Developer Guide for Proactive Outreach Manager

Release 3.0
Issue 1
February 2014

Notice

While reasonable efforts have been made to ensure that the information in this document is complete and accurate at the time of printing, Avaya assumes no liability for any errors. Avaya reserves the right to make changes and corrections to the information in this document without the obligation to notify any person or organization of such changes.

Documentation disclaimer

"Documentation" means information published by Avaya in varying mediums which may include product information, operating instructions and performance specifications that Avaya may generally make available to users of its products and Hosted Services. Documentation does not include marketing materials. Avaya shall not be responsible for any modifications, additions, or deletions to the original published version of documentation unless such modifications, additions, or deletions were performed by Avaya. End User agrees to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation, to the extent made by End User.

Link disclaimer

Avaya is not responsible for the contents or reliability of any linked websites referenced within this site or documentation provided by Avaya. Avaya is not responsible for the accuracy of any information, statement or content provided on these sites and does not necessarily endorse the products, services, or information described or offered within them. Avaya does not guarantee that these links will work all the time and has no control over the availability of the linked pages.

Warranty

Avaya provides a limited warranty on Avaya hardware and software. Refer to your sales agreement to establish the terms of the limited warranty. In addition, Avaya's standard warranty language, as well as information regarding support for this product while under warranty is available to Avaya customers and other parties through the Avaya Support website: http://support.avaya.com or such successor site as designated by Avaya. Please note that if you acquired the product(s) from an authorized Avaya Channel Partner outside of the United States and Canada, the warranty is provided to you by said Avaya Channel Partner and not by Avaya.

Licenses

THE SOFTWARE LICENSE TERMS AVAILABLE ON THE AVAYA WEBSITE, HTTP://SUPPORT.AVAYA.COM/LICENSEINFO OR SUCH SUCCESSOR SITE AS DESIGNATED BY AVAYA, ARE APPLICABLE TO ANYONE WHO DOWNLOADS, USES AND/OR INSTALLS AVAYA SOFTWARE, PURCHASED FROM AVAYA INC., ANY AVAYA AFFILIATE, OR AN AVAYA CHANNEL PARTNER (AS APPLICABLE) UNDER A COMMERCIAL AGREEMENT WITH AVAYA OR AN AVAYA CHANNEL PARTNER. UNLESS OTHERWISE AGREED TO BY AVAYA IN WRITING, AVAYA DOES NOT EXTEND THIS LICENSE IF THE SOFTWARE WAS OBTAINED FROM ANYONE OTHER THAN AVAYA, AN AVAYA AFFILIATE OR AN AVAYA CHANNEL PARTNER; AVAYA RESERVES THE RIGHT TO TAKE LEGAL ACTION AGAINST YOU AND ANYONE ELSE USING OR SELLING THE SOFTWARE WITHOUT A LICENSE. BY INSTALLING, DOWNLOADING OR USING THE SOFTWARE, OR AUTHORIZING OTHERS TO DO SO, YOU, ON BEHALF OF YOURSELF AND THE ENTITY FOR WHOM YOU ARE INSTALLING, DOWNLOADING OR USING THE SOFTWARE (HEREINAFTER REFERRED TO INTERCHANGEABLY AS "YOU" AND "END USER"), AGREE TO THESE TERMS AND CONDITIONS AND CREATE A BINDING CONTRACT BETWEEN YOU AND AVAYA INC. OR THE APPLICABLE AVAYA AFFILIATE ("AVAYA").

License types

Designated System(s) License (DS). End User may install and use each copy or an Instance of the Software only on a number of Designated Processors up to the number indicated in the order. Avaya may require the Designated Processor(s) to be identified in the order by type, serial number, feature key, Instance, location or other specific designation, or to be provided by End User to Avaya through electronic means established by Avaya specifically for this purpose.

Concurrent User License (CU). End User may install and use the Software on multiple Designated Processors or one or more Servers, so long as only the licensed number of Units are accessing and using the Software at any given time. A "Unit" means the unit on which Avaya, at its sole discretion, bases the pricing of its licenses and can be, without limitation, an agent, port or user, an e-mail or voice mail account in the name of a person or corporate function (e.g., webmaster or helpdesk), or a directory entry in the administrative database utilized by the Software that permits one user to interface with the Software. Units may be linked to a specific, identified Server or an Instance of the Software.

Database License (DL). End User may install and use each copy or an Instance of the Software on one Server or on multiple Servers provided that each of the Servers on which the Software is installed communicates with no more than an Instance of the same database.

CPU License (CP). End User may install and use each copy or Instance of the Software on a number of Servers up to the number indicated in the order provided that the performance capacity of the Server(s) does not exceed the performance capacity specified for the Software. End User may not re-install or operate the Software on Server(s) with a larger performance capacity without Avaya's prior consent and payment of an upgrade fee.

Named User License (NU). You may: (i) install and use the Software on a single Designated Processor or Server per authorized Named User (defined below); or (ii) install and use the Software on a Server so long as only authorized Named Users access and use the Software. "Named User", means a user or device that has been expressly authorized by Avaya to access and use the Software. At Avaya's sole discretion, a "Named User" may be, without limitation, designated by name, corporate function (e.g., webmaster or helpdesk), an e-mail or voice mail account in the name of a person or corporate function, or a directory entry in the administrative database utilized by the Software that permits one user to interface with the Software.

Shrinkwrap License (SR). You may install and use the Software in accordance with the terms and conditions of the applicable license agreements, such as "shrinkwrap" or "clickthrough" license accompanying or applicable to the Software ("Shrinkwrap License").

Copyright

Except where expressly stated otherwise, no use should be made of materials on this site, the Documentation, Software, Hosted Service, or hardware provided by Avaya. All content on this site, the documentation, Hosted Service, and the Product provided by Avaya including the selection, arrangement and design of the content is owned either by Avaya or its licensors and is protected by copyright and other intellectual property laws including the sui generis rights relating to the protection of databases. You may not modify, copy, reproduce, republish, upload, post, transmit or distribute in any way any content, in whole or in part, including any code and software unless expressly authorized by Avaya. Unauthorized reproduction, transmission, dissemination, storage, and or use without the express written consent of Avaya can be a criminal, as well as a civil offense under the applicable law.

Third Party Components

"Third Party Components" mean certain software programs or portions thereof included in the Software or Hosted Service may contain software (including open source software) distributed under third party agreements ("Third Party Components"), which contain terms regarding the rights to use certain portions of the Software ("Third Party

Terms”). As required, information regarding distributed Linux OS source code (for those Products that have distributed Linux OS source code) and identifying the copyright holders of the Third Party Components and the Third Party Terms that apply is available in the Documentation or on Avaya’s website at: http://support.avaya.com/Copyright or such successor site as designated by Avaya. You agree to the Third Party Terms for any such Third Party Components

Preventing Toll Fraud

“Toll Fraud” is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or is not working on your company's behalf). Be aware that there can be a risk of Toll Fraud associated with your system and that, if Toll Fraud occurs, it can result in substantial additional charges for your telecommunications services.

Avaya Toll Fraud intervention

If you suspect that you are being victimized by Toll Fraud and you need technical assistance or support, call Technical Service Center Toll Fraud Intervention Hotline at +1-800-643-2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Support website: http://support.avaya.com or such successor site as designated by Avaya. Suspected security vulnerabilities with Avaya products should be reported to Avaya by sending mail to: securityalerts@avaya.com.

Trademarks

The trademarks, logos and service marks (“Marks”) displayed in this site, the Documentation, Hosted Service(s), and Product(s) provided by Avaya are the registered or unregistered Marks of Avaya, its affiliates, or other third parties. Users are not permitted to use such Marks without prior written consent from Avaya or such third party which may own the Mark. Nothing contained in this site, the Documentation, Hosted Service(s) and Product(s) should be construed as granting, by implication, estoppel, or otherwise, any license or right in and to the Marks without the express written permission of Avaya or the applicable third party.

Avaya is a registered trademark of Avaya Inc.

All non-Avaya trademarks are the property of their respective owners. Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.

Downloading Documentation

For the most current versions of Documentation, see the Avaya Support website: http://support.avaya.com, or such successor site as designated by Avaya.

Contact Avaya Support

See the Avaya Support website: http://support.avaya.com for Product or Hosted Service notices and articles, or to report a problem with your Avaya Product or Hosted Service. For a list of support telephone numbers and contact addresses, go to the Avaya Support website: http://support.avaya.com (or such successor site as designated by Avaya), scroll to the bottom of the page, and select Contact Avaya Support.

Contents

Chapter 1: Introduction

Purpose

This document describes the methods and properties used for the Web interface of Avaya Proactive Outreach Manager, and various custom classes and application files.

Intended audience

This document is intended for users, system administrators, and implementation engineers who are responsible for making changes to Avaya Proactive Outreach Manager through the Web interface, and various custom classes and application files.

Related resources

Documentation

For information on feature administration, interactions, considerations, and security, see the following POM documents available on the Avaya Support site at http://www.avaya.com/support:

Title	Description	Audience	Document location
Proactive Outreach Manager Overview and Specification	Provides general information about the product overview and the integration with other products.	Users	The latest PDF is available on the Avaya Support site at Proactive Outreach Manager Overview and Specification.

Title	Description	Audience	Document location
Implementing Proactive Outreach Manager	Provides information about installing and configuring Proactive Outreach Manager.	Implementation engineers	The latest PDF is available on the Avaya Support site at Implementing Proactive Outreach Manager.
Upgrading Proactive Outreach Manager	Provides information about upgrading Proactive Outreach Manager.	Implementation engineers	The latest PDF is available on the Avaya Support site at Upgrading Proactive Outreach Manager.
Using Proactive Outreach Manager	Provides general information about field descriptions and procedures for using Proactive Outreach Manager.	Users	The latest PDF is available on the Avaya Support site at Using Proactive Outreach Manager.
Troubleshooting Proactive Outreach Manager	Provides general information about troubleshooting and resolving system problems, and detailed information about and procedures for finding and resolving specific problems.	System administrators Implementation engineers Users	The latest PDF is available on the Avaya Support site at Troubleshooting Proactive Outreach Manager.

Alternatively you can see the relevant Avaya Aura® Experience Portal 7.0 documentation from the Avaya Support site. You must install Avaya Aura® Experience Portal before you install POM. You will find references to Avaya Aura® Experience Portal documentation at various places in the POM documentation.

Training

The following courses are available on the Avaya Learning website at www.avaya-learning.com. After logging in to the website, enter the course code or the course title in the **Search** field and click **Go** to search for the course.

Course code	Course title
4C000074W	Avaya Proactive Outreach Manager (POM) Administration and Configuration

Course code	Course title
5C00092V	Avaya Aura® Experience Portal, Avaya Aura® Orchestration Designer, Avaya Proactive Outreach Manager Installation, Maintenance and Troubleshooting Essentials
5C00090V	Avaya Aura® Experience Portal, Avaya Aura® Orchestration Designer, Avaya Proactive Outreach Manager Maintenance and Troubleshooting
5C00092I	Avaya Aura® Experience Portal, Avaya Aura® Orchestration Designer, Avaya Proactive Outreach Manager Installation, Maintenance and Troubleshooting Essentials
5C00090I	Avaya Aura® Experience Portal, Avaya Aura® Orchestration Designer, Avaya Proactive Outreach Manager Maintenance and Troubleshooting
5C00092V	Avaya Aura® Experience Portal, Avaya Aura® Orchestration Designer, Avaya Proactive Outreach Manager Installation, Maintenance and Troubleshooting Essentials
3C00070O	Designing Avaya Proactive Outreach Manager

Avaya Mentor videos

Avaya Mentor videos are available to provide technical content on how to install, configure, and troubleshoot Avaya products.

Videos are available on the Avaya support site, listed under the video document type, and on the Avaya-run channel on YouTube.

To find videos on the Avaya support site, select the product name, and check the *videos* checkbox to see a list of available videos.

Note:

Videos are not available for all products.

To find the Avaya Mentor videos on YouTube, go to http://www.youtube.com/AvayaMentor and perform one of the following actions:

- Enter a key word or key words in the Search Channel to search for a specific product or topic.
- Scroll down Playlists, and click the name of a topic to see the available list of videos posted on the site.

Support

Visit the Avaya Support website at http://support.avaya.com for the most up-to-date documentation, product notices, and knowledge articles. You can also search for release notes, downloads, and resolutions to issues. Use the online service request system to create a service request. Chat with live agents to get answers to questions, or request an agent to connect you to a support team if an issue requires additional expertise.

Warranty

Avaya Inc. provides a 90-day limited warranty on Proactive Outreach Manager. Refer to your sales agreement or other applicable documentation to establish the terms of the limited warranty. In addition, Avaya's standard warranty language as well as details regarding support for Proactive Outreach Manager, while under warranty, is available on the support Web site at http://www.avaya.com/support.

Chapter 2: Pluggable Data Connector

Overview of Pluggable Data Connector

Pluggable Data Connector (PDC) of Proactive Outreach Manager (POM) is a Avaya Aura® Orchestration Designer 6.x plug-in. The PDC extends the capability of Avaya Aura® Orchestration Designer application to interface and integrate with POM using the core functionality of the PDC framework.

The PDC implements the functionality of seamless connectivity with POM for various types of data exchange and establishes a connection between the VXML application and the POM server.

The PDC also extends the functionality of the Data node palette of Graphical Call flow editor in Avaya Aura® Orchestration Designer. The PDC provides the nodes to customize actions for POM servers:

Implementing PDC

Installing PDC

Before you begin

- Install the Avaya Aura® Orchestration Designer 6.x application.
- Download the POM PDC installer jar from http://support.avaya.com.

Procedure

1. Start Eclipse.
2. From the menu options, select the **Help > Software Updates > Available software** tab.
3. Click **Add Site > Archive**.
4. From the list of available software, browse to the `.jar` file.
5. Click **Open** and select the `.jar` file.
6. Click **Install**.

   The system first displays the Progress Information screen and then displays the Summary Information screen.

   The system might display a Security Information screen. You must acknowledge the message to continue with the installation.
7. Click **Finish**.

   The system displays the Operation In Progress screen.
8. Click **Details** to view the details of the installation.

   The system displays the Software Update screen and a message to restart the server.
9. Click **Yes** to restart Eclipse for the changes to take effect.

Upgrading PDC

If you have an earlier version of PDC installed, upgrade the PDC to work with the new release of POM.

Procedure

1. From the Eclipse menu option, select **Help > Install New Software > Available software**.
2. Click **Add Site** and then click **Archive**.
3. On the local file system, select the `POM PDC installable.jar` file.
4. Click **Open**.

   The system adds the jar file to the list of available software.
5. From the Eclipse menu option, select **Help > About Eclipse SDK > Installation Details**.
6. Select the existing connector for Proactive Outreach Manager.
7. Click **Update**.

   The system displays the Operation In Progress screen.
8. Follow the installation progress wizard. If you do not encounter any errors, restart Eclipse.

   You must take a backup of all current projects and disable the PDC plug-in for each project.
9. Save the project and then enable the PDC plug-in to update the project variables and the jar files.

Configuring PDC

Procedure

1. Create a new Avaya Aura® Orchestration Designer Speech Project, or use an existing Speech Project.
2. Select the project name in the Speech Navigator tab.
3. Right-click the project name and go to **Properties**.
4. In the left pane, select **Orchestration Designer**.
5. Select the **Pluggable Connectors** tab.
6. Select **POM Connector** from the list.
7. Specify the configuration settings as:
   - (Optional) Primary Host: Specify the IP address or the host name of the primary POM server.
   - (Optional) Secondary Host: Specify the IP address or the host name of the secondary POM server.

- User: Specify the login name of the Avaya Aura® Experience Portal user. If you turn on multitenancy, the user obtains privileges to view, update, and add data to the user organization.
- Password: Specify the password of the Avaya Aura® Experience Portal user.

8. Click **OK**.

Using PDC

Before you begin

Ensure you have at least one data node in the Palette.

About this task

Follow the steps to add data nodes to the Palette.

Procedure

1. From the **Palette** tab, select **Data** node.
2. Drag and drop the data node in the call flow to add the node in the call flow.
3. Double-click the data node.
   In the **Palette** tab, you can view the list of POM specific nodes.
4. Drag and drop each node in the call flow.
5. Click the **Properties** tab and set the appropriate properties.

Project variables

When you enable the POM PDC, the PDC creates two complex project variables.

pomInfo

The system deletes the pomInfo complex variable when you disable the PDC. The following fields remain in the variable:

Variable name	Data type	User input required
CampaignName	String	Yes
CompletionCode	String	Yes
EMailID	String	Yes

Variable name	Data type	User input required
FirstName	String	Yes
LastName	String	Yes
PhoneNumber1	String	Yes
PhoneNumber2	String	Yes
PrimaryPomServer	String	Yes
SecondaryPomServer	String	Yes
SessionID	String	Yes
ContactGroupName	String	Yes
ContactID	String	No. The system populates this value.
JobID	String	No. The system populates this value.
VDN	String	Yes
AddressLine1	String	Yes
AddressLine2	String	Yes
AddressLine3	String	Yes
AddressLine4	String	Yes
AddressLine5	String	Yes
Country	String	Yes
Language	String	Yes
PhoneNumber1CountryCode	String	Yes
PhoneNumber1Timezone	String	Yes
PhoneNumber2CountryCode	String	Yes
PhoneNumber2TimeZone	String	Yes
ZipCode	String	Yes

pomDynamicAttributes

This complex variable holds the array of custom attributes associated with a contact. The fields in the variable are:

- AttributeName
- AttributeType
- AttributeValue

phoneinfo

This complex variable has the following fields:

Variable name	Data type	User input required
CountryCode	String	Yes
PhoneAttributeName	String	Yes
PhoneNumber	String	Yes
Timezone	String	Yes

PDC nodes

Update disposition

Use this option to update custom completion codes. Based on these codes, you can terminate and track the campaign execution. The POM Web service user must have access to campaigns and completion codes.

The following table lists the property names and the property values:

Property Name	Value
Completion Code Name Variable	Use a simple or complex variable. **Note:** Ensure that the custom completion codes are a part of the running campaign.
Completion Code Name Variable Field	Use the field from the complex variable.
Completion Code Name Constant	If you know the name of the custom completion code, you can enter the name as a constant. **Note:** You can set the constant value or use the name variable.

Add contact info

Use this option to add a new contact or update an existing contact.

The POM Web service user must have access to the contact lists and private attributes.

The following table lists the property names and the property values:

Property Name	Value
Adding a new contact:	
ContactID Variable	Create a complex variable or select the existing complex variable from the drop-down list. **Note:** You must set this mandatory attribute.
ContactID Variable Field	Use available fields for the complex variable.
Contact list Name Variable	Use a simple or complex variable. **Note:** You must set this mandatory attribute.
Contact list Name Variable Field	Use a simple or complex variable.
First Name Variable	Create a complex variable or select the existing complex variable from the drop-down list.
First Name Variable Field	Use available fields for the complex variable.
Last Name Variable	Create a complex variable or select the existing complex variable from the drop-down list.
Last Name Variable Field	Use available fields for the complex variable.
Phone # 1 Variable	Create a complex variable or select the existing complex variable from the drop-down list.
Phone # 1 Variable Field	Use available fields for the complex variable.
Phone # 1 Country Code Variable	Create a complex variable or select the existing complex variable from the drop-down list.
Phone # 1 Time Zone Variable	Create a complex variable or select the existing complex variable from the drop-down list.
Phone # 2 Variable	Create a complex variable or select the existing complex variable from the drop-down list.
Phone # 2 Variable Field	Use available fields for the complex variable.

Property Name	Value
Phone # 2 Country Code Variable	Create a complex variable or select the existing complex variable from the drop-down list.
Phone # 2 Time Zone Variable	Create a complex variable or select the existing complex variable from the drop-down list.
Email ID Variable	Create a complex variable or select the existing complex variable from the drop-down list.
Email ID Variable Field	Use available fields for the complex variable.
Language Field	Use a simple or complex variable.
AddressLine1 Field	Use a simple or complex variable.
AddressLine2 Field	Use a simple or complex variable.
AddressLine3 Field	Use a simple or complex variable.
AddressLine4 Field	Use a simple or complex variable.
AddressLine5 Field	Use a simple or complex variable.
Country Field	Use a simple or complex variable.
Zip code Field	Use a simple or complex variable.
Automatically update time zone for phone numbers Field	Use the selection box to automatically update the time zone for the phone numbers depending on the country code specified while entering the phone number.
Check phone numbers for reject patterns Field	Use the selection box, if you do not want to save the phone numbers matching the reject patterns.
Check phone numbers for phone formats rule Field	Use the selection box, if you do not want to save the phone numbers matching the phone formats.
Check phone numbers and emails for DNC Field	Use the selection box, if you do not want to save the phone numbers and email addresses existing in the DNC list.
Update existing on duplicate record found Field	Specify if you want to update an existing contact. You can choose to either update the existing record, or ignore the newly added record.
No of additional attributes	Use to specify any additional custom attributes as integer value. If you add custom attributes, you must specify the variable and the variable field values for each custom attributes.
For updating a contact:	
ContactID Variable	For updating, select pomInfo from the drop-down list.
ContactID Variable Field	Contact ID
For fields like **Phone number**, **First Name**, **Last Name**, and **Email ID**, provide specific values or accept inputs.	

Add contact to running job

Use this option to add contact information dynamically to running jobs or campaigns, while the campaign or job is still running in the background.

Before you add any contact to a running job, ensure you have at least one instance of the job running. The POM Web service user should have access to the specified campaign.

The following table lists the property names and the property values:

Property Name	Value
Campaign Name Variable	Use a simple or complex variable.
Campaign Name Variable Field	The campaign name value specified using the campaign name variable.
Campaign Name Variable Constant	If you know the campaign name, enter the campaign name as a constant. **Note:** You can set the constant value or use the name variable.
Note: If the campaign includes more than one job instance, then add the contact to the first available running job instance.	

Get contact info

Use this option to retrieve the existing contact information. The POM Web service user must have access to the contact lists and private attributes.

The following table lists the property names and the property values:

Property Name	Value
POM Contact Info Variable	Use a simple, or complex variable. **Note:** This is a mandatory property.

Property Name	Value
Dynamic Attribute Variable	pomDynamicAttributes dynamic variable. **Note:** Dynamic attributes are stored as arrays.

Update contact attribute

Use this option to update or modify the existing contact information.

The POM Web service user must have access to the contact lists and private attributes.

Note:

Ensure that the attribute you want to update exists in the POM database. The predefined attributes are case-sensitive, and you must map the attributes according to the following table:

Attribute name	Name to be used in the node
Id	UserContactId
First Name	FirstName
Last Name	LastName
Phone Number1	PhoneNumber1
Phone Number1 Country Code	PhoneNumber1CtryCode
Phone Number1 Time Zone	TimeZone
Phone Number2	PhoneNumber2
Phone Number2 Country Code	PhoneNumber2CtryCode
Phone Number2 Time Zone	PhoneNumber2Tz
Email	Email
Language	Language
Title	TitlePredefined
Address Line1	AddrLine1Predefined
Address Line2	AddrLine2Predefined
Address Line3	AddrLine3Predefined
Address Line4	AddrLine4Predefined
Address Line5	AddrLine5Predefined
Country	CountryPredefined

Attribute name	Name to be used in the node
Zip Code	ZipcodePredefined
Time Zone	TimeZone

The following table lists the property names and the property values:

Property Name	Value
Attribute Name Variable	Use a simple or complex variable.
Attribute Name Variable Field	The attribute name value selected above.
Attribute Name Variable Constant	If you know the attribute name, you can enter the attribute name as a constant. **Note:** You can set the constant value, or use the name variable.
Attribute Value Variable	Use a simple or complex variable.
Attribute Value Variable Field	The attribute value selected above.
Attribute Value Variable Constant	If you know the attribute value, you can enter the attribute name as a constant. **Note:** You can set the constant value, or use the value variable.

Get contact attribute

Use this option to retrieve the existing contact information for a specific contact ID. The POM Web service user must have access to the contact lists and private attributes.

Note:

If you need to retrieve values of multiple contact attributes in the Avaya Aura® Orchestration Designer speech or message flow application, then instead of reading multiple attributes values individually, use the get contact info node for reading whole contact record in one request then iterate through the required attributes in the Avaya Aura® Orchestration Designer application.

The following table lists the property names and the property values:

Property Name	Value
Attribute Name Variable	Use a simple or complex variable.
Attribute Name Variable Field	Use the attribute name value selected in Attribute Name Variable.
Attribute Name Variable Constant	If you know the attribute name, you can enter the attribute name as constant. **Note:** You can set the constant value, or use the name variable.
Result Variable	Use a simple or complex variable. **Note:** The retrieved value of the attribute is stored in the result variable.
Result Variable Field	Use a simple or complex field.

Add to DNC list

Use the Add to DNC list option to add telephone numbers or e-mail addresses or both to the DNC list.

You can add contacts to the DNC list only if the POM Web service user is a global user and not an Org user.

Note:

A global user does not belong to any organization and performs the POM Administration and POM Campaign Manager roles. An organization user or Org user belongs to an organization created in Avaya Aura® Experience Portal and performs the Org POM Campaign Manager role.

The following table lists the property names and the property values:

Property Name	Value
Address for DNC List Variable	Use a simple, or session or complex variable.
Address for DNC List Variable Field	Select variable field from the complex variable containing the address value to be added to the DNC list.

Property Name	Value
Address for DNC List Variable Constant	If you know the address variable name, you can enter the address as a constant. **Note:** You can set the constant value or use the address variable.
Organization for DNC List Variable	Use the variable for adding the organization to the DNC list.
Organization for DNC List Variable Field	Use the variable field for adding the organization to the DNC list.
Organization for DNC List Variable Constant	Use the literal for adding the organization to the DNC list.
Check phone numbers for reject patterns	Use the selection box, if you do not want to save the phone numbers matching the reject patterns.
Check phone numbers for phone formats rule	Use the selection box, if you do not want to save the phone numbers matching the phone formats.

Remove from DNC list

Use the remove from DNC list option to remove phone numbers and/or e-mail addresses from the DNC list.

You can remove contacts from the DNC list only if the POM Web service user is a global user and not an Org user.

Note:

A global user does not belong to any organization and performs the POM Administration and POM Campaign Manager roles. An organization user or Org user belongs to an organization created in Avaya Aura® Experience Portal and performs the Org POM Campaign Manager role.

The following table lists the property names and the property values:

Property Name	Value
Address for DNC List Variable	Use a simple, or session or complex variable.
Address for DNC List Variable Field	Select variable field from the complex variable containing the address value to be removed from the DNC list.
Address for DNC List Variable Constant	If you know the address variable name, you can enter the address as a constant.

Property Name	Value
	Note: You can set the constant value or use the address variable.
Organization for DNC List Variable	Use the variable for removing the organization from the DNC list.
Organization for DNC List Variable Field	Use the variable field for removing the organization from the DNC list.
Organization for DNC List Variable Constant	Use the literal for removing the organization from the DNC list.
Check phone numbers for reject patterns	Use the selection box, if you do not want to save the phone numbers matching the reject patterns.
Check phone numbers for phone formats rule	Use the selection box, if you do not want to save the phone numbers matching the phone formats.

Update phone number

Use the update phone number option to update the existing information of any given contact.

The following table lists the property names and the property values:

Property name	Value
Phone Attribute Name Variable	Use a simple or complex variable like phoneInfo.
Phone Attribute Name Variable Field	The attribute name value selected as phone attribute name variable.
Phone Attribute Name Variable Constant	If you know the attribute name, you can enter the attribute name as a constant. **Note:** You can set the constant value, or use the name variable.
Phone Attribute Value Variable	Use a simple or complex variable like phoneInfo.
Phone Attribute Value Variable Field	The attribute value selected as phone attribute value variable.

Property name	Value
Phone Attribute Value Variable Constant	If you know the attribute value, you can enter the attribute name as a constant. **Note:** You can set the constant value, or use the value variable.
CountryCode Value Variable	Use a simple or complex variable like phoneInfo.
CountryCode Value Variable Field	The attribute value selected as countrycode value variable
CountryCode Value Variable Constant	If you know the attribute value, you can enter the attribute name as a constant. **Note:** You can set the constant value, or use the value variable.
TimeZone Value Variable	Use a simple or complex variable like phoneInfo.
TimeZone Value Variable Field	The attribute value selected as timezone value variable.
TimeZone Value Variable.Constant	If you know the attribute value, you can enter the attribute name as a constant. **Note:** You can set the constant value, or use the value variable.
Automatic Update for TimeZone Variable	Boolean value. If you set this value to True, the system automatically updates the time zone for the phone numbers depending on the country code specified while entering the phone number.
Check for Rejection Patterns	Boolean value. If you set this value to True, the system checks the updated phone numbers with the specified rejection patterns in the global configurations.
Check for Phone Formats Rule	Boolean value. If you set this value to True, the system checks the updated phone numbers with the specified phone format rules in the global configurations.
Check for DNC Addresses	Boolean value. If you set this value to True, the system checks the updated phone numbers with the existing DNC list.

Get phone number

Use the get phone number option to retrieve the existing phone number of any given contact.

The following table lists the property names and the property values:

Property name	Value
Phone Attribute Name Variable	Use a simple or complex variable like phoneInfo.
Phone Attribute Name Variable Field	The attribute name value selected as phone attribute name variable.
Phone Attribute Name Variable Constant	If you know the attribute name, you can enter the attribute name as a constant. **Note:** You can set the constant value, or use the name variable.
Phone Info Result Variable	Use complex variable like phoneInfo.

Building Avaya Aura® Orchestration Designer applications

Describing a sample Avaya Aura® Orchestration Designer application scenario

Before you begin

- Install the Avaya Aura® Orchestration Designer 6.x application
- Install and enable the PDC plug-in

About this task

The sample Avaya Aura® Experience Portal application is for a Blood Donation campaign organized by ABC Company. ABC Company runs this campaign every quarter to replenish blood stocks and uses a list of potential blood donors from reputed city colleges. The campaign is to invite donors to donate blood on a weekend and offers donors the choice to opt in for or

opt out of the donation drive. If a donor opts in for the donation, the schedule and venue details are sent to the donor.

Procedure

1. Begin the application flow from *Approot*.
2. Use the Announcement node to greet the customer. You can play a welcome prompt such as Welcome to the blood donation drive of ABC Company.
3. Use the Menu node to give different choices to donors. For example:
   a. Press `1` if you wish to donate blood.
   b. Press `2` if you do not wish to donate blood.
   c. Press `3` if you do not wish to receive such calls in future.
   d. Press `9` if you wish to speak to our coordinator.
4. If the user presses `1`, then use the data node to:
   a. Select the *Update Disposition* PDC node from the left Palette.
   b. Specify the INVACP completion code to the *Update Disposition* node in the database.

      The INVACP is a custom completion code that means that the customer accepted the invitation for blood donation.
   c. Select the *Add Contact To Running Job* PDC node to add this contact to another running job of an infinite email campaign.

      The mail campaign sends email messages along with the venue details to the interested donors.
5. If the user presses `2`, then use the data node to:
   a. Select the *Update Disposition* PDC node from the left palette.
   b. Specify the INVREJ completion code to the *Update Disposition* node in the database.

      The INVREJ is a custom completion code that means that the customer rejected the invitation for blood donation.
6. If the user presses `3`, then use the data node to:
   a. Select the *Add To DNC List* PDC node from the left palette.
   b. Specify the session variable *dnis* to update the dialed number in the database into the DNC list of POM.

      You must set the global restriction for DNC to ensure that none of the numbers in the POM DNC list get a call in future even if you run the same campaign again.
   c. Select the *Update Disposition* PDC node from the left palette.
   d. Specify the DNCREQ completion code to the Update Disposition node in the database.

      The DNCREQ is a custom completion code that means that the customer has activated the Do Not Call option.

7. If the user presses `9`, then use the transfer node of Avaya Aura® Orchestration Designer to transfer the call to an agent or coordinator.

Using the Avaya Aura® Orchestration Designer application in POM

Procedure

1. Deploy the sample Avaya Aura® Orchestration Designer application on one of the application servers.
2. Create one application on Avaya Aura® Experience Portal with application type as `POM:Application`.
3. Assign the URL of the sample Avaya Aura® Orchestration Designer application as `http://<application server IP address>:<port number>/application name/Start`.
4. Save the application, and ensure that the application is accessible.
5. Create one campaign strategy for voice campaign. In the ResultProcessor node, use the Custom node under the Application node for assigning different applications for results such as Answer Human or Call Answered.
6. In the Custom node, all applications with the application type `POM:Application` are available in a drop-down list. Select the sample Avaya Aura® Orchestration Designer application configured in Step 1.
7. Enter the VDN number used in the Avaya Aura® Orchestration Designer application for transferring calls to an agent.

Chapter 3: Web services

You can use Web services to gain access to POM features and functionality. With Web services, you can perform and track routine operations related to campaigns and also gain access to call pacing functionality POM uses 2 Web services, namely, VP_POMAgentAPIService, and VP_POMCmpMgmtService For more details, refer to VP_POMAgentAPIService Web service on page 33, and VP_POMCmpMgmtService Web service on page 64.

About VP_POMAgentAPIService Web service

VP_POMAgentAPIService Web service

POM uses the *VP_POMAgentAPIService* Web service to perform routine operations related to campaigns. For example:

- Retrieving and updating the completion codes
- Retrieving and updating the attributes
- Adding contacts to running jobs
- Saving the contact information
- Adding and removing information in the DNC list
- Checking if the address and phone number is included in the DNC list
- Retrieving customer data

The following is a list of methods available in the Dialog Designer or Avaya Aura® Orchestration Designer application:

- GetContactData method on page 35
- GetContactDataFromList method on page 36
- GetContactAttributeValue method on page 38
- GetContactAttributeValueFromList method on page 40
- SaveContact method on page 41
- SaveContactToList method on page 42
- AddContactToJob method on page 45

Configuring the VP_POMAgentAPIService Web service

Procedure

1. Use a Web browser to open the page http://<IP address>/axis2/services/VP_POMAgentAPIService?wsdl, where IP address is the address of the Avaya Aura® Experience Portal server.
2. Enter a valid EPM user name and password.
3. Save the Web Service Definition Language (WSDL) file.

   You can use this file to build a Web service client and gain access to the Agent API Web service (VP_POMAgentAPIService).

   Note:

   Ensure that the Web service client you generate is an axis2 client. If you upgrade POM from an earlier version to POM 3.0, ensure you regenerate the client using

the new `.wsdl` files. For more information on generating the client, refer to the Apache axis2 documentation from http://ws.apache.org/axis2/.

This Web service conforms to all the current World Wide Web Consortium (W3C) standards.

You can refer to Agent API WSDL file on page 204 for a sample WSDL file.

4. You must mention an endpoint URL to create the axis2 client. The endpoint URL for Agent API Web service is https://<EPserverIPaddress>/axis2/services/VP_POMAgentAPIService.

GetContactData method

POM 3.0 does not support GetContactData. You can use GetContactDataFromList in place of GetContactData. If you are using earlier version of POM, you can use the GetContactData method.

Use the GetContactData method to retrieve contact information from the contact group. The POM Web service user must have access to the contact groups and private attributes.

Parameters for GetContactData method

Parameter	Type	Description
ContactID	String	Unique identification of the contact.
ContactGroup Name	String	The name of the contact group where the contact information is stored.

This method returns the contact data object of type ContactType.

Members of ContactType object

Parameter	Type	Description
ContactID	String	Unique identification for the contact. This parameter is mandatory.
ContactGroupName	String	The name of the contact group. This parameter is mandatory.
FirstName	String	The first name of the contact.
LastName	String	The last name of the contact.
PhoneNumber1	String	The phone number of the contact.
PhoneNumber2	String	The alternative phone number of the contact.

Parameter	Type	Description
Email	String	The e-mail address of the contact.
Language	String	The language of communication to be used.
TimeZone	String	The time zone of the contact.
LastAttemptTime	dateTime	The time at which the contact was last contacted.
LastSuccessfulAttemptTime	dateTime	The time at which the contact was last successfully contacted.
LastCompletionCodeId	Integer	The latest updated completion code.
AttributeObj	AttributeType[]	To specify the custom attributes for the contact record.

Faults for GetContactData method

Return code	Fault message
-2	Contact record not found.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-29	Access Denied - Not a valid attribute for your organization.

GetContactDataFromList method

Use the GetContactDataFromList method to retrieve contact information from the contact list. The POM Web service user must have access to the contact lists and private attributes.

Parameters for GetContactDataFromList method

Parameter	Type	Description
UserContactID	String	Unique identification of the contact.
ContactListName	String	The name of the list where the contact information is stored.

This method returns the contact data object of type ContactDataType.

Parameters of the ContactDataType object

Parameter	Type	Description
UserContactID	String	Unique identification for the contact.
ContactListName	String	The name of the contact list.
FirstName	String	The first name of the contact.
LastName	String	The last name of the contact.
PhoneNumber1	String	The phone number of the contact.
PhoneNumber2	String	The alternative phone number of the contact.
Email	String	The e-mail address of the contact.
LastModifiedOn	dateTime	The last time the address of the contact is modified.
LastModifiedBy	String	The user id who updated the address of the contact.
Language	String	The language of communication to be used.
TimeZone	String	The date and time zone of the contact.
LastAttemptTime	dateTime	The time at which the contact was last contacted.
LastSuccessfulAttemptTime	dateTime	The time at which the contact was last successfully contacted.
LastCompletionCodeId	Integer	The latest completion code updated.
AttributeObj	AttributeType [ ]	To specify the custom attributes for the contact record.
Title	String	The salutation to be used before the first name of the contact. For example, Mr., Mrs, Dr., Ms
AddressLine1	String	The address of the contact.
AddressLine2	String	Additional space provided for noting down the address of the contact.
AddressLine3	String	Additional space provided for noting down the address of the contact.
AddressLine4	String	Additional space provided for noting down the address of the contact.
AddressLine5	String	Additional space provided for noting down the address of the contact.
Country	String	The country of the contact record.
ZipCode	String	The zip code for the contact record.
PhoneNumber1CountryCode	String	The country code for the phone number of the contact record.

Parameter	Type	Description
PhoneNumber2CountryCode	String	The country code for the alternative phone number of the contact record.
PhoneNumber2TimeZone	String	The date and time zone of the PhoneNumber2.

Faults for GetContactDataFromList method

Return code	Fault message
-2	Contact record not found.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-29	Access Denied - Not a valid attribute for your organization.

