



Avaya Callback Assist

Web services API guide

Release 4.1.8.1

July 2015

© 2015 Avaya Inc.
All Rights Reserved.

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](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”).

Avaya grants You a license within the scope of the license types described below, with the exception of Heritage Nortel Software, for which the scope of the license is detailed below. Where the order documentation does not expressly identify a license type, the applicable license will be a Designated System License. The applicable number of licenses and units of capacity for which the license is granted will be one (1), unless a different number of licenses or units of capacity is specified in the documentation or other materials available to You. “Software” means computer programs in object code, provided by Avaya or an Avaya Channel Partner, whether as stand-alone products, pre-installed on hardware products, and any upgrades, updates, patches, bug fixes, or modified versions thereto.

“Designated Processor” means a single stand-alone computing device. “Server” means a Designated Processor that hosts a software application to be accessed by multiple users. “Instance” means a single copy of the Software executing at a particular time: (i) on one physical machine; or (ii) on one deployed software virtual machine (“VM”) or similar deployment.

License type(s)

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 one 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 products, 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.

THIS PRODUCT IS LICENSED UNDER THE AVC PATENT PORTFOLIO LICENSE FOR THE PERSONAL USE OF A CONSUMER OR OTHER USES IN WHICH IT DOES NOT RECEIVE REMUNERATION TO (i) ENCODE VIDEO IN COMPLIANCE WITH THE AVC STANDARD ("AVC VIDEO") AND/OR (ii) DECODE AVC VIDEO THAT WAS ENCODED BY A CONSUMER ENGAGED IN A PERSONAL ACTIVITY AND/OR WAS OBTAINED FROM A VIDEO PROVIDER LICENSED TO PROVIDE AVC VIDEO. NO LICENSE IS GRANTED OR SHALL BE IMPLIED FOR ANY OTHER USE. ADDITIONAL INFORMATION

MAY BE OBTAINED FROM MPEG LA, L.L.C. SEE

[HTTP://WWW.MPEGLA.COM](http://www.mpegla.com).

Note to Service Provider

The product or Hosted Service may use Third Party Components subject to Third Party Terms that do not allow hosting and require a Service Provider to be independently licensed for such purpose. It is your responsibility to obtain such licensing.

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.

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.

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.

Table of Content

About this document.....	5
Audience	5
Related documents	5
Introduction.....	6
Callback Assist Web Services API.....	7
Typical usage scenarios.....	7
Immediate Callback Request	7
Scheduled Callback Request	7
Reschedule an existing Callback Request	8
Upload Recorded Customer Name	8
Other scenarios	9
Authentication	9
WS-Security.....	9
Web Service Definition Language Document (WSDL).....	11
Web Service API WSDL version.....	11
Changes from previous versions of Callback Assist	11
WebCallbackWs 4.1.8	13
API References	47
Methods.....	47
WebCallbackWs418	47
Callback Assist Dynamic Routing Engine Interface.....	72
Authentication	72
Web Service Definition Language Document (WSDL).....	72
WSDL Definition Language	72
Example Implementation	74
API Reference.....	75

About this document

The *Avaya Callback Assist Web services API guide* provides detailed information about Avaya Callback Assist (CBA) Web Services API, the process to execute the Web Services API, common scenarios and the new Avaya Callback Assist Dynamic Routing WSDL interface.

Audience

This document is intended primarily for developers willing to use Callback Assist API for creating a client application and Dynamic Routing engines.

Related documents

The *Avaya Callback Assist Web services API guide* is part of the Callback Assist documentation set. The other documents in the documentation set are as follows:

- *Avaya Callback Assist overview and planning guide*
Provides an overview and functional description of the Callback Assist application. The document also lists the features of the Callback Assist application and the environment configuration requirements to install the application.
- *Installing and configuring Avaya Callback Assist*
Provides environment configuration requirements and procedural information to install and configure Callback Assist.
- *Administering Avaya Callback Assist*
Provides information on administering and using Callback Assist.
- *Avaya Callback Assist reporting guide*
Provides information on the Callback Assist Reporting feature.
- *Avaya Callback Assist migration guide*
Provides information on preparing the user for migrating to the latest version of Callback Assist and procedures to migrate.
- *Avaya Callback Assist Release Notes*
Provides information on the latest version of Callback Assist.

Introduction

Callback Assist Web services API is a standards-compliant method of programmatically creating callback requests, checking requests status, consuming configurations, checking requests data and checking agenda availability.

Starting from release 3.2, Callback Assist Web services API feature introduces a new type of Callback configuration called *Web Enabled Configurations*. The current Callback configurations are called *Voice Enabled Configurations* and there is an option to have both configurations tied up which is called *Composite Configurations*. This new type of configuration exposes a new entity called *WebCallback Request*, the request which is created through the Web service API.

Important:

Callback Assist Web services API works mostly with Web Enabled Configurations [can be Web only or Composite (Web and Voice) configurations]. Only two methods in the latest endpoint accept Voice Only Configurations, namely *cancelCallbackRequests* and *getPendingCallbackRequests*. To achieve maximum flexibility and integration capabilities, use the Web Enabled Configurations, if there is no specific requirement to use Voice Only Configurations.

Web services API exposes a subset of Callback Assist features related to offering call flow functionality, for example creating immediate and schedule callback requests and validating agenda availability. Web services API uses the SOAP XML-based messaging format over the standard HTTP application protocol. SOAP has the advantage of being widely used, with clients available in most programming languages.

The Callback Assist Dynamic Routing Interface is a WSDL document that developers can use to build their own routing engine allowing customers to take routing decisions based on their own rules and override the Callback Assist configuration. This simple interface provides customer information like ANI, DNIS, User to User Information (UUI), UCID, phone number and callback ID (global unique identifier) to help the developers to take the best routing decision.

Callback Assist Web Services API

Typical usage scenarios

As a method of exposing offering functionality, the Web services API facilitates the creation of callback requests, checking requests status, and consuming configurations. The following section provides a typical usage scenario for an application talking to the Callback Assist Web services API.

Immediate Callback Request

Perform the following tasks:

1. Obtain the list of available Web Callback Enabled Configurations (includes Composite Configurations) and choose one or query by id or Agent queue VDN. See the following methods for more information:
 - [getWebCallbackConfigurations](#)
 - [getWebCallbackConfigurationById](#)
 - [getWebCallbackConfigurationsByAgentQueue](#)
2. Query the agenda availability to validate which type of callback request can be made at that specific time. Required parameter is the configuration id. As a result, you can create an immediate callback request or a scheduled callback request. See the following method for more information: [getAvailability](#).
3. If you chose immediate callback request in the previous step, then create the request by providing the following information:
 - configuration id
 - phone number to call the customer
 - customer time zone
 - the list of answers related to configuration questions

As result, the API returns the WebCallback request id.
See the following method for more information: [createImmediateWebCallbackRequest](#).

Scheduled Callback Request

Perform the following tasks:

1. Obtain the list of available Web Callback Enabled Configurations (includes Composite Configurations) and choose either query by id or query by Agent queue VDN. See the following methods for more information:
 - [getWebCallbackConfigurations](#)
 - [getWebCallbackConfigurationById](#)
 - [getWebCallbackConfigurationsByAgentQueue](#)
2. Query the agenda availability to validate which type of callback request can be made at that specific time. Required parameter is the configuration id. As a result, you can create an immediate callback request or a scheduled callback request. See the following method for more information: [getAvailability](#).
3. If you chose scheduled callback request in the previous step, you must query the next available time slots from the agenda. The required information is as follows:
 - the configuration id

- a date representing the customer expected date and time to be called back
- the time zone of the customer

Important:

The time zone reference of the date parameter and time zone parameter itself must match in order to query based on customer time. As a result, the API returns a list of next available time slots.

See the following method for more information: [getNextAvailableTimeSlots](#).

4. Create the request by providing the following information:

- configuration id
- phone number to call the customer
- time slot chosen by customer
- customer time zone
- the list of answers related to configuration questions

As result, the API returns the WebCallback request id.

See the following method for more information: [createScheduledWebCallbackRequest](#).

Reschedule an existing Callback Request

Perform the following tasks:

1. Once the callback request created via either [createImmediateWebCallbackRequest](#) or [createScheduledWebCallbackRequest](#). As result, the API returns the WebCallback request id.
2. Reschedule an existing callback request by providing the following information:
 - request id
 - time slot chosen by customer

As result, the API returns the WebCallback request id.

See the following method for more information: [rescheduleWebCallbackRequest](#).

Upload Recorded Customer Name

Perform the following tasks:

1. Once the callback request created via either [createImmediateWebCallbackRequest](#) or [createScheduledWebCallbackRequest](#). As result, the API returns the WebCallback request id.
2. Upload recorded customer name for an existing callback request by providing the following information:
 - request id
 - binary file

As result, the API returns the URL of the uploaded audio file location.

See the following method for more information: [uploadRecordedCustomerNameForRequest](#).

Other scenarios

Get the expected wait time of a particular queue: The Web services API provides a simple interface to query the expected wait time of a particular Agent queue while the Customer is navigating a Web Site or by a thirty party IVR application.

- Obtain the expected wait time (EWT) of an available Web Callback Enabled Configuration by querying by configuration id. This method is only supported on SIP platform Callback Assist installations.
See the following method for more information: [getExpectedWaitTime](#).

Query the status of a WebCallback request: You can query the current status of a WebCallback request to provide real time feedback to customers about the status of their request.

- Obtain the request status by request id.
See the following method for more information: [getWebCallbackRequestStatus](#).

Query a WebCallback request: The main purpose is to provide a simple interface to query the request detailed information for screen pop-up purposes.

- Obtain a representation of a WebCallbackRequest with the request date time, type (scheduled or immediate), customer answers and current status. Required parameter is the request id.
See the following method for more information: [getWebCallbackRequest](#).

Cancel a callback request: The main purpose is to provide a simple interface to cancel an existent callback request which has an outstanding status (that is, pending Callback Requests to be processed). This method supports canceling both Web callback requests and voice callback requests.

- Cancel the callback request by request id or by phone number.
See the following method for more information: [cancelCallbackRequest](#).

Get enabled sites: You can obtain the list of enabled sites configured on the Web Admin application.
See the following method for more information: [getEnabledSites](#).

Get pending callback requests: Returns the list of callback request with pending status.
See the following method for more information: [getPendingCallbackRequests](#).

Authentication

The system performs the Web Callback services authentication at the application level using WS-Security on SOAP message.

WS-Security

- Using UsernameToken and Password Text

See the following for more information:

- WS-Security Specification:
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
- Java implementation:
<http://ws.apache.org/wss4j/>
- CXF and Spring Configuration:
<https://cwiki.apache.org/CXF20DOC/ws-security.html>

Note:

- The system validates the login credentials against the existing user database.
- There is no specific Role or Permission in Callback Assist for granting access to Web services.
- You must provide user credentials for all requests for the Callback Assist Web services server.

Web Service Definition Language Document (WSDL)

The following section provides sample Web Service Definition Language Document (WSDL) code for accessing the Callback Assist Web Services API, although you are not limited to these codes for accessing the Web service. In addition, you can use any SOAP consumer library available in your language of choice. Typically, writing a SOAP Web service client consists of generating the stubs library and then using the Web service client to invoke Web service methods. When using C#, the stubs are generated for you when you add a Web reference to your project using Visual Studio. In other languages and libraries, the stubs are typically generated by running a special application and feeding the Web Service Definition Language (WSDL) file to it.

Web Service API WSDL version

Starting from Callback Assist release 4.1, there are different versions of the Web services API based on Callback Assist versions. For backward compatibility, the Web services API release 4.1 WSDL document is maintained and there is also a new WSDL definition with new methods and features.

WSDL document location

After you install the CBA Web Services API, the WSDL documents are available at the following location:

- Release 4.1 URL:
`http://<SERVER_IP_ADDRESS>:8081/webcallback/WebCallback41?wsdl`
- Release 4.1.8 URL:
`http://<SERVER_IP_ADDRESS>:8081/webcallback/WebCallback418?wsdl`

Note: Web services API release 3.2 and 4.0 WSDL documents are now removed.

