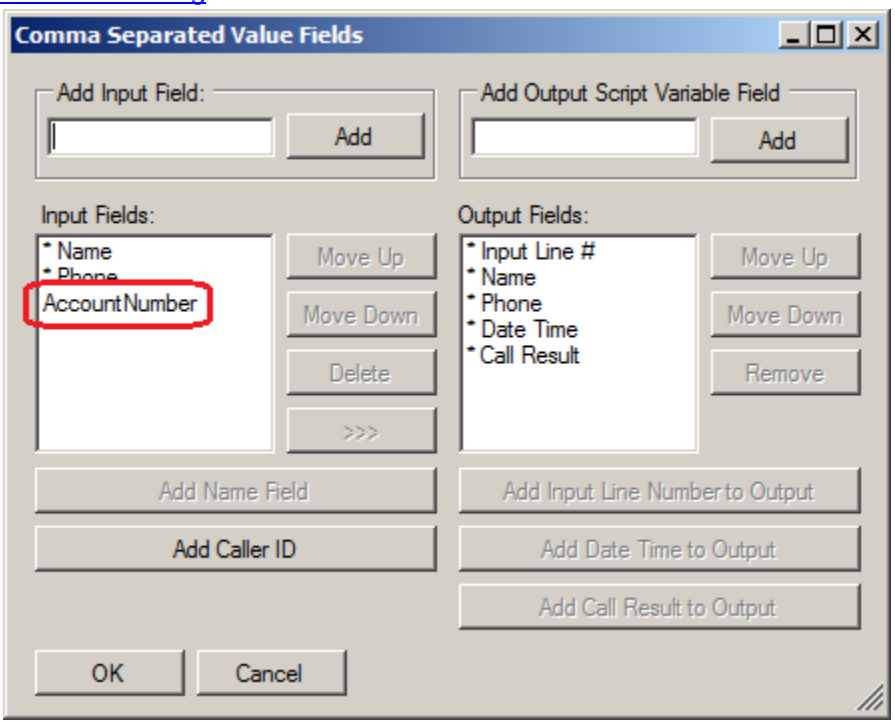
the transfer destination. The called party experience upon answering the ringing call would be to be transferred to extension 110, in this case a Mitel Workgroup extension.

7.2 CALL AND TRANSFER WITH CALL PROPERTY

This is the same as the previous example with an extra step where the application sets the value of the **CSV Input** field named "**Account Number**" as a Mitel call property named **_ST_ACCOUNT**:



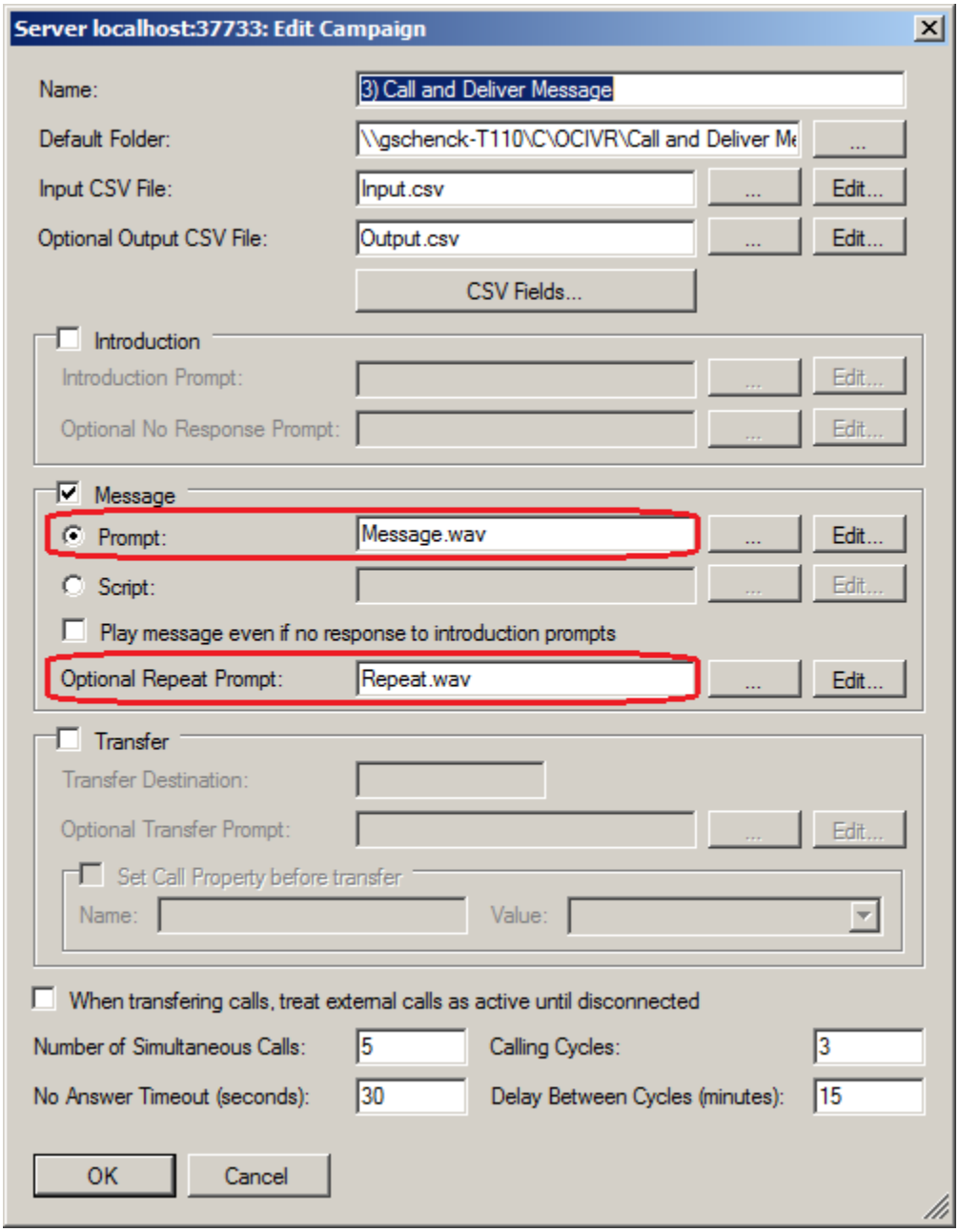
For this example, to work, the CSV input file must contain a **Account Number** field as specified in the CSV Fields Dialog:



7.3 CALL AND DELIVER MESSAGE

This is also a very simple campaign. It calls each party, plays a pre-recorded message, and, at the end of the message, prompts the called party to press the one key (**1**) if they want to hear the message again.

