Avaya Dynamic Self Service
application developers guide

Release 2.9.2

Issue 1

November 2017

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.

HostedService

THE FOLLOWING APPLIES IF YOU PURCHASE A HOSTED SERVICE SUBSCRIPTION FROM AVAYA OR AN AVAYA CHANNEL PARTNER (AS APPLICABLE), THE TERMS OF USE FOR HOSTED SERVICES ARE AVAILABLE ON THE AVAYA WEBSITE, HTTP://SUPPORT.AVAYA.COM/ LICENSEINFO UNDER THE LINK "Avaya Terms of Use for Hosted Services" OR SUCH SUCCESSOR SITE AS DESIGNATED BY AVAYA, AND ARE APPLICABLE TO ANYONE WHO ACCESSES OR USES THE HOSTED SERVICE. BY ACCESSING OR USING THE HOSTED SERVICE, OR AUTHORIZING OTHERS TO DO SO, YOU, ON BEHALF OF YOURSELF AND THE ENTITY FOR WHOM YOU ARE DOING SO (HEREINAFTER REFERRED TO INTERCHANGEABLY AS "YOU" AND "END USER"), AGREE TO THE TERMS OF USE. IF YOU ARE ACCEPTING THE TERMS OF USE ON BEHALF A COMPANY OR OTHER LEGAL ENTITY, YOU REPRESENT THAT YOU HAVE THE AUTHORITY TO BIND SUCH ENTITY TO THESE TERMS OF USE. IF YOU DO NOT HAVE SUCH AUTHORITY, OR IF YOU DO NOT WISH TO ACCEPT THESE TERMS OF USE, YOU MUST NOT ACCESS OR USE THE HOSTED SERVICE OR AUTHORIZE ANYONE TO ACCESS OR USE THE HOSTED SERVICE. YOUR USE OF THE HOSTED SERVICE SHALL BE LIMITED BY THE NUMBER AND TYPE OF LICENSES PURCHASED UNDER YOUR CONTRACT FOR THE HOSTED SERVICE, PROVIDED, HOWEVER, THAT FOR CERTAIN HOSTED SERVICES IF APPLICABLE, YOU MAY HAVE THE OPPORTUNITY TO USE FLEX LICENSES, WHICH WILL BE INVOICED ACCORDING TO ACTUAL USAGE ABOVE THE CONTRACT LICENSE LEVEL. CONTACT AVAYA OR AVAYA'S CHANNEL PARTNER FOR MORE INFORMATION ABOUT THE LICENSES FOR THE APPLICABLE HOSTED SERVICE, THE AVAILABILITY OF ANY FLEX LICENSES (IF APPLICABLE), PRICING AND BILLING INFORMATION, AND OTHER IMPORTANT INFORMATION REGARDING THE HOSTED SERVICE.

Support Tools:

"AVAYA SUPPORT TOOLS"MEAN THOSE SUPPORT TOOLS PROVIDED TO PARTNERS OR CUSTOMERS IN CONNECTION WITH MAINTENANCE SUPPORT OF AVAYA EQUYIPMENT (E.G., SAL, SLA MON, AVAYA DIAGNOISTIC SERVER, ETC.) AVAYA SUPPORT TOOLS ARE INTENDED TO BE USED FOR LAWFUL DIAGNOSTIC AND NETWORK INTEGRITY PURPOSES ONLY. The customer is responsible for understanding and complying with applicable legal requirements with regard to its network. The Tools may contain diagnostic capabilities that allow Avaya, authorized Avaya partners, and authorized customer administrators to capture packets, run diagnostics, capture key strokes and information from endpoints including contact lists, and remotely control and monitor end-user devices. The customer is responsible for enabling these diagnostic capabilities, for ensuring users are aware of activities or potential activities and for compliance with any legal requirements with respect to use of the Tools and diagnostic capabilities on its network, including, without limitation, compliance with laws regarding notifications regarding capture of personal data and call recording.

Licenses

THE SOFTWARE LICENSE TERMS OR SUPPORT TOOLS LICENSE TERMS AVAILABLE ON THE AVAYAWEBSITE, 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 and Support Tools, 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. "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.

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.

Heritage Nortel Software
"Heritage Nortel Software" means the software that was acquired by Avaya as part of its purchase of the Nortel Enterprise Solutions Business in December 2009. The Heritage Nortel Software currently available for license from Avaya is the software contained within the list of Heritage Nortel Products located at http://support.avaya.com/LicenseInfo/ under the link "Heritage Nortel Products", or such successor site as designated by Avaya. For Heritage Nortel Software, Avaya grants Customer a license to use Heritage
Nortel Software provided hereunder solely to the extent of the authorized activation or authorized usage level, solely for the purpose specified in the Documentation, and solely as embedded in, for execution on, or (in the event the applicable Documentation permits installation on non-Avaya equipment) for communication with Avaya equipment. Charges for Heritage Nortel Software may be based on extent of activation or use authorized as specified in an order or invoice.

Support Tools: Avaya Support Tools are provided as an entitlement of Avaya Support Coverage (e.g., maintenance) and the entitlements are established by Avaya. The scope of the license for each Tool is described in its License terms and/or the applicable service description document.

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

Third Party Components
"Third Party Components" mean certain software programs or portions thereof included in the Software or Hosted Service may contain software (including open source software) distributed under third party agreements ("Third Party Components"), which contain terms regarding the rights to use certain portions of the Software ("Third Party Terms"). As required, information regarding distributed Linux OS source code (for those Products that have distributed Linux OS source code) and identifying the copyright holders of the Third Party Components and the Third Party Terms that apply is available in the Documentation or on Avaya's website at: http://support.avaya.com/Copyright or such successor site as designated by Avaya. You agree to the Third Party Terms for any such Third Party Components.

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.

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.

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.

All non-Avaya trademarks are the property of their respective owners, and "Linux" is a registered trademark of Linus Torvalds.

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.

Table of contents

Chapter 1: About this document

Avaya Dynamic Self Service application developers guide provides information on the Call Logger for the Dynamic Self Service (DSS) application. The document also provides information on building the OD modules for the External Voice Application (EVA) nodes.

Intended audience

This document is intended primarily for developers willing to use call trace logging mechanism of the DSS application. The developers can also use this document to build OD modules for DSS applications.

Send us your comments

Avaya appreciates any comments or suggestions that you might have about this documentation. Send your comments to infodev@avaya.com.

Chapter 2: Rest API client

The following section provides detailed information on Rest API client.

URL Format Convention for DSS rest services

URL Format

Base URL (<base_url>): http://<host>:8943/api-rest/<version>/<method_cathegory> or http://<host>/api-rest/<version>/<method_cathegory>

Complete URL: <base_url>/<method_name>

API Rest version format (<version>)

'v'+ <major-number-version>

Method-category assignment

<method_category>	<method_name>
authentication	• login • logout
Schedule	• schedule
Schedule Groups	• schedulegroups

URL examples

- http://135.122.60.181:8081/api-rest/v1/authentication/login.xml
- http://135.122.60.181:8081/api-rest/v1/authentication/logout.xml
- http://135.122.60.182:8081/api-rest/v1/schedulegroups/schedulegroup?name=Open Hours&includesDate=20160412T130000-0400
- http://135.122.60.182:8081/api-rest/v1/schedules/schedule?name=Call Center Hours&includesDate=20160412Z

JSON API

Methods

Login

This method has to be called to retrieve a session token that must be sent to any of the other methods. Without the session token, the rest of the methods would return an error response. The same session token can be reused multiple times, until expired or invalidated (through a call to the Logout method), and is valid for all the other methods.

To obtain a token, this method receives and validates a username and password pair. The username and password are managed as with any regular DSS user.

Each user can have maximum one valid token at any single time. Thus, calling the Login method with the credentials of a user who already has a valid token will cause the existing token to be invalidated and a new token generated and returned. In consequence, if multiple applications or multiple instances of the same application want to interact with this API, then each instance should use a different user.

The session token has to be sent to all the other methods in an app_token HTTP header.

Method details:

- Location: <base_url>/login.json
- Accepts: application/json
- HTTP Method: PUT
- Input Parameters: sent in the HTTP request content
    - username: the username for a DSS user. The username must be created using the DSS Admin Tool.
    - password: the password for the specific user.

Example of a request:

```
PUT /api-rest/v1/authentication/login.json HTTP/1.1

Accept: application/json, application/javascript, text/javascript

Content-Length: 55

Content-Type: application/json

Host: 127.0.0.1:808

Connection: Keep-Alive

Accept-Encoding: gzip,deflate
```

```
{"username":"admin","password":"Avaya1"}
```

Example of a response:

```
HTTP/1.1 200 OK

Server: Apache-Coyote/1.1

Content-Type: application/json

Transfer-Encoding: chunked

Date: Fri, 01 Nov 2013 13:24:07 GMT

55

{"username":"admin","sessionToken":"9708ac82-ae06-4c88-b447-
da308dd16374"}

0
```

Logout

Invalidates the session token. The system does not return an error if an invalid or empty token is sent. In such a case, the system only mentions that the system is not able to invalidate the specific token.

Method details:

- Location: <base_url>/logout.json
- Accepts: application/json
- HTTP Method: GET
- Input Parameters: only the session token is required, in the app_token HTTP header.

Example of a request:

```
GET /api-rest/v1/authentication/logout.json HTTP/1.1

Accept: application/json, application/javascript, text/javascript

app_token: 9708ac82-ae06-4c88-b447-da308dd16374

Host: 127.0.0.1:808

Connection: Keep-Alive

Accept-Encoding: gzip,deflate
```

Example of a response:

If the token was invalidated successfully:

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: application/json
Transfer-Encoding: chunked
Date: Fri, 01 Nov 2013 13:24:07 GMT

{"logout":true}
```

If the token was not invalidated, could be triggered why an invalid or missing token:

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Content-Type: application/json
Transfer-Encoding: chunked
Date: Tue, 10 Dec 2013 15:09:41 GMT

{"logout":false}
```

Schedule

Retrieves the schedule matching the given mandatory name, where 'name' is the query string input parameter. As an optional parameter, we can send a date and in that case it will return a Schedule matching the given name and that also includes the given date.

Method details:

- Location: <base_url>/schedule.json
- Accepts: application/json
- HTTP Method: GET

Input Parameters: only the session token is required, in the app_token HTTP header.

Schedule Groups

Retrieves the schedule group matching the given mandatory name, where 'name' is the query string input parameter. As an optional parameter, we can send a date and in that case it will return a Schedule group matching the given name in any of the Schedules the Schedule Group contains.

Method details:

- Location: <base_url>/schedulegroups.json
- Accepts: application/json
- HTTP Method: GET
- Input Parameters: only the session token is required, in the app_token HTTP header.

This method runs four REST GET Services. The details are as follows:

Service	Details
First service: http://localhost:8081/api-rest/v1/schedules?offset=0&limit=50	This service will list all available schedules. **Note**: **offset** and **limit** parameters are the optional attributes in this URL.

Second Service:
http://localhost:8081/api-rest/v1/schedules/schedule?name=schedule 1&includesDate=20160329
	This service will retrieve the schedule from the DB, which in this case, its name equals (case sensitive) to "schedule 1", and an (optional) second query parameter in which includes a given date. If the schedule table does not contain this date, an HTTP 404 error will be returned.
	
	Note: **name** parameter is a mandatory attribute in this URL.
	
	Example of a returned JSON object:
	
	`{`
	`"schedule": {`
	`"id": 50,`
	`"name": "schedule 1"`
	`}`
	`}`

Third Service:
 	
http://localhost:8081/api-rest/v1/schedulegroups?offset=0&limit=50
	This service will retrieve all the Schedule Groups (that is, current Holidays), and will return an array of Schedule Groups with the contained Schedules each.
	
	Example of a returned JSON object:
	
	{
	"data": [
	{
	"id": 50,
	"name": "A Schedule Group",
	"schedules": [
	{
	"id": 50,
	"name": "schedule 1"
	}
	]
	},
	{
	"id": 52,
	"name": "Empty Schedule Group",
	"schedules": []
	},
	{
	"id": 51,
	"name": "Yet another Schedule Group",
	"schedules": [
	{

	<pre>"id": 50,
	"name": "schedule 1"
	},
	{
	"id": 51,
	"name": "schedule 2"
	}
	]
	}
	]
	}</pre>

Fourth Service:
 	
http://localhost:8081/api-rest/v1/schedulegroups/schedulegroup?name=Another Schedule Group&includesDate=20160329Z
	This service will retrieve the Schedule Group which name matches "Another Schedule Group" and includes the (optional) given date.
	Example of a returned JSON object:
	
	<pre>{
	"scheduleGroup": {
	"id": 50,
	"name": "Another Schedule Group",
	"schedules": [
	{
	"id": 50,
	"name": "schedule 1"
	}
	]
	}
	}</pre>

Chapter 3: Call Logger

The Call Logger is a call trace logging mechanism for tracing the exact path that the caller went through. The Call Logger publishes the traces immediately after the caller leaves each node. Based on the implementation, the Call Logger either buffers the traces till the call ends, or processes the traces at real time.

At present, DSS provides two implementations: the Call Logger DB, which saves all traces into specific tables in the data base, and the Call Logger File, which saves all the traces to a specific file.
The following sections provide a description on both the options.

Call Logger DB

The Call Logger DB buffers all the call traces in memory and stores the call traces into the database when the call ends, all at a single time. This prevents the continuous database hits and prevents the caller to wait online.

The system stores every trace, that is, every node that the caller went through as a single record in the traceinfo database table. The record has the following attributes:

- **trace_info_id:** Is a unique number that identifies each call trace.
- **ucid:** The Universal Call ID. All traces within the same call will have the same UCID.
- **context_id:** Reserved for future usage.
- **source_node:** The source node from where the node transition was initiated.
- **source_node_type:** The type of the source node that the caller went through. The types are the same as the ones present in the DSS Admin Tool. The available node types are as follows: initial, announcement, menu, transferAgent, transferNumber, transferInternal, externalApplication, ddApplication, allocation, and hangUp.
- **call_center_service:** The service used by the transfer node, if any.
- **timestamp:** The time stamp in which the caller left the current node.
- **sequence_number:** The order in which the caller went through every node.
- **reached_by_link_code:** The link code used to reach the current node.
- **ani:** The Automatic Number Identifier, that is, the caller's phone number.
- **dnis:** The Dialed Number Identification Service, that is, the number dialed to reach current call flow.

- **transfer_to_vdn:** The VDN that is, the SIP URI, to which the call got transferred in a Transfer Node.
- **trace_info_type:** Used internally by DSS to identify some node types.
- **target_node:** The target node which was reached by the link code.
- **target_node_type:** The target node type which was reached by the link code.
- **callfow_name:** The name of the call flow to which all defined nodes belong to.
- **mpp_session_id:** The MPP session id used by the current call for tracing the DSS event with MPP events.

For more information on activating or deactivating Call Logger DB, see *Configuration and Properties* section in the *Installing and configuring Avaya Dynamic Self Service application guide*.

Call Logger JDBC

Create a 'traceinfo' table for Call Logger JDBC. The table must contain the columns and data types as given in the following table.

DBMS	Columns and data type
PostgreSQL	CREATE TABLE traceinfo (trace_info_id numeric(19,0), ucid character varying(255) NOT NULL, context_id character varying(255), source_node character varying(255), source_node_type character varying(255), call_center_service character varying(255), "timestamp" timestamp with time zone NOT NULL, sequence_number numeric(5,0) NOT NULL, reached_by_link_code character varying(255), ani character varying(255), dnis character varying(255), transfer_to_vdn character varying(255),

	trace_info_type character varying(255), target_node character varying(255), target_node_type character varying(255), uui character varying(255), callflow_name character varying(255), mpp_session_id character varying(255))
Oracle	CREATE TABLE "XXXXXX"."TRACEINFO" ("TRACE_INFO_ID" NUMBER(19,0), "UCID" VARCHAR2(255 BYTE) NOT NULL ENABLE, "CONTEXT_ID" VARCHAR2(255 BYTE), "SOURCE_NODE" VARCHAR2(255 BYTE) NOT NULL ENABLE, "SOURCE_NODE_TYPE" VARCHAR2(255 BYTE) NOT NULL ENABLE, "CALL_CENTER_SERVICE" VARCHAR2(255 BYTE), "TIMESTAMP" DATE NOT NULL ENABLE, "SEQUENCE_NUMBER" NUMBER(5,0) NOT NULL ENABLE, "REACHED_BY_LINK_CODE" VARCHAR2(255 BYTE), "ANI" VARCHAR2(255 BYTE), "DNIS" VARCHAR2(255 BYTE), "TRANSFER_TO_VDN" VARCHAR2(255 BYTE), "TRACE_INFO_TYPE" VARCHAR2(255 BYTE), "TARGET_NODE" VARCHAR2(255 BYTE), "TARGET_NODE_TYPE" VARCHAR2(255 BYTE), "UUI" VARCHAR2(255 BYTE), "CALLFLOW_NAME" VARCHAR2(255 BYTE), "MPP_SESSION_ID" VARCHAR2(255 BYTE))
Other DBMS such as	Same columns as above field. The data types must be analogous and respective

SQL Server, MySQL, etc.	to the DBMS.

To enable the Call Logger JDBC, you must edit the *<DSS_HOME>/configuration-files/flow-engine-settings.properties* file, and set callLogger.jdbc.active to true. You must also configure the following properties:

- callloggerjdbc.driver
- callloggerjdbc.url
- callloggerjdbc.username
- callloggerjdbc.password

After changing the JDBC Call Logger details, restart the flow engine using the following command:

```
/sbin/service dss-dialogs restart.
```

For more details, see the **Call Logger JDBC** section of the *Installing and configuring Avaya Dynamic Self Service application guide*.

Call Logger File

In contrast to the Call Logger DB, the Call Logger File stores all the call traces in a file in real time. This file is used primarily for troubleshooting purposes. Avaya recommends that you do not keep this file in an active state, as this might reduce performance significantly. The Call Logger File is implemented on top of Log4j.

For more information on activating or deactivating Call Logger file appender, see *Configuration and Properties* section in the *Installing and configuring Avaya Dynamic Self Service application guide.*

Chapter 4: Building an OD module for External Voice Application nodes

Add Orchestration Design (OD) as a module for External Voice Application (EVA) nodes. The OD module must return a string parameter with the name *routeCode*. DSS uses the *routeCode* parameter as the link code for the next node.

OD module can receive and return as many parameters as required. The name of the inbound and outbound parameters must match the parameter names added in the External Voice Application node.

Also, note that:

- DSS supports modules built with both OD 6 and OD 7 applications.
- DSS also supports VXML applications that are not built with OD application. In this case, the VXML application must be compatible with the Avaya Media Processing Platform (MPP).
- In the case of an OD module, the regular session parameters that are stored in the session object in OD are not available due to the OD architecture. Thus, the required information must be sent using the call flow variable and parameter assignment in the EVA node. The OD session parameters that are not available include most of the fields in the session OD variable, such as *session.dnis* and *session.ani*.
- The OD modules must be installed on a separate servlet container or a web server. The OD module cannot be installed on any of the Tomcat servers used by DSS.

Concatenate or Assign Module

Concatenate or Assign module is an OD application which allows users to join up to 10 strings into 1. DSS also uses this module to assign the value of a variable into another variable, for example UUI. This module is used in the External Voice Application node.

Input parameters

Call variable name	Description
input 01..10	These are the 10 input strings
separator	Use this parameter in case you want to concatenate the input strings using a separator.

Output parameters

- resultCode is evaluated inside the node.
- The value used is always default.
- If you do not use the mandatory parameters, the system will return an error.

Call variable name	Description
concatenated	The result displays the concatenation of the input01 to input10 in the given order with the string you have defined as separator between each input.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/dssConcatenate/Start
- Relative URL: /external-modules/dssConcatenate/Start

Consultative Transfer Module

Consultative Transfer module is an OD application which allows users to transfer a call to a specific VDN with the UUI attached. In case there is any issue with the transfer, the control remains in DSS and the node will returns the default link code. If there is any error or exception, the system will return an error link code.

Note: If you are using ICR Call Flow type, then UUI will not be propagated as UUI is not currently supported for ICR Call Flow module.

Input parameters

Call variable name	Description
vdn	A number to which a call is transferred.
aai	UUI. The content of this parameter is set in the UUI/AAI SIP header.
connectTimeout	The duration in seconds after which a transfer is timed out.

Output parameters

- routeCode is evaluated inside the node.
- Returns a noTransfer link code in case if a transfer is not completed. The options are:
    - Noanswer
    - noauthorization
    - baddestination
    - noroute
    - noresource
    - protocol.nn
    - unsupported.transfer.consultationx
    - unsupported.uri
- Returns the default link code in case transfer is successfully completed.
  Note: The link code is returned to DSS, even if a voice-related action is not performed, such as announcing a message, since we may still want to call a web service or otherwise end the call.

Call variable name	Description
cause	Displays a reason for not transferring one of the following codes: • noasnwer • noauthorization • baddestination • noroute • noresource • protocol.nn • unsupported.transfer.consultation unsupported.uri

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/consultation-transfer-module/Start
- Relative URL: /external-modules/consultation-transfer-module/Start

Context Store Integration Module

Context Store Integration module is an OD application developed for the EVA node.

Input parameters

contextID and operation are mandatory. The rest options are depended on the number of parameter the user wants to send or receive.

Call variable name	Description
contextID	A context identifier, usually UCID or EduID
operation	Retrieves information or uses UPSERT to create or update a context.
propertyKey01 .. 10	These are 10 property keys depending on which the operations are performed to specify which parameter is being sent or received.
propertyValue01 .. 10	When the operation is UPSERT, then you need to specify the values for each property key.

Output parameters

resultCode always shows the default values. If there is an error, then an error link code is returned.

Call variable name	Description
outPropertyValue01 .. 10	On each one of this outPropertyValueXX, the user receives the value of the propertyKeyXX defined in the input parameter.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/csIntegrationModule/Start
- Relative URL: /external-modules/csIntegrationModule/Start

Dynamic Routing Integration Module

Dynamic Routing Integration module is an OD application developed for the EVA node.

Input parameters

segmentationFunction () and interactionID_UCID are mandatory. The other parameters must be created as segParamKey and segParamValue.

Call variable name	Description
interactionID_UCID	UCID
decisionFunctionName	This is the name of the decision function to select from Dynamic Routing.
segmentationTableName	This is the name of the segmentation table to be used for the segmentation function defined.
segAttributeKey01 .. 10	These are 10 segmentation attribute names the user can provide to Dynamic Routing.
segAttributeValue01 .. 10	These are the 10 values for the segmentation attribute names.
customParamKey01..03	These are 3 Custom Parameter names the user can provide to Dynamic Routing.
customParamValue01..03	These are the 3 values for the Custom Parameter names.
outParamKey01..10	These are the 10 possible output parameters that are found in the extended response.

Output parameters

resultCode is evaluated inside the node.

Call variable name	Description
destinationName	This is the name of the selected

Call variable name	Description
	destinationunsupported.uri
destinationAddress	This is the address of the selected destination.
destinationType	This is the type of the selected destination.
countDecision	For details, see the latest *Dynamic Routing 3.1 admin guide* located at: https://support.avaya.com.
companyName	For details, see the latest *Dynamic Routing 3.1 admin guide* located at: https://support.avaya.com.
interactionID	For details, see the latest *Dynamic Routing 3.1 admin guide* located at: https://support.avaya.com.
locationName	This is the name of the location of the selected destination.
segmentationLabel	This is the label of the segmentation selected based on the parameters provided.
outParamValue01..10	These are 10 Output parameter values for the output parameter names defined when the module are invoked.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/drIntegrationModule/Start
- Relative URL: /external-modules/drIntegrationModule/Start

Get DateTime Module

Get DateTime module is an OD application which allows users to get the server current date or time.

Input parameters

Call variable name	Description
dateTimePattern	This is the format in which you wish to receive the date or time. If the format is not provided, the default format is yyyyMMdd. You can also include time as yyyyMMdd hh:mm:ss.
timeZoneCanonicalId	If a time zone is not defined, this will take the time zone of the server using the canonical id as given on the following webpage: http://joda-time.sourceforge.net/timezones.html .

Output parameters

resultCode always shows the default values. If there is an error, then an error link code is returned.

Call variable name	Description
formattedDateTime	This is the formatted date and time.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/getDateTime/Start
- Relative URL: /external-modules/getDateTime/Start

POM save Contact to List Integration Module

POM save Contact to List Integration module is an OD application developed for the EVA node.

Input parameters

Call variable name	Description
inputName01..10	This is the name of the property of the contact list the user wants to update from, currently

Call variable name	Description
	retrieves from the POM contact List.
inputValue01..10	This is the value the user wants for the inputName with the same number.

Output parameters

None

Possible link codes

Link code	Description
default	Invoked in case of the successful execution.
error	Invoked in case of an error in the execution.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/pom-save-contact-to-list/Start
- Relative URL: /external-modules/pom-save-contact-to-list/Start

POM Update CompletionCode Integration Module

pom-update-completion-code module is an OD application developed for the EVA node.

Input parameters

Call variable name	Description
completionCode	This is the string used for updating the completion code for this call.

Output parameters

None

Possible link codes

Link code	Description
default	Invoked in case of the successful execution.
error	Invoked in case of an error in the execution.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/pom-update-completion-code/Start
- Relative URL: /external-modules/pom-update-completion-code/Start

POM variables Integration Module

POMVariables module is an OD application developed for the EVA node.

Input parameters

Call variable name	Description
inputName01..10	This is the name of the property of the contact list the user wants to retrieve from. Currently retrieves from the POM contact List

Output parameters

Call variable	Description
outputValue01..10	This is the value from the contact list for the current call. The corresponding inputNameXX value set in the inputNameXX.

Possible link codes

Link code	Description
default	Invoked in case of the successful execution.
error	Invoked in case of an error in the execution.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/POMVariables/Start
- Relative URL: /external-modules/POMVariables/Start

Breeze Interoperability Module

Start an Engagement Designer workflow from DSS

This external module is used to start a given Engagement Designer (ED) workflow from DSS. An OD application workflow is added into the external modules server and can be invoked from an EVA node.

Preconditions

To use this module, you need to make sure that Experience Portal is on version 7.1.x and Engagement Designer is properly configured.

Configuring Engagement Designer

Procedure

1. Log in into the AEP.
2. Navigate to **System Configuration** > **EPM Servers** > **Data Storage Settings**.
3. In Engagement Development Platform, specify the Platform Host Address attribute with the Engagement Designer cluster hostname.

Input parameters

This module can send up to eight custom non-mandatory parameters to the Engagement Designer workflow. There are two mandatory parameters: *workflowName* and *eventType*.

The following are the full list of Input parameters:

Call variable name	Description
workflowName	The name of the workflow to be started.
eventType	The type of the event used by the workflow while starting the task.
parameterKey01..08	The parameter name defined in the Event Schema for Key01 up to Key08 or as many defined. **Note**: In case you want more parameters to send to the workflow, you must store those parameters into Context store and then retrieve the parameters from the Workflow. The current platform only allows you to send up to 8 parameters when a call flow is called directly from DSS.
parameterValue01..08	The value to be sent corresponding to the parameterKey01 up to parameterKey08 or as many defined.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/start-ed-workflow/Start
- Relative URL: /external-modules/start-ed-workflow/Start

Send a custom event to an Engagement Designer Workflow from DSS

This external module is used for sending a custom event to a given Engagement Designer (ED) workflow from DSS. An OD application workflow is added into the external modules server and can be invoked from an EVA node.

Preconditions

To use this module, you need to make sure that Experience Portal is on version 7.1.x and Engagement Designer is properly configured.

For configuring the Engagement Designer settings, see ***Error! Reference source not found.***.

Input parameters

This module can send up to eight custom non-mandatory parameters to the Engagement Designer workflow. In case that you need to send more than eight parameters to a workflow, you need to use Context Store. There are two mandatory parameters: *instanceId* and *eventType*.

The full lists of parameters are as following:

Call variable name	Description
instanceId	The instance ID for the running the workflow.
eventType	The type of the event used by the workflow while starting the task.
parameterKey01..08	The parameter name defined in the Event Schema for Key01 up to Key08 or as many defined. **Note**: In case you want more parameters to send to the workflow, you must store those parameters into Context store and then retrieve the parameters from the Workflow. The current platform only allows you to send up to 8 parameters when a call flow is called directly from DSS.
parameterValue01..08	The value to be sent corresponding to the parameterKey01 up to parameterKey08 or as many defined.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/send-ed-event/Start
- Relative URL: /external-modules/send-ed-event/Start

Audio Variable Module

An OD application audio-variable is added as part of the DSS EVA node. If you are using this module, you must configure the *audio-variable.properties* file located in the DSS *configuration-files* folder.

Property files

Property	Description
user	The user name of an existing user which will be used only for this module in DSS. For example: *AVUser*. This will be used to access the REST API

Property	Description
password	The password of the user.
pin	The user pin of the same user.
server	The IP added or HOST name of the server you want to access.
httpPort	This is the port number of the DSS web application. 8943 is the usually used port number.
httpsPort	If the DSS web application port is secured, you must use httpsPort.
backendContext	You must use as *api-rest* as the BackendContext value.
locale	In case of needing i18n, configure the locale required here.
timeout	The maximum time for service timeout in miliseconds.

Input parameters

decisionFunctionName, *segmentationTableName*, *dnis*, *ani*, and *ucid* are mandatory parameters. The rest should be created as *segParamKey* and *segParamValue*.

Call variable name	Description
audioID	ID or string configured in the audio ID property of the audio file from a prompt that you want to reproduce.

Output parameters

resultCode is evaluated inside the EVA node. If the result is ok, the value is default. If the audio ID is not found or the DB fails, an error occurs.

Module deployment

The audio-variable module is installed along with the DSS 2.9.0.0 application. To configure the audio-variable for EVA use, navigate to *http://<DSS-IP>:8945/audio-variable/Start*.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/audio-variable/Start
- Relative URL: /external-modules/audio-variable/Start

DateTimeFormatter Module

An OD application which allow users to transform a given date and time or a timestamp into a different date and time representation.

Input parameters

Call variable name	Description
inputDate	The date to format in the following format: yyyyMMdd.
inputTime	The time, in a separate field, that you want to include into the formatted output timestamp. If time is not provided, then the system adds the midnight time.
inputTimeZoneCanonicalId	If the time zone is not mentioned, the module uses the time zone of the server using the canonical ID: http://joda-time.sourceforge.net/timezones.html .
inputTimestamp	If there are two separate fields used for date and

Call variable name	Description
	time, you can use a single field also provided the input format used is in inputTimestampPattern.
inputTimestampPattern	This is the current inputTimestamp format defined.
outputDateTimePattern	This is the format in which you want to receive the date / time, the default is yyyyMMdd.

Note: inputDate, inputTime, and inputTimeZoneCanonicalID are required only if nothing is passed in to inputTimestamp and inputTimestampPattern fields. In case you have a custom date (or time, or date time) in a custom format and you need a different format for the same, you must leave inputDate, inputTime, and inputTimeZoneCanonicalId fields blank and you must send that value into the inputTimestamp field, specifying the inputTimestampPattern.

For example: If you are using the date in the format - date 11/23/2016 and you want to change the format to 20161123, then you must pass "11/23/2016" in inputTimestamp, and "MM/dd/yyyy" in inputTimestampPattern. Then, given that the output pattern we are requesting is the default one (yyyyMMdd), we must leave the outputDateTimePattern field blank.

Output parameters

resultCode is evaluated inside the node. The value used is always default. If you use the mandatory parameters, the system returns an error.

Call variable name	Description
formattedDateTime	The formatted date or time format.

Module deployment

This module is already installed in DSS v2.8.x.

Module usage from EVA

In the URL field, enter the following URL:

http://IP_ADDRESS:8082/dateTimeFormatter/Start

Here, you must replace the IP_ADDRESS value with the DSS IP address.

URL to use in the EVA node

- Full URL: http://<DSS_HOST>:8945/dateTimeFormatter/Start
- Relative URL: /external-modules/dateTimeFormatter/Start

Appendix I: Adding a variable as input parameter for all call flow type

DSS now allows you to define a global call flow variable for sending values to all the call flow type. This variable applies same global value across the call flow application. This value is defined in Configurable Application Variables (CAV) on the Experience Portal application.

Input parameters, such as query string parameters, will be automatically bound to Call Flow variables if they are defined.

Appendix II: Output variable required for ICR7

DSS sends all values which ICR expects from Wait Treatment Applications (WTA) and Error Handing Applications (EHA). Ensure that all variable defined in ICR must also be defined in the DSS application as Call Flow Variables so that values for all the required variables are send properly.

Appendix III: How to get the URL from a DSS audio file

About this task

To get the URL from a DSS audio file and reproduce the URL, you must use the DSS REST services. You need to use the REST services one for logging in and another one to get the audio URL.

Procedure

For logging in:

- **URL**: http://<DSS_HOST>:8943/api-rest/v1/authentication/login
- **Operation**: PUT
- **Headers**: None
- **Request**: Login JSON

```
{
  "username": "<dss_username>",
  "password": "<dss_password>"
}
```

- **Response**: Login JSON - Response

```
{
  "username": "<dss_username>",
  "sessionToken": "9b79400e-b615-49fe-8a48-9ff2391be9ac"
}
```

For getting the audio file URL:

- **URL**: http://<DSS_HOST>:8943/api-rest/v1/audio/get_audio_file_url
- **Parameters**:
  - **pin**: A valid pin from a user.
  - **audioId**: This will be the user defined Id from the audio you are trying to retrieve
- **Example**:http://<DSS_HOST>:8943/api-rest/v1/audio/get_audio_file_url?pin=12346&audioId=bestAgent
- **Operation**: GET
- **Headers**: app_token:<token obtained from login>
- **Response**: Login JSON – Response

```
{
  "message": "http://135.122.60.182:8944/audio-
cache/99256bc7-83ea-418f-9d0a-eb85afa7bef9.wav"
}
```

To have the URL into account:

You must use a unique login details for this task, as every time a user logs in, the previous tokens become invalid.