Changes from previous versions of Callback Assist

The changes applied on Callback Assist release 4.1 include the following:

- Due to High Availability feature, now all returned Date values are expressed in UTC time and not server's time zone. This change affects older interfaces too.
- Due to High Availability feature, there is a new method called *getEnabledSites*.
- Site identifier (*siteId*) is now required to create callback requests.
See methods [createScheduledWebCallbackRequest](#) and [createImmediateWebCallbackRequest](#).
- Method *getPendingCallbackRequests* parameter *configId* changed to *configurationId*.
- *WebCallbackRequest* type includes *siteId* information.

The changes applied on Callback Assist release 4.1.8 include the following:

- *WebCallbackRequest* object changed as *WebCallbackRequest 418* and it includes customer name and audio URL information.
See method [getWebCallbackRequest](#).
- *WebCallbackConfig* object changed as *WebCallbackConfig418* and it includes *ewtThresholdMax*, *ewtThresholdMin* and language of the customer.
See methods [getWebCallbackConfigurationsByAgentQueue](#), [getWebCallbackConfigurationById](#) and [getWebCallbackConfigurations](#).

- Two new methods added for supporting customization in Customer First CBA outbound application. Those are:
 1. [rescheduleWebCallbackRequest](#)
 2. [uploadRecordedCustomerNameForRequest](#)

For older API interface versions, see *Avaya Callback Assist Web services API Guide 4.1* available at Avaya Support website.

WebCallbackWs 4.1.8

```

<wsdl:definitions name="WebCallback418"
targetNamespace="http://impl.webcallback.callback.csi.avaya.com/"
xmlns:ns1="http://webcallback.callback.csi.avaya.com/"
xmlns:ns2="http://schemas.xmlsoap.org/soap/http"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://impl.webcallback.callback.csi.avaya.com/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <wsdl:import
location="http://135.122.99.106:8081/webcallback/WebCallback418?w
sdl=WebCallbackWs418.wsdl"
namespace="http://webcallback.callback.csi.avaya.com/">
  <wsdl:binding name="WebCallback418SoapBinding"
type="ns1:WebCallbackWs418">
    <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
    <wsdl:operation name="getWebCallbackRequestStatus">
      <soap:operation soapAction="" style="document"/>
      <wsdl:input name="getWebCallbackRequestStatus">
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output name="getWebCallbackRequestStatusResponse">
        <soap:body use="literal"/>
      </wsdl:output>
      <wsdl:fault name="WebCallbackFault">
        <soap:fault name="WebCallbackFault" use="literal"/>
      </wsdl:fault>
      <wsdl:fault name="InvalidLicenseFault">
        <soap:fault name="InvalidLicenseFault"
use="literal"/>
      </wsdl:fault>
      <wsdl:fault name="InvalidRequestIdFault">
        <soap:fault name="InvalidRequestIdFault"
use="literal"/>
      </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getNextAvailableTimeSlots">
      <soap:operation soapAction="" style="document"/>
      <wsdl:input name="getNextAvailableTimeSlots">
        <soap:body use="literal"/>
      </wsdl:input>
      <wsdl:output name="getNextAvailableTimeSlotsResponse">
        <soap:body use="literal"/>
      </wsdl:output>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>

```

```

        </wsdl:output>
        <wsdl:fault name="InvalidTimeZoneFault">
            <soap:fault name="InvalidTimeZoneFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="createScheduledWebCallbackRequest">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="createScheduledWebCallbackRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output
name="createScheduledWebCallbackRequestResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="CustomerHasPendingRequestFault">
            <soap:fault name="CustomerHasPendingRequestFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidPhoneNumberFault">
            <soap:fault name="InvalidPhoneNumberFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidTimeZoneFault">
            <soap:fault name="InvalidTimeZoneFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidVDNException">
            <soap:fault name="InvalidVDNException"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidScheduleSlotFault">
            <soap:fault name="InvalidScheduleSlotFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="WebCallbackFault">

```

```

        <soap:fault name="WebCallbackFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidAnswersFault">
        <soap:fault name="InvalidAnswersFault"
use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidLicenseFault">
        <soap:fault name="InvalidLicenseFault"
use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidUuIFault">
        <soap:fault name="InvalidUuIFault" use="literal"/>
    </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getAvailability">
    <soap:operation soapAction="" style="document"/>
    <wsdl:input name="getAvailability">
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="getAvailabilityResponse">
        <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="InvalidTimeZoneFault">
        <soap:fault name="InvalidTimeZoneFault"
use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="WebCallbackFault">
        <soap:fault name="WebCallbackFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidLicenseFault">
        <soap:fault name="InvalidLicenseFault"
use="literal"/>
    </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="rescheduleWebCallbackRequest">
    <soap:operation soapAction="" style="document"/>
    <wsdl:input name="rescheduleWebCallbackRequest">
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output
name="rescheduleWebCallbackRequestResponse">
        <soap:body use="literal"/>

```

```

        </wsdl:output>
        <wsdl:fault name="InvalidScheduleSlotFault">
            <soap:fault name="InvalidScheduleSlotFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidRequestIdFault">
            <soap:fault name="InvalidRequestIdFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getEnabledSites">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="getEnabledSites">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="getEnabledSitesResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getWebCallbackConfigurations">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="getWebCallbackConfigurations">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output
name="getWebCallbackConfigurationsResponse">
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```

        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getWebCallbackRequest">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="getWebCallbackRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="getWebCallbackRequestResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidRequestIdFault">
            <soap:fault name="InvalidRequestIdFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation
name="getWebCallbackConfigurationsByAgentQueue">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input
name="getWebCallbackConfigurationsByAgentQueue">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output
name="getWebCallbackConfigurationsByAgentQueueResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
    </wsdl:operation>

```

```

        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="cancelCallbackRequest">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="cancelCallbackRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="cancelCallbackRequestResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="InvalidPhoneNumberFault">
            <soap:fault name="InvalidPhoneNumberFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidRequestIdFault">
            <soap:fault name="InvalidRequestIdFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidCriteriaFault">
            <soap:fault name="InvalidCriteriaFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="createImmediateWebCallbackRequest">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="createImmediateWebCallbackRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output
name="createImmediateWebCallbackRequestResponse">
            <soap:body use="literal"/>
        </wsdl:output>

```

```

        <wsdl:fault name="CustomerHasPendingRequestFault">
            <soap:fault name="CustomerHasPendingRequestFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidPhoneNumberFault">
            <soap:fault name="InvalidPhoneNumberFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidTimeZoneFault">
            <soap:fault name="InvalidTimeZoneFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidVDNException">
            <soap:fault name="InvalidVDNException"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidScheduleSlotFault">
            <soap:fault name="InvalidScheduleSlotFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidAnswersFault">
            <soap:fault name="InvalidAnswersFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidUuIFault">
            <soap:fault name="InvalidUuIFault" use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getExpectedWaitTime">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="getExpectedWaitTime">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output name="getExpectedWaitTimeResponse">
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>

```

```

        </wsdl:output>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation
name="uploadRecordedCustomerNameForRequest">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="uploadRecordedCustomerNameForRequest">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output
name="uploadRecordedCustomerNameForRequestResponse">
            <soap:body use="literal"/>
        </wsdl:output>
        <wsdl:fault name="InvalidDeploymentFault">
            <soap:fault name="InvalidDeploymentFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="WebCallbackFault">
            <soap:fault name="WebCallbackFault" use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidLicenseFault">
            <soap:fault name="InvalidLicenseFault"
use="literal"/>
        </wsdl:fault>
        <wsdl:fault name="InvalidRequestIdFault">
            <soap:fault name="InvalidRequestIdFault"
use="literal"/>
        </wsdl:fault>
    </wsdl:operation>
    <wsdl:operation name="getWebCallbackConfigurationById">
        <soap:operation soapAction="" style="document"/>
        <wsdl:input name="getWebCallbackConfigurationById">
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output
name="getWebCallbackConfigurationByIdResponse">

```

```

        <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="WebCallbackFault">
        <soap:fault name="WebCallbackFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidLicenseFault">
        <soap:fault name="InvalidLicenseFault"
use="literal"/>
    </wsdl:fault>
</wsdl:operation>
<wsdl:operation name="getPendingCallbackRequests">
    <soap:operation soapAction="" style="document"/>
    <wsdl:input name="getPendingCallbackRequests">
        <soap:body use="literal"/>
    </wsdl:input>
    <wsdl:output name="getPendingCallbackRequestsResponse">
        <soap:body use="literal"/>
    </wsdl:output>
    <wsdl:fault name="InvalidCalledNumberFault">
        <soap:fault name="InvalidCalledNumberFault"
use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidLimitFault">
        <soap:fault name="InvalidLimitFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="WebCallbackFault">
        <soap:fault name="WebCallbackFault" use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidLicenseFault">
        <soap:fault name="InvalidLicenseFault"
use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidCallingNumberFault">
        <soap:fault name="InvalidCallingNumberFault"
use="literal"/>
    </wsdl:fault>
    <wsdl:fault name="InvalidOffsetFault">
        <soap:fault name="InvalidOffsetFault" use="literal"/>
    </wsdl:fault>
</wsdl:operation>
</wsdl:binding>

```

```

    <wsdl:service name="WebCallback418">
      <wsdl:port binding="tns:WebCallback418SoapBinding"
name="WebCallbackWs418ImplPort">
        <soap:address
location="http://135.122.99.106:8081/webcallback/WebCallback418"/
>
      </wsdl:port>
    </wsdl:service>
  </wsdl:definitions>

```

```

<wsdl:definitions name="WebCallbackWs418"
targetNamespace="http://webcallback.callback.csi.avaya.com/"
xmlns:ns1="http://webcallback.callback.csi.avaya.com/"
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://webcallback.callback.csi.avaya.com/"
xmlns:ns0="http://model.webcallback.callback.csi.avaya.com"
xmlns:ns1="http://exceptions.webcallback.callback.csi.avaya.com"
xmlns:tns="http://webcallback.callback.csi.avaya.com/">
      <xsd:import
namespace="http://model.webcallback.callback.csi.avaya.com"/>
      <xsd:complexType name="ArrayOfDateTime">
        <xsd:sequence>
          <xsd:element maxOccurs="unbounded" minOccurs="0"
name="dateTime" type="xsd:dateTime"/>
        </xsd:sequence>
      </xsd:complexType>
      <xsd:element name="getWebCallbackRequestStatus"
type="tns:getWebCallbackRequestStatus"/>
      <xsd:complexType name="getWebCallbackRequestStatus">
        <xsd:sequence>
          <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
        </xsd:sequence>
      </xsd:complexType>
      <xsd:element name="getWebCallbackRequestStatusResponse"
type="tns:getWebCallbackRequestStatusResponse"/>
      <xsd:complexType
name="getWebCallbackRequestStatusResponse">
        <xsd:sequence>
          <xsd:element name="return" type="xsd:int"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:schema>
  </wsdl:types>

```

```

        <xsd:element name="getNextAvailableTimeSlots"
type="tns:getNextAvailableTimeSlots"/>
        <xsd:complexType name="getNextAvailableTimeSlots">
            <xsd:sequence>
                <xsd:element name="configurationId"
type="xsd:long"/>
                <xsd:element minOccurs="0" name="from"
type="xsd:dateTime"/>
                <xsd:element name="timeZone" type="xsd:float"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="getNextAvailableTimeSlotsResponse"
type="tns:getNextAvailableTimeSlotsResponse"/>
        <xsd:complexType
name="getNextAvailableTimeSlotsResponse">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="return"
type="tns:ArrayOfDateTime"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="createScheduledWebCallbackRequest"
type="tns:createScheduledWebCallbackRequest"/>
        <xsd:complexType
name="createScheduledWebCallbackRequest">
            <xsd:sequence>
                <xsd:element name="configurationId"
type="xsd:long"/>
                <xsd:element minOccurs="0" name="phoneNumber"
nillable="true" type="xsd:string"/>
                <xsd:element name="timeZone" type="xsd:float"/>
                <xsd:element minOccurs="0" name="from"
type="xsd:dateTime"/>
                <xsd:element minOccurs="0" name="answers"
type="ns0:ArrayOfTextAnswer"/>
                <xsd:element minOccurs="0" name="vdn"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="uui"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="siteId"
nillable="true" type="xsd:long"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element
name="createScheduledWebCallbackRequestResponse"
type="tns:createScheduledWebCallbackRequestResponse"/>