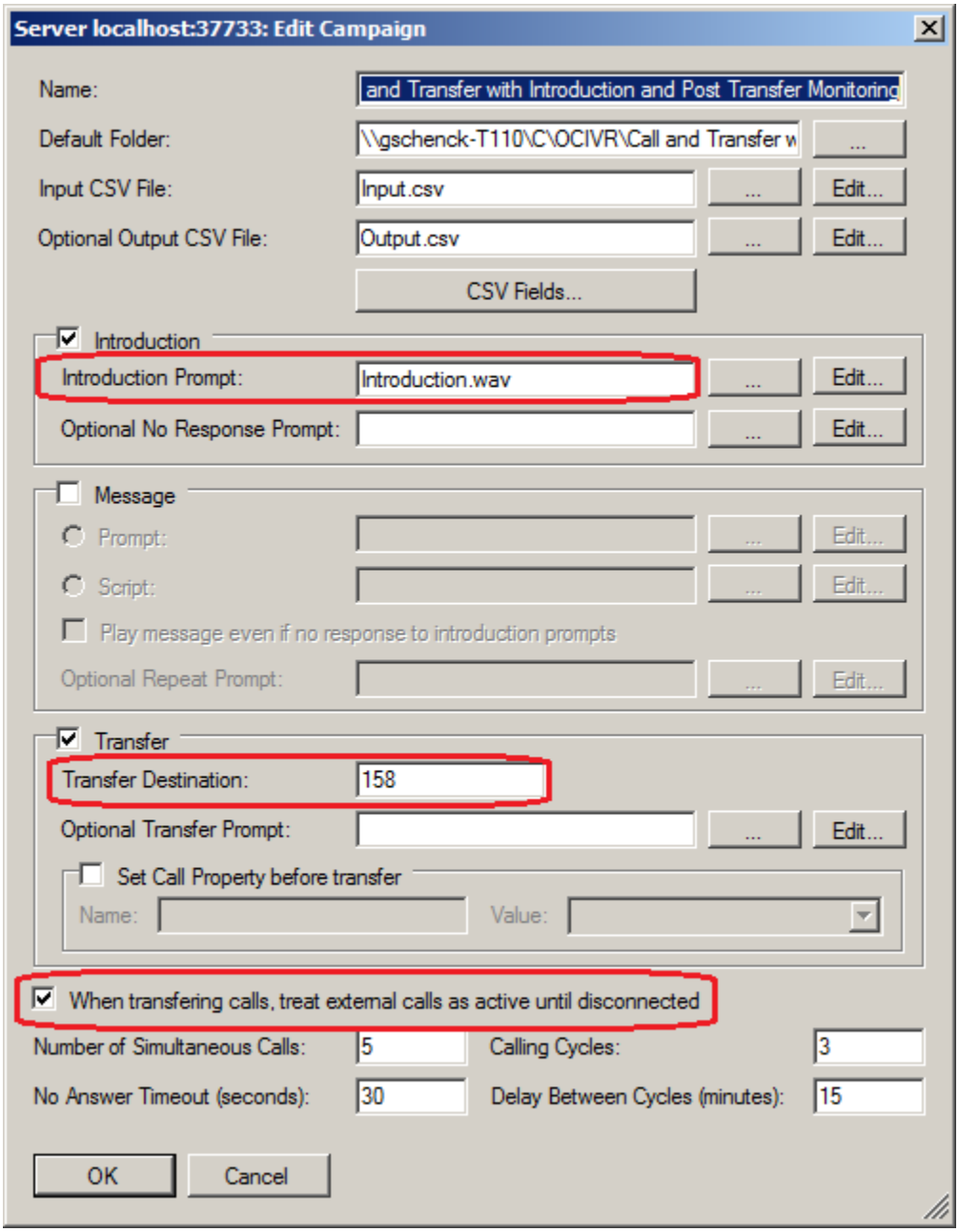
Note! One digit is hard coded in the OCIVR. If you want to use different digits you must use the script feature:



7.4 CALL AND TRANSFER WITH INTRODUCTION AND POST TRANSFER MONITORING

This campaign calls each party, prompts him or her to press any key to accept the call, and then transfers him or her to a Workgroup (x158.) After the transfer operation, the application continues to monitor the external call. If it detects the number of active calls has reached the **Number of**

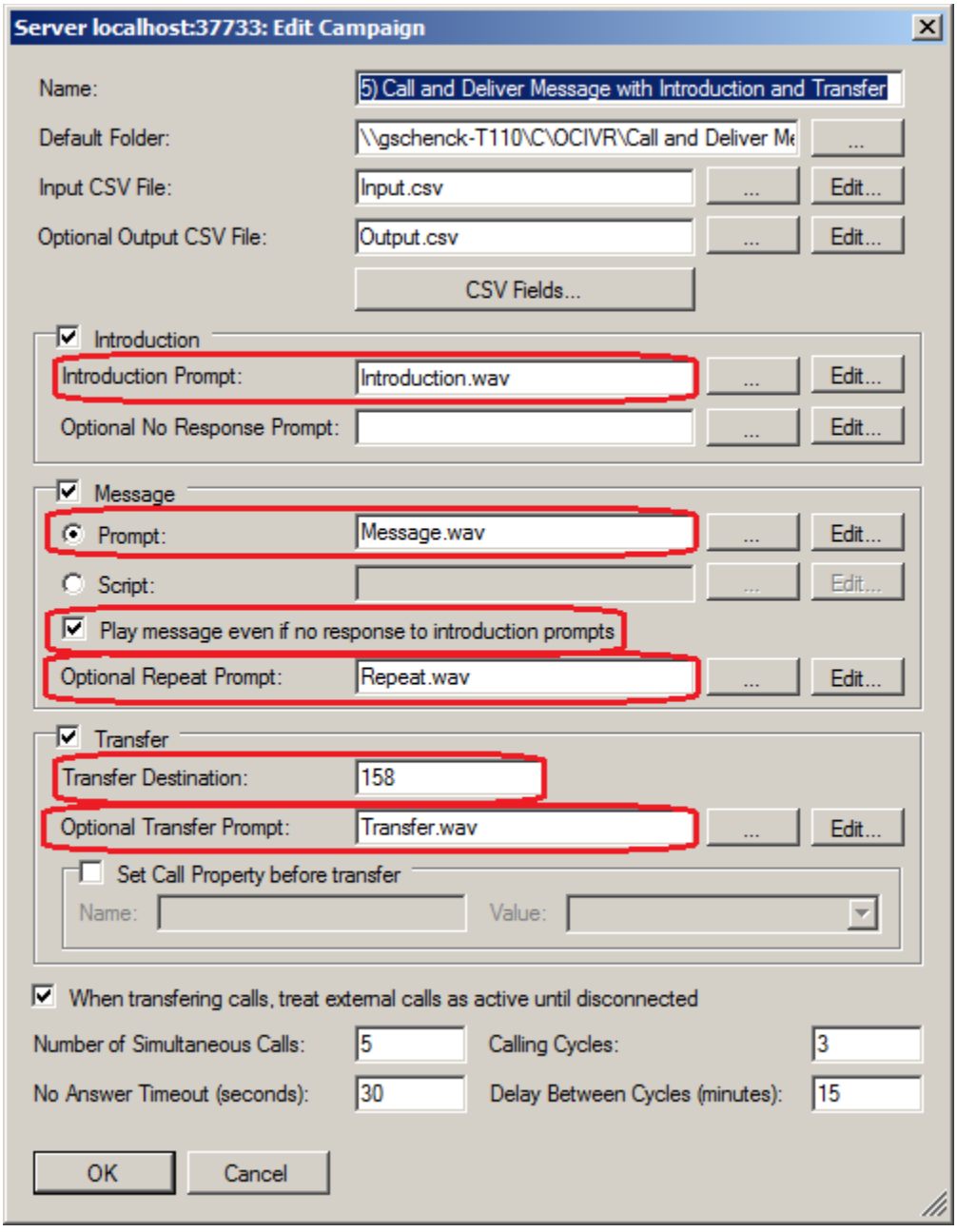
Simultaneous Calls setting, it waits for a call to disconnect before making the next call:



7.5 CALL AND DELIVER MESSAGE WITH INTRODUCTION AND TRANSFER

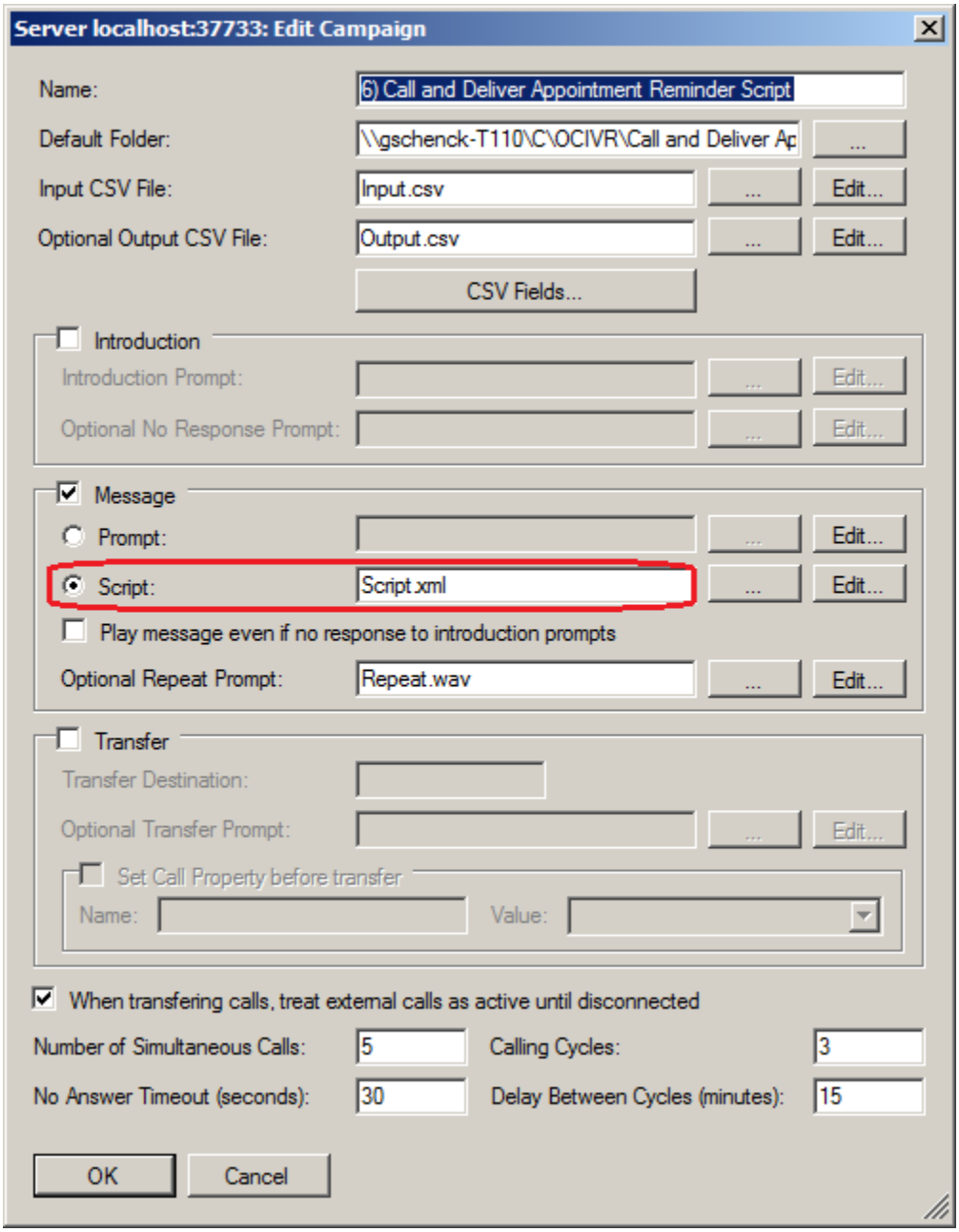
The screenshot below presents a more complicated example that uses a few campaign options. In this scenario, the application calls each party and plays the Introduction Prompt requesting them to press any key to accept the call. If the called party fails to respond, then after two additional re-prompts the Message Prompt is still played because the **Play message even if no response to introduction prompts** is selected. In this case, after the message is played, the call disconnects.

Assuming that the called party presses a key in response to the **Introduction Prompt**, the Message Prompt is played, following the message, the called party is prompted to press the one ('**1**') digit to hear the message again (**Optional Repeat Prompt**) and then is given the option to press the asterisk key ('*') to be transferred to a representative (**Optional Transfer Prompt**). If the called party responds by pressing the asterisk key, they are transferred to extension 158 (**Transfer Destination**):



7.6 CALL AND DELIVER APPOINTMENT REMINDER SCRIPT

This example highlights the use of an OCIVR script by selecting the script feature. The campaign settings are relatively straightforward and are shown in the figure below:

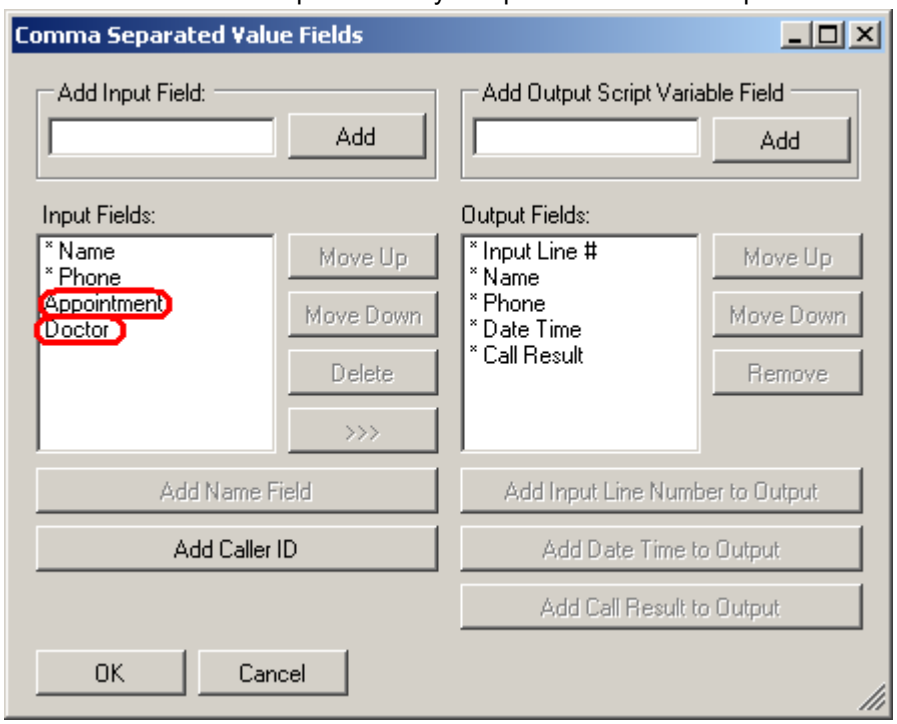


This looks similar to the Play and Deliver Message example except that instead of playing the same pre-recorded message for each party called, it uses a script to customize the Message based on data read from the Input CSV file.