GetContactAttributeValue method

POM 3.0 does not support GetContactAttributeValue method. You can use GetContactAttributeValueFromList method in place of GetContactAttributeValue method. If you are using earlier versions of POM, you can use GetContactAttributeValue method.

Use the Get ContactAttributeValue method to retrieve specific contact attribute values from the contact group. The POM Web service user must have access to the contact groups and private attributes.

Parameters for GetContactAttributeValue method

Parameter	Type	Description
ContactID	String	Unique identifier for the contact.
ContactGroupName	String	Name of the group where the contact information is saved.
AttributeName	String	The specific attribute name from the given contact group.

This method returns the attribute value.

The predefined attributes are case-sensitive and must be mapped. The following table shows the mapping:

Attribute Name	Name to be used in the method
Id	UserContactId

Attribute Name	Name to be used in the method
First Name	FirstName
Last Name	LastName
Phone Number1	PhoneNumber1
Phone Number1 Country Code	PhoneNumber1CtryCode
Phone Number1 Time Zone	TimeZone
Phone Number2	PhoneNumber2
Phone Number2 Country Code	PhoneNumber2CtryCode
Phone Number2 Time Zone	PhoneNumber2Tz
Email	Email
Language	Language
Title	TitlePredefined
Address Line1	AddrLine1Predefined
Address Line2	AddrLine2Predefined
Address Line3	AddrLine3Predefined
Address Line4	AddrLine4Predefined
Address Line5	AddrLine5Predefined
Country	CountryPredefined
Zip Code	ZipcodePredefined

Faults for GetcontactAttributeValue method

Return code	Fault message
-2	Contact record not found.
-5	Attribute record not found. AttributeName is case-sensitive.
-14	Contact list not found.
-24	Not a valid attribute for this contact record.
-28	Access Denied - Not a valid contact list for your organization.
-29	Access Denied - Not a valid attribute for your organization.
-46	Failed to get attribute value.

GetContactAttributeValueFromList method

Use the GetContactAttributeValueFromList method to retrieve specific contact attribute value from the contact list. The POM Web service user must have access to the contact lists and private attributes.

Parameters for GetContactAttributeValueFromList method

Parameter	Type	Description
ContactID	String	Unique identifier for the contact.
ContactListName	String	Name of the list where the contact information is saved.
AttributeName	String	The specific attribute name from the contact list.

This method returns the attribute value in string.

The predefined attributes are case-sensitive and must be mapped. The following table shows the mapping:

Attribute Name	Name to be used in the method
Id	UserContactId
First Name	FirstName
Last Name	LastName
Phone Number1	PhoneNumber1
Phone Number1 Country Code	PhoneNumber1CtryCode
Phone Number1 Time Zone	TimeZone
Phone Number2	PhoneNumber2
Phone Number2 Country Code	PhoneNumber2CtryCode
Phone Number2 Time Zone	PhoneNumber2Tz
Email	Email
Language	Language
Title	TitlePredefined
Address Line1	AddrLine1Predefined
Address Line2	AddrLine2Predefined

Attribute Name	Name to be used in the method
Address Line3	AddrLine3Predefined
Address Line4	AddrLine4Predefined
Address Line5	AddrLine5Predefined
Country	CountryPredefined
Zip Code	ZipcodePredefined

Faults for GetContactAttributeValueFromList method

Return code	Fault message
-2	Contact record not found.
-5	Attribute record not found. AttributeName is case-sensitive.
-14	Contact list not found.
-24	Not a valid attribute for this contact record.
-28	Access Denied - Not a valid contact list for your organization.
-29	Access Denied - Not a valid attribute for your organization.
-46	Failed to get attribute value.

SaveContact method

POM 3.0 does not support SaveContact method. You can use SaveContactToList in the place of SaveContact method. If you are using earlier versions of POM, you can use SaveContact method.

Use the SaveContact method to save a contact information to any defined contact group.

The first parameter for this method is Contact, an object type of ContactType.

Members of ContactType object

Parameter	Type	Description
ContactID	String	The unique identification for the contact. This parameter is mandatory.
ContactGroupName	String	The name of the contact group. This parameter is mandatory.
FirstName	String	The first name of the contact.
LastName	String	The last name of the contact.

Parameter	Type	Description
Phonenumber 1	Integer	The phone number of the contact.
Phonenumber 2	Integer	The alternative phone number of the contact.
Email	String	The e-mail address of the contact.
LastModifiedOn	String	The last time the address of the contact is modified.
LastModifiedBy	String	The user id who updated the address of the contact.
Language	String	The language of communication to be used.
Time Zone	DateTime	The date and time zone of the contact.
LastAttemptTime	DateTime	The time at which the contact was last contacted.
LastSuccessfulAttemptTime	DateTime	The time at which the contact was last successfully contacted.
LastCompletionCodeId	Integer	The latest completion code updated.
AttributeObj	AttributeType [ ]	To specify the custom attributes for the contact record.

This method returns a zero if the contact is successfully to the POM database.

Faults for the SaveContact method

Return code	Fault message
-5	Attribute record not found. Attribute name is case-sensitive.
-6	Failed to save contact.
-14	Contact list not found.
-28	Access Denied.-. Not a valid contact list for your organization.
-29	Access Denied - Not a valid attribute for your organization.
-38	Invalid value for attribute.

SaveContactToList method

Use the SaveContactToList method to save a contact information to any defined contact list.

The first parameter for this method is ContactToBeSaved, an object type of ContactDataType.

Parameters of SaveContactToList method

Parameter	Type	Description
ContactToBeSaved	ContactDataType	The object type of ContactDataType.
AutomaticUpdateForTimeZone (optional)	Boolean	By default set to false. If AutomaticUpdateForTimeZone is set to true, POM automatically updates the time zone for the phone numbers depending on the country code specified while entering the phone number.
CheckForRejectPattern (optional)	Boolean	By default set to false. If CheckForRejectPattern is set to true, POM applies the global and country specific rejection patterns for the phone number.
CheckForPhoneFormatsRule (optional)	Boolean	By default set to false. If CheckForPhoneFormatRules is set to true, POM applies the country specific phone format patterns for the phone number.
UpdateExisting (optional)	Boolean	By default set to false. If UpdateExisting is set to true, POM updates the existing record, or ignores the newly added record
CheckDNC (optional)	Boolean	By default set to false. If CheckDNC is set to true, POM checks if the contact exists in the DNC list.

Members of ContactDataType object

Parameter	Type	Description
UserContactID	String	The unique identification for the contact. This parameter is mandatory.
ContactListName	String	The name of the contact list. This parameter is mandatory.
Title	String	The salutation to be used before the first name of the contact. For example, Mr., Mrs, Dr., and Ms.
FirstName	String	The first name of the contact.
LastName	String	The last name of the contact.
PhoneNumber1	String	The phone number of the PhoneNumber1.
TimeZone	String	The time zone of PhoneNumber1.
PhoneNumber1CountryCode	String	The country code for the phone number1 of the contact record.
PhoneNumber2	String	The alternative phone number of the contact.
PhoneNumber2CountryCode	String	The country code for the phone number2 of the contact record.

Parameter	Type	Description
PhoneNumber2TimeZone	String	The time zone of PhoneNumber2.
Email	String	The e-mail address of the contact.
Language	String	The language of communication to be used.
AddressLine1	String	The address of the contact.
AddressLine2	String	Additional space provided for noting down the address of the contact.
AddressLine3	String	Additional space provided for noting down the address of the contact.
AddressLine4	String	Additional space provided for noting down the address of the contact.
AddressLine5	String	Additional space provided for noting down the address of the contact.
Country	String	The country of the contact record.
ZipCode	String	The zip code for the contact record.
AttributeObj	AttributeType[]	To specify the custom attributes for the contact record.
LastCompletionCodeId	Integer	The latest completion code updated.
LastAttemptTime	dateTime	The time at which the contact was last contacted.
LastSuccessfulAttemptTime	dateTime	The time at which the contact was last successfully contacted.
LastModifiedOn	dateTime	The last time the address of the contact is modified.
LastModifiedBy	String	The user id who updated the address of the contact.

This method returns a zero if the contact is successfully saved to the POM database.

Faults for the SaveContactToList method

Return code	Fault message
-5	Attribute record not found. AttributeName is case-sensitive.
-6	Failed to save contact.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-29	Access Denied - Not a valid attribute for your organization.
-38	Invalid value for attribute.

AddContactToJob method

POM 3.0 does not support AddContactToJob method. You can use AddContactFromListToJob in place of AddContactToJob. If you are using an earlier version of POM, you can use the AddContactToJob method.

Use the AddContactToJob method to add a contact record to the running campaign job. The POM Web service user should have access to the contact groups and campaign jobs.

Parameters for AddContactToJob method

Parameter	Type	Description
CampaignName	String	The name of the campaign.
ContactID	String	A unique ID with which you can internally identify the contacts imported in the POM database.
ContactGroupName	String	Logical groups of contacts based on different criteria.

Faults for AddContactToJob method

Return code	Fault message
-1	Job record not found.
-2	Contact record not found.
-7	Job contact record not found.
-8	Cannot add contact to the job. It already exists.
-14	Contact list not found.
-26	Campaign record not found.
-27	No running job found for campaign + XXXXX.
-35	Cannot add contact list to job. It already exists.
-34	Cannot add contact record. Invalid job status for Job ID XXX

AddContactFromListToJob method

Use the AddContactFromListToJob method to add a contact record to the running campaign job. The POM Web service user must have access to the contact lists and campaign jobs.

Parameters for AddContactFromListToJob method

Parameter	Type	Description
CampaignName	String	The name of the campaign.
ContactID	String	A unique ID with which you can internally identify the contacts imported in the POM database.
ContactListName	String	Logical lists of contacts based on different criteria.
ContactPriority (optional)	Priority	To set the priority of the contact while filtering contact records. You can have 3 values for priority as LOW, MEDIUM, and HIGH. By default set to MEDIUM. For example, if you set the batch size in the campaign creation wizard as 1, and set the priority of a contact record as HIGH, POM fetches that contact record on HIGH priority over the existing records.

Faults for AddContactFromListToJob method

Return code	Fault message
-1	Job record not found.
-2	Contact record not found.
-7	Job contact record not found.
-8	Cannot add contact to the job. It already exists.
-14	Contact list not found.
-26	Campaign record not found.
-27	No running job found for campaign + XXXXX.
-28	Access Denied - Not a valid contact list for your organization.
-30	Access Denied - Not a valid campaign for your organization
-34	Cannot add contact. Invalid job status for job ID + XXXXX.

IsDNC method

Use the IsDNC method to check if a given phone number, or e-mail address, or SIP address exist in the DNC list.

Parameters for IsDNC method

Parameter	Type	Description
Address	String	Phone number, or e-mail address, or SIP address.
OrgName (optional)	String	Name of the organization. If you provide the OrgName, POM uses Org's DNC list for that organization to execute the DNC operation. As POM checks only Org's DNC list, ensure you check the Common DNC list for PhoneNo and mail as well. While applying DNC for Org based campaigns POM uses addresses in Common DNC list and the Org DNC list, and then applies the DNC.
CheckForRejectPattern (optional)	Boolean	By default set to false. If CheckForRejectPattern is set to true, POM applies the global and country specific rejection patterns for the phone number.
CheckForPhoneFormatRules (optional)	Boolean	By default set to false. If CheckForPhoneFormatRules is set to true, POM applies the country specific phone format patterns for the phone number.

This method returns a True if the phone number, or e-mail address, or SIP address exists in the DNC list, and returns a False if the phone number, or e-mail address, or SIP address does not exist in the DNC list.

Faults for IsDNC method

Return code	Fault message
-18	Address cannot be null.
-19	Invalid address for DNC + XXXX
-21	Access denied.
-40	DNC list not found.
-43	Failed to check DNC address existence.

AddToDNCList method

Use the AddToDNCList method to add phone numbers, or e-mail addresses, or SIP address to the Do Not Call (DNC) list. You can add contacts to the DNC list only if the POM Web service user is a global user and not an Org user.

Parameters for AddToDNCList method

Parameter	Type	Description
Address	String	Phone number, or e-mail address, or SIP address.
OrgName (optional)	String	Name of the organization. If you provide the OrgName, POM uses the appropriate DNC list for that organization to execute the DNC operation.
CheckForRejectPattern (optional)	Boolean	By default set to false. If CheckForRejectPattern is set to true, POM applies the global and country specific rejection patterns for the phone number.
CheckForPhoneFormatRules (optional)	Boolean	By default set to false. If CheckForPhoneFormatRules is set to true, POMapplies the country specific phone format patterns for the phone number.

The AddToDNCList method returns a True value if the phone number or e-mail address or SIP addresses are successfully added to the DNC list.

Faults for AddToDNCList method

Return code	Fault message
-9	Address already there in DNC list.
-18	Address cannot be null.
-19	Invalid address for DNC + XXXX.
-21	Access denied.
-40	DNC list not found.
-41	Failed to add DNC address
-43	Failed to check DNC address existence.

RemoveFromDNCList method

Use the RemoveFromDNCList method to remove phone numbers, or e-mail addresses, or SIP address from the Do Not Call (DNC) list. You can remove contacts to the DNC list only if the POM Web service user is a global user and not an Org user.

Parameters of RemoveFromDNCList method

Parameter	Type	Description
Address	String	Phone number, or e-mail address, or SIP address.

Parameter	Type	Description
OrgName (optional)	String	Name of the organization. If you provide the OrgName, POM uses the appropriate DNC list for that organization to execute the DNC operation.
CheckForRejectPattern (optional)	Boolean	By default set to false. If CheckForRejectPattern is set to true, POM applies the global and country specific rejection patterns for the phone number.
CheckForPhoneFormatRules (optional)	Boolean	By default set to false. If CheckForPhoneFormatRules is set to true, POMapplies the country specific phone format patterns for the phone number.

The RemoveFromDNCList method returns a True value if the phone number, or e-mail address, or SIP address is successfully removed from the DNC list.

Faults for RemoveFromDNCList method

Return code	Fault message
-17	Cannot remove address — not in the DNC list.
-18	Address cannot be null.
-19	Invalid address for DNC. + XXXX.
-21	Access denied.
-40	DNC list not found.
-42	Failed to remove DNC address.
-43	Failed to check DNC address existence.

UpdateContactAttributeValue method

POM 3.0 does not support the UpdateContactAttributeValue method. You can use UpdateContactAttributeValueToList in place of UpdateContactAttributeValue method. If you are using earlier versions of POM, you can use the UpdateContactAttributeValue method.

Use the UpdateContactAttributeValue method to update the individual contact attribute values for the given contact record. The POM Web service user must have access to the contact groups and private attributes.

Parameters for UpdateContactAttributeValue method

Parameter	Type	Description
ContactID	String	Unique identification of the contact.

Parameter	Type	Description
ContactGroupName	String	The name of the group where contact record is saved.
AttributeName	String	The name of the attribute to be updated.
AttributeValue	String	The value of the attribute to be updated.

The predefined attributes are case sensitive and must be mapped according to the following table:

Attribute Name	Name to be used in the method
Id	UserContactId
First Name	FirstName
Last Name	LastName
Phone Number1	PhoneNumber1
Phone Number1 Country Code	PhoneNumber1CtryCode
Phone Number1 Time Zone	TimeZone
Phone Number2	PhoneNumber2
Phone Number2 Country Code	PhoneNumber2CtryCode
Phone Number2 Time Zone	PhoneNumber2Tz
Email	Email
Language	Language
Title	TitlePredefined
Address Line1	AddrLine1Predefined
Address Line2	AddrLine2Predefined
Address Line3	AddrLine3Predefined
Address Line4	AddrLine4Predefined
Address Line5	AddrLine5Predefined
Country	CountryPredefined
Zip Code	ZipcodePredefined

Faults for UpdateContactAttributeValue method

Return code	Fault message
-2	Contact record not found.
-5	Attribute record not found. AttributeName is case-sensitive.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-29	Access Denied - Not a valid attribute for your organization.
-45	Failed to update attribute value.

UpdateContactAttributeValueToList method

Use the UpdateContactAttributeValueToList method to update the individual contact attribute values for the given contact record. The POM Web service user must have access to the contact lists and private attributes.

Parameters for UpdateContactAttributeValueToList method

Parameter	Type	Description
ContactID	String	Unique identification of the contact.
ContactListName	String	The name of the list where contact record is saved.
AttributeName	String	The name of the attribute to be updated.
AttributeValue	String	The value of the attribute to be updated.

The predefined attributes are case-sensitive and you must map the attributes according to the following table:

Attribute Name	Name to be used in the method
Id	UserContactId
First Name	FirstName
Last Name	LastName
Phone Number1	PhoneNumber1
Phone Number1 Country Code	PhoneNumber1CtryCode
Phone Number1 Time Zone	TimeZone

Attribute Name	Name to be used in the method
Phone Number2	PhoneNumber2
Phone Number2 Country Code	PhoneNumber2CtryCode
Phone Number2 Time Zone	PhoneNumber2Tz
Email	Email
Language	Language
Title	TitlePredefined
Address Line1	AddrLine1Predefined
Address Line2	AddrLine2Predefined
Address Line3	AddrLine3Predefined
Address Line4	AddrLine4Predefined
Address Line5	AddrLine5Predefined
Country	CountryPredefined
Zip Code	ZipcodePredefined

Faults for UpdateContactAttributeValueToList method

Return code	Fault message
-2	Contact record not found.
-5	Attribute record not found. AttributeName is case-sensitive.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-29	Access Denied - Not a valid attribute for your organization.
-45	Failed to update attribute value.

GetAllCompletionCodesForCampaign method

Use the GetAllCompletionCodesFor Campaign method to retrieve all custom completion codes for a campaign.

Parameters for GetAllCompletionCodesForCampaign method

Parameter	Type	Description
JobID	Integer	A job is a running instance of any campaign. To internally identify a job, use a unique ID for each running instance.

This method returns the completion codes for the selected campaign.

Faults for GetAllCompletionCodesForCampaign method

Return code	Fault message
-1	Job record not found.
-30	Access Denied - Not a valid campaign for your organization.
-32	No custom completion codes defined for this campaign.

UpdateCompletionCode method

Use the UpdateCompletionCode method to update the completion code for a given contact record. The POM Web service user must have access to campaigns and completion codes.

Parameters for UpdateCompletionCode method

Parameter	Type	Description
PIMSessionID	Double	A unique identifier for a session.
CompletionCode	String	The custom completion code.

This method returns a zero if the custom completion code is successfully updated for the specified contact.

Faults for UpdateCompletionCode method

Return code	Fault message
-1	Job record not found.
-3	Completion code record not found.
-4	Not a valid completion code for this campaign.
-10	Not a system completion code.
-16	Attempt record not found for pimSessionID – XXXX.

Return code	Fault message
-30	Access Denied - Not a valid campaign for your organization.
-31	Record for given POM Session ID not found.

DeleteContact method

POM 3.0 does not support the DeleteContact method. You can use DeleteContactFromList in place of DeleteContact. If you are using earlier versions of POM, you can use the DeleteContact method.

Use the DeleteContact method to permanently delete a particular contact record from the POM database. You cannot delete a contact record if the contact record is a part of a queue, or an active campaign job is using the contact record.

Parameters for DeleteContact method

Parameter	Type	Description
ContactID	String	The contacts imported in the POM database are internally identified by a unique ID.
ContactGroupName	String	Logical groups of contacts based on different criteria.

Faults for DeleteContact method

Return code	Fault message
-2	Contact record not found.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-33	Cannot delete contact. It is being used.
-44	Failed to delete contact record.

DeleteContactFromList method

Use the DeleteContactFromList method to permanently delete a particular contact record from the POM database. You cannot delete a contact record if the contact record is a part of a queue, or an active campaign job is using the contact record.

Parameters for DeleteContactFromList method

Parameter	Type	Description
ContactID	String	The contacts imported in the POM database are internally identified by a unique ID.
ContactListName	String	Logical lists of contacts based on different criteria.

Faults for DeleteContactFromList method

Return code	Fault message
-2	Contact record not found.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-33	Cannot delete contact. It is being used.
-44	Failed to delete contact record.

AddContactGroupToJob method

POM 3.0 does not support AddContactGroupToJob method. You can use AddContactFromListToJob in place of AddContactGroupToJob. If you are using earlier versions of POM, you can use the AddContactGroupToJob method.

Use the AddContactGroupToJob method to add a contact group to a running campaign.

Parameters for AddContactGroupToJob method

Parameter	Type	Description
CampaignName	String	The name of a campaign.
ContactGroupName	String	Logical groups of contacts based on different criteria.

Faults for AddContactGroupToJob method

Return code	Fault message
-1	Job record not found.
-2	Contact record not found.
-7	JobContact record not found.

Return code	Fault message
-8	Cannot add contact record to the job. It already exists.
-14	Contact list not found.
-26	Campaign record not found.
-27	No running job found for campaign + XXXXX.
-28	Access Denied - Not a valid contact list for your organization.
-30	Access Denied - Not a valid campaign for your organization.
-34	Cannot add contact record. Invalid job status for Job ID +XXXXX.

AddContactListToJob method

Use the AddContactListToJob method to add a contact list to a running campaign.

Parameters for AddContactListToJob method

Parameter	Type	Description
CampaignName	String	The name of a campaign.
ContactListName	String	Logical lists of contacts based on different criteria.
ContactPriority (optional)	Priority	To set the priority of the contact while filtering contact records. You can have 3 values for priority as LOW, MEDIUM, and HIGH. By default set to MEDIUM. For example, if you set the priority of contact list as HIGH, POM fetches that contact list on HIGH priority during next batch.
ApplyFilter (optional)	Boolean	By default set to false. If ApplyFilter is set to true, POM applies the filter criteria specified during campaign creation.

Faults for AddCotactListToJob method

Return code	Fault message
-1	Job record not found.
-2	Customer record not found.
-7	JobContact record not found.
-8	Cannot add customer record to the job. It already exists.
-14	Contact list not found.

Return code	Fault message
-26	Campaign record not found.
-27	No running job found for campaign + XXXXX.
-28	Access Denied - Not a valid outreach list for your organization.
-30	Access Denied - Not a valid campaign for your organization.
-34	Cannot add customer record. Invalid job status for Job ID +XXXXX.
-35	Cannot add contact list to job. It already exists.

GetPhoneNumber method

Use the GetPhoneNumber method to retrieve a phone number from the contact record. The GetPhoneNmber method searches for the given contact ID, contact list name, and phone attribute name. The GetPhoneNumber returns the phone number and its associated timezone and country code attribute values.

Parameters of GetPhoneNumber method

Parameter	Type	Description
ContactID	String	Unique identification of the contact.
ContactListName	String	The name of the list where the contact information is stored.
PhoneAttributeName	String	The phone number attribute name to be retrieved.

This method returns the data object of type PhoneType.

Members of PhoneType object

Parameter	Type	Description
CountryCode	String	The country code value associated with the phone attribute.
PhoneNumber	String	The value of the phone number.
TimeZone	String	The time zone value associated with the phone attribute.
Phone AttributeName	String	The phone attribute name to be retrieved.

Faults for GetPhoneNumber method

Return code	Fault message
-2	Contact record not found.
-5	Attribute record not found. AttributeName is case-sensitive.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-37	Failed to get phone number.

ScheduleCallBack method

Use this method to schedule a call back to a specific customer if the call is not answered.

Parameters

Parameter	Type	Description
UserContactID	String	Unique identification of the customer.
ContactGroupName	String	The name of the group where the customer information is stored.
CampaignName	String	The campaign name for which you want to retrieve the job ID for scheduling the callback. For multiple active jobs, first job ID is used.
Address (optional)	String	Any free form phone number or e-mail address of the contact for callback. **Note:** This is an optional parameter. If you do not specify a value, the system picks up the default value specified in the campaign strategy.
StartTime	String	The preferred data and time to start or enable call back for the given customer record. The format is yyyy/MM/dd HH:mm.
EndTime	String	The date and time to end or disable call back for the given customer record. The format is yyyy/MM/dd HH:mm.
TimeZone (optional)	String	The time zone for the given customer record. The format is yyyy/MM/dd HH:mm.

Parameter	Type	Description
ContactAttribute Name (optional)	String	The name of the contact attribute of PhoneType for the given contact record.
Notes	String	The call back notes for reference.

Faults

Return code	Fault message
-2	Contact record not found.
-19	Invalid address for customer. + XXXX.
-26	Campaign record not found.
-28	Access Denied - Not a valid contact group for your organization.
-30	Access Denied - Not a valid campaign for your organization.
-39	Failed to schedule the callback.

UpdatePhoneNumber method

Use the UpdatePhoneNumber method to update the phone number, the associated timezone and country code attribute values for a given contact record. The parameter list for this method includes PhoneObject and object of type PhoneType.

Parameters of UpdatePhoneNumber method

Parameter	Type	Description
ContactID	String	Unique identification of the contact.
ContactListName	String	The name of the list where contact is stored.
PhoneObejct	PhoneType	The value of phone number.
AutomaticUpdateForTimeZone (optional)	Boolean	Use the selection box to automatically update the time zone for the phone numbers depending on the country code specified while entering the phone number.
CheckForRejectPattern (optional)	Boolean	Use the selection box, if you do not want to save the phone numbers matching the reject patterns.
CheckForPhoneFormatsRule (optional)	Boolean	Use the selection box, if you do not want to save the phone numbers matching the phone formats.

Parameter	Type	Description
CheckDNC (optional)	Boolean	Use the selection box to check if the phone number exists in the DNC list.

This method returns a zero if the given phone number is updated successfully. This method returns the data object of type PhoneType.

Members of PhoneType object

Parameter	Type	Description
CountryCode	String	The country code value associated with the phone attribute.
PhoneNumber	String	The value of the phone number.
TimeZone	String	The time zone value associated with the phone attribute.
Phone AttributeName	String	The phone attribute name to be retrieved.

Faults for UpdatePhoneNumber method

Return code	Fault message
-2	Contact record not found.
-5	Attribute record not found. AttributeName is case-sensitive.
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-36	Failed to update phone number.
-38	Invalid value for attribute.

GetAttributesList method

Use the GetAttributeList method to retrieve the list of attributes specific to the logged in user's organization. For non-Org users, POM displays the entire list of all attributes configured on the system. The list includes the predefined attribute names, custom attribute names and the respective attribute types. The method does not have any input parameter, but returns an array of AttributeType object. If you provide the contact list name as an input, the system displays the associated attributes.

Parameters of GetAttributeList method

Parameter	Type	Description
ContactListName (optional)	String	Name of the contact list.

Parameters of AttributeType

Parameter	Type	Description
Name	String	Name of the attribute key name
Type	String	Data type of attribute
DisplayName	String	Localized name of the attribute

Fault for GetAttributeType method

Return code	Fault message
-14	Contact list not found.
-28	Access Denied - Not a valid contact list for your organization.
-44	Failed to get attribute list.

EmptyContactList method

Use the EmptyContactList method to simply empty a contact list by name, that is, delete all records in the list. It will delete all records in the list that are older than the time when the method was invoked. You cannot empty a contact list if the contact list is a part of a queue, or an active campaign job is using the contact list.

Parameters for EmptyContactList method

Parameter	Type	Description
ContactListName	String	The name of the contact list you want to empty.

This method returns total contacts remaining since you started emptying the contact list.

Note:

Emptying and adding new records may progress simultaneously on a list

Faults for EmptyContactList method

Return code	Fault message
-14	Contact list not found
-28	Access Denied - Not a valid contact list for your organization
-61	Failed to empty contact list

GetContactListEmptyStatus method

Use this method to fetch the status of this task, returning if the deletion has completed or not, and the total number of records in the contact list when the method is invoked.

Parameters for GetContactListEmptyStatus method

Parameter	Type	Description
ContactListName	String	The name of the contact list

This method will return an object of ContactListStatusType.

Parameters of ContactListStatusType

Parameter	Type	Description
EmptyStats	ListStatus	Can be either LIST_BEING_IDLE or LIST_BEING EMPTIED
TotalCount	Long	The total count of records remaining

Faults for GetContactListStatus method

Return code	Fault message
-14	Contact list not found
-28	Access Denied - Not a valid contact list for your organization
-61	Failed to empty contact list

UpdateCampaignAttributeValue method

Use the UpdateCampaigntAttributeValue method to update the individual campaign attribute values for the given contact record. The POM Web service user must have access to the contact groups and private attributes.

Parameters for UpdateCampaignAttributeValue method

Parameter	Type	Description
JobId	Integer	Unique identification of the job.
AttributeName	String	The name of the attribute to be updated.
AttributeValue	Double	The value of the attribute to be updated.
Operation (optional)	OperationType	Can have 3 values: ADD, MINUS, and ASSIGN. By default ASSIGN is set. For example, if you set the operation type as ADD, POM increments existing attribute value with the specified number.

Faults for UpdateCampaignAttributeValue method

Return code	Fault message
-1	Job record not found.
-45	Failed to update campaign attribute value.

UpdateAgentAttributeValue method

Parameters for UpdateAgentAttributeValue method

Parameter	Type	Description
JobId	Integer	Unique identification of the job.
AgentId	String	Unique identification of the agent.
AttributeName	String	The name of the attribute to be updated.
AttributeValue	Double	The value of the attribute to be updated.
Operation	OperationType	Can have 3 values: ADD, MINUS, and ASSIGN. By default ASSIGN is set. For example, if you set the operation type as ADD, POM increments existing attribute value with the specified number.

Faults for UpdateAgentAttributeValue method

Return code	Fault message
–1	Job record not found.
-45	Failed to update attribute value.
-66	Active session not found for agent.

About VP_POMCmpMgMtService Web service

VP_POMCmpMgmtService Web service

POM uses *VP_POMCmpMgmtService* to manage campaign-related features, such as starting, stopping, and pausing campaigns and to gain access to the call pacing features. Use call pacing to restrict or determine the maximum number of simultaneous calls allowed for each campaign. This Web service needs a valid Avaya Aura® Experience Portal user for authorization.

If you have turned the multi-tenancy on, ensure that the Avaya Aura® Experience Portal user belongs to the same organization as that of the campaign. For details on multi-tenancy, refer to the *Organizational level access* section in the *Administering Avaya Aura® Experience Portal* guide. Alternatively, this Web service considers a global user as a valid user who can gain access to all campaigns across organizations.

You need to create a campaign with call pacing turned on. When the campaign starts, the Campaign Manager checks the license availability. If a license is available and call pacing is enabled, then the Campaign Manager will refer to the call attempt value. POM considers the call attempt as complete when a call is transferred to an agent, disconnected, or unsuccessful.POM uses the nonzero value specified in the call attempts and enables the maximum attempts. POM keeps running with this value and guarantees that the attempts do not exceed the specified value at any point. Of the following methods listed for the *VP_POMCmpMgmtService* Web service, the first two are related to call pacing:

- GetCampaignDetails method on page 66
- GetActiveJobs method on page 67
- SetMaxAttemptsCount method on page 67
- PauseActiveJob method on page 68
- ResumePausedJob on page 69

Configuring the VP_POMCmpMgmtService Web service

Procedure

1. Use a Web browser to open the page http://<IP address>/axis2/services/VP_POMCmpMgmtService?wsdl, where IP address is the address of the Avaya Aura® Experience Portal server.
2. Enter a valid EPM user name and password.
3. Save the Web Service Definition Language (WSDL) file.

   You can use this file to build a Web service client and gain access to the CmpMgmt Web service (VP_POMCmpMgmtService).

   Note:

   Ensure that the Web service client you generate is an axis2 client. If you upgrade POM from an earlier version to POM 3.0, ensure you regenerate the client using new `.wsdl` files. For more information on generating the client, refer to the Apache axis2 documentation from http://ws.apache.org/axis2/.

   This Web service conforms to all the current World Wide Web Consortium (W3C) standards.

   You can refer to VP_POMCmpMgmt WSDL file on page 185 for a sample WSDL file.

4. You must mention an endpoint URL to create the axis2 client. The endpoint URL for CmpMgmt Web service is https://<EPserverIPaddress/axis2/services>/VP_POMCMpMgmtService.

GetCampaignDetails method

Use the CampaignDetails method to get the details of the campaign job.

Parameters for GetCampaignDetails method

Parameter	Type	Description
JobState (optional)	String [ ]	The job state such as JOB_QUEUED, JOB_ACTIVE, JOB_PAUSED, or JOB_COMPLETED. You can use these state names to filter the job IDs matching the corresponding state.

This method returns an array of CampaignNameJobIDsAndStates object.

Note:

If JobState parameter value is not passed or is set as null, the GetCampaignDetails method returns a list of campaign names specific to organization. In such cases, the JobID and JobState are set to less than 0 and null respectively.

Members of CampaignNameJobIDAndStates

Parameter	Type	Description
CampaignName	String	Name of the campaign
JobID (optional)	Integer	A job is a running instance of a campaign. This is internally identified with the help of a unique ID for each running instance of the job.
JobState (optional)	String	The job state such as JOB_QUEUED, JOB_ACTIVE, JOB_PAUSED, or JOB_COMPLETED for the job.

Faults for GetCampaignDetails method

Return code	Fault message
-53	Invalid input for job state.
-58	Failed to get campaign details.

GetActiveJobs method

Use the GetActiveJobs method to retrieve the active jobs for a particular campaign. Ensure that the call pacing feature is turned on.

Parameter

Parameter	Type	Description
CampaignName	String	The campaign name to retrieve the active jobs.

The method returns a Jobs parameter of Set<Integer> type, which has the job IDs for the active jobs. Jobs in Stopped, Stopping, and Completed state are not considered as active.

Note:

POM considers paused jobs as active jobs.

Faults

Return code	Fault message
-22	Call pacing not enabled for campaign + XXXX.
-26	Campaign record not found.
-27	No running job found for campaign + XXXX.

SetMaxAttemptsCount method

Use the SetMaxAttemptsCount method to set the maximum number of simultaneous calls. A campaign with call pacing enabled always starts with 0 as the value for the Count parameter, and so does not place calls till the count is set to a nonzero positive value. POM uses the nonzero value specified in the call attempts and starts the maximum attempts. POM keeps running with this value and guarantees that the attempts do not exceed the specified value at any point.

Note:

If you are upgrading from POM 2.0 to POM 2.5 Service Pack 1, then use SetMaxAttemptsCount method for earlier campaigns. Use SetMaxAttemptsCountForTask for new campaigns created using POM 2.5 Service Pack 1.

Parameters

Parameter	Type	Description
JobID	Integer	Job ID of an active job.
Count	Integer	Maximum number of call attempts allowed.

Faults

Return code	Fault message
-1	Job record not found.
-20	Agent Job Summary record not found.
-22	Call custom pacing not enabled for campaign + XXXX.
-23	Invalid value for call attempts count.
-50	Failed to set maximum call attempts count.

PauseActiveJob method

Use the PauseActiveJob method to pause an active job in a running campaign.

Parameter

Parameter	Type	Description
JobID	Integer	A job is a running instance of any campaign. This is internally identified with the help of a unique ID for each running instance.

This method returns True if the active job is successfully paused.

Faults

Return code	Fault message
-1	Job record not found.
-30	Access denied. Not a valid campaign for your organization.
-55	Job state is not active.

ResumePausedJob

Use the ResumePausedJob method to resume a paused job.

Parameter

Parameter	Type	Description
JobID	Integer	A job is a running instance of any campaign. This is internally identified with the help of a unique ID for each running instance.

This method returns a True if a paused campaign resumes successfully.

Faults

Return code	Fault message
-1	Job record not found.
-30	Access denied - Not a valid campaign for your organization.
-56	Job state is not paused.

StopJob method

Use the StopJob method to stop an active job or terminate a paused job.

Parameter

Parameter	Type	Description
JobID	Integer	A job is a running instance of any campaign. This is internally identified with the help of a unique ID for each running instance.

This method returns a True if an active, or a paused job stops successfully.

Faults

Return code	Fault message
-1	Job record not found.
-30	Access denied - Not a valid campaign for your organization.
-57	Job state is not in paused or active state.

RunCampaign method

Use the RunCampaign method to run a campaign based on some event. For example, if an IT group is running a campaign, you must call all the support personnel belonging to that IT group if the file server becomes nonfunctional. In such cases, you cannot schedule a campaign as the time at which the problem occurs is unknown. You can then program the campaign to run based on the occurrence of the event.

Parameter

Parameter	Type	Description
CampaignName	String	Each campaign has a unique campaign name.

This method returns True if a campaign job is successfully queued.

Faults

Return code	Fault Message
-26	Campaign record not found.
-51	Infinite campaign job already running.
-52	Failed to start the campaign job.

GetCampaignJobs method

Use the GetCampaignJobs method to retrieve the job ID and the job state for a given campaign. If you do not specify any job state, then the system retrieves all the job ID, irrespective of a specific state, for the given campaign. This list displays all jobs including finished jobs.

Parameters

Parameter	Type	Description
CampaignName	String	Each campaign is identified with a unique campaign name. Specify the campaign name for which you want to retrieve the job ID and job state.
JobState (optional)	String[]	The job state, such as JOB_QUEUED, JOB_ACTIVE, JOB_PAUSED, or JOB_COMPLETED. You can use these names to filter the job IDs matching the corresponding state.

This method returns an array of `JobIdsAndStates` object.

Faults

Return code	Fault message
-1	Job record not found.
-26	Campaign record not found.
-30	Access Denied - Not a valid campaign for your organization.
-53	Invalid job state.

GetActiveJobTaskIDs method

Use the GetActiveJobTaskIDs method to retrieve all the active jobs and their task IDs for the specified campaign.

Parameters for GetActiveJobTaskIDs method

Parameter	Type	Description
CampaignName	String	The name of the campaign.

This method returns an array of JobIDAndTask which has 2 fields; JobID and TaskID for the specified campaign.

Faults for GetActiveJobTaskIDs method

Return code	Fault message
-26	Campaign record not found.
-27	No running job found for campaign + XXXX.
-47	Failed to get job and action ID values.

SetMaxAttemptsCountForTask method

Use the SetMaxAttemptsCountForTask method to set the maximum number of simultaneous calls. A campaign with tasks having call pacing enabled always starts with 0 as the value for the Count parameter, and so does not place calls till the count is set to a nonzero positive value. POM uses the nonzero value specified in the call attempts, and launches the maximum

attempts. POM keeps running with this value and guarantees that the attempts do not exceed the specified value at any point.

Parameters

Parameter	Type	Description
JobID	Integer	A job is a running instance of any campaign. This is internally identified with the help of a unique ID for each running instance.
TaskID	Integer	Task Id retrieved from the GetActiveJobTaskIds and GetActiveJobTaskIdForTask method.
Count	Integer	Maximum number of call attempts allowed.

Faults

Return code	Fault message
-1	Job record not found.
-20	Agent job summary record not found
-23	Invalid value for call attempts count.
-48	Task not found
-49	Custom call pacing disabled for task + XXXX.
-50	Failed to set the maximum call attempts count.

GetActiveJobTaskIdForTask method

Use the GetActiveJobTaskIdforTask method to get active job and task id for a specified task name. The GetActiveJobTaskIdForTask method can be used to set the maximum attempt count for every action used in the campaign strategy.

Parameters

Parameter	Type	Description
CampaignName	String	The name of the campaign.
TaskName	String	The name of the task.

This method returns an array of JobIDAndTaskID which has 2 fields; JobID and TaskID.

Faults

Return code	Fault message
-26	Campaign record not found.
-27	No running job found for campaign + XXXX.
-47	Failed to get job and action ID values.
-48	Task not found.
-49	Custom call pacing disabled for task + XXXX.

GetImportJobStatus method

Parameters for GetImportJobStatus method

Parameter	Type	Description
DataSourceName	String	Name of the data source
JobStates	ImportJobState[ ]	Job status. You can specify multiple job states. The valid values can be: • COMPLETED • QUEUED • RUNNING • ERROR • FILE_COPYING • PAUSING • PAUSED • STOPPING • WAITING_TO_RESUME • DELETING_CONTACTS • CREATING_HISTORY

This method returns an array of ImportJobStatus object. The members of ImportJobStatus are: Add from WSDL

Parameter	Type	Description
ImportName	String	Name of the data source

Parameter	Type	Description
ListName	String	Job status
Status	ImportJobState[ ]	Job status. You can specify multiple job states. The valid values can be: • COMPLETED • QUEUED • RUNNING • ERROR • FILE_COPYING • PAUSING • PAUSED • STOPPING • WAITING_TO_RESUME • DELETING_CONTACTS • CREATING_HISTORY
SuccessCount	Long	
UpdateCount	Long	
RuntimeErrorCount	Long	
ValidationFailedCount	Long	
DuplicateIgnoredCount	Long	
MatchPhoneRejectPatternCount	Long	
DeleteCount	Long	Total number of contacts deleted
MatchesDncCount	Long	Total number of contacts existing in the DNC list
PhoneFormatFailedCount	Long	Total number of contacts where the phone format does not match
ProcessedRecordCount	Long	Total number of contacts processed

Fault messages for GetImportJobStauts method

Return code	Fault message
-12	Data Source record not found.
-60	Access Denied - Not a valid data import for your organization.

Return code	Fault message
-63	Failed to get data import job details.

ScheduleDataSource method

Parameters for ScheduleDataSource method

Parameter	Type	Description
DataSourceName	String	Name of the data source for which you want to schedule imports.
StartTime	String	Specify the start date and time for the triggering the import. Ensure that the start date and format is yyyy/MM/dd HH:mm:ss.
TimeZone	String	Specify the time zone you want to use for scheduling the imports.

Fault messages for ScheduleDataSource method

Return code	Fault message
-12	Data source not found.
-59	Failed to add schedule.
-60	Access Denied - Not a valid data import for your organization.

ScheduleCampaign method

Use the ScheduleCampaign method to schedule jobs for finite or infinite campaigns.

Parameters for ScheduleCampaign method

Parameter	Type	Description
CampaignName	String	Name of the campaign for which you want to schedule jobs.
StartTime	String	Specify the start date and time for the triggering campaign. Ensure that the start date and format is yyyy/MM/dd HH:mm:ss.

Parameter	Type	Description
TimeZone	String	Specify the time zone you want to use for scheduling the jobs.
ArchivalScheduleFrequency	ArchivalFrequencyType	Specify the archival frequency for the campaign. It can be either Hourly, or RunEveryNHours, or DailyAt
ArchivalTimeInHrsMins	String	Specify the archival frequency in hours and minutes.
ArchivalInHrs	String	Specify the archival frequency in hours.

Fault messages for ScheduleCampaign method

Return code	Fault message
-26	Campaign record not found.
-30	Access Denied - Not a valid campaign for your organization.
-59	Failed to add schedule.

ScheduleRecurringDataSource method

Parameters for ScheduleRecurringDataSource method

Parameter	Type	Description
CampaignName	String	Name of the data source for which you want to schedule imports.
StartTime	String	The start date and time for the triggering the import. Ensure that the start date and format is yyyy/MM/dd HH:mm:ss.
EndTime	String	The start date and time for the terminating the import. Ensure that the start date and format is yyyy/MM/dd HH:mm:ss.
TimeZone	String	The time zone you want to use for scheduling the imports.
ScheduleFrequency	String	The frequency for the import. You can specify any one of the values: • RunEveryNMins: This option creates a import job for every N specified minutes. • Daily : This option creates a import job daily at the start time you mention during

Parameter	Type	Description
		scheduling and continues till the end date time. • Weekly : This option creates import jobs on specified days and weekly recurring jobs are automatically created. You can select the days of the week. For example, if you select Monday and Friday, then the weekly import jobs are created on Monday and Friday at the start time mentioned during schedule. • Monthly: This option creates the import jobs on a monthly basis depending on the start date till the finish date. • Yearly : This option creates the import jobs on a yearly basis depending on the start date till the finish date.
WeekDaysOnly	String	Creates a import job on all the days of the week except the weekend days you mention in the **POM Home > Configurations > Global Configurations > Campaign Settings** field.
SelectedDays	String	This array can be used to select specific days only for weekly recurring schedule.
RunEveryMinutes	String	This parameter can be used to specify value in minutes for RunEveryNMinutes schedule.

Fault messages for ScheduleRecurringDataSource method

Return code	Fault message
-12	Data source not found.
-59	Failed to add schedule.
-60	Access Denied - Not a valid data import for your organization.

ScheduleRecurringCampaign method

Parameters for ScheduleRecurringCampaign method

Parameter	Type	Description
CampaignName	String	Name of the campaign for which you want to schedule jobs.
StartTime	String	The start date and time for the triggering campaign. Ensure that the start date and format is yyyy/MM/dd HH:mm:ss.
EndTime	String	The start date and time for the terminating the campaign. Ensure that the start date and format is yyyy/MM/dd HH:mm:ss.
TimeZone	String	Time zone you want to use for scheduling the jobs.
ScheduleFrequency	String	The frequency for finite campaign. You can specify any one of the values: • RunEveryNMins: This option creates a job for the campaign every N specified minutes. • Daily : This option creates a job daily at the start time you mention during scheduling and continues till the end datetime. • Weekly : This option creates jobs on specified days and weekly recurring jobs are automatically created. You can select the days of the week. For example, if you select Monday and Friday, then the weekly jobs are created on Monday and Friday at the start time mentioned during schedule. • Monthly: This option creates the jobs on a monthly basis depending on the start date till the finish date. • Yearly : This option creates the jobs on a yearly basis depending on the start date till the finish date.
WeekDaysOnly (Optional)	String	Creates a job on all the days of the week except the weekend days you mention in the **POM Home > Configurations > Global Configurations > Campaign Settings** field.
SelectedDays (Optional)	String[ ]	This array can be used to select specific days only for weekly recurring schedule.

Parameter	Type	Description
ArchivalScheduleFrequency	ArchivalFrequencyType	Specify the archival frequency for the campaign. It can be either Hourly, or RunEveryNHours, or DailyAt
ArchivalTimeInHrsMins	String	Specify the archival frequency in hours and minutes.
ArchivalInHrs	String	Specify the archival frequency in hours.
RunEveryMinutes (Optional)	String	This parameter can be used to specify value in minutes for RunEveryNMinutes schedule.

Fault messages for ScheduleRecurringCampaign method

Return code	Fault message
-26	Campaign record not found.
-30	Access Denied - Not a valid campaign for your organization.
-59	Failed to add schedule.

GetContactListNames method

Parameters for GetContactListNames method

Parameter	Type	Description
CampaignName	String	The name of the campaign.

Faults for GetContactListNames method

Return code	Fault message
-26	Campaign record not found.
-30	Access Denied - Not a valid campaign for your organization.
-68	Infinite campaign without associated contact list.

Chapter 4: Custom connectors, interface definitions, and class files

Creating a custom data import connector

You can use a custom connector to import data from your database or any other source, such as ERP systems or a CRM software. To implement the custom interface, you must implement `ContactListCreator`:

```
public interface ContactListCreator

{

void init() throws Exception;

boolean hasMoreContacts();

ArrayList<PimContact>getNextContactBatch()throws Exception;

}
```

Before you begin

To use the provided customer interfaces, you must copy the following .jar files in your development environment:

- `avaya-pim-common.jar` file. This .jar file references:
  - `import com.avaya.pim.jdbc.bo.AttributeBO`
  - `avaya-pim-hibernate.jar` file. This .jar file references:
    - `import com.avaya.pim.jdbc.hibernate.PimAttribute`
    - `import com.avaya.pim.jdbc.hibernate.PimContact`
    - `import com.avaya.pim.jdbc.hibernate.PimContactAttribute`
  - `hibernate3.jar`. This .jar file references:
    - `import org.hibernate.Session;`

Note:
POM uses the hibernate feature for database interactions.

These files are in the `$POM_HOME/lib/common` folder.

- `avaya-pim-core.jar`. You can find this file in the `$POM_HOME/lib/core folder`.

About this task

To implement the ContactListCreator interface:

Procedure

1. To implement java class, extend the ImportDsJob class.

```
public class CustomImport extends ImportDsJob implements
ContactListCreator
```

```
{
```

```
.
```

```
.
```

```
}
```

2. Use the constructor with the importId parameter.

```
public CustomImport(Session session, int importId) throws Exception
```

```
{
```

```
super(session, importId);
```

```
}
```

3. Implement the following methods from Interface ContactListCreator:

```
void init() throws Exception;
```

To initialize the variables in the user class:

```
ArrayList<PimContact>getNextContactBatch()throws Exception;
```

To create a list of PimContact objects:

```
boolean hasMoreContacts();
```

After every call to `getNextContactBatch()`, the POM Import Manager calls this function to check for more contacts.

4. To create a list of PimContacts:
   a. Create a list to store contacts:

```
ArrayList<PimContact> = new ArrayList<PimContact>();
```

   b. Create a PIMContact object using contact information from your source:

```
PimContact contactObj = new PimContact();
        contactObj.setUserContactId(contactId);
        contactObj.setFirstName(firstName);
        contactObj.setLastName(lastName);
        contactObj.setPhoneNumber1(phoneNumber1);
        contactObj.setPhoneNumber2(phoneNumber2);
        contactObj.setEmail(email);
        contactObj.setLanguage(language);
        contactObj.setTimeZone(timzone)
```

```
Date now = new Date();
contactObj.setLastModifiedOn(now);
```

c. Associate attributes with objects:

i.
```
PimAttribute tmpAttribute
=AttributeBO.getAttributeObj(attributeName);
```

Pass the attribute name to `AttributeBO.getAttributeObj()` function.

Warning:

If this attribute is present in the POM system, the function returns the object. If the attribute is missing, the function returns null. Ensure that you add the attribute in the POM system first.

If the attribute you try to import does not belong to your organization, the function `AttributeBO.getAttributeObj(attributeName)` throws exceptions, and you need to resolve the exceptions.

To add attributes, refer to *Adding Attributes* section from the *Using POM* guide.

ii.
```
PimContactAttribute contactAttributeObj = new
PimContactAttribute(contactObj,  tmpAttribute,
attributeValue);
```

This function takes the contact object, the attribute object, and the attribute values, and binds them together to the `contactAttributeObj`.

iii. Create a set to store contact attributes:

```
Set <PimContactAttribute>contactAttributeSet = new
HashSet<PimContactAttribute>();
contactAttributeSet.add(contactAttributeObj);
```

iv. Associate the set to contact object:

```
contactObj.setPimContactAttributes(contactAttributeSet);
```

d. Add this contact object to a contact list:

```
numberList.add(contactObj);
```

e. Return this contact list:

```
return numberList;
```

After you finish creating the contacts, `hasMoreContacts()` must return false.

5. Create a jar file from this class, for example`CustomImport.jar`.
6. Copy the jar file to the `$POM_HOME/lib/custom` folder on all the POM servers.
7. Restart POM service by typing`/sbin/service POM restart`.
8. Restart VPMS service by typing`/sbin/service vpms restart`.
9. To create a data source from the jar:

a. In the left pane, select **POM Configurations > Contacts > Contact Data Sources**.
b. Click **Add**.
c. Select the **Custom** option button.
d. Specify the name and description.
e. Select the contact list from the drop-down list.
f. Click **Next.**
g. Specify the class name.
h. Click **Finish** to finish the data source creation.

Note:

If you have not added attributes, and you are using the custom class to import contact data, the system does not display any error messages while adding the contact data source. After running the import, check the Import Monitor to verify if the import is successful.

Creating a custom class for post processing of jobs

About this task

Ensure you have the `avaya-pim-pomapi.jar` file.

Procedure

1. Copy the jar file in your development environment. You can find the .jar file in `$POM_HOME/lib/common` folder.
2. Use the .jar file to process all the successful contacts after campaign execution. Successful contacts refers to all the contacts POM contacted using the campaign. For example, you can export all contacts with customized details, such as campaign name, campaign ID, phone number, and e-mail address. The custom class must implement the following `PomJobPostProcessor` interface:

```
public interface PomJobPostProcessor

{

public void processContactAttempt(PomCampaignInfo pomCampaignInfo,
PomCampaignJobInfo pomCampaignJobInfo, PomContactInfo pomContactInfo,
PomAttemptInfo attemptInfo) throw Exception;

 }
```

The interface uses the following four objects:

```
public class PomCampaignInfo
```

```
{

private int campaignId; // campaign ID


private String orgName; // Organization to which the campaign belongs


private String contactStrategyName;// contact strategy used in the
campaign


private int isInfinite;// campaign is finite or infinite


private int priority; // campaign priority


private String name; // campaign name


private String description; // campaign description


private boolean enablePacing; // call pacing is enabled or disabled


private String createdBy; // name of the user who created the campaign


private String lastModifiedBy; // name of the user who last modified the
campaign


private Date lastModifiedOn; // date and time when the campaign was last
modified


private Date lastJobStartTime; // date and time when the last job of the
campaign was started


private String dialingPrefix; // the dialing prefix used to make outbound
calls


private String smsPrefix;// the SMS prefix used to send SMS


private boolean enableComplianceTimers; // compliance timers used or not


private int startOfVoiceTimeout; // start of voice timeout value for the
given campaign


private int liveVoiceTimeout; // start of live voice timeout value for the
given campaign

}
```

```
public class PomCampaignJobInfo

{
```

```
private int jobId; // job ID

private Date startTime;// date and time when the job starts

private Date endTime; // date and time when the job finishes

private String finishReason; // reason for job finish

}
```

```
public class PomContactInfo

{

private String contactId; // contact ID

private int contactGroupId;// contact group ID to which the contact belongs

private String contactGroupName; // contact group name to which the
contact belongs

private String phoneNumber1; // primary phone number of the contact

private String phoneNumber2; // secondary or alternative phone number of
the contact

private String firstName; // first name of the contact

private String lastName; // last name of the contact

private String email; // e-mail address of the contact


private String language; // default language selected for the contact

private String timeZone; // time zone selected for the contact

private Date lastAttemptTime; // when the contact was last attempted

private Date lastSuccessfulAttemptTime; // date and time when the contact
was last attempted

private HashMap<String, String>contactAttribute; // key value pairs for
contact custom attributes and their values
```

```
public class PomAttemptInfo
```

```
{

private long pimSessionId; // POM session ID


private String pimCompletionCode; // system and custom completion code for
the specific attempt


private String pimSysCompletionCode; // system completion code


private int pimJobId; // internal contact ID


private String pimServerName;// POM server name


private String sessionId; // voice portal session ID

private Date contactAttemptTime; // date and time when the attempt is made


private Date ringbackStartTime; // date and time of start of ringing


private Date lastNwDispositionTime; // date and time when the last
disposition was received


private Date callStartTime; // date and time when the call starts


private Date callCompletionTime; // date and time when the call ends


private Long callConnectTime; // date and time when the call was connected


private Integer startOfVoiceOffset; // offset in milliseconds from the
connect time


private Integer firstPromptOffset; // offset in milliseconds of the start
of play of the first prompt


private String mediaServerName; // name of the media server


private int channelType; // the type of communication channel used -
email, voice, or sms


private String address; // address used for the attempt
```

Creating a custom application class for Custom action

While creating a campaign strategy, if you want to use custom implementation for a given campaign, you can use the Custom action node instead of using the standard call, SMS, or e-mail action node. This custom action uses the Application node and you need to specify the custom application class or file name under the Application node.

Before you begin

Ensure you have the following files:

- `avaya-pim-pomapi.jar` file.
- `AgentAPIClient.jar` file.

.

Procedure

1. Copy the jar files in your development environment.
   You can find the `avaya-pim-pomapi.jar` file in the `$POM_HOME/lib/common` folder and the `AgentAPIClient.jar` file in the `$POM_HOME/lib` folder.

   Note:

   If you are using a custom class, then copy the `AgentAPIClient.jar` file from the `$POM_HOME/lib` folder to the `$POM_HOME/lib/custom` folder and restart POM service by typing `/sbin/service POM restart`.

2. The custom class must implement the following`PomActionProcessor` interface:

```
public interface PomActionProcessor {


public boolean processContact (PomInfo pomInfo, String parentStateName);

}
```

   where `pomInfo` is an object which contains the information of the contact to be processed, and `parentStateName` is the name of the state under which the Custom action node is added in the campaign strategy.

3. Call the processContact function for every contact in the campaign.

```
// class PomInfo with follwing members

JobId // ID of the job
contactId // User Contact ID
contactGroupId // ID of the Contact Group
contactGroupName // Name of the Contact Group
```

```
address // Address used in the Contact Attempt
campaignName // name of the campaign
```

Example

A sample class file:

```
public class TestActionImplementation implements PomActionProcessor {


@Override

public boolean processContact(PomInfo pomInfo) {

System.out.println("Inside custom action");

System.out.println("User Contact ID - " + pomInfo.getContactId());

System.out.println("User Contact Group Name - " + pomInfo.getContactGroupName());

System.out.println("Address - " + pomInfo.getAddress());

return true;

}


}
```

Creating a custom application class for custom application node

While creating a campaign strategy, if you want to use custom implementation for a campaign, you can use the custom application node for all other values except for Answer Human, Answer Machine, Call Answered, and Fax Machine. Use the Result node and specify the custom application class or file name under the Application node.

The custom class must implement the following `PomResultProcessor` interface:

```
    New interface PomResultProcessor // This will have a function with following
signature

public boolean processResult (PomInfo pomInfo, String result);
```

where `pomInfo` is an object which has the information of all the contacts to be processed, and `result` is the value of the result for which the custom application is added in the campaign strategy.

```
// class PomInfo with following members

JobId // ID of the job
contactId // User Contact ID
contactGroupId // ID of the Contact Group
contactGroupName // Name of the Contact Group
```

```
address // Address used in the Contact Attempt
campaignName // name of the campaign
```

Before you begin

Ensure you have the following files:

- `avaya-pim-pomapi.jar` file.
- `AgentAPIClient.jar` file.

Procedure

1. Copy the jar files in your development environment.

   You can find the `avaya-pim-pomapi.jar` file in the `$POM_HOME/lib/common` folder and the `AgentAPIClient.jar` file in the `$POM_HOME/lib` folder.

2. If you are using a custom class, then copy the `AgentAPIClient.jar` file from the `$POM_HOME/lib` folder to the `$POM_HOME/lib/custom` folder and restart POM service by typing `/sbin/service POM restart` on the command line.

Example

```
public class TestResultProcessor implements PomResultProcessor {

@Override

public boolean processResult(PomInfo pomInfo, String result) {

System.out.println("Inside custom result processing for " + result);

System.out.println("User Contact ID - " + pomInfo.getContactId());

System.out.println("User Contact Group Name - " + pomInfo.getContactGroupName());

System.out.println("Address - " + pomInfo.getAddress());

return true;

}

}
```

Creating a customer connector class for publish

Use to select the custom processor class that POM invokes for each completion code.

Procedure

1. Create a client program to connect to ActiveMQ on POM server.

2. Run the client program by typing `$POM_HOME/bin/testResProcClient.sh`.

Example

The sample client program is divided into 3 subsections:

- Connecting to JMS
- Receiving messages
- Exporting contacts to a file

Connecting to JMS:

```
private void connectToJMS(String brokerIpAddress,String port)
{
try
{
if(brokerIpAddress == null)
{
msgConsumer = null;
return;
}
// Create a ConnectionFactory
ActiveMQConnectionFactory connectionFactory = new
ActiveMQConnectionFactory( "tcp://" + brokerIpAddress + ":" + port);
// Create a Connection
connection = connectionFactory.createConnection();
connection.start();
connection.setExceptionListener(this);
// Create a Session
session = connection.createSession(false, Session.AUTO_ACKNOWLEDGE);
// Create the destination (Topic or Queue)
Destination destination = session.createTopic("pom.attempt.result.topic");
// Create a MessageConsumer from the Session to the Topic or Queue
msgConsumer = session.createConsumer(destination);
} catch (Exception e)
{
tracer.LogAndPrintStackTrace(e);
}
}
```

Receiving messages:

```
public void receiveMessage()
{
try
{
if(msgConsumer == null)
{
connectToJMS(ActiveMQBO.getMasterActiveMQAddress(), this.brokerPort);
}
if(msgConsumer == null)
{
Thread.sleep(1000);
System.out.println("Cannot open connection!!");
return;
}
Message message = msgConsumer.receive(1000);
if (message == null)
{
if(ctr > 30)
{
System.out.println("Waiting for Attempt Result Message..."+ "(" + msgCount +")");
```

```
tracer.finest("Waiting for Attempt Result Message...");
ctr = 0;
}
ctr++;
Thread.sleep(10);
return;
}
else if (message instanceof ObjectMessage)
{
ObjectMessage objMessage = (ObjectMessage) message;
PomResultInfo pomResultInfo = (PomResultInfo)objMessage.getObject();
msgCount++;
exportContactToFile(pomResultInfo);
}
else
{
tracer.finest("Received Unexpected Message Type: " + message);
}
}
```

Exporting contacts to a file:

```
public void exportContactToFile(PomResultInfo pomResultInfo)
{
openExportFile(pomResultInfo.getCampaignName(),pomResultInfo.getJobId());
try
{
if (exportContactCSV.length() == 0)
{
tracer.finest("Writing headerline for file : " + exportContactCSV);
exportWriteBuffer.write(headerLine);
exportWriteBuffer.flush();
}
exportContactData.setLength(0);
exportContactData.append(pomResultInfo.getContactId() + ",")
.append(pomResultInfo.getPimSessionId() + ",")
.append(pomResultInfo.getAddress() + ",")
.append(pomResultInfo.getContactGroupName() + ",")
.append(pomResultInfo.getZoneName() + ",")
.append(pomResultInfo.getResult() + ",").append("\n");
exportWriteBuffer.write(exportContactData.toString());
exportWriteBuffer.flush();
} catch (IOException e)
{
tracer.LogAndPrintStackTrace(e);
} finally
{
closeWriter();
}
}
```

Chapter 5: Agent Desktop Customization

Agent Desktop API

Introduction to Avaya Proactive Outreach Manager desktop API

An Application Programming Interface (API) defines the way different software components can integrate with each other. Using POM, you can integrate the agent functionality by designing a set of procedures or routines and manage the agents using a desktop.

The API's are divided into 2 main categories: Commands (requests) and Notifications and or Responses.

- Commands are requests sent by the agent desktop, that is, initiated by the agent. Examples include login, hold, unhold, transfer, and conference.
- Notifications and or Responses are events sent by POM to the agent desktop. Examples include call notifications, state change notifications, call state change notifications, agent capabilities or responses to the commands.

The requests sent through by the agent or the desktop can be asynchronous. Asynchronous events are when the system cannot or does not respond back immediately with the return values. Depending on the other activities running on the system, or the number of parameters or arguments for the command, the system might give back delayed responses. To accommodate the asynchronous behavior of the POM system, the APIs also demonstrate asynchronous behavior.

For example, an agent issues a hold request or command using the API AGTHoldCall(). POM sends back a response to the call when the command reaches POM. Meanwhile POM completes the Hold request and sends the actual response, that is, success or failure, in an asynchronous manner by invoking the callback function AGTHoldCallRESP.

Note:

POM Agent Manager is a POM component which takes care of all agent activities. It maintains the agent state machine. It interacts/controls various other components like the call pacer, nailer, router and blender to achieve state transitions.

The API's are structured such that the Agent State Machine is closely associated with POM, so that POM has complete control of the agent state, and the agent desktop acts as an interface

for the agent. This helps to minimize agent state handling within the desktop, and eliminate the need of two state machines – one for the desktop and the other in POM.

The desktop acts like a stateless interface and POM controls the desktop. POM defines desktop agent capabilities, such as CanHold, CanTransfer, CanConf, CanConsult.

At any point of time, POM provides these capabilities to the agent desktop via notifications so that various GUI controls can be enabled disabled accordingly. For example, when the agent performs a hold operation, POM sets the CanHold capability to false and enables CanUnhold. Similarly at various points in time the agent moves between Idle, Talking, Held, Consult, Conference, Wrapup call states.

POM notifies the desktop whenever the agent's call state changes. Similarly POM notifies the agent about the agent state such as Ready, NotReady, PendingNotReady.

POM also notifies the desktop about its nailing state such as NailedUp, PendingNailup, PendingNailupDrop, NailingLost, NotNailedUp.

Classes and interfaces

You can use the following classes and interfaces for the integration with the desktop:

- POMAgentFactory: Use to initialize the library.
- POMAgent: This is the agent object. You must get the POMAgent API from the Software Development Kit (SDK) and then use it for invoking commands.
- POMAgentHandlerInterface: You must implement the interface so that you can enable the notifications and responses to commands. POM uses the interface as a callback mechanism for responding to commands and for sending notifications.

API Components

The desktop APIs are available as a combination of 3 Dynamic Link Libraries (DLL)'s

- POMDesktopAPI.dll : The API's are available through the `POMDesktopAPI. dll`.
- Avaya.POM.Agent.ObjectModel.dll : The data structures used for the API's are available through the `Avaya.POM.Agent.ObjectModel.dll`.
- log4net.dll : The `log4net.dll` acts as a helper DLL for the API's.

Commands, Notifications, and Responses

List of commands and responses

Command	Usage	Response
AGTLogon	Used to Login to POM	AGTLogonRESP
AGTLogoff	Used to Logoff from POM	AGTLogoffRESP
AGTStateChange	Used to change agent state to Ready/NotReady/ PendingNotReady	AGTStateChangeRESP
AGTHoldCall	Used to put customer call on hold.	AGTHoldCallRESP
AGTUnholdCall	Used to unhold customer call.	AGTUnholdCallRESP
AGTReleaseLine	Agent can use this to disconnect customer call.	AGTReleaseLineRESP
AGTGetCompCodes	Returns the list of custom codes associated with the current campaign.	AGTGetCompCodesRESP
AGTWrapupContact	Wraps up the customer call.	AGTWrapupContactRESP
AGTExtendWrapup	Allows agent to get more time to wrapup call.	AGTExtendWrapupRESP
AGTGetConsultTypes	Returns the types (Agent/ Campaign) of consult supported for the current job.	AGTGetConsultTypesRESP
AGTGetConsultDestsForType	Returns the available destinations for the selected type of consult.	AGTGetConsultDestsForTypeRESP
AGTConsultCall	Used to consult another party.	AGTConsultCallRESP
AGTCompleteTransfer	Used to complete the transfer for the ongoing consult call.	AGTCompleteTransferRESP
AGTCancelConsult	Used to cancel an ongoing/ pending consult.	AGTCancelConsultRESP
AGTStartConf	To start a conference for an ongoing consult.	AGTStartConfRESP

Command	Usage	Response
AGTEndConf	Used by the agent to end the conference	AGTEndConfRESP
AGTConfChangeOwnership	Used by conference owner to transfer ownership to the passive agent.	AGTConfChangeOwnershipRESP
AGTRedial	Used by an agent to Redial the customer in Wrap-up mode	AGTRedialRESP
AGTSendDTMF	Used by an agent to send DTMF over the line.	AGTSendDTMFRESP
AGTGetCallbackTypes	Used by an agent to find out types of callbacks supported for the current job.	AGTGetCallbackTypesRESP
AGTGetCallbackDestsForType	Used by an agent to get a list of available destinations for the selected type of callback.	AGTGetCallbackDestsForTypeRESP
AGTCreateCallback	Used by an agent to create a callback.	AGTCreateCallbackRESP
AGTGetErrorString	Used by desktop to get error code details.	AGTGetErrorStringRESP
AGTPreviewDial	Used by agent to accept and dial the preview contact.	AGTPreviewDialRESP
AGTPreviewCancel	Used by agent to cancel the preview contact and move to wrapup state.	AGTPreviewCancelRESP
AGTGetCustomerDetails	Used by desktop to get details of the customer contact.	AGTGetCustomerDetailsRESP
AGTSetCustomerDetail	Used to set and attribute value of a contact.	AGTSetCustomerDetailRESP
AGTBlendToInbound	Sent by blender to POM informing it to move the agent to Inbound	AGTBlendToInboundRESP
AGTBlendToOutbound	Sent by blender to POM informing it to move the agent to Outbound	AGTBlendToOutboundRESP
AGTNailupAgent	Sent by desktop after moving the agent from Inbound to outbound.	AGTNailupAgentRESP
AGTReadyForNailup	Sent by desktop after processing AGTNailupChange – PendingNailup notitication.	AGTReadyForNailupRESP

Command	Usage	Response
AGTLostNailing	Sent by desktop if it detects that the nailing is lost.	AGTLostNailingRESP
AGTPendingLogout	Sent by desktop in error situations, so that POM can logout the agent once the current call is wrapped up.	AGTPendingLogoutRESP
AGTAddAgentNote	Used by desktop to add an agent note.	AGTAddAgentNoteRESP
AGTRefreshAgentNotes	Used by desktop to refresh agent notes and get the new ones.	AGTRefreshAgentNotesRESP
AGTGetTimezones	Used by desktop to find out supported timezones for callbacks.	AGTGetTimezonesRESP
AGTAvailableForNailup	Sent by desktop after the agent goes to Ready state for the first time after login.	AGTAvailableForNailupRESP
AGTAgentDisconnected	Sent by Proxy if it detects that the desktop is not reachable.	AGTAgentDisconnectedRESP
GetAgentStatusResponse	Desktop is expected to send this in response to GetAgentStatus notification	GetAgentStatusResponseRESP
AGTAddToDNC	Sent by desktop to add an agent to DNC list.	AGTAddToDNCRESP
AGTGetZoneList	Command to get current zones on POM.	AGTGetZoneListRESP
AGTIsInDNC	Sent by desktop to check if a contact address is in DNC list.	AGTIsInDNCRESP
AGTSaveAgentForHA	Desktop should send this after logging in so that POM will save the agent state if the desktop comes back up after crashing.	AGTSaveAgentForHARESP
AGTSkillsChanged	Sent by AACC via AAAD informing POM that the agents skill has been modified.	AGTSkillsChangedRESP
AGTGetContactAttributes	Sent by desktop to find out all attributes associated with a particular contact.	AGTGetContactAttributesRESP

List of notifications

Notification	Usage
AGTStateChangedNotify	Sent by POM when the agent state changes (Ready/ NotReady/PendingNotReady)
AGTCallNotify	Sent by POM when a new contact is sent to the desktop.
AGTAutoReleaseLine	Sent by POM when the customer disconnects the call.
AGTConsultNotify	Sent by POM informing an agent that it is in consult now.
AGTConsultCancelled	Sent by POM to an agent when the consult gets over.
AGTTransferNotify	Sent by POM to the passive agent when the consult gets transferred to it.
AGTConferenceNotify	Sent by POM to the passive agent when the consult gets into a conference..
AGTConferenceEnded	Sent by POM to the an agent when the conference is ended by the other agent.
AGTConferenceOwnershipChanged	Sent by POM to the passive agent when the conference owner transfers the conference ownership to the passive agent.
AGTCapabilitiesChanged	Sent by POM whenever it detects change in desktop capabilities based on agent/job/nailing states.
AGTNailupChange	Sent by POM whenever it detects change in nailing state of an agent.
AGTCallStateChangedNotify	Sent by POM whenever it detects change in call state of an agent.
AGTDialFailed	Sent by POM whenever it detects failure while dialing a contact for either a redial or a preview dial.
AGTConsultDialFailed	Sent by POM whenever it detects failure while dialing for a consult.
AGTConsultPending	Sent by POM informing a busy agent that a consult is pending.
AGTPendingConsultComplete	Sent by POM to the active agent when the pending consult gets into a consult.
POMAvailable	Internal notification.
POMNotAvailable	Internal notification.
AGTPreviewCallbackPending	Sent by POM when a callback is pending for an agent.

Notification	Usage
AGTAgentLoggedOut	Sent by POM when it detects that the agent has logged out.
AGTEnableCancelConsult	Sent by POM informing the desktop that it can enable the cancel consult operation.
AGTPreviewCallbackCancelled	Sent by POM informing the agent that the current callback on the agents desktop has been cancelled.
AGTInvalidCommandName	Sent by POM if it receives an unknown command.
AGTCustomerDetailsChanged	Sent by POM to a passive party in consult/conference if the owner changes any customer attribute value.
AGTBlendedToInbound	Sent by POM informing the agent that it has been blended to Inbound.
AGTBlendedToOutbound	Sent by POM informing the agent that it has been blended to Outbound.
AGTZoneDown	Sent by POM when a zone is marked DOWN for failover.
AGTJobAttached	Sent by POM whenever an agent is attached to a new job.

Agent Activities

Login and Logout

An agent has to login to POM for it to be considered for an outbound job. POM will provide APIs so that an agent can login to POM. Once logged in to POM the agent can also logout. It is expected that the agent will provide AgentID/AgentExtension along with the login request. POM will use this information to check the skillset of the agent. POM will check this information over AACC or CCElite. The agent will also have to provide the POM IP to which it wants to connect. Agents in POM will be associated with zones. So, it is also expected that the agent will provide the zone name in the login request.

POM will also have provision for an agent to force login itself to POM in situations where the desktop crashes. Once force logged in, the agent state on POM will be reset and it will be a fresh start for the agent.

API	Command/Response/Notification
AGTLogon	Command sent by desktop to POM

API	Command/Response/Notification
AGTLogonRESP	Asynchronous callback response to AGTLogon
AGTLogout	Command sent by desktop to POM
AGTLogoutRESP	Asynchronous callback response to AGTLogout

Agent state

An agent has to move to Ready state after logging-in so that it can receive outbound calls. Once the agent moves to ready state, POM will consider this agent for call pacing. An agent can also move to NotReady state while he is in Ready state. But, the agent has to be in Idle (not in a call) state for it to move from Ready to NotReady state.

If the agent tries to move to NotReady state while handling a call, POM will move the agent to PendingNotReady state. The agent will stay in PendingNotReady state until he wraps up the current call and also if it does not have any pending consults/callbacks. Once the agent is done wrapping up the current call and handling the pending requests (consult/callbacks) POM will move the agent to NotReady state from PendingNotReady state.

POM will notify the agent about its current state through the AGTStateChangedNotify notification.

Note:

An agent has to handle a maximum of an existing call, up-to 2 pending consult requests and one pending callback before POM will move it to NotReady state. If an agent issues a NotReady command when it does not have any pending requests (pending consults/pending callback), then the agent will be moved to NotReady state as soon as the agent wraps up the current call. If the agent is not in a call then it will be immediately moved to NotReady state.

Walk-away agent

The API also has a provision to handle walk-away agents. Let us assume that the agent has moved away from his desk while handling a call. On hearing nobody on the agent end, the customer will hang up the call. The call will be auto-wrapped up by the agent if the campaign has set a wrapup timer. POM is not aware that the agent has walked away from the desktop. POM will then handover the next outbound call to this agent. Once again the customer will hang up the call on hearing silence. Once again the call will get auto-wrapped up. To prevent such situations, it is recommend that the desktop detect such situations and automatically send AGTStateChange with the walkedAway flag set to true. POM will then move this agent to NotReady state and not consider this agent for call pacing. One of the ways that the desktop can determine if the agent has walked away is, if there has been no button clicks for two consecutive calls on the desktop by the agent.

Changing Aux Reason code

An agent can also change the aux reason code after going to NotReady state. The agent has to invoke AGTStateChange with new reason code.

API	Command/Response/Notification
AGTStateChange	Command sent by desktop to POM to change state. Same command can be used to move to either Ready or NotReady state.
AGTStateChangeRESP	Asynchronous callback response to AGTStateChange
AGTStateChangedNotify	Asynchronous notification sent by POM when the agent state changes.

Nailing

POM nails the agent if there is a job running which matches the skillset of the logged in and ready agent. When the agent logs in for the first time, it has to send AGTAvailableForNailup so that POM can consider it for nailing. Also, the agent has to be in ready state after that. POM will send a sequence of notifications along with flags indicating the next nailing action. If POM wants to nail an agent it will first send AGTNailupChange with flag PendingNailup. On processing this flag the desktop should send AGTReadyForNailup. Once POM received AGTReadyForNailup, POM will then attempt to send a nailing call to the agent. As soon as the agent picks up the nailing call, POM will once again send AGTNailupChange with flag NailedUp. If the agent is unable to pick-up the nailing call, then POM will send AGTNailupChange with flag NotNailedUp.

An agent will be un-nailed if the job to which the agent is attached gets over OR the agent tries to logout after moving to NotReady state from Ready state. POM will send AGTNailupChange with PendingNailupDrop flag. Once the nailing call gets disconnected then POM will send AGTNailupChange with flag NotNailedup.

If the nailup gets dropped by the desktop unexpectedly then the desktop should send AGTLostNailing. Also, if POM detects that the nailing got dropped unexpectedly then POM sends AGTNailupChange with the flag set to NailingLost.

API	Command/Response/Notification
AGTAvailableForNailup	Command sent by desktop to POM. This is an indication for POM that it can consider this agent for nailing.
AGTAvailableForNailup RESP	Asynchronous callback response to *AGTAvailableForNailup*
AGTNailupChange	Asynchronous notification sent by POM when the nailing state changes.
AGTReadyForNailup	Command sent by desktop to POM on receiving AGTNailupChange – PendingNailup.

API	Command/Response/Notification
AGTReadyForNailupRESP	Asynchronous callback response to AGTReadyForNailup
AGTLostNailing	Command sent by desktop when it detects that the nailing got lost unexpectedly (i.e. nailing was not intentionally dropped by POM)
AGTLostNailingRESP	Asynchronous callback response to AGTLostNailing

New call notification

Predictive/Progressive Campaigns: In case of Predictive and Progressive campaigns, the customer is dialed first and then connected to the agent. POM sends a notification to the agent whenever the customer call is connected to the agent so that the agent has enough information about the customer. In the new call notification POM sends the agent script URL, campaign name, skillset and basic information about the customer.

API	Command/Response/Notification
AGTCallNotify	Asynchronous notification sent by POM when the customer call is connected to the agent.

Preview Campaigns: In case of Preview campaigns, the new call notification is sent to the agent first. The customer is dialed only if the agent decides to talk to the customer (AGTPreviewDial command). The agent can decide to cancel the request (AGTPreviewCancel command) and in this case, the agent has to provide a completion code.

POM supports timed and non-timed preview campaigns. In the case of timed preview, POM will send a time value along with AGTCallNotify. This time value is the time given to the agent to decide on dialing/cancelling the preview. If the agent does not make a decision within the time value, then it is expected that the desktop should send AGTPreviewDial automatically after the timer expires. In the case of non-timed preview the agent has infinite time to make a decision on accepting the preview or rejecting it.

If the dial attempt fails, the POM will notify the desktop by sending a notification AGTDialFailed along with the reason code.

API	Command/Response/Notification
AGTCallNotify	Asynchronous notification sent by POM when the customer call is connected to the agent.
AGTPreviewDial	Command sent by agent when it wants to accept the preview request and dial the customer OR supposed to be sent by the desktop on preview timer expiry.
AGTPreviewDialRESP	Asynchronous callback response to AGTPreviewDial

API	Command/Response/Notification
AGTPreviewCancel	Command sent by agent when it wants to reject the preview request.
AGTPreviewCancelRESP	Asynchronous callback response to AGTPreviewCancel
AGTDialFailed	Asynchronous notification sent by POM if the customer call dial attempt fails.

Customer data

Desktop should invoke AGTGetCustomerDetails to get all the contact details. POM will then send back all system attribute values like Title, First Name, Last Name and so on. Also, POM will send back all custom attributes. Along with each attribute, POM also sends the type of the attribute. The attribute types supported by POM are: READ_ONLY, WRITE, SCREEN_POP, MASKED_WRITE. It is expected that the desktop honor these types. It is expected that the desktop invoke AGTGetCustomerDetails after receving AGTCallNotify notification.

Desktop can also modify the value of the customers attribute by using AGTSetCustomerDetail. Only one attribute at a time is allowed. If the attribute is of READY_ONLY type POM will send back error.

API	Command/Response/Notification
AGTGetCustomerDetails	Command sent by agent when it wants to get all the customer details.
AGTGetCustomerDetailsRESP	Asynchronous callback response sent by POM in response to AGTGetCustomerDetails.
AGTSetCustomerDetail	Command sent by agent to change a customer attribute.
AGTSetCustomerDetailRESP	Asynchronous callback response sent by POM in response to AGTSetCustomerDetail.

Agent capabilities

The agent capabilities are controlled by POM. Capabilities control the various actions that an agent can perform at any given point of time. For more information about the capability matrix, see Capability Matrix on page 162 POM will send AGTCapabilitiesChanged notification whenever the agents capabilities change. Example: The capabilities control whether an agent can hold/consult an agent while talking to the customer. There are approximately 17 capabilities. This allows the desktop developer to enable, disable, hide, or unhide buttons on the desktop based on these values. The desktop does not have to maintain its own state machine. POM maintains that internally and the capabilities are a reflection of the current agent

state on POM. It is expected that the desktop developer adhere to these capabilities sent from POM.

API	Command/Response/Notification
AGTCapabilitiesChanged	Asynchronous notification sent by POM whenever it detects change in agent desktop capabilities.

Call state

The desktop will be notified from time to time about the current call state by POM. This information can be used by the desktop to inform the agent about its current call state according to POM. Some of the call states are: NoState, Preview, Idle, Dialing, Talking, Held, Wrapup, Consult, ConferenceOwner, ConferencePassive, Callback.

API	Command/Response/Notification
AGTCallStateChangedNotify	Asynchronous notification sent by POM whenever POM detects change in call state.

DNC

An agent can add a contact address to the DNC list. Desktop should invoke AGTAddToDNC to add a contact address to the DNC list.

API	Command/Response/Notification
AGTAddToDNC	Command sent by desktop to add a contact address to the contact list.
AGTAddToDNCRESP	Asynchronous callback response for AGTAddToDNC.

Hold and unhold

An agent can put the customer on hold if required. The customer will hear hold music (depending on the configuration) when the agent puts the customer on hold. Once the customer is on hold, the agent can unhold the customer call.

API	Command/Response/Notification
AGTHoldCall	Command sent by desktop/agent to put the customer call on hold.
AGTHoldCallRESP	Asynchronous callback response for AGTHoldCall.

API	Command/Response/Notification
AGTUnholdCall	Command sent by desktop/agent to remove the customer from hold state back to talking state.
AGTUnholdCallRESP	Asynchronous callback response for AGTUnholdCall.

Send DTMF

An agent can send DTMF while talking to a customer. This feature is useful for situations wherein the agent is talking a answering machine OR in case of a consult/conference the consulted party is an IVR system. The agent can send one digit at a time.

For detecting Inband DTMF coming to MPP (IVR) you need to turn on inband detection under **MPP Servers > VoIP Settings**, and set "Inband DTMF Detection Enabled" to "Yes". For hearing Inband DTMF make sure you have configured your gateway (CM) to send/receive Inband DTMF.

API	Command/Response/Notification
AGTSendDTMF	Command sent by desktop/agent to send DTMF.
AGTSendDTMFRESP	Asynchronous callback response for AGTSendDTMF

Consult

An agent can consult another party while talking to the customer. POM supports two types of consults: Agent, External. Consult if the first step required before attempting a transfer or a conference. The agent has to decide what the consult could result in and accordingly invoke AGTGetConsultTypes. After getting the supported types from POM, the agent has to then invoke AGTGetConsultDestsForType for a particular type of Consult. In the case of Agent type of consult POM will send back a list of agents that belong to the same skillset as the consulting agent. If the agent selects External type of consult then, POM will send back the address list configured for the campaign. The agent can also enter free form numbers (if permitted by POM configuration) for performing an Agent/External type of consult. The agent has to use AGTConsultCall command request to initiate the consult. The other agent, which is being consulted gets a notification AGTConsultNotify when the consult gets started between the two agents. POM supports only one consult at a time for an agent.

- Pending Consult: If Agent A tries to consult agent B, while agent B is already busy talking to a customer, then the consult request from Agent A to Agent B is sent as a pending consult (AGTConsultPending notification) request to Agent B. Agent A can cancel this pending consult request if required, but Agent B cannot cancel the pending request. POM

will send a maximum of two pending consult requests to any agent, if the agent is already in a call with the customer.

- Agents in Consult: The consult initiating agent is considered as the Active agent, while the consulted agent is considered as a Passive agent by POM. Once both the agents are in consultation with each other, the customer will hear music on hold (if configured on POM).
- Cancelling/Leaving a consult: Once the agents are in consultation with each other, either of them can cancel the consult. They have to use AGTCancelConsult command request to cancel the consult. If the Active agent cancels the consult then the passive agent will receive AGTConsultCancelled notification. If the Passive agent cancels the consult then the active agent will receive AGTConsultCancelled notification from POM.
- External Consult: If an agent attempts to consult an external agent and if the dial attempt fails then POM will send back a notification to this agent AGTConsultDialFailed with appropriate reason.
- Customer: The customer cannot be dropped directly while in consult. The consult has to be ended before the customer can be dropped from the call. Ending a consult just removes the passive agent from the consult. The customer call still stays active with the active agent. But the customer can drop the call at any point of time. If the customer drops the call, POM will drop the consult and put the active agent in wrapup mode, while the passive agent is moved to Idle state.

API	Command/Response/Notification
AGTGetConsultTypes	Command sent by desktop to know consult types supported by POM.
AGTGetConsultTypesRESP	Asynchronous callback response for AGTGetConsultTypes
AGTGetConsultDestsForType	Command sent by desktop to get a list of agents which can be consulted for a particular type of transfer or conference.
AGTGetConsultDestsForTypeRESP	Asynchronous callback response for AGTGetConsultDestsForType
AGTConsultCall	Command sent by desktop/agent to initiate a consult.
AGTConsultCallRESP	Asynchronous callback response for AGTConsultCall
AGTCancelConsult	Command sent by desktop/agent to cancel a consult.
AGTCancelConsultRESP	Asynchronous callback response for AGTCancelConsult
AGTConsultNotify	Asynchronous notification sent by POM whenever a passive agent gets into a consult with an active agent.

API	Command/Response/Notification
AGTConsultPending	Asynchronous notification sent by POM whenever a passive agent is busy handling other calls.
AGTConsultCancelled	Asynchronous notification sent by POM to the other agent whenever an agent cancels an ongoing consult.
AGTConsultDialFailed	Asynchronous notification sent by POM whenever a consult attempt fails.

POM system plays a beep sound in following ways:

- When the agent clicks **Consult**, both the consult recipient and owner hear beep. The system does not play a beep to the contact.
- When the consult owner agent clicks **Cancel**, and the consult is established, both the consult recipient and owner hear beep. The system does not play a beep to the contact.
- When the consult recipient agent clicks **Cancel**, and the consult is established, the consult recipient, consult owner, and the contact hear the beep.
- When the agent clicks **Transfer** in the consult window, and the consult is established, the transfer recipient, transfer initiator, and the contact hear the beep.
- When the agent clicks **End Conference**, or **Leave Conference**, both the agents and the contact hear the beep.
- When the agent clicks **Changed Ownership** when the conference is going on, everyone in the conference, the agents and the contact hear the beep.
- When the contact disconnects the call while the conference is going on, both the agents hear the beep. The system does not play a beep to the contact.
- When the agent and the contact are talking, if the contact disconnects the call, the agent hears the beep.

Transfer

An agent (A) can transfer the current customer call to another agent (B). But first agent (A) has to enter a consult with the other agent (B) before attempting a transfer. As of now POM does not support external type of transfer (because of third party issues). If an agent (A) transfers a call to the other consulted agent (B), POM will send AGTTransferNotify notification to the other agent (B) and handover the call control to the other agent (B). So, this means that the call is transferred from the Active consulting agent (A) to the Passive consulted agent (B). The first agent (A)(earlier Active agent) is removed from the call. So at the end of the transfer the consulted (B) (earlier Passive agent) becomes the owner of the customer call. The earlier Active agent (A) is free to accept another call from POM and it cannot provide disposition for the call.

To perform a transfer an agent has to invoke AGTCompleteTransfer command. After the transfer is complete the agent is moved out of hold state and is now able to talk to the agent (B).

Note:

POM does not support external transfer on non-SIP endpoints.

API	Command/Response/Notification
AGTCompleteTransfer	Command sent by desktop/agent to initiate a transfer.
AGTCompleteTransferRESP	Asynchronous callback response for AGTCompleteTransfer.
AGTTransferNotify	Asynchronous notification sent by POM to the consulted agent when the transfer gets completed.

Callbacks

An agent can schedule a callback, while in a preview or while talking to a customer or during wrapup. An agent has to first find out the types of callbacks that are supported for that campaign. As of now POM supports the following types of callbacks: Standard, Agent and Campaign. The agent has to invoke AGTGetCallbackTypes command to find out the supported callback types. After selecting the callback type the agent has to invoke AGTGetCallbackDestsForType command to get a list of destinations supported for a particular type of callback. In order to create a callback the agent has to invoke AGTCreateCallback. The agent can specify a free form number as well while creating a callback.

Types of callback: A Standard type of callback means that any agent which has the same skillset as the callback creator could get the callback when the callback time arrives. An Agent type of callback means that the preferred agent for the callback (when the timer expires) is the callback creator, but if this agent is not available at that point of time, POM will hand over the callback to any agent which has the same skillset as the callback creator. A Campaign type of callback means that the callback will be handed over to any agent which matches the skillset of the selected campaign.

Callback Time: When the scheduled callback time arrives, POM will select an agent depending on the type of callback created. Once the agent is selected POM will send a pending callback notification AGTPreviewCallbackPending to the agent few seconds before the scheduled callback time (depending on the POM configuration). This is to help the agent be aware of a pending callback request. Once the pre-timer expires the agent can decide on an action for the callback. A callback is like a preview call for the agent. The agent can decide to cancel the callback, or reschedule it or accept it. If the agent accepts the preview, it has to invoke AGTPreviewDial command. If the dial fails POM will send back a notification AGTPreviewDial failed with an appropriate error code. If the agent decides to reject the pending callback, then it has to invoke AGTPreviewCancel command and then wrapup the call by providing an

appropriate completion code. The agent can reschedule the callback as well by invoking AGTCreateCallback once again.

Callback expiry: If an agent receives a pending callback request and if by the time the agent accepts the callback, the callback expires. Then POM cancels the callback and sends a notification AGTPendingCallbackCancelled to this agent.

API	Command/Response/Notification
AGTGetCallbackTypes	Command sent by desktop/agent to find out types of callbacks supported by the current campaign.
AGTGetCallbackTypesRESP	Asynchronous callback response for AGTGetCallbackTypes.
AGTGetCallbackDestsForType	Command sent by desktop/agent to find out destinations allowed for a selected type of callback.
AGTGetCallbackDestsForType RESP	Asynchronous callback response for AGTGetCallbackDestsForType.
AGTCreateCallback	Command sent by desktop/agent to create a callback.
AGTCreateCallbackRESP	Asynchronous callback response for AGTCreateCallback.
AGTPreviewDial	Command sent by desktop/agent to dial the customer after receiving a pending callback.
AGTPreviewDialRESP	Asynchronous callback response for AGTPreviewDial.
AGTPreviewCancel	Command sent by desktop/agent to cancel the pending callback request.
AGTPreviewCancelRESP	Asynchronous callback response for AGTPreviewCancel.
AGTPreviewCallbackPending	Asynchronous notification sent by POM to the selected agent who will receive the callback.
AGTPreviewCallbackCancelled	Asynchronous notification sent by POM to the selected agent for the pending callback, when the callback expires.

Conference

An agent (A) can conference the current customer call to another agent (B). But first agent (A) has to enter a consult with the other agent (B) before attempting a conference. POM supports only 3 party conference (i.e. one customer and 2 agents). If an agent (A) conferences a call to the other consulted agent (B), POM will send AGTConferenceNotify notification to the other agent (B) and add the Passive Agent (B) in a conference with the customer.

In order to start a conference agent (A) has to invoke AGTStartConf command. As soon as the conference is completed, the customer is moved from hold state to in-conference with both agents.

Ending the conference: Both agents can get themselves out of the conference albeit in a different way. The agent (A) who initiated the conference is considered as the conference owner, while the consulted agent (B) is considered as the passive party in the conference. If conference owner agent (A) wants to drop the conference, it has to invoke AGTEndConf command. In this case POM will send drop the passive agent (B) from the conference and send a notification AGTConferenceEnded to the passive agent (B). If the passive agent (B) wants to leave the conference, then it has to also invoke AGTEndConf command. In this case POM sends a notification AGTConferenceEnded to the conference owner.

Customer: The customer cannot be dropped directly while in conference. The conference has to be ended before the customer can be dropped from the call. Ending a conference just removes the passive agent from the conference. The customer call still stays active with the active agent. But the customer can drop the call at any point of time. If the customer drops the call, POM will drop the conference and put the conference owner agent in wrapup mode, while the passive agent is moved to Idle state.

Conference Ownership: The conference owner agent (A) can transfer the conference ownership to the passive agent (B) once the conference is ON. To transfer the conference ownership from the conference owner agent (A) to passive agent (B), agent (A) has to invoke AGTConfChangeOwnership command. Once the ownership is transferred to the passive agent, the passive agent will receive a notification from POM AGTConferenceOwnershipChanged. Once the ownership is transferred, the conference owner agent (A) now becomes the passive agent in the conference, while the earlier passive agent (B) now becomes the conference owner.

API	Command/Response/Notification
AGTStartConf	Command sent by desktop/agent to initiate the conference.
AGTStartConfRESP	Asynchronous callback response for AGTStartConf
AGTEndConf	Command sent by an agent in a conference to end the conference.
AGTEndConfRESP	Asynchronous callback response for AGTEndConf.
AGTConferenceNotify	Asynchronous notification sent by POM to the passive agnet in the conference while has moved from consult to conference state.
AGTConferenceEnded	Asynchronous notification sent by POM to the conference owner if the passive agent leaves the conference. POM can also send this notification to the passive agent in the conference if the conference owner ends the conference.

API	Command/Response/Notification
AGTConfChangeOwnership	Command sent by conference owner desktop/agent to transfer the conference ownership to the passive agent in the conference.
AGTConfChangeOwnershipRESP	Asynchronous callback response for AGTConfChangeOwnership.
AGTConfOwnershipChanged	Asynchronous notification sent by POM to the passive agent when the conference owner transfers the conference ownership to it.

Wrapup

Redial: An agent can redial the customer while in wrapup mode. The agent may want to perform this action if the customer call got dropped by an error or for other situations. The agent has to invoke AGTRedial command to perform a redial to the customer. The agent can select the number to dial from the available contact phone addresses.

Extending Wrapup: If a wrapup time is configured for the campaign, then it is expected that when the configured wrapup time expires the desktop will auto-wrapup the call with the default completion code supplied with the release response/notification. But if required the agent can ask for an extension in wrapup time. Based on the configuration in the POM campaign, an agent can have multiple extensions.

Wrapping up: An agent has to send a completion code for wrapping up the call. An agent can invoke AGTGetCompCodes command to find out the list of available completion codes for the current job.

API	Command/Response/Notification
AGTRedial	Command sent by desktop/agent to redial the customer.
AGTRedial RESP	Asynchronous callback response for AGTRedial.
AGTGetCompCodes	Command sent by desktop/agent to find out the list of completion code.
AGTGetCompCodesRESP	Asynchronous callback response for AGTGetCompCodes.
AGTExtendWrapup	Command sent by desktop/agent to get an extension time in wrapup mode.
AGTExtendWrapup RESP	Asynchronous callback response for AGTExtendWrapup.
AGTWrapupContact	Command sent by desktop/agent to wrapup the call.
AGTWrapupContact RESP	Asynchronous callback response for AGTWrapupContact.

Agent notes

An agent can create some notes and save them while talking to the customer. These notes are particularly useful in case of consult/conference/transfer. The agent has to invoke AGTAddAgentNote to add a note. To get the latest set of notes the agent has to invoke AGTRefreshAgentNotes. An agent can also create agent notes while creating a callback. So, when the agent invokes AGTRefreshAgentNotes, POM will send back all the agent notes along with the callback notes.

API	Command/Response/Notification
AGTAddAgentNote	Command sent by desktop/agent to add a note.
AGTAddAgentNote RESP	Asynchronous callback response for AGTAddAgentNote.
AGTRefreshAgentNotes	Command sent by desktop/agent to get all the saved agent notes for the current call.
AGTRefreshAgentNotesRESP	Asynchronous callback response for AGTRefreshAgentNotes.

Error

Asynchronous responses of the commands could return error codes. The desktop should invoke AGTGetErrorString to get the actual error message.

Note:

POM will not take care of localization related to error messages.

API	Command/Response/Notification
AGTGetErrorString	Command sent by desktop/agent to error details.
AGTGetErrorStringRESP	Asynchronous callback response for AGTGetErrorString.

Client

After client is down

When the client goes down, the network connection between the client and POM server will break and POM will now be aware that the client has gone down. In such situations the agent manager process on POM (server) will perform the following activities:

It will queue up all pending responses/notifications and wait for the client to come up.

After client is up again

As soon as the client comes up, agent manager and the client will perform the following activities:

1. Agent manager will request the agent status (GetAgentStatus) of each agent which was managed by the client.
2. If the client has the current state of each agent (GetAgentStatusResponse), then agent manager will try to restore and bring the agent desktop and POM in sync.
3. If the client does not have the agent status, POM will reset the agent state and now the agent will have to re-login.

Server

Server goes down

As soon as the client detects that the server (agent manager on POM) has gone down, it will perform the following activities:

1. The client will send back an error response POM_NOT_AVAILABLE for each new command request from the desktop.
2. Also, the client will keep on trying to connect to the server every few seconds.

Note:

If the server goes down before sending back a response to a command sent by the client, then the client will not auto generate POM_NOT_AVAILABLE error for this command.

Server is up again

1. As soon as the server comes up, the client will be able to connect and then the activities mentioned in the section After client is up again will be performed.
2. If the agent states are in sync between the client and server, then agent manager will send all queued responses and notifications.

Insert content for the first section.

Zones

POM will support zoning. So it is mandatory for an agent to tell POM which zone it belongs to. The login command will be modified to take care of this. The login command AGTLogon will also take as input the zone name.

On the POM end, the user can configure a CCElite/AACC server for each zone. Also, the agent manager process on POM can be managing one or more zones. So, the client could possibly make multiple socket connections to the agent manager, one for each zone. If the agent managers are located on different POM servers then the socket connections will be made accordingly.

HA with Zones

If a zone (say ZONE #1) goes down, then the agent manager process in the other zone (say ZONE #2) will automatically take over the responsibilities of the agent manager from the earlier zone (ZONE #1). In such scenario it is expected that the agent will have to re-login to POM and the client library will automatically connect to the agent manager which now manages Zone #1.

There will be a configured agent manager managing a zone. So, if that agent manager goes down, then any other agent manager could take up responsibilities of this zone. But, if the systems from Zone #1 are up again after some time, then the designated agent manager will take back the responsibilities from the temporarily managing zone agent manager. This will be seamlessly managed automatically by the client libraries with support from agent manager.

POM Agent factory methods and commands

Boolean init(PAMSocketInfo[] pamAddresses)

The desktop developer should invoke this to initialize the library. This is the first API call that should be made before using the library.

pamAddress List of IP address and corresponding port of POM agent managers.

deinit()

The desktop developer should invoke this when the desktop us shutting down to clean-up the resources used by the POM Desktop library.

POMAgent getPOMAgent(String id, POMAgentHandlerInterface handler)

The desktop developer should invoke this to get an agent object. This agent object can then be used to send commands for this agent. POM in turn would invoke the handler for the asynchronous responses and notifications. Please ensure that a valid POMAgent object is used before making any command request.

id	Agent ID
handler	Callback handler object.

Boolean removePOMAgent(String id)

The desktop developer should invoke this to when it is done using the POMAgent object which was obtained with getPOMAgent. Please ensure that this is invoked after processing the response for AGTLogoff command.

id	Agent ID.

Commands

Actual asynchronous response for each command is available from POM in the corresponding command”RESP” method. Each command also has a int return type. Each command request can return one of the following error code based on the error:

Error Code: 9999 (POM_NOT_AVAILABLE)

This will be sent back by the library if any command could not be sent to POM. This will generated internally and sent back with the appropriate response RESP.

Error Code: 9998 (PAM_NOT_AVAILABLE_FOR_ZONE)

This will be sent back by the library if an agent manager is not found managing the zone provided in the AGTLogon command. This is sent only for AGTLogon command.

Error Code: 9997 (INVALID_ARGUMENT)

This will be sent back by the library if any argument provided to the API is not found right. This error can be sent by any command.

Error Code 9996 (SDK_Failure)

If the library is unable to send the command to POM, then this error code is generated. This error can be sent by any command.

int AGTLogon (String agentExt, String pwd, boolean isForce, String locale, String timeZone, String zoneName)

agentExt This is the agent extension. This is the SIP URL that is used by POM to send the nailing call in AACC mode.

pwd Password used for authenticating the agent on AACC or CC Elite.

isForce This is a flag which can be used by the desktop to allow a forceful login to POM. It is set to true when the desktop wants to allow this agent to be able to login in the following scenarios:

- - even if this agent has already logged in from a different desktop.

  the desktop/agent PC crashed while the agent was already logged in.

locale Locale of the agent.

timeZone Timezone of the agent.

Note:

POM restricts only Null values of locale and timeZone during login.

zoneName Zone of the agent.

Description

The agent sends this command from the desktop when the agent wants to login to POM. POM can then use this agent for a campaign when the agent becomes ready.

Error codes

Error code	Error message	Description
2	This agent is not registered with the system	POM system does not recognize the logged in agent. You can force login again.
6	Unable to verify password	You can see this message only if you POM integration with CC Elite. POM is unable to get the password from AES. Check the password of the agent and check the AES Web service. .
7	Agent is already logged in	POM displays the message if POM thinks the agent has already logged in. You can force login again.
9	Agent skills not found	Check the agent ID. The agent ID must match the agent ID you specify in Contact Center. If the error message persists, check the AES or AACC Web services.

Error code	Error message	Description
10	Unable to change the state of the agent	POM is unable to change the agent state. Login again to the POM system.
11	Internal error. Unable to login agent	The agent cannot login to the POM system. Login again to the POM system.
12	Login failure. Zone not found	The agent cannot login due to zone error. Check the zone for which the agent logs in. The zone must match one of the zones specified in POM.
13	Login failure. Invalid locale	The agent cannot login due to locale error. Check the locale of the agent. The locale should not be Null. If Null, specify a locale for the agent. **Note:** POM checks only Null value for locale values, and does not restrict any other string value
15	Login failure. Invalid Timezone	The agent cannot login due to timezone error. Check the timezone of the agent. The timezone should not be Null. If Null, specify timezone of the agent. **Note:** POM checks only Null value for time zone values, and does not restrict any other string value
16	Login failure. Invalid Agent Name	You can see this message only if you have POM integration with AACC. The agent name cannot be Null. You must specify a value for agent name.
17	Login failure. Authentication of agent failed.	You will see this message only if you have POM integration with CC Elite. Check the agent password. The agent password must match the agent password you specify in the Contact Center.

Related topics:

AGTLogonRESP(int result) on page 117

AGTLogonRESP(int result)

Syntax

```
int result
```

Return values

result: “0” indicates success.

int AGTLogon(String agentExt, String pwd, boolean isForce, String locale, String timeZone, String zoneName, String agentName, POMAgentSkill[] agentSkills)

agentExt This is the agent extension. This is the SIP URL that is used by POM to send the nailing call in AACC mode.

pwd Password used for authenticating the agent on AACC/CCElite.

isForce This is a flag which can be used by the desktop to allow a forceful login to POM. It is set to true when the desktop wants to allow this agent to be able to login in the following scenarios:

- - even if this agent has already logged in from a different desktop.

the desktop/agent PC crashed while the agent was already logged in.

locale Locale of the agent.

timeZone Timezone of the agent.

Note:

POM restricts only Null values of locale and timeZone during login.

zoneName Zone of the agent.

agentName Displays the name of the agent. The Agent Manager references the agent name while integrating with AAAD.

agentSkills The agent skillset array to be sent by AAAD.

Description

The agent sends this command from the desktop when the agent wants to login to POM. POM can then use this agent for a campaign when the agent becomes ready.

Error codes

Error code	Error message	Description
2	This agent is not registered with the system	POM system does not recognize the logged in agent. You can force login again.
6	Unable to verify password	You can see this message only if you POM integration with CC Elite. POM is unable to get the password from AES. Check the password of the agent and check the AES Web service. .
7	Agent is already logged in	POM displays the message if POM thinks the agent has already logged in. You can force login again.

Error code	Error message	Description
9	Agent skills not found	Check the agent ID. The agent ID must match the agent ID you specify in Contact Center. If the error message persists, check the AES or AACC Web services.
10	Unable to change the state of the agent	POM is unable to change the agent state. Login again to the POM system.
11	Internal error. Unable to login agent	The agent cannot login to the POM system. Login again to the POM system.
12	Login failure. Zone not found	The agent cannot login due to zone error. Check the zone for which the agent logs in. The zone must match one of the zones specified in POM.
13	Login failure. Invalid locale	The agent cannot login due to locale error. Check the locale of the agent. The locale should not be Null. If Null, specify a locale for the agent. **Note:** POM checks only Null value for locale values, and does not restrict any other string value
15	Login failure. Invalid Timezone	The agent cannot login due to timezone error. Check the timezone of the agent. The timezone should not be Null. If Null, specify timezone of the agent. **Note:** POM checks only Null value for time zone values, and does not restrict any other string value
16	Login failure. Invalid Agent Name	You can see this message only if you have POM integration with AACC. The agent name cannot be Null. You must specify a value for agent name.
17	Login failure. Authentication of agent failed.	You will see this message only if you have POM integration with CC Elite. Check the agent password. The agent password must match the agent password you specify in the Contact Center.

Related topics:

AGTLogonRESP(int result) on page 119

AGTLogonRESP(int result)

Syntax

```
int result
```

Return values

result: “0” indicates success.

AGTLogoff()

No parameters

Description

The agent sends this command from the desktop when it wants to logout from POM. An agent can send this command ONLY when it is in “NotReady” state.

Error code	Error message	Proposed solution
541	Unable to logoff. Please move to not ready state	

Related topics:

AGTLogoffRESP(int result) on page 120

AGTLogoffRESP(int result)

Syntax

```
int result
```

Return values

result: “0” indicates success.

int AGTStateChange(POMAgentState agentState, String reasonCode, String reasonName, Boolean hasWalkedAway)

agentState An agent can specify the state to which the agent wants to transition to from the current state. The current agent states are:

- Ready: The agent is ready to pick calls.
- NotReady: The agent wants to take a break
- PendingNotReady: The agent is in the middle of the call and has issued a request to go to NotReady state, but POM cannot send the agent to NotReady.

reasonCode This is the AUX or reason code selected by agent from the desktop while going to NotReady state.

reasonName This is the AUX or reason code selected by agent from the desktop while going to NotReady state.

hasWalkedAway This flag is set to true if the agent is considered a walkaway agent. If an agent does not perform any actions for two consecutive calls, then the desktop can assume that the agent has walked away and change its state to NotReady, so that the next call is not sent to this agent. Except for a non-timed preview campaign an agent does not necessarily have to click buttons on the desktop to handle a call, assuming that the customer disconnects the calls and auto wrap up is implemented.

Description

The agent sends this command when it wants to go Ready and NotReady. An agent goes to Ready state when it wants to accept calls from POM. Unless the agent is ready POM will not handover a call to the agent. If the agent is in NotReady state, POM will not pass on the outbound call to the agent. An agent can go to NotReady state from a Ready state only after the current call is done OR to put it in other words, the agent can go to NotReady state only if it is in Idle state. If the agent is in the middle of a call OR in wrap up state, the agent can issue a NotReady request. POM accepts this request and puts the agent in PendingNotReady state. Once the agent is done disposing the existing call, POM will immediately put the agent in NotReady state and not give it a new call until it goes back to Ready State. An important point to note is that if this agent has pending consults + pending callbacks, POM will not move the agent from PendingNotReady to NotReady until the pending consults + pending callbacks are completed.

Error codes

Error code	Error message	Description
63	State change request sent more than once with the same reason code	When the agent tries to change the agent state to NotReady with the same reason, POM returns this error. The agent must select other reason code to mention different reason for NotReady.
64	Unable to change the current state	POM does not allow the agent to change the state to NotReady or Ready from the current state.

Related topics:

AGTStateChangeRESP(POMAgentState pomAgentState, int result) on page 121

AGTStateChangeRESP(POMAgentState pomAgentState, int result)

pomAgentState Agent state of type POMAgentState

Return values

result: “0” indicates success.

int AGTHoldCall(String sessionID)

sessionID Unique ID of the session for the entire contact processing duration.

Description

The agent sends this command when it wants to put the customer on hold. We can associate a Hold application with a strategy. This hold application will get played to the customer if an agent puts it on Hold. If the customer disconnects the call when an agent has put the customer on Hold, POM will drop the call and move the agent to Wrapup state.

Related topics:

AGTHoldCallRESP(String sessionID, int result) on page 122

AGTHoldCallRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTUnholdCall(String sessionID)

sessionID Unique ID of the contact for the entire contact processing duration.

Description

The agent sends this command when it wants to take the customer out of hold state.

Related topics:

AGTUnholdCallRESP(String sessionID, int result) on page 123

AGTUnholdCallRESP(String sessionID, int result)

The `API name` <Insert a brief description of API>.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

---

int AGTReleaseLine(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

The agent sends this command when the agent wants to disconnect the customer call. POM moves the agent to wrapup state after the agent disconnects the customer call.

An agent can issue this command only when the agent is in Talking state.

Error codes

Error code	Error message	Description
81	Unable to release the call	POM cannot release the call for the agent. The agent has some pending commands which the agent must address.

Related topics:

AGTReleaseLineRESP(WrapupData wrapupDetails, String sessionID, int result) on page 123

AGTReleaseLineRESP(WrapupData wrapupDetails, String sessionID, int result)

wrapupDetail This structure will comprise of the following 3 values:

- *acwMaxTime*: Maximum time allowed to an agent to wrapup the call.
- *acwExtendable*: Boolean value indicating if the agent can request more wrapup time.
- *defaultCompCode*: The default completion code which can be used to wrapup the call.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

---

int AGTGetCompCodes(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

The agent sends this command when the agent wants a list of disposition codes associated with the campaign that the agent is handling. POM sends back the custom completion code list associated with the campaign that this agent is part of.

This command can be issued in Talking/Wrapup state.

Related topics:

AGTGetCompCodesRESP(POMCompletionCode[] completionCodesList, String sessionID, int result) on page 124

AGTGetCompCodesRESP(POMCompletionCode[] completionCodesList, String sessionID, int result)

compCodeList List of custom completion codes associated with the current job.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success

int AGTWrapupContact(POMCompletionCode completionCode, String sessionID)

completionCode An agent can provide a completion code (disposition code) to dispose the call.

sessionID Unique ID of the call for the entire contact processing duration.

Description

The agent sends this command when the agent wants to wrap up the existing call. The agent has to select an completion code to wrap up the call. An agent can get the completion code list either by invoking AGTGetCompCodes OR as a default completion code from AGTAutoReleaseLine or AGTReleaseLine response.

Error codes

Error code	Error message	Description
202	Unable to wrapup with this completion code	POM is unable to update the given completion code for the contact. Try to update the contact with a different completion code.

Related topics:

AGTWrapupContactRESP(String sessionID, int result) on page 125

AGTWrapupContactRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTExtendWrapup(String sessionID)

String sessionID Unique ID of the call for the entire contact processing duration.

Description

The agent sends this command when it wants to extend the wrap up time of the current call. This command can be issued only in wrapup state. We can specify a wrapup time and number of wrapup extensions in the strategy editor.

- Wrapup time: Time value in seconds before which the agent should provide a disposition for the call.
- Wrapup extensions: If an agent needs more time to wrapup a call, for example it needs to perform from CRM activities, then it can ask for an wrapup extension from POM. If the POM user has provided extensions in the contact strategy, then POM sends back the extension time allowed for this agent. This way the agent can get more than the wrapup time (above) to dispose the current call.

Related topics:

AGTExtendWrapupRESP(WrapupData wrapupDetails, String sessionID, int result) on page 126

AGTExtendWrapupRESP(WrapupData wrapupDetails, String sessionID, int result)

wrapupDetail This structure will comprise of the following 3 values:

- *acwMaxTime*: Maximum time allowed to an agent to wrapup the call.
- *acwExtendable*: Boolean value indicating if the agent can request more wrapup time.
- *defaultCompCode*: The default completion code which can be used to wrapup the call.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

---

int AGTGetConsultTypes(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

The agent sends this command to find out the types of consult supported for the current job that the agent is attached to.

The currently supported types are:

- Agent: You can consult any agent which is ready + nailed + attached to the same job
- External – You can consult any external party. In POM you can define external party by adding a contact in Address book. You can also enter a free form number for consultation.

This command can be sent only in Talking state.

Related topics:

AGTGetConsultTypesRESP(POMDestinationType[] destinationTypes, bool allowFreeForm, String sessionID, int result) on page 127

AGTGetConsultTypesRESP(POMDestinationType[] destinationTypes, bool allowFreeForm, String sessionID, int result)

destinationTypes	POMDestinationType enumerator
allowFreeForm	Boolean flag indicating if the agent can enter a free form number for consultation.
sessionID	Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

---

int AGTGetConsultDestsForType(POMDestinationType destinationType, String sessionID)

destinationType	The selected transfer type Agent/External.
sessionID	Unique ID of the call for the entire contact processing duration.

Description

The agent sends this command after selecting a consultation type. This command can be sent only after AGTGetConsultTypes and in Talking state.

Note:

POM supports maximum packet size of 16000 bytes. If the list of agents, or list of external addresses exceeds 16000 bytes, POM truncates the list of agents or list of external addresses.

Error codes

Error code	Error message	Description
382	No destinations available	POM is unable to send external destinations for consult. The administrator must select some addresses for the campaign for which the contact is being processed.
383	Agents are not available	POM is unable to send agent destinations for consult. There are no available agents for consult in the running job.

Related topics:

AGTGetConsultDestsForTypeRESP(POMDestination[] destinations, String sessionID, int result) on page 128

AGTGetConsultDestsForTypeRESP(POMDestination[] destinations, String sessionID, int result)

destinations List of available agents for the selected type of consult.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTConsultCall(POMDestination destination, String sessionID)

destination The selected destination from the list sent by POM in response to AGTGetTransferDestsForType or AGTGetConfDestsForType.

sessionID Unique ID of the call for the entire contact processing duration.

Description

The agent sends this command after selecting a destination returned by either AGTGetConsultDestsForType. An agent has to consult before attempting a transfer or a conference.

Error codes

Error code	Error message	Description
141	Unable to consult as the selected agent is currently not ready	Agent cannot consult with other agent if the agent is in either Not Ready, or in process of logging out from the system, or the agent is unnailed, If such agent is selected, POM returns the error. The agent must choose some other agent for consult.
143	Unable to consult as the selected destination is invalid	If the agent desktop sends invalid or null destination, POM returns the error.
144	Unable to consult as the selected agent already has maximum pending consults	Agent cannot consult with other agent if the agent has more than 2 pending consults.

Related topics:

AGTConsultCallRESP(String sessionID, int result) on page 129

AGTConsultCallRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTCompleteTransfer(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

The agent sends this command when it wants to transfer the consultation call to the passive agent in the consult. POM transfers the customer call to the passive agent, who now becomes the call owner. The active agent which issues this command is moved directly to “Idle” state and is assumed ready for the next call.

Error codes

Error code	Error message	Description
421	System error. Unable to consult	POM is unable to transfer the call. The agent must retry the transfer or cancel the consult.

Related topics:

AGTCompleteTransferRESP(bool canDispose, POMWrapupDetails wrapupDetails, String sessionID, int result) on page 130

AGTCompleteTransferRESP(bool canDispose, POMWrapupDetails wrapupDetails, String sessionID, int result)

canDispose In case of an external transfer this will be true and the agent needs to provide disposition for the call.

wrapupDetails This structure will comprise of the following 3 values:

- *acwMaxTime*: Maximum time allowed to an agent to wrapup the call.
- *acwExtendable*: Boolean value indicating if the agent can request more wrapup time.
- *defaultCompCode*: The default completion code which can be used to wrapup the call.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTCancelConsult(String destAgentID, String sessionID)

destAgentID The selected destination for which the consult cancel request is intended.

sessionID Unique ID of the call for the entire contact processing duration.

Description

The active agent in a consult (customer call owner) sends this command to POM requesting it to cancel the ongoing consult with the passive agent. POM informs the passive agent about this action and makes the passive agent ready to accept a new call by moving him to Idle state. A pending consult request can also be cancelled using this command.

Error codes

Error code	Error message	Description
401	System error. Unable to cancel the consult	If the system is unable to find the instance of consult, the system displays the error on the agent desktop.

Related topics:

AGTCancelConsultRESP(String sessionID, int result) on page 131

AGTCancelConsultRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

---

int AGTStartConf(String sessionID)

sessionID Uique ID of the call for the entire contact processing duration.

Description

The agent sends this command when it wants to conference with the passive agent in the consult. POM joins the customer call with the active and the passive agents. The active agent’s state changes to ConferenceOwner, while the passive agent becomes ConferencePassive.

Related topics:

AGTStartConfRESP(String sessionID, int result) on page 131

AGTStartConfRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTEndConf(String sessionID)

sessionID Unique ID of the contact for the entire contact processing duration.

Description

Agent's involved in a conference can send this command when they want to end the conference. This command can be sent only in conference state. When the conference is ended the passive agent is informed about it and the active agent is moved back from conference state to Idle state. The passive agent can now receive new calls. The conference owner is moved from Conference to Talking state.

Error codes

Error code	Error message	Description
303	Unable to end the conference as passive agent not found	If the system is unable to find any passive agent in the agent buffer, the system displays the error on the agent desktop.

Related topics:

AGTEndConfRESP(String sessionID, int result) on page 132

AGTEndConfRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: "0" indicates success.

int AGTConferenceOwnershipChanged(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

The Conference owner can issue this command to transfer the conference ownership to the passive agent in the conference. POM will then move the passive agent from ConferencePassive state to ConferenceOwner state and vice-versa for the original conference owner.

Related topics:

AGTConfChangeOwnershipRESP(String sessionID, int result) on page 133

AGTConfChangeOwnershipRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTRedial(POMContactNumber contactNumber)

contactNumber The contact which needs to be redialed.

Description

An agent sends this command when it wants to redial the customer that it was talking to before the call got disconnected. This is a useful command which can be used if the customer call gets disconnected because of some reason (like network not reachable/weak signal/ disconnect by mistake). The agent can dial any number from the available numbers assigned to the contact. This command can be issued only in wrapup state.

Error codes

Error code	Error message	Description
362	Unable to redial as the address is invalid	If the customer number to redial is Null, the system displays the error on the agent desktop.

Related topics:

AGTRedialRESP(String sessionID, int result) on page 133

AGTRedialRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTSendDTMF(String dtmf, String sessionID)

dtmf The dtmf digit that agent wants to send on the call.

sessionID Unique ID of the call for the entire contact processing duration.

Description

An agent can send this command if it wants to send dtmf during the call. This command is useful if the agent is interacting with an IVR or an answering machine. The agent can send one dtmf at a time.

Error codes

Error code	Error message	Description
341	Cannot send DTMF as invalid data received	If the DTMF input is invalid or Null, the system displays the error on the agent desktop.
343	System error. Unable to send DTMF	If the telephony fails to send DTMF input, the system displays the error on the agent desktop.

Related topics:

AGTSendDTMFRESP(String sessionID, int result) on page 134

AGTSendDTMFRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTGetCallbackTypes(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

An agent can send this command to check the types of callbacks supported by POM. Currently we support the following callback types:

- Standard: Any agent having the right skill (skill assigned to the job in which the callback is scheduled) will receive the callback.
- Agent: Preferrably POM will try to send the callback to the agent which scheduled the callback OR Any agent having the right skill (skill assigned to the job in which the callback is scheduled).
- Campaign: Any agent having the right skill (skill assigned to the campaign selected for the callback) will receive the callback.

An agent has to be ready and nailed to receive a callback.

Related topics:

AGTGetCallbackTypesRESP(POMCallbackType[] callbackTypes, String defaultExpiryTime, String sessionID, int result) on page 135

AGTGetCallbackTypesRESP(POMCallbackType[] callbackTypes, String defaultExpiryTime, String sessionID, int result)

callbackTypes	:POMCallbackTypes enumerator
defaultExpiryTimes	Default expiry time value
sessionID:	Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

---

int AGTGetCallbackDestsForTypes(String sessionID)

sessionID	Unique ID of the contact for the entire contact processing duration.

Description

An agent can send this command to request POM to send back the list of available destinations for the selected callback type. An agent can set callbacks only in Talking/Wrapup state.

Note:

POM supports maximum packet size of 16000 bytes. If the list of campaigns or list of agents exceed 16000 bytes, POM truncates the list of campaigns or list of agents.

Error codes

Error code	Error message	Description
221	Cannot get destinations as callback type is invalid	If the callback type in the API is Null, the system displays the error on the agent desktop,
222	Unable to get zone of the contact	If the system is unable to find the zone of a contact on which the callback is being created, the system displays the error on the agent desktop.
223	Unable to get campaigns for the organization	If the system is unable to find any campaign for the organization, the system displays the error on the agent desktop.

Related topics:

AGTGetCallbackDestsForTypeRESP(POMCallbackType callbackType, POMCallbackDest[] callbackDests, String sessionID, int result) on page 136

AGTGetCallbackDestsForTypeRESP(POMCallbackType callbackType, POMCallbackDest[] callbackDests, String sessionID, int result)

callbackType	Type of the callback destinations sent back from POM.
callbackDestinations	List of available destinations for callback.
sessionID	Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTGetCallbackDestsForType(POMCallbackType callbackType, String zoneName, String sessionID)

callbackType	The callback type which needs to be scheduled.
zoneName	Agent list of only this zone is expected.
sessionID	Unique ID of the contact for the entire contact processing duration.

Description

An agent can send this command to request POM to send back the list of available destinations for the selected callback type. An agent can set callbacks only in Preview, Talking & Wrapup state.

Note:

POM supports maximum packet size of 16000 bytes. If the list of campaigns or list of agents exceed 16000 bytes, POM truncates the list of campaigns or list of agents.

Error codes

Error code	Error message	Description
221	Cannot get destinations as callback type is invalid	If the callback type in the API is Null, the system displays the error on the agent desktop,
222	Unable to get zone of the contact	If the system is unable to find the zone of a contact on which the callback is being created, the system displays the error on the agent desktop.
223	Unable to get campaigns for the organization	If the system is unable to find any campaign for the organization, the system displays the error on the agent desktop.

Related topics:

AGTGetCallbackDestsForTypeRESP(POMCallbackType callbackType, POMCallbackDest[] callbackDests, String sessionID, int result) on page 137

AGTGetCallbackDestsForTypeRESP(POMCallbackType callbackType, POMCallbackDest[] callbackDests, String sessionID, int result)

callbackType	Type of the callback destinations sent back from POM.
callbackDestinations	List of available destinations for callback.
sessionID	Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTCreateCallback(POMCallbackType callbackType, POMCallbackDest callbackDest, String callbackTime, String callbackTimezone, String callbackExpiryTime,

POMContactNumber contactNumber, String agentNotes, String sessionID)

callbackType	The callback type which needs to be scheduled.
callbackDest	The callback destination which should be dialed
callbackTime	The callback time in UTC timezone.
callbackTimezone	The timezone for which the callback should be dialed
callbackExpiryTime	Time limit in minutes after which the callback expires, starting from callbackTime.
contactNumber	The contact number which should be dialed
agentNotes	The agent can set notes which can be referred during the callback.
sessionID	Unique ID of the call for the entire contact processing duration.

Description

An agent can send this command to request POM to create a callback. This command can be issued in Talking/Wrapup state only. According to the type of callback POM can send a pending callback notification to the agent few minutes (configurable value) before its scheduled time.

Error codes

Error code	Error message	Proposed solution
484	Unable to create callback as the campaign is not found	For a standard callback, if the system cannot find the campaign job in which the callback is created, the system displays the error on the agent desktop. then date parsing fails and this error is displayed.
485	Unable to create the callback as invalid callback type received	If the system receives an invalid or Null callback type from the desktop, the system displays the error on the agent desktop.
486	Unable to create callback as callback time received in invalid format	If the date format given in this API does not match with yyyy/MM/dd/ HH:mm, the system displays the error on the agent desktop.
487	Unable to create the callback as the expiry time is invalid	If the expiry time sent in this API is invalid or lesser than 0, the system displays the error on the agent desktop.
488	Unable to create callback as the callback time is invalid	If the callback scheduled time is earlier than current time, the system

Error code	Error message	Proposed solution
		displays the error on the agent desktop.

Related topics:

AGTCreateCallbackRESP(String sessionID, int result) on page 139

AGTCreateCallbackRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTGetErrorString(String errorCode)

errorCode The error code sent by POM in response to a command failure

Description

Desktop can invoke this function to get details of the error sent in response to a command. The error code sent back is a number. AGTGetErrorString will return a string which will provide some details about the error.

Related topics:

AGTGetErrorStringRESP(String errorMsg, String localizedErrorMsg, int result) on page 139

AGTGetErrorStringRESP(String errorMsg, String localizedErrorMsg, int result)

errorMsg Error message.

localizedErrorMsg Localized error message based on the locale passed during agent login.

Return values

result: “0” indicates success.

int AGTPreviewDial(POMContactNumber contactNumber, String sessionID)

contactNumber The contact number that needs to be dialed for preview.

sessionID Unique ID of the call for the entire contact processing duration.

Description

Agent can send this command if it wants to accept the preview notification and call the customer. Agent can select amongst the available numbers and invoke AGTPreviewDial. This command can be invoked only for a PreviewCampaign and in Preview state.

Related topics:

AGTPreviewDialRESP(String sessionID, int result) on page 140

AGTPreviewDialRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTPreviewCancel(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

Agent can send this command if it does not want to accept a preview contact and instead cancel the dial request. POM moves this agent from Preview state to Wrapup state. An agent can decide if it wants to reject a contact dial attempt for any reason and provide those details during wrapup.

Related topics:

AGTPreviewCancelRESP(WrapupData wrapupDetails, String sessionID, int result) on page 141

AGTPreviewCancelRESP(WrapupData wrapupDetails, String sessionID, int result)

wrapupDetails This structure will comprise of the following 3 values:

- *acwMaxTime*: Maximum time allowed to an agent to wrapup the call.
- *acwExtendable*: Boolean value indicating if the agent can request more wrapup time.
- *defaultCompCode*: The default completion code which can be used to wrapup the call.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTGetCustomerDetails(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

Agent can invoke this command if it wants complete details of a contact. Details include Title/ FirstName/LastName/Address/Phone/Email fields and some custom fields. Usually this command should be invoked after receiving a new call notification, that is, AGTCallNotify.

Related topics:

AGTGetCustomerDetailsRESP(POMCustomerDetails customerDetails, String sessionID, int result) on page 141

AGTGetCustomerDetailsRESP(POMCustomerDetails customerDetails, String sessionID, int result)

customerDetails POMCustomerDetails object containing customer data.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTSetCustomerDetail(POMKeyValuePair pomKeyValuePair, String sessionID)

pomKeyValuePair New value that needs to be changed for a key.

sessionID Unique ID of the call for the entire contact processing duration.

Description

Agent can invoke this command if it wants to change a field value of a customer. For example on receiving a call, the customer could request an address/phone/email value change.

Error code	Error message	Description
641	Attribute is read only, cannot update this attribute	If the agent tries to update a read only attribute through the agent desktop, the system displays the error on the agent desktop.
642	Invalid value entered. Unable to save attribute	If the validation of the value of the contact attribute fails on the system, the system displays the error on the agent desktop. For example, if the attribute is float type and a string value like "ABCD" is provided for the attribute, the system displays the error on the agent desktop.

Related topics:

AGTSetCustomerDetailRESP(String sessionID, int result) on page 142

AGTSetCustomerDetailRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTBlendToInbound()

No parameters.

Description

Sent by AACC blender to move an agent from Outbound to Inbound.

Error code	Error message	Description
581	Agent is already assigned to Inbound	If an inbound agent is being tried to move to inbound again using this API, the system displays the error on the agent desktop.

Related topics:

AGTBlendToInboundRESP(int result) on page 143

AGTBlendToInboundRESP(int result)

Return values

result: “0” indicates success.

int AGTBlendToOutbound()

No parameters.

Description

Sent by AACC blender to move an agent from Inbound to Outbound.

Error code	Error message	Description
601	Agent is already assigned to Outbound	If an outbound agent is being tried to move to outbound again using this API, the system displays the error on the agent desktop.

Related topics:

AGTBlendToOutboundRESP(int result) on page 143

AGTBlendToOutboundRESP(int result)

Return values

result: “0” indicates success.

int AGTNailupAgent()

No parameters.

Description

This command is sent by the blender when it moves the agent from Inbound to the Outbound queue.

Related topics:

AGTNailupAgentRESP(int result) on page 144

AGTNailupAgentRESP(int result)

No parameters.

Return values

result: “0” indicates success.

---

int AGTReadyForNailup()

No parameters.

Description

This command is sent by the desktop when it has processed the AGTNailupChange-PendingNailup notification. This gives an agent enough time to be prepared for nailup. It also acts as a indication to the desktop (with a softphone) that the next call will be a nailing call which should be autoanswered and its not a generic inbound call.

Related topics:

AGTReadyForNailupRESP(int result) on page 144

AGTReadyForNailupRESP(int result)

No parameters.

Return values

result: “0” indicates success.

---

int GetAgentStatusResponse(POMAgentStatus status)

status Value having the agent state, call state, and call state value.

Description

This command should be sent by desktop on receiving GetAgentStatus notification. Agent manager sends GetAgentStatus whenever it wants to check the current agent state on the desktop. This is mainly used during High Availability scenario. So, whenever Agent Manager is restarted, it will send GetAgentStatus whenever the desktop gets connected to Agent Manager after restart. The desktop is expected to set its current known Agent State, Nailing State and Call State.

Related topics:

GetAgentStatusResponseRESP(int result) on page 145

GetAgentStatusResponseRESP(int result)

No parameters.

Return values

result: “0” indicates success.

---

int AGTLostNailing()

No parameters.

Description

This command is sent by the desktop when it detects loss of nailing in the softphone or hard phone using CTI.

Related topics:

AGTLostNailingRESP(WrapupData wrapupDetails, Sting sessionID, int result) on page 145

AGTLostNailingRESP(WrapupData wrapupDetails, Sting sessionID, int result)

wrapupDetails This structure will comprise of the following 3 values:

- *acwMaxTime*: Maximum time allowed to an agent to wrapup the call.
- *cwExtendable*: Boolean value indicating if the agent can request more wrapup time.
- *adefaultCompCode*: The default completion code which can be used to wrapup the call.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTPendingLogout()

No parameters.

Description

This command can be sent at any state by an agent to logout from the desktop. If the agent is in the middle of the call, then POM will wait for wrap up and move the agent to not ready state and then logout the agent. This should be used when the desktop detects an erroneous condition.

Related topics:

AGTPendingLogoutRESP(int result) on page 146

AGTPendingLogoutRESP(int result)

No parameters.

Return values

result: “0” indicates success.

int AGTAddAgentNote(String agentNote)

agentNote Notes to be stored by POM.

Description

An agent can use this command to store agent notes for a contact in POM. In case of consult/transfer/conference/callbacks, these notes can be used as reference by an agent.

Error code	Error message	Description
561	Invalid data received. Unable to add agent notes	If the agent notes are Null or empty, the system displays the error on the agent desktop.
562	System error. Unable to add agent notes	If the system fails to add the agent notes, the system displays the error on the agent desktop.

Error code	Error message	Description
563	System error. Unable to add agent notes	If the system fails to add agent notes because contact is not found, the system displays the error on the desktop.

Related topics:

AGTAddAgentNoteRESP(String sessionID, int result) on page 147

AGTAddAgentNoteRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

---

int AGTRefreshAgentNotes(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

An agent can use this command to get agent notes which were stored for this contact for the current job. Notes could have been saved by an agent during talking/consult/transfer/ conference/callbacks.

Note:

POM supports maximum packet size of 16000 bytes. If the notes exceed 16000 bytes, POM truncates the notes.

Related topics:

AGTRefreshAgentNotesRESP(String[] agentNotes, String sessionID, int result) on page 147

AGTRefreshAgentNotesRESP(String[] agentNotes, String sessionID, int result)

agentNotes Agent notes added earlier for the call.

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success.

int AGTGetTimezones()

No parameters.

Description

This command is used to get a list of all supported timezones by POM. This command should be invoked before creating callbacks.

Related topics:

AGTGetTimezonesRESP(POMKeyValuePair[] timezones, int result) on page 148

AGTGetTimezonesRESP(POMKeyValuePair[] timezones, int result)

timeZones List of supported timezones.

Return values

result: “0” indicates success.

AGTAvailableForNailup()

No parameters.

Description

This is a command to be sent by Desktop only once after the agent gets Ready for the first time after logging in. This has to be called because in CC Elite environment we cannot change state while nailed up, so desktop has to send this after the agent goes to ready state.

Related topics:

AGTAvailableForNailupRESP(int result) on page 148

AGTAvailableForNailupRESP(int result)

No parameters.

Return values

result: “0” indicates success.

int AGTAgentDisconnected()

No parameters.

Description

This command should be sent by a component which monitors the desktop. Sending this command will allow POM to remove this agent from pacing and POM will stop sending new notifications to this agent.

Related topics:

AGTAgentDisconnectedRESP(int result) on page 149

AGTAgentDisconnectedRESP(int result)

No parameters.

Return values

result: “0” indicates success.

int AGTAddToDNC(POMAttribute[] addressList)

addressList List of addresses to be added to DNC list.

Description

The agent can add a contact address or more addresses into the DNC list maintained by POM.

Error code	Error message	Description
621	Unable to add to DNC. Contact address already present.	If the contact address is already in DNC, the system displays the error on the agent desktop.
622	System error. Unable to add to DNC	If the system is unable to add the contact address to DNC, the system displays the error on the agent desktop.
623	Invalid data received. Unable to add to DNC	If the contact attribute for which the system tries to add the contact address to DNC is Null, or not present in the system, the system displays the error on the agent desktop.

Related topics:

AGTAddToDNCRESP(String sessionID, int result) on page 150

AGTAddToDNCRESP(String sessionID, int result)

sessionID Unique ID of the call for the entire contact processing duration.

Return values

result: “0” indicates success

---

int AGTIsInDNC(String addressValue, String sessionID)

The agent can invoke this command to verify if the address is already in the POM DNC list.

addressValue Address to be verified in POM DNC list.

sessionID Unique ID of the call for the entire contact processing duration.

The agent can invoke this command to verify if the address is already in the POM DNC list.

Related topics:

AGTIsInDNCRESP(bool present, int result) on page 150

AGTIsInDNCRESP(bool present, int result)

present True if the value is in the POM DNC list, false otherwise.

Return values

result: “0” indicates success.

---

int AGTGetZoneList()

No parameters.

The agent can invoke this command to get a list of zones configured on POM.

Related topics:

AGTGetZoneListRESP(String[] zoneList, int result) on page 151

AGTGetZoneListRESP(String[] zoneList, int result)

zoneList List of zones configured on POM.

Return values

result: “0” indicates success.

---

int AGTSaveAgentForHA()

No parameters.

The agent should invoke this if it is interested in POM restoring the agent desktop state in a scenario where the desktop crashed and was restarted. So, if the agent invokes AGTSaveAgentForHA and after some time while performing some activity for a customer the desktop crashed, then if the agent re-logs in then POM will try to restore the agent state back to what it was before the crash situation.

Related topics:

AGTSaveAgentForHARESP(int result) on page 151

AGTSaveAgentForHARESP(int result)

Return values

result: “0” indicates success.

---

int AGTSkillsChanged(POMAgentSkill[] agentSkills)

agentSkills The current agent skill set.

AAAD will send this whenever it detects any changes in the agent skillset on AACC.

Related topics:

AGTSkillsChangedRESP(int result) on page 151

AGTSkillsChangedRESP(int result)

No parameters.

Return values

result: “0” indicates success.

int AGTGetContactAttributes()

No parameters.

The agent can send this command to get the list of attributes and their types, associated with the current contact. The attribute values are not sent in response to this command request. AGTGetCustomerDetails should be invoked to get contact attribute values.

Related topics:

AGTGetContactAttributesRESP(int result) on page 152

AGTGetContactAttributesRESP(int result)

No parameters.

Return values

result: “0” indicates success.

Commands

AGTStateChangedNotify(POMAgentState agentState)

agentState Agent state. Value can be one of Ready/NotReady/PendingNotReady

Description

This notification is sent by POM whenever it detects change in agent state. If the agent requests to go to Ready state by invoking AGTStateChange, along with a response for AGTStateChange command POM sends back this notification with the agentState as Ready. If the agent requests to go to NotReady state during a call, POM will send back AGTStateChange notification with the agentState as PendingNotReady. But, as soon as POM finds that the agent is done with his calls, the agent which was put in PendingNotReady state is immediately sent to NotReady state and POM informs this to the desktop by sending AGTStateChange notification.

AGTCallNotify(POMContact contact, String sessionID)

contact POMContact object containing the customer and job related information about the contact which will be (preview campaign)/got (Predictive + Progressive campaign) dialed.

sessionID sessionID: Unique ID of the call for the entire contact processing duration.

Description

In case of Predictive and Progressive campaigns this notification is sent by POM whenever POM connects the outbound call to the customer with the agent. The selected agent is sent this notification, so that the agent gets to know the customer that got dialed. The desktop should then invoke AGTGetCustomerDetails command to get more details about the customer.

AGTAutoReleaseLine(WrapupData wrapupDetails, String sessionID)

wrapupDetails This structure will comprise of the following 3 values:

- *acwMaxTime*: Maximum time allowed to an agent to wrapup the call.
- *acwExtendable*: Boolean value indicating if the agent can request more wrapup time.
- *defaultCompCode*: The default completion code which can be used to wrapup the call.

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM whenever it detects that the customer has disconnected the call. The desktop should allow the agent to wrapup the call by creating a timer with the time value set to acwMaxTime. If the agent needs more time to wrapup the call and if acwExtendable is set to true, then the agent can invoke AGTExtendWrapup and get some more time from POM. defaultCompCode can be used to auto wrapup the call, if the agent does not select a completion code during the wrapup time. Desktop can perform an autowrapup after the acwMaxTime expires and the agent does not invoke AGTWrapupContact. Desktop can use the defaultCompCode for AGTWrapupContact.

AGTConsultNotify(POMContact contact, String requestingAgentId, String sessionID)

contact	POMContact object containing the customer and job related information about the contact for which this consult request has been sent.
requestingAgentID	Agent ID of the consult initiating agent.
sessionID	Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM when an agent (active) requests a consult with another agent (passive). If the passive agent is busy on another call, POM sends AGTPendingConsult to the passive agent. AGTConsultNotify is sent only when the passive agent is successfully taken into the consultation call.

AGTConsultCancelled(String requestingAgentId, String sessionID)

requestingAgentID	Agent ID of the agent which initiated the cancel consult request.
sessionID	Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM when a consult is cancelled/dropped. POM sends this notification to an agent informing about the cancel operation performed by the other agent. This notification is sent either because of AGTCancelConsult or AGTAutoReelaseLine operations.

AGTTransferNotify(POMContact contact, String sessionID)

contact	POMContact object containing the customer and job related information about the contact for which this transfer happened.
sessionID	Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM when an agent (active) completes the transfer to the consulted agent (passive). AGTTransferNotify is sent by POM to the passive agent informing it about the call ownership. The earlier active agent is moved to Idle state, while the new active agent is moved to Talking/Busy state.

AGTConferenceNotify(POMContact pomContact, String sessionID)

contact POMContact object containing the customer and job related information about the contact for which this conference happened.

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM when an agent (active) starts the conference with the consulted agent (passive). AGTConferenceNotify is sent by POM to the passive agent informing it about the conference. The earlier active agent is moved to ConferenceOwner state, while the passive agent is moved to ConferencePassive state.

AGTConferenceEnded(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM to an agent informing it that the other agent has ended the conference. This can also be sent by POM when the customer releases the call.

int AGTConferenceOwnershipChanged(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

The Conference owner can issue this command to transfer the conference ownership to the passive agent in the conference. POM will then move the passive agent from ConferencePassive state to ConferenceOwner state and vice-versa for the original conference owner.

Related topics:

AGTConfChangeOwnershipRESP(String sessionID, int result) on page 133

AGTCapabilitiesChanged(POMCapabilities capabilities)

capabilities There are almost 17 capabilities associated with an agent. This object contains the status of those capabilities.

Description

POM sends this notification from time to time, whenever it detects a Call/Agent/Nailed/Job status change. Some of the capabilities are CanHold/CanTransfer/CanConsult. This is a very critical notification because this notification helps the desktop in becoming stateless. The desktop can hide/unhide various controls based on values in capabilities.

AGTNailupChange(POMNailupStatus nailupStatus)

nailupStatus Values are PendingNailup/NailedUp/PendingNailupDrop/NotNailed/NailingLost

Description

This notification is sent by POM to the desktop informing it about the agent's current nailing state. Its sent whenever a nailing state change is detected.

AGTCallStateChangedNotify(POMCallState callState)

callState Provides current call state value. Some values are Preview/Dialing/Talking/Held/Wrapup.

Description

This notification is sent by POM to the desktop whenever it detects a change in the current call state. The call state changes based on the agent state machine inside POM and also based on actions taken by the agent during a call.

AGTDialFailed(WrapupData wrapupDetails, POMDialFailReason dialFailReason, String sessionID)

Syntax

```
WrapupData wrapupDetails, POMDialFailReason dialFailReason, String sessionID
```

wrapupDetails This structure will comprise of the following 3 values:

- *acwMaxTime*: Maximum time allowed to an agent to wrapup the call.
- *acwExtendable*: Boolean value indicating if the agent can request more wrapup time.
- *defaultCompCode*: The default completion code which can be used to wrapup the call.

dialFailReason An enumerator value associated with a possible dial fail reason.

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM to the agent if the Preview Dial attempt fails or a Redial attempt fails. POM moves the agent to Wrapup state after sending this notification. The agent can use the defaultCompCode to auto dispose the call. The action to be taken after receiving this notification is similar to AGTReleaseLine and AGTAutoReleaseLine.

AGTConsultDialFailed(POMDialFailReason dialFailReason, String sessionID)

dialFailReason An enumerator value associated with a possible dial fail reason.

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM to the agent that attempts a consult to another agent. The agent stays in Talking state. A consult could fail because of various reasons especially if the party being consulted is an external number.

AGTConsultPending(String requestingAgent, String requestingAgentID, String requestingCampaign, String sessionID)

requestingAgent Agent requesting the consult.

requestingAgentId Agent ID of the agent requesting the consult.

consultrequestingCampaign Name of the campaign to which the consult requesting agent is part of.

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM when the agent (active) requests a consult with another agent (passive) when the passive agent is already busy in a call i.e. the passive agent is either in Busy state. The receiving agent can decide on an action with the current call based on this notification.

AGTPendingConsultComplete(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM to the active agent when the consult is successfully completed with the passive agent.

AGTPreviewCallbackPending(String dueTime, String sessionID)

dueTime Time at which the callback is due.

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM to an agent a few seconds (configurable value on POM) before the scheduled callback time. This notification is useful for the agent in deciding his action for the current call.

AGTPreviewCallbackPending(String dueTime, String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent by POM to an agent if the callback is cancelled by POM because it has expired.

AGTAgentLoggedOut()

No parameters.

Description

This notification is sent by POM when it successfully logs out an agent. This is mainly useful if the agent has issued a AGTPendingLogout request. This notification provides the agent a confirmation about his logout operation.

AGTCustomerDetailsChanged(POMAttribute attribute, String sessionID)

The `API name` <Insert a brief description of API>.

attribute POMAttribute that has changed.

sessionID Unique ID of the call for the entire contact processing duration.

This notification is sent by POM to find out the current agent state. This is sent either when agent manager restarts while the desktop is in use OR if agent manager does not detect any activity from the agent for a considerable amount of time.

The desktop developer is supposed to fill up the current agent state in the POMAgentStatus structure and send back in the notification method response.

AGTEnableCancelConsult(String sessionID)

sessionID Unique ID of the call for the entire contact processing duration.

Description

This notification is sent to the active agent so that the desktop can enable the cancel consult feature. There are certain scenarios/situations in which the consult cannot be cancelled. This notification is used by POM to take care of such scenarios.

AGTInvalidCommandName(String commandName)

commandName Details of the data that POM received.

Description

This notification is sent to the desktop if an invalid command is received by POM.

POMAgentStatus GetAgentStatus(String agentID)

This notification is sent by POM to find out the current agent state. This is sent either when agent manager restarts while the desktop is in use OR if agent manager does not detect any activity from the agent for a considerable amount of time.

The desktop developer is supposed to fill up the current agent state in the POMAgentStatus structure and send back in the notification method response.

agentID ID of the requested agent's status.

This notification is sent by POM to find out the current agent state. This is sent either when agent manager restarts while the desktop is in use OR if agent manager does not detect any activity from the agent for a considerable amount of time.

The desktop developer is supposed to fill up the current agent state in the POMAgentStatus structure and send back in the notification method response.

POMAvailable()

No parameters.

This notification is sent by POM whenever the library is connected to agent manager for the first time OR if the agent manager restarts while the desktop is still in use.

POMNotAvailable()

No parameters.

This notification is sent by the library whenever it detects that the connection to the agent manager is lost.

AGTBlendedtoOutbound()

No parameters.

This notification is sent by POM when the agent is moved to Outbound.

AGTBlendedToInbound()

No parameters.

This notification is sent by POM when the agent is moved to Inbound.

AGTZoneDown(String zoneName, int gracePeriodInMillis)

zoneName Name of the zone.

gracePeriodinMillis Grace period in milliseconds after which the agent will be forcefully logged out by POM.

This notification is sent by POM whenever it detects that a zone is going down. It is expected that the agent finishes off all his current activity before gracePeriodInMiilis, otherwise the agent will be forcefully logged out.

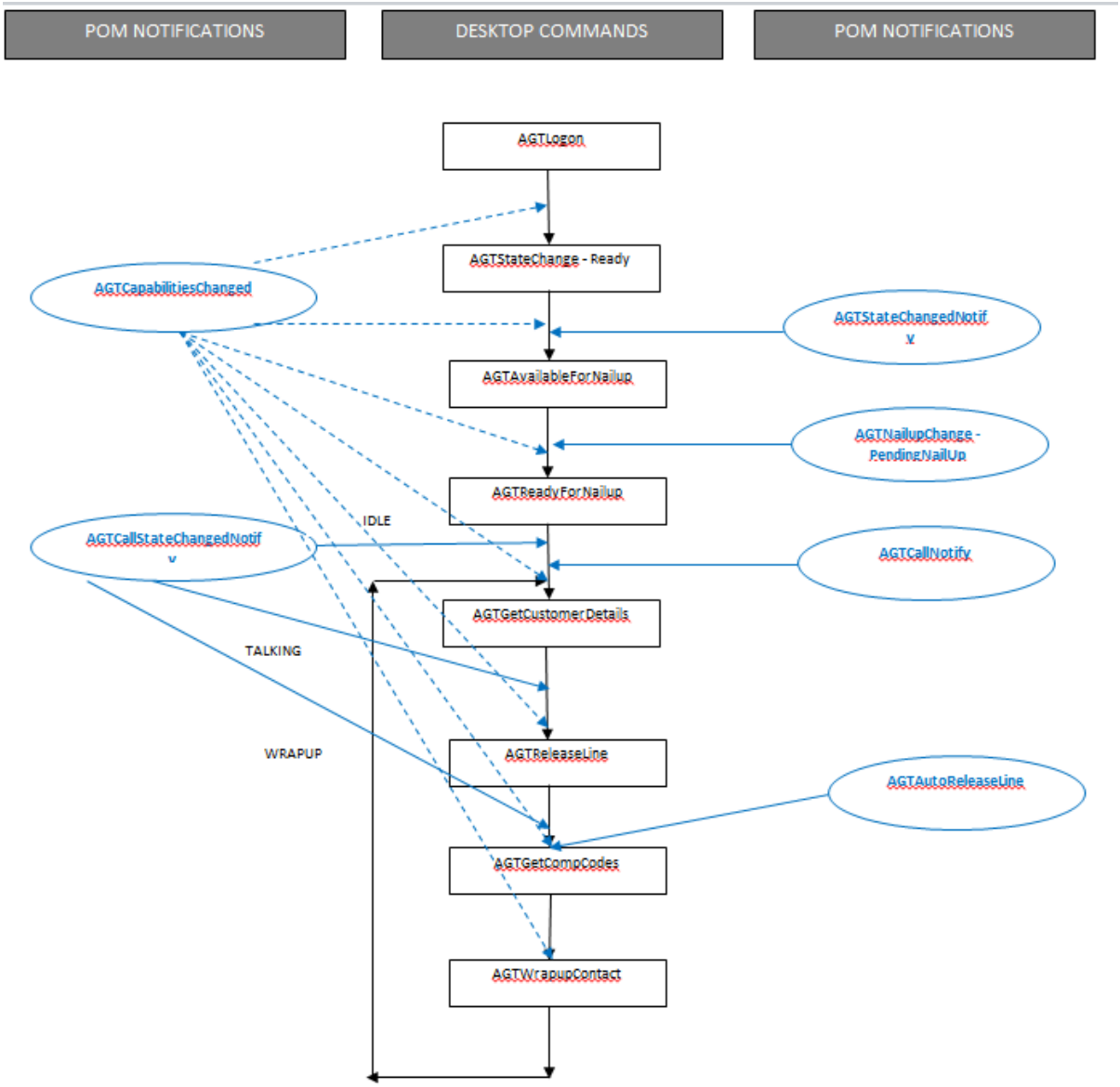
AGTJobAttached(String campaignName)

campaignName Name of the campaign.

This notification is sent by POM to whenever the agent is attached to a new job. The associated campaign name is sent as an argument.

Call flow and capability matrix

Simple Call Flow



Capability matrix

	Hold	Unhold	Transfer	Conference	Release	Disposition	Originate	CreateCallback	SendDTMF	UpdateRecord	LeaveConference	LeaveConsult	EndConference	ChangeOwnership	Ready	NotReady	Record

Idle	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N
Held	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N
Talking	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	N	N	N	N	Y	N
Consult (Owner)	N	N	Y	Y	N	N	N	N	N	N	N	Y	N	N	N	Y	N
Consult (Recipient)	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	Y	N
Conference (Owner)	N	N	N	N	N	N	N	N	N	Y	N	N	Y	Y	N	Y	N
Conference (Recipient)	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	Y	N
Wrap up	N	N	N	N	N	Y	N	Y	N	N	N	N	N	N	N	Y	N
NotReady	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
Pending Not Ready	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
Ready	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N
Recording Present	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N
Preview/ Callback	N	N	N	N	N	N	N	Y	N	Y	N	N	N	N	N	Y	N

Dialing	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N

Logs and traces, Error messages, Troubleshooting, and Miscellaneous Notes

Logs and traces

The library generates its own logs using Apache `log4net` library. A configuration file is shipped with the library, which stores the path locations of the dependent libraries and also the `log4net` library. The log4net configuration file `log4net.xml` is also shipped along with the POM Desktop API libraries.

Error messages

If an API sends back an error code, then the desktop developer invokes AGTGetErrorString to get the appropriate error message. The error messages are localized and the appropriate localized message are available based on the locale provided with AGTGetErrorString. This API request is sent to POM and the value is returned back from POM. These error codes are generally sent with the response for a command. All response messages ending with RESP have an error code resolved with a localized error message.

Note:

All error codes do not have an appropriate error message associated with them. There are certain error codes for which the AGTGetErrorString should not be invoked because the error codes are detected at the API level, POM does not receive the request . Only errors that POM detects, have an appropriate error message response with AGTGetErrorString. POM sends back all such error codes while invoking the commands.

Following error codes are not associated with error messages:

Error Code: 9999 (POM_NOT_AVAILABLE)

POM sends back the error code if the command is not sent to POM. POM generate the error code internally and sends the error ocde back with the appropriate response RESP.

Error Code: 9998 (PAM_NOT_AVAILABLE_FOR_ZONE)

POM sends back the error code if POM the agent manager does not manage the zones provided in the AGTLogon command.

Error Code: 9997 (INVALID_ARGUMENT)

POM sends back if an incorrect argument is sent to the API.

Error Code 9996 (SDK_Failure)

POM generates the error code if the library is unable to send the command to POM.

Note:

Ensure that you localize or translate the error codes if required, and manage the error codes.

Troubleshooting

The library logs are very important and useful in debugging issues in the first phase. The logging levels should be set to FINEST level to track issues. The logs will list all communication messages between the desktop library and POM. It will also log exceptions/errors which are useful in understanding issues. The logs will also indicate the flow of data between the library user and POM.

POM API error codes

Error Code	Message
0	Success
1	Failure
2	This agent is not registered with the system
3	Unable to login. Check media server.
5	Agent is already nailed.
6	Unable to verify password.
7	Agent is already logged in.
8	Agent is forcefully logging in to the system.
9	Agent skills not found.
10	Unable to change the state of the agent.
11	Internal error. Unable to login agent.
12	Login failure. Zone not found.
61	Agent is already in ready state.
62	Agent walkaway received when agent is handling a call.
63	State change request sent more than once with the same reason code.
64	Unable to change the current state.

Error Code	Message
81	Unable to release the call.
101	Unable to hold.
102	Unable to hold as call is disconnected.
103	Unable to hold as agent is not on call.
121	Unable to unhold.
122	Unable to unhold as call is disconnected.
123	Unable to unhold as agent is not on call.
141	Unable to consult as the selected agent is currently not ready.
142	Unable to consult as the selected agent has logged out.
143	Unable to consult as the selected destination is invalid.
144	Unable to consult as the selected agent already has maximum pending consults.
181	Unable to conference.
182	Unable to conference as agent selected is not in a consult with this agent.
183	Unable to conference as agent selected for conference has logged out.
184	Unable to conference as agent selected for conference is currently not ready.
201	Unable to wrapup as contact record not found.
202	System error. Unable to wrapup with this completion code.
221	Cannot get destinations as callback type is invalid.
222	Unable to get zone of the contact.
223	Unable to get campaigns for the organization.
241	Unable to dial as agent is not in a preview.
242	Unable to cancel as agent is not in a preview.
261	Moving agent to pending not ready state.
281	Conference owner cannot leave the conference.
301	Cannot end the conference as the agent is not in a conference.
302	Cannot end the conference as the agent is not the owner of the conference.
303	Unable to end the conference as passive agent not found.
321	Cannot change ownership as this is not an agent type of conference.

Error Code	Message
341	Cannot send DTMF as invalid data received.
342	Unable to send DTMF as agent is not in a call.
343	System error. Unable to send DTMF.
361	Unable to redial as agent is not in wrapup.
362	Unable to redial as the address is invalid.
382	No destinations available.
383	Agents are not available.
401	System error. Unable to cancel the consult.
402	Unable to cancel the pending consult as it has already started.
421	System error. Unable to consult.
441	System error. Unable to start recording.
461	System error. Unable to stop recording.
481	System error. Unable to stop recording.
482	not found. Unable to create the callback.
483	Unable to create the callback as the agent is not in a call.
484	Unable to create callback as the campaign is not found.
485	Unable to create the callback as invalid callback type received.
486	Unable to create the callback as callback time received in invalid format.
487	Unable to create callback as the expiry time is invalid.
488	Unable to create callback as the callback time is invalid.
501	Unable to extend the wrapup time as agent is not in wrapup state.
502	System error. Unable to find the customer details.
541	Unable to logoff. Please move to not ready state.
561	Invalid data received. Unable to add agent notes.
562	System error. Unable to add agent notes.
563	System error. Unable to add agent notes.
581	Agent is already assigned to Inbound.
601	Agent is already assigned to Outbound.
621	Unable to add to DNC. Contact address already present.
622	System error. Unable to add to DNC.

Error Code	Message
623	Invalid data received. Unable to add to DNC.
624	Invalid address. Unable to add to DNC.
641	Attribute is read only, cannot update this attribute.
642	Invalid value entered. Unable to save attribute.
1001	Unable to find error string for this error code.
9001	Initialization failed as server is not reachable.
9002	Agent is not registered.
9003	System error. Unexpected command received
9004	This request cannot be processed in the current agent state.
9005	System is not available.
9006	Unable to process request as agent is not nailed.
9007	System error. Please check media server.
9008	System error. Invalid data received in the request.
9009	System error. Media server not reachable.
9010	Parameters received in request are less than expected.
9011	System error. Invalid command name received.
9012	Please wait while system is initializing.
9050	System error. Unable to process the request.

Other error codes

Error code	Error message	Description
1001	AGTGetErrorString_Return_Code_Not_Found	Unable to find error string for this error code
9001	INIT_FAILED_SERVER_NOT_REACHABLE	Initialization failed as server is not reachable
9002	AGENT_NOT_REGISTERED	Agent is not registered
9003	AgentAPINotification_NOT_RESP	System error. Unexpected command received
9004	Agent_REQ_RECVD_IN_WRONG_STATE	This request cannot be processed in the current agent state

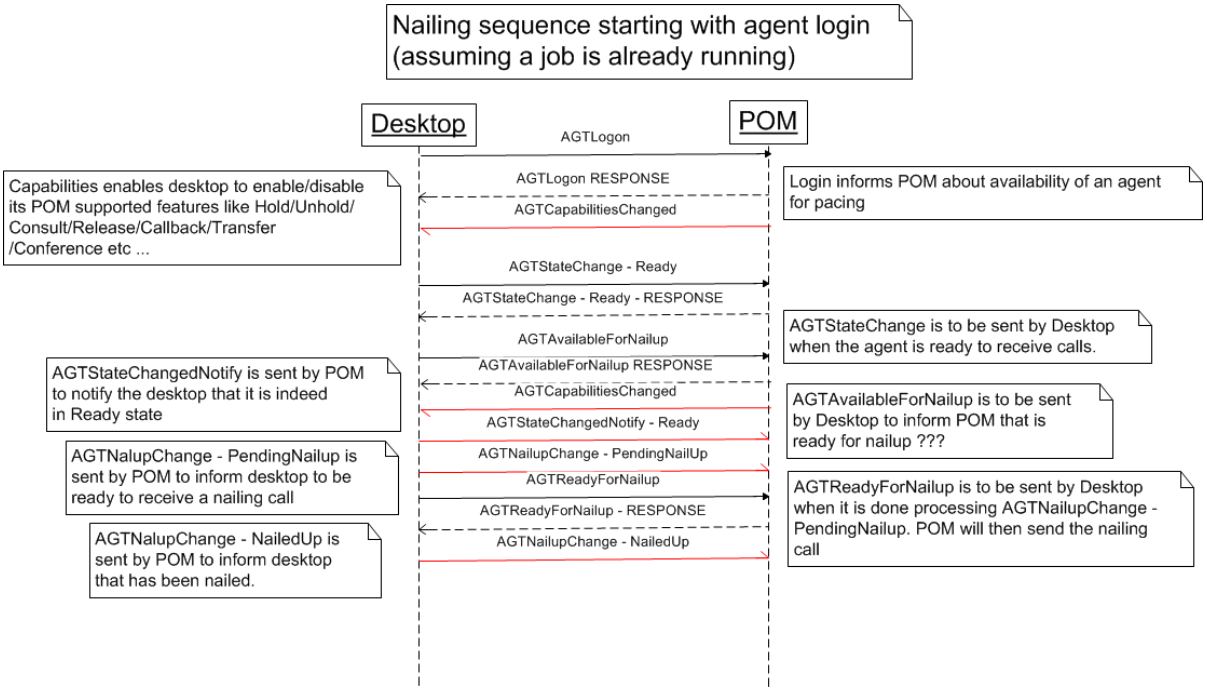
Error code	Error message	Description
9005	POM_NOT_AVAILABLE	System is not available
9006	AGENT_NOT_NAILED	Unable to process the request as agent is not nailed
9007	Telephony_Is_Down	System error. Please check the media server
9008	COMMAND_DATA_INCORRECT	System error. Invalid data received in the request
9009	MPP_Is_Down	System error. Media server not reachable
9010	Incorrect_Parameter_Count	Parameters received in request are less than expected
9011	Incorrect_Command_Name	System error. Invalid command name received
9012	System_Not_Initialized	Please wait while system is initializing
9050	AGENT_GENERIC_ERROR	System error. Unable to process the request

Miscellaneous notes

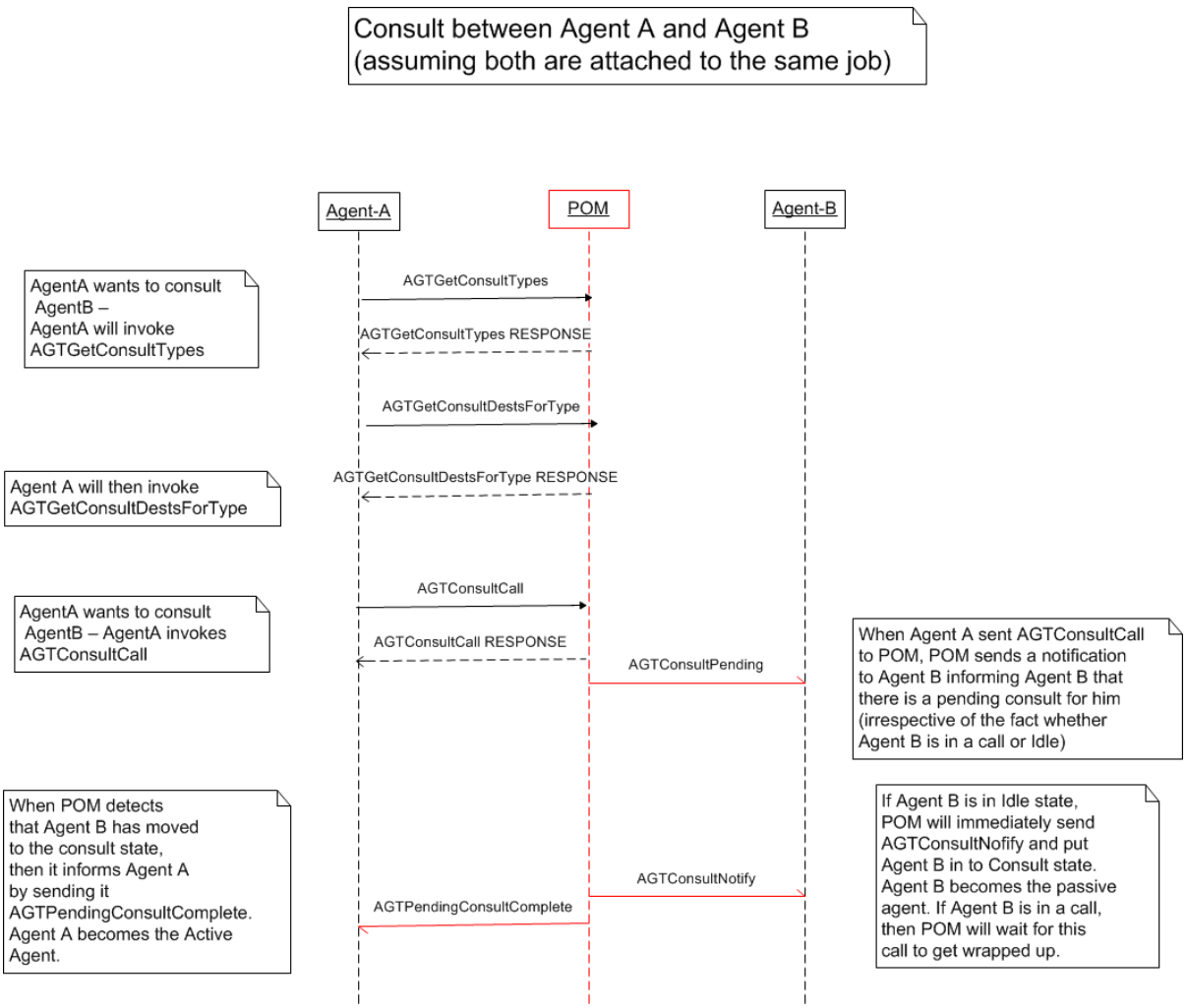
1. Consult is not a capability. So you can use the following logic to control the accessibility of the consult button/menu:
    a. Enable Consult button IF (LEAVECONSULT : FALSE, CANTRANSFER: TRUE, CANCONFERENCE : TRUE)
    b. Disable Consult button IF (LEAVECONSULT : TRUE, CANTRANSFER: TRUE, CANCONFERENCE : TRUE)
2. Callback – Callbacks can be set for Free form number (i.e. a contact number not currently associated with the contact on POM) and in this scenario it is expected the user specifies the string “ExternalNumber” in the contact number object.
3. Agent walkaway – It is expected that the desktop developer track agent actions, such that if for two consecutive calls the agent has not clicked anything on the desktop, then the desktop should send AGTStateChange (NotReady) with the walkaway flag set to true. This will ensure that we POM stops sending calls to the agent if the agent has walked away from his station.
4. It is the customer’s preference to allow/disallow the passive agent involved in a consult/conference to modify/update/work on the desktop. Only relevant buttons should be enabled. The passive party should not be allowed to modify customer details. Only agent notes should be enabled.

Sequence Diagrams

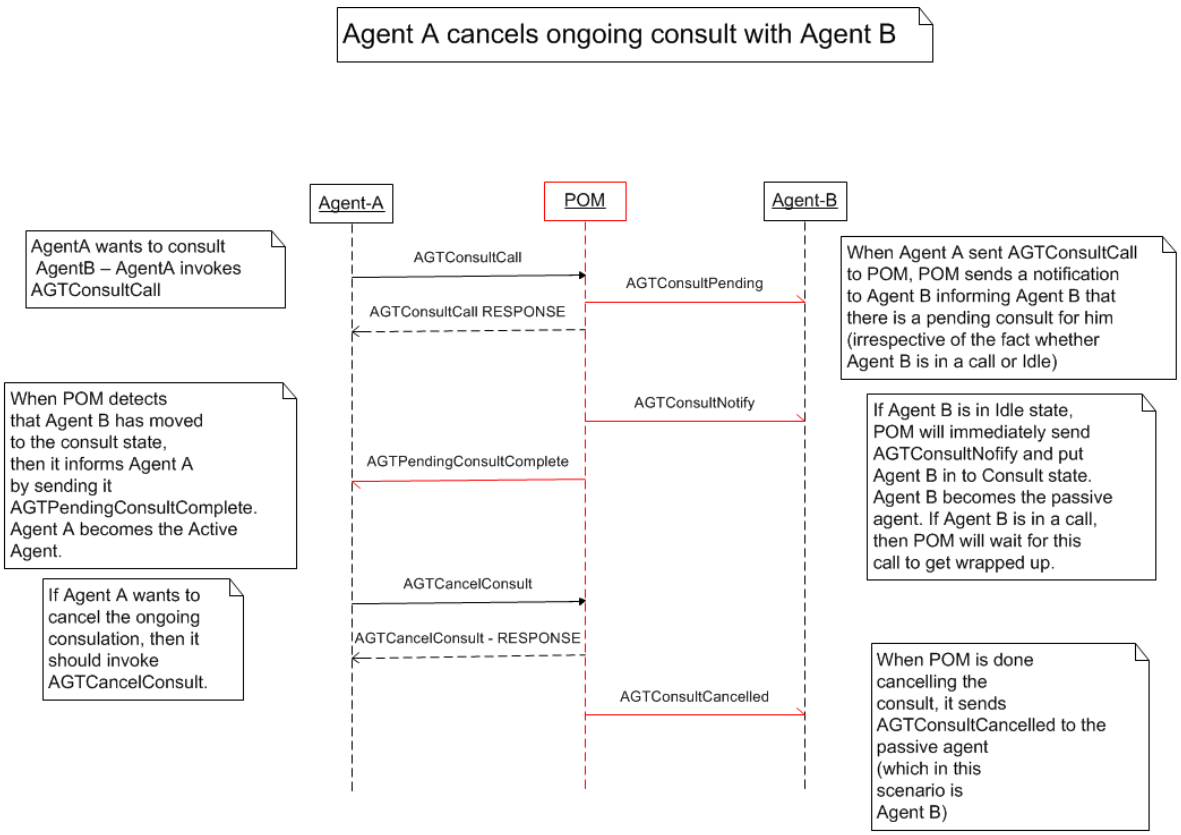
Sequence 1 - Nailing



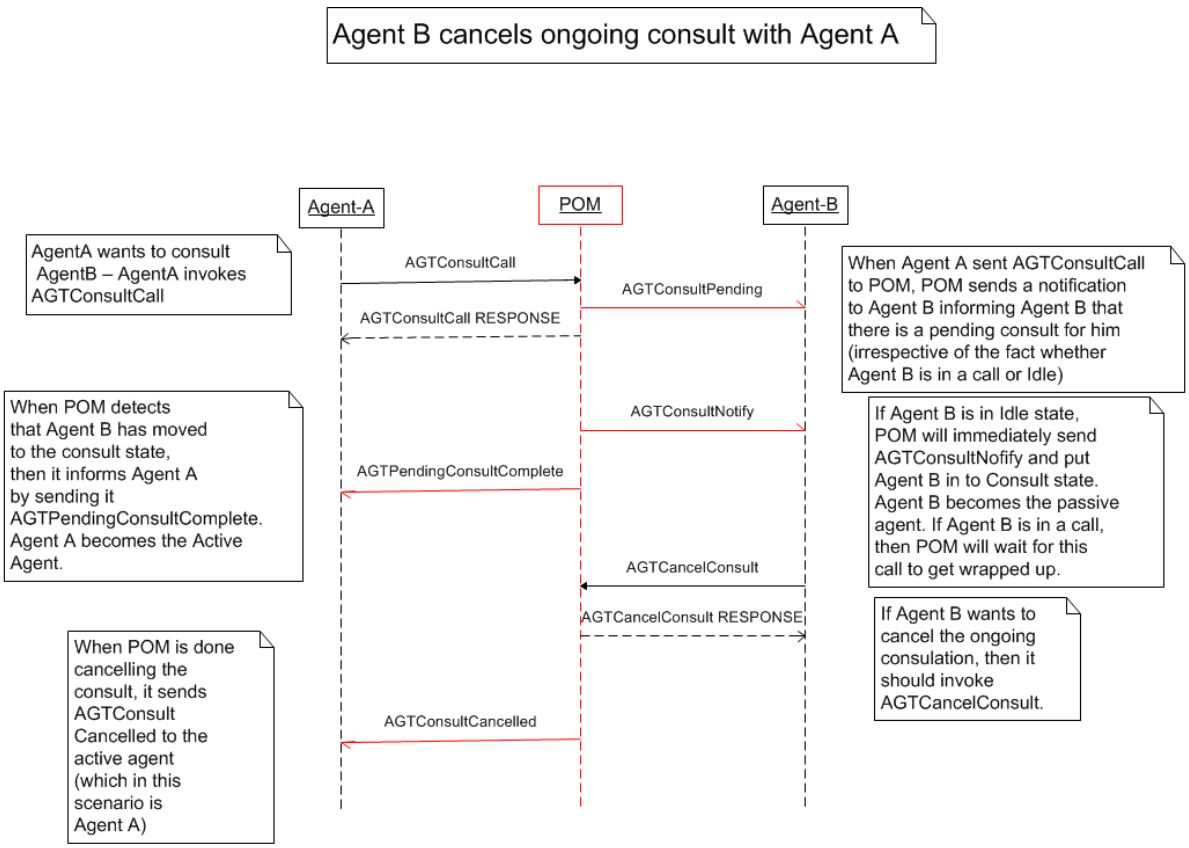
Sequence 2 - Consult



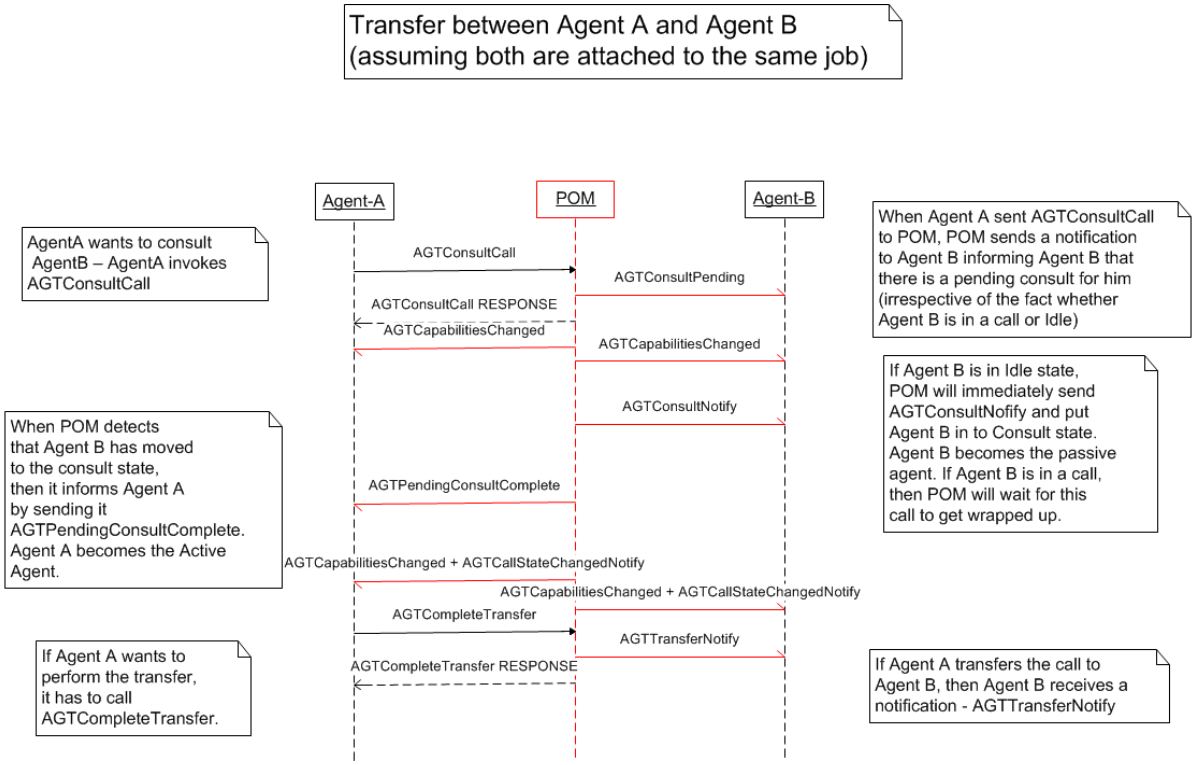
Sequence 3 - Cancel consult by active agent



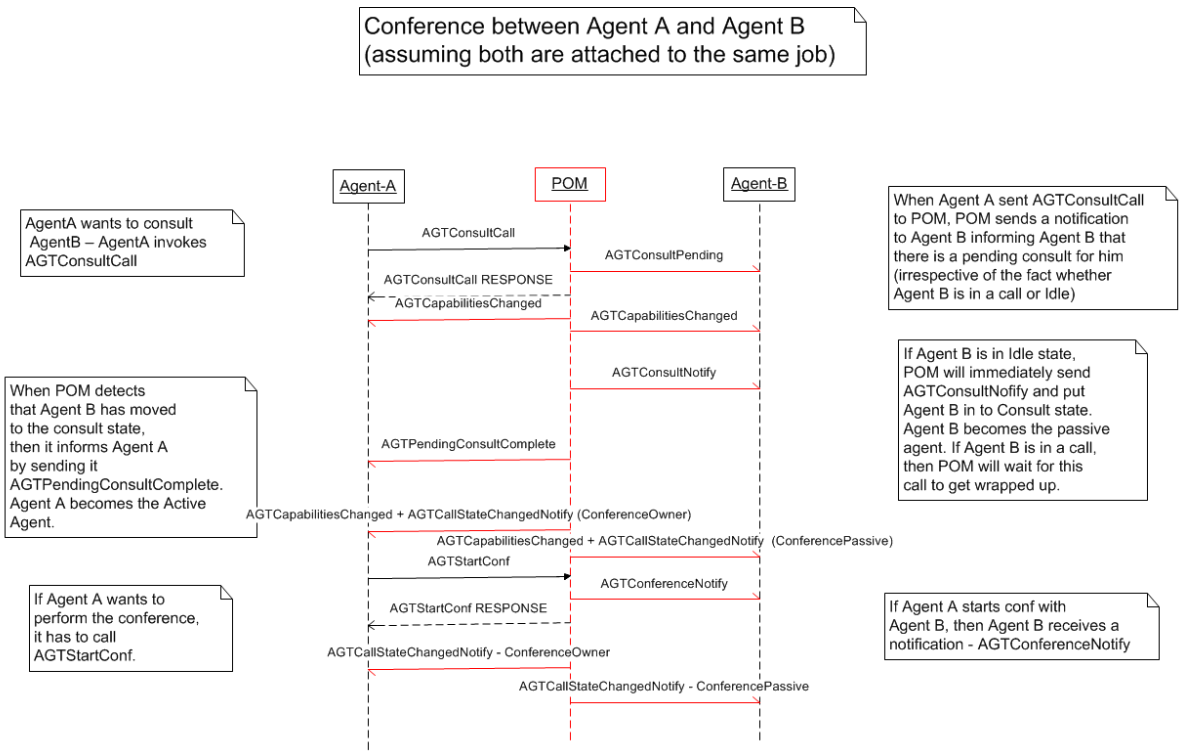
Sequence 4 - Cancel consult by passive agent



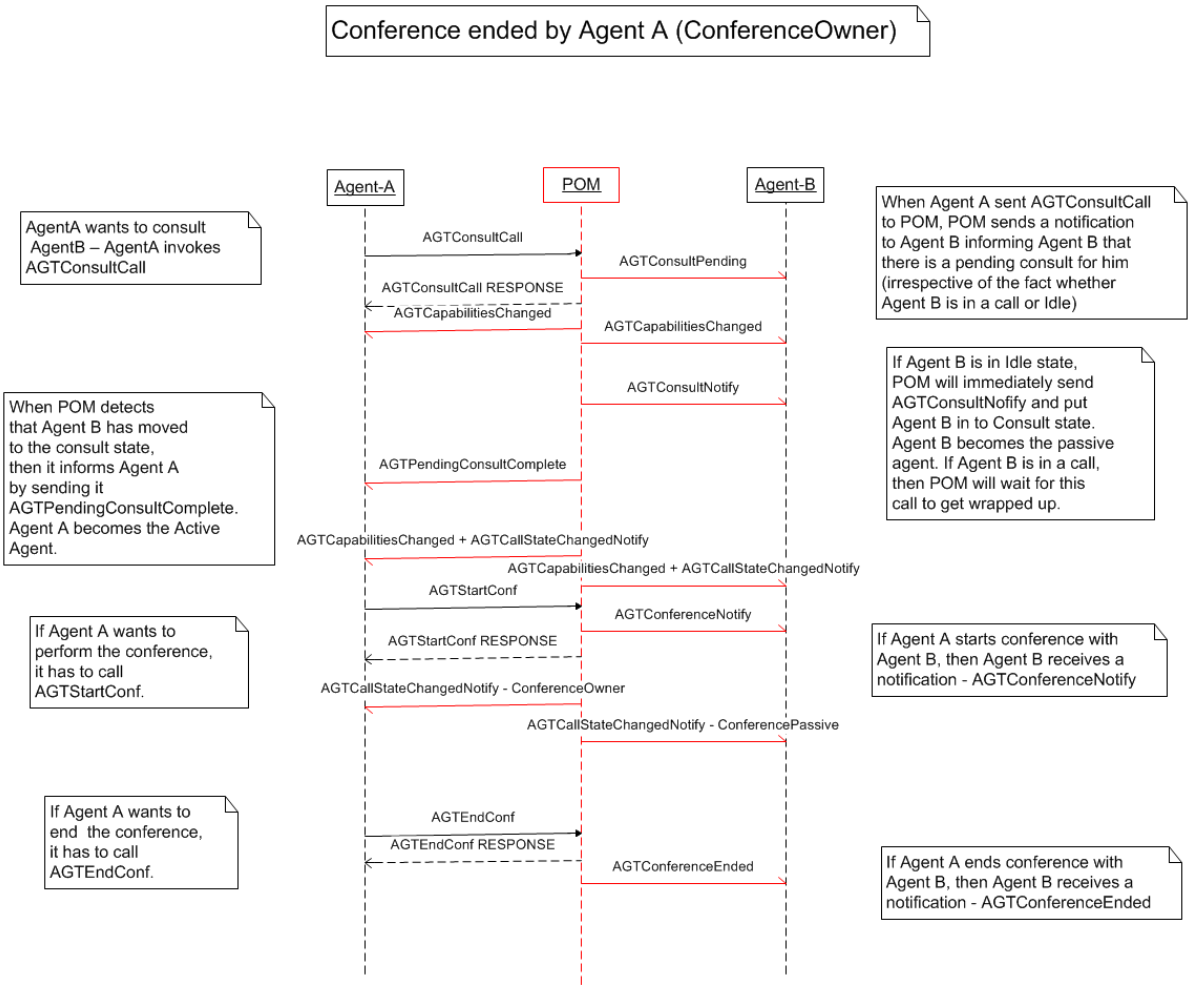
Sequence 5 - Transfer by active agent



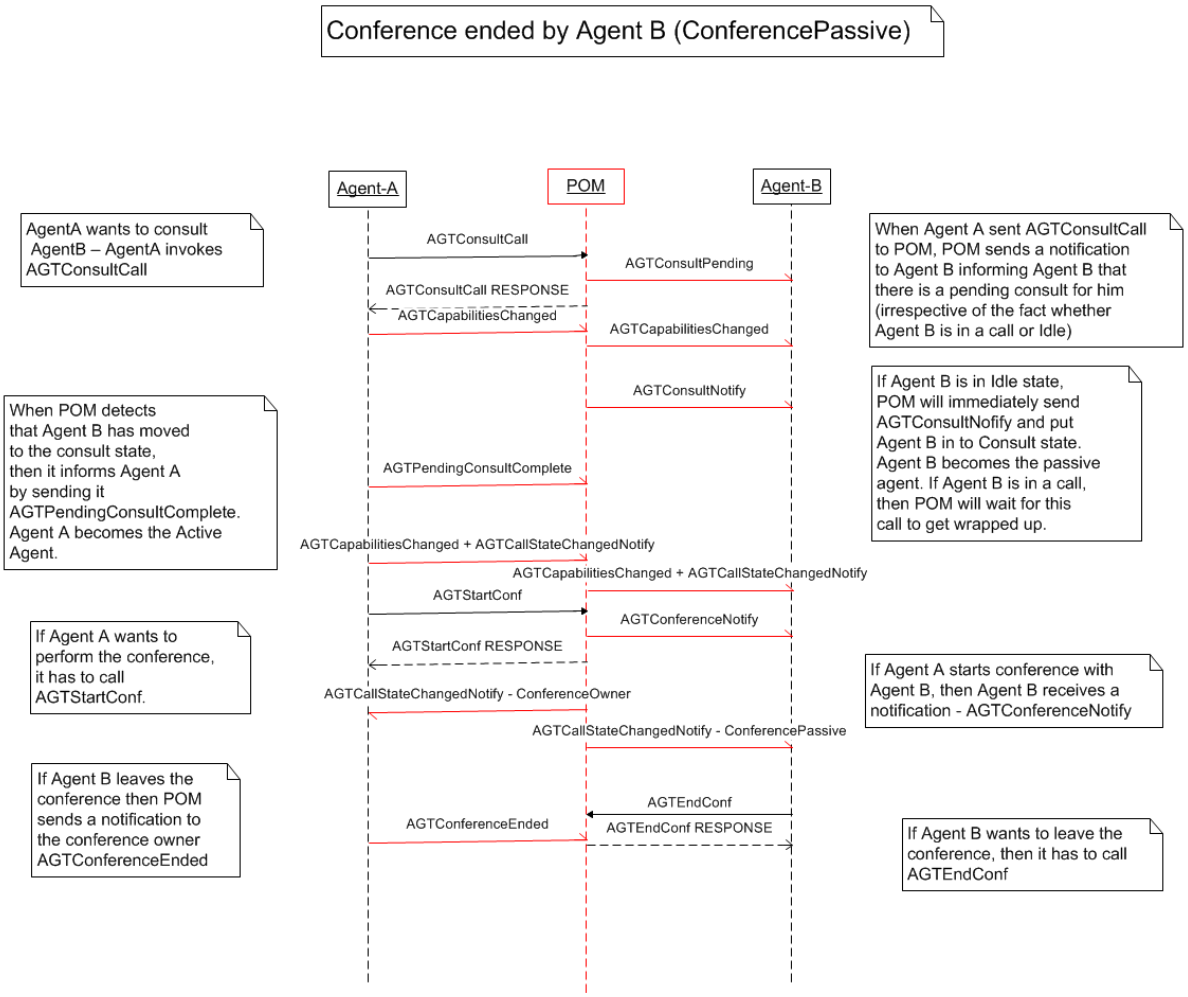
Sequence 6 - Conference by active agent



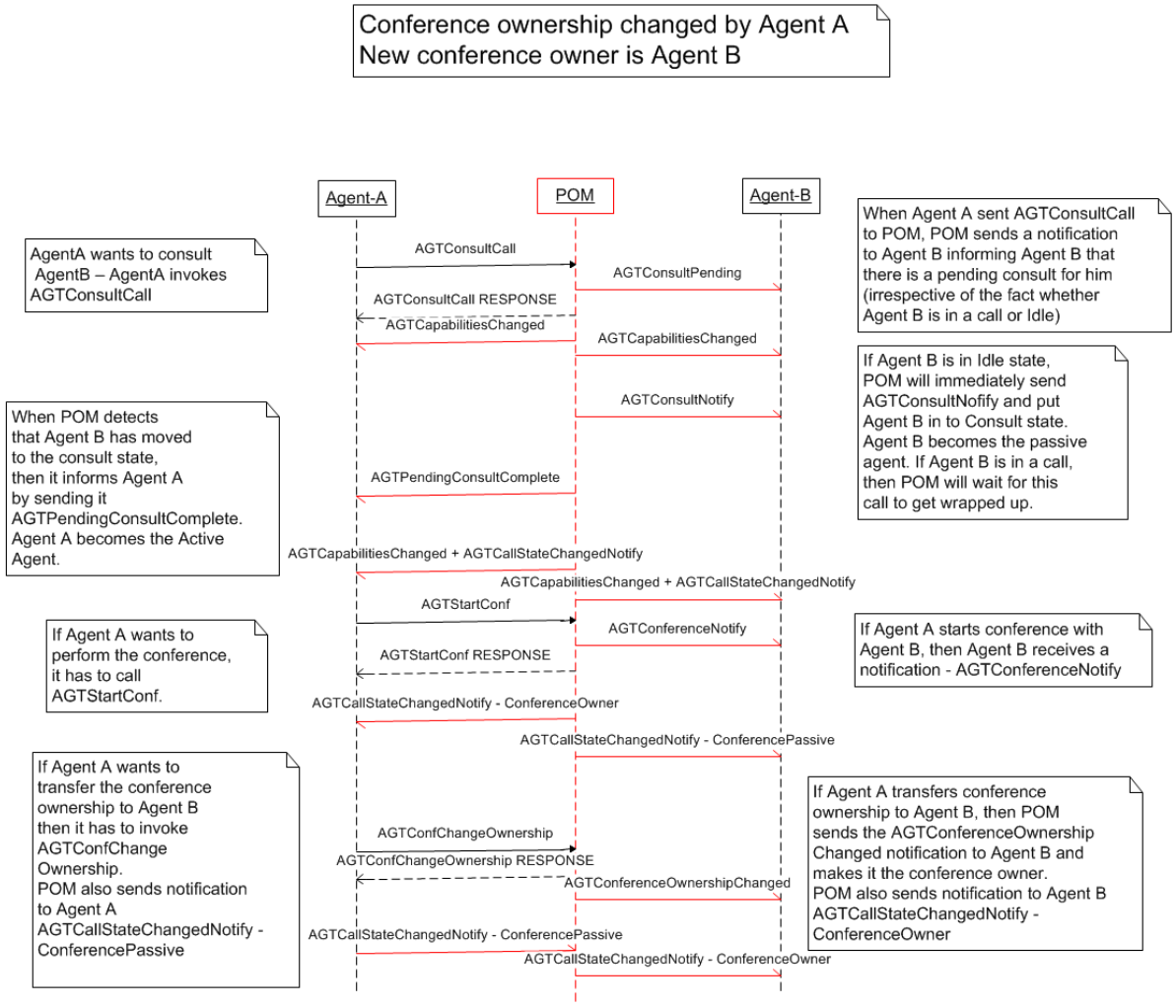
Sequence 7 - Conference end by conference owner



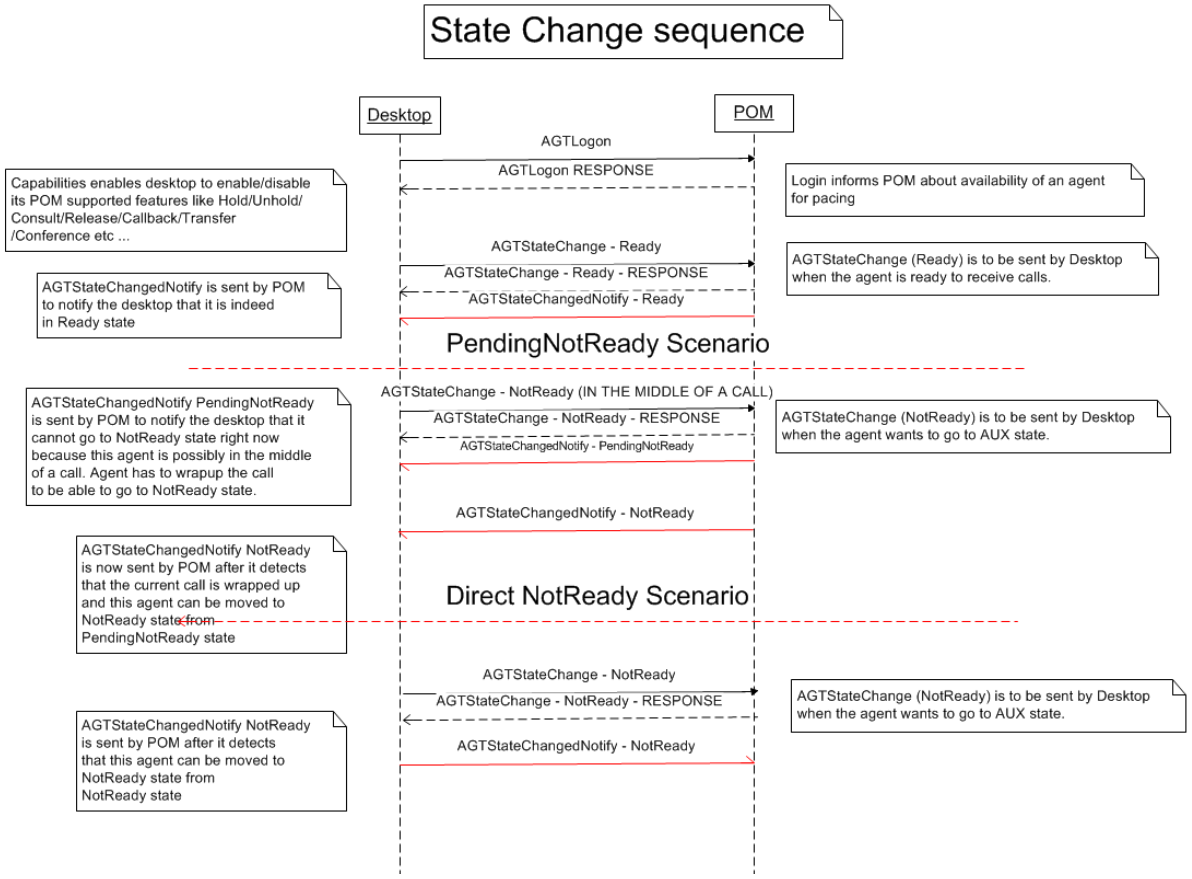
Sequence 8 - Conference left by passive agent in a conference



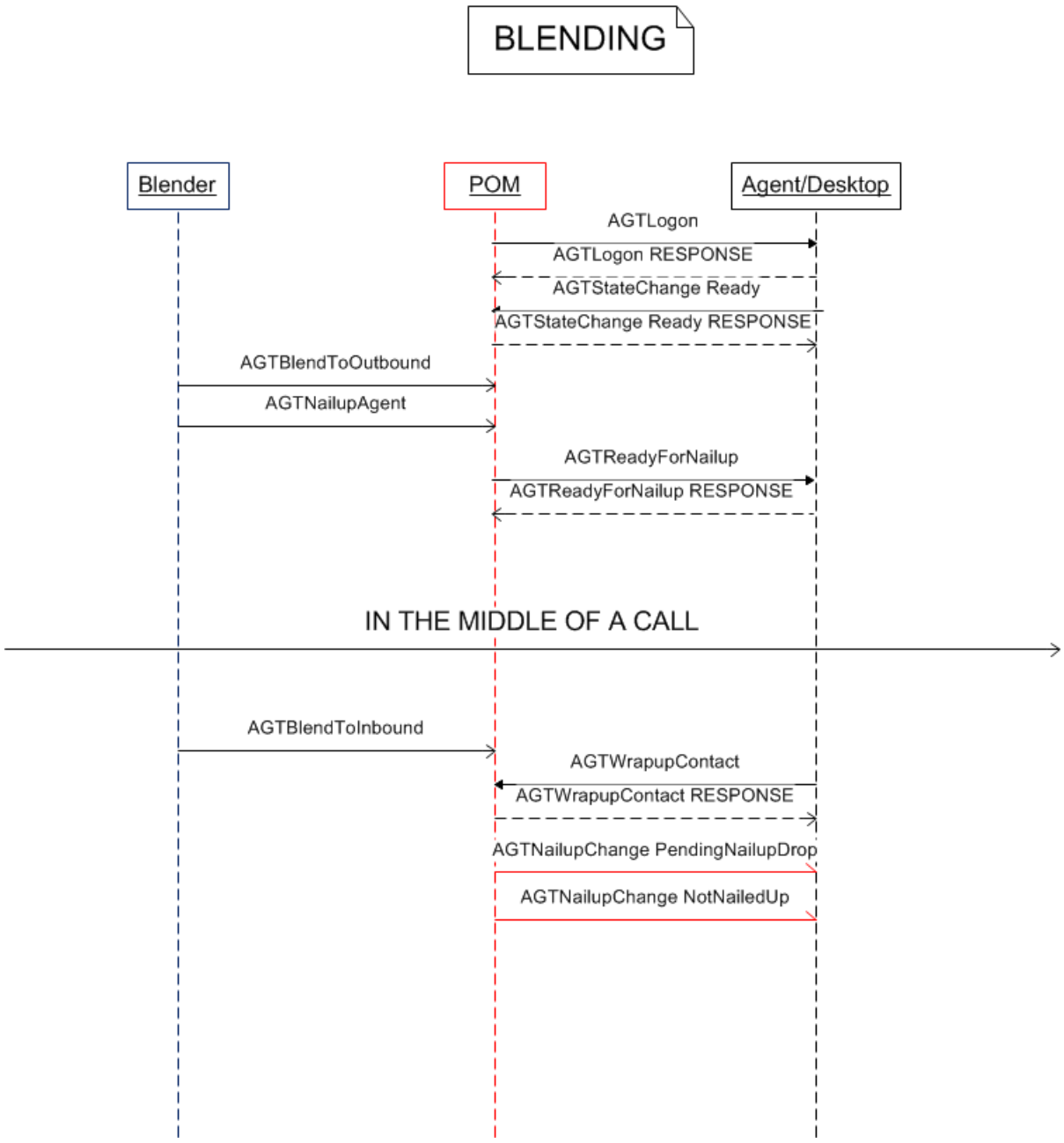
Sequence 9 - Conference ownership changed by conference owner



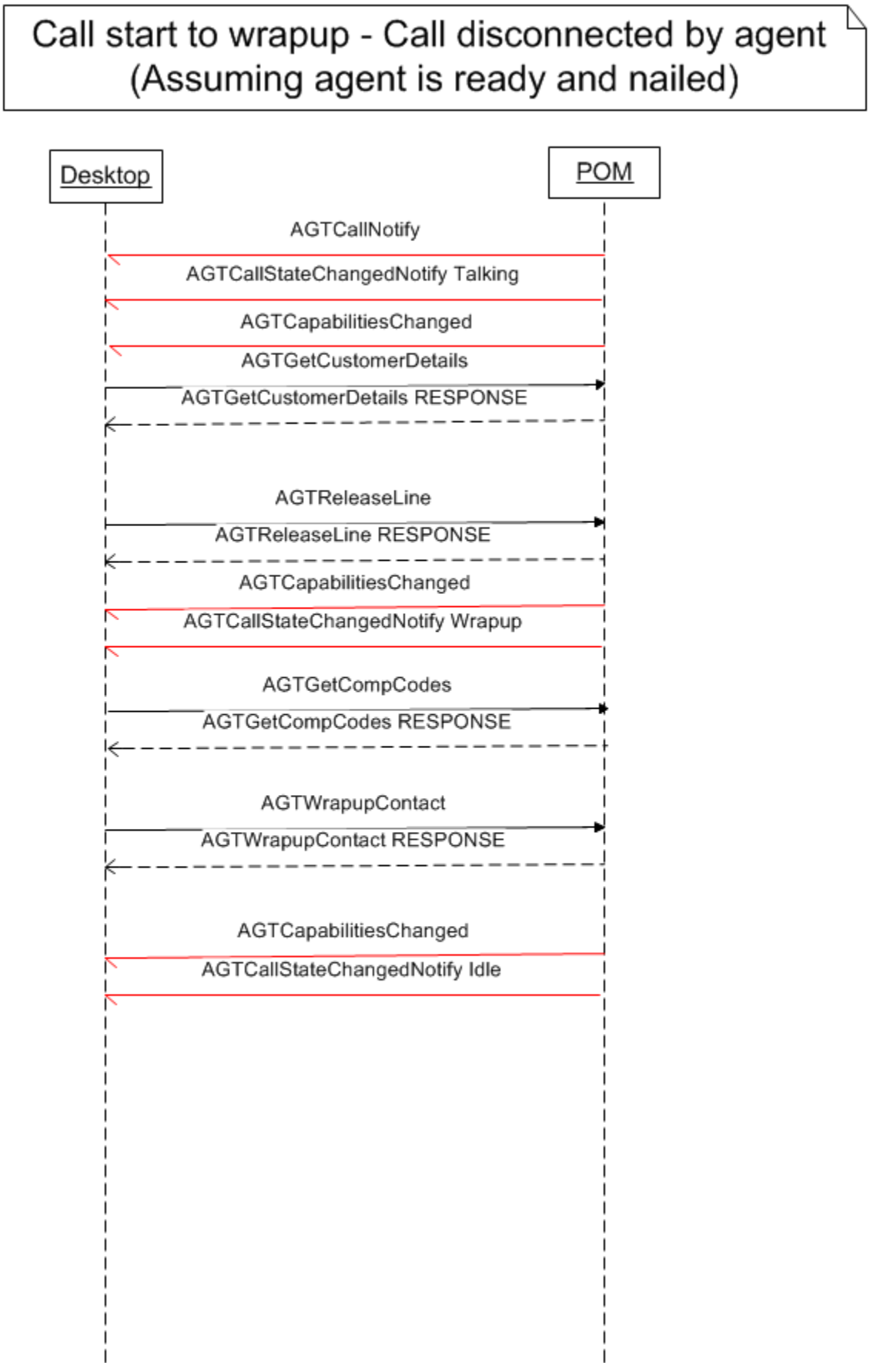
Sequence 10 - State change sequence



Sequence 11 - Blending



Sequence 12 - Call drop by agent


Call start to wrapup - Call disconnected by agent
(Assuming agent is ready and nailed)
Desktop
POM
AGTCallNotify
AGTCallStateChangedNotify Talking
AGTCapabilitiesChanged
AGTGetCustomerDetails
AGTGetCustomerDetails RESPONSE
AGTReleaseLine
AGTReleaseLine RESPONSE
AGTCapabilitiesChanged
AGTCallStateChangedNotify Wrapup
AGTGetCompCodes
AGTGetCompCodes RESPONSE
AGTWrapupContact
AGTWrapupContact RESPONSE
AGTCapabilitiesChanged
AGTCallStateChangedNotify Idle

Sequence 13 — Call drop by customer


Call start to wrapup - Call disconnected by customer
(Assuming agent is ready and nailed)
Desktop
POM
AGTCallNotify
AGTCallStateChangedNotify Talking
AGTCapabilitiesChanged
AGTGetCustomerDetails
AGTGetCustomerDetails RESPONSE
AGTAutoReleaseLine
AGTCapabilitiesChanged
AGTCallStateChangedNotify Wrapup
AGTGetCompCodes
AGTGetCompCodes RESPONSE
AGTWrapupContact
AGTWrapupContact RESPONSE
AGTCapabilitiesChanged
AGTCallStateChangedNotify Idle

Sample Code

Sample code to use the API library

The code is written in VB.net

Initialize the library

```
Dim libInitDone As Boolean = False
libInitDone = POMAgentFactory.init(pamSocketInfoarray)

If libInitDone = False Then
MsgBox("Library could not be initialized. None of the PAM are reachable.")
      loginForm.Visible = True
End If
```

Create agent and log in

```
Private agentHandler As POMAgentHandler
agentHandler = New POMAgentHandler
agent = POMAgentFactory.getPOMAgent(AgentID, agentHandler)

Dim returnCode As Int32 = agent.AGTLogon(AgentExt, Password, forceLogin,
agentLocale, timeZone, zoneName)
```

Log off agent

```
agent.AGTLogoff()
```

Appendix A: Sample WSDL files for Web services

VP_POMCmpMgmt WSDL file

The following is a WSDL file for VP_POMCmpMgmt Web service. The Web service is installed on all the POM servers. For details on accessing the WSDL file, refer to Configuring the VP_POMCmpMgmtService Web service on page 65.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="CmpMgmt" targetNamespace="http://services.pim.avaya.com/
CmpMgmt/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:tns="http://
services.pim.avaya.com/CmpMgmt/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
  <wsdl:types>
    <xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://services.pim.avaya.com/CmpMgmt/">
      <xsd:element name="GetActiveJobs">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="CampaignName" type="xsd:string"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetActiveJobsResponse">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element maxOccurs="unbounded" minOccurs="0" name="Jobs"
type="xsd:int"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetActiveJobsFaultInfo">
          <xsd:complexType>
              <xsd:sequence>

                  <xsd:element name="RetCode" type="xsd:int">
                  </xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string"/>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="SetMaxAttemptsCount">
          <xsd:complexType>
              <xsd:sequence>

                  <xsd:element name="JobID" type="xsd:int" />
                  <xsd:element name="Count" type="xsd:int" />
              </xsd:sequence>
```

```
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SetMaxAttemptsCountResponse">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="Result" type="xsd:int"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SetMaxAttemptsCountFaultInfo">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="RetCode" type="xsd:int">
                </xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="PauseActiveJob">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="JobID" type="xsd:int"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="PauseActiveJobResponse">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="IsPaused" type="xsd:boolean"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="PauseActiveJobFaultInfo">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="RetCode" type="xsd:int">
                </xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ResumePausedJob">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="JobID" type="xsd:int"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ResumePausedJobResponse">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="IsResumed" type="xsd:boolean"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ResumePausedJobFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
```

```
                <xsd:element name="RetCode" type="xsd:int">
                </xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="StopJob">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="JobID" type="xsd:int"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="StopJobResponse">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="IsStopped" type="xsd:boolean"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="StopJobFaultInfo">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="RetCode" type="xsd:int"/>
                <xsd:element name="FaultMsg" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="RunCampaign">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="CampaignName" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="RunCampaignResponse">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="IsQueued" type="xsd:boolean"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="RunCampaignFaultInfo">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="RetCode" type="xsd:int">
                </xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetCampaignJobs">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="CampaignName" type="xsd:string"/>
                <xsd:element maxOccurs="unbounded" minOccurs="0" name="JobState"
type="xsd:string"/>
```

```
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetCampaignJobsResponse">
        <xsd:complexType>
            <xsd:sequence>

              <xsd:element maxOccurs="unbounded" minOccurs="0" name="JobDetails"
type="tns:JobIDsAndStates"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetCampaignJobsFaultInfo">
        <xsd:complexType>
            <xsd:sequence>

                <xsd:element name="RetCode" type="xsd:int">
                </xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:complexType name="JobIDsAndStates">
        <xsd:sequence>
            <xsd:element name="JobID" type="xsd:int"/>
            <xsd:element name="JobState" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="GetActiveJobTaskIds">
        <xsd:complexType>
            <xsd:sequence>

                    <xsd:element name="CampaignName" type="xsd:string"></
xsd:element>
                </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetActiveJobTaskIdsResponse">
        <xsd:complexType>
            <xsd:sequence>

                  <xsd:element name="JobAndTaskDetails" type="tns:JobIDAndTask"
minOccurs="0" maxOccurs="unbounded"></xsd:element>
                </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:complexType name="JobIDAndTask">
        <xsd:sequence>
            <xsd:element name="JobID" type="xsd:int"></xsd:element>
            <xsd:element name="TaskID" type="xsd:int"></xsd:element>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="GetActiveJobTaskIdsFaultInfo">
        <xsd:complexType>
            <xsd:sequence>

                    <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                    <xsd:element name="FaultMsg" type="xsd:string">
                </xsd:element>
                  </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SetMaxAttemptsCountForTask">
```

```
          <xsd:complexType>
              <xsd:sequence>

                  <xsd:element name="JobID" type="xsd:int"></xsd:element>
                        <xsd:element name="TaskID" type="xsd:int"></xsd:element>
                        <xsd:element name="Count" type="xsd:int"></xsd:element>
                    </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="SetMaxAttemptsCountForTaskResponse">
          <xsd:complexType>
              <xsd:sequence>

                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="SetMaxAttemptsCountForTaskFaultInfo">
          <xsd:complexType>
              <xsd:sequence>

                        <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                        <xsd:element name="FaultMsg" type="xsd:string">
                  </xsd:element>
                    </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetActiveJobTaskIdForTask">
          <xsd:complexType>
              <xsd:sequence>

                  <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
                  <xsd:element name="TaskName" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetActiveJobTaskIdForTaskResponse">
          <xsd:complexType>
              <xsd:sequence>

                  <xsd:element name="ActiveJobsAndTaskDetails"
type="tns:JobIDAndTask" minOccurs="0" maxOccurs="unbounded"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetActiveJobTaskIdForTaskFaultInfo">
          <xsd:complexType>
              <xsd:sequence>

                        <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                        <xsd:element name="FaultMsg" type="xsd:string">
                  </xsd:element>
                    </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetCampaignDetails">
          <xsd:complexType>

              <xsd:sequence>
                  <xsd:element name="JobState" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetCampaignDetailsResponse">
```

```
          <xsd:complexType>
              <xsd:sequence>

                  <xsd:element name="CampaignDetails"
type="tns:CampaignNameJobIDsAndStates" minOccurs="0" maxOccurs="unbounded"></
xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetCampaignDetailsFaultInfo">
          <xsd:complexType>
              <xsd:sequence>

                        <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                      <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>

      <xsd:complexType name="CampaignNameJobIDsAndStates">
          <xsd:sequence>
                   <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
                     <xsd:element name="JobID" type="xsd:int" minOccurs="0"
maxOccurs="1"></xsd:element>
                     <xsd:element name="JobState" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
          </xsd:sequence>
      </xsd:complexType>
      <xsd:element name="ScheduleCampaign">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
                  <xsd:element name="StartTime" type="xsd:string"></xsd:element>
                  <xsd:element name="TimeZone" type="xsd:string"></xsd:element>
                  <xsd:element name="ArchivalScheduleFrequency"
type="tns:ArchivalFrequencyType" default="Hourly" minOccurs="0" maxOccurs="1">
                  </xsd:element>
                  <xsd:element name="ArchivalTimeInHrsMins" type="xsd:string"
minOccurs="0" maxOccurs="1">
                  </xsd:element>
                  <xsd:element name="ArchivalInNHours" type="xsd:string"
minOccurs="0" maxOccurs="1"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="ScheduleCampaignResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="ScheduleCampaignFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string">
                  </xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="ScheduleRecurringCampaign">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
```

```
                    <xsd:element name="StartTime" type="xsd:string"></xsd:element>
                    <xsd:element name="EndTime" type="xsd:string"></xsd:element>
                    <xsd:element name="TimeZone" type="xsd:string"></xsd:element>
                    <xsd:element name="ScheduleFrequency"
                        type="tns:RecurringFrequencyType">
                    </xsd:element>
                    <xsd:element name="WeekDaysOnly" type="xsd:boolean"
                        minOccurs="0" maxOccurs="1" default="false">
                    </xsd:element>
                    <xsd:element name="SelectedDays" type="xsd:string"
                        minOccurs="0" maxOccurs="unbounded">
                    </xsd:element>
                    <xsd:element name="RunEveryMinutes" type="xsd:string"
                        minOccurs="0" maxOccurs="1">
                    </xsd:element>
                    <xsd:element name="ArchivalScheduleFrequency"
                        type="tns:ArchivalFrequencyType" minOccurs="0"
                        maxOccurs="1">
                    </xsd:element>
                    <xsd:element name="ArchivalInNHours" type="xsd:string"
                        minOccurs="0" maxOccurs="1">
                    </xsd:element>
                    <xsd:element name="ArchivalTimeInHrsMins" type="xsd:string"
minOccurs="0" maxOccurs="1"></xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ScheduleRecurringCampaignResponse">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="Result" type="xsd:int"></xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ScheduleRecurringCampaignFaultInfo">
            <xsd:complexType>
                <xsd:sequence>
                          <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                          <xsd:element name="FaultMsg" type="xsd:string">
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ScheduleDataSource">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="DataSourceName"
                        type="xsd:string">
                    </xsd:element>
                    <xsd:element name="StartTime" type="xsd:string"></xsd:element>
                    <xsd:element name="TimeZone" type="xsd:string"></xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ScheduleDataSourceResponse">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="Result" type="xsd:int"></xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ScheduleDataSourceFaultInfo">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="RetCode" type="xsd:int"></xsd:element>
```

```
                    <xsd:element name="FaultMsg" type="xsd:string">
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ScheduleRecurringDataSource">
            <xsd:complexType>
                <xsd:sequence>
                   <xsd:element name="DataSourceName" type="xsd:string"></xsd:element>
                    <xsd:element name="StartTime" type="xsd:string"></xsd:element>
                    <xsd:element name="EndTime" type="xsd:string"></xsd:element>
                    <xsd:element name="TimeZone" type="xsd:string"></xsd:element>
                    <xsd:element name="ScheduleFrequency"
                        type="xsd:string">
                    </xsd:element>
                    <xsd:element name="WeekDaysOnly" type="xsd:boolean"
                        minOccurs="0" maxOccurs="1" default="false">
                    </xsd:element>
                    <xsd:element name="SelectedDays" type="xsd:string"
                        minOccurs="0" maxOccurs="unbounded">
                    </xsd:element>
                  <xsd:element name="RunEveryMinutes" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ScheduleRecurringDataSourceResponse">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="Result" type="xsd:int"></xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ScheduleRecurringDataSourceFaultInfo">
            <xsd:complexType>
                <xsd:sequence>
                          <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                          <xsd:element name="FaultMsg" type="xsd:string">
                    </xsd:element>
                      </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="GetImportJobStatus">
            <xsd:complexType>
                <xsd:sequence>
                   <xsd:element name="DataSourceName" type="xsd:string"></xsd:element>
                    <xsd:element name="JobStates" type="tns:ImportJobState"
default="RUNNING" minOccurs="0" maxOccurs="unbounded">
                    </xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="GetImportJobStatusResponse">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="ImportJobs" type="tns:ImportJobStatus"
minOccurs="0" maxOccurs="unbounded"></xsd:element>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="GetImportJobStatusFaultInfo">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                    <xsd:element name="FaultMsg" type="xsd:string">
```

```
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>

    <xsd:complexType name="ImportJobStatus">
        <xsd:sequence>
            <xsd:element name="ImportName" type="xsd:string"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="ListName" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
            <xsd:element name="ImportJobId" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="Status" type="tns:ImportJobState" minOccurs="0"
                maxOccurs="1">
            </xsd:element>
            <xsd:element name="SuccessCount" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="UpdateCount" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="RunTimeErrorCount" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="ValidationFailedCount" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="DuplicateIgnoredCount" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="MatchPhoneRejectPatternCount"
                type="xsd:long" minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="DeleteCount" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="MatchesDncCount" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="PhoneFormatFailedCount" type="xsd:long"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
           <xsd:element name="ProcessedRecordCount" type="xsd:long" minOccurs="0"
maxOccurs="1"></xsd:element>
        </xsd:sequence>
    </xsd:complexType>

    <xsd:simpleType name="ImportJobState">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="COMPLETED"></xsd:enumeration>
            <xsd:enumeration value="QUEUED"></xsd:enumeration>
            <xsd:enumeration value="RUNNING"></xsd:enumeration>
            <xsd:enumeration value="ERROR"></xsd:enumeration>
            <xsd:enumeration value="FILE_COPYING"></xsd:enumeration>
            <xsd:enumeration value="PAUSING"></xsd:enumeration>
            <xsd:enumeration value="PAUSED"></xsd:enumeration>
            <xsd:enumeration value="STOPPING"></xsd:enumeration>
            <xsd:enumeration value="WAITING_TO_RESUME"></xsd:enumeration>
            <xsd:enumeration value="DELETING_CONTACTS"></xsd:enumeration>
            <xsd:enumeration value="CREATING_HISTORY"></xsd:enumeration>
        </xsd:restriction>
    </xsd:simpleType>
```

```
      <xsd:simpleType name="ArchivalFrequencyType">
          <xsd:restriction base="xsd:string">
              <xsd:enumeration value="Hourly"></xsd:enumeration>
              <xsd:enumeration value="RunEveryNHours"></xsd:enumeration>
              <xsd:enumeration value="DailyAt"></xsd:enumeration>
          </xsd:restriction>
      </xsd:simpleType>

      <xsd:simpleType name="RecurringFrequencyType">
          <xsd:restriction base="xsd:string">
              <xsd:enumeration value="RunEveryNMins"></xsd:enumeration>
              <xsd:enumeration value="Daily"></xsd:enumeration>
              <xsd:enumeration value="Weekly"></xsd:enumeration>
              <xsd:enumeration value="Monthly"></xsd:enumeration>
              <xsd:enumeration value="Yearly"></xsd:enumeration>
          </xsd:restriction>
      </xsd:simpleType>
      <xsd:element name="GetContactListNames">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactListNamesResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ListNames" type="xsd:string" minOccurs="0"
maxOccurs="unbounded"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactListNamesFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                      <xsd:element name="FaultMsg" type="xsd:string">
                  </xsd:element>
                        <xsd:element name="RetCode" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="ResumePausedJobRequest">
    <wsdl:part name="parameters" element="tns:ResumePausedJob">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="StopJobResponse">
    <wsdl:part name="parameters" element="tns:StopJobResponse">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetActiveJobsResponse">
    <wsdl:part name="parameters" element="tns:GetActiveJobsResponse">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="SetMaxAttemptsCountFault">
    <wsdl:part name="parameters" element="tns:SetMaxAttemptsCountFaultInfo">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetActiveJobsRequest">
    <wsdl:part name="parameters" element="tns:GetActiveJobs">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetActiveJobsFault">
```

```
    <wsdl:part name="parameters" element="tns:GetActiveJobsFaultInfo">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="ResumePausedJobFault">
    <wsdl:part name="parameters" element="tns:ResumePausedJobFaultInfo">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignJobsRequest">
    <wsdl:part name="parameters" element="tns:GetCampaignJobs">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="PauseActiveJobFault">
    <wsdl:part name="parameters" element="tns:PauseActiveJobFaultInfo">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="SetMaxAttemptsCountResponse">
    <wsdl:part name="parameters" element="tns:SetMaxAttemptsCountResponse">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="RunCampaignResponse">
    <wsdl:part name="parameters" element="tns:RunCampaignResponse">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignJobsResponse">
    <wsdl:part name="parameters" element="tns:GetCampaignJobsResponse">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="StopJobRequest">
    <wsdl:part name="parameters" element="tns:StopJob">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="RunCampaignRequest">
    <wsdl:part name="parameters" element="tns:RunCampaign">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="PauseActiveJobResponse">
    <wsdl:part name="parameters" element="tns:PauseActiveJobResponse">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="PauseActiveJobRequest">
    <wsdl:part name="parameters" element="tns:PauseActiveJob">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="StopJobFault">
    <wsdl:part name="parameters" element="tns:StopJobFaultInfo">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="SetMaxAttemptsCountRequest">
    <wsdl:part name="parameters" element="tns:SetMaxAttemptsCount">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignJobsFault">
    <wsdl:part name="parameters" element="tns:GetCampaignJobsFaultInfo">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="ResumePausedJobResponse">
    <wsdl:part name="parameters" element="tns:ResumePausedJobResponse">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="RunCampaignFault">
    <wsdl:part name="parameters" element="tns:RunCampaignFaultInfo">
    </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetActiveJobTaskIdsRequest">
      <wsdl:part name="parameters" element="tns:GetActiveJobTaskIds"></wsdl:part>
```

```
  </wsdl:message>
  <wsdl:message name="GetActiveJobTaskIdsResponse">
      <wsdl:part name="parameters"
          element="tns:GetActiveJobTaskIdsResponse">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetActiveJobTaskIdsFault">
      <wsdl:part name="parameters"
          element="tns:GetActiveJobTaskIdsFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="SetMaxAttemptsCountForTaskRequest">
      <wsdl:part name="parameters" element="tns:SetMaxAttemptsCountForTask"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="SetMaxAttemptsCountForTaskResponse">
      <wsdl:part name="parameters"
element="tns:SetMaxAttemptsCountForTaskResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="SetMaxAttemptsCountForTaskFault">
      <wsdl:part name="parameters"
          element="tns:SetMaxAttemptsCountForTaskFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetActiveJobTaskIdForTaskRequest">
      <wsdl:part name="parameters" element="tns:GetActiveJobTaskIdForTask"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetActiveJobTaskIdForTaskResponse">
      <wsdl:part name="parameters"
element="tns:GetActiveJobTaskIdForTaskResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetActiveJobTaskIdForTaskFault">
      <wsdl:part name="parameters"
          element="tns:GetActiveJobTaskIdForTaskFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignDetailsRequest">
      <wsdl:part name="parameters" element="tns:GetCampaignDetails"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignDetailsResponse">
      <wsdl:part name="parameters" element="tns:GetCampaignDetailsResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignDetailsFault">
      <wsdl:part name="parameters" element="tns:GetCampaignDetailsFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleCampaignRequest">
      <wsdl:part name="parameters" element="tns:ScheduleCampaign"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleCampaignResponse">
      <wsdl:part name="parameters" element="tns:ScheduleCampaignResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleCampaignFault">
      <wsdl:part name="parameters" element="tns:ScheduleCampaignFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleRecurringCampaignRequest">
      <wsdl:part name="parameters" element="tns:ScheduleRecurringCampaign"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleRecurringCampaignResponse">
      <wsdl:part name="parameters"
```

```
element="tns:ScheduleRecurringCampaignResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleRecurringCampaignFault">
      <wsdl:part name="parameters"
element="tns:ScheduleRecurringCampaignFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleDataSourceRequest">
      <wsdl:part name="parameters" element="tns:ScheduleDataSource"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleDataSourceResponse">
      <wsdl:part name="parameters" element="tns:ScheduleDataSourceResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleDataSourceFault">
      <wsdl:part name="parameters" element="tns:ScheduleDataSourceFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleRecurringDataSourceRequest">
      <wsdl:part name="parameters" element="tns:ScheduleRecurringDataSource"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleRecurringDataSourceResponse">
      <wsdl:part name="parameters"
element="tns:ScheduleRecurringDataSourceResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleRecurringDataSourceFault">
      <wsdl:part name="parameters"
element="tns:ScheduleRecurringDataSourceFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetImportJobStatusRequest">
      <wsdl:part name="parameters" element="tns:GetImportJobStatus"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetImportJobStatusResponse">
      <wsdl:part name="parameters" element="tns:GetImportJobStatusResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetImportJobStatusFault">
      <wsdl:part name="parameters"
          element="tns:GetImportJobStatusFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactListNamesRequest">
      <wsdl:part name="parameters" element="tns:GetContactListNames"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactListNamesResponse">
      <wsdl:part name="parameters" element="tns:GetContactListNamesResponse">
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactListNamesFault">
      <wsdl:part name="parameters" element="tns:GetContactListNamesFaultInfo">
wsdl:part>
  </wsdl:message>
  <wsdl:portType name="VP_POMCmpMgmtService">
    <wsdl:operation name="GetActiveJobs">
      <wsdl:input message="tns:GetActiveJobsRequest">
    </wsdl:input>
      <wsdl:output message="tns:GetActiveJobsResponse">
    </wsdl:output>
      <wsdl:fault name="fault" message="tns:GetActiveJobsFault">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="SetMaxAttemptsCount">
      <wsdl:input message="tns:SetMaxAttemptsCountRequest">
    </wsdl:input>
      <wsdl:output message="tns:SetMaxAttemptsCountResponse">
```

```
    </wsdl:output>
      <wsdl:fault name="fault" message="tns:SetMaxAttemptsCountFault">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="PauseActiveJob">
      <wsdl:input message="tns:PauseActiveJobRequest">
    </wsdl:input>
      <wsdl:output message="tns:PauseActiveJobResponse">
    </wsdl:output>
      <wsdl:fault name="fault" message="tns:PauseActiveJobFault">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="ResumePausedJob">
      <wsdl:input message="tns:ResumePausedJobRequest">
    </wsdl:input>
      <wsdl:output message="tns:ResumePausedJobResponse">
    </wsdl:output>
      <wsdl:fault name="fault" message="tns:ResumePausedJobFault">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="StopJob">
      <wsdl:input message="tns:StopJobRequest">
    </wsdl:input>
      <wsdl:output message="tns:StopJobResponse">
    </wsdl:output>
      <wsdl:fault name="fault" message="tns:StopJobFault">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="RunCampaign">
      <wsdl:input message="tns:RunCampaignRequest">
    </wsdl:input>
      <wsdl:output message="tns:RunCampaignResponse">
    </wsdl:output>
      <wsdl:fault name="fault" message="tns:RunCampaignFault">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="GetCampaignJobs">
      <wsdl:input message="tns:GetCampaignJobsRequest">
    </wsdl:input>
      <wsdl:output message="tns:GetCampaignJobsResponse">
    </wsdl:output>
      <wsdl:fault name="fault" message="tns:GetCampaignJobsFault">
    </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="GetActiveJobTaskIds">
        <wsdl:input message="tns:GetActiveJobTaskIdsRequest"></wsdl:input>
        <wsdl:output message="tns:GetActiveJobTaskIdsResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:GetActiveJobTaskIdsFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="SetMaxAttemptsCountForTask">
        <wsdl:input message="tns:SetMaxAttemptsCountForTaskRequest"></wsdl:input>
       <wsdl:output message="tns:SetMaxAttemptsCountForTaskResponse"></wsdl:output>
          <wsdl:fault name="fault" message="tns:SetMaxAttemptsCountForTaskFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetActiveJobTaskIdForTask">
        <wsdl:input message="tns:GetActiveJobTaskIdForTaskRequest"></wsdl:input>
        <wsdl:output message="tns:GetActiveJobTaskIdForTaskResponse"></wsdl:output>
           <wsdl:fault name="fault" message="tns:GetActiveJobTaskIdForTaskFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetCampaignDetails">
        <wsdl:input message="tns:GetCampaignDetailsRequest"></wsdl:input>
        <wsdl:output message="tns:GetCampaignDetailsResponse"></wsdl:output>
```

```
            <wsdl:fault name="fault" message="tns:GetCampaignDetailsFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="ScheduleCampaign">
        <wsdl:input message="tns:ScheduleCampaignRequest"></wsdl:input>
        <wsdl:output message="tns:ScheduleCampaignResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:ScheduleCampaignFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="ScheduleRecurringCampaign">
        <wsdl:input message="tns:ScheduleRecurringCampaignRequest"></wsdl:input>
        <wsdl:output message="tns:ScheduleRecurringCampaignResponse"></wsdl:output>
           <wsdl:fault name="fault" message="tns:ScheduleRecurringCampaignFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="ScheduleDataSource">
        <wsdl:input message="tns:ScheduleDataSourceRequest"></wsdl:input>
        <wsdl:output message="tns:ScheduleDataSourceResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:ScheduleDataSourceFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="ScheduleRecurringDataSource">
        <wsdl:input message="tns:ScheduleRecurringDataSourceRequest"></wsdl:input>
        <wsdl:output message="tns:ScheduleRecurringDataSourceResponse"></
wsdl:output>
            <wsdl:fault name="fault"
message="tns:ScheduleRecurringDataSourceFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetImportJobStatus">
        <wsdl:input message="tns:GetImportJobStatusRequest"></wsdl:input>
        <wsdl:output message="tns:GetImportJobStatusResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:GetImportJobStatusFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetContactListNames">
        <wsdl:input message="tns:GetContactListNamesRequest"></wsdl:input>
        <wsdl:output message="tns:GetContactListNamesResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:GetContactListNamesFault"></
wsdl:fault>
        </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="CmpMgmtSOAP" type="tns:VP_POMCmpMgmtService">

      <soap:binding style="document"
          transport="http://schemas.xmlsoap.org/soap/http" />
      <wsdl:operation name="GetActiveJobs">

          <soap:operation
              soapAction="http://services.pim.avaya.com/CmpMgmt/GetActiveJobs" />
          <wsdl:input>

              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>

              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">

              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="SetMaxAttemptsCount">

          <soap:operation
```

```
                soapAction="http://services.pim.avaya.com/CmpMgmt/
SetMaxAttemptsCount" />
            <wsdl:input>

                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>

                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">

                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="PauseActiveJob">

            <soap:operation
                soapAction="http://services.pim.avaya.com/CmpMgmt/PauseActiveJob" />
            <wsdl:input>

                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>

                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">

                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="ResumePausedJob">

            <soap:operation
                soapAction="http://services.pim.avaya.com/CmpMgmt/ResumePausedJob" />
            <wsdl:input>

                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>

                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">

                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="StopJob">

            <soap:operation
                soapAction="http://services.pim.avaya.com/CmpMgmt/StopJob" />
            <wsdl:input>

                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>

                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">

                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
```

```
      </wsdl:operation>
      <wsdl:operation name="RunCampaign">

          <soap:operation
              soapAction="http://services.pim.avaya.com/CmpMgmt/RunCampaign" />
          <wsdl:input>

              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>

              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">

              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="GetCampaignJobs">

          <soap:operation
              soapAction="http://services.pim.avaya.com/CmpMgmt/GetCampaignJobs" />
          <wsdl:input>

              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>

              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">

              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="GetActiveJobTaskIds">

          <soap:operation
              soapAction="http://services.pim.avaya.com/CmpMgmt/
GetActiveJobTaskIds" />
          <wsdl:input>

              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>

              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">

              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="SetMaxAttemptsCountForTask">

          <soap:operation
              soapAction="http://services.pim.avaya.com/CmpMgmt/
SetMaxAttemptsCountForTask" />
          <wsdl:input>

              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>

              <soap:body use="literal" />
```

```
            </wsdl:output>
            <wsdl:fault name="fault">

                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetActiveJobTaskIdForTask">

            <soap:operation
                soapAction="http://services.pim.avaya.com/CmpMgmt/
GetActiveJobTaskIdForTask" />
            <wsdl:input>

                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>

                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">

                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetCampaignDetails">
            <soap:operation
               soapAction="http://services.pim.avaya.com/CmpMgmt/GetCampaignDetails" /
>
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="ScheduleCampaign">
            <soap:operation
               soapAction="http://services.pim.avaya.com/CmpMgmt/ScheduleCampaign" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="ScheduleRecurringCampaign">
            <soap:operation
                soapAction="http://services.pim.avaya.com/CmpMgmt/
ScheduleRecurringCampaign" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
```