```

```

        <xsd:complexType
name="createScheduledWebCallbackRequestResponse">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="return"
type="ns0:OperationResult"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="getAvailability"
type="tns:getAvailability"/>
    <xsd:complexType name="getAvailability">
        <xsd:sequence>
            <xsd:element name="configurationId"
type="xsd:long"/>
            <xsd:element name="timeZone" type="xsd:float"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="getAvailabilityResponse"
type="tns:getAvailabilityResponse"/>
    <xsd:complexType name="getAvailabilityResponse">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="return"
type="ns0:CallbackAvailability"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="rescheduleWebCallbackRequest"
type="tns:rescheduleWebCallbackRequest"/>
    <xsd:complexType name="rescheduleWebCallbackRequest">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
            <xsd:element minOccurs="0" name="from"
type="xsd:dateTime"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="rescheduleWebCallbackRequestResponse"
type="tns:rescheduleWebCallbackRequestResponse"/>
    <xsd:complexType
name="rescheduleWebCallbackRequestResponse">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="return"
type="ns0:OperationResult"/>
        </xsd:sequence>
    </xsd:complexType>

```

```

        <xsd:element name="getEnabledSites"
type="tns:getEnabledSites"/>
        <xsd:complexType name="getEnabledSites">
            <xsd:sequence/>
        </xsd:complexType>
        <xsd:element name="getEnabledSitesResponse"
type="tns:getEnabledSitesResponse"/>
        <xsd:complexType name="getEnabledSitesResponse">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="return"
type="ns0:ArrayOfSite"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="getWebCallbackConfigurations"
type="tns:getWebCallbackConfigurations"/>
        <xsd:complexType name="getWebCallbackConfigurations">
            <xsd:sequence/>
        </xsd:complexType>
        <xsd:element name="getWebCallbackConfigurationsResponse"
type="tns:getWebCallbackConfigurationsResponse"/>
        <xsd:complexType
name="getWebCallbackConfigurationsResponse">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="return"
type="ns0:ArrayOfWebCallbackConfig418"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="getWebCallbackRequest"
type="tns:getWebCallbackRequest"/>
        <xsd:complexType name="getWebCallbackRequest">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="getWebCallbackRequestResponse"
type="tns:getWebCallbackRequestResponse"/>
        <xsd:complexType name="getWebCallbackRequestResponse">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="return"
type="ns0:WebCallbackRequest418"/>
            </xsd:sequence>
        </xsd:complexType>
    
```

```

        <xsd:element
name="getWebCallbackConfigurationsByAgentQueue"
type="tns:getWebCallbackConfigurationsByAgentQueue"/>
        <xsd:complexType
name="getWebCallbackConfigurationsByAgentQueue">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="agentQueue"
nillable="true" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element
name="getWebCallbackConfigurationsByAgentQueueResponse"
type="tns:getWebCallbackConfigurationsByAgentQueueResponse"/>
        <xsd:complexType
name="getWebCallbackConfigurationsByAgentQueueResponse">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="return"
type="ns0:ArrayOfWebCallbackConfig418"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="createImmediateWebCallbackRequest"
type="tns:createImmediateWebCallbackRequest"/>
        <xsd:complexType
name="createImmediateWebCallbackRequest">
            <xsd:sequence>
                <xsd:element name="configurationId"
type="xsd:long"/>
                <xsd:element minOccurs="0" name="phoneNumber"
nillable="true" type="xsd:string"/>
                <xsd:element name="timeZone" type="xsd:float"/>
                <xsd:element minOccurs="0" name="answers"
type="ns0:ArrayOfTextAnswer"/>
                <xsd:element minOccurs="0" name="vdn"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="uui"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="siteId"
nillable="true" type="xsd:long"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element
name="createImmediateWebCallbackRequestResponse"
type="tns:createImmediateWebCallbackRequestResponse"/>
        <xsd:complexType
name="createImmediateWebCallbackRequestResponse">
            <xsd:sequence>

```

```

        <xsd:element minOccurs="0" name="return"
type="ns0:OperationResult"/>
    </xsd:sequence>
</xsd:complexType>
    <xsd:element name="cancelCallbackRequest"
type="tns:cancelCallbackRequest"/>
    <xsd:complexType name="cancelCallbackRequest">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="cancelCriteria"
type="ns0:CancelRequestCriteria"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="cancelCallbackRequestResponse"
type="tns:cancelCallbackRequestResponse"/>
    <xsd:complexType name="cancelCallbackRequestResponse">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="return"
type="ns0:ArrayOfCancelRequestResult"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="getExpectedWaitTime"
type="tns:getExpectedWaitTime"/>
    <xsd:complexType name="getExpectedWaitTime">
        <xsd:sequence>
            <xsd:element name="configurationId"
type="xsd:long"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="getExpectedWaitTimeResponse"
type="tns:getExpectedWaitTimeResponse"/>
    <xsd:complexType name="getExpectedWaitTimeResponse">
        <xsd:sequence>
            <xsd:element name="return" type="xsd:int"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="uploadRecordedCustomerNameForRequest"
type="tns:uploadRecordedCustomerNameForRequest"/>
    <xsd:complexType
name="uploadRecordedCustomerNameForRequest">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
            <xsd:element minOccurs="0" name="audioFileUpload"
type="ns0:AudioFileUpload"/>

```

```

        </xsd:sequence>
    </xsd:complexType>
    <xsd:element
name="uploadRecordedCustomerNameForRequestResponse"
type="tns:uploadRecordedCustomerNameForRequestResponse"/>
    <xsd:complexType
name="uploadRecordedCustomerNameForRequestResponse">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="return"
type="ns0:AudioFile"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="getWebCallbackConfigurationById"
type="tns:getWebCallbackConfigurationById"/>
    <xsd:complexType name="getWebCallbackConfigurationById">
        <xsd:sequence>
            <xsd:element name="configurationId"
type="xsd:long"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element
name="getWebCallbackConfigurationByIdResponse"
type="tns:getWebCallbackConfigurationByIdResponse"/>
    <xsd:complexType
name="getWebCallbackConfigurationByIdResponse">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="return"
type="ns0:WebCallbackConfig418"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:element name="getPendingCallbackRequests"
type="tns:getPendingCallbackRequests"/>
    <xsd:complexType name="getPendingCallbackRequests">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="deliveryDateFrom"
type="xsd:dateTime"/>
            <xsd:element minOccurs="0" name="deliveryDateTo"
type="xsd:dateTime"/>
            <xsd:element minOccurs="0" name="configurationId"
nillable="true" type="xsd:long"/>
            <xsd:element minOccurs="0" name="channelType"
nillable="true" type="ns0:ChannelType"/>
            <xsd:element minOccurs="0" name="callingNumber"
nillable="true" type="xsd:string"/>
            <xsd:element minOccurs="0" name="calledNumber"
nillable="true" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>

```

```

        <xsd:element minOccurs="0" name="offset"
nillable="true" type="xsd:long"/>
        <xsd:element minOccurs="0" name="limit"
nillable="true" type="xsd:long"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:element name="getPendingCallbackRequestsResponse"
type="tns:getPendingCallbackRequestsResponse"/>
<xsd:complexType
name="getPendingCallbackRequestsResponse">
    <xsd:sequence>
        <xsd:element minOccurs="0" name="return"
type="ns0:PendingCallbackRequestResult"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:element name="InvalidLicenseFault" nillable="true"
type="ns1:InvalidLicenseFault"/>
<xsd:element name="InvalidRequestIdFault"
nillable="true" type="ns1:InvalidRequestIdFault"/>
<xsd:element name="WebCallbackFault" nillable="true"
type="ns1:WebCallbackFault"/>
<xsd:element name="InvalidTimeZoneFault" nillable="true"
type="ns1:InvalidTimeZoneFault"/>
<xsd:element name="InvalidUuIFault" nillable="true"
type="ns1:InvalidUuIFault"/>
<xsd:element name="InvalidPhoneNumberFault"
nillable="true" type="ns1:InvalidPhoneNumberFault"/>
<xsd:element name="CustomerHasPendingRequestFault"
nillable="true" type="ns1:CustomerHasPendingRequestFault"/>
<xsd:element name="InvalidScheduleSlotFault"
nillable="true" type="ns1:InvalidScheduleSlotFault"/>
<xsd:element name="InvalidVDNException" nillable="true"
type="ns1:InvalidVDNException"/>
<xsd:element name="InvalidAnswersFault" nillable="true"
type="ns1:InvalidAnswersFault"/>
<xsd:element name="InvalidCriteriaFault" nillable="true"
type="ns1:InvalidCriteriaFault"/>
<xsd:element name="InvalidDeploymentFault"
nillable="true" type="ns1:InvalidDeploymentFault"/>
<xsd:element name="InvalidOffsetFault" nillable="true"
type="ns1:InvalidOffsetFault"/>
<xsd:element name="InvalidCallingNumberFault"
nillable="true" type="ns1:InvalidCallingNumberFault"/>
<xsd:element name="InvalidLimitFault" nillable="true"
type="ns1:InvalidLimitFault"/>
<xsd:element name="InvalidCalledNumberFault"
nillable="true" type="ns1:InvalidCalledNumberFault"/>

```

```

    </xsd:schema>
    <xsd:schema attributeFormDefault="qualified"
elementFormDefault="qualified"
targetNamespace="http://model.webcallback.callback.csi.avaya.com"
xmlns:tns="http://model.webcallback.callback.csi.avaya.com">
        <xsd:complexType name="ArrayOfTextAnswer">
            <xsd:sequence>
                <xsd:element maxOccurs="unbounded" minOccurs="0"
name="TextAnswer" nillable="true" type="tns:TextAnswer"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="WebCallbackConfig418">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="agentQueue"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="ewtThresholdMax"
type="xsd:int"/>
                <xsd:element minOccurs="0" name="ewtThresholdMin"
type="xsd:int"/>
                <xsd:element minOccurs="0" name="id"
type="xsd:long"/>
                <xsd:element minOccurs="0" name="language"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="name"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="questionList"
nillable="true" type="tns:ArrayOfTextQuestion"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="CancelRequestCriteria">
            <xsd:sequence>
                <xsd:element minOccurs="0"
name="callbackConfigurationId" nillable="true" type="xsd:long"/>
                <xsd:element minOccurs="0" name="phoneNumber"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:simpleType name="RequestStatus">
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="SUCCESS"/>
                <xsd:enumeration value="OUTSTANDING"/>
                <xsd:enumeration value="CANCELED"/>
                <xsd:enumeration value="IN_PROGRESS"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:schema>

```

```

        <xsd:enumeration value="TERMINATED"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="RequestType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="IMMEDIATE"/>
        <xsd:enumeration value="SCHEDULED"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="WebCallbackRequest418">
    <xsd:complexContent>
        <xsd:extension base="tns:WebCallbackRequest41">
            <xsd:sequence>
                <xsd:element minOccurs="0"
name="customerNameMessage" nillable="true" type="tns:AudioFile"/>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Site">
    <xsd:sequence>
        <xsd:element minOccurs="0" name="auxiliarAddress"
nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0" name="defaultSite"
type="xsd:boolean"/>
        <xsd:element minOccurs="0" name="name"
nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0" name="primaryAddress"
nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0" name="siteId"
nillable="true" type="xsd:long"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="WebCallbackRequest41">
    <xsd:complexContent>
        <xsd:extension base="tns:WebCallbackRequest40">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="siteId"
nillable="true" type="xsd:long"/>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>