The script is as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<script directory="C:\Outbound Campaign IVR Samples\Call and Deliver Appointment Reminder Script">
  <play>
    <file>"Reminder.wav"</file>
    <datetime>Appointment</datetime>
    <file>"With.wav"</file>
    <file>Doctor</file>
  </play>
</script>
```

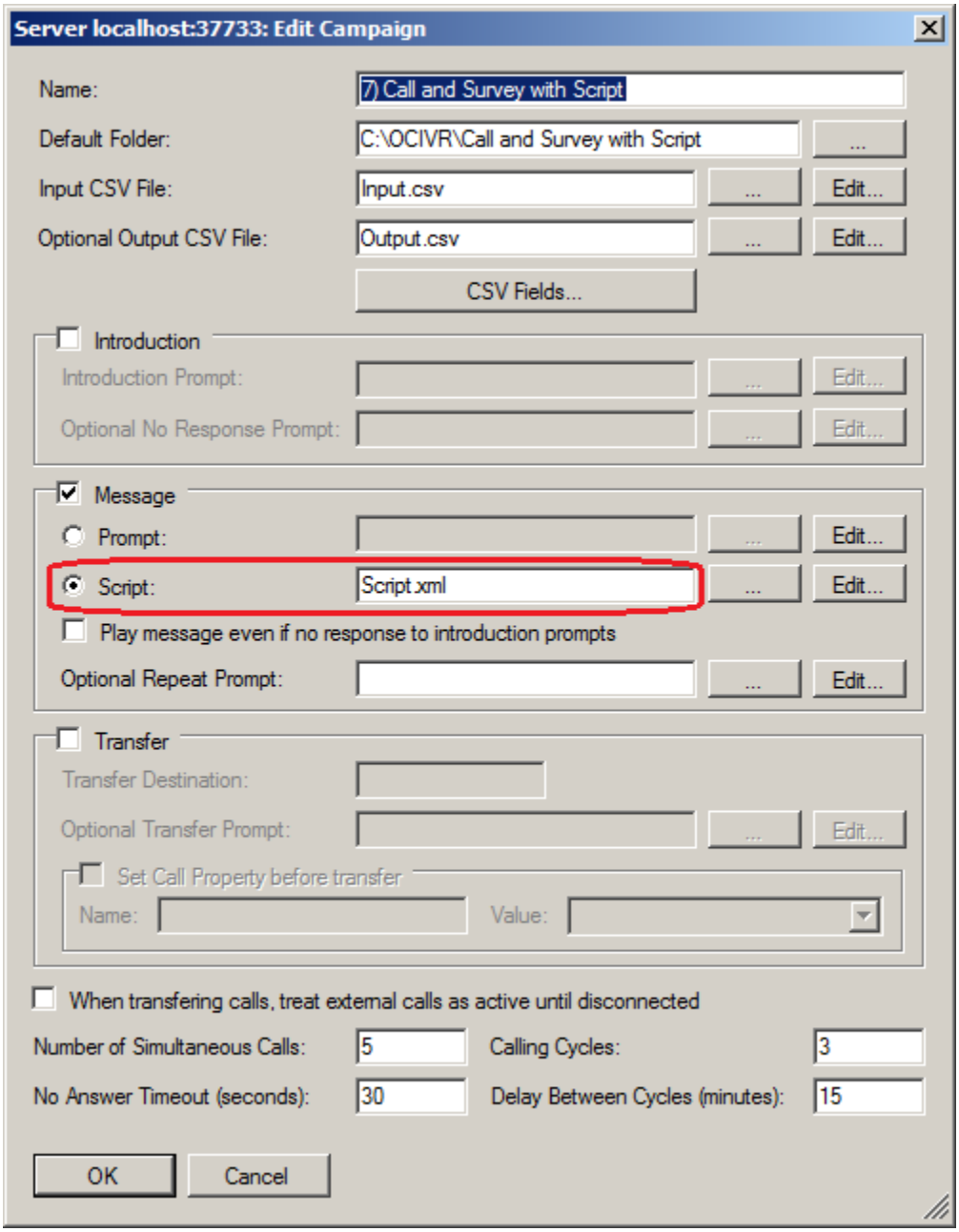
For this script to function, the **Input CSV File** must contain two fields: **Appointment** and **Doctor**. Because the script references the **Appointment** field with a datetime play element, the Appointment data should contain a string that can be interpreted as a valid date and time. The **Doctor** is referenced by a File element which means that the contents of the field should contain the name of a wave file. The wave file should be in the script's directory as specified with the script element's directory attribute:



7.7 CALL AND SURVEY WITH SCRIPT

This is another example of a campaign that plays a script rather than a single pre-recorded prompt. This campaign calls each party and as soon as they connect starts to execute a script. This script first plays a custom greeting which includes the date of their auto service and then asks them if they want to take a

survey about their repair experience. If they do, they are asked three questions and then given the chance to review their responses and either accept them or try again. After they accept their responses, the call is complete. The digits captured in response to the queries are saved as part of the **Output CSV** file:



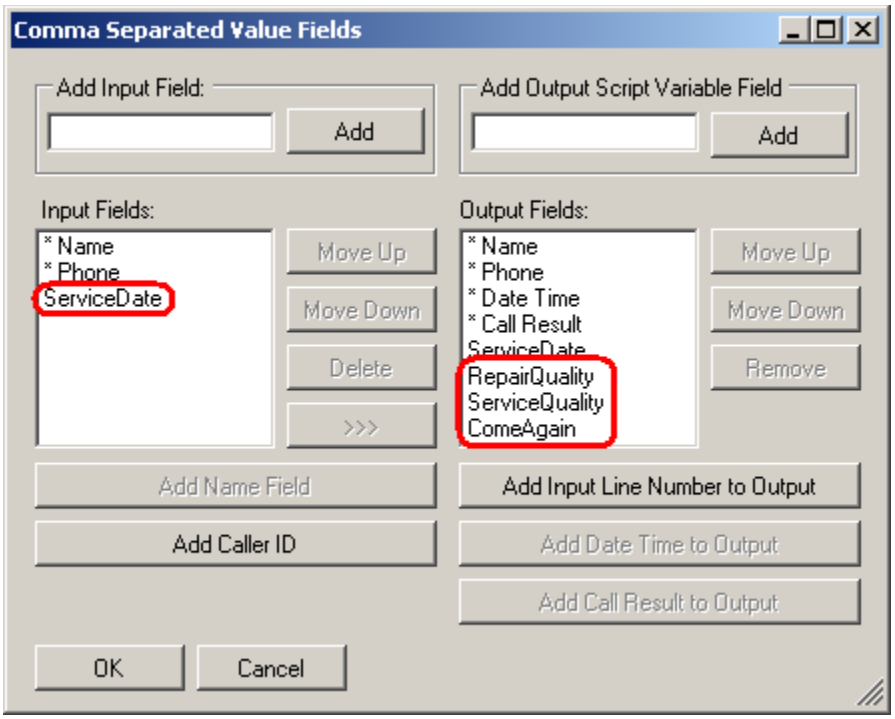
The script starts by playing a custom greeting message including the month and day of their recent service using the **ServiceDate** field from the input *CSV* file. By using the *on1="SURVEY" exitdigits="12"* attributes we get the result that if the called party enters **1** in response to our prompt, the script will branch to the *SURVEY* label and if they enter **2**, then the script will continue on the next script element following the play. In this case, we immediately go to a label at the end of the script where we play a goodbye message and that completes the call.

If they do enter **1** and branch to the *SURVEY* labe,l then we ask them three questions about their experience and then, we play the results back and ask the user to accept or reenter the results and using the *on1="THANK_YOU_GOODBYE" and on2="REPEAT_SURVEY*" we handle the response by either completing the call or looping back up and asking the questions again.

