Avaya Aura® Contact Center Screen Pop within agent desktop with multiple parameters

Abstract

AACC 6.2 SIP supports an out of the box screenpop where a single URL and a single parameter passed to the URL is popped in or outside the Avaya Aura® Agent Desktop. That single parameter passed has to be one of the contact intrinsics. This application note will combine multiple parameters in the string passed to the URL using the Orchestration Designer application which routes the contact to the agent. This solution is only possible for voice contacts being routed in a SIP AACC environment.

Introduction

Although in an AACC SIP 6.2 environment it is only possible to add one parameter to a URL added in the multimedia admin tool this one parameter could contain multiple values within it when passed to the URL for popping.

In this document an example of how this could be carried out using AACC is outlined. For our example the google search engine URL is used as the screen pop.

If a sample URL to pop is http://www.google.com/search?q=CLID%3D12345+CDN%3D56789 although two parameters are required the value of the CLID and CDN could be combined within an orchestration designer call variable and attached to the URL in one parameter.

CCMM admin tool would store URL: http://www.google.com/search?q=%VALUE%

Adding a new intrinsic allows us to store the combined variables together on the contact: “Search_URL”.

In our sample within orchestration designer a call variable is used to combine the text “CLID%3D” and “+CDN%3D” with the values for the CLID and the CDN giving a combined variable value of “CLID%3D12345+CDN%3D56789”.

Using the exposed AACC web service Contact Service it is possible to attach the new call variable value to the contact within the contact intrinsics.

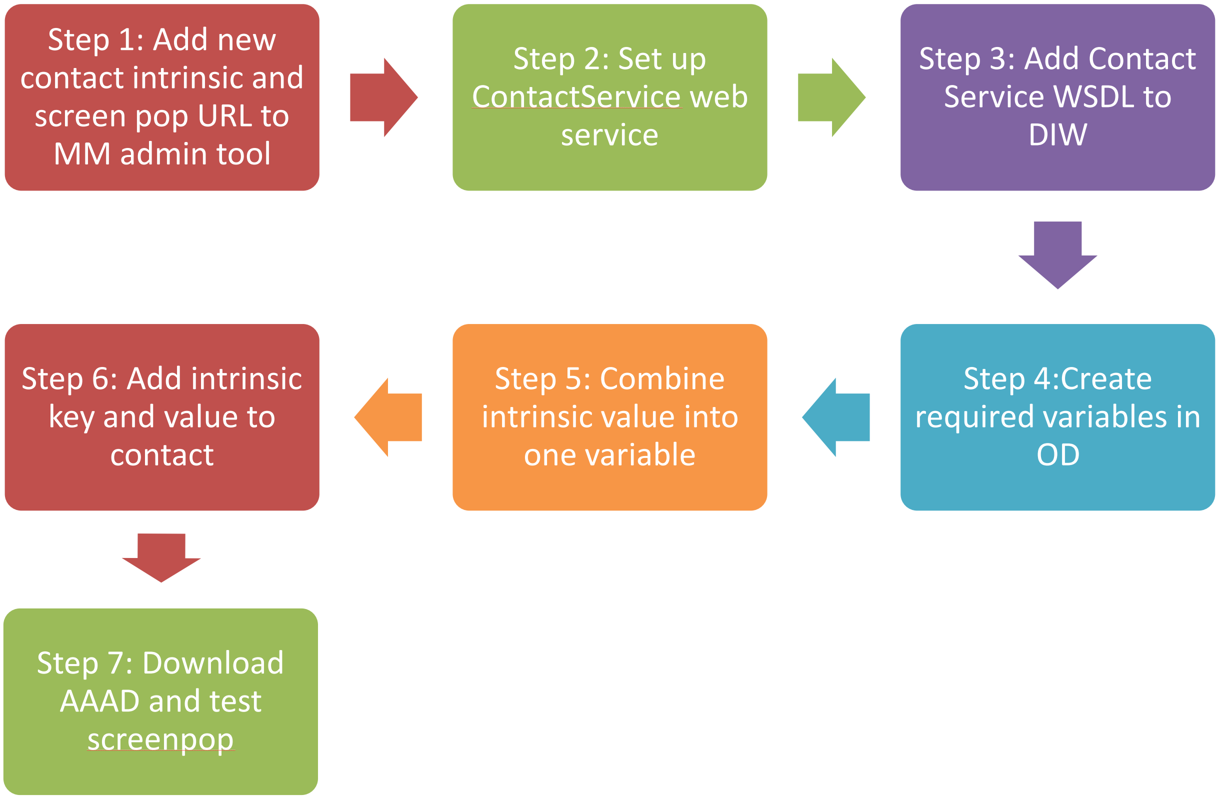
The format of this call variable which when attached to the contact is stored on it as an intrinsic must be agreed with the owner of the URL who will be responsible for parsing.

When the contact arrives on the agent desktop the full URL is available for screenpop.

This approach will only work on voice contacts in a SIP AACC environment.

There is an 80 character limitation on the size of the orchestration designer call variable, it was increased to 80 in AACC 6.2 SP5 RU03.

At a high level the steps to produce the google screenpop with two intrinsic values passed are as follows

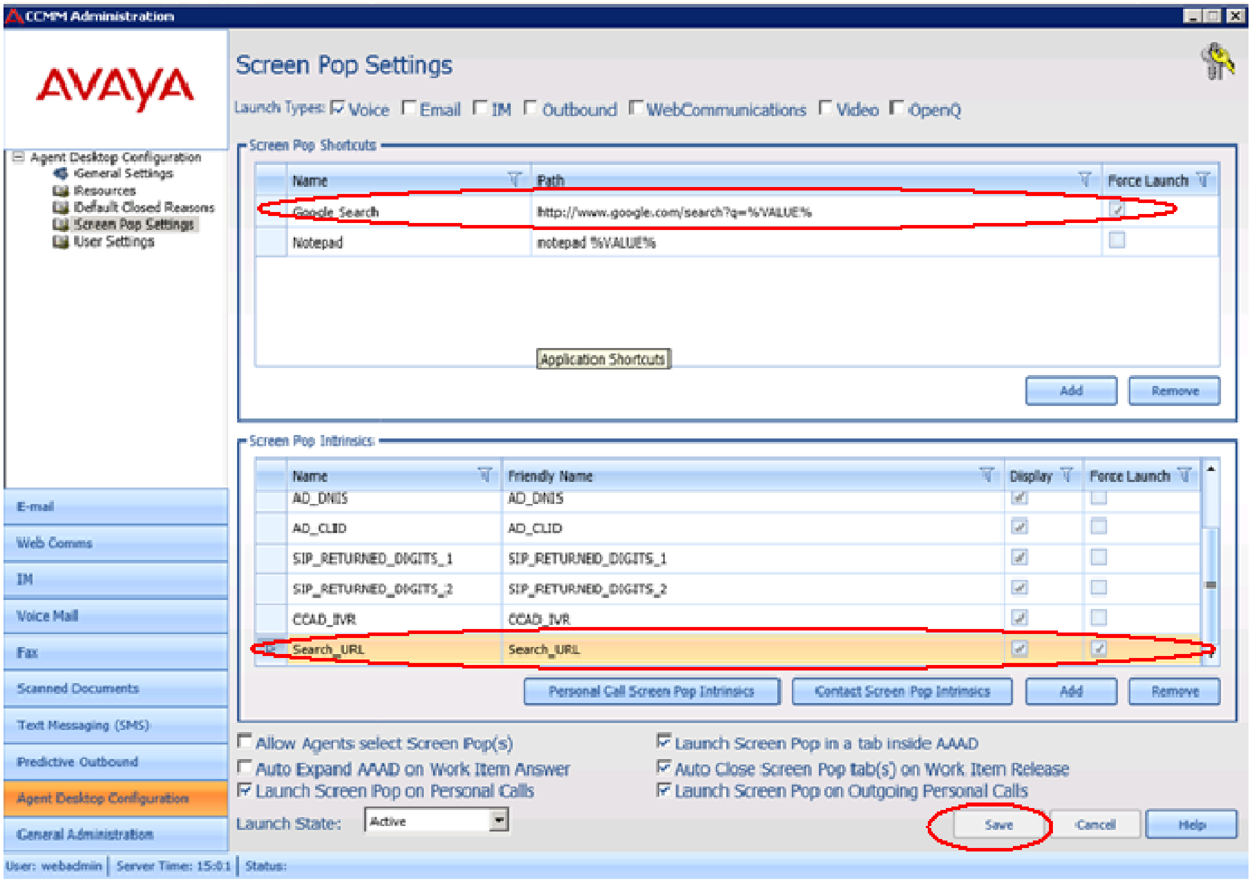

Step 1: Add new contact intrinsic and screen pop URL to MM admin tool
Step 2: Set up ContactService web service
Step 3: Add Contact Service WSDL to DIW
Step 4:Create required variables in OD
Step 5: Combine intrinsic value into one variable
Step 6: Add intrinsic key and value to contact
Step 7: Download AAAD and test screenpop

Step 1: CCMM Admin – adding URL and intrinsic details to pop

From the CCMA Launchpad, select Multimedia and launch the Multimedia admin tool.