```
      <wsdl:operation name="ScheduleDataSource">
          <soap:operation
             soapAction="http://services.pim.avaya.com/CmpMgmt/ScheduleDataSource" /
>
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="ScheduleRecurringDataSource">
          <soap:operation
              soapAction="http://services.pim.avaya.com/CmpMgmt/
ScheduleRecurringDataSource" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="GetImportJobStatus">
          <soap:operation
             soapAction="http://services.pim.avaya.com/CmpMgmt/GetImportJobStatus" /
>
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="GetContactListNames">
          <soap:operation
              soapAction="http://services.pim.avaya.com/CmpMgmt/
GetContactListNames" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
  </wsdl:binding>
  <wsdl:service name="VP_POMCmpMgmtService">
    <wsdl:port name="CmpMgmtSOAP" binding="tns:CmpMgmtSOAP">
      <soap:address location="http://www.example.org/"/>
    </wsdl:port>
  </wsdl:service>
</wsdl:definitions>
```

Agent API WSDL file

The following is a WSDL file for VP_POMAgentAPIService. This Web service is installed on all the POM servers. For details on accessing the WSDL file, refer to Configuring the VP_POMAgentAPIService Web service on page 34.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions name="AgentAPI" targetNamespace="http://services.pim.avaya.com/
AgentAPI/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://
services.pim.avaya.com/AgentAPI/" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:types>
    <xsd:schema attributeFormDefault="qualified" elementFormDefault="qualified"
        targetNamespace="http://services.pim.avaya.com/AgentAPI/">
      <xsd:element name="GetContactData">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ContactID" type="xsd:string"></xsd:element>
                  <xsd:element name="ContactGroupName" type="xsd:string"></
xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactDataResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Contact" type="tns:ContactType"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:complexType name="ContactType">
          <xsd:sequence>
              <xsd:element name="ContactId" type="xsd:string"></xsd:element>
            <xsd:element name="ContactGroupName" type="xsd:string"></xsd:element>
            <xsd:element name="PhoneNumber1" type="xsd:string" minOccurs="0"></
xsd:element>
              <xsd:element name="PhoneNumber2" type="xsd:string" minOccurs="0"></
xsd:element>
              <xsd:element name="FirstName" type="xsd:string" minOccurs="0"></
xsd:element>
              <xsd:element name="LastName" type="xsd:string" minOccurs="0"></
xsd:element>
           <xsd:element name="Email" type="xsd:string" minOccurs="0"></xsd:element>
            <xsd:element name="LastModifiedOn" type="xsd:dateTime" minOccurs="0"></
xsd:element>
              <xsd:element name="LastModifiedBy" type="xsd:string" minOccurs="0"></
xsd:element>
              <xsd:element name="Language" type="xsd:string" minOccurs="0"></
xsd:element>
              <xsd:element name="TimeZone" type="xsd:string" minOccurs="0"></
xsd:element>
           <xsd:element name="LastAttemptTime" type="xsd:dateTime" minOccurs="0"></
xsd:element>
              <xsd:element name="LastSuccessfulAttemptTime" type="xsd:dateTime"
minOccurs="0"></xsd:element>
           <xsd:element name="LastCompletionCodeId" type="xsd:int" minOccurs="0"></
xsd:element>
           <xsd:element name="AttributeObj" type="tns:AttributeType" minOccurs="0"
```

```
maxOccurs="unbounded"></xsd:element>
          </xsd:sequence>
      </xsd:complexType>
      <xsd:element name="GetContactAttributeValue">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ContactID" type="xsd:string"></xsd:element>
               <xsd:element name="ContactGroupName" type="xsd:string"></xsd:element>
                 <xsd:element name="AttributeName" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactAttributeValueResponse">
          <xsd:complexType>
              <xsd:sequence>
                 <xsd:element name="AttributeValue" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="SaveContact">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Contact" type="tns:ContactType"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="SaveContactResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="AddContactToJob">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
                  <xsd:element name="ContactID" type="xsd:string"></xsd:element>
                  <xsd:element name="ContactGroupName" type="xsd:string"></
xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="AddContactToJobResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="IsDNC">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Address" type="xsd:string"></xsd:element>
                  <xsd:element name="OrgName" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
                  <xsd:element name="CheckForRejectPattern" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
                  <xsd:element name="CheckForPhoneFormatsRule" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="IsDNCResponse">
          <xsd:complexType>
```

```
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:boolean"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddToDNCList">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Address" type="xsd:string"></xsd:element>
                <xsd:element name="OrgName" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
                <xsd:element name="CheckForRejectPattern" type="xsd:boolean"
maxOccurs="1" minOccurs="0" default="false"></xsd:element>
                <xsd:element name="CheckForPhoneFormatsRule" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddToDNCListResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:boolean"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="RemoveFromDNCList">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Address" type="xsd:string"></xsd:element>
                <xsd:element name="OrgName" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
                  <xsd:element name="CheckForRejectPattern" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
                  <xsd:element name="CheckForPhoneFormatsRule" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="RemoveFromDNCListResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:boolean"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:complexType name="AttributeType">
        <xsd:sequence>
            <xsd:element name="Name" type="xsd:string"></xsd:element>
            <xsd:element name="Value" type="xsd:string"></xsd:element>
            <xsd:element name="Type" type="xsd:string"
                minOccurs="0">
            </xsd:element>
            <xsd:element name="DisplayName" type="xsd:string"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="Masked" type="xsd:boolean"
                minOccurs="0" maxOccurs="1">
            </xsd:element>
            <xsd:element name="ReadOnly" type="xsd:boolean" minOccurs="0"
maxOccurs="1"></xsd:element>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="UpdateContactAttributeValue">
        <xsd:complexType>
            <xsd:sequence>
```

```
                <xsd:element name="ContactID" type="xsd:string"></xsd:element>
             <xsd:element name="ContactGroupName" type="xsd:string"></xsd:element>
               <xsd:element name="AttributeName" type="xsd:string"></xsd:element>
                <xsd:element name="AttributeValue" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="UpdateContactAttributeValueResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:int"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetContactDataFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="RetCode" type="xsd:int"></xsd:element>
              <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
          </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetContactAttributeValueFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="RetCode" type="xsd:int"></xsd:element>
              <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
          </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactToJobFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="RetCode" type="xsd:int"></xsd:element>
              <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
          </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="IsDNCFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="RetCode" type="xsd:int"></xsd:element>
              <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
          </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="UpdateContactAttributeValueFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
               <xsd:element name="RetCode" type="xsd:int"></xsd:element>
               <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetAllCompletionCodesForCampaign">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="JobID" type="xsd:int"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetAllCompletionCodesForCampaignResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="CompletionCodes" type="xsd:string"
```

```
maxOccurs="unbounded" minOccurs="0"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetAllCompletionCodesForCampaignFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SaveContactFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              <xsd:element name="RetCode" type="xsd:int"></xsd:element>
          </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddToDNCListFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              <xsd:element name="RetCode" type="xsd:int"></xsd:element>
          </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="RemoveFromDNCListFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              <xsd:element name="RetCode" type="xsd:int"></xsd:element>
          </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="UpdateCompletionCode">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="PimSessionID" type="xsd:double"></xsd:element>
               <xsd:element name="CompletionCode" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="UpdateCompletionCodeResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:int"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="UpdateCompletionCodeFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeleteContact">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="ContactID" type="xsd:string"></xsd:element>
                <xsd:element name="ContactGroupName" type="xsd:string"></
xsd:element>
```

```
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeleteContactResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:boolean"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeleteContactFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactGroupToJob">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
                <xsd:element name="ContactGroupName" type="xsd:string"></
xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactGroupToJobResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:int"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactGroupToJobFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:complexType name="ContactDataType">
        <xsd:sequence>
            <xsd:element name="UserContactId" type="xsd:string"></xsd:element>
            <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
          <xsd:element name="Title" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
           <xsd:element name="FirstName" type="xsd:string" minOccurs="0">
</xsd:element>
           <xsd:element name="LastName" type="xsd:string" minOccurs="0">
</xsd:element>
           <xsd:element name="PhoneNumber1" type="xsd:string" minOccurs="0">
</xsd:element>
           <xsd:element name="TimeZone" type="xsd:string" minOccurs="0">
</xsd:element>
           <xsd:element name="PhoneNumber1CountryCode" type="xsd:string"
minOccurs="0" maxOccurs="1"></xsd:element>
           <xsd:element name="PhoneNumber2" type="xsd:string" minOccurs="0">
</xsd:element>
          <xsd:element name="PhoneNumber2TimeZone" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
           <xsd:element name="PhoneNumber2CountryCode" type="xsd:string"
minOccurs="0" maxOccurs="1"></xsd:element>
           <xsd:element name="Email" type="xsd:string" minOccurs="0"></xsd:element>
```

```
            <xsd:element name="Language" type="xsd:string" minOccurs="0"></
xsd:element>
            <xsd:element name="AddressLine1" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
              <xsd:element name="AddressLine2" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
              <xsd:element name="AddressLine3" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
              <xsd:element name="AddressLine4" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
              <xsd:element name="AddressLine5" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
              <xsd:element name="Country" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
              <xsd:element name="ZipCode" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
            <xsd:element name="AttributeObj" type="tns:AttributeType" minOccurs="0"
maxOccurs="unbounded"></xsd:element>
             <xsd:element name="LastCompletionCodeId" type="xsd:int" minOccurs="0"></
xsd:element>
            <xsd:element name="LastAttemptTime" type="xsd:dateTime" minOccurs="0"></
xsd:element>
             <xsd:element name="LastSuccessfulAttemptTime" type="xsd:dateTime"
minOccurs="0"></xsd:element>
             <xsd:element name="LastModifiedOn" type="xsd:dateTime" minOccurs="0"></
xsd:element>
             <xsd:element name="LastModifiedBy" type="xsd:string" minOccurs="0"></
xsd:element>
          </xsd:sequence>
      </xsd:complexType>
      <xsd:complexType name="PhoneType">
          <xsd:sequence>
              <xsd:element name="CountryCode" type="xsd:string" minOccurs="0"
maxOccurs="1" nillable="true"></xsd:element>
              <xsd:element name="PhoneNumber" type="xsd:string" minOccurs="0"
maxOccurs="1" nillable="true"></xsd:element>
              <xsd:element name="TimeZone" type="xsd:string" minOccurs="0"
maxOccurs="1" nillable="true"></xsd:element>
              <xsd:element name="PhoneAttributeName" type="xsd:string" minOccurs="0"
maxOccurs="1" nillable="true"></xsd:element>
          </xsd:sequence>
      </xsd:complexType>
      <xsd:element name="UpdatePhoneNumber">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ContactID" type="xsd:string"></xsd:element>
                <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
                 <xsd:element name="PhoneObject" type="tns:PhoneType"></xsd:element>
                  <xsd:element name="AutomaticUpdateForTimeZone" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
                <xsd:element name="CheckForRejectPattern" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
                  <xsd:element name="CheckForPhoneFormatsRule" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
                  <xsd:element name="CheckDNC" type="xsd:boolean" minOccurs="0"
maxOccurs="1" default="false"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="UpdatePhoneNumberResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
```

```
      </xsd:element>
      <xsd:element name="UpdatePhoneNumberFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetPhoneNumber">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ContactID" type="xsd:string"></xsd:element>
                <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
                  <xsd:element name="PhoneAttributeName" type="xsd:string"></
xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetPhoneNumberResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Phone" type="tns:PhoneType"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetPhoneNumberFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="ScheduleCallBack">
          <xsd:complexType>
              <xsd:sequence>
                 <xsd:element name="UserContactID" type="xsd:string"></xsd:element>
                  <xsd:element name="ContactListName"    type="xsd:string"></
xsd:element>
                  <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
                  <xsd:element name="StartTime" type="xsd:string"></xsd:element>
                  <xsd:element name="EndTime" type="xsd:string"></xsd:element>
                  <xsd:element name="TimeZone" type="xsd:string" minOccurs="0"
maxOccurs="1">    </xsd:element>
                  <xsd:element name="ContactAttributeName" type="xsd:string"
minOccurs="0" maxOccurs="1"></xsd:element>
                  <xsd:element name="Address" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
                  <xsd:element name="Notes" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="ScheduleCallBackResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="ScheduleCallBackFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
```

```
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactFromListToJob">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
                <xsd:element name="ContactID" type="xsd:string"></xsd:element>
              <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
                <xsd:element name="ContactPriority" type="tns:Priority"
default="MEDIUM" minOccurs="0" maxOccurs="1"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactFromListToJobResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:int"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactFromListToJobFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                 <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                 <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SaveContactToList">
        <xsd:complexType>
            <xsd:sequence>
               <xsd:element name="ContactToBeSaved" type="tns:ContactDataType"></
xsd:element>
                <xsd:element name="AutomaticUpdateForTimeZone" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
                <xsd:element name="CheckForRejectPattern" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
                <xsd:element name="CheckForPhoneFormatsRule" type="xsd:boolean"
minOccurs="0" maxOccurs="1" default="false"></xsd:element>
              <xsd:element name="UpdateExisting" type="xsd:boolean" minOccurs="0"
maxOccurs="1" default="false"></xsd:element>
                <xsd:element name="CheckDNC" type="xsd:boolean" minOccurs="0"
maxOccurs="1" default="false"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SaveContactToListResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:int"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SaveContactToListFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetContactDataFromList">
        <xsd:complexType>
            <xsd:sequence>
```

```
                  <xsd:element name="UserContactID" type="xsd:string"></xsd:element>
                 <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactDataFromListResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ContactRecord" type="tns:ContactDataType"></
xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactDataFromListFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
                <xsd:element name="RetCode" type="xsd:int"></xsd:element>
            </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactAttributeValueFromList">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ContactID" type="xsd:string"></xsd:element>
                <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
                  <xsd:element name="AttributeName" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactAttributeValueFromListResponse">
          <xsd:complexType>
              <xsd:sequence>
                 <xsd:element name="AttributeValue" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactAttributeValueFromListFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                 <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                 <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="UpdateContactAttributeValueToList">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ContactID" type="xsd:string"></xsd:element>
                <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
                  <xsd:element name="AttributeName" type="xsd:string"></xsd:element>
                 <xsd:element name="AttributeValue" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="UpdateContactAttributeValueToListResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="UpdateContactAttributeValueToListFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
```

```
                <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactListToJob">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="CampaignName" type="xsd:string"></xsd:element>
              <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
                <xsd:element name="ContactPriority" type="tns:Priority"
default="MEDIUM" minOccurs="0" maxOccurs="1"></xsd:element>
               <xsd:element name="ApplyFilter" type="xsd:boolean" default="false"
minOccurs="0" maxOccurs="1"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactListToJobResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:int"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="AddContactListToJobFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
               <xsd:element name="RetCode" type="xsd:int"></xsd:element>
               <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeleteContactFromList">
        <xsd:complexType>
            <xsd:sequence>
                 <xsd:element name="ContactID" type="xsd:string"></xsd:element>
                 <xsd:element name="ContactListName" type="xsd:string"></
xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeleteContactFromListResponse">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="Result" type="xsd:boolean"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeleteContactFromListFaultInfo">
        <xsd:complexType>
            <xsd:sequence>
               <xsd:element name="RetCode" type="xsd:int"></xsd:element>
               <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetAttributesList">
        <xsd:complexType>
            <xsd:sequence>
              <xsd:element name="ContactListName" type="xsd:string" minOccurs="0"
maxOccurs="1"></xsd:element>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="GetAttributesListResponse">
```

```
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="attributes" type="tns:AttributeType"
minOccurs="0" maxOccurs="unbounded"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetAttributesListFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                 <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                 <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
            </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="EmptyContactList">
          <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="EmptyContactListResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="TotalContacts" type="xsd:long"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="EmptyContactListFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactListEmptyStatus">
          <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="ContactListName" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactListEmptyStatusResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="ContactListEmptyStatus"
type="tns:ContactListStatus"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetContactListEmptyStatusFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:complexType name="ContactListStatus">
          <xsd:sequence>
              <xsd:element name="EmptyStatus" type="tns:ListStatus"></xsd:element>
              <xsd:element name="TotalCount" type="xsd:long"></xsd:element>
          </xsd:sequence>
```

```
      </xsd:complexType>
      <xsd:simpleType name="Priority">
          <xsd:restriction base="xsd:string">
              <xsd:enumeration value="LOW"></xsd:enumeration>
              <xsd:enumeration value="MEDIUM"></xsd:enumeration>
              <xsd:enumeration value="HIGH"></xsd:enumeration>
          </xsd:restriction>
      </xsd:simpleType>
      <xsd:simpleType name="ListStatus">
          <xsd:restriction base="xsd:string">
              <xsd:enumeration value="LIST_BEING_IDLE"></xsd:enumeration>
              <xsd:enumeration value="LIST_BEING_EMPTIED"></xsd:enumeration>
          </xsd:restriction>
      </xsd:simpleType>
      <xsd:element name="UpdateCampaignAttributeValue">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="JobId" type="xsd:int"></xsd:element>
                 <xsd:element name="AttributeName" type="xsd:string"></xsd:element>
                  <xsd:element name="AttributeValue"
                      type="xsd:double">
                  </xsd:element>
                  <xsd:element name="Operation" type="tns:OperationType"
default="ADD" minOccurs="0"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="UpdateCampaignAttributeValueResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="UpdateCampaignAttributeValueFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string">
                  </xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="UpdateAgentAttributeValue">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="JobId" type="xsd:int"></xsd:element>
                  <xsd:element name="AgentId" type="xsd:string"></xsd:element>
                 <xsd:element name="AttributeName" type="xsd:string"></xsd:element>
                  <xsd:element name="AttributeValue"
                      type="xsd:double">
                  </xsd:element>
                  <xsd:element name="Operation" type="tns:OperationType"
                      default="ADD" minOccurs="0">
                  </xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="UpdateAgentAttributeValueResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="Result" type="xsd:int"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
```

```
      <xsd:element name="UpdateAgentAttributeValueFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string">
                  </xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:simpleType name="OperationType">
          <xsd:restriction base="xsd:string">
              <xsd:enumeration value="ADD"></xsd:enumeration>
              <xsd:enumeration value="MINUS"></xsd:enumeration>
              <xsd:enumeration value="ASSIGN"></xsd:enumeration>
          </xsd:restriction>
      </xsd:simpleType>
      <xsd:element name="GetCampaignAttributesList">
          <xsd:complexType>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetCampaignAttributesListResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="campaignAttributes" type="tns:AttributeType"
minOccurs="0" maxOccurs="unbounded"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetCampaignAttributesListFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetAgentAttributesList">
          <xsd:complexType>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetAgentAttributesListResponse">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="agentAttributes" type="tns:AttributeType"
minOccurs="0" maxOccurs="unbounded"></xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
      <xsd:element name="GetAgentAttributesListFaultInfo">
          <xsd:complexType>
              <xsd:sequence>
                  <xsd:element name="RetCode" type="xsd:int"></xsd:element>
                  <xsd:element name="FaultMsg" type="xsd:string">
                  </xsd:element>
              </xsd:sequence>
          </xsd:complexType>
      </xsd:element>
    </xsd:schema>
  </wsdl:types>
  <wsdl:message name="GetContactDataRequest">
      <wsdl:part name="parameters" element="tns:GetContactData"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactDataResponse">
     <wsdl:part name="parameters" element="tns:GetContactDataResponse"></wsdl:part>
  </wsdl:message>
```

```
  <wsdl:message name="GetContactAttributeValueRequest">
      <wsdl:part name="parameters" element="tns:GetContactAttributeValue"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactAttributeValueResponse">
     <wsdl:part name="parameters" element="tns:GetContactAttributeValueResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="SaveContactRequest">
      <wsdl:part name="parameters" element="tns:SaveContact"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="SaveContactResponse">
      <wsdl:part name="parameters" element="tns:SaveContactResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactToJobRequest">
      <wsdl:part name="parameters" element="tns:AddContactToJob"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactToJobResponse">
      <wsdl:part name="parameters" element="tns:AddContactToJobResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="IsDNCRequest">
      <wsdl:part name="parameters" element="tns:IsDNC"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="IsDNCResponse">
      <wsdl:part name="parameters" element="tns:IsDNCResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddToDNCListRequest">
      <wsdl:part name="parameters" element="tns:AddToDNCList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddToDNCListResponse">
      <wsdl:part name="parameters" element="tns:AddToDNCListResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="RemoveFromDNCListRequest">
      <wsdl:part name="parameters" element="tns:RemoveFromDNCList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="RemoveFromDNCListResponse">
      <wsdl:part name="parameters" element="tns:RemoveFromDNCListResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateContactAttributeValueRequest">
      <wsdl:part name="parameters" element="tns:UpdateContactAttributeValue"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateContactAttributeValueResponse">
      <wsdl:part name="parameters"
element="tns:UpdateContactAttributeValueResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactDataFault">
      <wsdl:part name="parameters" element="tns:GetContactDataFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactAttributeValueFault">
      <wsdl:part name="parameters"
element="tns:GetContactAttributeValueFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactToJobFault">
      <wsdl:part name="parameters" element="tns:AddContactToJobFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="IsDNCFault">
      <wsdl:part name="parameters" element="tns:IsDNCFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateContactAttributeValueFault">
      <wsdl:part name="parameters"
```

```
element="tns:UpdateContactAttributeValueFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAllCompletionCodesForCampaignRequest">
     <wsdl:part name="parameters" element="tns:GetAllCompletionCodesForCampaign"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAllCompletionCodesForCampaignResponse">
      <wsdl:part name="parameters"
element="tns:GetAllCompletionCodesForCampaignResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAllCompletionCodesForCampaignFault">
      <wsdl:part name="parameters"
element="tns:GetAllCompletionCodesForCampaignFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="SaveContactFault">
      <wsdl:part name="parameters" element="tns:SaveContactFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddToDNCListFault">
      <wsdl:part name="parameters" element="tns:AddToDNCListFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="RemoveFromDNCListFault">
      <wsdl:part name="parameters" element="tns:RemoveFromDNCListFaultInfo">
wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateCompletionCodeRequest">
      <wsdl:part name="parameters" element="tns:UpdateCompletionCode"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateCompletionCodeResponse">
      <wsdl:part name="parameters" element="tns:UpdateCompletionCodeResponse">
wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateCompletionCodeFault">
      <wsdl:part name="parameters" element="tns:UpdateCompletionCodeFaultInfo">
wsdl:part>
  </wsdl:message>
  <wsdl:message name="DeleteContactRequest">
      <wsdl:part name="parameters" element="tns:DeleteContact"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="DeleteContactResponse">
      <wsdl:part name="parameters" element="tns:DeleteContactResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="DeleteContactFault">
     <wsdl:part name="parameters" element="tns:DeleteContactFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactGroupToJobRequest">
      <wsdl:part name="parameters" element="tns:AddContactGroupToJob"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactGroupToJobResponse">
      <wsdl:part name="parameters" element="tns:AddContactGroupToJobResponse">
wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactGroupToJobFault">
      <wsdl:part name="parameters" element="tns:AddContactGroupToJobFaultInfo">
wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdatePhoneNumberRequest">
      <wsdl:part name="parameters" element="tns:UpdatePhoneNumber"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdatePhoneNumberResponse">
      <wsdl:part name="parameters" element="tns:UpdatePhoneNumberResponse">
wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdatePhoneNumberFault">
      <wsdl:part name="parameters" element="tns:UpdatePhoneNumberFaultInfo">
wsdl:part>
```

```
  </wsdl:message>
  <wsdl:message name="GetPhoneNumberRequest">
      <wsdl:part name="parameters" element="tns:GetPhoneNumber"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetPhoneNumberResponse">
     <wsdl:part name="parameters" element="tns:GetPhoneNumberResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetPhoneNumberFault">
      <wsdl:part name="parameters" element="tns:GetPhoneNumberFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleCallBackRequest">
      <wsdl:part name="parameters" element="tns:ScheduleCallBack"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleCallBackResponse">
      <wsdl:part name="parameters" element="tns:ScheduleCallBackResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="ScheduleCallBackFault">
      <wsdl:part name="parameters" element="tns:ScheduleCallBackFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactFromListToJobRequest">
      <wsdl:part name="parameters" element="tns:AddContactFromListToJob"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactFromListToJobResponse">
      <wsdl:part name="parameters" element="tns:AddContactFromListToJobResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactFromListToJobFault">
      <wsdl:part name="parameters"
          element="tns:AddContactFromListToJobFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="SaveContactToListRequest">
      <wsdl:part name="parameters" element="tns:SaveContactToList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="SaveContactToListResponse">
      <wsdl:part name="parameters" element="tns:SaveContactToListResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="SaveContactToListFault">
      <wsdl:part name="parameters"
          element="tns:SaveContactToListFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactDataFromListRequest">
     <wsdl:part name="parameters" element="tns:GetContactDataFromList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactDataFromListResponse">
      <wsdl:part name="parameters" element="tns:GetContactDataFromListResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactDataFromListFault">
      <wsdl:part name="parameters"
          element="tns:GetContactDataFromListFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactAttributeValueFromListRequest">
     <wsdl:part name="parameters" element="tns:GetContactAttributeValueFromList"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactAttributeValueFromListResponse">
      <wsdl:part name="parameters"
```

```
element="tns:GetContactAttributeValueFromListResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactAttributeValueFromListFault">
      <wsdl:part name="parameters"
          element="tns:GetContactAttributeValueFromListFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateContactAttributeValueToListRequest">
      <wsdl:part name="parameters"
element="tns:UpdateContactAttributeValueToList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateContactAttributeValueToListResponse">
      <wsdl:part name="parameters"
element="tns:UpdateContactAttributeValueToListResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateContactAttributeValueToListFault">
      <wsdl:part name="parameters"
          element="tns:UpdateContactAttributeValueToListFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactListToJobRequest">
      <wsdl:part name="parameters" element="tns:AddContactListToJob"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactListToJobResponse">
      <wsdl:part name="parameters" element="tns:AddContactListToJobResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="AddContactListToJobFault">
      <wsdl:part name="parameters"
          element="tns:AddContactListToJobFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="DeleteContactFromListRequest">
      <wsdl:part name="parameters" element="tns:DeleteContactFromList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="DeleteContactFromListResponse">
      <wsdl:part name="parameters" element="tns:DeleteContactFromListResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="DeleteContactFromListFault">
      <wsdl:part name="parameters"
          element="tns:DeleteContactFromListFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAttributesListRequest">
      <wsdl:part name="parameters" element="tns:GetAttributesList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAttributesListResponse">
      <wsdl:part name="parameters" element="tns:GetAttributesListResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAttributesListFault">
      <wsdl:part name="parameters" element="tns:GetAttributesListFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="EmptyContactListRequest">
      <wsdl:part name="parameters" element="tns:EmptyContactList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="EmptyContactListResponse">
      <wsdl:part name="parameters" element="tns:EmptyContactListResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="EmptyContactListFault">
      <wsdl:part name="parameters" element="tns:EmptyContactListFaultInfo"></
wsdl:part>
```

```
  </wsdl:message>
  <wsdl:message name="GetContactListEmptyStatusRequest">
      <wsdl:part name="parameters" element="tns:GetContactListEmptyStatus"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactListEmptyStatusResponse">
      <wsdl:part name="parameters"
element="tns:GetContactListEmptyStatusResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetContactListEmptyStatusFault">
      <wsdl:part name="parameters"
          element="tns:GetContactListEmptyStatusFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateCampaignAttributeValueRequest">
      <wsdl:part name="parameters" element="tns:UpdateCampaignAttributeValue"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateCampaignAttributeValueResponse">
      <wsdl:part name="parameters"
element="tns:UpdateCampaignAttributeValueResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateCampaignAttributeValueFault">
      <wsdl:part name="parameters"
          element="tns:UpdateCampaignAttributeValueFaultInfo">
      </wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateAgentAttributeValueRequest">
      <wsdl:part name="parameters" element="tns:UpdateAgentAttributeValue"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateAgentAttributeValueResponse">
      <wsdl:part name="parameters"
element="tns:UpdateAgentAttributeValueResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="UpdateAgentAttributeValueFault">
      <wsdl:part name="parameters"
element="tns:UpdateAgentAttributeValueFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignAttributesListRequest">
      <wsdl:part name="parameters" element="tns:GetCampaignAttributesList"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignAttributesListResponse">
      <wsdl:part name="parameters"
element="tns:GetCampaignAttributesListResponse"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetCampaignAttributesListFault">
      <wsdl:part name="parameters"
element="tns:GetCampaignAttributesListFaultInfo"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAgentAttributesListRequest">
     <wsdl:part name="parameters" element="tns:GetAgentAttributesList"></wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAgentAttributesListResponse">
      <wsdl:part name="parameters" element="tns:GetAgentAttributesListResponse"></
wsdl:part>
  </wsdl:message>
  <wsdl:message name="GetAgentAttributesListFault">
      <wsdl:part name="parameters" element="tns:GetAgentAttributesListFaultInfo"></
wsdl:part>
  </wsdl:message>
  <wsdl:portType name="VP_POMAgentAPIService">
    <wsdl:operation name="GetContactData">
        <wsdl:input message="tns:GetContactDataRequest"></wsdl:input>
```

```
        <wsdl:output message="tns:GetContactDataResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:GetContactDataFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetContactAttributeValue">
        <wsdl:input message="tns:GetContactAttributeValueRequest"></wsdl:input>
        <wsdl:output message="tns:GetContactAttributeValueResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:GetContactAttributeValueFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="SaveContact">
        <wsdl:input message="tns:SaveContactRequest"></wsdl:input>
        <wsdl:output message="tns:SaveContactResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:SaveContactFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="AddContactToJob">
        <wsdl:input message="tns:AddContactToJobRequest"></wsdl:input>
        <wsdl:output message="tns:AddContactToJobResponse"></wsdl:output>
           <wsdl:fault name="fault" message="tns:AddContactToJobFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="IsDNC">
        <wsdl:input message="tns:IsDNCRequest"></wsdl:input>
        <wsdl:output message="tns:IsDNCResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:IsDNCFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="AddToDNCList">
        <wsdl:input message="tns:AddToDNCListRequest"></wsdl:input>
        <wsdl:output message="tns:AddToDNCListResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:AddToDNCListFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="RemoveFromDNCList">
        <wsdl:input message="tns:RemoveFromDNCListRequest"></wsdl:input>
        <wsdl:output message="tns:RemoveFromDNCListResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:RemoveFromDNCListFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="UpdateContactAttributeValue">
        <wsdl:input message="tns:UpdateContactAttributeValueRequest"></wsdl:input>
        <wsdl:output message="tns:UpdateContactAttributeValueResponse"></
wsdl:output>
            <wsdl:fault name="fault"
message="tns:UpdateContactAttributeValueFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetAllCompletionCodesForCampaign">
        <wsdl:input message="tns:GetAllCompletionCodesForCampaignRequest"></
wsdl:input>
        <wsdl:output message="tns:GetAllCompletionCodesForCampaignResponse"></
wsdl:output>
            <wsdl:fault name="fault"
message="tns:GetAllCompletionCodesForCampaignFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="UpdateCompletionCode">
        <wsdl:input message="tns:UpdateCompletionCodeRequest"></wsdl:input>
        <wsdl:output message="tns:UpdateCompletionCodeResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:UpdateCompletionCodeFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="DeleteContact">
        <wsdl:input message="tns:DeleteContactRequest"></wsdl:input>
        <wsdl:output message="tns:DeleteContactResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:DeleteContactFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="AddContactGroupToJob">
        <wsdl:input message="tns:AddContactGroupToJobRequest"></wsdl:input>
        <wsdl:output message="tns:AddContactGroupToJobResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:AddContactGroupToJobFault"></
```

```
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="UpdatePhoneNumber">
        <wsdl:input message="tns:UpdatePhoneNumberRequest"></wsdl:input>
        <wsdl:output message="tns:UpdatePhoneNumberResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:UpdatePhoneNumberFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetPhoneNumber">
        <wsdl:input message="tns:GetPhoneNumberRequest"></wsdl:input>
        <wsdl:output message="tns:GetPhoneNumberResponse"></wsdl:output>
           <wsdl:fault name="fault" message="tns:GetPhoneNumberFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="ScheduleCallBack">
        <wsdl:input message="tns:ScheduleCallBackRequest"></wsdl:input>
        <wsdl:output message="tns:ScheduleCallBackResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:ScheduleCallBackFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="AddContactFromListToJob">
        <wsdl:input message="tns:AddContactFromListToJobRequest"></wsdl:input>
        <wsdl:output message="tns:AddContactFromListToJobResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:AddContactFromListToJobFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="SaveContactToList">
        <wsdl:input message="tns:SaveContactToListRequest"></wsdl:input>
        <wsdl:output message="tns:SaveContactToListResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:SaveContactToListFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetContactDataFromList">
        <wsdl:input message="tns:GetContactDataFromListRequest"></wsdl:input>
        <wsdl:output message="tns:GetContactDataFromListResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:GetContactDataFromListFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetContactAttributeValueFromList">
        <wsdl:input message="tns:GetContactAttributeValueFromListRequest"></
wsdl:input>
        <wsdl:output message="tns:GetContactAttributeValueFromListResponse"></
wsdl:output>
            <wsdl:fault name="fault"
message="tns:GetContactAttributeValueFromListFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="UpdateContactAttributeValueToList">
        <wsdl:input message="tns:UpdateContactAttributeValueToListRequest"></
wsdl:input>
        <wsdl:output message="tns:UpdateContactAttributeValueToListResponse"></
wsdl:output>
            <wsdl:fault name="fault"
message="tns:UpdateContactAttributeValueToListFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="AddContactListToJob">
        <wsdl:input message="tns:AddContactListToJobRequest"></wsdl:input>
        <wsdl:output message="tns:AddContactListToJobResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:AddContactListToJobFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="DeleteContactFromList">
        <wsdl:input message="tns:DeleteContactFromListRequest"></wsdl:input>
        <wsdl:output message="tns:DeleteContactFromListResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:DeleteContactFromListFault"></
wsdl:fault>
        </wsdl:operation>
```

```
    <wsdl:operation name="GetAttributesList">
        <wsdl:input message="tns:GetAttributesListRequest"></wsdl:input>
        <wsdl:output message="tns:GetAttributesListResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:GetAttributesListFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="EmptyContactList">
        <wsdl:input message="tns:EmptyContactListRequest"></wsdl:input>
        <wsdl:output message="tns:EmptyContactListResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:EmptyContactListFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetContactListEmptyStatus">
        <wsdl:input message="tns:GetContactListEmptyStatusRequest"></wsdl:input>
        <wsdl:output message="tns:GetContactListEmptyStatusResponse"></wsdl:output>
           <wsdl:fault name="fault" message="tns:GetContactListEmptyStatusFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="UpdateCampaignAttributeValue">
        <wsdl:input message="tns:UpdateCampaignAttributeValueRequest"></wsdl:input>
        <wsdl:output message="tns:UpdateCampaignAttributeValueResponse">
wsdl:output>
            <wsdl:fault name="fault"
message="tns:UpdateCampaignAttributeValueFault"></wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="UpdateAgentAttributeValue">
        <wsdl:input message="tns:UpdateAgentAttributeValueRequest"></wsdl:input>
        <wsdl:output message="tns:UpdateAgentAttributeValueResponse"></wsdl:output>
           <wsdl:fault name="fault" message="tns:UpdateAgentAttributeValueFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetCampaignAttributesList">
        <wsdl:input message="tns:GetCampaignAttributesListRequest"></wsdl:input>
        <wsdl:output message="tns:GetCampaignAttributesListResponse"></wsdl:output>
           <wsdl:fault name="fault" message="tns:GetCampaignAttributesListFault"></
wsdl:fault>
        </wsdl:operation>
    <wsdl:operation name="GetAgentAttributesList">
        <wsdl:input message="tns:GetAgentAttributesListRequest"></wsdl:input>
        <wsdl:output message="tns:GetAgentAttributesListResponse"></wsdl:output>
            <wsdl:fault name="fault" message="tns:GetAgentAttributesListFault"></
wsdl:fault>
        </wsdl:operation>
  </wsdl:portType>
  <wsdl:binding name="AgentAPISOAP" type="tns:VP_POMAgentAPIService">


      <soap:binding style="document"
          transport="http://schemas.xmlsoap.org/soap/http" />
      <wsdl:operation name="GetContactData">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/GetContactData" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="GetContactAttributeValue">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/
```

```
GetContactAttributeValue" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="SaveContact">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/SaveContact" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="AddContactToJob">
          <soap:operation
             soapAction="http://services.pim.avaya.com/AgentAPI/AddContactToJob" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="IsDNC">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/IsDNC" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="AddToDNCList">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/AddToDNCList" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="RemoveFromDNCList">
          <soap:operation
```

```
            soapAction="http://services.pim.avaya.com/AgentAPI/RemoveFromDNCList" /
>
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="UpdateContactAttributeValue">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/
UpdateContactAttributeValue" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>

      <wsdl:operation name="GetAllCompletionCodesForCampaign">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/
GetAllCompletionCodesForCampaign" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>

      <wsdl:operation name="UpdateCompletionCode">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/
UpdateCompletionCode" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="DeleteContact">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/DeleteContact" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
```

```
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="AddContactGroupToJob">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/
AddContactGroupToJob" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="UpdatePhoneNumber">
            <soap:operation
               soapAction="http://services.pim.avaya.com/AgentAPI/UpdatePhoneNumber" /
>
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetPhoneNumber">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/GetPhoneNumber" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="ScheduleCallBack">
            <soap:operation
               soapAction="http://services.pim.avaya.com/AgentAPI/ScheduleCallBack" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="AddContactFromListToJob">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/
AddContactFromListToJob" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
```

```
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="SaveContactToList">
          <soap:operation
             soapAction="http://services.pim.avaya.com/AgentAPI/SaveContactToList" /
>
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="GetContactDataFromList">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/
GetContactDataFromList" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="GetContactAttributeValueFromList">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/
GetContactAttributeValueFromList" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="UpdateContactAttributeValueToList">
          <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/
UpdateContactAttributeValueToList" />
          <wsdl:input>
              <soap:body use="literal" />
          </wsdl:input>
          <wsdl:output>
              <soap:body use="literal" />
          </wsdl:output>
          <wsdl:fault name="fault">
              <soap:fault use="literal" name="fault" />
          </wsdl:fault>
      </wsdl:operation>
      <wsdl:operation name="AddContactListToJob">
          <soap:operation
```

```
                soapAction="http://services.pim.avaya.com/AgentAPI/
AddContactListToJob" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="DeleteContactFromList">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/
DeleteContactFromList" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetAttributesList">
            <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/GetAttributesList" /
>
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="EmptyContactList">
            <soap:operation
              soapAction="http://services.pim.avaya.com/AgentAPI/EmptyContactList" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetContactListEmptyStatus">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/
GetContactListEmptyStatus" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
```

```
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="UpdateCampaignAttributeValue">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/
UpdateCampaignAttributeValue" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="UpdateAgentAttributeValue">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/
UpdateAgentAttributeValue" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetCampaignAttributesList">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/
GetCampaignAttributesList" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
        <wsdl:operation name="GetAgentAttributesList">
            <soap:operation
                soapAction="http://services.pim.avaya.com/AgentAPI/
GetAgentAttributesList" />
            <wsdl:input>
                <soap:body use="literal" />
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal" />
            </wsdl:output>
            <wsdl:fault name="fault">
                <soap:fault use="literal" name="fault" />
            </wsdl:fault>
        </wsdl:operation>
    </wsdl:binding>

    <wsdl:service name="VP_POMAgentAPIService">
      <wsdl:port binding="tns:AgentAPISOAP" name="AgentAPISOAP">
        <soap:address location="http://services.pim.avaya.com/"/>
      </wsdl:port>
```

```
  </wsdl:service>
</wsdl:definitions>
```

Index

A

B

C

D

E

G

H

I

Z