```
<?xml version="1.0" encoding="utf-8"?>


<script directory="C:\Outbound Campaign IVR Samples\Call and Survey with
Script">
  <!-- ======================================== -->
  <play on1="SURVEY" exitdigits="12" repeat="0">
    <file>"Greeting"</file>
    <month>ServiceDate</month>
    <day>ServiceDate</day>
    <file>"OneForYesTwoForNo"</file>
  </play>
  <goto>"THANK_YOU_GOODBYE"</goto>


  <!-- ======================================== -->
  <label id="SURVEY"/>


  <play var="RepairQuality" on1set="High" on2set="Medium"
on3set="Low">
    <file>"RepairQuality"</file>
    <file>"SelectHighMediumLow</file>
  </play>


  <play var="ServiceQuality" on1set="High" on2set="Medium"
on3set="Low">
    <file>"ServiceQuality"</file>
    <file>"SelectHighMediumLow</file>
  </play>


  <play var="ComeAgain" on1set="Yes" on2set="No">
    <file>"ComeAgain"</file>
```

```
    <file>"OneForYesTwoForNo</file>
  </play>

  <!-- ========================================= -->
<label id="SURVEY_VERIFY_RESULTS"/>
  <play on1="THANK_YOU_GOODBYE" on2="REPEAT_SURVEY">
    <file>"SurveyResultsRepairQuality</file>
    <file>RepairQuality</file>
    <file>"SurveyResultsServiceQuality"</file>
    <file>ServiceQuality</file>
    <file>"SurveyResultsComeAgain"</file>
    <file>ComeAgain</file>
    <file>"VerifyResults"</file>
    <file>"OneForYesTwoForNo</file>
  </play>

  <label id="REPEAT_SURVEY"/>
  <set var="RepairQuality"/>
  <set var="ServiceQuality"/>
  <set var="ComeAgain"/>
  <goto>"SURVEY"</goto>

  <!-- ========================================= -->
  <label id="THANK_YOU_GOODBYE"/>
  <play timeout="500">
    <file>"ThankYouGoodbye"</file>
   </play>
   </script>
```

The figure below shows the input field used to provide a custom greeting containing the month and day of their service and the output script variable fields written to in the ***CSV* Fields** dialog:



7.8 PHARMACY REFILL SCRIPT

This example shows a more ambitious script only campaign. Like the last example, the only feature of the campaign used is the Message phase with the script option enabled.

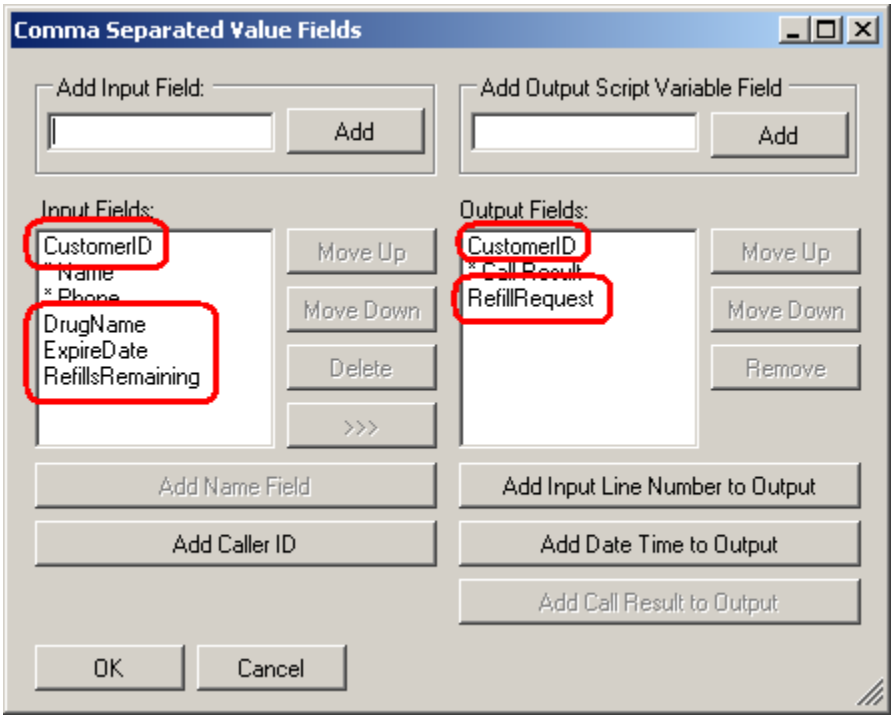
The script is shown below. It takes advantage of several additional script features not seen in previous examples. At the start of the script it creates a script variable containing the extension of the pharmacist. This way the extension can be easily changed if needed without having to dig into the script. Next the script uses a switch element to test if the CSV input variable DrugName is empty. If it is the script, it branches to the *NO_DRUG_NAME_PROVIDED* label. If not, then it prompts the party with details of their prescription refill including the drug name. The branch to *NO_DRUG_NAME_PROVIDED* also prompts the party but with a more generic message that does not try to play the drug name. The idea is that you might record a set of mainstream drug names and pass those as an input field but still have the campaign make sense if the drug name isn't one that you've recorded a prompt for. For this scheme to work, the **Input *CSV* File** would have to leave the **Drug Name** field blank if it is not in the list of recorded drug names. The play elements for both the specific and generic drug names use the on*set and on* attributes to set a specific value and branch within the script in response to the party's input to the prompt. In this case, they have three choices, entering **1** to request a refill, **2** to decline, and **3** to speak with a pharmacist. In addition, we also handle the case of no response (in case we get an answering machine.) If they enter **3**, we branch to the TRANSFER label where we use the *setproperty* element to set a number of Mitel call properties based on the data associated with the call and then we use the transfer element to actually transfer the call to the pharmacist's number defined at the start of the script. Setting the call properties this way would make sense if a separate client application such as the Mitel Professional Services Generic CRM Connector (also known as, EasyPop) is running and can display the contents of the properties as well as respond in other ways to the data.

```
<?xml version="1.0" encoding="utf-8"?>

<script directory="C:\Outbound Campaign IVR Samples\Pharmacy Refill">
  <!-- Constants -->
  <define var="PhamacistNumber">"109"</define>

  <!-- Did they provide a drug name? -->
  <switch var="DrugName">
    <case value="">"NO_DRUG_NAME_PROVIDED"</case>
  </switch>

  <!-- They provided a drug name -->
  <play repeat="1" var="RefillRequest" on1set="Yes" on2set="No"
      on3set="Transfer" onnoresponseset="NoResponse" on1="REFILL_YES"
      on2="REFILL_NO" on3="TRANSFER" onnoresponse="REFILL_NO">
    <file>"Greeting"</file>
    <file>"Prescription"</file>
    <file>DrugName</file>
    <file>"Expire"</file>
    <dayoftheweek>ExpireDate</dayoftheweek>
    <month>ExpireDate</month>
    <day>ExpireDate</day>
    <file>"YouHave"</file>
    <int>RefillsRemaining</int>
    <file>"Remaining"</file>
    <file>"Refill"</file>
    <file>"OneTwoThreeForYesNoPharmacist"</file>
  </play>

  <label id="NO_DRUG_NAME_PROVIDED"/>
  <play repeat="1" var="RefillRequest" on1set="Yes" on1="REFILL_YES"
      on2set="No" on2="REFILL_NO" on3set="Transfer" on3="TRANSFER"
      onnoresponseset="NoResponse" onnoresponse="REFILL_NO">
    <file>"Greeting"</file>
    <file>"PrescriptionGeneric"</file>
    <dayoftheweek>ExpireDate</dayoftheweek>
    <month>ExpireDate</month>
```

```
    <day>ExpireDate</day>
    <file>"YouHave"</file>
    <int>RefillsRemaining</int>
    <file>"Remaining"</file>
    <file>"Refill"</file>
    <file>"OneTwoThreeForYesNoPharmacist"</file>
  </play>