Select “Agent Desktop Configuration” and the “Screen Pop Settings”.

Add the URL to the upper section of the form Screenpop Shortcuts and the new intrinsic key which will hold both search values in the lower Screen Pop Intrinsics section:



Screen Pop Settings

Launch Types: ☑ Voice ☐ Email ☐ IM ☐ Outbound ☐ WebCommunications ☐ Video ☐ OpenQ

Screen Pop Shortcuts

Name	Path	Force Launch
Google Search	http://www.google.com/search?q=%VALUE%	☑
Notepad	notepad %VALUE%	☐

Screen Pop Intrinsics

Name	Friendly Name	Display	Force Launch
AD_DNIS	AD_DNIS	☑	☐
AD_CLID	AD_CLID	☑	☐
SIP_RETURNED_DIGITS_1	SIP_RETURNED_DIGITS_1	☑	☐
SIP_RETURNED_DIGITS_2	SIP_RETURNED_DIGITS_2	☑	☐
CCAD_IVR	CCAD_IVR	☑	☐
Search_URL	Search_URL	☑	☑

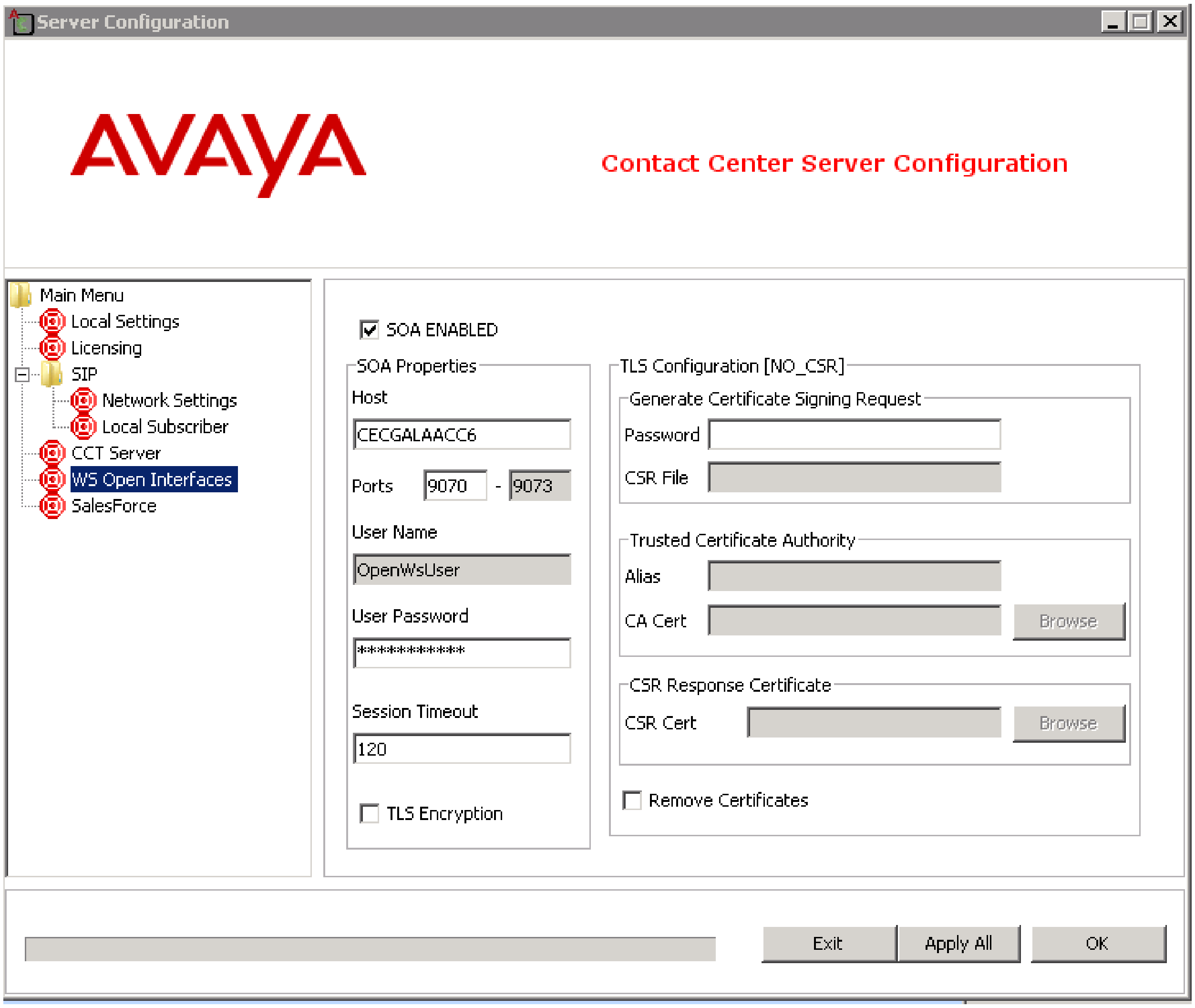
☐ Allow Agents select Screen Pop(s)
☐ Auto Expand AAAD on Work Item Answer
☑ Launch Screen Pop on Personal Calls
☑ Launch Screen Pop in a tab inside AAAD
☑ Auto Close Screen Pop tab(s) on Work Item Release
☑ Launch Screen Pop on Outgoing Personal Calls

Launch State: Active

User: webadmin | Server Time: 15:01 | Status:

Step 2: Set up ContactService webservice in AACC - Used to attach the new intrinsic value

Enable the ContactService web service by enabling SOA in AACC, the SOA developer kit license must be included in the AACC order. On the AACC server open the Contact Center Server Configuration application, Start -> All Programs -> Avaya -> Contact Center -> Manager Server -> Server Configuration.



☑ SOA ENABLED

SOA Properties

Host: CECGALAACC6

Ports: 9070 - 9073

User Name: OpenWsUser

User Password: ************

Session Timeout: 120

☐ TLS Encryption

☐ Remove Certificates

On the WS Open Interfaces form tick the SOA enabled check box and note the ports configured.

The username and password configured on this form is required when calling the ContactService web service.

Ensure TLS encryption is unticked if it is not relevant.

The WSDL to the webservices should be tested in Internet explorer to ensure they are up: http://localhost:9070/SOAOI/scripting/services/ContactService?WSDL this WSDL value will be required when configuring the web service in the database integration wizard for use within orchestration designer scripts.

Orchestration Designer

An application developed within the orchestration designer tool will be used to route voice contacts to the correct skillsets in the contact center. It is possible to combine contact intrinsics into one orchestration designer call variable and then using the exposed ContactService web service add that call variable value to the intrinsics of the contact. This is then available on the AAAD for use on the screen popped URL.

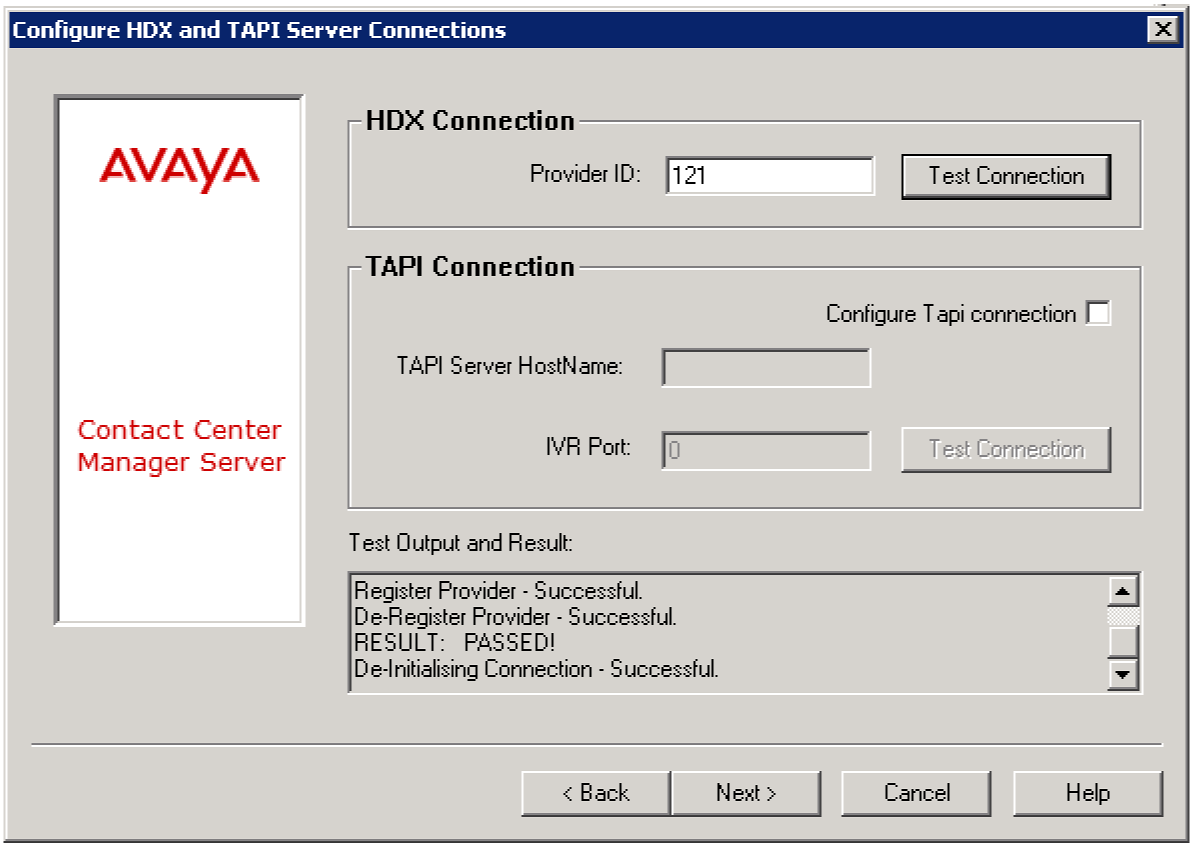
Step 3: Add the ContactService web service to the Database Integration Wizard

Before a web service can be called from an application within orchestration designer it must first be entered in the database integration wizard application.