```

```

        </xsd:complexType>
        <xsd:complexType name="OperationResult">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="nodeIP"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="CancelRequestResult">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0"
name="successfullyCanceled" type="xsd:boolean"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="ArrayOfCancelRequestResult">
            <xsd:sequence>
                <xsd:element maxOccurs="unbounded" minOccurs="0"
name="CancelRequestResult" nillable="true"
type="tns:CancelRequestResult"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="ArrayOfSite">
            <xsd:sequence>
                <xsd:element maxOccurs="unbounded" minOccurs="0"
name="Site" nillable="true" type="tns:Site"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="TextAnswer">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="answerText"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="questionId"
type="xsd:long"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="PendingCallbackRequestResult">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="result"
nillable="true" type="tns:ArrayOfPendingCallbackRequest"/>
                <xsd:element minOccurs="0"
name="totalNumberOfRecords" nillable="true" type="xsd:long"/>
            </xsd:sequence>
        </xsd:complexType>
    
```

```

        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="AudioFileUpload">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="binaryFile"
nillable="true" type="xsd:base64Binary"/>
            <xsd:element minOccurs="0" name="name"
nillable="true" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="ArrayOfPendingCallbackRequest">
        <xsd:sequence>
            <xsd:element maxOccurs="unbounded" minOccurs="0"
name="PendingCallbackRequest" nillable="true"
type="tns:PendingCallbackRequest"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:simpleType name="ChannelType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="CHANNEL_TYPE_VOICE"/>
            <xsd:enumeration value="CHANNEL_TYPE_WSAPI"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="ArrayOfWebCallbackConfig418">
        <xsd:sequence>
            <xsd:element maxOccurs="unbounded" minOccurs="0"
name="WebCallbackConfig418" nillable="true"
type="tns:WebCallbackConfig418"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="TextQuestion">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="id"
type="xsd:long"/>
            <xsd:element minOccurs="0" name="questionText"
nillable="true" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="PendingCallbackRequest">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="attempts"
nillable="true" type="xsd:int"/>

```

```

        <xsd:element minOccurs="0" name="calledNumber"
nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0" name="callingNumber"
nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0" name="channelType"
nillable="true" type="tns:ChannelType"/>
        <xsd:element minOccurs="0" name="configId"
nillable="true" type="xsd:long"/>
        <xsd:element minOccurs="0"
name="configurationName" nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0" name="deliveryDateTime"
type="xsd:dateTime"/>
        <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0" name="requestStatus"
nillable="true" type="tns:RequestStatus"/>
        <xsd:element minOccurs="0" name="requestType"
nillable="true" type="tns:RequestType"/>
        <xsd:element minOccurs="0"
name="scheduledDateTime" type="xsd:dateTime"/>
        <xsd:element minOccurs="0" name="specificVDN"
nillable="true" type="xsd:string"/>
        <xsd:element minOccurs="0"
name="userRequestDateTime" type="xsd:dateTime"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="AudioFile">
    <xsd:sequence>
        <xsd:element minOccurs="0" name="url"
nillable="true" type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="WebCallbackRequest40">
    <xsd:complexContent>
        <xsd:extension base="tns:WebCallbackRequest">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="requestId"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="uui"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0" name="vdn"
nillable="true" type="xsd:string"/>
            </xsd:sequence>
        </xsd:extension>
    </xsd:complexContent>

```

```

        </xsd:complexType>
        <xsd:complexType name="WebCallbackRequest">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="callbackConfigId"
type="xsd:long"/>
                <xsd:element minOccurs="0" name="id"
type="xsd:long"/>
                <xsd:element minOccurs="0" name="immediate"
type="xsd:boolean"/>
                <xsd:element minOccurs="0" name="phoneNumber"
nillable="true" type="xsd:string"/>
                <xsd:element minOccurs="0"
name="scheduledDateTime" type="xsd:dateTime"/>
                <xsd:element minOccurs="0" name="status"
type="xsd:int"/>
                <xsd:element minOccurs="0" name="textAnswer"
nillable="true" type="tns:ArrayOfTextAnswer"/>
                <xsd:element minOccurs="0" name="timezone"
type="xsd:float"/>
                <xsd:element minOccurs="0" name="userRequestTime"
type="xsd:dateTime"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="CallbackAvailability">
            <xsd:sequence>
                <xsd:element minOccurs="0" name="callbackConfigId"
type="xsd:long"/>
                <xsd:element minOccurs="0" name="immediateEnabled"
type="xsd:boolean"/>
                <xsd:element minOccurs="0" name="scheduledEnabled"
type="xsd:boolean"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:complexType name="ArrayOfTextQuestion">
            <xsd:sequence>
                <xsd:element maxOccurs="unbounded" minOccurs="0"
name="TextQuestion" nillable="true" type="tns:TextQuestion"/>
            </xsd:sequence>
        </xsd:complexType>
    </xsd:schema>
    <xsd:schema attributeFormDefault="qualified"
elementFormDefault="qualified"
targetNamespace="http://exceptions.webcallback.callback.csi.avaya
.com"
xmlns:tns="http://exceptions.webcallback.callback.csi.avaya.com">
        <xsd:complexType name="InvalidUuIFault">

```

```

        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidPhoneNumberFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidOffsetFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="WebCallbackFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidVDNException">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidScheduleSlotFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="CustomerHasPendingRequestFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidCalledNumberFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidCriteriaFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidAnswersFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidTimeZoneFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidLimitFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidLicenseFault">
        <xsd:sequence/>
    </xsd:complexType>
    <xsd:complexType name="InvalidDeploymentFault">
        <xsd:sequence/>

```

```

        </xsd:complexType>
        <xsd:complexType name="InvalidCallingNumberFault">
            <xsd:sequence/>
        </xsd:complexType>
        <xsd:complexType name="InvalidRequestIdFault">
            <xsd:sequence/>
        </xsd:complexType>
    </xsd:schema>
</wsdl:types>
<wsdl:message name="getExpectedWaitTime">
    <wsdl:part element="ns1:getExpectedWaitTime"
name="parameters"/>
</wsdl:message>
<wsdl:message name="createScheduledWebCallbackRequest">
    <wsdl:part element="ns1:createScheduledWebCallbackRequest"
name="parameters"/>
</wsdl:message>
<wsdl:message name="InvalidLicenseFault">
    <wsdl:part element="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
</wsdl:message>
<wsdl:message name="cancelCallbackRequest">
    <wsdl:part element="ns1:cancelCallbackRequest"
name="parameters"/>
</wsdl:message>
<wsdl:message name="getWebCallbackConfigurationsResponse">
    <wsdl:part
element="ns1:getWebCallbackConfigurationsResponse"
name="parameters"/>
</wsdl:message>
<wsdl:message name="getWebCallbackRequest">
    <wsdl:part element="ns1:getWebCallbackRequest"
name="parameters"/>
</wsdl:message>
<wsdl:message name="getNextAvailableTimeSlotsResponse">
    <wsdl:part element="ns1:getNextAvailableTimeSlotsResponse"
name="parameters"/>
</wsdl:message>
<wsdl:message name="cancelCallbackRequestResponse">
    <wsdl:part element="ns1:cancelCallbackRequestResponse"
name="parameters"/>
</wsdl:message>

```

```

    <wsdl:message
name="createImmediateWebCallbackRequestResponse">
    <wsdl:part
element="ns1:createImmediateWebCallbackRequestResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="getWebCallbackRequestStatusResponse">
    <wsdl:part
element="ns1:getWebCallbackRequestStatusResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="getAvailabilityResponse">
    <wsdl:part element="ns1:getAvailabilityResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="InvalidAnswersFault">
    <wsdl:part element="ns1:InvalidAnswersFault"
name="InvalidAnswersFault"/>
    </wsdl:message>
    <wsdl:message
name="getWebCallbackConfigurationsByAgentQueueResponse">
    <wsdl:part
element="ns1:getWebCallbackConfigurationsByAgentQueueResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="InvalidDeploymentFault">
    <wsdl:part element="ns1:InvalidDeploymentFault"
name="InvalidDeploymentFault"/>
    </wsdl:message>
    <wsdl:message name="getWebCallbackConfigurationsByAgentQueue">
    <wsdl:part
element="ns1:getWebCallbackConfigurationsByAgentQueue"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="CustomerHasPendingRequestFault">
    <wsdl:part element="ns1:CustomerHasPendingRequestFault"
name="CustomerHasPendingRequestFault"/>
    </wsdl:message>
    <wsdl:message name="getWebCallbackConfigurationById">
    <wsdl:part element="ns1:getWebCallbackConfigurationById"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="getExpectedWaitTimeResponse">
    <wsdl:part element="ns1:getExpectedWaitTimeResponse"
name="parameters"/>

```

```

    </wsdl:message>
    <wsdl:message name="getNextAvailableTimeSlots">
        <wsdl:part element="ns1:getNextAvailableTimeSlots"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="uploadRecordedCustomerNameForRequest">
        <wsdl:part
element="ns1:uploadRecordedCustomerNameForRequest"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="WebCallbackFault">
        <wsdl:part element="ns1:WebCallbackFault"
name="WebCallbackFault"/>
    </wsdl:message>
    <wsdl:message name="InvalidTimeZoneFault">
        <wsdl:part element="ns1:InvalidTimeZoneFault"
name="InvalidTimeZoneFault"/>
    </wsdl:message>
    <wsdl:message name="rescheduleWebCallbackRequestResponse">
        <wsdl:part
element="ns1:rescheduleWebCallbackRequestResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="InvalidCalledNumberFault">
        <wsdl:part element="ns1:InvalidCalledNumberFault"
name="InvalidCalledNumberFault"/>
    </wsdl:message>
    <wsdl:message name="getEnabledSites">
        <wsdl:part element="ns1:getEnabledSites"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="InvalidCriteriaFault">
        <wsdl:part element="ns1:InvalidCriteriaFault"
name="InvalidCriteriaFault"/>
    </wsdl:message>
    <wsdl:message name="InvalidRequestIdFault">
        <wsdl:part element="ns1:InvalidRequestIdFault"
name="InvalidRequestIdFault"/>
    </wsdl:message>
    <wsdl:message name="createImmediateWebCallbackRequest">
        <wsdl:part element="ns1:createImmediateWebCallbackRequest"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="getWebCallbackRequestResponse">

```

```

        <wsdl:part element="ns1:getWebCallbackRequestResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="getPendingCallbackRequests">
        <wsdl:part element="ns1:getPendingCallbackRequests"
name="parameters"/>
    </wsdl:message>
    <wsdl:message
name="uploadRecordedCustomerNameForRequestResponse">
        <wsdl:part
element="ns1:uploadRecordedCustomerNameForRequestResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="InvalidUuIFault">
        <wsdl:part element="ns1:InvalidUuIFault"
name="InvalidUuIFault"/>
    </wsdl:message>
    <wsdl:message name="InvalidScheduleSlotFault">
        <wsdl:part element="ns1:InvalidScheduleSlotFault"
name="InvalidScheduleSlotFault"/>
    </wsdl:message>
    <wsdl:message name="InvalidVDNException">
        <wsdl:part element="ns1:InvalidVDNException"
name="InvalidVDNException"/>
    </wsdl:message>
    <wsdl:message name="InvalidLimitFault">
        <wsdl:part element="ns1:InvalidLimitFault"
name="InvalidLimitFault"/>
    </wsdl:message>
    <wsdl:message name="InvalidPhoneNumberFault">
        <wsdl:part element="ns1:InvalidPhoneNumberFault"
name="InvalidPhoneNumberFault"/>
    </wsdl:message>
    <wsdl:message name="getWebCallbackRequestStatus">
        <wsdl:part element="ns1:getWebCallbackRequestStatus"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="rescheduleWebCallbackRequest">
        <wsdl:part element="ns1:rescheduleWebCallbackRequest"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="InvalidOffsetFault">
        <wsdl:part element="ns1:InvalidOffsetFault"
name="InvalidOffsetFault"/>

```

```

    </wsdl:message>
    <wsdl:message name="getWebCallbackConfigurationByIdResponse">
      <wsdl:part
element="ns1:getWebCallbackConfigurationByIdResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="getEnabledSitesResponse">
      <wsdl:part element="ns1:getEnabledSitesResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="InvalidCallingNumberFault">
      <wsdl:part element="ns1:InvalidCallingNumberFault"
name="InvalidCallingNumberFault"/>
    </wsdl:message>
    <wsdl:message name="getPendingCallbackRequestsResponse">
      <wsdl:part element="ns1:getPendingCallbackRequestsResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message
name="createScheduledWebCallbackRequestResponse">
      <wsdl:part
element="ns1:createScheduledWebCallbackRequestResponse"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="getAvailability">
      <wsdl:part element="ns1:getAvailability"
name="parameters"/>
    </wsdl:message>
    <wsdl:message name="getWebCallbackConfigurations">
      <wsdl:part element="ns1:getWebCallbackConfigurations"
name="parameters"/>
    </wsdl:message>
    <wsdl:portType name="WebCallbackWs418">
      <wsdl:operation name="getWebCallbackRequestStatus">
        <wsdl:input message="ns1:getWebCallbackRequestStatus"
name="getWebCallbackRequestStatus"/>
        <wsdl:output
message="ns1:getWebCallbackRequestStatusResponse"
name="getWebCallbackRequestStatusResponse"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
        <wsdl:fault message="ns1:InvalidRequestIdFault"
name="InvalidRequestIdFault"/>
      </wsdl:operation>
    </wsdl:portType>
  </wsdl:binding>
</wsdl:service>

```

```

        </wsdl:operation>
        <wsdl:operation name="getNextAvailableTimeSlots">
            <wsdl:input message="ns1:getNextAvailableTimeSlots"
name="getNextAvailableTimeSlots"/>
            <wsdl:output
message="ns1:getNextAvailableTimeSlotsResponse"
name="getNextAvailableTimeSlotsResponse"/>
            <wsdl:fault message="ns1:InvalidTimeZoneFault"
name="InvalidTimeZoneFault"/>
            <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
            <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
        </wsdl:operation>
        <wsdl:operation name="createScheduledWebCallbackRequest">
            <wsdl:input
message="ns1:createScheduledWebCallbackRequest"
name="createScheduledWebCallbackRequest"/>
            <wsdl:output
message="ns1:createScheduledWebCallbackRequestResponse"
name="createScheduledWebCallbackRequestResponse"/>
            <wsdl:fault message="ns1:CustomerHasPendingRequestFault"
name="CustomerHasPendingRequestFault"/>
            <wsdl:fault message="ns1:InvalidPhoneNumberFault"
name="InvalidPhoneNumberFault"/>
            <wsdl:fault message="ns1:InvalidTimeZoneFault"
name="InvalidTimeZoneFault"/>
            <wsdl:fault message="ns1:InvalidVDNException"
name="InvalidVDNException"/>
            <wsdl:fault message="ns1:InvalidScheduleSlotFault"
name="InvalidScheduleSlotFault"/>
            <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
            <wsdl:fault message="ns1:InvalidAnswersFault"
name="InvalidAnswersFault"/>
            <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
            <wsdl:fault message="ns1:InvalidUuIFault"
name="InvalidUuIFault"/>
        </wsdl:operation>
        <wsdl:operation name="getAvailability">
            <wsdl:input message="ns1:getAvailability"
name="getAvailability"/>
            <wsdl:output message="ns1:getAvailabilityResponse"
name="getAvailabilityResponse"/>
            <wsdl:fault message="ns1:InvalidTimeZoneFault"
name="InvalidTimeZoneFault"/>

```

```

        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
    </wsdl:operation>
    <wsdl:operation name="rescheduleWebCallbackRequest">
        <wsdl:input message="ns1:rescheduleWebCallbackRequest"
name="rescheduleWebCallbackRequest"/>
        <wsdl:output
message="ns1:rescheduleWebCallbackRequestResponse"
name="rescheduleWebCallbackRequestResponse"/>
        <wsdl:fault message="ns1:InvalidScheduleSlotFault"
name="InvalidScheduleSlotFault"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
        <wsdl:fault message="ns1:InvalidRequestIdFault"
name="InvalidRequestIdFault"/>
    </wsdl:operation>
    <wsdl:operation name="getEnabledSites">
        <wsdl:input message="ns1:getEnabledSites"
name="getEnabledSites"/>
        <wsdl:output message="ns1:getEnabledSitesResponse"
name="getEnabledSitesResponse"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
    </wsdl:operation>
    <wsdl:operation name="getWebCallbackConfigurations">
        <wsdl:input message="ns1:getWebCallbackConfigurations"
name="getWebCallbackConfigurations"/>
        <wsdl:output
message="ns1:getWebCallbackConfigurationsResponse"
name="getWebCallbackConfigurationsResponse"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
    </wsdl:operation>
    <wsdl:operation name="getWebCallbackRequest">
        <wsdl:input message="ns1:getWebCallbackRequest"
name="getWebCallbackRequest"/>
        <wsdl:output message="ns1:getWebCallbackRequestResponse"
name="getWebCallbackRequestResponse"/>

```

```

        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
        <wsdl:fault message="ns1:InvalidRequestIdFault"
name="InvalidRequestIdFault"/>
    </wsdl:operation>
    <wsdl:operation
name="getWebCallbackConfigurationsByAgentQueue">
        <wsdl:input
message="ns1:getWebCallbackConfigurationsByAgentQueue"
name="getWebCallbackConfigurationsByAgentQueue"/>
        <wsdl:output
message="ns1:getWebCallbackConfigurationsByAgentQueueResponse"
name="getWebCallbackConfigurationsByAgentQueueResponse"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
    </wsdl:operation>
    <wsdl:operation name="createImmediateWebCallbackRequest">
        <wsdl:input
message="ns1:createImmediateWebCallbackRequest"
name="createImmediateWebCallbackRequest"/>
        <wsdl:output
message="ns1:createImmediateWebCallbackRequestResponse"
name="createImmediateWebCallbackRequestResponse"/>
        <wsdl:fault message="ns1:CustomerHasPendingRequestFault"
name="CustomerHasPendingRequestFault"/>
        <wsdl:fault message="ns1:InvalidPhoneNumberFault"
name="InvalidPhoneNumberFault"/>
        <wsdl:fault message="ns1:InvalidTimeZoneFault"
name="InvalidTimeZoneFault"/>
        <wsdl:fault message="ns1:InvalidVDNException"
name="InvalidVDNException"/>
        <wsdl:fault message="ns1:InvalidScheduleSlotFault"
name="InvalidScheduleSlotFault"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidAnswersFault"
name="InvalidAnswersFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
        <wsdl:fault message="ns1:InvalidUuIFault"
name="InvalidUuIFault"/>
    </wsdl:operation>
    <wsdl:operation name="cancelCallbackRequest">

```

```

        <wsdl:input message="ns1:cancelCallbackRequest"
name="cancelCallbackRequest"/>
        <wsdl:output message="ns1:cancelCallbackRequestResponse"
name="cancelCallbackRequestResponse"/>
        <wsdl:fault message="ns1:InvalidPhoneNumberFault"
name="InvalidPhoneNumberFault"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
        <wsdl:fault message="ns1:InvalidRequestIdFault"
name="InvalidRequestIdFault"/>
        <wsdl:fault message="ns1:InvalidCriteriaFault"
name="InvalidCriteriaFault"/>
    </wsdl:operation>
    <wsdl:operation name="getExpectedWaitTime">
        <wsdl:input message="ns1:getExpectedWaitTime"
name="getExpectedWaitTime"/>
        <wsdl:output message="ns1:getExpectedWaitTimeResponse"
name="getExpectedWaitTimeResponse"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
    </wsdl:operation>
    <wsdl:operation
name="uploadRecordedCustomerNameForRequest">
        <wsdl:input
message="ns1:uploadRecordedCustomerNameForRequest"
name="uploadRecordedCustomerNameForRequest"/>
        <wsdl:output
message="ns1:uploadRecordedCustomerNameForRequestResponse"
name="uploadRecordedCustomerNameForRequestResponse"/>
        <wsdl:fault message="ns1:InvalidDeploymentFault"
name="InvalidDeploymentFault"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
        <wsdl:fault message="ns1:InvalidRequestIdFault"
name="InvalidRequestIdFault"/>
    </wsdl:operation>
    <wsdl:operation name="getWebCallbackConfigurationById">
        <wsdl:input
message="ns1:getWebCallbackConfigurationById"
name="getWebCallbackConfigurationById"/>

```

```

        <wsdl:output
message="ns1:getWebCallbackConfigurationByIdResponse"
name="getWebCallbackConfigurationByIdResponse"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
    </wsdl:operation>
    <wsdl:operation name="getPendingCallbackRequests">
        <wsdl:input message="ns1:getPendingCallbackRequests"
name="getPendingCallbackRequests"/>
        <wsdl:output
message="ns1:getPendingCallbackRequestsResponse"
name="getPendingCallbackRequestsResponse"/>
        <wsdl:fault message="ns1:InvalidCalledNumberFault"
name="InvalidCalledNumberFault"/>
        <wsdl:fault message="ns1:InvalidLimitFault"
name="InvalidLimitFault"/>
        <wsdl:fault message="ns1:WebCallbackFault"
name="WebCallbackFault"/>
        <wsdl:fault message="ns1:InvalidLicenseFault"
name="InvalidLicenseFault"/>
        <wsdl:fault message="ns1:InvalidCallingNumberFault"
name="InvalidCallingNumberFault"/>
        <wsdl:fault message="ns1:InvalidOffsetFault"
name="InvalidOffsetFault"/>
    </wsdl:operation>
</wsdl:portType>
</wsdl:definitions>

```

API References

Methods

This section defines the methods to expose as Web Callback service API to allow you to extend Callback offering capability outside the Voice User Interface and allow other types of view interfaces.

WebCallbackWs418

getWebCallbackConfigurations	
Description	Returns all Web enabled configurations available on the system.
Preconditions	• System properly licensed. • Provide Credentials.
Returns	a <code>java.util.List</code> of <code>WebCallbackConfig418</code>
Throws	<code>InvalidLicenseFault</code>: if the system is not properly licensed. <code>WebCallbackFault</code>: when the system does not find any <code>WebCallbackConfig</code> or if there is an internal error.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://webcallback.callback.csi.avaya.com/"
  <soapenv:Header/>
  <soapenv:Body>
    <web:getWebCallbackConfigurations/>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  <soap:Body>
    <ns1:getWebCallbackConfigurationsResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/"
      <ns1:return>
        <ns2:WebCallbackConfig418
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">
          <ns2:agentQueue>1</ns2:agentQueue>
          <ns2:ewtThresholdMax>99</ns2:ewtThresholdMax>
          <ns2:ewtThresholdMin>1</ns2:ewtThresholdMin>
        </ns2:WebCallbackConfig418>
      </ns1:return>
    </ns1:getWebCallbackConfigurationsResponse>
  </soap:Body>
</soap:Envelope>
```

```

        <ns2:id>1</ns2:id>
        <ns2:language>en-us</ns2:language>
        <ns2:name>SIP_CF_PQ</ns2:name>
        <ns2:questionList/>
    </ns2:WebCallbackConfig418>
    <ns2:WebCallbackConfig418
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">
        <ns2:agentQueue>1</ns2:agentQueue>
        <ns2:ewtThresholdMax>99</ns2:ewtThresholdMax>
        <ns2:ewtThresholdMin>1</ns2:ewtThresholdMin>
        <ns2:id>2</ns2:id>
        <ns2:language>en-us</ns2:language>
        <ns2:name>TEST_PQ</ns2:name>
        <ns2:questionList/>
    </ns2:WebCallbackConfig418>
</ns1:return>
</ns1:getWebCallbackConfigurationsResponse>
</soap:Body>
</soap:Envelope>

```

getWebCallbackConfigurationById	
Description	Returns a unique Web enabled Callback configuration by id.
Parameters	configId -long
Returns	WebCallbackConfig418
Throws	InvalidLicenseFault: if the system is not properly licensed. WebCallbackFault: when the system does not find any WebCallbackConfig or if there is an internal error.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://webcallback.callback.csi.avaya.com/"
  <soapenv:Header/>
  <soapenv:Body>
    <web:getWebCallbackConfigurationById>
      <web:configurationId>1</web:configurationId>
    </web:getWebCallbackConfigurationById>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  <soap:Body>
    <ns1:getWebCallbackConfigurationByIdResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/"
      <ns1:return>
        <ns2:agentQueue
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">1</ns
2:agentQueue>
        <ns2:ewtThresholdMax
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">99</n
s2:ewtThresholdMax>
        <ns2:ewtThresholdMin
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">1</ns
2:ewtThresholdMin>
        <ns2:id
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">1</ns
2:id>
        <ns2:language
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">en-
us</ns2:language>
```

```

        <ns2:name
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">SIP_C
F_PQ</ns2:name>

        <ns2:questionList
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com"/>

        </ns1:return>

    </ns1:getWebCallbackConfigurationByIdResponse>
</soap:Body>
</soap:Envelope>

```

getWebCallbackConfigurationsByAgentQueue	
Description	Returns all Web enabled Callback configuration by agentQueue.
Parameters	agentQueue -String
Returns	a java.util.List of WebCallbackConfig418
Throws	WebCallbackFault: when the system does not find any WebCallbackConfig or if there is an internal error. InvalidLicenseFault: when the system is not properly licensed.

Example of a SOAP Request:

```

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://webcallback.callback.csi.avaya.com/">
<soapenv:Header />
<soapenv:Body>
<web:getWebCallbackConfigurationsByAgentQueue>
<web:agentQueue>1</web:agentQueue>
</web:getWebCallbackConfigurationsByAgentQueue>
</soapenv:Body>
</soapenv:Envelope>

```

Example of a SOAP Response:

```

<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:getWebCallbackConfigurationsByAgentQueueResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
        <ns2:WebCallbackConfig418
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">

```

```

        <ns2:agentQueue>1</ns2:agentQueue>
        <ns2:ewtThresholdMax>99</ns2:ewtThresholdMax>
        <ns2:ewtThresholdMin>1</ns2:ewtThresholdMin>
        <ns2:id>1</ns2:id>
        <ns2:language>en-us</ns2:language>
        <ns2:name>SIP_CF_PQ</ns2:name>
        <ns2:questionList/>
    </ns2:WebCallbackConfig418>
    <ns2:WebCallbackConfig418
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">
        <ns2:agentQueue>1</ns2:agentQueue>
        <ns2:ewtThresholdMax>99</ns2:ewtThresholdMax>
        <ns2:ewtThresholdMin>1</ns2:ewtThresholdMin>
        <ns2:id>2</ns2:id>
        <ns2:language>en-us</ns2:language>
        <ns2:name>TEST_PQ</ns2:name>
        <ns2:questionList/>
    </ns2:WebCallbackConfig418>
</ns1:return>
</ns1:getWebCallbackConfigurationsByAgentQueueResponse>
</soap:Body>
</soap:Envelope>

```

getAvailability	
Description	Returns the available options to leave a callback request based on the specified configuration id and caller time zone. See CallbackAvailability for availability options. Time zone parameter must follow the following convention: Use a float value to express shift from UTC time. For Example: - if UTC-2 then -2 should be passed as parameter. - if UTC+4 then 4 should be passed as parameter. - if UTC+5:30 5.5 should be passed as parameter.
Parameters	configId - long timezone -float
Returns	CallbackAvailability to leave a request
Throws	WebCallbackFault: if there is an unexpected error. InvalidLicenseFault: when the system is not properly licensed.

	InvalidTimeZoneFault: when the time zone parameter goes beyond the fixed [-11, +14] interval of possible time zones.
--	---

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://webcallback.callback.csi.avaya.com/">
<soapenv:Header />
<soapenv:Body>
<web:getAvailability>
<web:configurationId>1</web:configurationId>
<web:timeZone>-6.0</web:timeZone>
</web:getAvailability>
</soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:getAvailabilityResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
        <ns2:callbackConfigId
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">1</ns
2:callbackConfigId>
        <ns2:immediateEnabled
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">true<
/ns2:immediateEnabled>
        <ns2:scheduledEnabled
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">true<
/ns2:scheduledEnabled>
      </ns1:return>
    </ns1:getAvailabilityResponse>
  </soap:Body>
</soap:Envelope>
```

getNextAvailableTimeSlots

Description	Returns a list of available time slots for the specified Web enabled callback configuration, date to start looking for slots, and caller time zone.
--------------------	---

	Time zone parameter must follow the following convention: Use a float value to express shift from UTC time. For Example: - if UTC-2 then -2 should be passed as parameter. - if UTC+4 then 4 should be passed as parameter. - if UTC+5:30 5.5 should be passed as parameter.
Parameters	configId - long from - Date timezone - float Important: Ensure that both the <i>time zone</i> parameter and the <i>Date's</i> time zone property are same. Otherwise the request will fail. For example, if the <i>From</i> parameter is set to 2010-11-24T18:00:00-03:00, that means the time zone is -3. Hence, the <i>time zone</i> parameter has to be '-3'.
Returns	a java.util.List of Date Important: the current format is YYYY-MM-DDTHH:MM:SS±HH:MM. Note that after the hour, minutes, and seconds, comes the time zone hours and minutes. For example, 2010-11-24T18:00:00-03:00, would be the date for November 24th, 2010, at 6:00 PM, with -3 as the time zone.
Throws	WebCallbackFault: when the system cannot obtain WebCallback Configuration with the given id. InvalidLicenseFault: when the system is not properly licensed. InvalidTimeZoneFault: when the time zone parameter goes beyond the fixed [-11, +14] interval of possible time zones.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Header />
  <soapenv:Body>
    <web:getNextAvailableTimeSlots>
      <web:configurationId>1</web:configurationId>
      <web:from>2010-09-30T12:49:45</web:from>
      <web:timeZone>-6.0</web:timeZone>
    </web:getNextAvailableTimeSlots>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```

<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:getNextAvailableTimeSlotsResponse
      xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
        <ns1:dateTime>2010-10-05T01:00:00-06:00</ns1:dateTime>
        <ns1:dateTime>2010-10-05T01:30:00-06:00</ns1:dateTime>
        <ns1:dateTime>2010-10-05T02:00:00-06:00</ns1:dateTime>
      </ns1:return>
    </ns1:getNextAvailableTimeSlotsResponse>
  </soap:Body>
</soap:Envelope>

```

createImmediateWebCallbackRequest

Description	Creates an immediate callback request based on specified configuration id, phone number, caller time zone and a list of TextAnswers. Time zone parameter must follow the following convention: Use a float value to express shift from UTC time. For Example: - if UTC-2 then -2 should be passed as parameter. - if UTC+4 then 4 should be passed as parameter. - if UTC+5:30 5.5 should be passed as parameter.
Parameters	configId - long phoneNumber - String timezone - float answers - a java.util.List of Text Answers vdn - String uui - String siteId - long
Returns	A long data type, corresponding to the Callback Request ID. A string data type, corresponding to the server ip address serving the request.
Throws	WebCallbackFault: when the system cannot obtain WebCallback Configuration with the given id, or when there is no availability to create an immediate Web Callback request. InvalidPhoneNumberFault: if phoneNumber is invalid. InvalidLicenseFault: when the system is not properly licensed. InvalidAnswersFault: when there are missing or invalid amount of answers to create a callback request for the

	given configuration. InvalidTimeZoneFault: when time zone parameter goes beyond the fixed [-11, +14] interval of possible time zones. CustomerHasPendingRequestFault: when there are pending requests for the given phone number.
--	--

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:mod="http://model.webcallback.callback.csi.avaya.com"
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <web:createImmediateWebCallbackRequest>
      <web:configurationId>3</web:configurationId>
      <web:phoneNumber>627263</web:phoneNumber>
      <web:timeZone>-3</web:timeZone>
      <web:answers>
        <mod:TextAnswer>
          <mod:answerText>PhoneNumber</mod:answerText>
          <mod:questionId>7</mod:questionId>
        </mod:TextAnswer>
        <mod:TextAnswer>
          <mod:answerText>CustomerName</mod:answerText>
          <mod:questionId>8</mod:questionId>
        </mod:TextAnswer>
      </web:answers>
      <web:vdn xsi:nil="true"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
      <web:uui xsi:nil="true"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>
      <web:siteId>2</web:siteId>
    </web:createImmediateWebCallbackRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:createImmediateWebCallbackRequestResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
```

```

        <ns2:nodeIP
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">127.0.0.1</
ns2:nodeIP>

        <ns2:requestId
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">e011798a-
6cdc-42dc-96a3-2650226130a3</ns2:requestId>

    </ns1:return>

</ns1:createImmediateWebCallbackRequestResponse>

</soap:Body>
</soap:Envelope>

```

createScheduledWebCallbackRequest	
Description	Creates a scheduled callback request based on the specified configuration id, phone number, schedule date and time, caller time zone and a list of TextAnswers. Time zone parameter must follow the following convention: Use a float value to express shift from UTC time. For Example: - if UTC-2 then -2 should be passed as parameter. - if UTC+4 then 4 should be passed as parameter. - if UTC+5:30 5.5 should be passed as parameter.
Parameters	configId - long phoneNumber - String timezone - float from - Date (must be a previously returned slot by getNextAvailableTimeSlots). answers -a java.util.List of TextAnswer vdn - String uui - String siteId - long Important: Ensure that both the <i>time zone</i> parameter and the <i>Date's time zone property</i> are the same. Otherwise the request will fail. For example, if the <i>From</i> parameter is set to 2010-11-24T18:00:00-03:00, that means the time zone is -3. Hence, the time zone parameter has to be '-3'.
Returns	A string data type, corresponding to the Callback Request ID. A string data type, corresponding to the server ip address serving the request.
Throws	WebCallbackFault: when the system cannot obtain WebCallback Configuration with the given id, or when there is no availability to create a scheduled Web Callback request. InvalidLicenseFault: when the system is not properly licensed.

	InvalidPhoneNumberFault: if the <code>phoneNumber</code> is invalid. InvalidAnswersFault: when there are missing or invalid amount of answers to create a callback request for the given configuration. InvalidTimeZoneFault: when the time zone parameter goes beyond the fixed [-11, +14] interval of possible time zones. CustomerHasPendingRequestFault: when there are pending requests for the given phone number. InvalidScheduleSlotFault: when the requested date is passed the configured amount of days.
--	---

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Body>
    <web:createScheduledWebCallbackRequest>
      <web:configurationId>5</web:configurationId>
      <web:phoneNumber>12345</web:phoneNumber>
      <web:timeZone>-3</web:timeZone>
      <web:from>2012-07-12T18:20:22-03:00</web:from>
      <web:answers>
        <mod:TextAnswer
xmlns:mod="http://model.webcallback.callback.csi.avaya.com">
          <mod:answerText>PhoneNumber</mod:answerText>
          <mod:questionId>13</mod:questionId>
        </mod:TextAnswer>
        <mod:TextAnswer
xmlns:mod="http://model.webcallback.callback.csi.avaya.com">
          <mod:answerText>CustomerName</mod:answerText>
          <mod:questionId>14</mod:questionId>
        </mod:TextAnswer>
      </web:answers>
      <web:siteId>2</web:siteId>
    </web:createScheduledWebCallbackRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:createScheduledWebCallbackRequestResponse
      xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
        <ns2:nodeIP
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">135.1
22.6.102</ns2:nodeIP>
        <ns2:requestId
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">5e694
287-7776-4ca9-8677-f263bba7d049</ns2:requestId>
      </ns1:return>
    </ns1:createScheduledWebCallbackRequestResponse>
  </soap:Body>
</soap:Envelope>
```

getWebCallbackRequestStatus	
Description	Returns the status of a Web enabled callback request.
Parameters	requestId - String (GUID)
Returns	An Integer value between 1 and 5, according to the following status list: 1 (SUCCESS): Callback Requests which were successfully delivered to the customer. 2 (OUTSTANDING): Pending Callback Requests to be processed. 3 (CANCELED): Callback Requests that were canceled by either the agent, or the customer. 4 (IN PROGRESS): Processed Callback Requests that are being delivered. 5 (TERMINATED): When a Callback Request was launched but, either the destination was invalid, or the max retries was reached.
Throws	WebCallbackFault: when the system cannot find a Callback request with the given ID, or when the request is not a WebCallback Request. InvalidLicenseFault: when the system is not properly licensed.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Header />
  <soapenv:Body>
    <web:getWebCallbackRequestStatus>
      <web:requestId>5e694287-7776-4ca9-8677-
f263bba7d049</web:requestId>
    </web:getWebCallbackRequestStatus>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  <soap:Body>
    <ns1:getWebCallbackRequestStatusResponse
      xmlns:ns1="http://webcallback.callback.csi.avaya.com/"
      <ns1:return>2</ns1:return>
    </ns1:getWebCallbackRequestStatusResponse>
  </soap:Body>
</soap:Envelope>
```

getWebCallbackRequest	
Description	Returns a WebCallbackRequest by the given request id.
Parameters	requestId - String (GUID)
Returns	WebCallbackRequest418
Throws	WebCallbackFault: when the system cannot find a Callback request with the given ID, or when the request is not a WebCallback Request. InvalidLicenseFault: when the system is not properly licensed.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Header />
  <soapenv:Body>
```

```
<web:getWebCallbackRequest>
  <web:requestId>4d08d0de-53a7-4165-b156-
efb511d01c23</web:requestId>
</web:getWebCallbackRequest>
</soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:getWebCallbackRequestResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
        <ns2:callbackConfigId
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">5</ns
2:callbackConfigId>
        <ns2:id
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">2</ns
2:id>
        <ns2:immediate
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">>false
</ns2:immediate>
        <ns2:phoneNumber
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">12345
</ns2:phoneNumber>
        <ns2:scheduledDateTime
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">2012-
07-12T18:00:00-03:00</ns2:scheduledDateTime>
        <ns2:status
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">2</ns
2:status>
        <ns2:textAnswer
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">
          <ns2:TextAnswer>
            <ns2:answerText>PhoneNumber</ns2:answerText>
            <ns2:questionId>13</ns2:questionId>
          </ns2:TextAnswer>
          <ns2:TextAnswer>
            <ns2:answerText>CustomerName</ns2:answerText>
            <ns2:questionId>14</ns2:questionId>
          </ns2:TextAnswer>
        </ns2:textAnswer>
        <ns2:timezone
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">-
3.0</ns2:timezone>
```

```

        <ns2:userRequestTime
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">2012-
07-11T18:20:50.947-03:00</ns2:userRequestTime>

        <ns2:requestId
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">4d08d
0de-53a7-4165-b156-efb511d01c23</ns2:requestId>

        <ns2:uui xsi:nil="true"
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>

        <ns2:vdn xsi:nil="true"
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"/>

        <ns2:siteId
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">2</ns
2:siteId>

        <ns2:customerNameMessage
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">

<ns2:url>http://135.122.99.103:8098/riak/audios/92238065-6852-
4420-b386-b5cb16e10bec.wav</ns2:url>

        </ns2:customerNameMessage>

    </ns1:return>
</ns1:getWebCallbackRequestResponse>
</soap:Body>
</soap:Envelope>

```

cancelWebCallbackRequest	
Description	Cancels the callback requests that match the given criteria if the requests are in a Pending State but not waiting in queue. Note: This method cancels both Web and Voice Callback requests. You can cancel Callback requests by GUID, by Phone Number, or by Phone Number and Callback Configuration ID. Take into account that the Phone number criterion will match both the ANI and the Phone Number which was given when requesting Callback.
Parameters	CancelRequestCriteria: A CancelRequestCriteria is a complex type which contains three values: requestId - String: the Global Unique Id (GUID) of the callback request to cancel. phoneNumber - String: the phone number of the requests to cancel, matching any the ANI of the callback request or the phone number where to reach the customer. If E.164 format is enabled the number should be in E.164 format or else the configId parameter also needs to be passed. callbackConfigurationId - long: if a phone number is provided and this value is not null, the system cancels request

	only for this callback configuration; if the value is null and a phone number is provided, then the system cancels the callback requests for any callback configuration that match the given phone. Important: When the given values do not match the Valid Canceling Criteria, the following message will be returned in a WebCallbackFault: Valid Canceling Criteria are: Only {guid}, or Only {phoneNumber}, or {phoneNumber} and {callbackConfigurationId}
Returns	A list of CancelRequestResult which contains each of the Callback Requests GUIDs and a boolean value indicating whether each of the Callback request could be canceled or not. Note: A request cannot be canceled when it is currently in progress.
Throws	WebCallbackFault: when there is no Pending Callback Request for the given canceling criteria, or when the CancelRequestCriteria is not valid, or when an internal error has occurred. InvalidLicenseFault: when the system is not properly licensed.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
      xmlns:web="http://webcallback.callback.csi.avaya.com/"
xmlns:mod="http://model.webcallback.callback.csi.avaya.com">
  <soapenv:Header />
  <soapenv:Body>
    <web:cancelCallbackRequest>
      <web:cancelCriteria>
        <mod:callbackConfigurationId>1</mod:callbackConfigurationId>
        <mod:phoneNumber>1151273899</mod:phoneNumber>
      </web:cancelCriteria>
    </web:cancelCallbackRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:cancelCallbackRequestResponse
      xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
```

```

<ns2:CancelRequestResult

xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">
  <ns2:requestId>17dc65-da81-4ffe-97c6-
21679b06dbba</ns2:requestId>
  <ns2:successfullyCanceled>true</ns2:successfullyCanceled>
</ns2:CancelRequestResult>
<ns2:CancelRequestResult

xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">
  <ns2:requestId>85395ef4-d827-424e-8fde-
ea965848e481</ns2:requestId>
  <ns2:successfullyCanceled>true</ns2:successfullyCanceled>
</ns2:CancelRequestResult>
</ns1:return>
</ns1:cancelCallbackRequestResponse>
</soap:Body>
</soap:Envelope>

```

getExpectedWaitTime	
Description	Returns the expected wait time (EWT) for the given WebCallback Configuration. The EWT depends of the BSR Call Center Application Id configured on the WebCallback Configuration. This method is available <i>only</i> on SIP platform. CTI and AACC-CS1000 platforms return a WebCallbackException.
Parameters	configId - long
Returns	the ewt in minutes
Throws	WebCallbackFault: when the Expected Wait time cannot be retrieved from BSR for the given configuration. InvalidLicenseFault: when the system is not properly licensed.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Header />
  <soapenv:Body>
    <web:getExpectedWaitTime>
      <web:configurationId>1</web:configurationId>
    </web:getExpectedWaitTime>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:getExpectedWaitTimeResponse
      xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>1093</ns1:return>
    </ns1:getExpectedWaitTimeResponse>
  </soap:Body>
</soap:Envelope>
```

getPendingCallbackRequests	
Description	Returns the list of callback request with pending status. It implements a pagination strategy using the offset and limit input parameters.
Parameters	• deliveryDateFrom - Date: delivery date to filter from (optional parameter). • deliveryDateTo - Date: delivery date to filter to (optional parameter). • configId - Long: if present, the system searches only the callback requests for the callback configuration with this id (optional parameter). • channelType - String: only valid values are: "CHANNEL_TYPE_VOICE" , "CHANNEL_TYPE_WSAPI" or "CHANNEL_TYPE_ANY" (if not present, "CHANNEL_TYPE_ANY" is assumed); filter requests for the given inbound channel (voice, web service or both). • callingNumber - String: ANI of the requests to search for (optional parameter). • calledNumber - String: phone number where to reach the customer at (optional parameter). If E.164 format is enabled the number should be in E.164

	format or else the configId parameter also needs to be passed. • <code>offset</code> – long: to retrieve result rows from a certain offset (mandatory parameter). • <code>limit</code> – long: maximum number of results to return. Maximum allowed value is 50. If you provide a higher value, the system takes 50.
Returns	a complex type that contains: • the total number of results matching the given search criteria. • a list of pending callbacks matching the search criteria, each result with the following: <code>configId</code>: the callback configuration ID. <code>configuration name</code>: the callback configuration name. <code>requestId</code>: the Global Unique Id (GUID) of the callback request. <code>calling number</code>: the ANI. <code>called number</code>: the number to reach the customer. <code>userRequestDateTime</code>: the time when the request was created. <code>deliveryDateTime</code>: the time for the next attempt to reach the customer. <code>scheduledDateTime</code>: only present if it is a scheduled request, it is the original time to call the customer. <code>request status</code>: status of the callback request. <code>requestType</code>: possible values are IMMEDIATE or SCHEDULED. <code>channel type</code>: possible values are VOICE or WEB. <code>attempts</code>: number of already executed attempt to reach the customer. <code>specific VDN</code>: if the request is a Webcallback, it is the specific VDN number where the agent queue is located for this request.
Throws	<code>InvalidLicenseFault</code>: if the system is not properly licensed. <code>WebCallbackFault</code>: when no WebCallbackConfig found or internal error.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Header />
  <soapenv:Body>
```

```

<web:getPendingCallbackRequests>
  <web:configId>1</web:configId>
  <web:channelType>CHANNEL_TYPE_ANY</web:channelType>
  <web:callingNumber xsi:nil="true"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" />
  <web:calledNumber xsi:nil="true"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
/>
  <web:offset>0</web:offset>
  <web:limit>10</web:limit>
</web:getPendingCallbackRequests>
</soapenv:Body>
</soapenv:Envelope>

```

Example of a SOAP Response:

```

<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:getPendingCallbackRequestsResponse
      xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
        <ns2:result
          xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">
          <ns2:PendingCallbackRequest>
            <ns2:attempts>0</ns2:attempts>
            <ns2:calledNumber>11112</ns2:calledNumber>
            <ns2:callingNumber>null</ns2:callingNumber>
            <ns2:channelType>CHANNEL_TYPE_WSAPI</ns2:channelType>
            <ns2:configId>1</ns2:configId>

<ns2:configurationName>WebConfiguration</ns2:configurationName>
            <ns2:requestId>1ea47a68-e15b-4baf-822d-
95e0d92163e5</ns2:requestId>
            <ns2:requestStatus>OUTSTANDING</ns2:requestStatus>
            <ns2:requestType>SCHEDULED</ns2:requestType>
            <ns2:scheduledDateTime>2011-10-29T00:00:00-06:00
</ns2:scheduledDateTime>
            <ns2:specificVDN ns3:nil="true"
              xmlns:ns3="http://www.w3.org/2001/XMLSchema-
instance" />
            <ns2:userRequestDateTime>2011-10-29T00:00:00-06:00
</ns2:userRequestDateTime>

```

```

        </ns2:PendingCallbackRequest>
        <ns2:PendingCallbackRequest>
            <ns2:attempts>0</ns2:attempts>
            <ns2:calledNumber>11111</ns2:calledNumber>
            <ns2:callingNumber>null</ns2:callingNumber>
            <ns2:channelType>CHANNEL_TYPE_WSAPI</ns2:channelType>
            <ns2:configId>1</ns2:configId>

<ns2:configurationName>WebConfiguration</ns2:configurationName>
    <ns2:requestId>534bb205-4549-4244-826b-
de343146d864</ns2:requestId>
    <ns2:requestStatus>OUTSTANDING</ns2:requestStatus>
    <ns2:requestType>IMMEDIATE</ns2:requestType>
    <ns2:specificVDN ns3:nil="true"
                                xmlns:ns3="http://www.w3.org/2001/XMLSchema-
instance" />
    <ns2:userRequestDateTime>2011-10-29T00:00:00-06:00
    </ns2:userRequestDateTime>
</ns2:PendingCallbackRequest>
<ns2:PendingCallbackRequest>
    <ns2:attempts>0</ns2:attempts>
    <ns2:calledNumber>11112</ns2:calledNumber>
    <ns2:callingNumber>null</ns2:callingNumber>
    <ns2:channelType>CHANNEL_TYPE_WSAPI</ns2:channelType>
    <ns2:configId>1</ns2:configId>

<ns2:configurationName>WebConfiguration</ns2:configurationName>
    <ns2:requestId>95a43b0c-d616-42c3-9f48-
017eae038294</ns2:requestId>
    <ns2:requestStatus>OUTSTANDING</ns2:requestStatus>
    <ns2:requestType>SCHEDULED</ns2:requestType>
    <ns2:scheduledDateTime>2011-10-28T23:00:00-06:00
    </ns2:scheduledDateTime>
    <ns2:specificVDN ns3:nil="true"
                                xmlns:ns3="http://www.w3.org/2001/XMLSchema-
instance" />
    <ns2:userRequestDateTime>2011-10-28T23:00:00-06:00
    </ns2:userRequestDateTime>
</ns2:PendingCallbackRequest>
</ns2:result>
<ns2:totalNumberOfRecords

```

```

xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">1440<
/ns2:totalNumberOfRecords>
    </ns1:return>
  </ns1:getPendingCallbackRequestsResponse>
</soap:Body>
</soap:Envelope>

```

getEnabledSites	
Description	Returns the list of enabled sites configured on the Web Admin application.
Parameters	none
Returns	The list of enabled sites if any or an empty list.
Throws	WebCallbackFault : when the system cannot retrieve the list of sites in case of internal errors. InvalidLicenseFault: when the system is not properly licensed.

Example of a SOAP Request:

```

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <web:getEnabledSites/>
  </soapenv:Body>
</soapenv:Envelope>

```

Example of a SOAP Response:

```

<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:getEnabledSitesResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
        <ns2:Site
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">
          <ns2:auxiliarAddress/>
          <ns2:isDefaultSite>true</ns2:isDefaultSite>
          <ns2:name>defaultSiteForTesting</ns2:name>
          <ns2:primaryAddress>127.0.0.1</ns2:primaryAddress>
          <ns2:siteId>2</ns2:siteId>
        </ns2:Site>
      </ns1:return>
    </ns1:getEnabledSitesResponse>
  </soap:Body>
</soap:Envelope>

```

```

        </ns2:Site>
    </ns1:return>
</ns1:getEnabledSitesResponse>
</soap:Body>
</soap:Envelope>

```

rescheduleWebCallbackRequest	
Description	Reschedule an existing callback request based on specified request id and reschedule date and time.
Parameters	requestId - String (GUID) from - Date (must be a previously returned slot by getNextAvailableTimeSlots).
Returns	A string data type, corresponding to the Callback Request ID. A string data type, corresponding to the server ip address serving the request.
Throws	WebCallbackFault: when the system cannot obtain callback request with the given guid, or reschedule option is not enabled, or max_retries limit reached, or no_answer_retries limit reached, or the current status of callback request is not valid for reschedule. InvalidLicenseFault: when the system is not properly licensed. InvalidScheduleSlotFault: when the requested date is passed the configured amount of days. InvalidRequestIdFault: when the request id is not a valid GUID.

Example of a SOAP Request:

```

<soapenv:Envelope
  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:web="http://webcallback.callback.csi.avaya.com/">
  <soapenv:Header/>
  <soapenv:Body>
    <web:rescheduleWebCallbackRequest>
      <web:requestId>0d3bd3e7-6d58-4528-b57e-56da892dd6b2</web:requestId>
      <web:from>2015-04-28T10:50:30.073Z</web:from>
    </web:rescheduleWebCallbackRequest>
  </soapenv:Body>
</soapenv:Envelope>

```

```
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  <soap:Body>
    <ns1:rescheduleWebCallbackRequestResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/"
      <ns1:return>
        <ns2:nodeIP
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">135.1
22.99.103</ns2:nodeIP>
        <ns2:requestId
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">0d3bd
3e7-6d58-4528-b57e-56da892dd6b2</ns2:requestId>
      </ns1:return>
    </ns1:rescheduleWebCallbackRequestResponse>
  </soap:Body>
</soap:Envelope>
```

uploadRecordedCustomerNameForRequest	
Description	Upload recorded customer name as a binary file for an existing callback request.
Parameters	requestId – String (GUID) audioFileUpload – A simple date type contains binaryFile as byte[] and file name as String
Returns	url – URL of the uploaded customer name audio file location.
Throws	WebCallbackFault: when the system cannot obtain callback request with the given guid. InvalidLicenseFault: when the system is not properly licensed. InvalidDeploymentFault: when the deployment type is not customer first. InvalidRequestIdFault: when the request id is not a valid GUID.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:web="http://webcallback.callback.csi.avaya.com/"
xmlns:mod="http://model.webcallback.callback.csi.avaya.com">
```

```
<soapenv:Header/>
<soapenv:Body>
  <web:uploadRecordedCustomerNameForRequest>
    <web:requestId>0d3bd3e7-6d58-4528-b57e-
56da892dd6b2</web:requestId>
    <web:audioFileUpload>
      <mod:binaryFile>cid:CustomerName.wav</mod:binaryFile>
      <mod:name>CustomerName.wav</mod:name>
    </web:audioFileUpload>
  </web:uploadRecordedCustomerNameForRequest>
</soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
--uuid:2c858c9b-99f5-4114-a339-cc71039b4436
Content-Type: application/xop+xml; charset=UTF-8;
type="text/xml";
Content-Transfer-Encoding: binary
Content-ID: <root.message@cxf.apache.org>

<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:uploadRecordedCustomerNameForRequestResponse
xmlns:ns1="http://webcallback.callback.csi.avaya.com/">
      <ns1:return>
        <ns2:url
xmlns:ns2="http://model.webcallback.callback.csi.avaya.com">http://135.122.99.103:8098/riak/audios/92238065-6852-4420-b386-b5cb16e10bec.wav</ns2:url>
      </ns1:return>
    </ns1:uploadRecordedCustomerNameForRequestResponse>
  </soap:Body>
</soap:Envelope>

--uuid:2c858c9b-99f5-4114-a339-cc71039b4436--
```

Callback Assist Dynamic Routing Engine Interface

Authentication

Avaya Callback Assist is a interface client-side application and does not expect any Authentication to be performed to consume the SOAP services.

Web Service Definition Language Document (WSDL)

The following section provides the Dynamic Routing Definition Language Document (WSDL) for implementing the API. In addition, you can use any SOAP library available in your language of choice to implement the service.

The WSDL file is also located at:

- `<installation_directory>/Avaya/callbackassist/support/dynamic-routing-example.tar`

Run the command `tar -xvf dynamic-routing-example.tar` to decompress the file and get the WSDL document called **dynamicRoutingInterface.wsdl**.

WSDL Definition Language

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:schema="http://avaya.com/ept/callbackassist/schemas"
xmlns:tns="http://avaya.com/ept/callbackassist/definitions"
targetNamespace="http://avaya.com/ept/callbackassist/definitions"
>
  <wsdl:types>
    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
targetNamespace="http://avaya.com/ept/callbackassist/schemas"
xmlns:schemas="http://avaya.com/ept/callbackassist/schemas">
      <xs:element name="RoutingDestinationRequest">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="ani" type="xs:string"
minOccurs="1" maxOccurs="1"/>
            <xs:element name="dnis" type="xs:string"
minOccurs="1" maxOccurs="1"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  </wsdl:types>
</wsdl:definitions>
```

```

        <xs:element name="uui" type="xs:string"
minOccurs="1" maxOccurs="1"/>
        <xs:element name="ucid" type="xs:string"
minOccurs="1" maxOccurs="1"/>
        <xs:element name="phone" type="xs:string"
minOccurs="1" maxOccurs="1"/>
        <xs:element name="callbackGuid"
type="xs:string" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
targetNamespace="http://avaya.com/ept/callbackassist/schemas"
xmlns:schemas="http://avaya.com/ept/callbackassist/schemas">
    <xs:element name="RoutingDestinationResponse">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="destination"
type="xs:anyURI" minOccurs="1" maxOccurs="1"/>
                <xs:element name="timeout" type="xs:int"
minOccurs="1" maxOccurs="1"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>
</wsdl:types>

<wsdl:message name="DynamicRoutingRequest">
    <wsdl:part element="schema:RoutingDestinationRequest"
name="RoutingDestinationRequest"/>
</wsdl:message>
<wsdl:message name="DynamicRoutingResponse">
    <wsdl:part element="schema:RoutingDestinationResponse"
name="DynamicRoutingResponse"/>
</wsdl:message>

<wsdl:portType name="DynamicRouting">
    <wsdl:operation name="getDynamicRouting">
        <wsdl:input message="tns:DynamicRoutingRequest"
name="RoutingDestinationRequest"/>
        <wsdl:output message="tns:DynamicRoutingResponse"
name="RoutingDestinationResponse"/>
    </wsdl:operation>
</wsdl:portType>

```

```

        </wsdl:operation>
    </wsdl:portType>

    <wsdl:binding name="DynamicRoutingBinding"
type="tns:DynamicRouting">
        <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
        <wsdl:operation name="getDynamicRouting">
            <soap:operation
soapAction="http://avaya.com/ept/callbackassist/getDynamicRouting
Destination"/>
                <wsdl:input name="RoutingDestinationRequest">
                    <soap:body use="literal"/>
                </wsdl:input>
                <wsdl:output name="RoutingDestinationResponse">
                    <soap:body use="literal"/>
                </wsdl:output>
            </wsdl:operation>
        </wsdl:binding>
        <wsdl:service name="DynamicRoutingService">
            <wsdl:port binding="tns:DynamicRoutingBinding"
name="DynamicRoutingEndpoint">
                <soap:address
location="http://localhost:8080/DynamicRoutingService/">
            </wsdl:port>
        </wsdl:service>
    </wsdl:definitions>

```

Example Implementation

Callback Assist binary file ships an example of a simple implementation of Dynamic Routing engine in order to help developers to create their own implementation. The example is located at:

- **<installation_directory>/Avaya/callbackassist/support/dynamic-routing-example.tar**

The sample includes the following:

- build.xml : Ant build file to create the example.war file.
- README.txt : A brief explanation on how to use the build.xml and Ant tool to build the .war application.
- lib: Contains the required libraries.
- webapp: Contains the required configuration files.
- src: Contains the source code of the sample implementation.

Starting point for the developers to create their own implementation.

API Reference

DynamicRoutingRequest	
Description	Queries the Dynamic Routing Engine for a destination Queue which is valid based on timeout response.
Preconditions	None
Returns	a RoutingDestinationResponse including SIP URI destination and a timeout value.

Example of a SOAP Request:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://avaya.com/ept/callbackassist/schemas">
  <soapenv:Header/>
  <soapenv:Body>
    <sch:RoutingDestinationRequest>
      <sch:ani>7204442134</sch:ani>
      <sch:dnis>8006660000</sch:dnis>

<sch:uui>04C8116573746545735575694465507275656261FA08000202074B89
27C5;encoding=hex</sch:uui>
      <sch:ucid>000202074B8927C5</sch:ucid>
      <sch:phone>7204442134</sch:phone>
      <sch:callbackGuid>75457e4c-e19f-3e97-af6f-
e11cd160e95b</sch:callbackGuid>
    </sch:RoutingDestinationRequest>
  </soapenv:Body>
</soapenv:Envelope>
```

Example of a SOAP Response:

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:sch="http://avaya.com/ept/callbackassist/schemas">
  <soapenv:Header/>
  <soapenv:Body>
    <sch:RoutingDestinationResponse>
<sch:destination>sip:3334587@mydomain.com</sch:destination>
      <sch:timeout>15</sch:timeout>
    </sch:RoutingDestinationResponse>
  </soapenv:Body>
</soapenv:Envelope>
```