  <label id="REFILL_YES"/>
  <play timeout="0">
    <file>"Pickup"</file>
  </play>
  <goto>"EXIT"</goto>

  <label id="REFILL_NO"/>
  <play timeout="0">
    <file>"Callback"</file>
  </play>
  <goto>"EXIT"</goto>

  <label id="TRANSFER"/>
  <play timeout="0">
    <file>"PleaseHold"</file>
  </play>
  <setproperty name="_ST_CUSTOMER_ID">CustomerID</setproperty>
  <setproperty name="_ST_CUSTOMER_NAME">* Name</setproperty>
  <setproperty name="_ST_CUSTOMER_PHONE">* Phone</setproperty>
  <setproperty name="_ST_CUSTOMER_DRUG_NAME">DrugName</setproperty>
  <setproperty name="_ST_CUSTOMER_EXPIRE_DATE">ExpireDate</setproperty>
  <setproperty
name="_ST_CUSTOMER_REFILLS_REMAINING">RefillsRemaining</setproperty>
  <transfer>PhamacistNumber</transfer>

  <label id="EXIT"/>
  <play timeout="500">
    <file>"ThankYouGoodbye"</file>
```

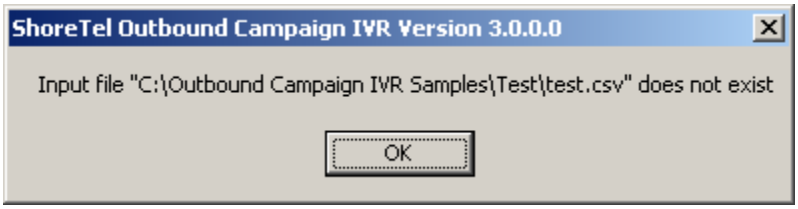
```
  </play>
</script>
```

This shows the **Input Fields** used to provide a custom greeting containing the customer ID and prescription information and the output script variable fields written to in the ***CSV* Fields** dialog:



8. TROUBLESHOOTING CAMPAIGNS

For a campaign that uses **Prompt** or **Text** features, the main things that can cause problems are referencing an input or prompt file which does not exist at the specified location. These errors are detected when the script is executed and are fatal, the campaign will not call any of the parties in the list. For example, starting a campaign which references a nonexistent *Input CSV File* would result in this popup error:



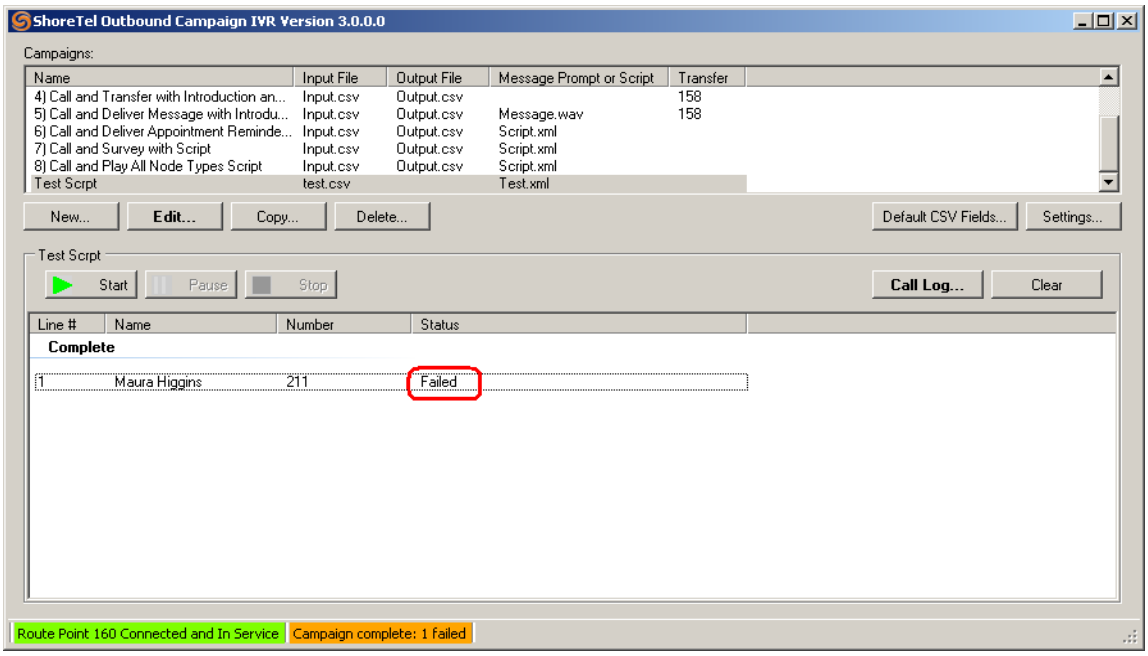
Similarly, starting a campaign that references a nonexistent prompt file would result in this popup error:

Campaigns with Scripts involve more complicated problems. You must carefully test a campaign involving a script with a few test parties before doing any kind of mass calling to actual customers. This is because invalid references in a script will not be detected until its run time which means it does not happen until campaign calls progress to that point. Consider the following script:

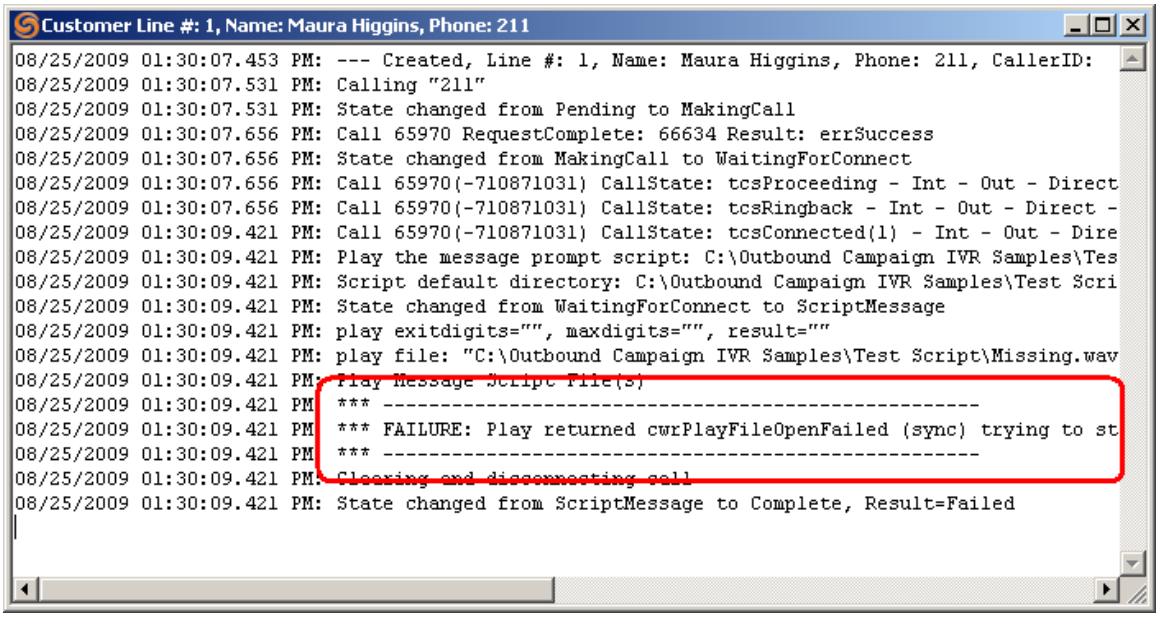
```
<?xml version="1.0" encoding="utf-8"?>