From the server to open the DIW, Start -> All Programs -> Avaya -> Contact Center -> Manager Server -> Database Integration Wizard.

DIW Form 1 Configure HDX and TAPI Server Connections

Enter a value of 121 for the HDX Connection if one is not already populated. Click the button marked "Test Connection" and check the output to ensure the connection was successful. If failure, try a different number. There is a predefined global variable called HDX_AppId pre-populated with 121. It can be used when calling the web services. If 121 is not successful registering in DIW HDX_AppId should be updated to the successfully registered number. Ignore the TAPI Connection section it is not relevant.



Provider ID: 121

Configure Tapi connection ☐

TAPI Server HostName:

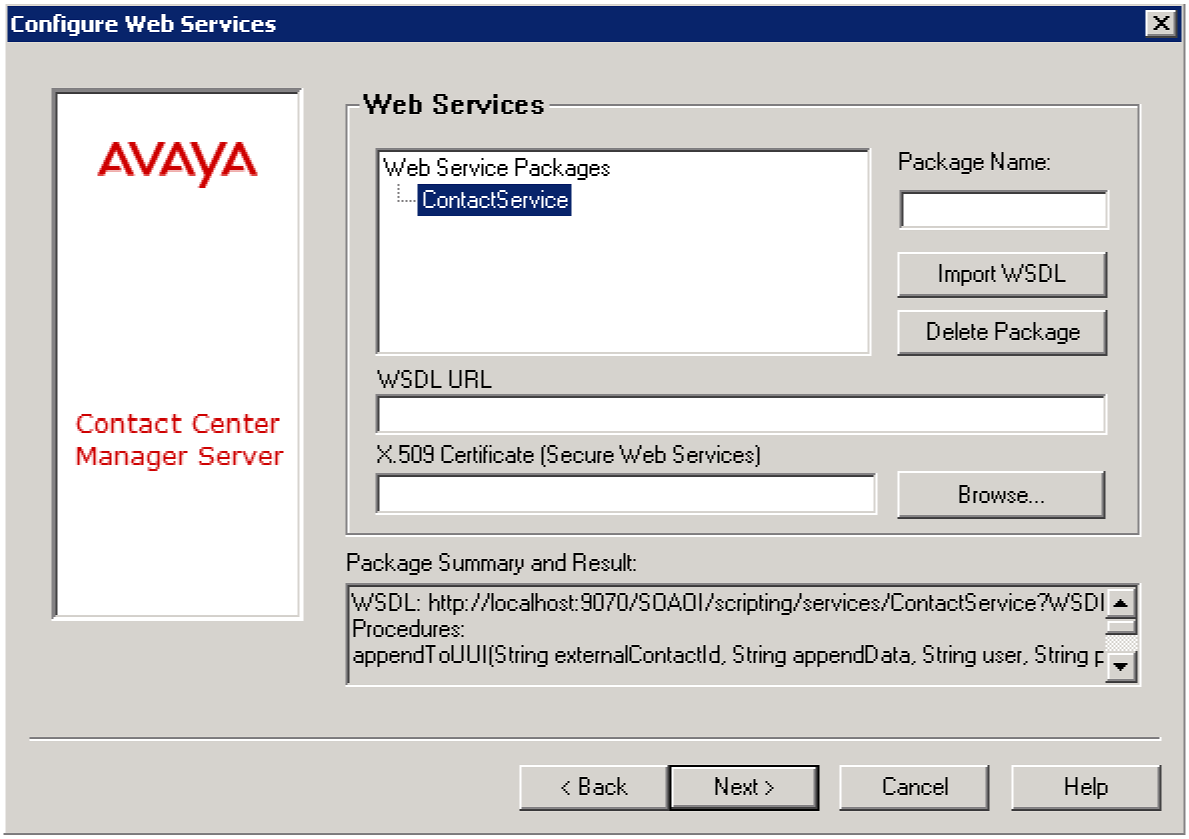
IVR Port: 0

Test Output and Result:

```
Register Provider - Successful.
De-Register Provider - Successful.
RESULT:   PASSED!
De-Initialising Connection - Successful.
```

DIW Form 3 Configure Web Services

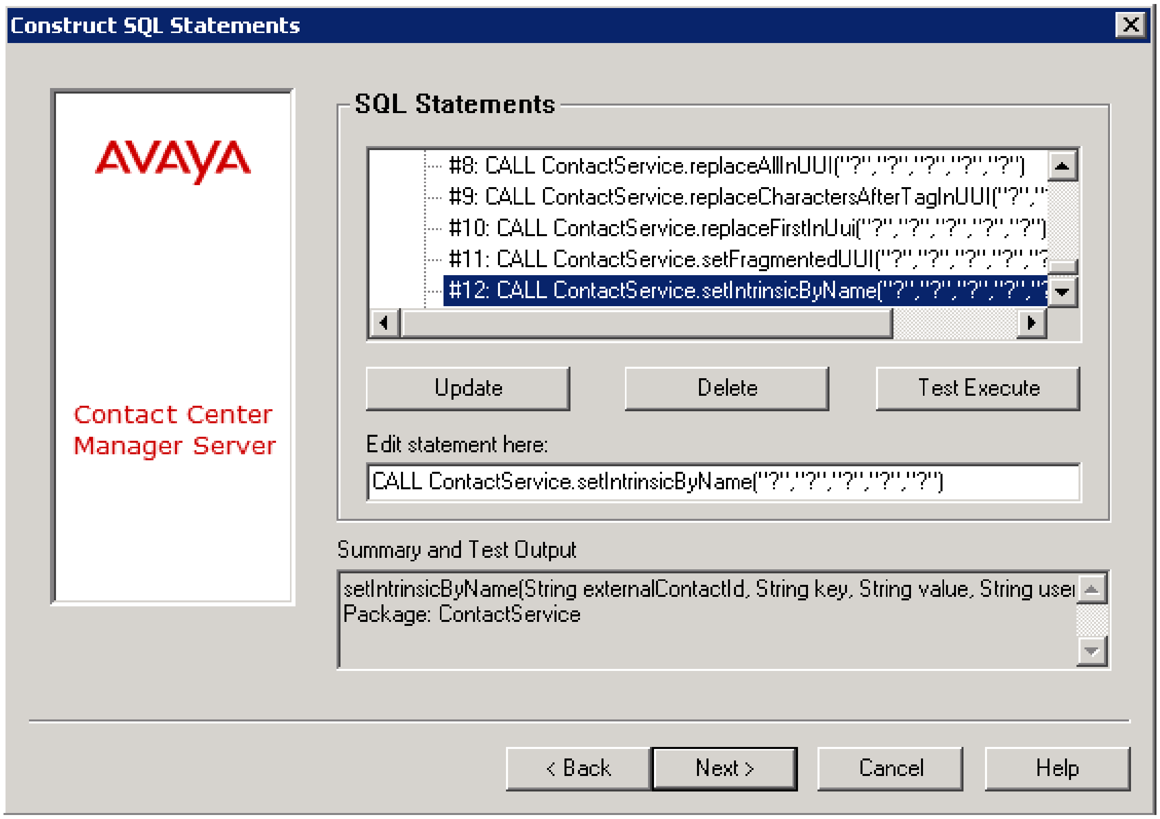
Click Next on the second form and on the third form enter the WSDL for the Contact Service web service, add a package name of your choice for example ContactService and click “Import WSDL”. If successful in the Package Summary and Result section click next.



```
WSDL: http://localhost:9070/SOAOI/scripting/services/ContactService?WSDL
Procedures:
appendToUUI(String externalContactId, String appendData, String user, String p
```

DIW Form 4 Construct SQL Statements

Scroll down on the Construct SQL Statements form to the WebServices section. Note the number of the setIntrinsicByName web method – this number will be used within the orchestration designer application hostblock in our example as this is the web method called to attach the intrinsic to the contact. This DIW form can be used to test web methods – it is possible to test the set intrinsics web method if a call is active on the system so the CmfProviderID is known and populate the externalContactId field of the web service.



```
#8: CALL ContactService.replaceAllInUUI("?","?","?","?","?")
#9: CALL ContactService.replaceCharactersAfterTagInUUI("?","
#10: CALL ContactService.replaceFirstInUui("?","?","?","?","?")
#11: CALL ContactService.setFragmentedUUI("?","?","?","?","?
#12: CALL ContactService.setIntrinsicByName("?","?","?","?","
```

Edit statement here: CALL ContactService.setIntrinsicByName("?","?","?","?","?")

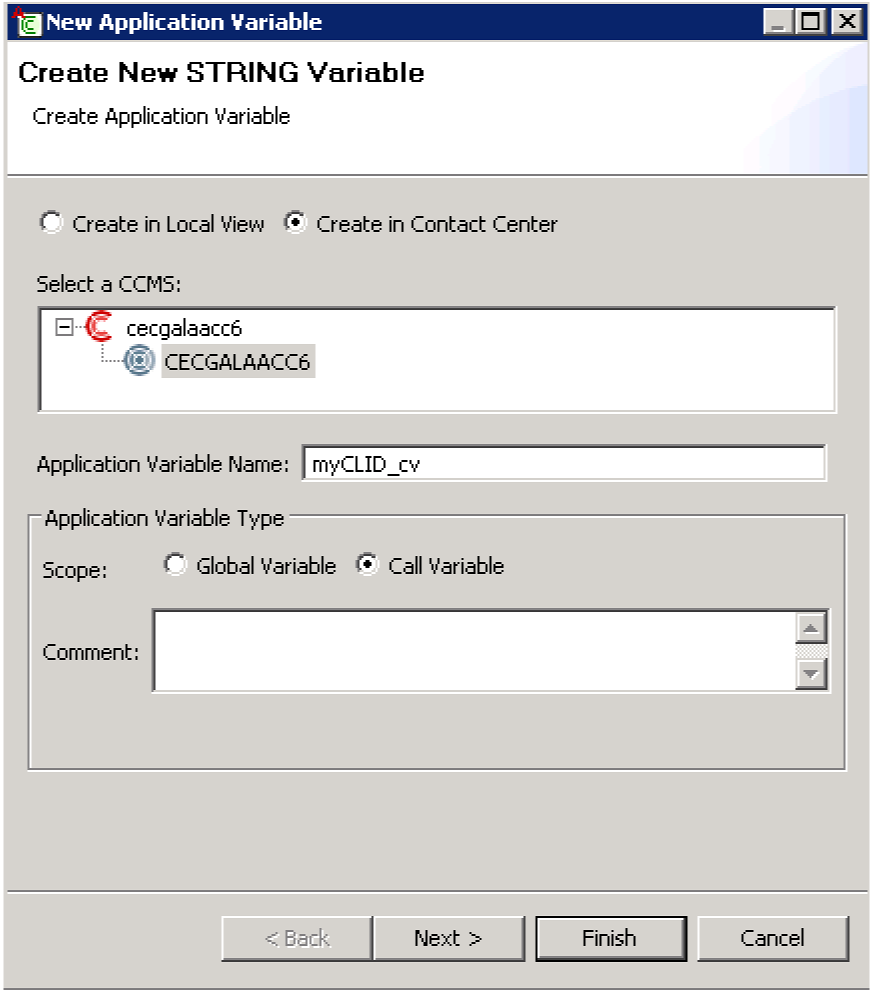
Summary and Test Output

```
setIntrinsicByName(String externalContactId, String key, String value, String use
Package: ContactService
```

Click next and click finish to save the settings in the DIW.

Step 4: Create required variables in OD

In Orchestration Designer expand Application variables, right click on string and add a new variable. Populate the first form as below choosing the call variable type and adding the name.



Create New STRING Variable

Create Application Variable

○ Create in Local View ● Create in Contact Center

Select a CCMS:

Application Variable Name: myCLID_cv

Application Variable Type

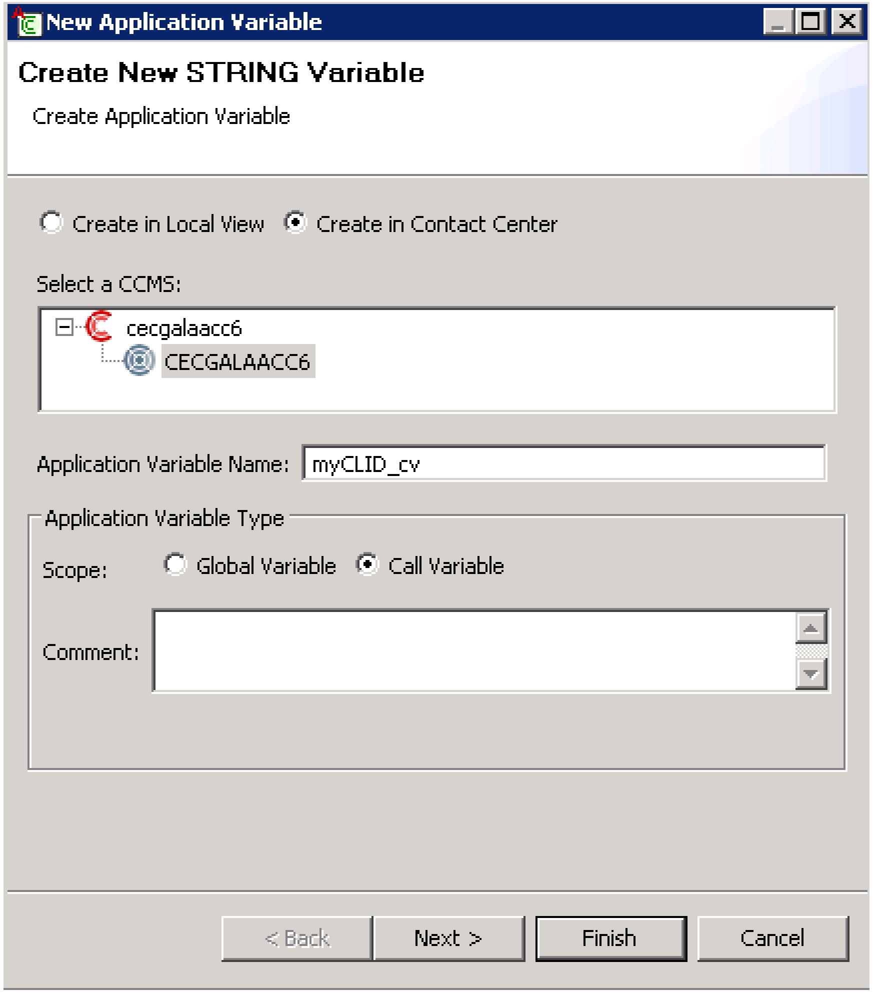
Scope: ○ Global Variable ● Call Variable

Comment:

Click next and click finish to create the variable.

Do for all variables required.

In Orchestration Designer expand Application variables, right click on string and add a new variable. Populate the first form as below choosing the call variable type and adding the name.



Click next and give a default value of 0. Click finish to create the variable.

Do for all variables required.

Variables used in this example to combine intrinsic values:

Parameter Name	**Description**	**Type**	**Variable required creation/out of the box**
myCLID_cv	call variable used to store the contact CLID value	String	Requires creation
myCDN_cv	call variable used to store the contact CDN value	String	Requires creation
intrinsicKey_gv	The key identifying the new intrinsic value, must match the intrinsic added in the CCMM admin tool which is used by AACC to complete the URL popped in AAAD. As this is a global variable it is	String	Requires creation

	populated on creation of the variable and not during the execution of the script like call variables.		
mySearchURL_cv	The intrinsic value attached to the contact data, contains a combination of values used to complete the URL popped in AAAD.	String	Requires creation
WSResult_cv	The response from the web service call. The ContactService Web Services return 0 on success.	String	Requires creation

Variables used by and populated for this example that are included in an installation of AACC:

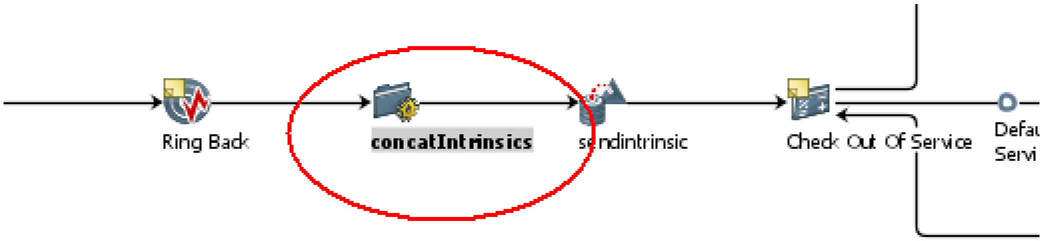
Parameter Name	**Description**	**Type**	**Variable required creation/out of the box**
HDX_AppId	Identifies the database integration wizard as a HDX application accessing AACC. This is a number set on the first form of the DIW and is the first parameter passed from the host block. In this example the variable is set to 121.	Integer	Out of the box
HDX_QueryNum_cv	A number assigned by the DIW application to identify the web method call.	String	Out of the box
externalCallId_cv	The cmfProviderID value on the contact retrieved within orchestration designer script from the contact data.	String	Out of the box
Username	Open web services SOA username OpenWsUser.	String	Out of the box
Password	Password of the OpenWsUser as set in server config.	String	Out of the box
HDXResp_cv	The response from the DIW HDX call.	String	Out of the box

Step 5: Combine the relevant intrinsic values into one call variable (mySearchURL_cv) in Orchestration designer

Intrinsics on contacts must be used in conjunction with the CONTACT DATA command and be assigned to a call variable before use.

String call variables you will need: myCLID_cv , myCDN_cv, and mySearch_URL_cv.

Add a custom block to the application that will process the voice contact and double click to open the block.



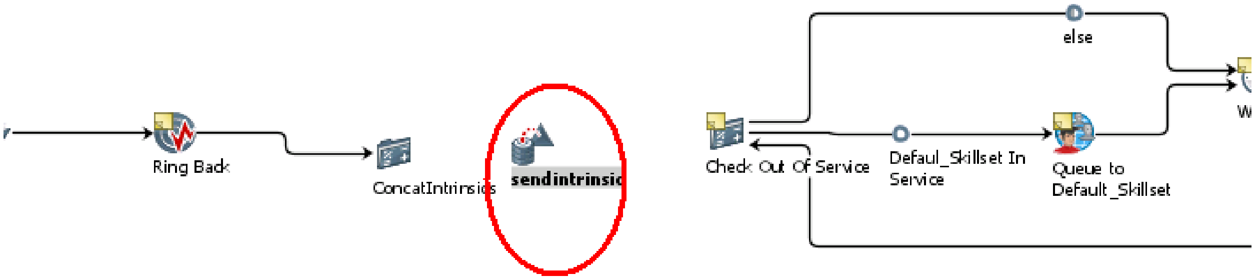
In the TFE Editor add code to collect the CLID and CDN values and APPEND all together into the mySearchURL_cv. This code has “if” checks because if an intrinsic does not exist the call variable is populated with “BAD or INVALID KEY NAME”. In this sample code just a log message is output when this occurs.

```
ASSIGN CONTACT DATA "AD_CLID" TO myCLID_cv
IF myCLID_cv = "BAD or INVALID KEY NAME" THEN
      LOG "No CLID populated"
ELSE
      ASSIGN "CLID%3D" TO mySearchURL_cv
      APPEND myCLID_cv TO mySearchURL_cv
END IF
ASSIGN CONTACT DATA "AD_CDN" TO myCDN_cv
IF (myCDN_cv = "BAD or INVALID KEY NAME") THEN
      LOG "No CDN populated"
ELSE
      APPEND "+CDN%3D" TO mySearchURL_cv
      APPEND myCDN_cv TO mySearchURL_cv
END IF
```

Step 6: Call the Contact Service web service from an Orchestration Designer host block to add the intrinsic key and value to the contact

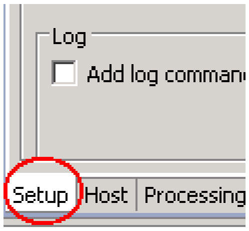
Now that the call variable mySearchURL_cv has been prepared we can send it to the web service for attachment in the contact intrinsics of the voice call.

Add a host block to the application and double click to open.



Host block setup tab

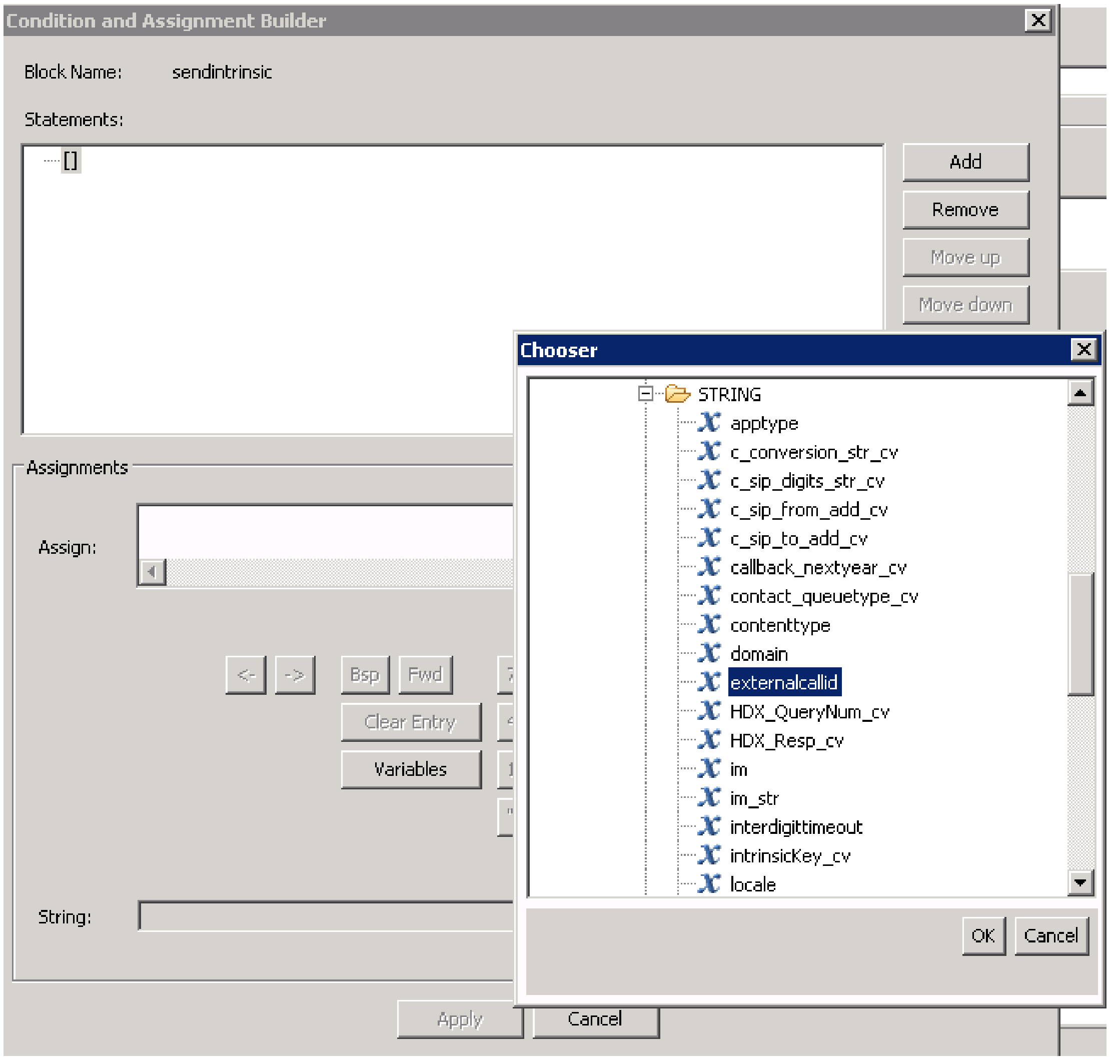
On the host block setup tab set up the required variable values for the call to the addintrinsic web method:



Click "Edit" on the host block set up tab and the Condition and Assignment Builder opens

Click "Add on the Condition and Assignment Builder and chose "Add Assignment".

Click "Variables" and chose the first variable to be assigned externalcallId and click "ok".



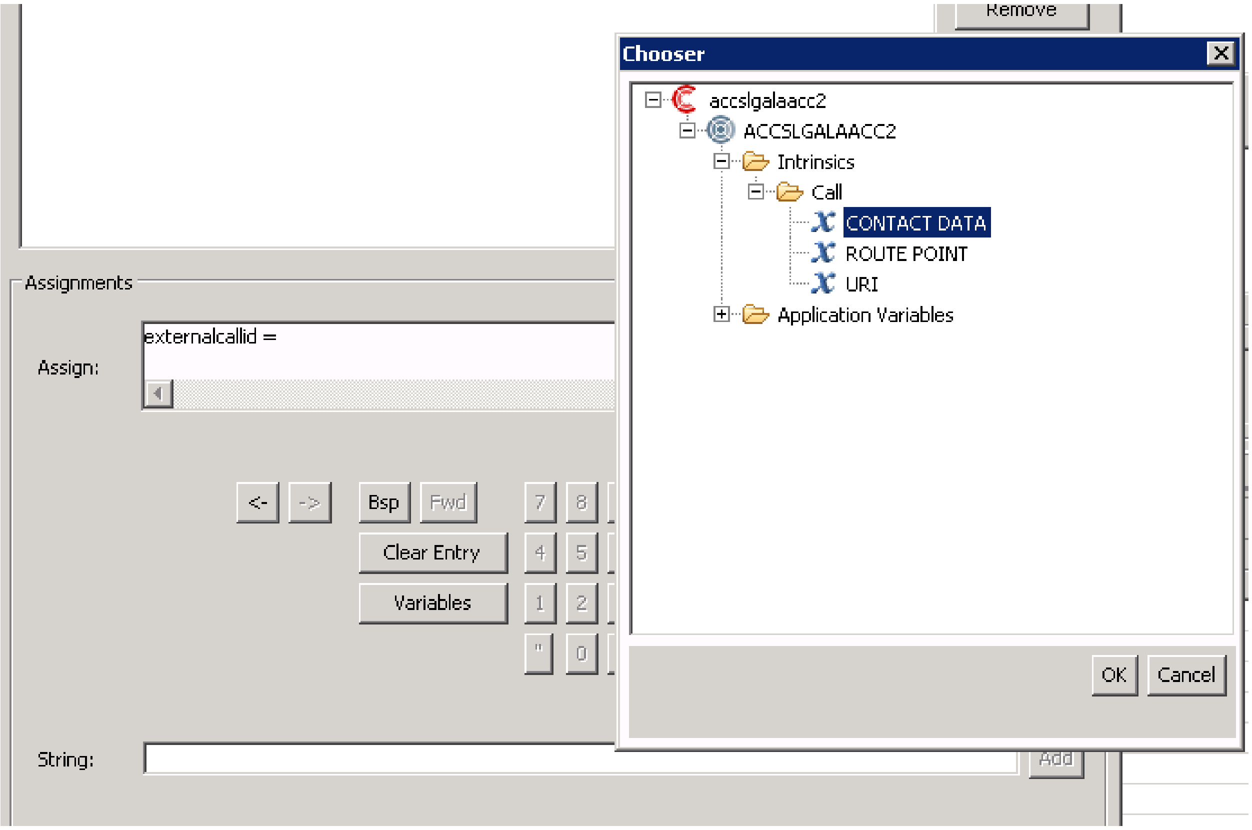
Externalcallid is added to the Assign textbox.

Select “Assignment” from the drop down list that appears under the Variables button and an “=” is added to the Assign textbox.

This variable will be assigned the value of ProviderContactID from the contacts intrinsics and to retrieve this value we use the CONTACT DATA keyword.

Click the “Variables” button, expand intrinsics -> Call and chose the CONTACT DATA intrinsic option. Click “ok” and CONTACT DATA is added to the assign textbox.

Tip: Do not type CONTACT DATA in directly into the assignment textbox it is important to add it from the chooser form. The string textbox should only be used for literals.

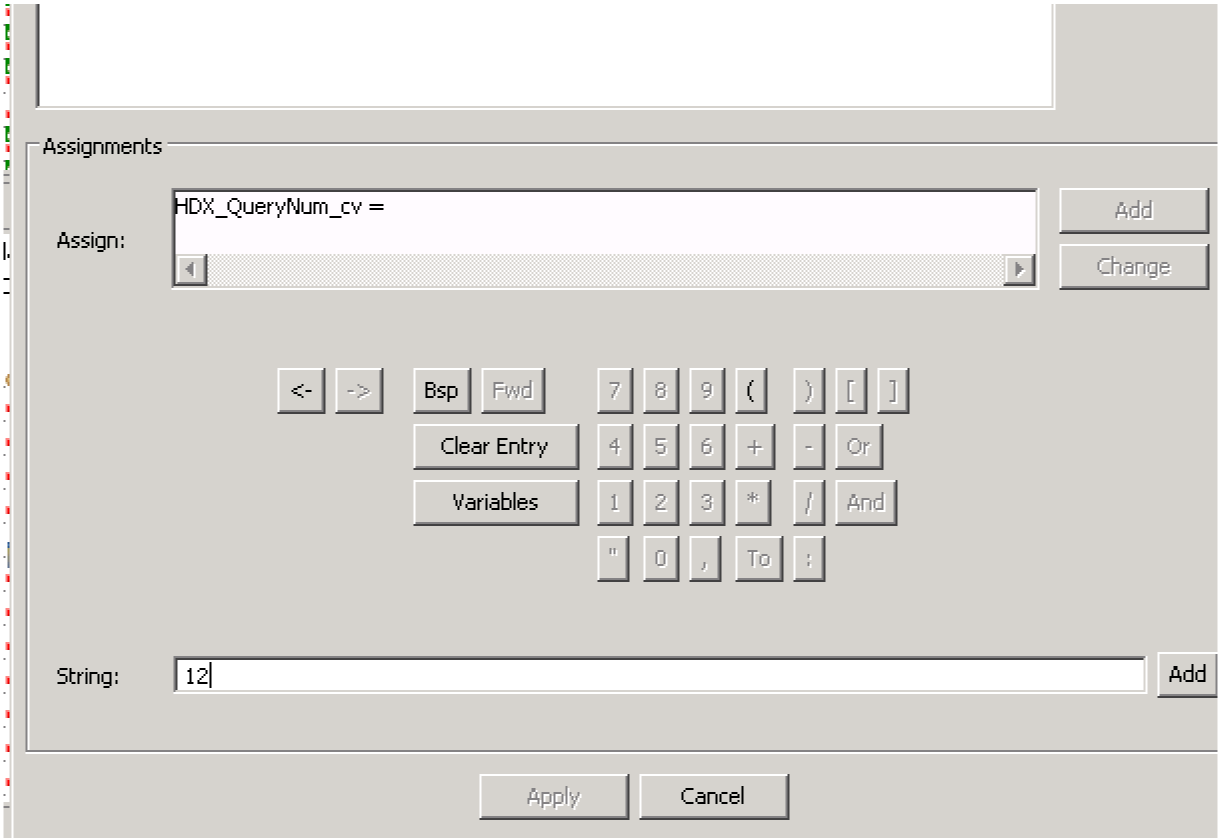


Assign: externalcallid =

In the String textbox type “ProviderContactID” and click “Add” on the String.

Click “Add” on the Assign.

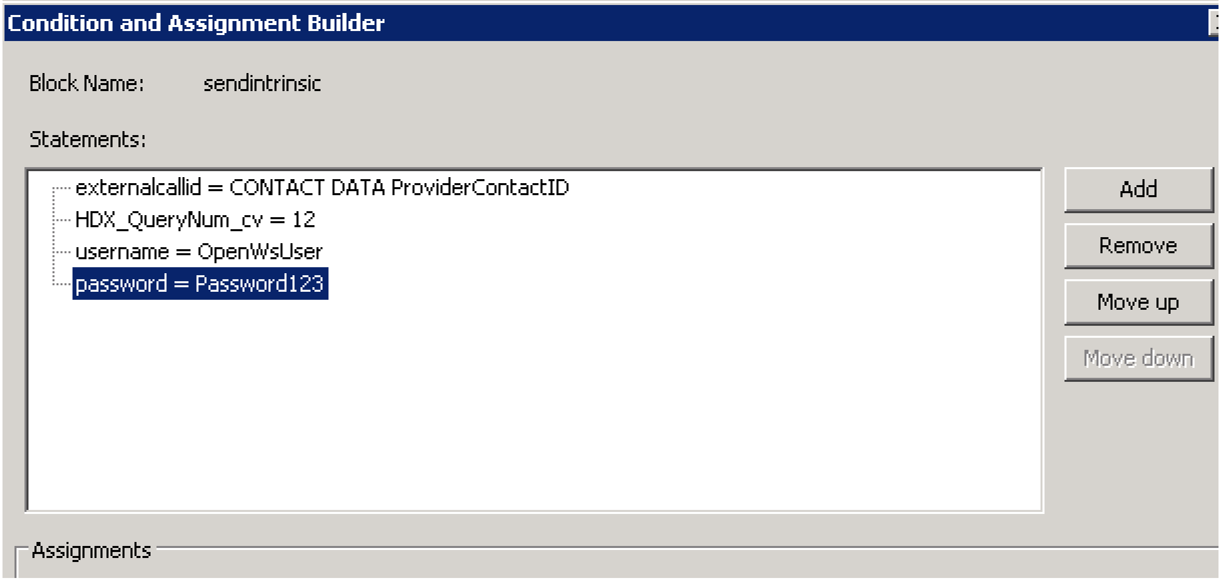
Add the next four variable assignments in the same way, assigning 12 to the HDX_Query_Num, the number that identifies for your environment the web method SetIntrinsicByName .



Assign: HDX_QueryNum_cv =

String: 12

Finally assign the web service username “OpenWSUser”and password “Password123” to the variables username and password.

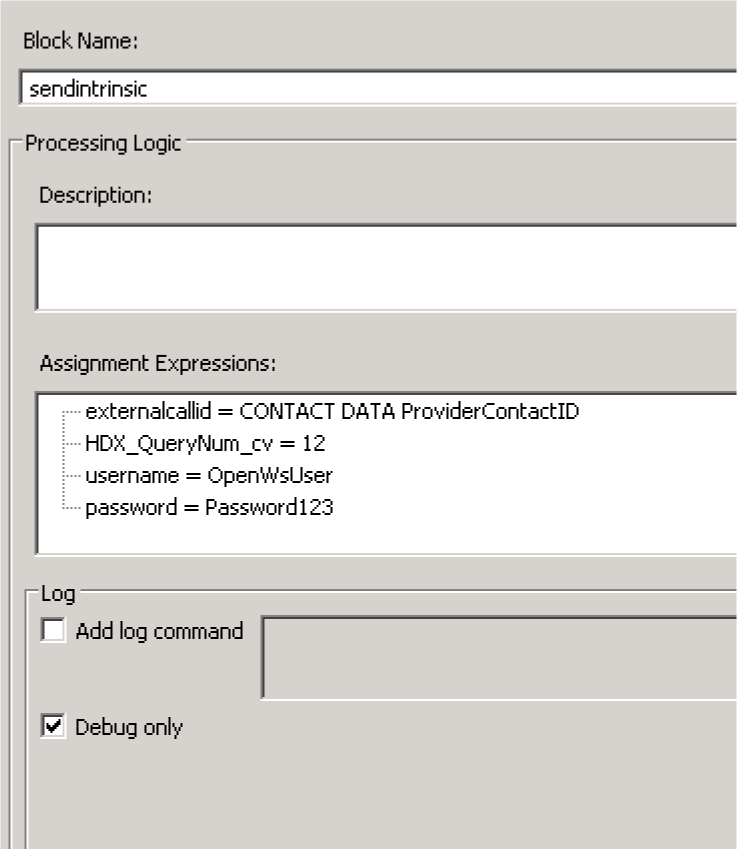


Block Name: sendintrinsic

Statements:

externalcallid = CONTACT DATA ProviderContactID
HDX_QueryNum_cv = 12
username = OpenWsUser
password = Password123

Click Apply and the Assignments are placed on the host block setup tab.



sendintrinsic

```
externalcallid = CONTACT DATA ProviderContactID
HDX_QueryNum_cv = 12
username = OpenWsUser
password = Password123
```

☐ Add log command

☑ Debug only

Host block host tab

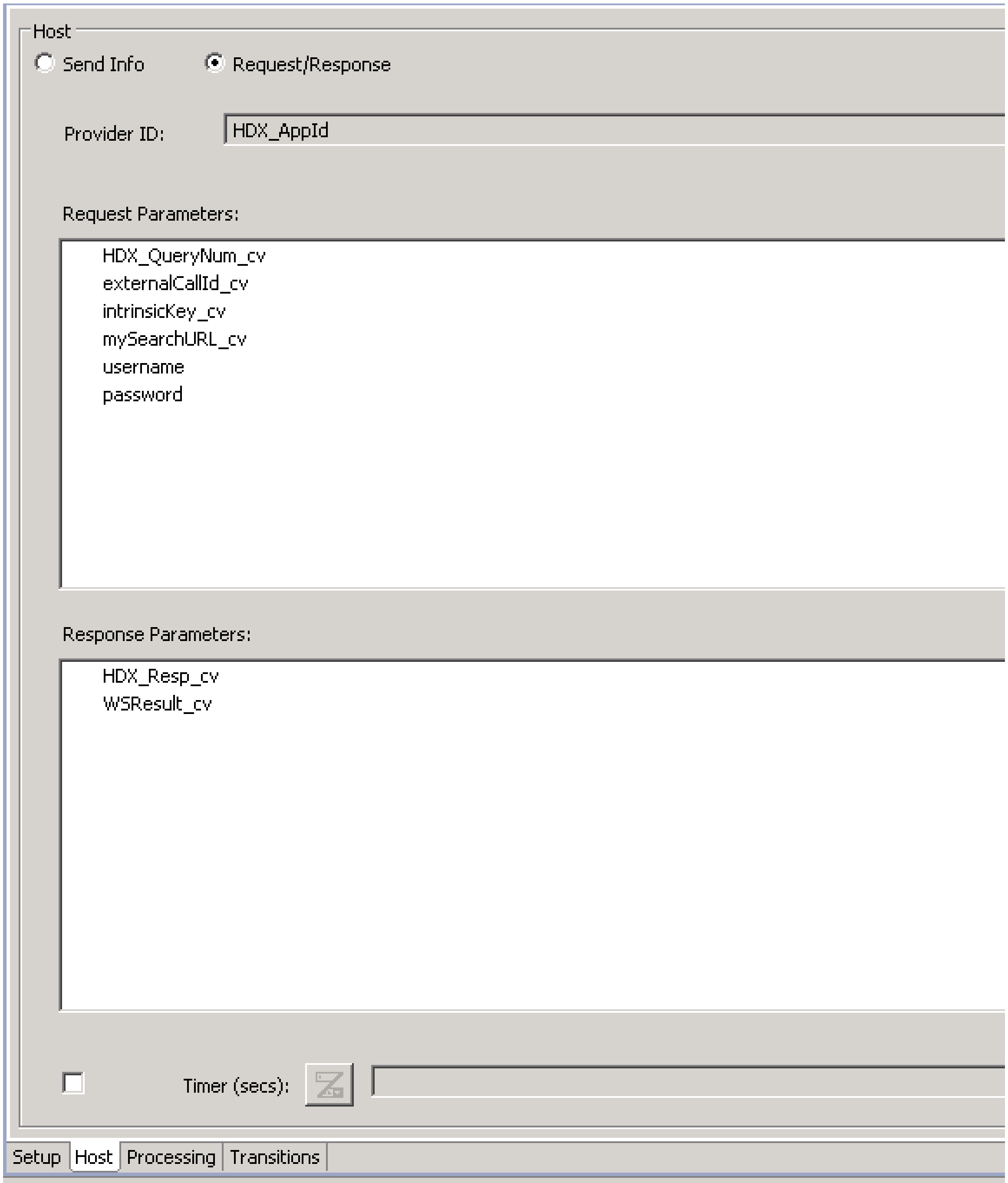
On the host tab of the host block add the parameters to call the web method.

Choose the Request/Response option.

Add the request parameters as populated in the following screenshot. The HDX_AppId is populated in the Provider ID field, request and response variables are grouped together in separate areas.

Use the “Add” button and the chooser form to select the variables.

The order of the variables is important, if they are added in the wrong order you can use the “Move Down” and “Move Up” buttons to correct the order.



Host

○ Send Info ● Request/Response

Provider ID: HDX_AppId

Request Parameters:

HDX_QueryNum_cv
externalCallId_cv
intrinsicKey_cv
mySearchURL_cv
username
password

Response Parameters:

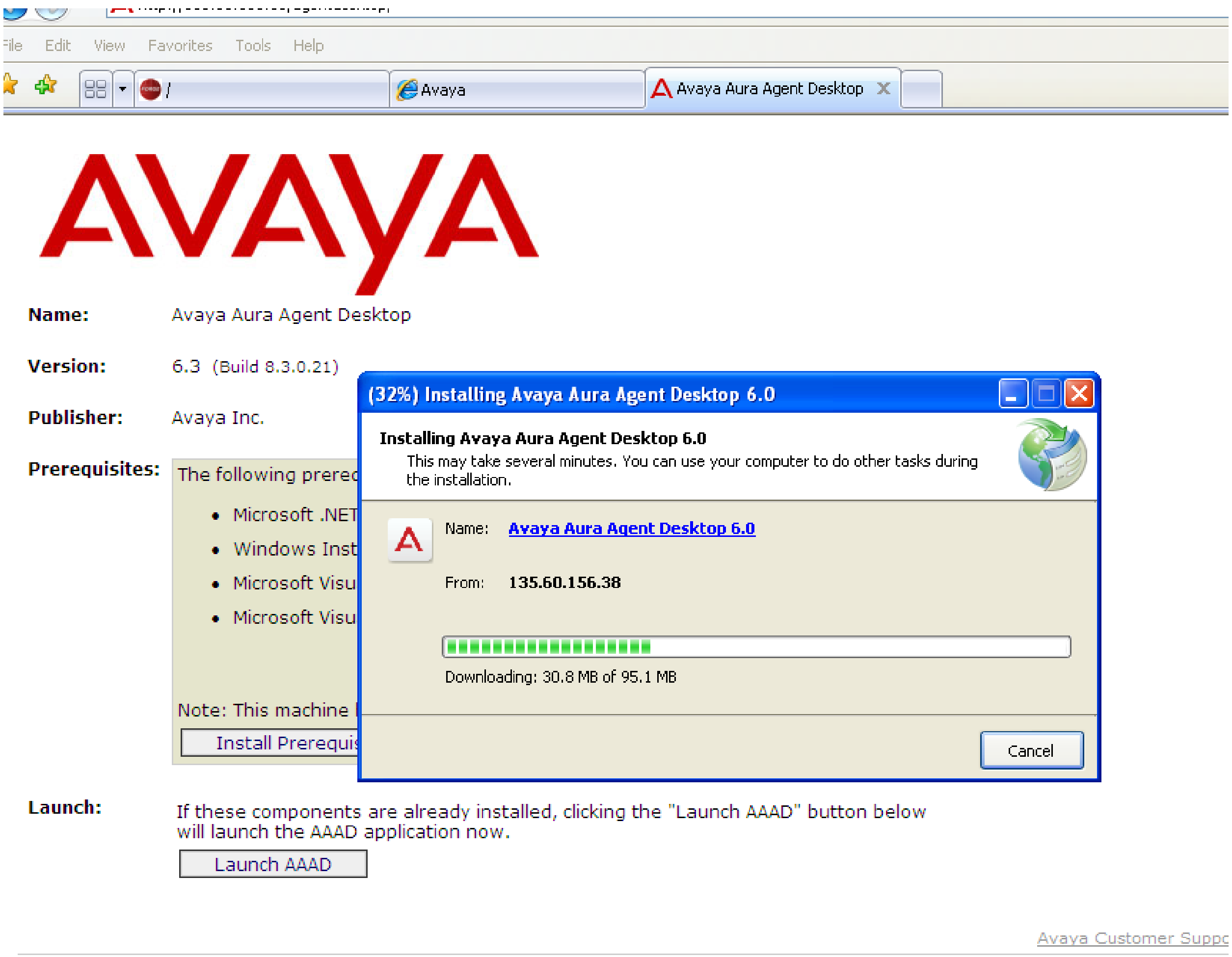
HDX_Resp_cv
WSResult_cv

☐ Timer (secs):

Save the application which validates it in orchestration designer.

Step 7: Download the AAAD and test screen pop

Download the AAAD from the AACC server which will have the latest CCMM admin settings:



Name: Avaya Aura Agent Desktop

Version: 6.3 (Build 8.3.0.21)

Publisher: Avaya Inc.

Prerequisites: The following prered

- Microsoft .NET
- Windows Inst
- Microsoft Visu
- Microsoft Visu

Note: This machine

(32%) Installing Avaya Aura Agent Desktop 6.0

Installing Avaya Aura Agent Desktop 6.0

This may take several minutes. You can use your computer to do other tasks during the installation.

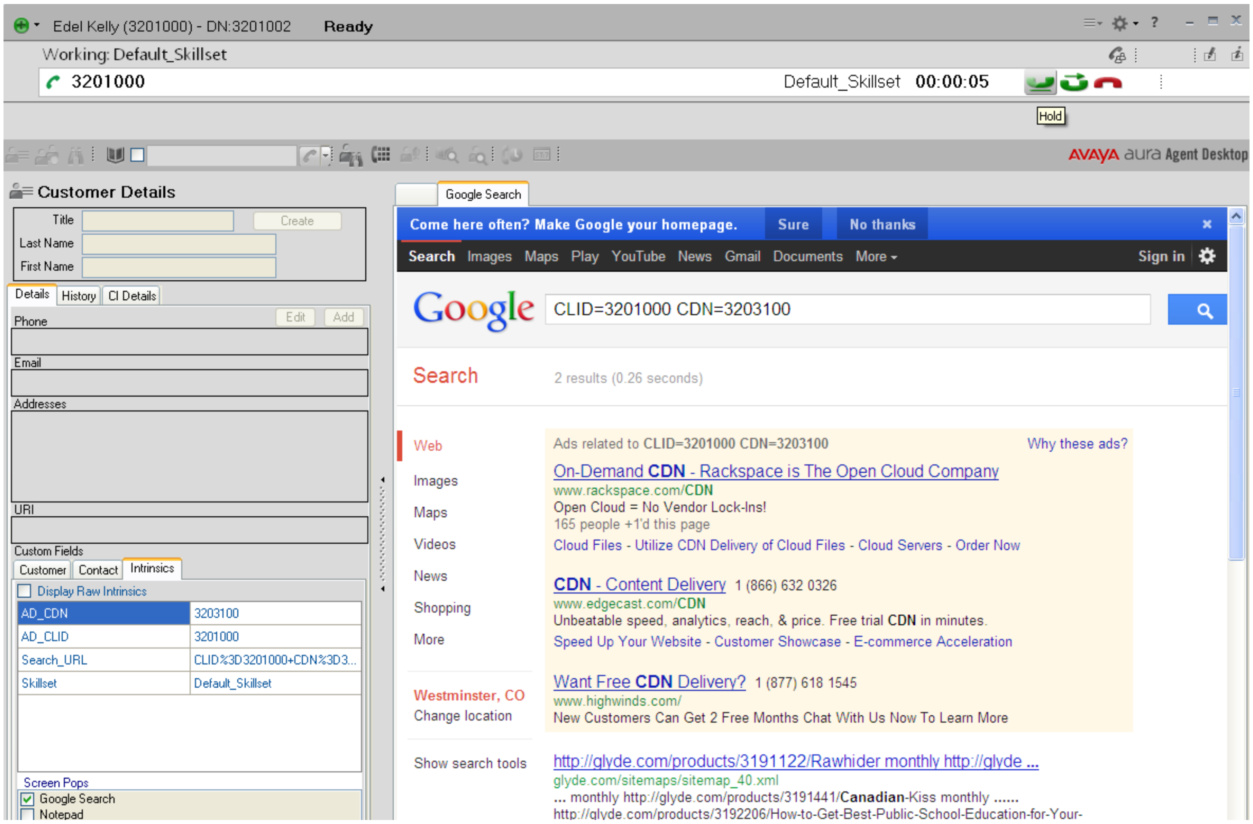
Name: Avaya Aura Agent Desktop 6.0

From: 135.60.156.38

Downloading: 30.8 MB of 95.1 MB

Launch: If these components are already installed, clicking the "Launch AAAD" button below will launch the AAAD application now.

Launch the AAAD and log in the agent. Route a skillset call through the updated orchestration designer script to the agent.



AD_CDN	3203100
AD_CLID	3201000
Search_URL	CLID%3D3201000+CDN%3D3...
Skillset	Default_Skillset

CLID=3201000 CDN=3203100

2 results (0.26 seconds)

Ads related to CLID=3201000 CDN=3203100

On-Demand CDN - Rackspace is The Open Cloud Company
www.rackspace.com/CDN
Open Cloud = No Vendor Lock-Ins!
165 people +1'd this page
Cloud Files - Utilize CDN Delivery of Cloud Files - Cloud Servers - Order Now

CDN - Content Delivery 1 (866) 632 0326
www.edgecast.com/CDN
Unbeatable speed, analytics, reach, & price. Free trial CDN in minutes.
Speed Up Your Website - Customer Showcase - E-commerce Acceleration

Want Free CDN Delivery? 1 (877) 618 1545
www.highwinds.com/
New Customers Can Get 2 Free Months Chat With Us Now To Learn More

http://glyde.com/products/3191122/Rawhider monthly http://glyde ...
glyde.com/sitemaps/sitemap_40.xml
... monthly http://glyde.com/products/3191441/Canadian-Kiss monthly ......
http://glyde.com/products/3192206/How-to-Get-Best-Public-School-Education-for-Your-

The AAAD will screenpop the URL as the image above displays with the two parameters.

Limitations and Assumptions

This solution has its limitations. The most important is that the orchestration designer call variables cannot be longer than 80 characters, this impacts the total number of characters of an intrinsic we can add to a voice contact and use as a parameter to the URL.

The URL format which will now contain multiple parameters should be agreed with the developers responsible for the application or page being popped. It will be the responsibility of this application or page external to AACC to parse the values and use them as required.

If during the concatenation of the parameters one is not populated on the contact intrinsics then in this application note example it will not be added however a rule of how this should be handled either not added or added with a default value should be dictated again by the developer or owner of the third party external application or page being popped.

The SOA developer kit license must be included in the AACC order to have access to the ContactService web service exposed by AACC to update the contact intrinsics.

Conclusion

If all steps within this application note were completed a google search within or outside the AAAD on the arrival of a skillset call is popped with multiple parameters in the search field.

The page or application popped with multiple parameters in this way must have the format of those multiple parameters agreed with the owner of the external application. The parsing of the parameters must be carried out by the external app.

The orchestration designer enforces a limitation of 80 characters in total for call variables which will limit the length of the concatenated values to a total of 80 also.

Additional References

More information on AACC topics discussed in this application note can be found in the following documents available on http://support.avaya.com.

Document Name	Description
NN44400-510 Avaya Aura® Configuration—Orchestration Designer Application Development	How to build applications/scripts to route skillset calls within AACC using orchestration designer. How to integrate orchestration designer with the ContactService web service.
NN44400-610 Avaya Aura® Contact Center Server Administration	How to configure screenpops within the AAAD using the CCMM administration tool.