<script directory="C:\Outbound Campaign IVR Samples\Test Script">
  <play>
    <file>"Missing.wav"</file>
   </play
        >
    </script>
```

This script has a single play statement with a literal wave file named "Missing.wav" which does not exist at *C:\Outbound Campaign IVR Samples\Test Script* on the hosting OCIVR server instance. If this script is executed as part of a campaign with a single party in the Input *CSV* file, then the call would be made but as soon as the called party answered the call would fail as shown here:

Double-clicking the call or clicking the **Call Log** option with the failed entry selected would display these results:



Notice the highlighted failure message.

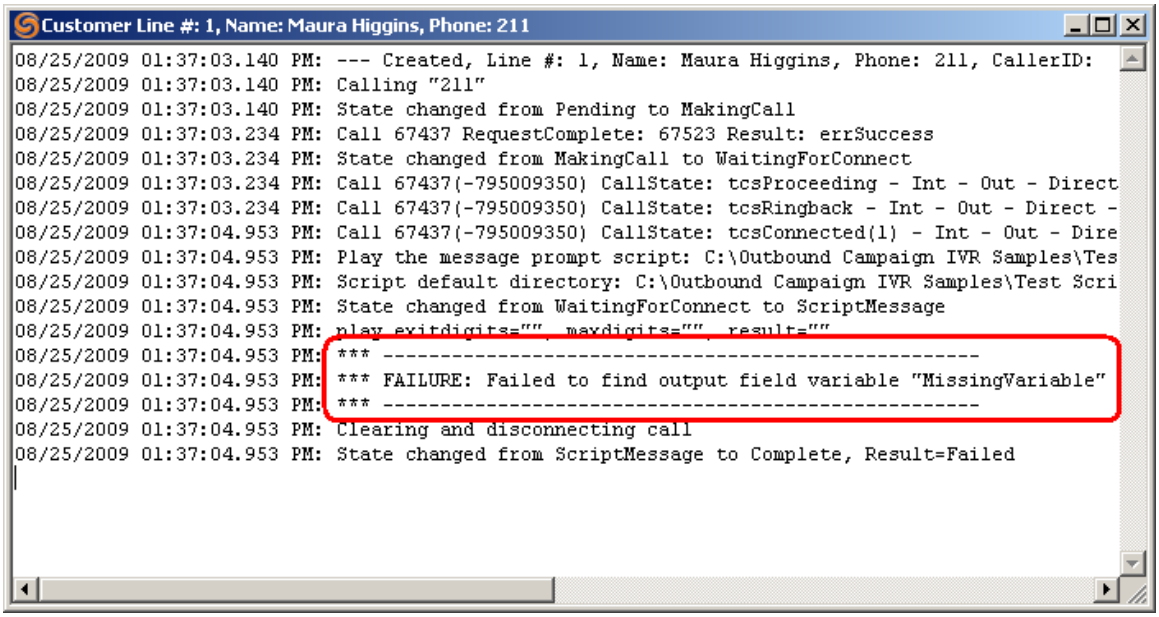


As another example, imagine a similar script but in this case, because the file element data is not quoted, the MissingVariable is assumed to represent a ***CSV* Input** field. However, for this test we do not define an input field of that name:

```
<?xml version="1.0" encoding="utf-8"?>
<script directory="C:\Outbound Campaign IVR Samples\Test Script">
```

```
<play>
  <file>MissingVariable</file>
   </play>
  </script>
```

Executing this script also results in an error when the called party answers. This shows the corresponding call log:

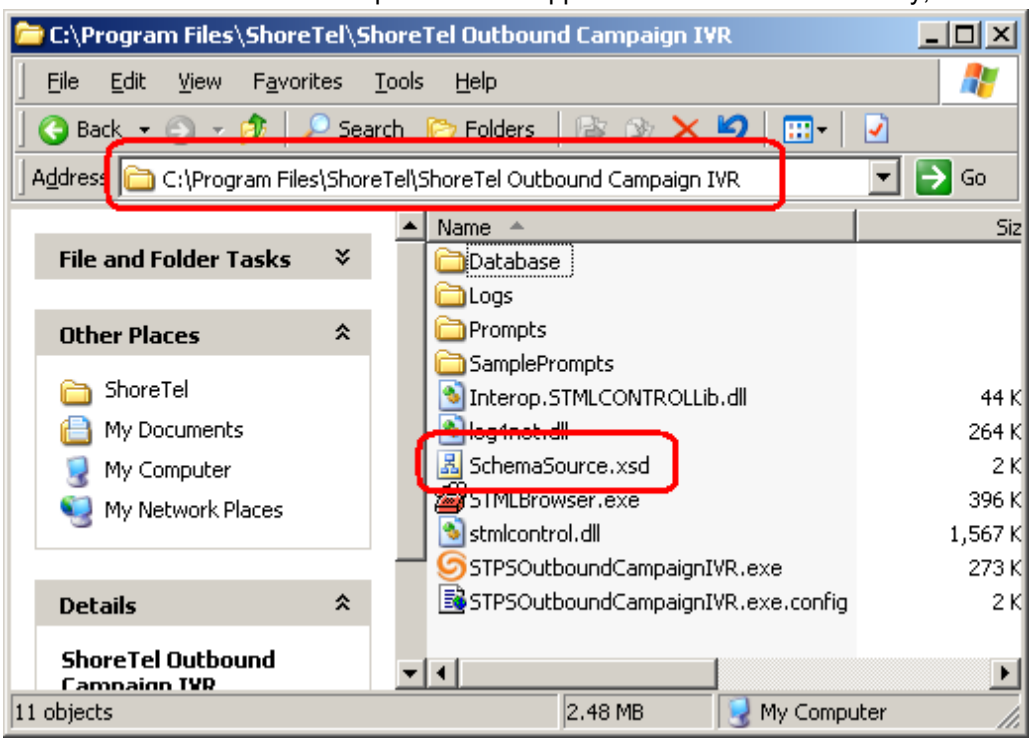


Script debugging requires repeated trials until a script runs to completion with all script scenarios exercised. Even if a script works correctly for a single call, it may fail when calling other parties if it references missing *CSV* input variables. For example, a script may play a file based on the value of a *CSV* input field. However, if the underlying file for one customer does not exist, then this will result in a failure calling that customer while calls to other customers will succeed.

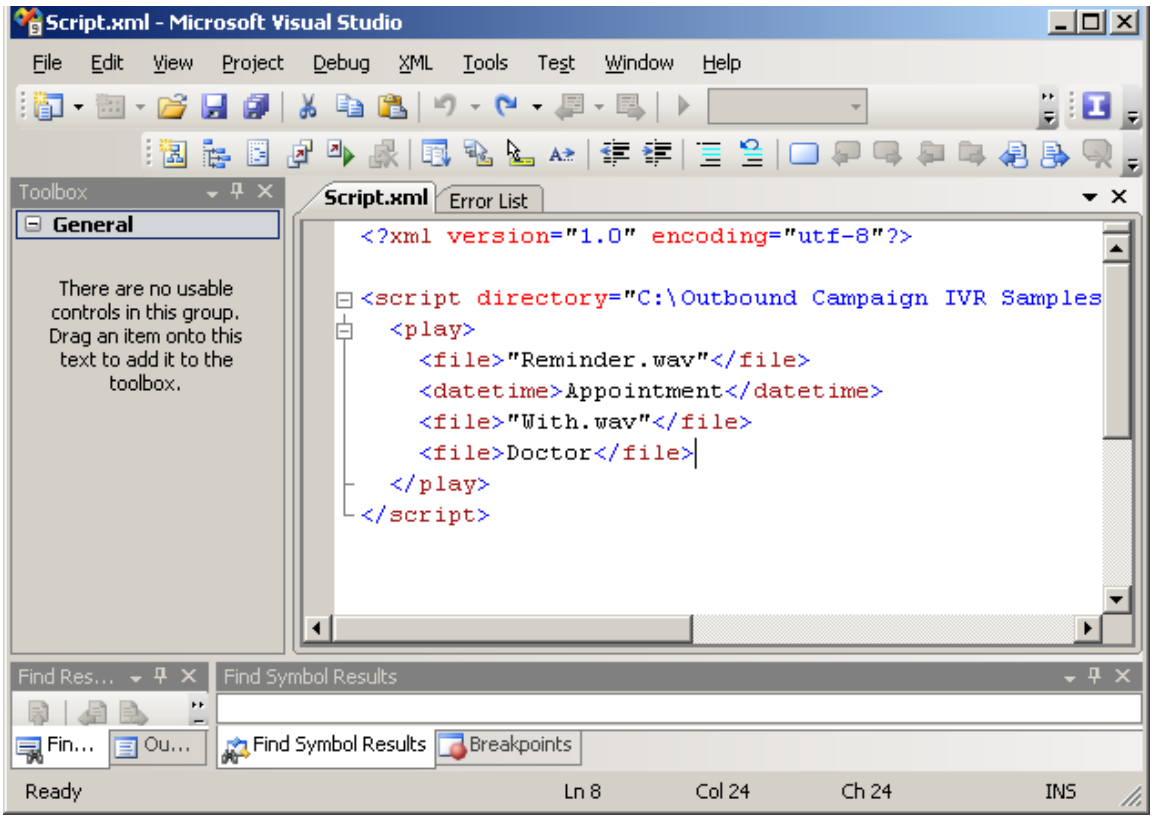
9. DEVELOPING SCRIPTS

The system provides an easy way to edit a script by entering the name of a script file in the **Campaign Configuration** dialog and clicking the **Edit** option. Assuming the file name ends in xml, then either the system default editor for xml files will be used or the configured **XML File Editor** from the **Admin Settings** dialog will be used. The default at **Admin Settings** dialog is **Windows Notepad** and, if cleared, the typical default is to open the XML file in Microsoft Internet Explorer which is only useful for viewing the file as opposed to editing the file.

One of the files distributed with the OCIVR application software package is a file called *SchemaSource.xsd*. This file is placed in the application's installation directory, for example:

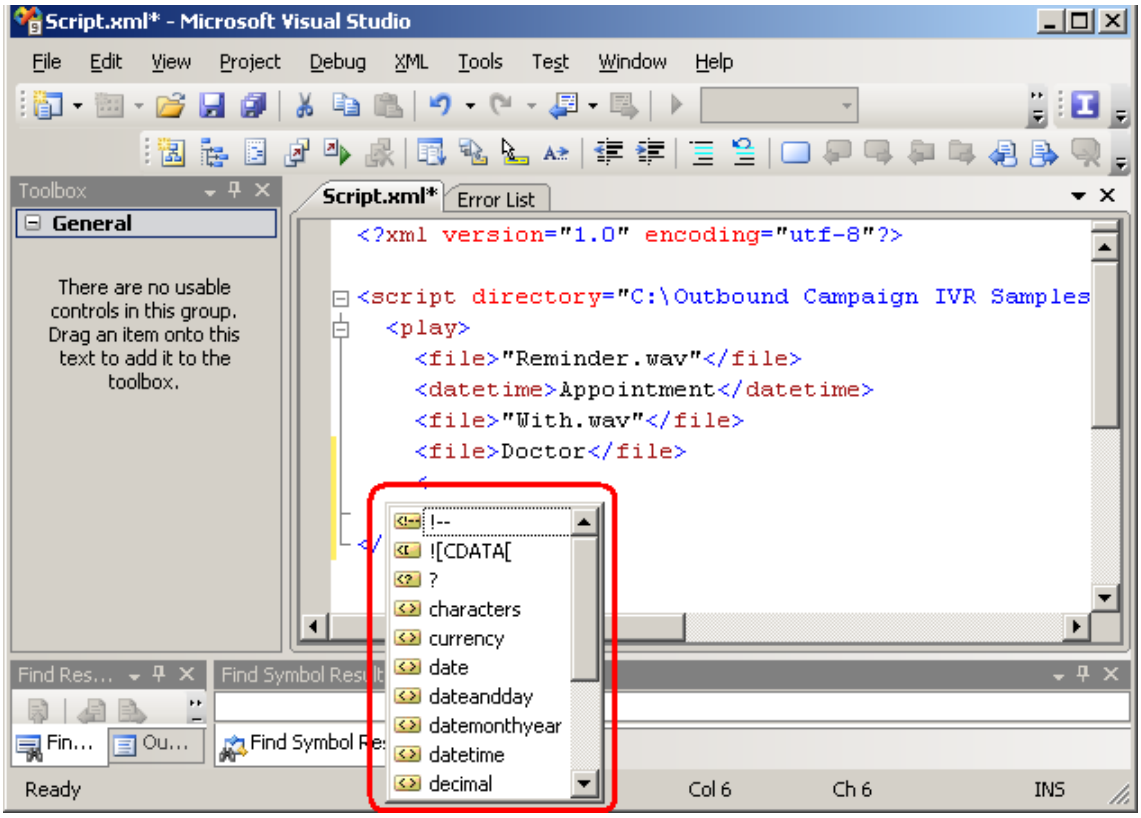


Some XML editors allow you to reference an XSD file to provide intelligent prompting and verification of XML validity. For example, using Microsoft's Visual Studio 2008, an XML file can be edited with XML awareness by loading the file into VS 2008. The following graphic shows a simple script loaded into Visual Studio 2008:



If the XML menu item “Schemas” is invoked and then in the schema dialog the **Add** option is clicked and the **SchemaSource.xsd** file located in the application directory is added, then additional IntelliSense features are now supported:

Within the Visual Studio editor this can be seen by typing the less then (<) character under an existing play element:



Using the XSD file in conjunction with an XML editor which supports it will help to generate valid XML script files.

10. APPENDIX A: CONFIGURING THE TEXT TO SPEECH SUPPORT

This application uses the Microsoft Server Text-to-Speech (TTS) engine to convert textual information into audio streams. To use TTS, customers will need to install the Microsoft Speech Platform Runtime (x86 version) and the voice file along with a voice for playback.

To configure the Test-to-Speech support, follow these steps:

1. Install the Microsoft Speech Platform x86 Runtime:
    a. The Microsoft Text-to-Speech components are now distributed in the zip package with the Advanced Application code. To install, locate the MSTextToSpeechRuntime.zip and unzip the two files to the hard drive.
    b. Run the **SpeechPlatformRuntime.msi** file and follow the prompts to complete the installation.
2. Install the voice file and select the voice:
    a. On the same computer, run the **MSSpeech_TTS_en-US_Helen.msi** file to install the voice files.
    b. There might be other voice files available from Microsoft as part of their Software Development Kits, but these are not distributed with the Mitel application. To select the voice file to use, run the admin program and after selecting the server, navigate to the **Settings** dialog, and in the **Text to Speech** pane at the top right corner, select the **Voice Name** drop-down list and then select a voice as shown in the following screen capture: