DR 3.2 Integration Guide

Dynamic Routing 3.2 Integration Guide

IMPORTANT
Updated Jan 16, 2018.
This document contains descriptions of all the public interfaces allowing 3rd party Applications to integrate with Dynamic Routing

1. DR 3.2 Integration Guide 2
1.1 DR 3.2 Admin API - REST Interface 3
1.2 DR 3.2 Alarms Setup and Description 17
1.2.1 DR 3.2 Alarm Descriptions 19
1.3 DR 3.2 CMS Connector Setup and Troubleshooting 21
1.3.1 DR 3.2 CMSC Troubleshooting 27
1.4 DR 3.2 ICR Integration 42
1.4.1 1. Installing and Configuring the Extension 45
1.4.2 2. Creating an SSA to Invoke the Extension 51
1.4.3 3. Dynamic Routing Configuration 62
1.4.4 4. Dynamic Routing - ICR Integration: Log Analysis 70
1.5 DR 3.2 Metrics Connector SDK (for Metrics Connectors) 134
1.6 DR 3.2 RDR Reference 139
1.7 DR 3.2 Routing Remoting Client SDK 151
1.8 DR 3.2 Routing REST Interface 155

DR 3.2 Admin API - REST Interface

- Overview
- Entities supported in V3 (DR3.1)
- What's new with V4 (DR3.2)
    - /v4/segmentationtables/rules/{id}
    - /v4/lookuptables
    - /v4/acds
    - /v4/companies
    - /v4/locations
    - /v4/agentgroups
    - /v4/applications
    - /v4/services
    - /v4/globalproperties
    - /v4/strategysettings
- Limits of POST, PUT, DELETE Commands
- Authentication and Authorization mechanism
- Operations supported
    - Retrieve operations (GET)
    - Insert Operations (POST)
    - Update Operations (PUT)
    - Delete Operations (DELETE)
- Error Codes
    - List of DR error codes
- Prerequisites for Segmentation Tables
- Prerequisites for Strategy Settings
- Prerequisites for Lookup Tables
- Prerequisites for Applications
- Prerequisites for Locations
- Prerequisites for Company
- Prerequisites for ACD
- DTO structure considerations
- Transactions
- Post Operation: Writing IDs to body

Overview

Dynamic Routing uses the Service Layer in Config Store to retrieve, insert, update or delete configuration entities needed for its routing operation. By using Routing Admin application, an operator can manage all these operations from a web interface, along with global and system properties, users and roles. Now, we present Admin API, a REST web service that enables an external systems to retrieve, insert, delete or modify Dynamic Routing configuration entities.

Entities supported in V3 (DR3.1)

V3 of the API supports CRUD operations on **Time tables, Lookup tables**, **Decision Functions**, **Segmentation Tables** and **Strategy Scripts**.

What's new with V4 (DR3.2)

- Max page size is limited to 20 entities; if a request asks for more than 20, the response will cap to 20.
- For Strategy Scripts and Decision Functions, the script code must be supplied now using standard JSON escaped strings. Base 64 is no longer supported

- Segmentation Table (ST) API changes: instead of a single entity containing ST metadata + Segmentation Rules + Strategy Settings, now metadata and rules are treated as different entities, to be able to better handle errors and validations, and produce much smaller payloads in most cases

Operation URL	http method	Description
/v4/segmentationtables/rules/{id}	GET	Get a single Segmentation Rule, by it's ID
/v4/segmentationtables/{tableId}/rules	DELETE	Delete a list of rules from the given Segmentation Table, the list of ID's needs to be provided in the body
/v4/segmentationtables/{tableId}/rules	GET	Retrieve a list of rules from the given Segmentation Table, the list is paginated (see swagger for further information)
/v4/segmentationtables/{tableId}/rules	POST	Insert a list of rules into the given Segmentation Table (see swagger for further information)
/v4/segmentationtables/{tableId}/rules	PUT	Update a list of rules in the given Segmentation Table (see swagger for further information)
/v4/segmentationtables/{tableId}/rules/{id}/moveRule	PUT	Move a rule in the Segmentation Table by one place in the specified direction (see swagger for further information)
/v4/segmentationtables/	DELETE	Delete a list of Segmentation Tables, the list of ID's needs to be provided in the body
/v4/segmentationtables/	GET	Retrieve a list of Segmentation Table, the list is paginated (see swagger for further information)
/v4/segmentationtables/	POST	Insert a list of Segmentation Tables (see swagger for further information)
/v4/segmentationtables/	PUT	Update a list of Segmentation Tables (see swagger for further information)
/v4/segmentationtables/{id}	GET	Get a single Segmentation Table, by it's ID
/v4/segmentationtables/{id}/getRulesOrder	GET	Get the order of a current set of rules, return a list of rule IDs (see swagger for further information)
/v4/segmentationtables/{id}/orderRules	PUT	Update the order of rules for the specified Segmentation Table (see swagger for further information)

- Added **LookupTables** (Metadata and single row operations)

	http method	Description
/v4/lookuptables	GET	Get a list of Lookup Tables' metadata. The list is paginated (see swagger for further information)
/v4/lookuptables/{id}	GET	Get a single Lookup Table's metadata, referenced by its ID
/v4/lookuptables	POST	Insert a list of Lookup Tables' metadata
/v4/lookuptables	PUT	Update a list of Lookup Tables' metadata
/v4/lookuptables	DELETE	Delete a list of Lookup Tables' metadata
/v4/lookuptables/{id}/entries	GET	Get the list of Lookup Table entries for a single table. The list is paginated (see swagger for further information)
/v4/lookuptables/{id}/entries/{key}	GET	Get one entry of a single Lookup Table, referenced by its key
/v4/lookuptables/{id}/entries	POST	Insert a list of Lookup Table entries
/v4/lookuptables/{id}/entries	PUT	Update a list of Lookup Table entries
/v4/lookuptables/{id}/entries	DELETE	Delete a list of Lookup Table entries

- Added **ACD** entity

Operation URL	http method	Description
/v4/acds	DELETE	Delete a list of ACDs
/v4/acds	GET	Get a list of ACDs, the result is paginated
/v4/acds	POST	Insert a list of ACDs
/v4/acds	PUT	Update a list of ACDs
/v4/acds/{id}	GET	Get a single ACD, referenced by its ID

- Added **Company** entity

Operation URL	http method	Description
/v4/companies	DELETE	Delete a list of Companies
/v4/companies	GET	Get a list of Companies, the result is paginated
/v4/companies	POST	Insert a list of Companies
/v4/companies	PUT	Update a list of Companies
/v4/companies/{id}	GET	Get a single Company, referenced by its ID

- Added **Location** entity

Operation URL	http method	Description
/v4/locations	DELETE	Delete a list of Locations
/v4/locations	GET	Get a list of Locations, the result is paginated
/v4/locations	POST	Insert a list of Locations
/v4/locations	PUT	Update a list of Locations
/v4/locations/{id}	GET	Get a single Location, referenced by its ID

- Added **Agent Group** entity

Operation URL	http method	Description
/v4/agentgroups	DELETE	Delete a list of Agent Groups
/v4/agentgroups	GET	Get a list of Agent Groups, the result is paginated
/v4/agentgroups	POST	Insert a list of Agent Groups
/v4/agentgroups	PUT	Update a list of Agent Groups
/v4/agentgroups/{id}	GET	Get a single Agent Group, referenced by its ID

- Added **Application** entity

Operation URL	http method	Description
/v4/applications	DELETE	Delete a list of Applications

/v4/applications	GET	Get a list of Applications, the result is paginated
/v4/applications	POST	Insert a list of Applications
/v4/applications	PUT	Update a list of Applications
/v4/applications/{id}	GET	Get a single Application, referenced by its ID

- Added **Service** entity

Operation URL	http method	Description
/v4/services	DELETE	Delete a list of Services
/v4/services	GET	Get a list of Services, the result is paginated
/v4/services	POST	Insert a list of Services
/v4/services	PUT	Update a list of Services
/v4/services/{id}	GET	Get a single Service, referenced by its ID

- Added **Global Property** (GP) entity

Operation URL	http method	Description
/v4/globalproperties	DELETE	Not supported
/v4/globalproperties	GET	Get a list of GPs, the result is paginated
/v4/globalproperties	POST	Insert a list of GPs
/v4/globalproperties	PUT	Update a list of GPs
/v4/globalproperties/{id}	GET	Get a single GP, referenced by its ID

- Added **StrategySettings** entity

Operation URL	http method	Description
/v4/strategysettings	DELETE	Delete a list of Strategy Settings
/v4/strategysettings	GET	Get a list of Strategy Settings, the result is paginated
/v4/strategysettings	POST	Insert a list of Strategy Settings
/v4/strategysettings	PUT	Update a list of Strategy Settings
/v4/strategysettings/{id}	GET	Get a single Strategy Settings, referenced by its ID
/v4/strategysettings/{name}/settings	GET	Get a single Strategy Settings, referenced by its Name and Segmentation Table Name

Limits of POST, PUT, DELETE Commands

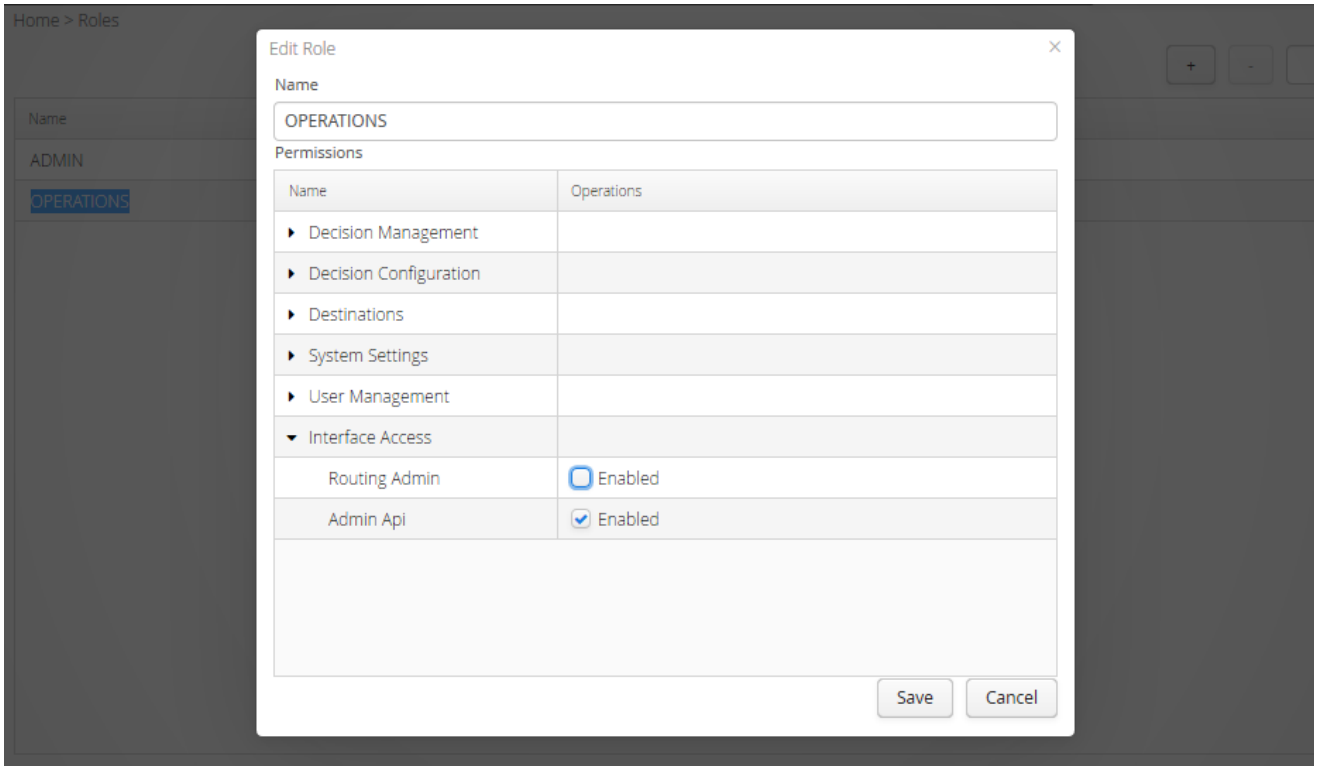
These limits are applicable to POST, PUT, DELETE commands:

- Admin API limits the number of entities that can be processed in a single request, in order to not overload the system. The limits are the sames ones imposed by Routing Admin. RA respects the limits set in System Properties, which can be adjusted up to a maximum value imposed by the product. If any request exceeds these limits, it will be rejected.

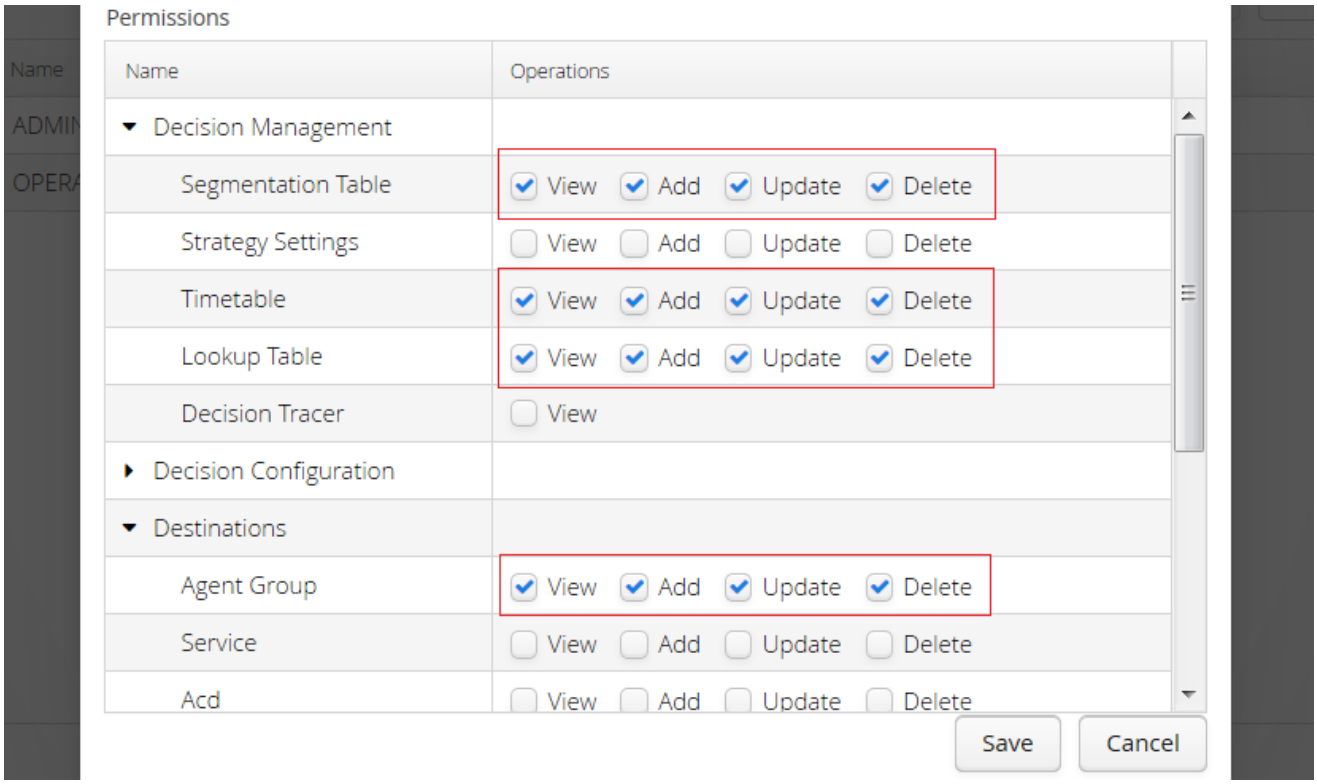
- the request body size is limited to 2 MB (mega bytes). If any incoming request is greater that this size, it will receive HTTP status code <413> and be rejected.
- To limit the size of headers, the IDs of newly created entities, are added to the body of the response, rather than the header.
- Admin API operations (especially with long lists) can take a long time (up to 15 minutes). If the request needs even more time than that, it will timeout and return an error code <504>.

Authentication and Authorization mechanism

Admin API supports Basic-Authentication. To consume this API, there must be a user in the system - created through Routing Admin - whose role enables access to Admin API, as illustrated in the following picture.



It is also possible to configure a more granular set of permissions to a given role, enabling some operations and not others. In the picture below, we can see the operations View, Add, Update and Delete enabled for entities such as Segmentation Table, Time table and Decision Tracer (among others), but not for Strategy Settings.



The Admin API client software must provide the following request headers, when invoking the API:

```
authorization: Basic ZHJhZG1pbjpBdmF5YTEyMw==
acept: application/json
Content-Type: application/json
```

Operations supported

We have enabled Swagger, where you can see the different operations supported. You can access Swagger at http://[IP]/dr-admin-api/swagger-ui.html

Dynamic Routing REST API

Configuration Data Model Management API

agent-group-rest-controller : Agent Group Rest Controller	Show/Hide | List Operations | Expand Operations
company-rest-controller : Company Rest Controller	Show/Hide | List Operations | Expand Operations
decision-function-rest-controller : Decision Functions operations	Show/Hide | List Operations | Expand Operations
location-rest-controller : Location Rest Controller	Show/Hide | List Operations | Expand Operations
lookup-table-rest-controller : Lookup Table Rest Controller	Show/Hide | List Operations | Expand Operations
segmentation-rule-rest-controller : Segmentation Rule Rest Controller	Show/Hide | List Operations | Expand Operations
segmentation-table-rest-controller : Segmentation Tables operations	Show/Hide | List Operations | Expand Operations
strategy-script-rest-controller : Strategy Scripts operations	Show/Hide | List Operations | Expand Operations
strategy-settings-rest-controller : Strategy Settings operations	Show/Hide | List Operations | Expand Operations
time-table-rest-controller : Time Tables operations	Show/Hide | List Operations | Expand Operations

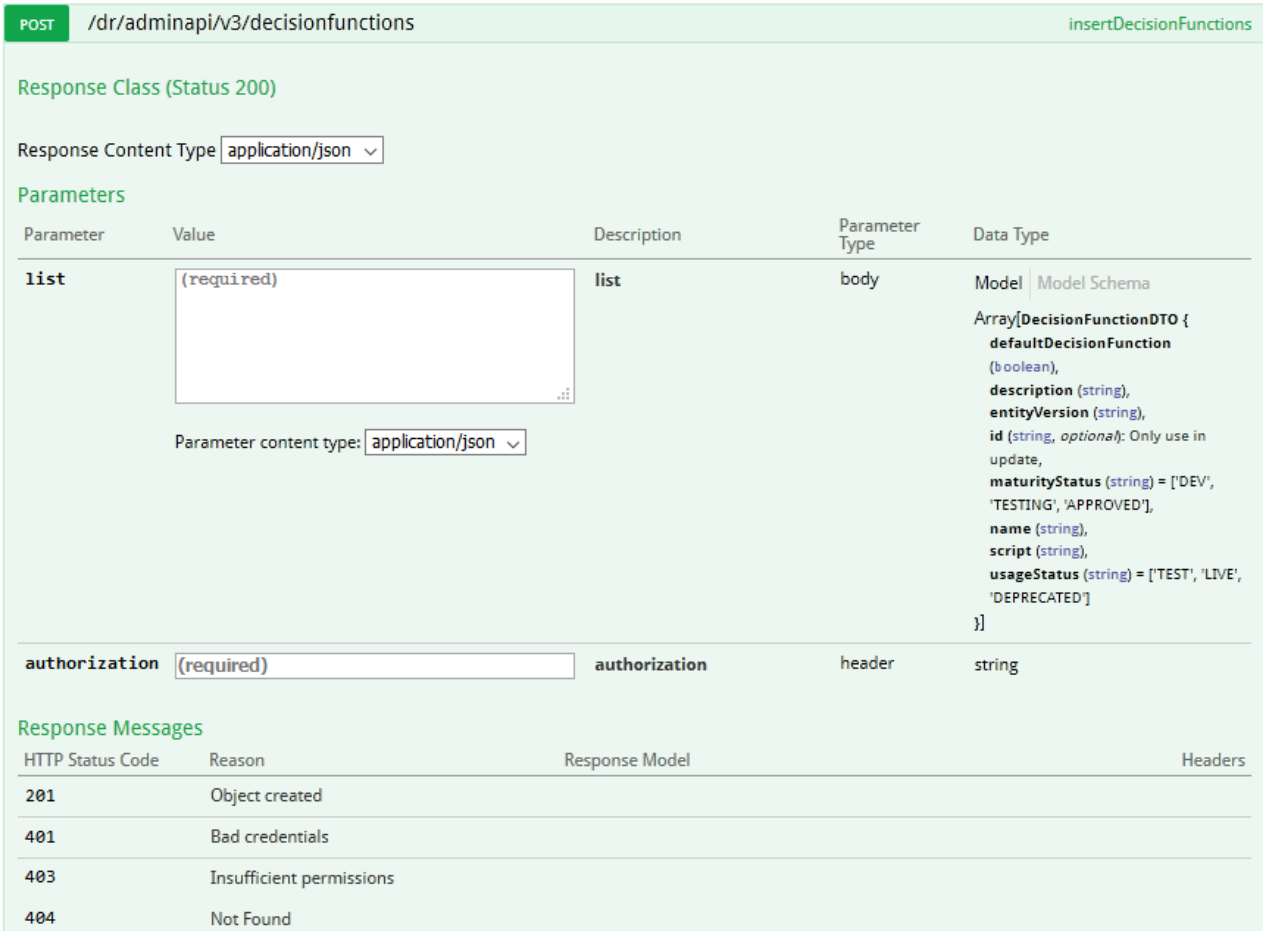
swagger | api-infos (/v2/api-docs?group=api-infos) | api_key | Explore

Dynamic Routing REST API

Configuration Data Model Management API

decision-function-rest-controller : Decision Functions operations — Show/Hide | List Operations | Expand Operations

DELETE	/dr/adminapi/v3/decisionfunctions	deleteDecisionFunctions
GET	/dr/adminapi/v3/decisionfunctions	getDecisionFunctions
POST	/dr/adminapi/v3/decisionfunctions	insertDecisionFunctions
PUT	/dr/adminapi/v3/decisionfunctions	updateDecisionFunctions
GET	/dr/adminapi/v3/decisionfunctions/{id}	getDecisionFunctionById



(*) Authorization: alphanumeric string with user and password encoded in Base64 (Basic authentication)

Retrieve operations (GET)

When the ID is provided in the URI, an entity is returned in JSON format. If the ID is not provided, a list of entities are returned in JSON format. See sample entities in JSON format in the Swagger document.

Pagination: we can indicate in the url, parameters for page and size, to retrieve a specific page and a number of elements on it
For instance http://10.130.124.34:8080/dr_admin_api_pu/dr/adminapi/v3/timetables**?page=1&size=10**

If the user indicates a page that doesn't exist, a link will be sent to the last page which has results.

In the Links header, the user will receive the link to the next and previous page if they exist. For example: Links: </dr_admin_api_pu/dr/adminapi/v3/timetables?page=2&size=1>; rel="next"

Insert Operations (POST)

Admin API supports bulk inserts. The client must not provide the IDs of the entities, since they are generated by DR.

The entity/ies to insert must not exist in Dynamic Routing, otherwise an error is returned.

See examples of JSON in the Swagger documentation

In the Location header, the client will receive the location of the created entities

Update Operations (PUT)

Admin API supports bulk updates. The client must provide the IDs of the entities to update.

The entity/ies to update must exist in Dynamic Routing, otherwise an error is returned.

See examples of JSON in the Swagger documentation

Delete Operations (DELETE)

Admin API supports bulk deletes. The client must provide the IDs of the entities to delete.

We must supply in the body of the request, a list of IDs of entity/ies to delete in JSON format.

The entity/ies to delete must exist in Dynamic Routing, otherwise an error is returned.

```
["1ded846e-7dae-403c-abb5-c12711863891","2ded846e-7dae-403c-abb5-c12711863
891","3ded846e-7dae-403c-abb5-c12711863891"]
```

Error Codes

All successful operations return HTTP code 200

If a user could not be authenticated, 401 is returned

If a user could be authenticated but he has no permissions, 403 is returned

If the request body size is greater than 2 MB (mega bytes), 413 is returned and the request is rejected.

If a request needs more than 15 minutes to process, it will timeout and return error code 504.

All other errors return code 500 plus a DR error code, a description, the id of the object and the id of the transaction in JSON format


```
{
  -"Error": {
       "error_code": "DR_5008"
       "description": "Entity already exists"
    }
    "id": "3d523635-4d84-4094-b6ad-3f92c098a471"
    "trxId": "12ecfaa4-dda5-4723-b035-84b9d50cb847"
}
```


List of DR error codes

Error	Description
DR_5008	Entity already exists
DR_5009	Entity does not exist
DR_5010	Entity could not be inserted
DR_5011	Entity could not be updated

DR_5012	Entity could not be deleted
DR_5013	Entity could not be retrieved
DR_5014	Entity name already exist
DR_5015	IDs do not match
DR_5016	Referece error (prerequisites don't exists)
DR_5017	Unique field duplicated
DR_5018	IDs must not be sent
DR_5099	Operation error *

(*) Logs and audit records can be found in /opt/avaya/dr/platform/gigaspaces-xap-premium/logs folder

Prerequisites for Segmentation Tables

For insert and update operations on Segmentation Rules the following constraints apply:

Strategy Settings with the specified name should exist.

Destination Aliases, should match the Destination Aliases in the Strategy Script (related through Strategy Settings).

If Strategy Settings contain time dependant variables, the Timetables should exist.

Destinations used in the Segmentation Rule, should exist.

For insert and update operations on Segmentation Tables the following constraints apply:

If LUT dimensions are provided, the provided LUT should exist.

Prerequisites for Strategy Settings

The Strategy Settings entity, is related to a Strategy Script, and a Timetable, and the following constraints apply:

The Timetable referenced by name must exist.

The intervals of the Timetable, need to be the same intervals defined in the time dependent variables.

The Strategy Script referenced should exist.

The Destination Aliases, variables, and variable types need to match the Strategy Script definitions.

Prerequisites for Lookup Tables

The operations related to Lookup Tables are categorised into: metadata operations and entries operations.

The Lookup Table referenced by ID must exist.

When inserting a LUT, not only its name must be unique but all the Dimension's names must be also unique.

Prerequisites for Applications

Mandatory Fields:

a. ID (for POST operation should not be sent)
b. Name
c. Channel Type (VOICE, CHAT, EMAIL, VIDEO, SMS, FACEBOOK, TWITTER, WEB)
d. Address: At least one entry of address must be exist.

Prerequisites for Locations

Mandatory Fields:

a. ID (for POST operation should not be sent)
b. Name

Optional:

a. Global Properties: should already exist to use them.

Prerequisites for Company

Mandatory Fields:

a. ID (for POST operation should not be sent)
b. Name

Optional:

a. Global Properties: should already exist to use them.

Prerequisites for ACD

Mandatory Fields:

a. ID (for POST operation should not be sent)
b. Name

Optional:

a. Global Properties: should already exist to use them.

DTO structure considerations

The values of a Strategy Settings, consist of a list of variables and their values, but since there are many different kinds of variables, the way to build the values differs.

The property "values" is a map, in which the key of the map, is the **name** of the variable, and the value is the **value** of the variable; the value can be a single value, or a set of values if the variable is "per Destination", "per Interval" or "per Destination Interval".

Example:

If the Strategy Settings has 4 variables {"SIMPLE_VAR", "PER_INTERVAL_VAR", "PER_DEST_VAR", "PER_DEST_INTERVAL_VAR"}, 2 Destination Aliases {"DA1", "DA2"}, and a Timetable with {"INTERVAL1", "INTERVAL2"} the following structure describes the settings:

property key	map key	map value	1st nesting level	2nd nesting level	Description(not json code)

values :	{				**values** is a map of key value pairs, where key is the variable name and value is the variable value (a JSON Object with fields: discriminator, value, and multiValue)
	SIMPLE_VAR:	{ value : "variable_value" }			Simple variable with the name SIMPLE_VAR and a single value = "variable_value"
	PER_DESTINATION_VAR:	{ multiValue : [	{discriminator : "DA1",		Per destination variable with the name PER_DESTINATION_VAR, which has a value for each Destination Alias in the referred script, the Destination Aliases are DA1 and DA2. In this case single value is empty, but multi value is filled with each Destination Alias value
			value : "variable_value" },		
			{ discriminator : "DA2",		
			value : "variable_value"}		
		] }			
	PER_INTERVAL_VAR:	{ multiValue : [	{discriminator : "INTERVAL1",		The structure of per interval variables is the same as per destination variables, but instead of using Destination Aliases as discriminators, use Interval names instead.
			value : "variable_value" },		
			{discriminator : "INTERVAL2",		
			value : "variable_value"}		
		] }			
	PER_DEST_INTERVAL_VAR:	{ multiValue : [	{discriminator : "DA1",		Per destination interval variables, have 2 levels multi variable values, first level is the Destination Alias, and inside that, there is another multi value, which is the Interval name.
			multiValue: [	{discriminator : "INTERVAL1",	
				value : "variable_value"},	
				{discriminator : "INTERVAL2",	
				value : "variable_value"}	
			]} ,		
			{discriminator : "DA2",		
			multiValue: [	{discriminator : "INTERVAL1",	
				value : "variable_value"},	
				{discriminator : "INTERVAL2",	
				value : "variable_value"}	
			]}		
		]}			

	}				

Another consideration is that as the variables support default values, the "value" field can be null (last leaf) in case the variable value keeps the default value. If the variable value contains a null, the default value for the variable will be used

Example:

```
"values": {

"routeC": {},

"queueCapThreshold": { "multiValue": [

{ "discriminator": "DEST_ALIAS1", "value": "00" },

{ "discriminator": "DEST_ALIAS2", "value": "00" },

{ "discriminator": "DEST_ALIAS3" },

{ "discriminator": "DEST_ALIAS4" },
```

In the example, routeC is a single variable, with the default value; queueCapTheresold is a per Destination variable, and the script has 4 Destination Aliases: DEST_ALIAS1 and DEST_ALIAS2 each have a defined value in the Strategy Settings, DEST_ALIAS3 and DEST_ALIAS4 have no specified value in the Strategy Settings, so the value from the Strategy Script Metadata will be used instead.

Transactions

Every operation on a single object is a transaction, but for bulk operations, all the objects are considered part of a single transaction. For instance, if we have an insert of a list of Timetables, if one of them fails all the other inserts are rolled back.

Post Operation: Writing IDs to body

POST /dr/adminapi/v4/agentGroup insertAgentGroups

Response Class (Status 200)

Response Content Type application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type

list	`},`
	`"nativeId": "3",`
	`"locationName": "Cincinnati",`
	`"companyName": "Accenture",`
	`"acdName": "AC3",`
	`"propertiesValues": [`
	`{`
	`"propertyName": "manualDisable",`
	`"propertyValue": "NO"`
	`},`
	`{`
	`"propertyName": "OpenHours",`
	`"propertyValue": "Default_OpenHours"`
	`}`
	`]`
	`},`
	`{`
	`"name": "A4",`
	`"channel": "VOICE",`
	`"addresses": {`
		list	body	Model Model Schema
				`[ { "acdName": "string", "companyName": "string" "id": "string", "locationName": "string "nativeId": "string", "propertiesValues": [ { "propertyName": "st "propertyValue": "s }`
				Click to set as parameter value

	Parameter content type: application/json			
authorization	basic ZHJhZG1pbjpBdmF5YTEyMyE=	authorization	header	string

Request Headers


```
{
  "Accept": "application/json",
  "authorization": "basic ZHJhZG1pbjpBdmF5YTEyMyE="
}
```


Response Body


```
{
  "return": "OK",
  "LOCATION": "/dr-admin-api/dr/adminapi/v4/agentGroup/",
  "Ids": [
    "83bf988e-123a-492b-b0fe-4392bc6e72f1",
    "a2020429-a1fa-444b-a4fc-5d8060d39e4b",
    "c1223f4d-d213-4f57-a56c-ebb4ed3e91b8"
  ]
}
```


Response Code

201

Response Headers


```
{
  "date": "Tue, 11 Jul 2017 09:58:14 GMT",
  "server": "nginx/1.10.2",
  "connection": "keep-alive",
  "content-length": "204",
  "content-type": "application/json"
}
```

DR 3.2 Alarms Setup and Description

- Configuring Alarms on System Manager (SMgr)
- Configuring each Dynamic Routing node to point to System Manager
- Dynamic Routing SNMP Threshold Configuration

Configuring Alarms on System Manager (SMgr)

PREREQUISITES: Avaya Aura System Manager 6.2 or newer

In order for System Manager to display DR3 alarms properly, a "**Common Alarm Definition File**" must be configured and installed on SMgr by creating an **Extension Pack**.. The extension pack for Dynamic Routing is available under Dynamic Routing Downloads at http://support.avaya.com

To install the extension pack on a running SMgr, follow these steps:

1. Copy the Dynamic Routing Extension Pack to the System Manager (this directory will be used for the $PATH_TO_EXTENSION_PACK} information in step 4.
2. Log into SMgr as root user.
3. Navigate (cd) to this folder: *$MGMT_HOME/plug/install/unix*
4. Execute the following tool
    - `./install_plugin_files.sh false Postgres 'jdbc:postgresql://localhost/avmgmt?user=${user}&password=${password}' $JBOSS_HOME avmgmt ${PATH_TO_EXTENSION_PACK}`

      Only the $PATH_TO_EXTENSION_PACK} information should be replaced by the Dynamic Routing Extension Pack file name and full path.
5. Log into System Manager web console and go to *Home / Services / Inventory / Manage Elements* and add a new entry with the following attributes:
    a. Type: Other Applications
    b. Name: a name for the administration site, e.g. "Routing Admin"
    c. Node: IP/hostname to access the administration site where Routing Admin is installed
    d. Protocol: URI
    e. Access profile type: EMURL
    f. Host: IP/hostname to access the administration site where Routing Admin is installed
    g. Port: 80 if Protocol is http (or 443 if Protocol is https)
    h. Path: /dr-admin/
    i. Order: 0

Configuring each Dynamic Routing node to point to System Manager

Edit the *<DR_HOME>/dr/config/processing-units/snmp.properties* file and set the System Manager server IP address/port and the community string for sending alarms.

Example:

```
snmp.trapDestinations=10.133.15.27/10162
snmp.oid=.1.3.6.1.4.1.6889.2.78
snmp.community=asmcpe
```

Notes:

Do not change the snmp.oid, change only the SMgr IP Address

To apply the changes, a Dynamic Routing Cluster Services restart is required.

- stop dr/cluster-services
- start dr/cluster-services

IMPORTANT:

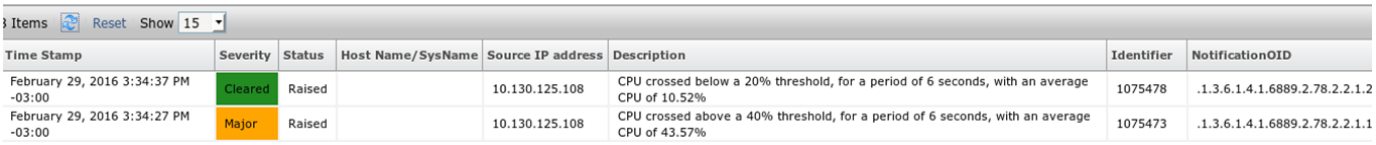
After restarting the services, it is recommended to execute the Sanity Test Checklist procedure described in the Dynamic Routing Cluster Management Guide.

Dynamic Routing SNMP Threshold Configuration

Dynamic Routing listens for events notified by GigaSpaces, that are configured in the alerts.xml file, and raises SNMP alarms.

To change the default values, edit the file /opt/Avaya/external/gigaspaces-xap-premium-10.2.0-ga/config/alerts/alerts.xml

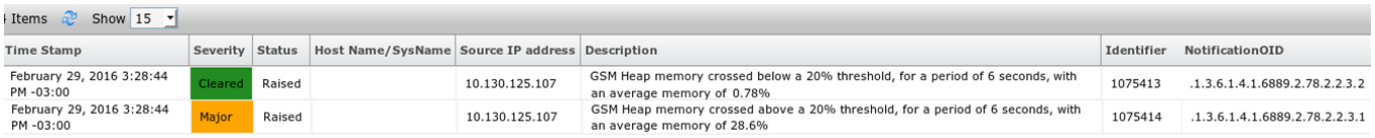
Example:



Items Reset Show 15

Time Stamp	Severity	Status	Host Name/SysName	Source IP address	Description	Identifier	NotificationOID
February 29, 2016 3:34:37 PM -03:00	Cleared	Raised		10.130.125.108	CPU crossed below a 20% threshold, for a period of 6 seconds, with an average CPU of 10.52%	1075478	.1.3.6.1.4.1.6889.2.78.2.2.1.2
February 29, 2016 3:34:27 PM -03:00	Major	Raised		10.130.125.108	CPU crossed above a 40% threshold, for a period of 6 seconds, with an average CPU of 43.57%	1075473	.1.3.6.1.4.1.6889.2.78.2.2.1.1

CPU SNMP Alarm displayed in SMgr Interface



Items Show 15

Time Stamp	Severity	Status	Host Name/SysName	Source IP address	Description	Identifier	NotificationOID
February 29, 2016 3:28:44 PM -03:00	Cleared	Raised		10.130.125.107	GSM Heap memory crossed below a 20% threshold, for a period of 6 seconds, with an average memory of 0.78%	1075413	.1.3.6.1.4.1.6889.2.78.2.2.3.2
February 29, 2016 3:28:44 PM -03:00	Major	Raised		10.130.125.107	GSM Heap memory crossed above a 20% threshold, for a period of 6 seconds, with an average memory of 28.6%	1075414	.1.3.6.1.4.1.6889.2.78.2.2.3.1

GSM Memory Alarm displayed in SMgr Interface

For each alarm, there are 3 different parameters to set:

```
<property key="high-threshold-perc" value="80" />,
<property key="low-threshold-perc"   value="60" />
<property key="measurement-period-milliseconds" value="60000" />
```

Available Thresholds:

Note: to apply the thresholds, a system restart is required.

- stop dr/cluster-services
- start dr/cluster-services

IMPORTANT

After restart the services, it is recommended to execute the Sanity Test Checklist procedure described in the Dynamic Routing Cluster Management Guide.

DR 3.2 Alarm Descriptions

Description of Dynamic Routing alarms can be found below. The text for each message is defined in property file alarm_messages_en_EN.properties

Type	OID	Alarm Type	Raised by	Message Key	Message
SNMP	.1.3.6.1.4.1.6889.2.78.5.4.1.1 .	Critical		StatePULifecycleEventListener.pu.status.changed	Raised when PU status cha
SNMP	1.3.6.1.4.1.6889.2.78.5.4.1.3	Critical		StatePULifecycleEventListener.pu.status.changed	Raised when PU is remove
SNMP	.1.3.6.1.4.1.6889.2.78.2.1.6	Major		AcdMetricsStatusProcessingListener.above.critical.mark	ACD Metrics stale percenta above critical threshold
SNMP	.1.3.6.1.4.1.6889.2.78.2.1.5	Major		AcdMetricsStatusProcessingListener.above.warning.mark	ACD Metrics stale percenta above warning threshold
SNMP	.1.3.6.1.4.1.6889.2.78.3.3.1.1	Major		ConfigStoreFeature.filter.destination.by.alias.name	Raised when an error occu data from Config Store
SNMP	.1.3.6.1.4.1.6889.2.78.3.3.2.1	Major		ConfigStoreFeature.get.acd.by.id	
SNMP	.1.3.6.1.4.1.6889.2.78.3.3.3.1	Major		ConfigStoreFeature.get.acd.by.name	
SNMP	.1.3.6.1.4.1.6889.2.78.3.3.4.1	Major		ConfigStoreFeature.get.agentgroup.by.name	
SNMP	.1.3.6.1.4.1.6889.2.78.3.3.5.1	Major		ConfigStoreFeature.get.agentgroup.list.by.service	
SNMP	.1.3.6.1.4.1.6889.2.78.3.3.8.1	Major		ConfigStoreFeature.get.agentgroup.by.acd.name	
SNMP	.1.3.6.1.4.1.6889.2.78.3.3.9.1	Major		ConfigStoreFeature.get.application.by.name ConfigStoreFeature.get.company.by.name ConfigStoreFeature.get.global.property ConfigStoreFeature.get.location.by.name ConfigStoreFeature.get.service.by.name ConfigStoreFeature.get.strategy.by.name ConfigStoreFeature.get.timetable.by.name	
SNMP	.1.3.6.1.4.1.6889.2.78.3.5.1.1	Major		MetricsFeature.get.metrics	Raised when a metric or cc found in the space
SNMP	.1.3.6.1.4.1.6889.2.78.3.5.2.1	Major		MetricsFeature.get.metrics.by.owner.id.and.metric.name	
SNMP	.1.3.6.1.4.1.6889.2.78.3.5.3.1	Major		MetricsFeature.get.counter.total	
SNMP	.1.3.6.1.4.1.6889.2.78.3.5.4.1	Major		MetricsFeature.get.metrics.acd.status	
SNMP	.1.3.6.1.4.1.6889.2.78.3.6.1.1	Major		SegmentationFeature.do.segmentation	Raised if an error occurs du segmentation, e.g.: Segme does not exist
SNMP	.1.3.6.1.4.1.6889.2.78.3.2.2.7	Major		RoutingService.get.df.script	Raised if a Decision Functi while processing a decisior
SNMP	.1.3.6.1.4.1.6889.2.78.3.2.2.8	Major		RoutingService.no.destination.selected.validation.enforced	Raised if no Destination is System Property 'contextStore.enforceDestS is set to 'YES'
SNMP	.1.3.6.1.4.1.6889.2.78.3.2.2.9	Major		RoutingService.no.destination.address.selected.validation.enforced	Raised if no Destination Ad selected and System Prope 'contextStore.enforceDestS is set to 'YES'
SNMP	.1.3.6.1.4.1.6889.2.78.3.2.2.10	Major		RoutingService.result.with.error.code	Raised when an error code a decision request
SNMP	.1.3.6.1.4.1.6889.2.78.3.2.2.5	Major		RoutingService.script.timeout	Raised if the execution time exceeds the timeout define
SNMP	.1.3.6.1.4.1.6889.2.78.3.2.2.5	Major		RoutingService.script.error	Raised if an error occurs w a Script

SNMP	.1.3.6.1.4.1.6889.2.78.3.2.2.6	Major		RoutingService.decision.general.error	Raised if an undefined erro processing a decision requ
SNMP	.1.3.6.1.4.1.6889.2.78.3.2.2.11	Major		RoutingService.decision.save.error	Raised if an error occurs w decision response
SNMP	.1.3.6.1.4.1.6889.2.78.3.8.1.1	Major		RateLimiter.limit.exceeded=Server is currently under heavy load	Raised when the decision r reaches the defined limit
SNMP	.1.3.6.1.4.1.6889.2.78.2.1.1	Major		MetricsStatusUpdater.add.duplicated.status	Error trying do add a duplic status into the grid for a sou ACD); the Add operation is system start-up or when a r added
SNMP	.1.3.6.1.4.1.6889.2.78.2.1.2	Major		MetricsStatusUpdater.metrics.status.not.found	Error querying for a metrics does not exist
SNMP	.1.3.6.1.4.1.6889.2.78.6.1.1.1	Warning		LicensingAlertsProcessor.license.state.grace.period	Raised when License state changes to GRACE_PERIC
SNMP	.1.3.6.1.4.1.6889.2.78.6.1.1.2	Major		LicensingAlertsProcessor.license.state.feature.not.enabled	Raised when License state changes to FEATURE_NO
SNMP	.1.3.6.1.4.1.6889.2.78.6.1.1.3	Major		LicensingAlertsProcessor.license.state.not.enough.counted.licenses	Raised when License state changes to NOT_ENOUGH CENSES
SNMP	.1.3.6.1.4.1.6889.2.78.9.1.1	Major		KafkaPersistence.error.saving.rdr	There is an error saving RD database
SNMP	.1.3.6.1.4.1.6889.2.78.6.9.1.2	Warning		KafkaIntegration.connection.up	Kafka is up and running
SNMP	.1.3.6.1.4.1.6889.2.78.6.9.1.3	Major		KafkaIntegration.connection.error	There is a connection issue
SNMP	.1.3.6.1.4.1.6889.2.78.6.9.1.4	Major		KafkaIntegration.error.saving.message.object-persisted	There is an error publishing Kafka
SNMP	.1.3.6.1.4.1.6889.2.78.6.9.1.5	Major		Elasticsearch.connection.up	Elasticsearch is up and run
SNMP	.1.3.6.1.4.1.6889.2.78.6.9.1.6	Major		Elasticsearch.connection.error	There is a connection issue Elasticsearch
SNMP	.1.3.6.1.4.1.6889.2.78.2.2.1.1	Major		CpuUtilizationAlert	Raised when CPU utilizatio
SNMP	.1.3.6.1.4.1.6889.2.78.2.2.1.2	Cleared		CpuUtilizationAlertCleared	Raised when CPU utilizatio normal
SNMP	.1.3.6.1.4.1.6889.2.78.2.2.2.1	Major		PhysicalMemoryUtilizationAlert	Raised when physical mem high
SNMP	.1.3.6.1.4.1.6889.2.78.2.2.2.2	Cleared		PhysicalMemoryUtilizationAlertCleared	Raised when physical mem back to normal
SNMP	.1.3.6.1.4.1.6889.2.78.2.2.3.1	Major		HeapMemoryUtilizationAlert	Raised when heap memory high
SNMP	.1.3.6.1.4.1.6889.2.78.2.2.3.2	Cleared		HeapMemoryUtilizationAlertCleared	Raised when heap memory back to normal
SNMP	.1.3.6.1.4.1.6889.2.78.2.2.4.1	Major		MirrorPersistenceFailureAlert	Raised when there is a failu persistence
SNMP	.1.3.6.1.4.1.6889.2.78.2.2.4.2	Cleared		MirrorPersistenceFailureAlertCleared	Raised when mirror persist normal

DR 3.2 CMS Connector Setup and Troubleshooting

- CMS Connector and Related Components
- CMS Configuration
- CMS Connector Configuration
  - cluster-locators.properties
  - cmsConfiguration.xml
  - cmsReportConfiguration.xml
  - Adding new ACD to Routing Admin
  - Re-IP the CMS Connector
- CMS Metrics
- Appendix A - Modify RTA Configuration

CMS Connector and Related Components

There are typically:

- a pair of CMSs (**CMS Primary**, **CMS Secondary**) working in Active/Standby mode; one CMS sending data to the CMS Connectors.
- a pair of CMS Connectors (**CMSC1** and **CMSC2**) working in Active/Active mode; both receiving data from the CMS and both sending data to one CORE per DC (the other COREs within the same DC, sync themselves).
- all **CORE**s working in Active mode; all containing a snapshot of real-time metrics, and all servicing decision requests.

CMS Configuration

The CMS needs to have the appropriate **report** to send the correct metrics to DR3. This report is called *dr304*. (the *dr304.gem* is available on the Dynamic Routing installation package *<MOUNT_POINT/Tools*)

Ensure that on the CMS Server(s):

- the minimum rt_socket version is **4.3.23 (if not, follow these steps):**
  - the .gem file for the report exists */export/home/pserv/rt_socket/GSM/dr304.gem*
  - the report is configured in the rta.conf file: */export/home/pserv/rt_socket/rta.conf*
- one rt_socket feed is required per ACD. It is a CMS limitation.
- the *hosts* file has entries for the CMS Connectors IP Addresses and hostnames.
- the *rta.conf* file contains configuration as shown below. Each 'session' section encapsulates information for sending an RT Socket report to one CMS Connector. Configure a 'session' for each CMS Connector.

```
# rta.conf - set configuration variables
#
set -a        # export all variables set in this file
HACMS=yes     # change to yes if HA CMS and auto-failover desired
#
#------------------------------- Session 1 -------------------------------
#----------------- (with EXAMPLE settings, NOT defaults) -----------------
HOST1=rtsocket               # the receiving server's host name in /etc/hosts
PORT1=7200                   # the receiving server's port
ACD1=1                       # ACD being monitored
POPTS1=""                    # applicable command line options
REPORT1=dr304                # respective custom report name
MONITOR_LIST1="1-2000"       # skills to monitor
REFRESH1=30                  # respective report refresh rate
DEST_APP1="Dynamic Routing"  # destination app for rt_socket or Generic-RTA
#------------------------------- Session 2 -------------------------------

HOST2=rtsocket2              # the receiving server's host name in /etc/hosts
PORT2=7200                   # the receiving server's port
ACD2=1                       # ACD being monitored
POPTS2=""                    # applicable command line options
REPORT2=dr304                # respective custom report name
MONITOR_LIST2="1-2000"       # skills to monitor
REFRESH2=30                  # respective report refresh rate
DEST_APP2="Dynamic Routing"  # destination app for rt_socket or Generic-RTA
```

To modify the *rta.conf* file, you need to know the VI editor. For more information, see Appendix A.

CMS Connector Configuration

The files below need to be correctly configured on the CMS Connectors.

In order to:

- setup the connection to Dynamic Routing locators, *configure cluster-locators.properties*
- setup the connection to CMS RT Socket servers, configure: *cmsConfiguration.xml*
- interpret the reports sent from the CMSs, configure: *cmsReportConfiguration.xml*

IMPORTANT:

- All files are in /opt/Avaya/cms-connector/config/application
- If changes are made to the *cmsConfiguration.xml,, cmsReportConfiguration.xml* or *configure cluster-locators.properties* files, the CMS Connector needs to be restarted for the changes to take effect.

cluster-locators.properties

This file is used to configure the communication between the CMS Connector and Dynamic Routing servers (Management Services Nodes). Each Data Center has one or two lookup locators that should be configured in this file, one line per Data Center:

```
cluster1.lookupHosts=172.28.128.3,172.28.128.4
cluster2.lookupHosts=172.28.128.4,172.28.128.5
```

Note:

If the deployment is for only one DC, only cluster1.lookuphosts should be configured (do not add or remove the cluster2 line).

cmsConfiguration.xml

This file is used to configure which CMS (RT_Socket) connections feed the CMS Connector.

Note: At any given time, the CMS Connector accepts only one RT Socket session (either from CMS Primary or CMS secondary). Similarly, only one CMS (RT Socket) sends data at a time; usually CMS Primary sends data, but if the primary goes down, then CMS Secondary sends data after the fail-over timeout.

IMPORTANT: The same configuration is needed on CMS Connector 1 (**CMSC1**) and CMS Connector 2 (**CMSC2**).

The IP Addresses are those of the **CMS Primary** and **CMS Secondary** respectively.

```
<?xml version="1.0" encoding="UTF-8"?>
<cmsConfiguration>
   <cms>
      <name>CMSpair1</name>
      <primary>135.122.60.98</primary>
      <secondary>135.122.60.99</secondary>
    </cms>
</cmsConfiguration>
```

The file configuration can be extended if there are multiple CMS pairs feeding the CMS Connectors, each CMS pair monitors a different set of Agent Groups. For example:

```
<?xml version="1.0" encoding="UTF-8"?>
<cmsConfiguration>
   <cms>
      <name>CMSpair1</name>
      <primary>135.122.60.98</primary>
```

```
    <secondary>135.122.60.99</secondary>
  </cms>

  <cms>
    <name>CMSpair2</name>
    <primary>135.122.60.100</primary>
    <secondary>135.122.60.101</secondary>
  </cms>
</cmsConfiguration>
```

cmsReportConfiguration.xml

The *cmsReportConfiguration.xml* file can be found under the *<DR_HOME>/cms-connector/config/application/* directory.

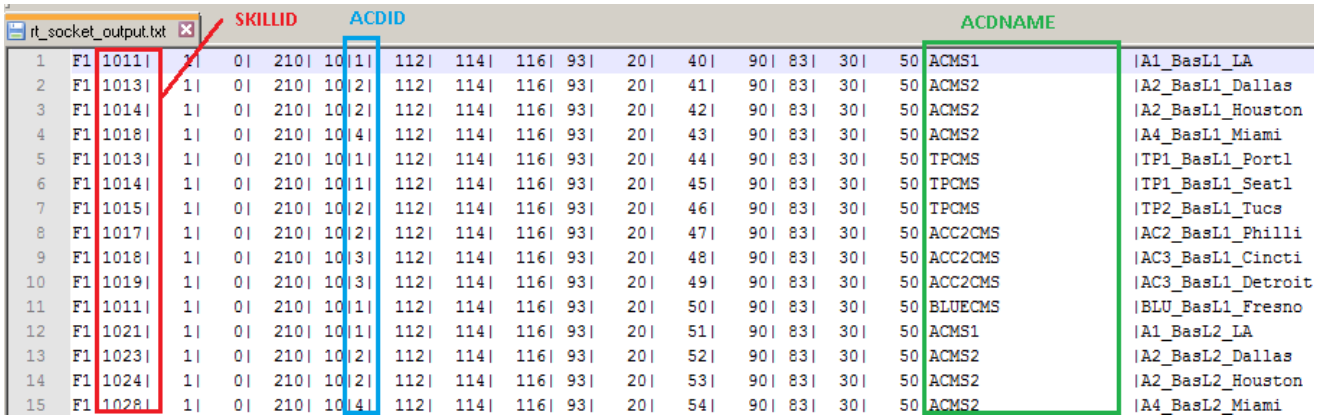
Make sure you configure the following:

report name="**dr304**":

primaryPort="7200"

<fields>: Field mapping has to be done to match the CMS RT_Socket report (on the CMS server). More details in the XML file attached below. No changes are needed to the field definitions when using the default **dr304** report. The default RTS report contains the following fields, using "(column-number)FIELD" notation:

(1)F1, (2)**SPLIT**, (3)INQUEUE, (4)AVAILABLE, (5)ASA, (6)ABNCALLS, (7)**ACDID**, (8)OLDESTCALL, (9)AVGTALKTIME, (10)AVGABNTIME, (11)STAFFED, (12)EWTHIGH, (13)EWTMEDIUM, (14)EWTLOW, (15)PERCENTINSL, (16)SERVICELEVEL, (17)CALLSOFFERED, (18)**ACDNAME**, (19)SKILLNAME



1	F1	1011|	1|	0|	210|	10	1|	112|	114|	116|	93|	20|	40|	90|	83|	30|	50	ACMS1	|A1_BasL1_LA
2	F1	1013|	1|	0|	210|	10	2|	112|	114|	116|	93|	20|	41|	90|	83|	30|	50	ACMS2	|A2_BasL1_Dallas
3	F1	1014|	1|	0|	210|	10	2|	112|	114|	116|	93|	20|	42|	90|	83|	30|	50	ACMS2	|A2_BasL1_Houston
4	F1	1018|	1|	0|	210|	10	4|	112|	114|	116|	93|	20|	43|	90|	83|	30|	50	ACMS2	|A4_BasL1_Miami
5	F1	1013|	1|	0|	210|	10	1|	112|	114|	116|	93|	20|	44|	90|	83|	30|	50	TPCMS	|TP1_BasL1_Portl
6	F1	1014|	1|	0|	210|	10	1|	112|	114|	116|	93|	20|	45|	90|	83|	30|	50	TPCMS	|TP1_BasL1_Seatl
7	F1	1015|	1|	0|	210|	10	2|	112|	114|	116|	93|	20|	46|	90|	83|	30|	50	TPCMS	|TP2_BasL1_Tucs
8	F1	1017|	1|	0|	210|	10	2|	112|	114|	116|	93|	20|	47|	90|	83|	30|	50	ACC2CMS	|AC2_BasL1_Philli
9	F1	1018|	1|	0|	210|	10	3|	112|	114|	116|	93|	20|	48|	90|	83|	30|	50	ACC2CMS	|AC3_BasL1_Cincti
10	F1	1019|	1|	0|	210|	10	3|	112|	114|	116|	93|	20|	49|	90|	83|	30|	50	ACC2CMS	|AC3_BasL1_Detroit
11	F1	1011|	1|	0|	210|	10	1|	112|	114|	116|	93|	20|	50|	90|	83|	30|	50	BLUECMS	|BLU_BasL1_Fresno
12	F1	1021|	1|	0|	210|	10	1|	112|	114|	116|	93|	20|	51|	90|	83|	30|	50	ACMS1	|A1_BasL2_LA
13	F1	1023|	1|	0|	210|	10	2|	112|	114|	116|	93|	20|	52|	90|	83|	30|	50	ACMS2	|A2_BasL2_Dallas
14	F1	1024|	1|	0|	210|	10	2|	112|	114|	116|	93|	20|	53|	90|	83|	30|	50	ACMS2	|A2_BasL2_Houston
15	F1	1028|	1|	0|	210|	10	4|	112|	114|	116|	93|	20|	54|	90|	83|	30|	50	ACMS2	|A4_BasL2_Miami

It is **mandatory** to define the key mappings in this file for each Agent Group.

For example: if the row is

<when key="**1ACMS1**" use="**f15ba477-9216-4bf5-968f-8444d21cdbc4**" />

Key="**1ACMS1**" is a combination of **ACDID** and **ACDNAME**, which are columns **7** and **18** from the RT Socket report.

|F1| 6| 0| 857| 0| 0| **1**| 0| 25| | 861| 0| 65535| 65535| 100| 0| 9| **ACMS1**| skill name

"**f15ba477-9216-4bf5-968f-8444d21cdbc4**" : is the **ACD entity ID** in Dynamic Routing for this ACD.

Adding new ACD to Routing Admin

For adding a new ACD to DR3, we have to follow these steps to make sure that the metrics are properly handled by the CMS Connector and sent to the DR3 cluster.

- Add a new ACD in Routing Admin.
- The **ACD entity ID** can be found in RA, by clicking on the ACD of interest and looking at the ACD Edit pop-up window.
- Copy that "ID".
- Open *cmsReportConfiguration.xml*

- Add an entry in the <key-mapping> section

  e.g. <when key="1ACMS1" use="f15ba477-9216-4bf5-968f-8444d21cdbc4" /> <!-- DR ACD Name = ACDLark1 -->

- Save the cmsReportConfiguration.xml file
- Restart the CMS Connector service
- Make sure the Native ID of the Agent Groups, linked to the ACD, match the Skill ID (column 2) in the rt socket feed.

Note: if there are **two CMS Connectors**, perform these steps on **both**

Find attached a sample file: *cmsReportConfiguration.xml*

Note: in order to view the CMSC feed (as in the example above),you need to change the log level as follows:

- Edit the file *<DR_HOME//cms-connector/config/logging/cms-connector-logging.xml*
- Set the log lvl to DEBUG

```
<logger name="com.avaya.ept.dr.services.connectors" level="DEBUG"
    additivity="false">
    <appender-ref ref="CMS_CONNECTOR" />
</logger>
```

- Save the file
- Wait until the connector refreshes the configuration (max 30 seconds), and you will see metrics in the file *<DR_HOME>/cms-connector/logs/application/cms-connector.log*
- Reset the log lvl back to INFO, by editing and saving the file again

```
<logger name="com.avaya.ept.dr.services.connectors" level="INFO"
    additivity="false">
    <appender-ref ref="CMS_CONNECTOR" />
</logger>
```

Re-IP the CMS Connector

In some cases, the CMS Connector may require re-IPing of its IP Address, for example, if a CMS Connector is installed with a temporary IP Address at a customer site, and then shipped to an outsourcer location (external call center provider) that requires a new IP Address. To re-IP the CMS Connector follow the steps below:

1. Edit the file */opt/Avaya/cms-connector/config/application/nicAddrFile.txt* and replace the old IP Address with the new IP Address and restart the CMS Connector service (stop dr/cms-connector; start dr/cms-connector).
2. Make correct name resolutions in the /etc/hosts file.

If a different CMS is also being used, remember to change the CMS IP Address in the file **cmsConfiguration.xml** as well. See section below.

CMS Metrics

This is the list of fields in "**dr304**" report:

	CMS/RTS Fields (columns)	RTS Field size	Type	DR3 Metric Name	Description
1	F1	2	string		Static "F1" string
2	SPLIT	4	number	SPLIT	The number of the split/skill for which data was collected i.e. the **Skill ID**
3	INQUEUE	4	number	INQUEUE	The number of split/skill ACD calls that are currently waiting in queue for this split/skill
4	AVAILABLE	4	number	AVAILABLE	The number of agents that are currently available in this split/skill
5	ANSTIME/ACDCALLS	5	number	ASA	Avg. speed of answer. The length of time that is spent by callers in queue or ringing before an agent answers the call.

6	ABNCALLS	3	number	ABNCALLS	Abandon calls. The number of CALLSOFFERED that are abandoned while in queue or ringing at an agent position
7	ACD	1	number	ACDNUM	The ACD number for which data was collected i.e. the **ACD ID**
8	OLDESTCALL	5	number	OLDESTCALL	Oldest call waiting. The number of seconds that the oldest split/skill ACD call has waited in queue or ringing
9	ACDTIME/ACDCALLS	5	number	AVGTALKTIME	Avg. ACD talk time, calculated as ACDTIME/ACDCALLS
10	ABNTIME/ABNCALLS	5	number	AVGABNTIME	Avg. time to abandon, calculated as ABNTIME/ABNCALLS
11	STAFFED	4	number	STAFFED	Number of Staffed agents in the split/skill, i.e. currently logged in (regardless of their Busy/ACW/AUX state)
12	EWTHIGH	5	number	EWTHIGH	Expected wait time (high.) The EWT metric for HIGH-PRIORITY calls in the skill/split
13	EWTMEDIUM	5	number	EWTMEDIUM	Expected wait time (medium). The EWT metric for MEDIUM-PRIORITY calls in the skill/split
14	EWTLOW	5	number	EWTLOW	Expected wait time (low). The EWT metric for LOW-PRIORITY calls in the skill/split
15	100*(ACCEPTABLE/CALLSOFFERED)	3	number	PERCENTINSL	The percentage of ACDCALLS that are answered by an agent within the predefined acceptable service level from the total calls offered to the ACD. Calculated as 100*(ACCEPTABLE/CALLSOFFERED)
16	SERVICELEVEL	4	number	SERVICELEVEL	The number of seconds within which calls must be answered or connected in order to be considered acceptable
17	CALLSOFFERED	5	number	CALLSOFFERED	The number of calls that queued to the split/skill and that completed during the interval
18	ACDNAME	20	string	ACDNAME	ACD Name (a unique ACD identifier string defined at CMS level) i.e. the **ACD Name**
19	SKILLNAME	20	string	SKILLNAME	Name of the split/skill when available. If not defined the skill ID is provided by CMS
	Separators	19			Pipe "|" to be used as separator between fields
	Total RECORD SIZE	128			

Appendix A - Modify RTA Configuration

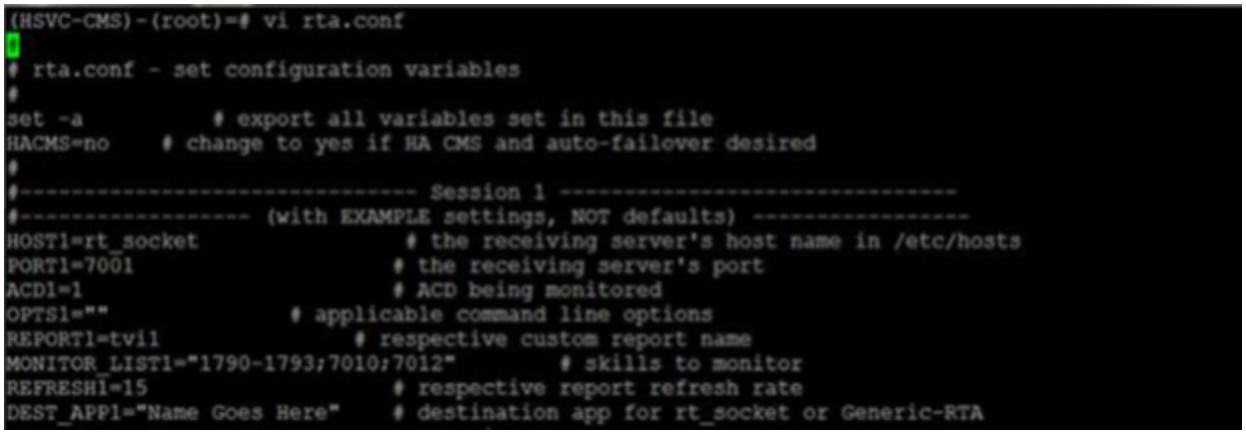
1. Login to CMS server as **root** using a telnet or ssh tool

2. Type command **cd** */export/home/rt_socket* on the terminal as shown below and hit **Enter**

3. Make a copy of *rta.conf* file before making any modifications. Command to run as an example

Command: cp rta.conf rta.conf.orig

4. Type command **vi** *rta.conf*. "**Monitor_List1**" has the skill range defined for session 1, so you can choose to either add a new skill number or range, or modify/delete an existing skill number. After making the required changes, press **ESC** and type **wq!** to save and quit the changes.



```
(HSVC-CMS)-(root)-# vi rta.conf
#
# rta.conf - set configuration variables
#
set -a          # export all variables set in this file
HACMS=no    # change to yes if HA CMS and auto-failover desired
#
#------------------------------ Session 1 -------------------------------
#------------------ (with EXAMPLE settings, NOT defaults) -----------------
HOST1=rt_socket               # the receiving server's host name in /etc/hosts
PORT1=7001                   # the receiving server's port
ACD1=1                       # ACD being monitored
OPTS1=""               # applicable command line options
REPORT1=tvi1                # respective custom report name
MONITOR_LIST1="1790-1793;7010;7012"        # skills to monitor
REFRESH1=15                  # respective report refresh rate
DEST_APP1="Name Goes Here"    # destination app for rt_socket or Generic-RTA
```

Note: The maximum supported input parameter split/skill field characters is **50**. The recommendation is not to break skills into small group unless necessary. If you choose to break the skill range or define single skill entries, then you should separate them using semicolon.

Example of breaking skill range or multiple skill/skill range in Monitor_List

Monitored_List1=”1790-1793;7010;7012”

5. Run command **./stoprta** and enter the session number to stop the session for which you have modified the skill

6. Run command **./startrta** and enter the session number to start the stopped session for which you have modified the skill

DR 3.2 CMSC Troubleshooting

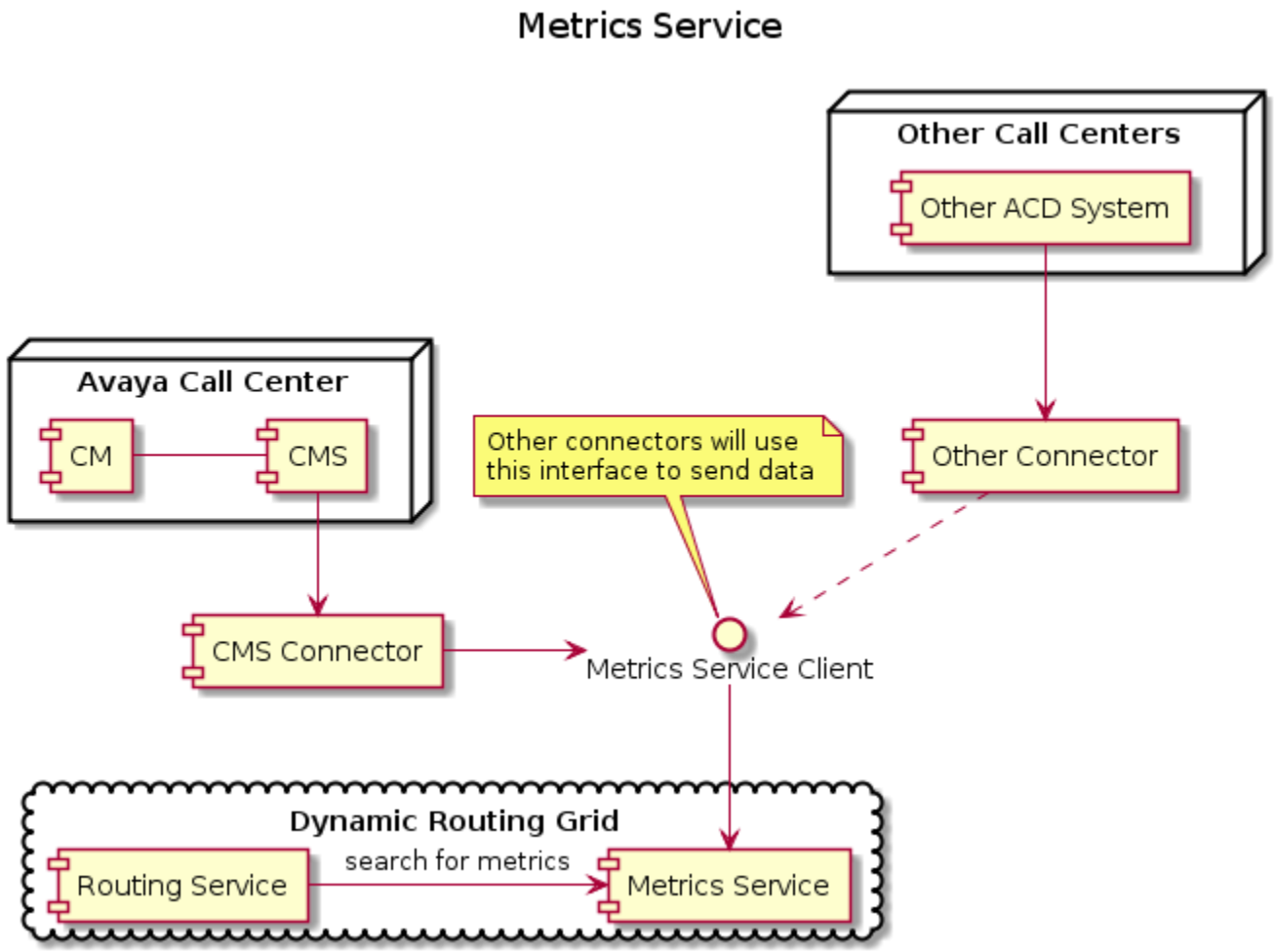
- Overview
- Diagnosing Issues with the Overall Solution
  - NTPD Service
- Diagnosing Issues on the CM
  - Using SAT
    - Using Function Keys in SAT
  - Checking out the CMS Configuration on CM
- Diagnosing Issues on the CMS
  - CM Related Settings
  - RT-Socket Channel Setup
    - RT-Socket Enablement
  - Verifying Custom Reports on CMS ASC II
- Diagnosing Issues on the CMS Connector (CMSC)
  - CMS Connector Logfile
  - Configuring syslog using logback.xml
  - CMS-Connector Connection Status Errors
- Diagnosing Issues in Dynamic Routing

NOTE:

The is not meant to be a comprehensive guide on the CM nor CMS. This guide contains some pointers only (and may be out of date). For debugging of the CM and CMS, please rely on the CM and CMS official customer documentation.

Overview

The picture below depicts the components involved in processing metrics.



In the case of the **Avaya CM**, ACD metrics are collected by the **CMS**. The **CMS** sends reports to the **CMS Connector** which in turn processes the metrics, maps them to Dynamic Routing entities such as ACDs, and sends them to Dynamic Routing. Dynamic Routing can then associate the metrics with the correct Agent Groups configured in its database and store the metrics in the data grid (memory space) for quick retrieval and use in making real-time decisions (such as routing).

Similarly, **Other** connectors can feed Dynamic Routing with metrics.

Diagnosing problems with the metric feed can take several steps, as the issue can be with the CMS, and/or CMS Connector, and/or Dynamic Rouitng.

Diagnosing Issues with the Overall Solution

NTPD Service

The time on all the Dynamic Routing nodes and CMS Connectors must be the same.

All servers used for Dynamic Rouitng should have Network Time Protocol (**NTP**) set-up to point to the same NTP Source Server (detailed instructions on Red Hat website).

In the following example, it seems the *ntpd* configuration is different.

```
[root@OTHER1 cms-connector]# date
Mon Nov 23 10:18:23 EST 2015
[root@OTHER1 cms-connector]# ps -ef |grep ntpd
root     14200 13294  0 10:18 pts/0    00:00:00 grep ntpd
root     19390     1  0 Oct26 ?        00:00:04 ntpd -s 135.9.81.247

[root@MGMT1 logs]# date
Mon Nov 23 10:18:54 EST 2015
[root@MGMT1 logs]# ps -ef |grep ntpd
ntp       2062     1  0 Oct26 ?        00:00:04 ntpd -u ntp:ntp -p /var/run/ntpd.pid -g
root     22005 15833  0 10:18 pts/1    00:00:00 grep ntpd
```

There is a difference of 30 seconds between the two servers; this is unacceptable. They are using different NTP settings... this would have to be fixed.

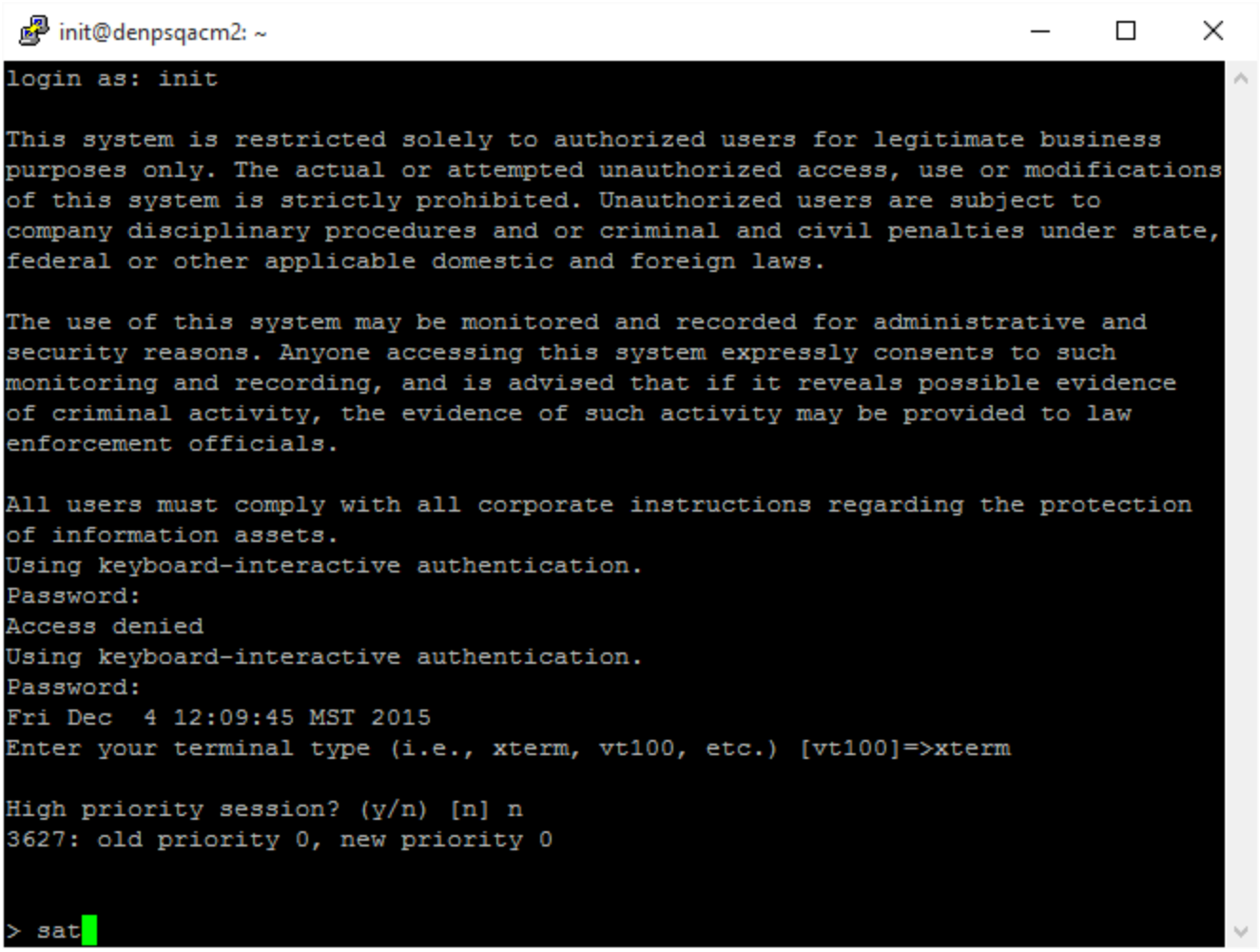
Diagnosing Issues on the CM

In this section, we are going to cover the configuration necessary on the CM, so it is able to communicate with the CMS.

Using SAT

To view the CM configuration use an application called SAT (System Access Terminal). To connect to it:

1. Log into the CM.
2. In the command shell, run the command ***sat*** and enter, again, the password.
3. When asked about the terminal type, we need to select ***SUNT***. This is very important, since it is the terminal type that allows us to use the function keys we are going to learn about soon.
4. The command prompt is displayed and, from there, we can start editing or viewing CM configuration.



Using Function Keys in SAT

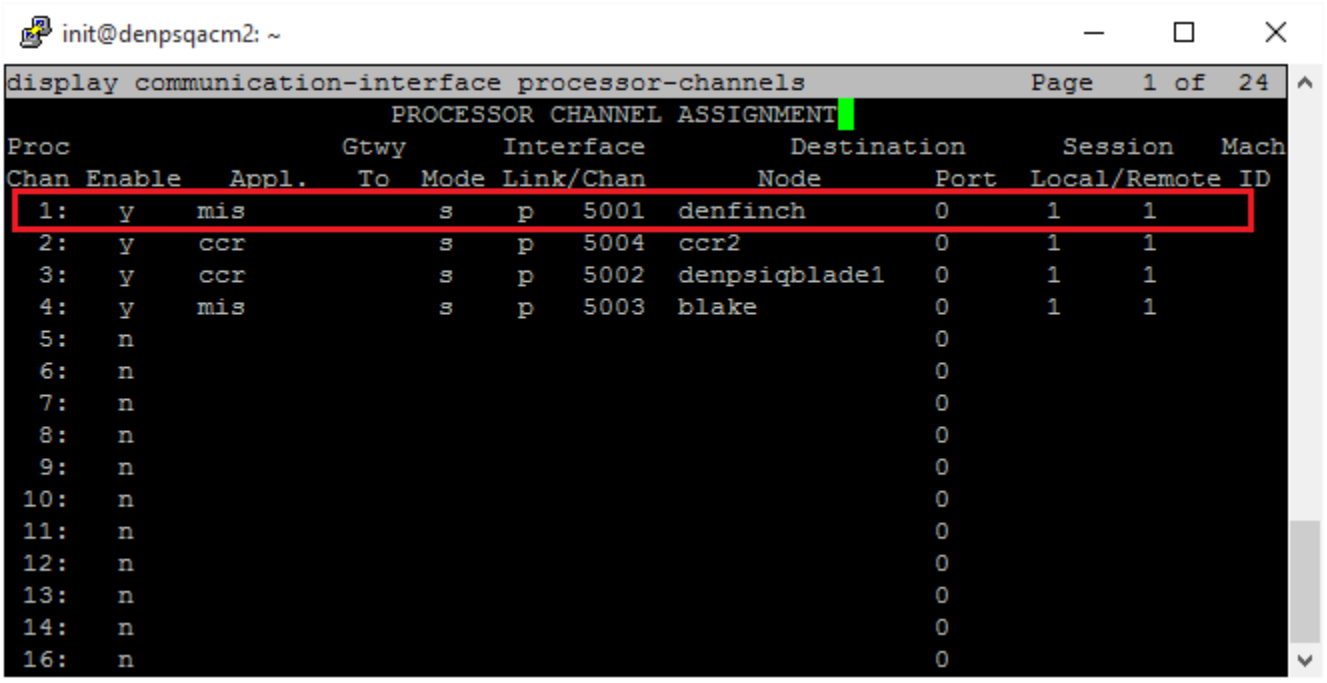
Function key	Operation performed
F1	Cancel
F2	Refresh
F3	Accept/Confirm
F4	Clear
F5	Help
F6	Update
F7	Next Page
F8	Previous Page

Also, we can use the arrow keys to navigate through the various fields of the pages.

Checking out the CMS Configuration on CM

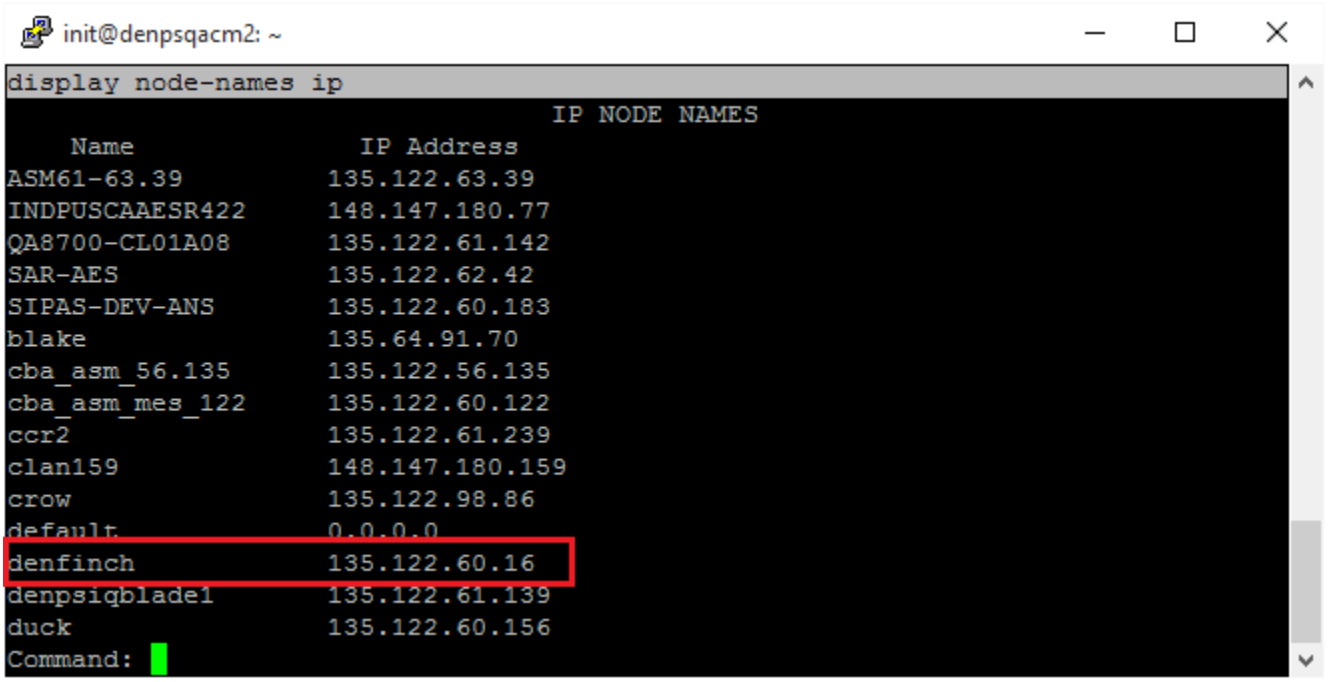
After we have logged into SAT, we run the following command: display communication-interface processor-channels

We should see the CMS configured as one of the processor channels in the page. The following picture illustrates what we should see: a row (number 1, outlined in red) that tells us the CMS - by its hostname, ***denfinch*** - is consuming the channel 5001, to receive data from CM and generate reports. So this tells us, the CMS has a channel configured, and this channel is properly set.



The other setting we may want to verify is the node name.To do it, in SAT, we type F1 to cancel the current operation and run the command ***display node-names ip***.

It is important to ensure the CMS is correctly mapped both in terms of IP and hostname. In the following picture, we see the hostname is correctly mapped to the CMS we are planning to use, so everything is configured as expected.



Since everything is fine, we can leave SAT. To do so, in the command prompt, just run the command ***logoff***.

Diagnosing Issues on the CMS

There are some settings in the CMS that might cause communication problems with the CMS, CMS Connector or both. Through this section, we are going to cover each one.

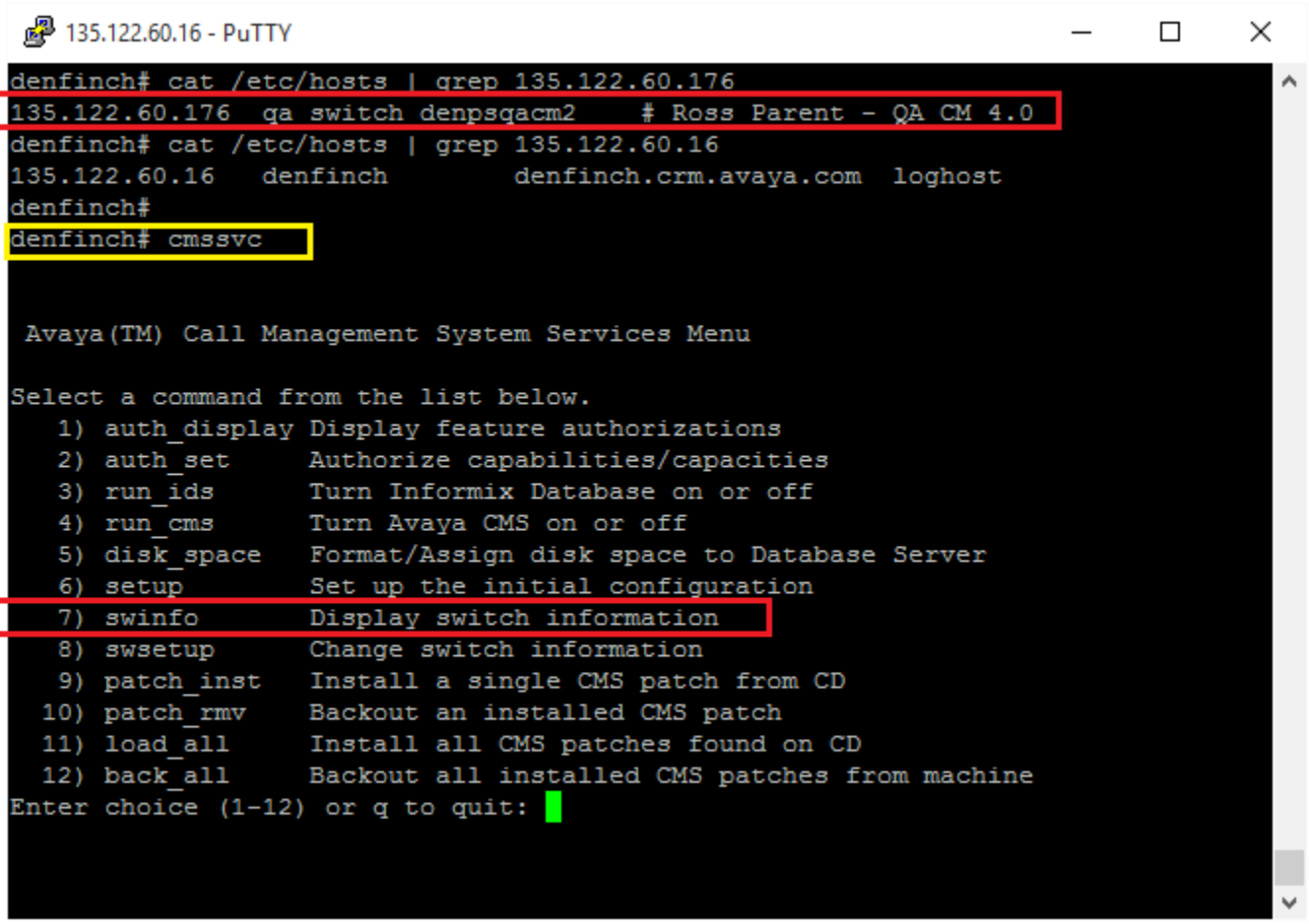
CM Related Settings

We need to verify the CM settings in the CMS. Each CMS can map up to 8 ACD boxes and all these ACDs have to be correctly configured. If, for any reason, an IP address and/or hostname to any CM is wrong (*typo, IP address of another CM instead of the one expected, etc.*), there might be either communication problems or the report we see may not correspond to the one we expect (*wrong or inexistent skills, etc.)*.

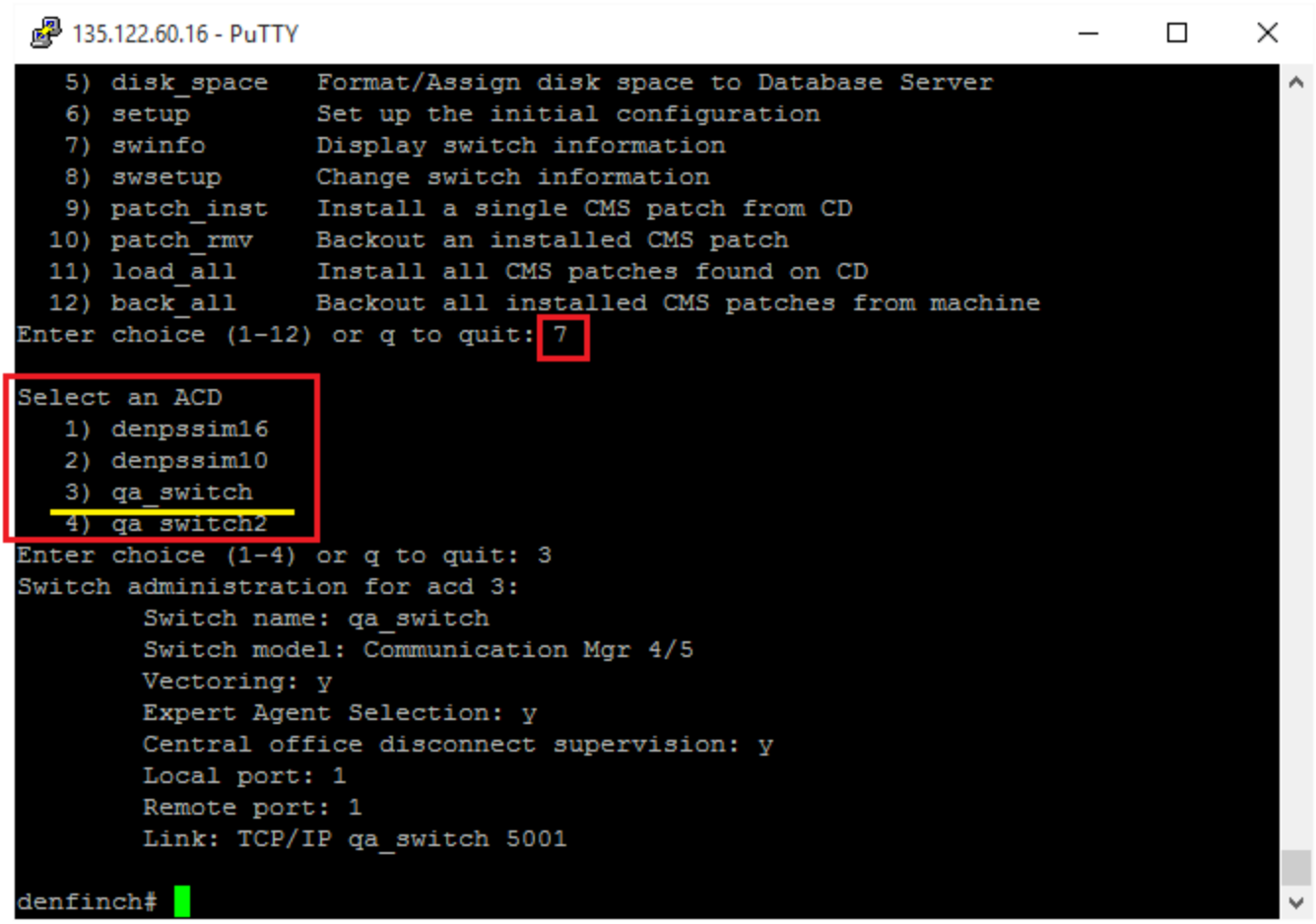
To verify the CM instances configured in the CMS, we have to:

1. Using a tool such as PuTTy, log into the CMS server
2. Verify if the ACD (CM) IP address is correctly set in the hosts file (*this configuration is in the /etc/hosts file*)
3. Run the cmssvc application
4. Select option 7, to display the switch information (press ***ENTER*** to confirm the option)

Step 2, 3, 4, shown below:



At this point, we are sure the CM is correctly mapped in the hosts file and we are already looking at the switch information page in the ***cmssvc*** ap plication. From the picture above, we verified that the hostnames used to identify the CM machine with IP address 135.122.60.176 are ***qa_switch*** and ***denpsqacm2***. The former name is displayed in the list of managed ACDs in this CMS, as we can see in the picture below, it is CM ACD ID #3:



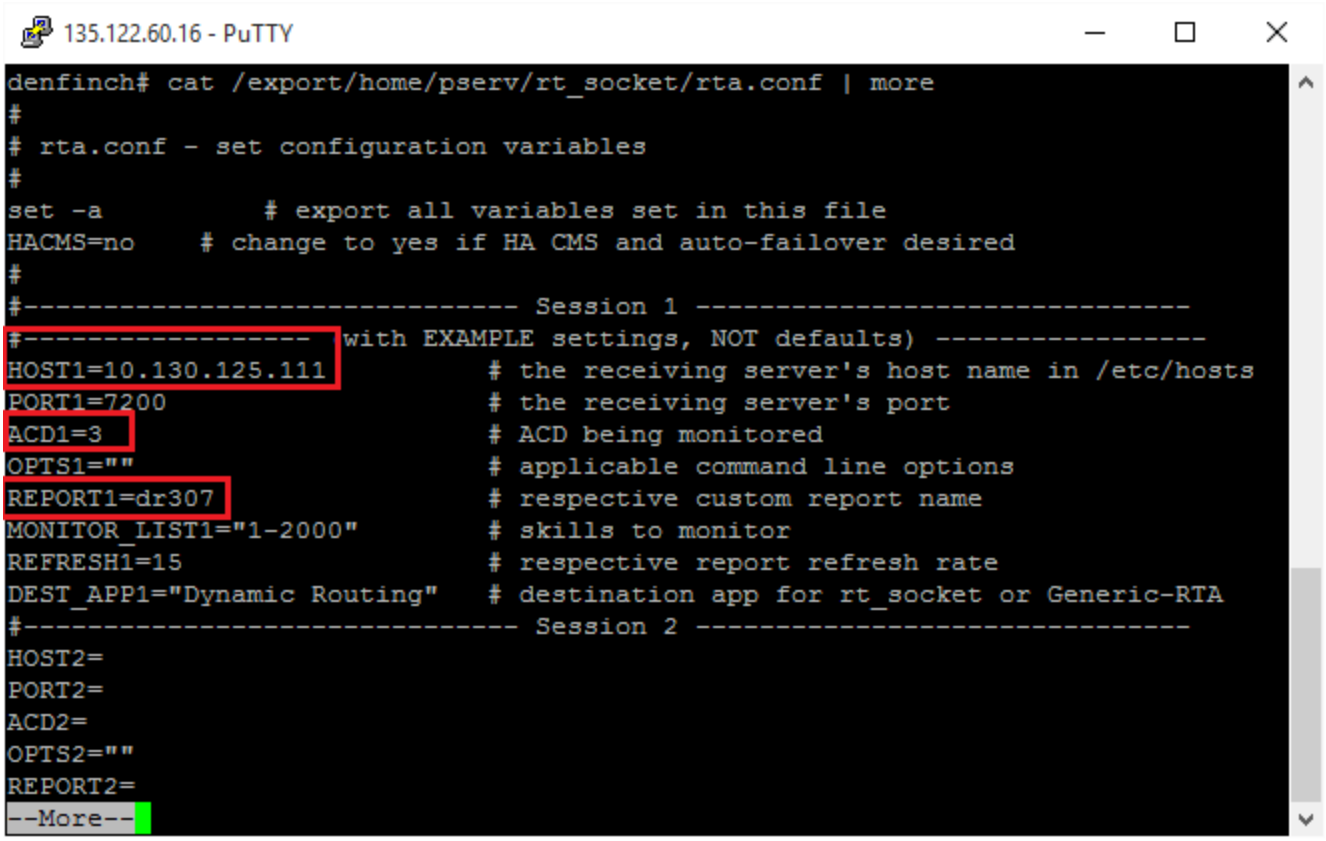
So, if the IP address is in the hosts file and the hostname used in the CMS ACD settings match and correspond to the CM we are planning to use, we have verified correct CM configuration on the CMS.

RT-Socket Channel Setup

A real-time socket is used to communicate CM metrics from the CMS to its clients. CMS Connector is one of its clients, and is part of Dynamic Routing.

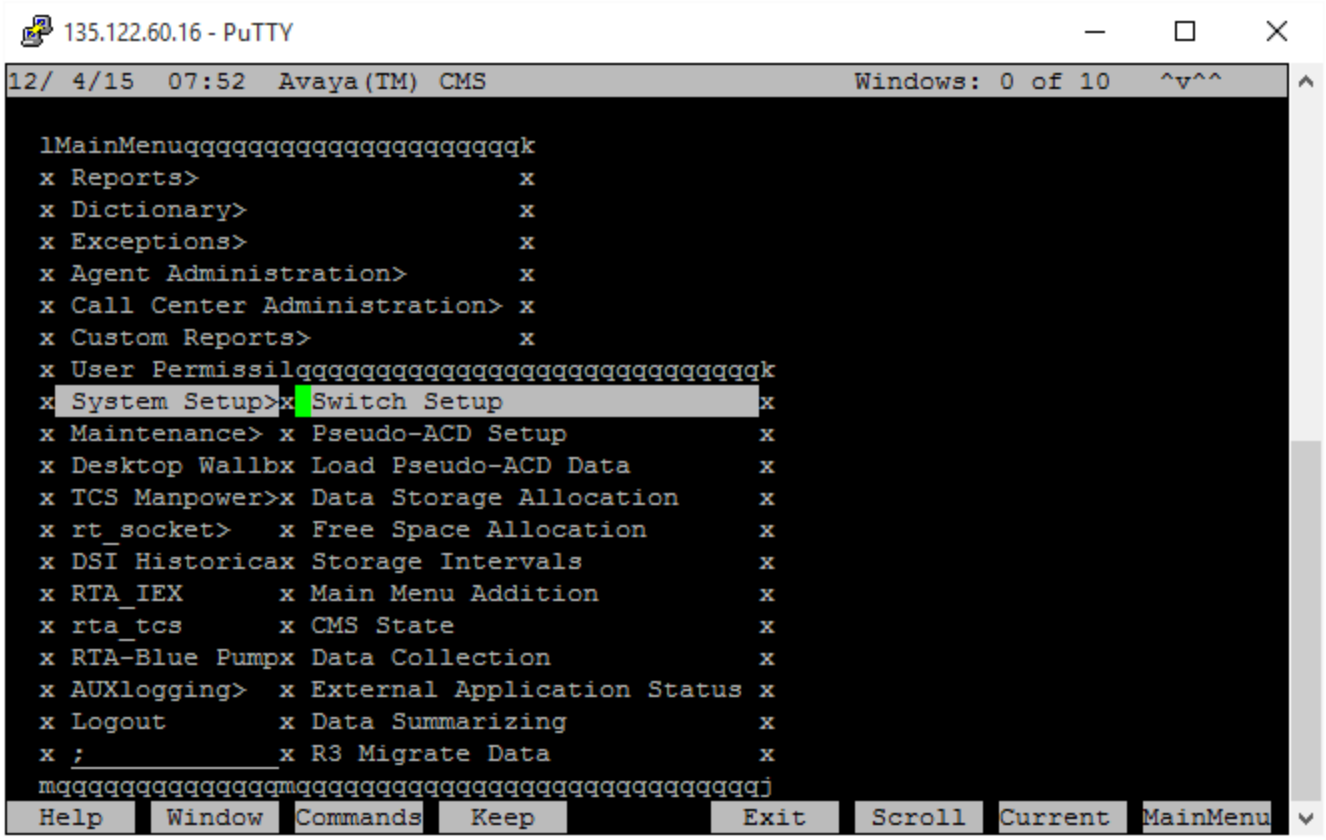
The RT-Socket sessions are all configured in the file /export/home/pserv/rt_socket/rta.conf. The following picture contains the information for session 1, which points to the CMS Connector we are using. The more relevant fields, as well as their respective meaning and context, are explained in the following table:

Field	Comments
HOST	The IP address of the CMS Connector
PORT	The CMS Connector port to send the metrics report to.
ACD	The ACD identifier in the CMS (this information is available in the CMS, it is an ID from the CMS perspective **)
REPORT	The report used to generate the metrics data to be transmitted
REFRESH	The interval used to refresh the metrics data

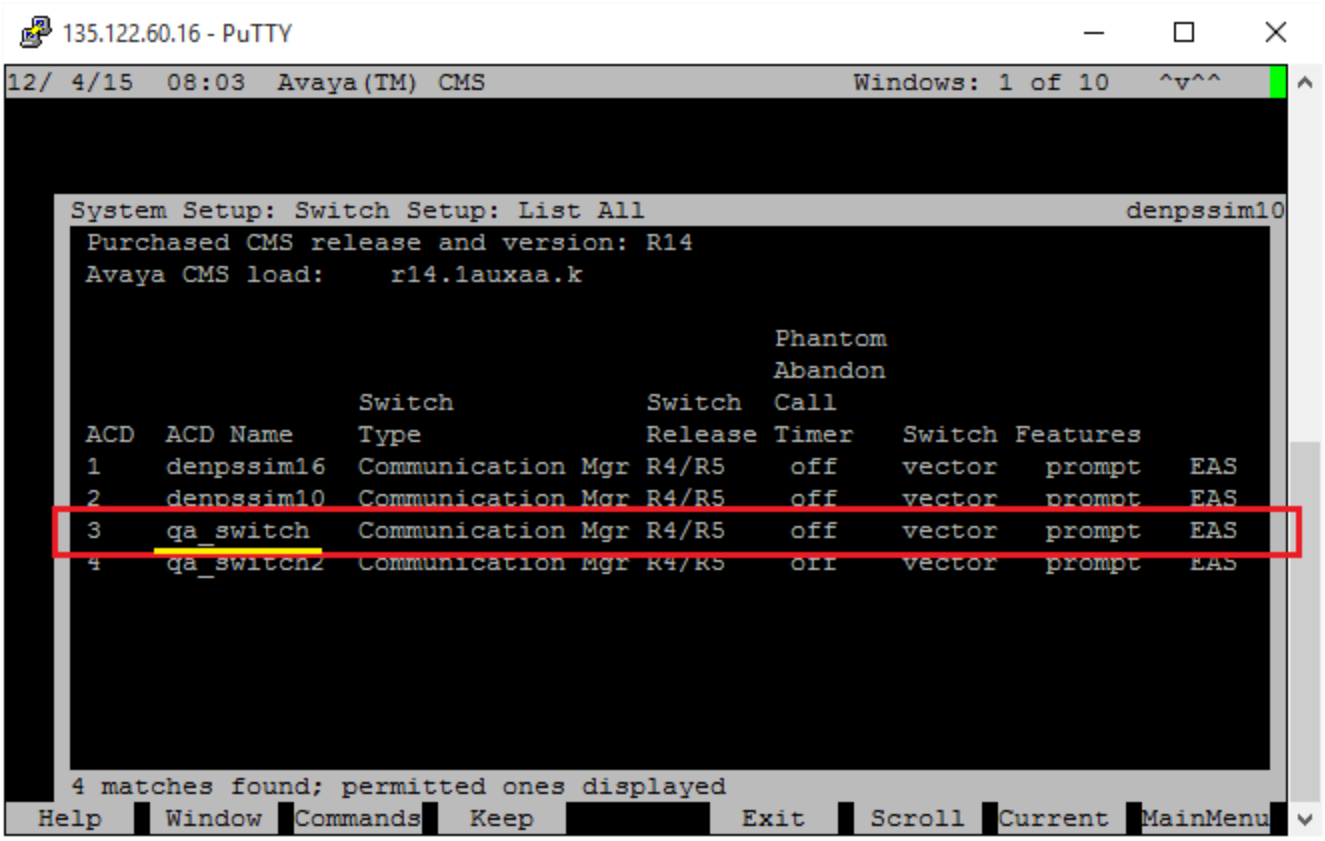


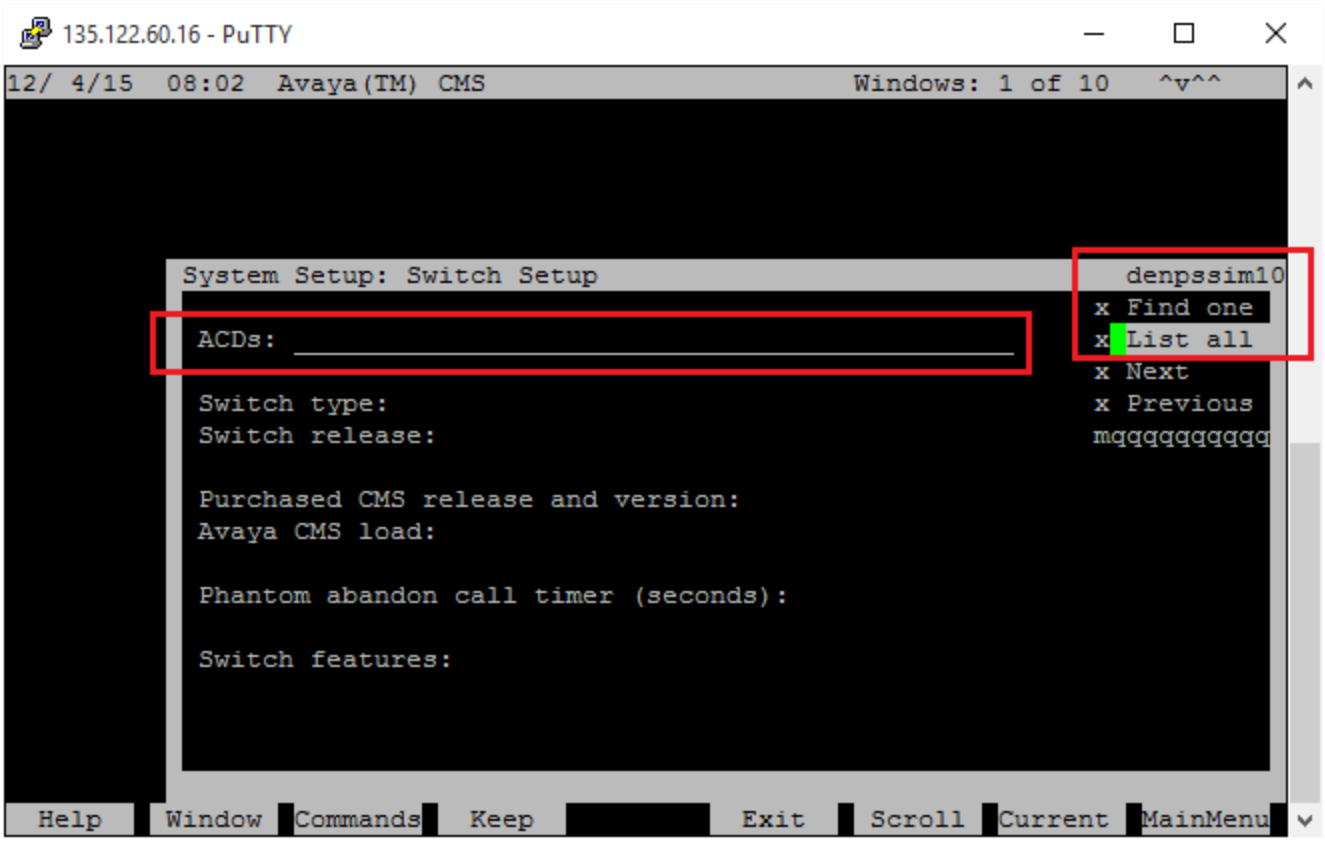
We see in the table above, that the CM ACD ID is something we can figure out by using a CMS application. When we used the ***cmssvc*** applicatio n to check the switch information, we already saw the CM we are planning to use, is listed as CM ACD ID #3. If we would like to confirm the CM ACD ID in the CMS settings, we could also use the ***CMS ASC II*** tool, and the way to go into it is described below.

1. Using a tool such as PuTTy, log into the CMS server
2. Run the following command: ***su - cms cms***
3. We will eventually be informed about backup states, but ignore all of them and press ***ENTER*** to continue. Then, the ***CMS ASC II*** applicati on should be displayed, as in the following picture:



In the CMS ASC II main menu, using the Down arrow key, go to ***System Setup*** sub-menu and, then, use the Right arrow key to expand it. Then hit ***ENTER*** to select the option ***Switch Setup***. The next page that is displayed asks for the ACD number we want to see details about. If we know it, just enter the number and press ENTER, then select '*Find One*' option and hit ***ENTER*** again. If we don't know the number, we can just hit ***ENTER***, then select '*List All*' in the pop-up menu that appears. We can see all these things in the picture below:

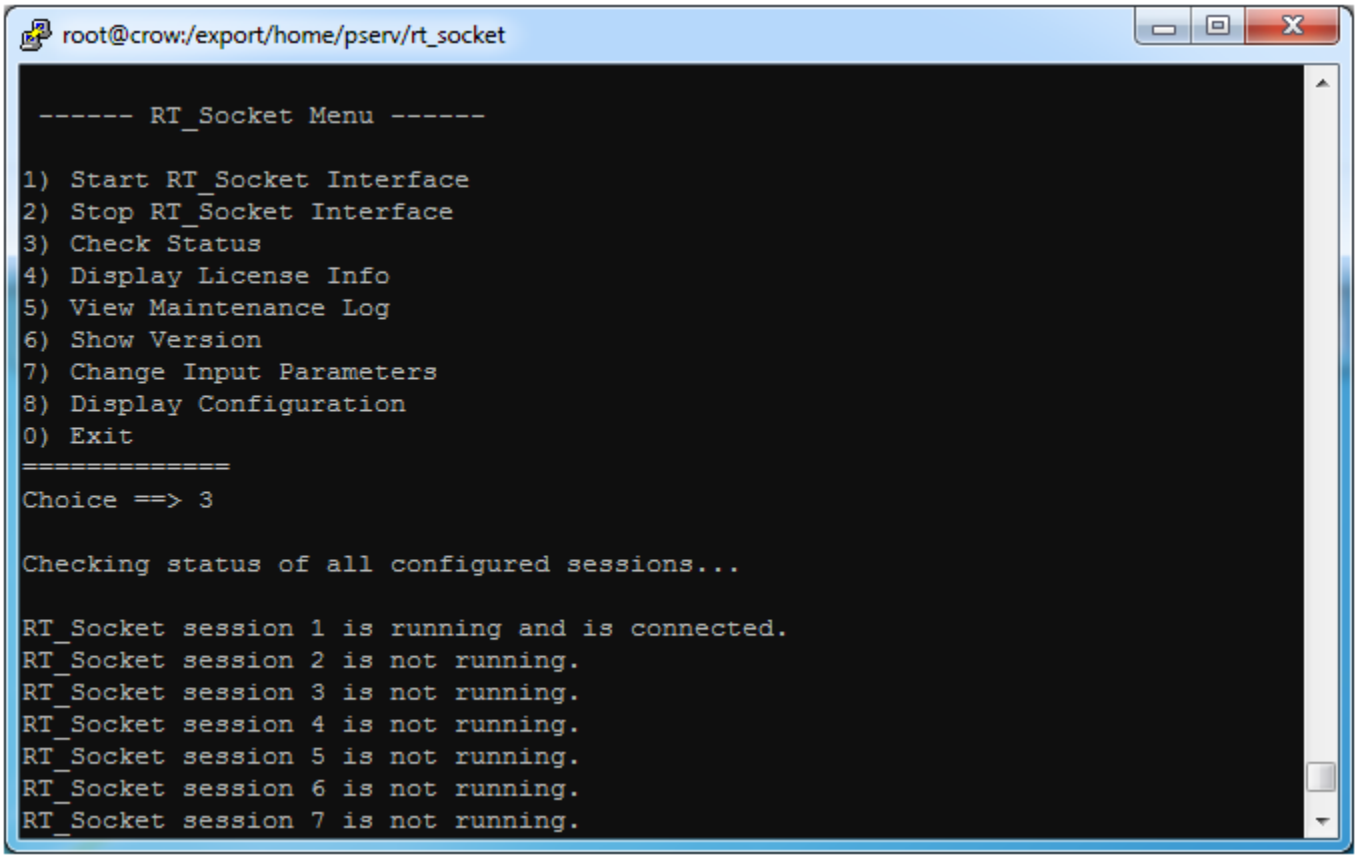




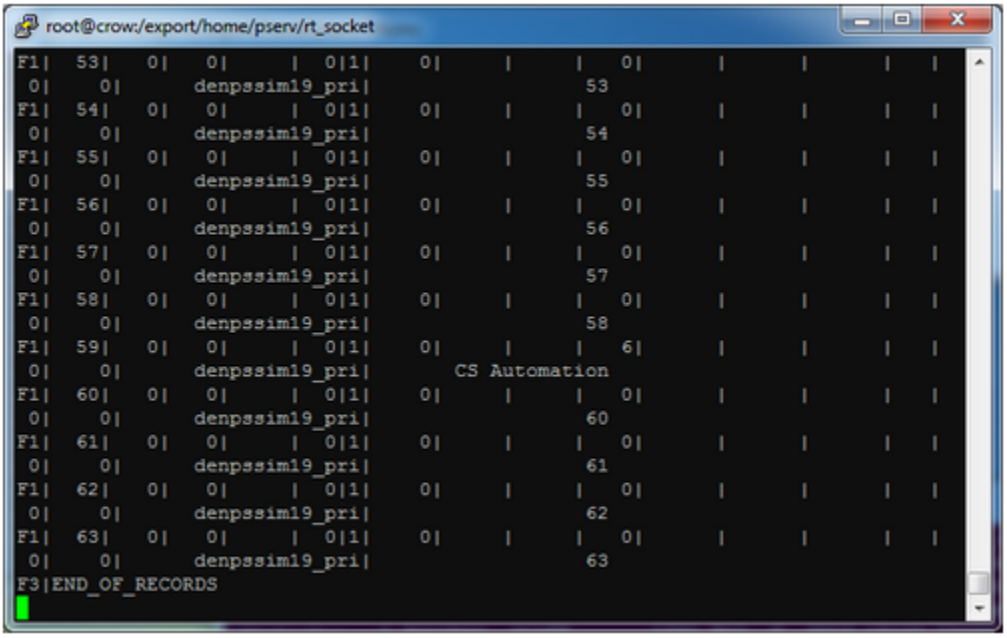
In this example, '*List all*' was selected and, in the output illustrated in the next picture, we can see that the ACD ID to the CM we are using in this tutorial is **3.**

RT-Socket Enablement

Another verification we have to make is the status of the real-time socket. The session we have configured has to be alive, up and running; otherwise there will be no communication between the endpoints (CMS, CMS clients). Operation on the RT-Socket sessions can be performed using the */export/home/pserv/rt_socket/menurta* application. We run this command in the shell and are able to see a screen like the following:

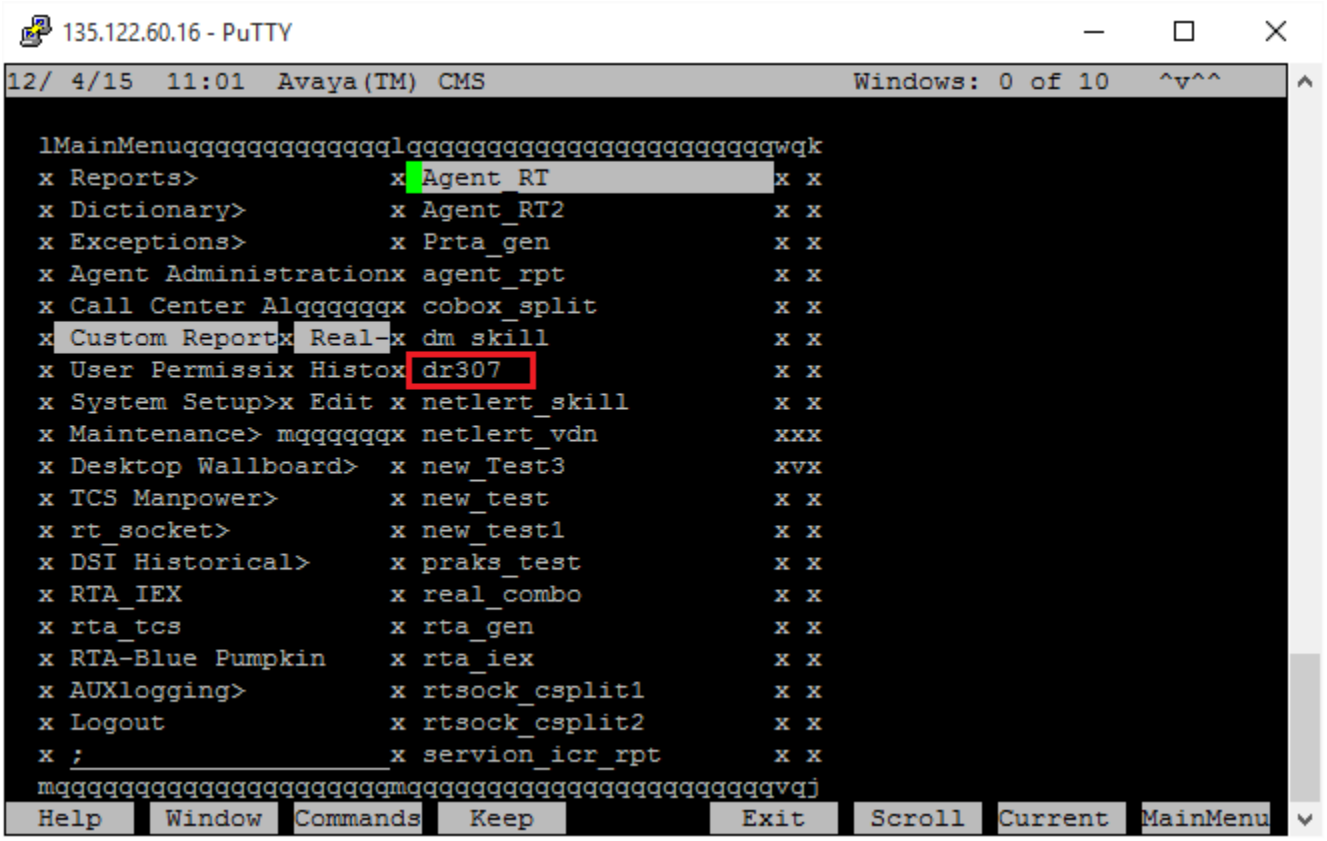


To (re)start a session, we have to select Option 1 and hit ***ENTER***. If the session is already running, we will be prompted to stop it and then it will be restarted.
Select Option 3 for the status. Once the session is running, this is what the output looks like:

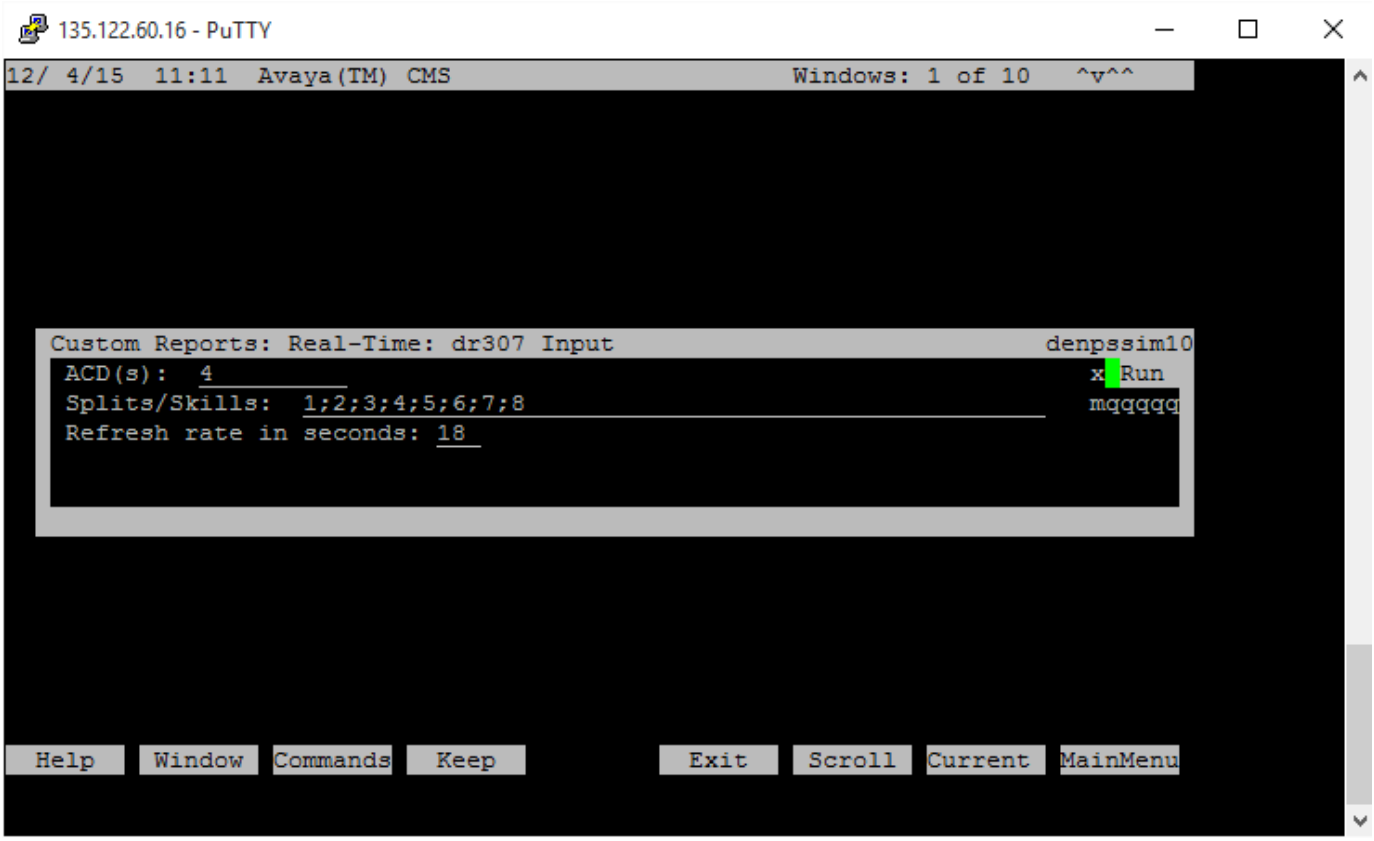


Verifying Custom Reports on CMS ASC II

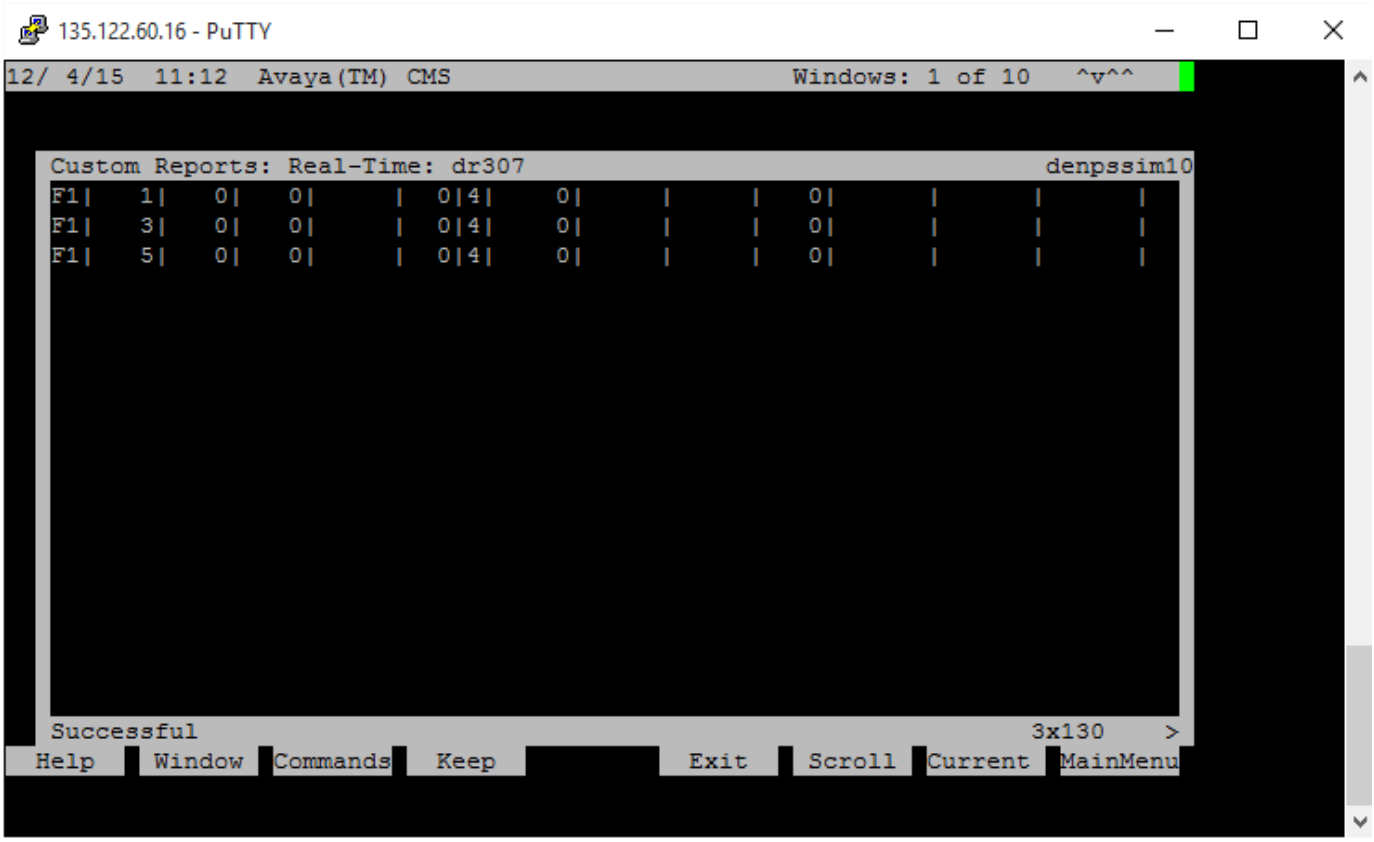
The CMS ASC II application can be used to verify the reports and, if we want, test them in the terminal/shell. To do so, we need to first find the report we want to run, by going to the CMS ASC II main menu and selecting ***Custom Reports*** > ***Real-Time*** | ***Historical*** and finally selecting the desired one from the list.In this example, for ***Dynamic Routing***, the report we would want to verify is real-time and called ***dr304***.



The next step is to inform the ***ACD ID***, which is **4** in this example. If we want, we can also provide the particular skills we want to look at, just in case we want to filter information in the report that will be displayed.



Once we hit ***ENTER*** twice (first one confirms the input, second one executes the ***Run*** command), a report is displayed, and refreshed at the interval defined by ust:



Very Important Information regarding the CM/CMS reporting mechanism

1. CM and CMS only communicate ACTIVE information. If there is no agent logged into a skill, no information will be reported for the skill (we don't even see metrics in the feed). This is expected behavior.
2. EWT is calculated ONLY WHEN a call is at least in queue.
3. Once an agent answers a call - and only then - the EWT starts to change.

PuTTy and Function Keys
Since PuTTy uses xterm as the terminal type, we may be unable to use function keys directly in the shell. But there's a workaround for it: press ***Ctrl + P*** and it will put us in the 'function' mode. Then, instead of the ***Fn*** button, just type in the number corresponding to the function. Example: if we would like to close a window, instead of using F5, just use ***Ctrl + P*** and, then, type in the number ***5***.

Some useful function keys are, in the context of the CMS ASC II:

Key	Operation
F1	Help
F3	Options
F5	Close window

Diagnosing Issues on the CMS Connector (CMSC)

CMS Connector Logfile

Edit the file "cms-connector-logging.xml" located in the *<DR_HOME>/cms-connector/config/logging* directory and change the log level from INFO to DEBUG.

Look for the following entry:

```
<logger name="com.avaya.ept.dr.services.connectors" level="INFO" additivity="false">
    <appender-ref ref="CMS_CONNECTOR" />
</logger>
```

It should be changed to:

```
<logger name="com.avaya.ept.dr.services.connectors" level="DEBUG" additivity="false">
    <appender-ref ref="CMS_CONNECTOR" />
</logger>
```

Let the system refresh the configuration by waiting at least 30 seconds and then look for the file "cms_connector.log" located inside *<DR_HOME>/cms-connector/logs/application* directory.

If you see actual/live data from your CMS skill reports, it means the feed from the CMS is being received and processed.

On the other hand, if you only see Simulated metrics (for Demo Data configuration) in the log file, it means something is not correct in the configuration.

Possible issues at this point are:

- the "cmsReportConfiguration.xml" file doesn't have the ACDs mapped to DR3 ACD entity IDs
- the IP Address/Port of the CMS is not configured in the "cmsConfiguration.xml file".

Configuring syslog using logback.xml

logback xml contains a declaration for a logger, which writes to syslog.

As of Dynamic Routing 3.2, connection errors that cause a metrics space to be unreachable (and so metrics are lost) are logged into syslog. For

these cases, by default, the logger logs into the local syslog at port 514 (the default port).

If another syslog should be used, this can be achieved by changing 2 properties in the logback.xml file:

```
<property scope="system" name="SYSLOG_HOST" value="localhost" />
<property scope="system" name="SYSLOG_PORT" value="514" />
```

The logs will appear in the alternate syslog, with the tag DR, e.g.

```
Jan 18 17:26:37 localhost DR[17954]: -03:00 2017 558 1 com.avaya.ept.dynamicrouting | 0
|@2017-01-18/17:26:37.558|locatorsUpdaterScheduler-1|ERROR|Cannot create the connection using locator 127.0.0.1
```

CMS-Connector Connection Status Errors

In order to check for errors with the CMS-Connector connection to Dynamic Routing severs, look at the log file located in: *<DR_HOME>/cms-connector/logs/gs*. The name of the logs varies, but follows the pattern */<SERVER_IP/service-${date}.log*

This is a typical message when the CMS-Connector has lost connection:

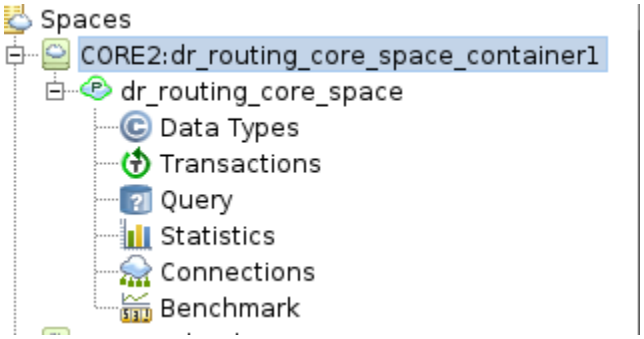
```
Failed to connect to LUS on 10.130.124.33:4174, retry in 5014ms; Caused by: java.net.ConnectException: Connection refused
```

Diagnosing Issues in Dynamic Routing

This section explains how to check Agent Group metrics at the DR3 metrics data grid level using Gigaspaces UI tool.

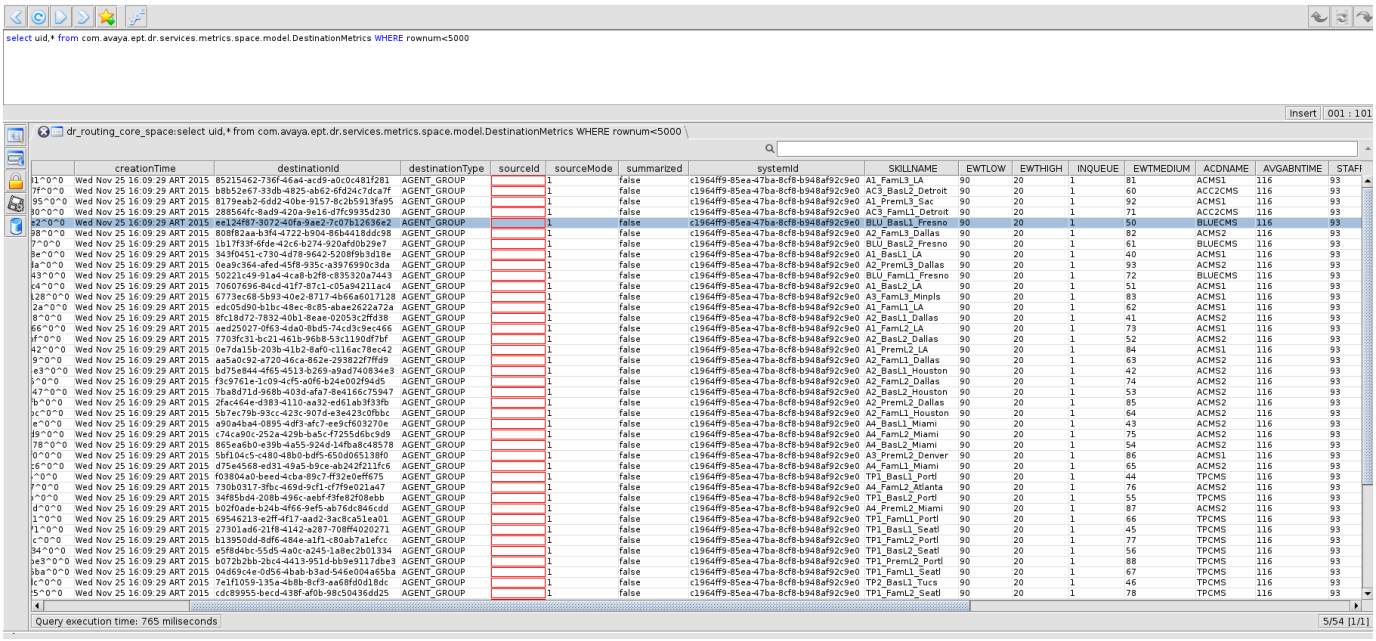
> **IMPORTANT**
> This content is targeted at Avaya Support personnel only. It assumes previous background on Gigaspaces, and a valid GS distribution running on a non-DR3 server with desktop interface.

Launch "XAP Management Center" (a.k.a. GS UI) using "gs-ui.sh/.bat" script, connect to DR3 cluster using Lookup Locators, navigate to the Space Browser and then select the "Routing Core" space.

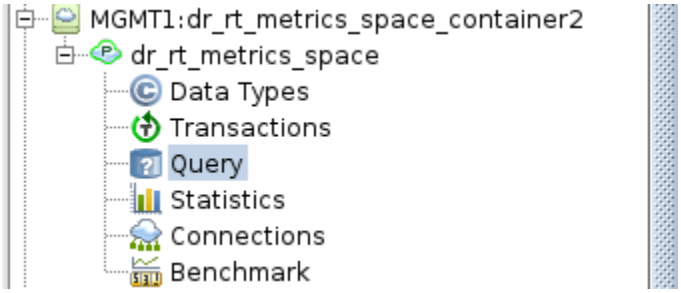


Execute the following query: select uid,* from com.avaya.ept.dr.services.metrics.space.model.DestinationMetrics WHERE rownum<5000

(Optional) You could correlate the "destinationId" on each DestinationMetrics object with the IDs in the AgentGroup table at DR3 configuration DB. However, this is not needed because SKILLNAME field should provide human-friendly identification for each Skill/AgentGroupsh



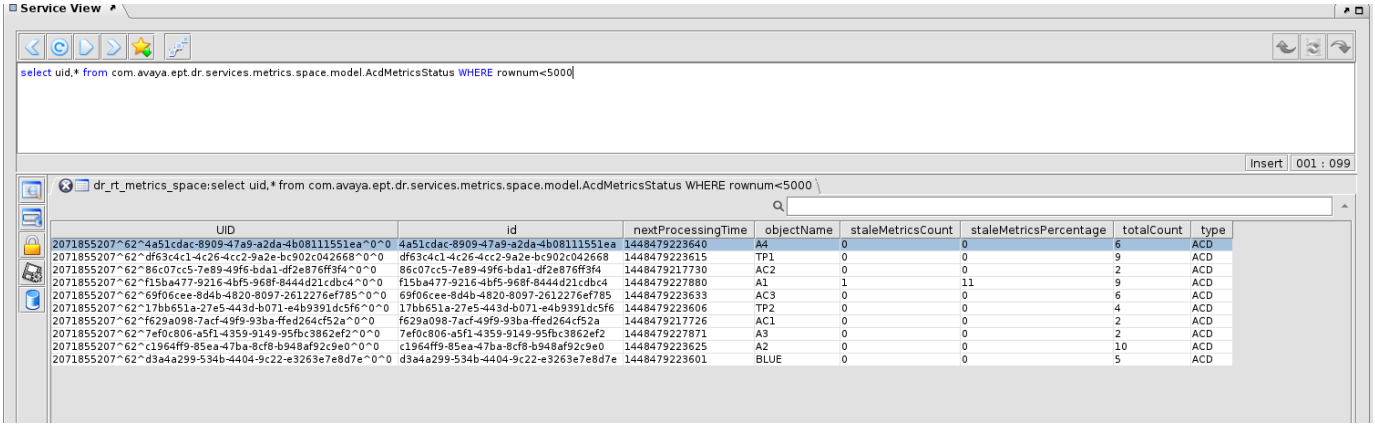
Verify the ACD metrics status in GigaSpaces from the "RT Metrics" space. Notice: "RT Metrics" does not contain data from RT socket, it only contains the data for ACD Status Semaphore display purposes.



select uid,* from com.avaya.ept.dr.services.metrics.space.model.AcdMetricsStatus WHERE rownum<5000

In the following example, the totalCount column should match the number of AgentGroups associated with the ACD.

In order to have ACD Semaphores Green with 0% stale, staleMetricsCount and staleMetricsPercentage must be 0. In the following example, there's 1 Agent Group causing problems, so we have 11% stale data (staleMetricsCount / totalCount == 1/9 == 0.11)



DR 3.2 ICR Integration

- Terminology
- Introduction
- Overall Architecture
    - Managing ICR Skills in AAEPM
- Next step: Installing and Configuring the Extension

Terminology

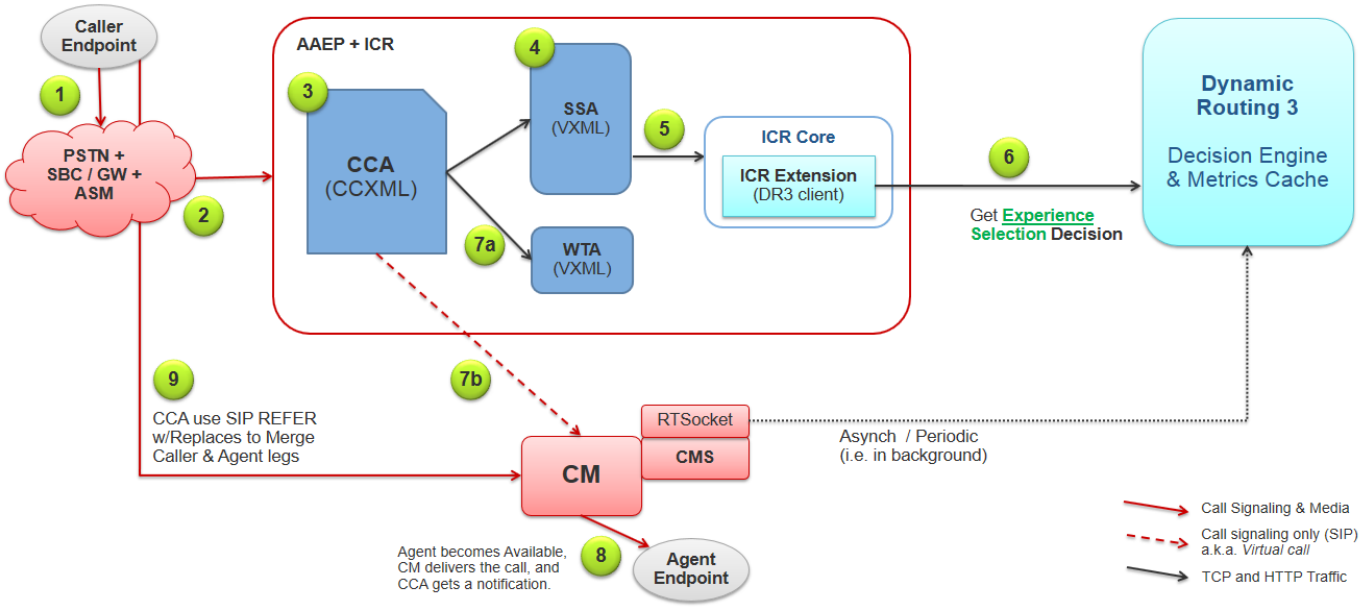
Term	Meaning
AAEP	Avaya Aura Experience Portal
AAEPM	Avaya Aura Experience Portal Management
AAEPMS	Avaya Aura Experience Portal Management System
ANI	Automatic Number Identification
ASR	Automatic Speech Recognition
BSR	Best Service Routing
DNIS	Dialed Number Identification Service
DSS	Dynamic Self Service
EHA	Error Handling Application
ICR	Intelligent Customer Routing
ICR-CCA	Intelligent Customer Routing - Call Control Application
MPP	Media Processing Platform
SSA	Self-Service Application
TTS	Text To Speech
WTA	Wait Treatment Application
OD	Orchestration Designer
DR3	Dynamic Routing
Extension module	Module developed to integrate ICR Core and Dynamic Routing (Dynamic Routing serves ICR Core with its experience selection capabilities)
CXI	CCXML Interpreter

Introduction

The Dynamic Routing ICR Core extension module (a.k.a. *extension*) is a component that allows ICR Core to use Dynamic Routing as a decision engine. It is packed as a JAR file and deployed as an ICR Core library, as described later in this page. Whenever it is installed and enabled in an ICR Core runtime, ICR Core's default routing mechanisms are ignored and Dynamic Routing is consumed through a RESTful web service to make decisions CCA uses to take a step in a call flow, such as a call transfer.

Overall Architecture

The following picture illustrates the typical architecture in which Dynamic Routing and ICR Core can be combined, through the use of this extension.



The call flows in this setup can be summarised as follows:

Step	Description
1	A call arrives to the system. It may come over the public telephony network (PSTN) or over TCP/IP, and is first handled by s session border controller (SBC). We may see a Gateway in the middle, responsible mostly for the message translation between different protocols. The AASM is aware of all the elements in the architecture, as well as all the routing policies to coordinate the call flow.
2	Simplifying a bit the next steps, the call arrives at the AAEP and, more specifically, at the ICR CCA.
3	This call control application, ICR CCA, is interpreted and executed by MPP's CCXML interpreter. This ICR CCA is configured in AAEPMS, and it can manage many applications such as a SSA, WTAs and EHAs. These manageable applications are all VXML, executed by MPP's voice browser, and the first one to be executed from the ICR CCA is the SSA.
4	The SSA starts executing. The SSA talks to ICR Core through the PDC.
5	The SSA invokes the extension module through a component called PDC. Internally, when the request arrives, ICR Core identifies the extension's entry point and forwards the request to it.
6	This entry point mentioned in the previous step is where a call to Dynamic Routing is made. Both the input and output of this communication are described in a next section. Dynamic Routing, then, determines the best experience for the call and sends it back to the extension, which, in turn, is responsible for preparing the information to be sent to the CCA.
7	This is composed of two actions happening in parallel: **Action** / **Description** WTA is started — When CCA has the destination, as well as all other attributes that compose the experience determined by Dynamic Routing, it starts a WTA that interacts with the caller while an agent is not yet ready to answer the call. Virtual call is placed — A virtual call is placed on the AACM, while control of the call flow is still with AAEP. This call is then queued to a skill and AACM works on managing the agents associated with that skill to get the call answered.
8	Once an agent is available to answer the call, WTA is completed and the agent is connected.
9	Caller and agent are merged and start interacting with each other

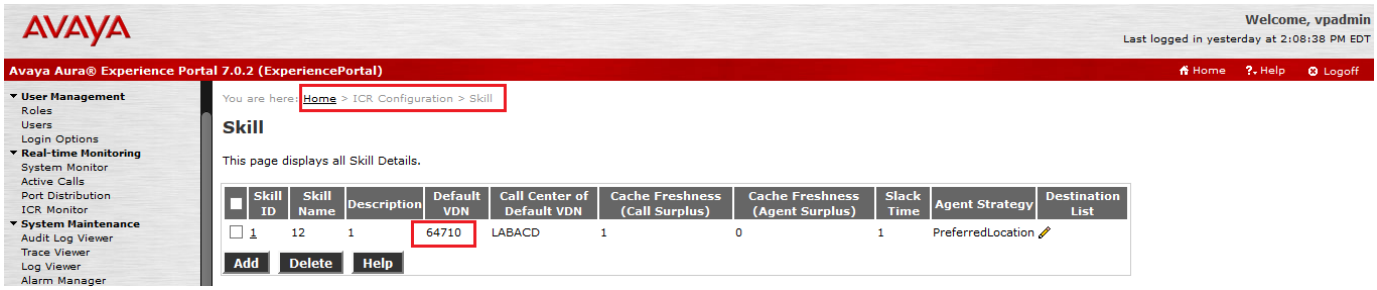
Managing ICR Skills in AAEPM

Invoking an ICR Core extension from an SSA requires a parameter called **ICR skill number**. This is not only a mandatory field, but it must be

assigned to a skill number that exists in the AAEP configuration. However, it is important to know this parameter is **meaningless** to Dynamic Routing.

ICR skills are configured in AAEPM, by selecting ICR Configuration » Skill in the left menu. Each skill is composed by a unique ID (the one used as the number when invoking the extension), a name, a default VDN, the default VDN's call center and a list of destinations. The skill involved in each ICR Core invocation is sent in the route request.

In the example below, we see an ICR skill whose identifier is **1**.



> **ICR Skill Number: useless to DR3, but necessary**
> Dynamic Routing does **NOT** use ICR Skill numbers to make its decisions. To Dynamic Routing, the meaningful information encapsulated by the ICR Core request comes in an object known as **caller context**. It is in this object that fields such as the decision function name, segmentation attributes and custom parameters are defined.
>
> Although it is meaningless to Dynamic Routing, ICR skill numbers specified in an invocation **MUST** correspond to existing ICR skills. If not, ICR Core throws an exception early in the flow, and the extension is not called. So, in order to make this integration work, **at least one ICR Skill must be configured in AAEPMS, although this information is ignored by the extension.**

From a functional perspective, this ICR Skill can be used as a **default Destination**. Whenever ICR SSA is unable to communicate with Dynamic Routing, the GENERIC_ERROR result will alert CCA of the situation. In these cases, the CCA will request a default destination from ICR Core, based in the ICR Skill Id used previously by SSA.

Next step: Installing and Configuring the Extension

The next page describes how the extension is installed and configured.

1. Installing and Configuring the Extension

- Extension Operation
  - Sending Segmentation Attributes to the Extension
  - Sending Custom Parameters to the Extension
  - Dynamic Routing Response Handling
  - Installing the Extension
  - Defining the Preferred ICR Core
  - Overall Flow
- Next step: Creating an SSA to Invoke the Extension

Extension Operation

Dynamic Routing ICR Core extension allows ICR Core to use Dynamic Routing as its decision engine. It has a single entry point, **ICRExtension**, responsible for the following actions:

1. Collect the parameters sent in the request, encapsulated by an object known as **caller context**;
2. Compose a Dynamic Routing request based on this input;
3. Invoke Dynamic Routing to make a decision to be used by the ICR CCA to take the next step in the call flow;
4. Parse the Dynamic Routing response and create a DestinationInformation object, to be sent to the application that invoked ICR Core - usually a SSA;

The table below illustrates the parameters that may be sent in the **caller context** object, and how they are manipulated by the extension:

Caller context key/value pair	Dynamic Routing request population	Mandatory Input from calling application?
customParams	The text is expected to come in the following format: customParam**1**=value**1**;...;customParam**N**=value**N** It is parsed, decomposed into key/value pairs and these pairs are added to the custom parameters map sent in the Dynamic Routing request.	NO
decisionFunction	The value is used to set the decision function in the Dynamic Routing request	YES
segAttributes	The text is expected to come in the following format: segAttr**1**=value**1**;...;segAttr**N**=value**N** It is parsed, decomposed into key/value pairs and these pairs are added to the segmentation attributes map sent in the Dynamic Routing request.	NO
requeue	Indicates whether the current request is related to a requeue performed by the ICR CCA. This entry is sent to Dynamic Routing through the custom paramaters map. Acceptable values to this field are **true** (whenever the call comes from a requeue scenario) and **false** (first call to the extension from a calling application, such as a SSA).	YES
ani	This value is used as the 'From' attribute in the Dynamic Routing request	YES
ucid	This value is used as the 'Interaction ID' attribute in the Dynamic Routing request	YES
dnis	This value is used as the 'To' attribute in the Dynamic Routing request	YES
appName	Identifies the calling application. This entry is sent to Dynamic Routing through the custom parameters map	NO
previousICRExecInfo	This entry is used to hold the Dynamic Routing extended response of the previous execution, in requeue scenarios. The extension caches the Dynamic Routing extended response of the latest execution and assigns this content to this entry, to make the history of the previous execution available to Dynamic Routing next time it is invoked. **The field is fed by DR3!** It is important to understand that Dynamic Routing is responsible for the content assigned to this field. The more information added to the extended response, each time Dynamic Routing is invoked, the more information available to Dynamic Routing if requeue happens and it is invoked again by the extension.	NO

All the parameters described above are identified by well known and constant, pre-defined keys. However, the extension is also able to handle other parameters set in the SSA level, as described in the following sub-sections.

Sending Segmentation Attributes to the Extension

A client like an SSA may set individual segmentation attributes as entries in the caller context. To let the extension recognize these entries as such, they **MUST START with the preffix drSeg**, followed by the actual name of the segmentation dimension. The extension takes this key and extracts the attribute name by removing the preffix.

Example: if a segmentation table has a dimension called **Package**, this attribute can be sent in the caller context identified by **drSegPackage**.

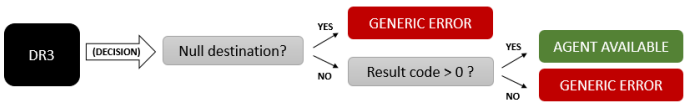
Sending Custom Parameters to the Extension

The extension considers as a custom parameter ANY entry, in the caller context, that doesn't match the well known attributes described in the table above and the segmentation attributes just described in the previous section (the ones identified by the preffix **drSeg**).

Dynamic Routing Response Handling

Dynamic Routing response is in JSON format and is used by the extension to compose the DestinationDetail object to be sent back to MPP.

The table below summarises how the mapping between Dynamic Routing's response and DestinationDetail is made.

DestinationDetail field	Value assigned to the field
refer	The value assigned to this attribute comes from the Dynamic Routing extended response entry identified by the key '*refer*' . Possible values are **true** and **false**.
result	The result is determined by the extension, depending on the Dynamic Routing response. The two possible values are **GENERIC_ERROR** and **AGENT AVAILABLE**, and the following picture illustrates the logic used by the extension to determine the actual value. 
destination	The destination identified by the Dynamic Routing. The complete destination address is composed by a VDN and a property that must be added to the extended response, called **sip.domain**. Without the SIP domain as part of the destination, **the transfer is very likely to fail**. So, make sure Dynamic Routing provides the SIP domain of the ACD system needed.
customData	The full Dynamic Routing extended response is set as the DestinationDetail's custom data, and used, later, as historical data whenever a requeue happens.

Installing the Extension

The connector is part of the Dynamic Routing installation package (ISO). The package can be found at *<MOUNT_POINT>/ICRExtension/ folder*.

To install the extension in an ICR Core server, first make sure your ICR Core is at least **version 7.0, Feature Pack 1** (7.0.1). Any previous version does not support extensions. You can find this information by looking at the file ***/opt/Avaya/ICR/ICR_InstallLog.log***, in the server the ICR Core is installed. After this verification, proceed with the following actions:

1. Log into the ICR Core server, as super user
   - Connect to the ICR Core server with non-root credentials;
   - Run ***su -***
   - Provide the root password
2. Upload the bundle to the server, in any directory of your preference (example: ***/tmp***)
3. Unpack the bundle (***tar -zxvf <CONNECTOR>.tar.gz***).

4. Move to the directory ***icrcore-dr3-extension***
5. Make sure the script is executable (***chmod +x install_extension.sh***)
6. Run the script ***install_extension.sh*** in the command line (***./install_extension.sh***)

Problems running the script?
In case you run into problems running the script, run the following command:

```
sed -i -e 's/\r$//' install_extension.sh
```

After the command completes, run the script again.

The installer is responsible for the following steps:

a. moving the extension and its dependencies to the expected folder (***/opt/Avaya/ICR/ICRCore/tomcat/webapps/icrcore/WEB-INF/lib***)
b. moving the extension's properties file (***dr3_icrext.properties***) to the expected directory (***/opt/Avaya/ICR/ICRCore/tomcat/webapps/icrcore/WEB-INF/classes***)
c. declaring the extension's entry point in the ICR Core application context file (***/opt/Avaya/ICR/ICRCore/tomcat/webapps/icrcore/WEB-INF/applicationContext.xml***)
d. restarting the icrcore service to apply the changes and load the extension

The block below shows the output of the installation:

Installation outcome

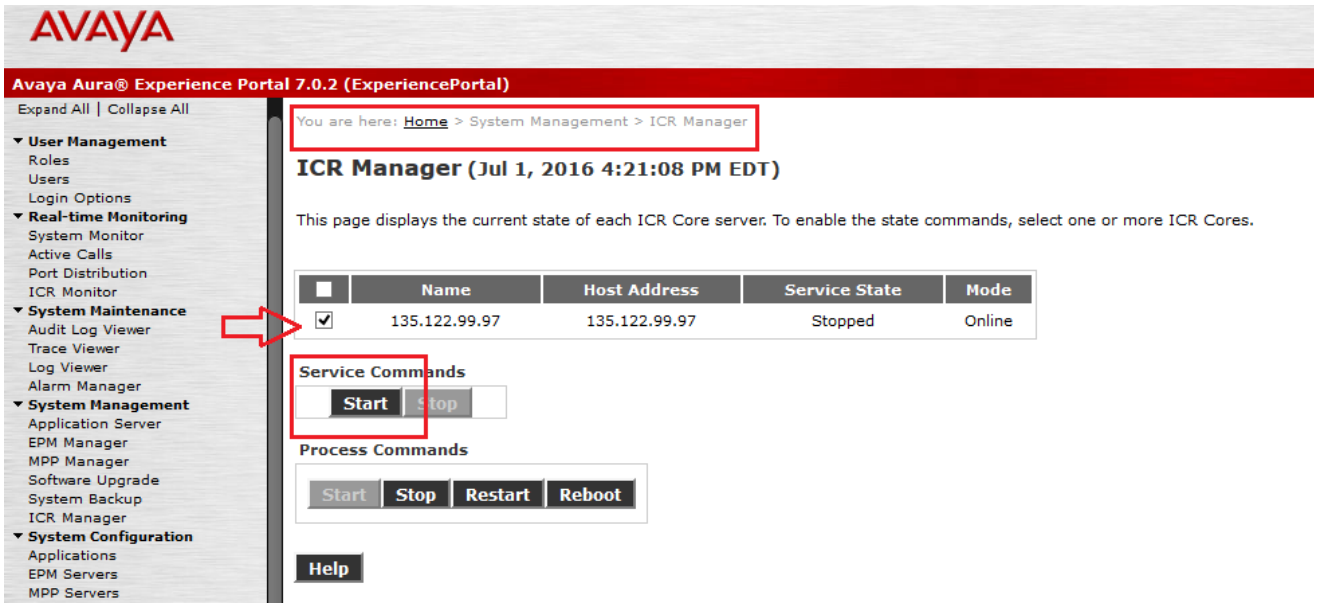
```
[root@aaepicr2 icrcore-dr3-ext]# ./install_extension.sh
-----------------------------------------------------
ICR Core DR3 Extension Installation
-----------------------------------------------------
Stopping ICRCore service at Sun Jul 17 19:13:55 EDT 2016
.... successful

ICRCore Stop Status:                                        [  OK
]
Starting ICRCore service at Sun Jul 17 19:14:07 EDT 2016
.... successful. ICRCore is ready at Sun Jul 17 19:14:22 EDT 2016

ICRCore Start Status:                                       [  OK
]

Installation outcome:

1. Extension and its dependencies moved to
/opt/Avaya/ICR/ICRCore/tomcat/webapps/icrcore/WEB-INF
2. dr3_icrext.properties moved to
/opt/Avaya/ICR/ICRCore/tomcat/webapps/icrcore/WEB-INF/classes
3. The ICR Core application context already includes a reference
to the entry point of the extension

Make sure you assign a value to the property dr3.service.base.url
that matches:

1. EITHER the IP of a server that contains a DR3 installed as
single-server OR
2. or the IP of the server running a load balancer of a DR3
installed as cluster

---- IMPORTANT ----
The ICR Core service must be restarted also in the AAEPMS !!!
```

7. Open the file ***dr3_icrext.properties*** located at ***/opt/Avaya/ICR/ICRCore/tomcat/webapps/icrcore/WEB-INF/classes***, and set the property ***dr3.service.base.url*** with the full location of the RESTful API used to place requests to DR, as follows:

dr3_icrext.properties

```
################################################################################
#####
# Properties file to be configured to run some tests in the PoC.
#
################################################################################
#####

# Environment setup
dr3.service.base.url=http://10.130.92.191/dr-decision-api/rest/getdestinat
ion
```

8. Go to the AAEPMS and start the ICR Core Service (AAEPMS » System Management)



At this point, ICR Core should be already started, and the extension already loaded and operational.

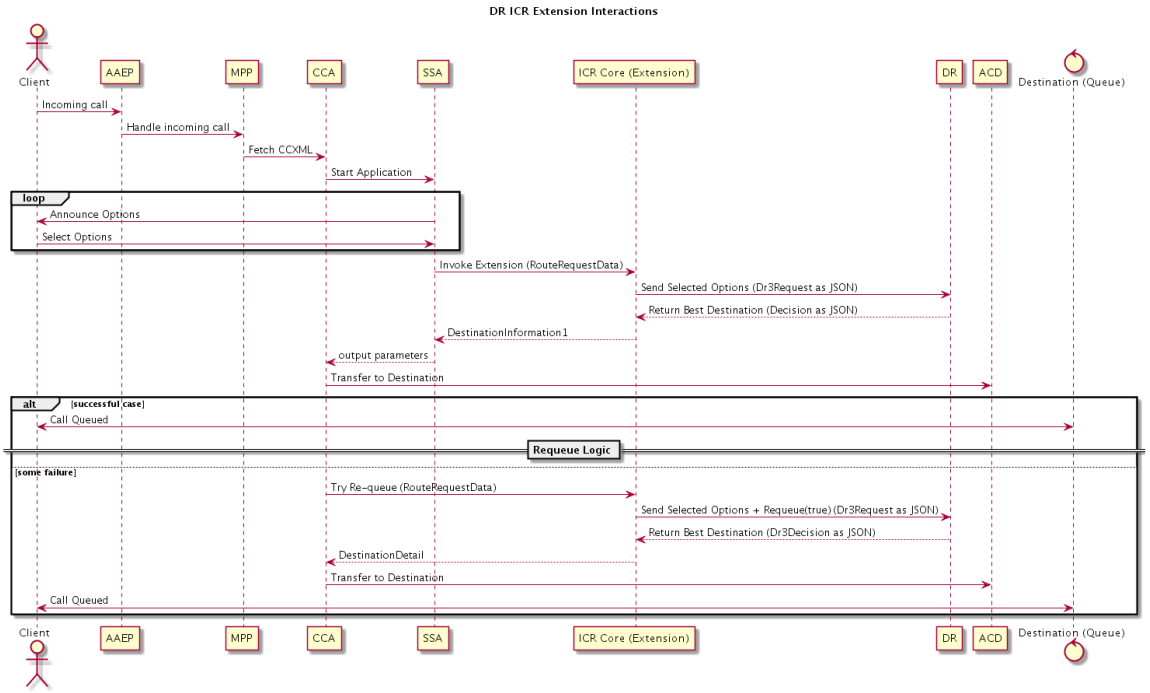
Defining the Preferred ICR Core

ICR Core is usually installed in a HA setup, in which the ICR Core instances are accessed through a load balancer responsible for the distribution of incoming requests amongst all the active instances.The scope of each ICR Core's instance is configured through an application called **runtime config**, illustrated in the following picture:

Whenever an ICR Core instance is configured as **Preferred**, which is the case of the single instance displayed in the picture, it will be seen as the instance to be chosen each time a request arrives. As well as a preferred instance, a failover ICR Core can also be defined. If neither preferred nor failover instances are configured through **runtimeconfig** (meaning that both check boxes are not selected, on each instance listed in the table), the load balacer takes place and balances the distribution of requests amongst the available ICR Core boxes.

Overall Flow

In the following diagram provides an overview of the call flow, the key components, as well as the type of the information exchanged amongst them. The ICR PDC, which communicates between the SSA and the ICR Core, is not mentioned in the diagram, as it is not a component client applications and/or customers interact with directly.



Next step: Creating an SSA to Invoke the Extension

The next page explains, in detail, how an SSA (OD application) may be composed/configured in order to invoke the Dynamic Routing ICR Core Extension and handle the extension responses.

2. Creating an SSA to Invoke the Extension

- Preparing the Orchestration Designer (OD)
  - Enabling the ICR Connector
  - Enabling the Use of CAV in OD Applications
- Example SSA Details
  - CallDR3 Node Details
  - Exit Node Details
- Next step: DR3 Configuration

In the previous section, the extension installation and configuration was described in details. At this point, the extension should be already functional, and ready to be invoked. One typical way of invoking it, is through an OD application, such as a Self-Service Application (SSA).

The SSA delivered with the extension, as an example application, is going to be explained in detail. Both the input fields - to DR3 - and the output parameters - consumed by ICR CCA - will be introduced, so that application creators are aware of all the mandatory information that must be provided to the extension and to the ICR CCA, both from the SSA.

Preparing the Orchestration Designer (OD)

OD's default settings are not enough to create an OD application invoking an ICR Core extension. Also, the default settings do not support the creation of Configurable Application Variables (CAV), a very useful instrument to collect information from users, through the AAEPMS. In the following sections, the configuration of these additional features will be explained in detail.

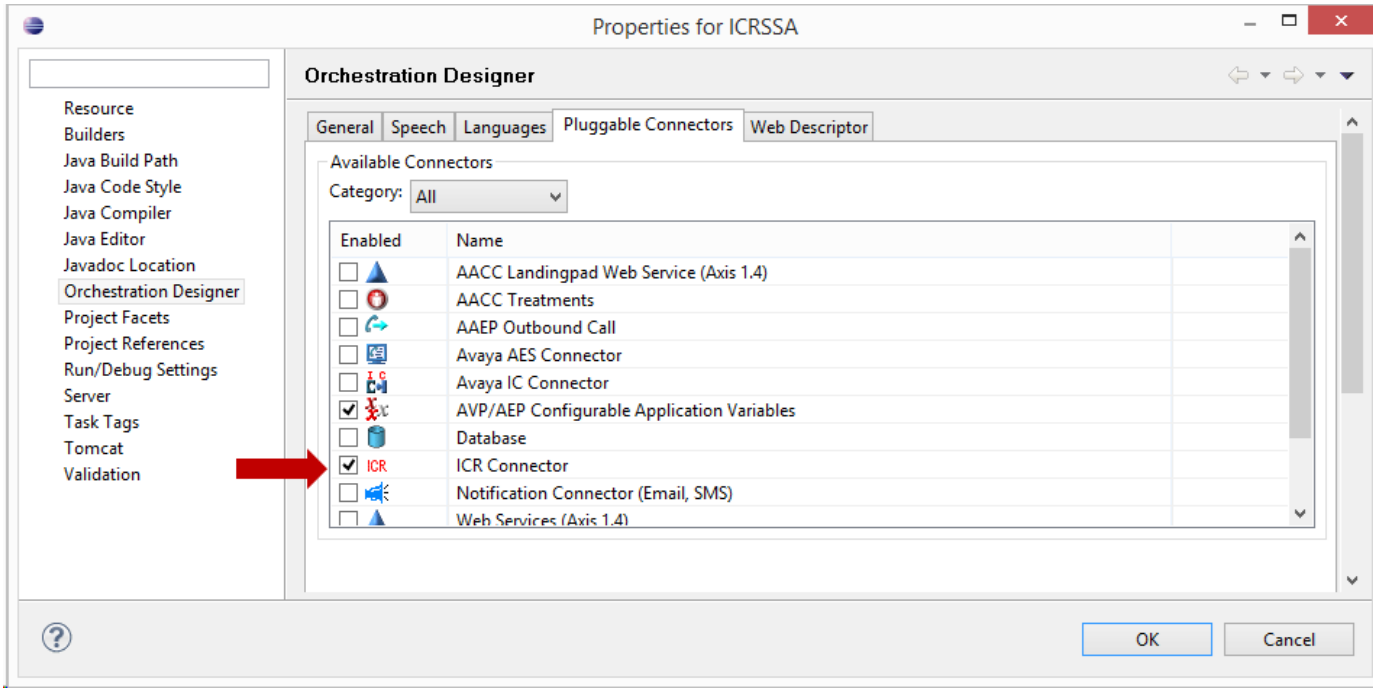
Enabling the ICR Connector

The **ICR Connector** (a.k.a. **PDC**) is delivered as part of the ICR Core bundle. To enable the connector in the Orchestration Designer, the plug-in called ***com.avaya.icr.PDC.ui.jar*** must be saved in the OD plugins directory (***<OD_HOME>/eclipse/plugins***). If OD is open at the time this plug-in is added to this directory, it must be restarted to reflect the changes.

When OD is restarted/opened, your OD application must enable the plug-in; to enable it, follow the steps below:

1. Select the project in the Project explorer
2. Click on the secondary button of the mouse, to open the context, floating menu
3. Select Properties in this menu
4. In the left menu of the Properties window, select the item Orchestration Designer
5. Select the tab Pluggable Connectors
6. Enable the item ICR Connector in the list of connectors that shows up
7. Press OK and the changes will be applied to the project

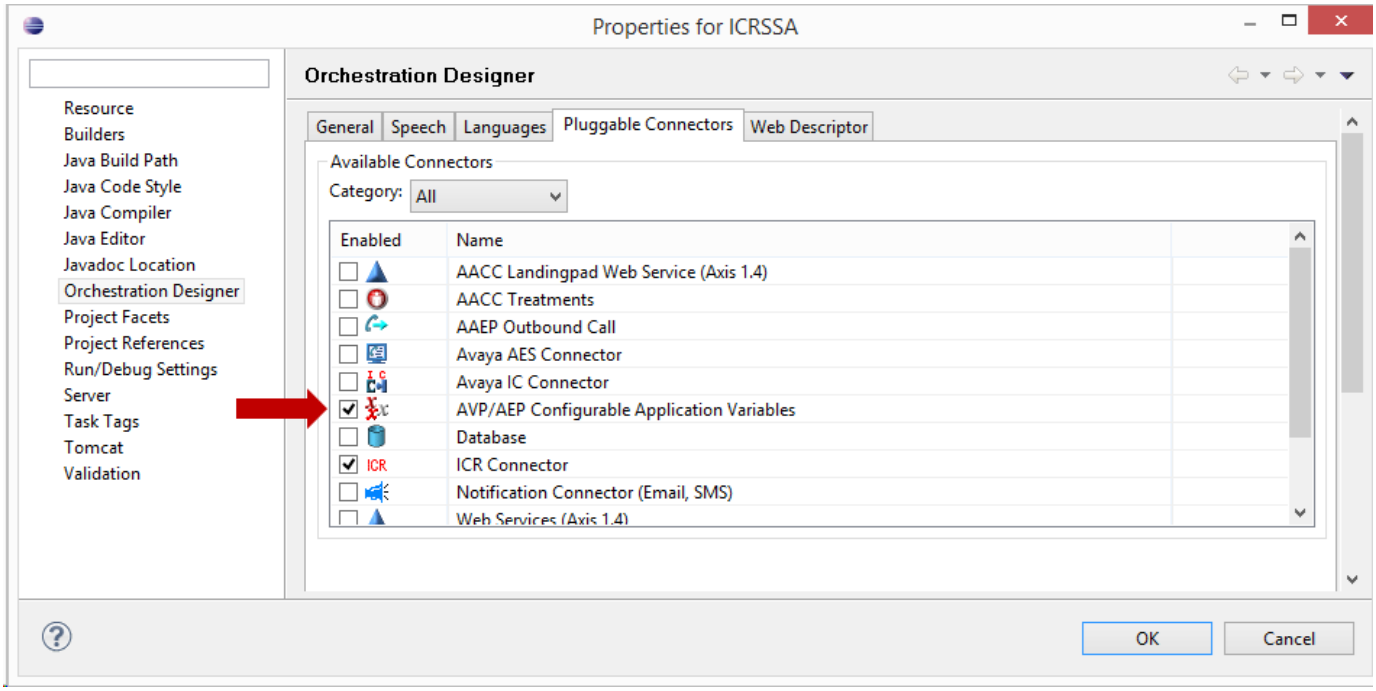
The following picture illustrates the connector in the Project Properties window.



Enabling the Use of CAV in OD Applications

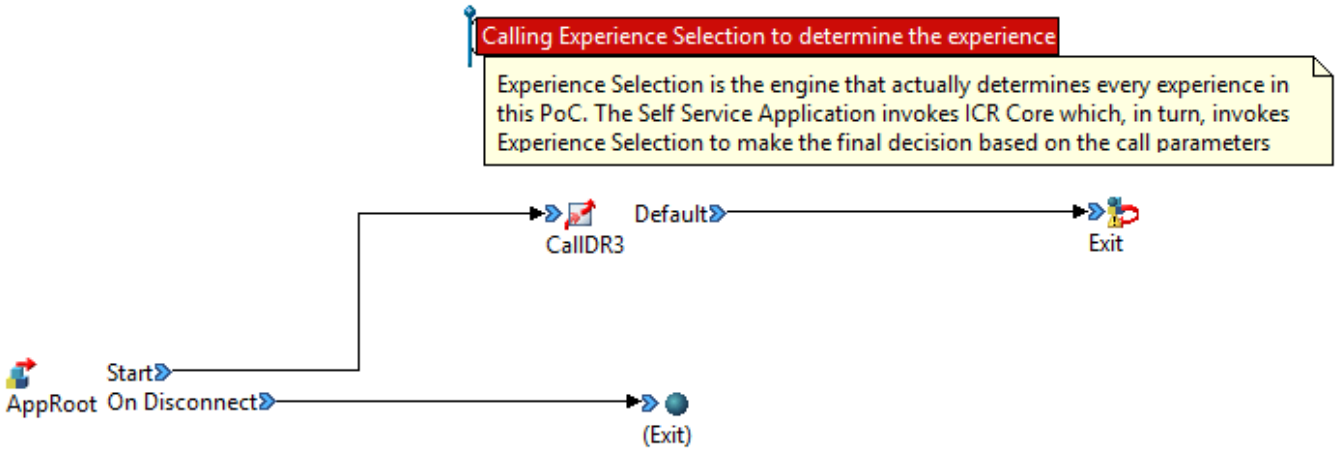
The same procedure described for ICR Connector enablement can be followed to enable the use of CAV in an OD project.

1. Select the project in the Project explorer
2. Click on the secondary button of the mouse, to open the context, floating menu
3. Select Properties in this menu
4. In the left menu of the Properties window, select the item Orchestration Designer
5. Select the tab Pluggable Connectors
6. Enable the item AVP/AEP Configurable Application Variables in the list of connectors that shows up
7. Press OK and the changes will be applied to the project



Example SSA Details

The example SSA is one of the deliverables packed with the extension. It is a reference application used to illustrate the steps one must follow to use the extension in the context of the architecture shown earlier in this document. This SSA is is an OD application managed by an ICR:CCA application configured in AAEPMS and can be deployed in web containers such as *Apache Tomcat*. It is fetched by the MPP, and its invocation happens within the context of the CCXML application fetched from the ICR Core. The overall flow of the application is illustrated in the following picture:



There is a node called CallDR3 from where - and through the PDC - the ICR Core extension is invoked. When this node completes its execution, the Exit node is responsible for filling in all the fields/attributes to be used in the CCXML to determine the logic to be executed next, and with which data.

CallDR3 Node Details

Before diving into the SSA flow, it is important to understand some of the variables declared in it. The table below summarises them:

Variable	Details

<table>
<tr>
<td>callerContext</td>
<td>This is a complex variable passed to the ICR PDC (and, consequently, to the extension), composed by the following fields:
<table>
<tr><th>Field</th><th>Value / meaning</th><th>Mandatory?</th></tr>
<tr><td>ani</td><td>The call originator (set as the 'From' attribute in the DR3 request)</td><td>YES</td></tr>
<tr><td>appName</td><td>The name of the application invoking the extension</td><td>NO</td></tr>
<tr><td>customParams</td><td>Fed by a CAV used to provide a set of parameters to be added to the custom parameters in the DR3 request. The parameters must be provided following the pattern below:

customParam1=value1;...;customParamN=valueN</td><td>NO</td></tr>
<tr><td>decisionFunction</td><td>Fed by a CAV, indicates the name of the decision function to be executed in DR3</td><td>YES</td></tr>
<tr><td>dnis</td><td>The number to be used as the destination of the call (set as the 'To' attribute in the DR3 request)</td><td>YES</td></tr>
<tr><td>requeue</td><td>Helps identifying whether the request is coming from a re-queue scenario or not. In this variable, it is a constant value, false.</td><td>NO</td></tr>
<tr><td>segAttributes</td><td>Adding this field to the caller context is one way - but not the only - to provide a set of key-value pairs to be added to the segmentation attributes map of the DR3 request. When adding this field to the caller context, make sure you follow this format:

segAttr1=value1;...;segAttrN=valueN

The name of the attributes must be the name of the segmentation dimensions defined in the segmentation table. Take this value as an example:

Package=PREMIUM;State=TX

In the segmentation table to be used by DR3 to process the request, the dimensions called Package and State must exist. The names used for each segmentation attribute, therefore, MUST correspond to dimensions declared in the segmentation table.</td><td>NO</td></tr>
<tr><td>segTableName</td><td>Fed by a CAV, indicates the name of the segmentation table to be used in DR3</td><td>NO</td></tr>
<tr><td>ucid</td><td>The universal call ID, read from the MPP session and used as the interaction ID of the DR3 request</td><td>YES</td></tr>
</table>
</td>
</tr>
</table>

<table>
<tr><td>requeueCallerContext</td><td>This context is very similar in structure to the caller context just described, with the only difference being that it is used in re-queue scenarios (assigned, therefore, to the shared variable callercontext, in the Exit node, as it will be seen in the next section of this page).

<table>
<tr><th>Field</th><th>Value / meaning</th><th>Mandatory?</th></tr>
<tr><td>ani</td><td>The call originator (set as the 'From' attribute in the DR3 request)</td><td>YES</td></tr>
<tr><td>appName</td><td>The name of the application invoking the extension</td><td>NO</td></tr>
<tr><td>customParams</td><td>Fed by a CAV used to provide a set of parameters to be added to the custom parameters in the DR3 request. The parameters must be provided following the pattern below:customParam1=value1;...;customParamN=valueN .</td><td>NO</td></tr>
<tr><td>decisionFunction</td><td>Fed by a CAV, indicates the name of the decision function to be executed in DR3</td><td>YES</td></tr>
<tr><td>dnis</td><td>The number to be used as the destination of the call (set as the 'To' attribute in the DR3 request)</td><td>YES</td></tr>
<tr><td>previousICRExecInfo</td><td>Holds the experience provided in the call before the current re-queue. This experience is represented by a set of attributes added to the DR3 extended response, but from the previous request (that failed and caused, therefore, the re-queue). From this variable, it is possible to determine the destination used in the previous experience, which may be valuable in determining the next one to be considered in the current request.
Notice the OD screenshot below ("CallDR3" node):
Assign < requeueCallerContext:previousICRExecInfo DestinationInformation1:extra_info></td><td>NO</td></tr>
<tr><td>requeue</td><td>Helps identifying whether the request is coming from a re-queue scenario or not. In this variable, it is a constant value, true.</td><td>NO</td></tr>
<tr><td>segAttributes</td><td>Adding this field to the caller context is one way - but not the only - to provide a set of key-value pairs to be added to the segmentation attributes map of the DR3 request. When adding this field to the caller context, make sure you follow this format:

segAttr1=value1;...;segAttrN=valueN

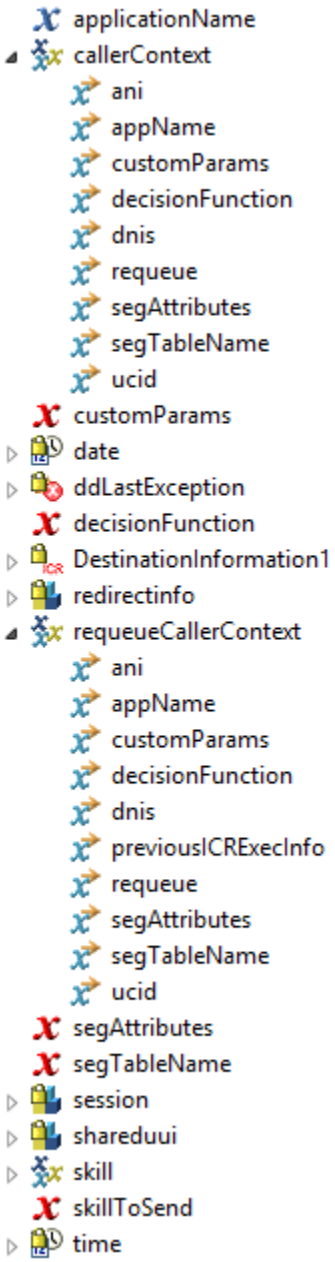
The name of the attributes must be the name of the segmentation dimensions defined in the segmentation table. Take this value as an example:

Package=PREMIUM;State=TX

In the segmentation table to be used by DR3 to process the request, the dimensions called Package and State must exist. The names used for each segmentation attribute, therefore, MUST correspond to dimensions declared in the segmentation table.</td><td>YES</td></tr>
<tr><td>segTableName</td><td>Fed by a CAV, indicates a set of attributes to be used in the call segmentation, in DR3. These attributes must be provided following the pattern below:

segAttr1=value1;...;segAttrN=valueN</td><td>NO</td></tr>
<tr><td>ucid</td><td>The universal call ID, read from the MPP session and used as the interaction ID of the DR3 request</td><td>YES</td></tr>
</table>
</td></tr>
<tr><td>decisionFunction</td><td>The SSA CAV to set the DR3 Decision Function name</td></tr>
<tr><td>segTableName</td><td>The SSA CAV to set the DR3 Segmentation Table name</td></tr>
<tr><td>segAttributes</td><td>The SSA CAV to set the DR3 segmentation attributes</td></tr>
<tr><td>skillToSend</td><td>The SSA CAV to set the ICR skill number to use. Be careful and use only numbers corresponding to ICR skills pre-configured in AAEPMS</td></tr>
<tr><td>skill</td><td>A complex variable composed by a field called 'value', that holds the ICR skill number to be used. This field is a mandatory input to invoke the ICR Core extension module</td></tr>
</table>

The following picture illustrates how these variables are configured in Orchestration Designer:



The SSA node that actually invokes DR3 looks like the picture below:

In the picture above, the field skill:value is assigned to the number provided by the user in a CAV. This is the ICR skill to be sent to the ICR Core extension module through PDC. Next, the values set on the SSA Configurable Variables are captured and assigned to the fields of the caller context variable, **callerContext**. This is a very important input from SSA that instructs the ICR Core extension on how to prepare the request to be sent to DR3.

Later, the PDC **InvokeExtension** node is used to invoke the extension itself. When the extension returns, the final assignment operation makes sure the details about the current execution are also captured and sent to CCA in a caller context entry called **previousICRExecInfo**.

Finally, the **Exit** node is used to set a bunch of variables CCA is going to use as input to take the next step in the call flow. The meaning and usage of these variables are detailed in the next section.

Note about additional parameters
Another way of sending segmentation attributes to the extension

As already described in previous sections, an entry named **segAttributes** can be added to the caller context, and segmentation attributes can be informed using a pre-defined format.

However, application creators may want to collect segmentation attributes from users, one at a time, through typing or voicing. In such cases, individual variables can be created and the values assigned to them can be added to the caller context, following another convention. In this case, entries in the caller context must be identified by a key with the prefix **drSeg**.

> Example: if a segmentation table has a dimension called **Package**, this attribute can be sent in the caller context identified by **drSegPackage**.

This prefix is used by the extension to identify the entry must be used as a segmentation attribute. Following the convention is, therefore, very important, as it helps the extension to build the DR3 request correctly.

Sending custom parameters to the extension

The extension considers as a custom parameter **ANY** entry, in the caller context, that does not match the well known attributes described in the table below and the segmentation attributes just described in the previous section (the ones identified by the prefix **drSeg**).

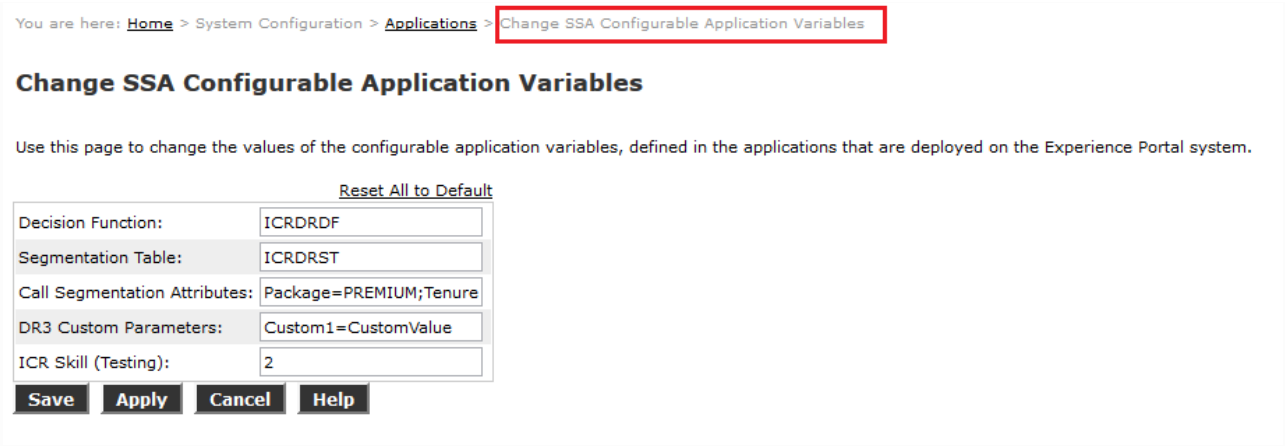
Exit Node Details

As just described, the **Exit** node is very important in this flow, as it is used to populate the many variables used by CCA to take the next step in a call flow. These values come from the extension, based on a routing decision made by DR3. The following picture shows all the assignments performed in the Exit node, complemented by a description of each variable, below.

Attribute	Description
ssaResult	The result of the SSA. The possible values to this attribute are: • AGENT_AVAILABLE: agents are available in a call center to receive calls; • AGENT_NOT_AVAILABLE: no agents are logged in a call center or all logged in agents are in AUX. In such conditions, ICR CCA starts the Generic EHA on SSA exit. The default EHA can also be overridden by starting the custom EHA on SSA exit; • GENERIC_ERROR: an error has occurred. ICR CCA starts the Generic EHA; • OUT_OF_HOURS: the call has arrived when a call center is closed. ICR CCA starts the Non Business Hours EHA; An empty value indicates the call is completed. In this case, it is disconnected. **How the extension determines the SSA result** The SSA result is determined by the extension, according to the following rules:
skillId	The skill ID identified by the SSA. ICR CCA uses the skillId value to display the calls on the ICR Monitor page of Experience Portal Manager and to requeue the calls. **Not used by the extension** It is important to reinforce Skill numbers have **no influence** BOTH in regular **and** requeue scenarios, whenever the extension is used to make routing decisions in an ICR Core setup.
vdn	The VDN to which ICR CCA queues or transfers a call. This information comes from DR3.
acd	The call center name to which ICR CCA queues or transfers a call.
acdtype	The call center type.
skilldefaultvdn	The VDN to which ICR CCA transfers a call when an error occurs.
ewt	The estimated wait time (EWT) on the destination specified by a VDN parameter. ICR CCA uses the ewt value to display the EWT for the calls in a queue on the ICR Monitor page of Experience Portal Manager.

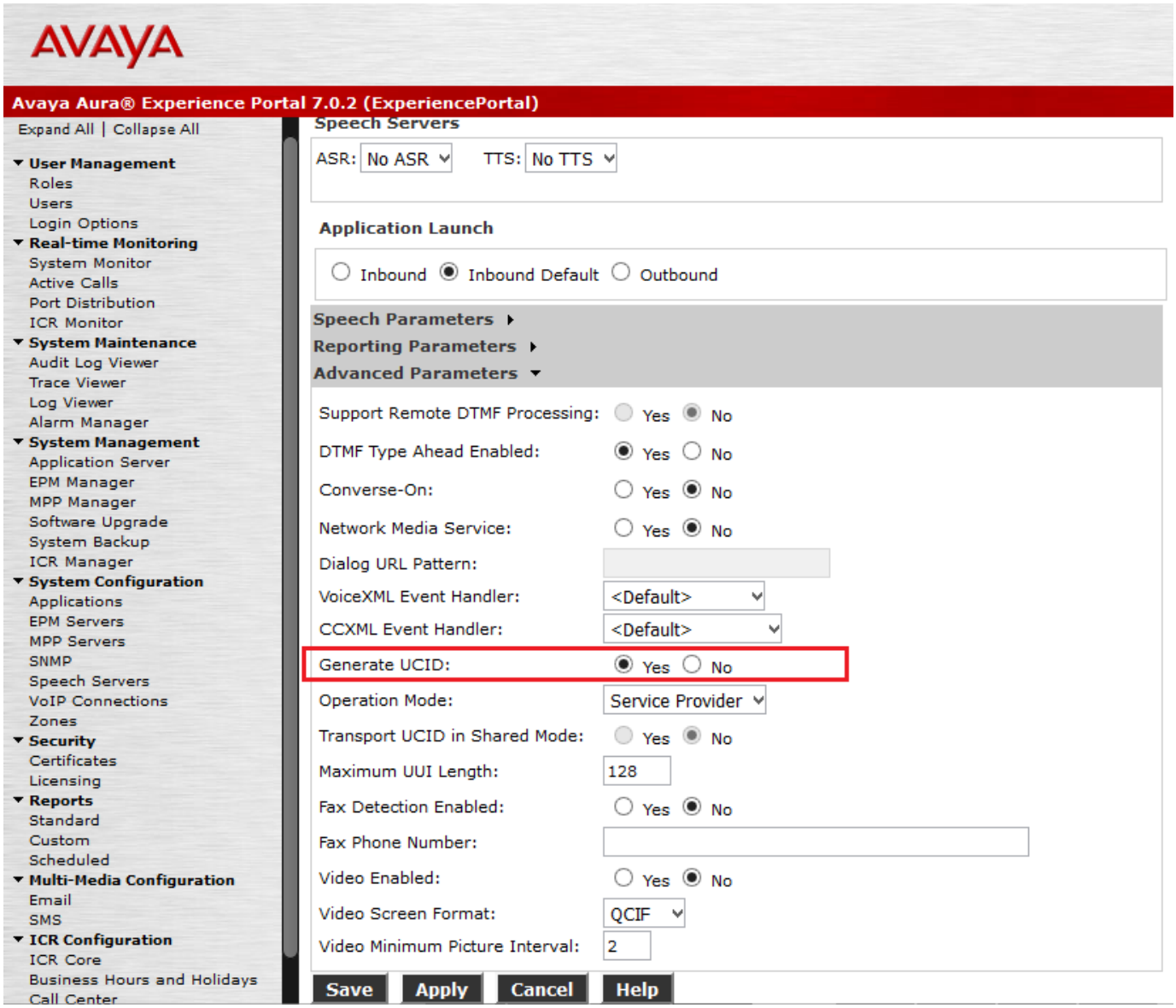
refer	A value that indicates whether to queue or transfer a call. The possible values are: • true: To transfer the call. • false: to queue the call This information is set in DR3, in the Decision Function.
ccxmlapplicationurl	The CCXML application URL
wta	The name of the WTA (configured in the AAEP as the managed application of the ICR:CCA) to start. When empty, the default WTA is used.
eha	The name of the EHA (configured in the AAEP as the managed application of the ICR:CCA) that starts when an error occurs. When empty, the generic EHA is used.
AAI	Information that you want to pass in the UUI header of a call in a queued or a transferred call.
querystring	The parameter to provide custom information to WTA
_avayaExitInfo1	The value that denotes that the SDR record is for SSA. You need this value to filter the SDR records related to SSA in the Custom Report feature of Experience Portal Manager.
UUI	The call's universal identifier
enableExternalMessage	The value that indicates whether ICR CCA can send external messages to WTA. ICR CCA sends the updated EWT and Queue position values to WTA. The possible values are: • true: To send an external message to WTA. • false: To prevent sending an external message to WTA. An empty value indicates that ICR CCA must not send the external message to WTA.
onRequeueUseExtension	The value that indicates whether ICR CCA can requeue the call using extension. The possible values are: • true: To use extension to get routing information on requeueing the call. • false: extension is not going to be used. An empty value indicates that ICR CCA **must not use** extension on requeueing the call. **Set this flag to true to involve the extension in requeue** Whenever you want to use the extension to handle requeue scenarios, set this flag to **true**.
callercontext	The complex variable that contains caller context information that is required when call is requeued using extension. Notice that, in re-queue scenarios, the SSA is feeding the extension with another caller context object. It is identical, in structure, to the object named **dr3Context**, but the flag 'requeue' is assigned to **true** this time. **Don't change the caller context structure** It is very important to follow exactly the structure defined to these caller context objects, because the extension module relies on these key/value pairs to find the information it needs to compose the DR3 request. **Requeue caller context is immutable** This object, defined in the SSA level, is NOT modified by the CCA and is passed AS IS in EVERY requeue. It means that, whenever a requeue happens more than once in a flow, given the configuration of this sample SSA, the same caller context, with the same values, will be sent to DR3 through the extension.

SSA Configurable Variables
If you are not familiar with the term 'Configurable Application Variable' (CAV), applied to OD applications, the picture below may help you understand what they mean, in practical terms.



These are the configurable variables defined to the SSA called **ICRSSA**, the OD application to serve as a reference when building your own OD applications. As you can see, they are variables exposed to customers, and whatever value is assigned to them is visible inside the application.

In order to have UCID automatically generated by the CCA, we must go to its Advanced Properties under **CCA » Advanced Parameters** and select Yes to the property called 'Generate UCID'. The picture below illustrates this action:



This property is set to No by default, so keep in mind this step is necessary.

Next step: DR3 Configuration

The next page explains, in details, how DR3 should be configured.

3. Dynamic Routing Configuration

- Dynamic Routing Request Composed by the Extension
- Example ICR-Dynamic Routing Decision Function
- Remaining Configuration
    - Reference Segmentation Table 'ICRDRST'
    - Reference ACD
    - Agent Groups
- Next section: Analyzing the Flows from Log Files

In this page, some reference scripts and entities will be detailed to give a practical example of how Dynamic Routing can be configured to receive a request from the extension and compose a response. The response can be used to create and send a DestinationDetail object back to the application that originated the invocation of the extension.

Dynamic Routing Request Composed by the Extension

When the ICR Core extension module invokes Dynamic Routing, the request gets filled in with the following fields:

Field	Value and general comments
To	Holds the DNIS of the call
From	Holds the ANI of the call
Interaction ID	Holds the UCID of the call
Segmentation attributes	Holds all the segmentation attributes provided by the user, through an SSA configurable variable (from AAEP)
Custom parameters	Holds the custom parameters provided by the user, through an SSA configurable variable (from AAEP). The custom parameters map contains: • Segmentation table name (a separate configurable variable/field in the SSA, displayed in AAEP) • Any other parameter filled in the SSA, through AAEP (there is a field in the SSA Configurable variables page where an arbitrary list of custom parameters can be provided, in the following format: customParam**1**=value**1**;...;customParam**N**=value**N**
Decision Function name	Holds the name of the decision function to be used in Dynamic Routing. This is a field filled in by the user, in an SSA configurable variable (from AAEP)

Example ICR-Dynamic Routing Decision Function

The following decision function serves as an example to customers to write their own scripts to handle calls coming from the ICR Core extension module. It contains the basic logic for receiving and handling a request, running the segmentation and composing a response.

Decision Function ICRDRDF

```
import groovy.transform.Field

/**************************************************************************
*********
 * This script is an example of how Dynamic Routing and ICR Core can be
integrated.
 **************************************************************************
*********/

debugEnable()

debug("**************************************************************
```

```
******************")
debug("***** Simple script to evaluate the integration between Dynamic
Routing and ICR *****")

// Constants
@Field final DESTINATION_FOUND = 10
@Field final SEGMENTATION_RETURNED_EMPTY_LIST = -1
@Field final UNABLE_TO_SELECT_DESTINATION = -3
@Field final UNABLE_TO_DO_SEGMENTATION = -4
@Field final ERROR_MISCONFIGURATION_NO_ADDRESS = -13

/**
 * Segmentation is run against some parameters coming in the request, as
usual.
 * The difference in this example is that both the segmentation table name
and the
 * segmentation and custom parameters are set by customers in a SSA,
through the use
 * of Configurable Application Variables (CAVs).
 *
 * For more details about the SSA configuration, refer to the DR3
Installation Guide.
 */
def segmentationTable = request.getCustomParameters().get("segTableName")
?: "ICRDRST";
def segmentationResult = segmentation.doSegmentation(segmentationTable,
request.getSegAttributes())

/**
 * Dynamic Routing may be invoked either in a regular condition (first call
from
 * ICR Core, for example) or in a re-queue scenario (when a previous
destination
 * used to queue the call on AACM caused an error and ICR Core is invoked
once more
 * by CCA to provide an alternative to it). Since ICR Core relies on ICR
Core to
 * compose the next experience to be delivered to CCA, DR3 is invoked
again, and
 * the 're-queue' flag is helpful as an additional filter when determining
the
 * next destination to be used.
 *
 * Another field that comes in the request, under re-queue conditions, is
the destination
 * selected in the previous DR3 execution, so it can be avoided this time,
even if
 * the segmentation shows it as a candidate.
 */
def previousDestination = "<unset>"

if (request.getCustomParameters().containsKey("requeue")) {
    if (request.getCustomParameters().get("requeue") == "true") {
```

```
        debug("This request is coming from a re-queue context, from ICR
CCA")
    }
}

if (request.getCustomParameters().containsKey("previousDestination")) {
    if (request.getCustomParameters().get("previousDestination") != null) {
        previousDestination =
request.getCustomParameters().get("previousDestination")
        debug("Re-Queue detected! Destination used before was " +
previousDestination)
    }
}

/**
 * The application the request is coming from. It is usually a SSA managed
by a CCA.
 */
if (request.getCustomParameters().containsKey("appName")) {
    debug("This request comes from an application named " +
request.getCustomParameters().get("appName"))
}

// If there was an error while doing segmentation or segmentation rule was
not found
if (decisionBuilder.lastDegubStepIsError() |
decisionBuilder.lastDebugStepIsNotFound()) {
    error(UNABLE_TO_DO_SEGMENTATION, "Error: Segmentation Result was not
found or there was an error")
    return
}

debug("Segmentation Result: LABEL=" + segmentationResult.label)

if (segmentationResult.destinationList.size == 0) {
    error(SEGMENTATION_RETURNED_EMPTY_LIST, "Error: Segmentation Result
has an empty destination list")
    return
}

// Getting Segmentation Result data
if (segmentationResult.destinationList.size == 1) {
    // Only one agent group
    decisionBuilder.resultCode(DESTINATION_FOUND);
    def destAliasDestination = segmentationResult.destinationList.first()
    debug("Single destination has been selected: " +
(destAliasDestination.destinationAlias ?
destAliasDestination.destinationAlias?.name + ":": "") +
destAliasDestination?.destination?.name + " / " +
destAliasDestination.destinationType.name())
    def destination = destAliasDestination.destination;

    // If there was an error while getting destination entity
```

```
    if (decisionBuilder.hasErrorCode()) {
        return
    }

    decisionBuilder.destination(destination)

    // If there was an error while doing segmentation or segmentation rule
was not found
    if (decisionBuilder.lastDegubStepIsError() |
decisionBuilder.lastDebugStepIsNotFound()) {
        error(UNABLE_TO_SELECT_DESTINATION, "Error: Destination was not
selected or there was an error")
        return
    }

    if (decisionBuilder.hasErrorCode()) {
        return
    }

    setAddress(destination)
    return
}

/**
 * It is very important to follow the rules below when using DR3 integrated
with ICR Core:
 *
 *   1. add a boolean entry identified by the key 'refer' in the extended
response.
 *      'true' indicates a blind transfer to the selected destination may
happen.
 *      'false' (preferred) indicates the transfer will be consultative,
which means
 *      that AAEP places a virtual call on AACM and keeps controlling the
flow (and a
 *      better experience, through WTA, will be provided to customers)
 *   2. add a string entry identified by the key 'sip.domain' indicating
the SIP
 *      domain of the entities (destination, ACD, etc) in the setup
 *   3. add a string entry identified by the key 'previousDestination',
which will be the
 *      selected destination in this execution. Whenever a re-queue
happens, this same
 *      destination address will be sent to DR3 in the next request made
from ICR Core
 *      extension module, and can be used to better define the next
destination to be used
 *      in the re-queue.
 *
 * Still about the SIP domain: you may prefer to set the SIP domain
information as an
 * ACD global property and retrieve this information from the ACD, in the
script. It
```

```
 * is completely OK if you do so, just remember to get this information and
set it as
 * the extended response entry in the way it is described above.
 */
def setAddress(destination) {
    def address = "<misconfiguration>"
    def addresses = destination?.addresses?.values()
    if (!addresses) {
        decisionBuilder.resultCode(ERROR_MISCONFIGURATION_NO_ADDRESS);
        decisionBuilder.appendStep("checkAddresses", destination,
addresses, "")
    }
    else if (addresses.size() > 1) {
        decisionBuilder.resultCode(ERROR_MISCONFIGURATION_NO_ADDRESS);
    }
    else {
        address = addresses.first()
        decisionBuilder.addExtendedResponse("refer", "false")
        decisionBuilder.addExtendedResponse("sip.domain", "edp.avaya.com")
        decisionBuilder.addExtendedResponse("previousDestination", address
+ "@" + "edp.avaya.com")
```

```
        }
        decisionBuilder.selectedDestAddress(address)
    }
```

There are 3 fields set in the Decision Function, that are very important to the ICR Core extension when building the DestinationDetail object to send to MPP. They are:

1. refer - true when a blind transfer has to be made; false when an assisted transfer is desired;
2. SIP domain, used to compose the address of the destination (determined by DR3)
    a. Whenever the client relies on the extension to provide the full destination address (SIP domain included), the SIP domain must be provided in the extended response, identified by this key (*sip.domain*). In this case, SIP domain can be provided hard-coded, as in the example above, or retrieved from a global property defined in the ACD or even the Destination
    b. If the client consuming the extension takes the responsibility to complete the destination address with the SIP domain, on its own, this extended response entry is not necessary
3. the current destination is set as the previousDestination. In cases of re-queue, this address is going to be considered the previous destination used, that made the re-queue happen

Such fields are set in the Dynamic Routing decision's extended response map, parsed and consumed in the ICR Core extension module.

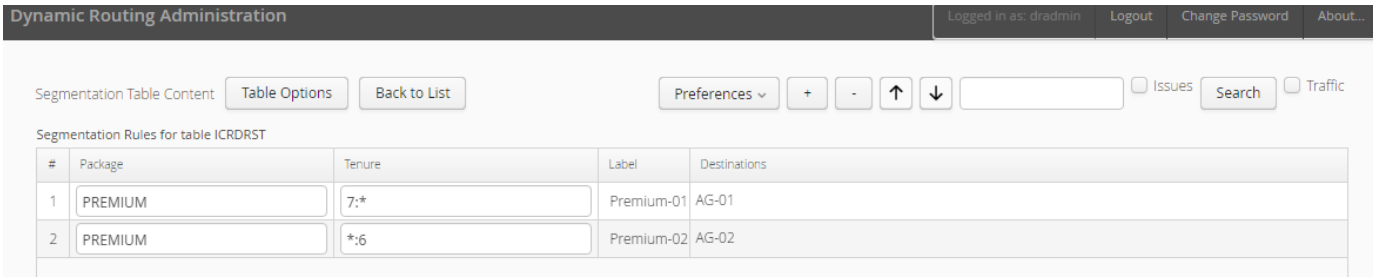
Remaining Configuration

The following elements are all entities in Dynamic Routing used to exercise the integration. They are, therefore, examples for customers to set their own environment and make use the ICR Core extension module.

Reference Segmentation Table 'ICRDRST'

Field	Description / value / comments
Name	ICRDRST
Description	Segmentation Table to exercise the ICR-Dynamic Routing integration
Dimensions	Package - String Tenure - Numeric
Min Destinations allowed	1
Max Destinations allowed	5
Display destinations	Display destination names & Destination aliases, up to 5
Include strategy	No strategy
Table evaluation mode	Row Priority Mode

Two segmentation rules were created. The first one exercises a valid VDN (the Phantom call to the AACM is going to work) and the other holds an invalid VDN, to simulate the SIP Phantom call failure (forcing the re-queue).



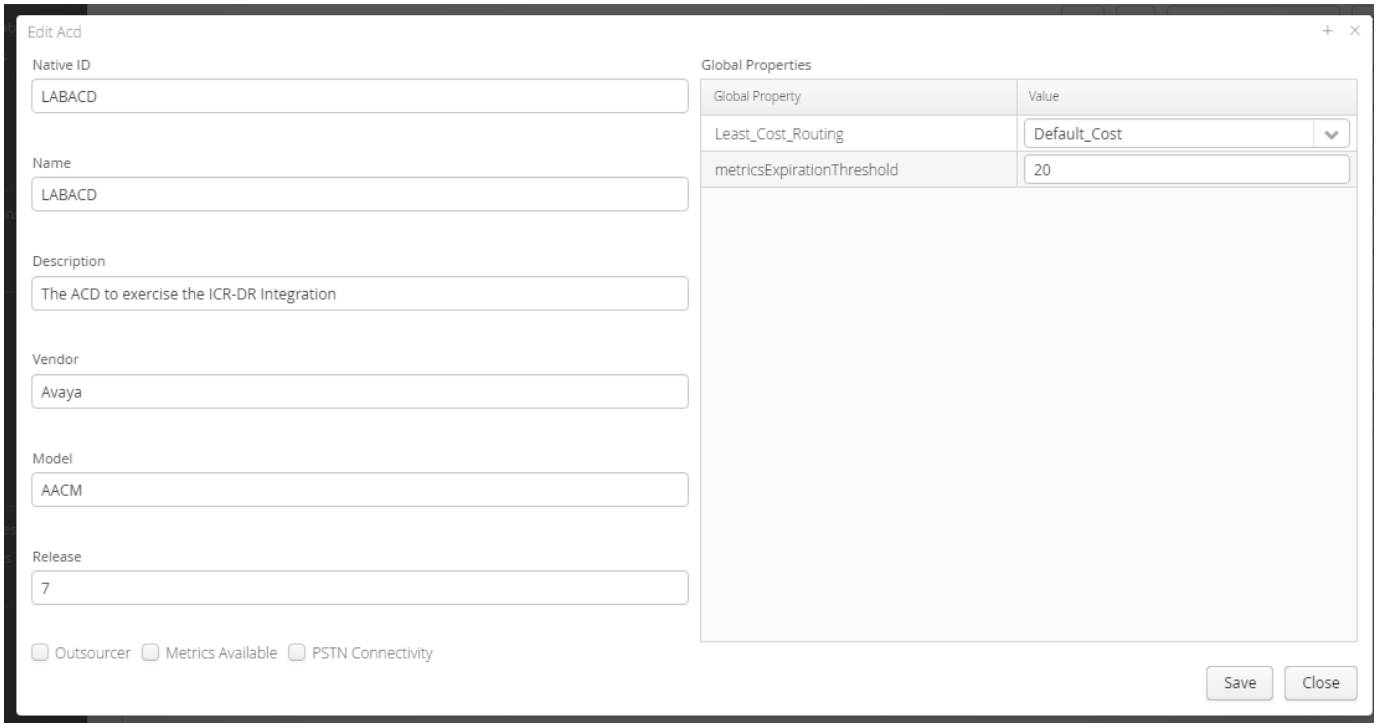
Reference ACD

To create a new ACD, we also have to save a company and a location in the system. The following tables describe these records.

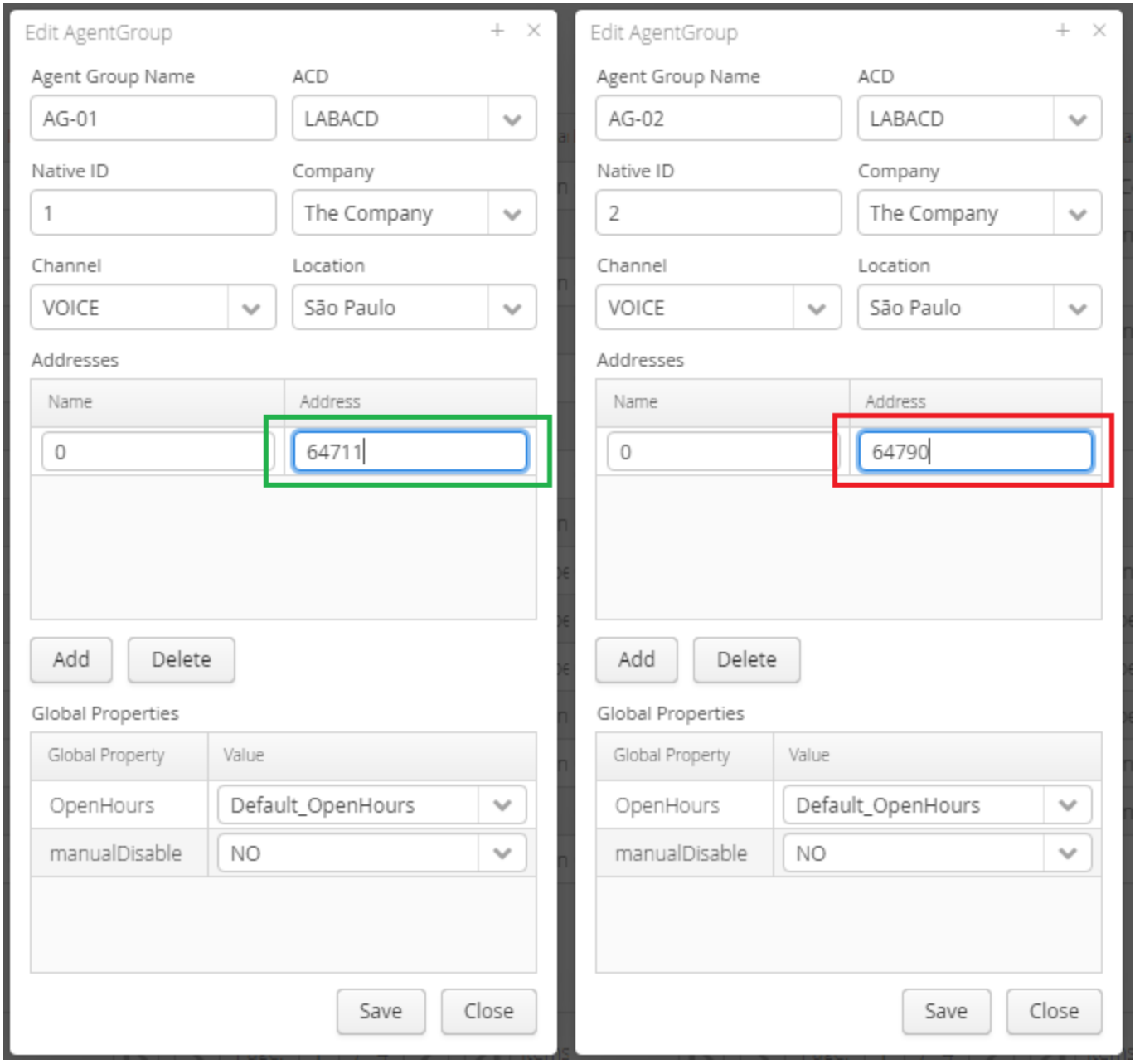
Location Field	Value
Name	São Paulo
Description	SP

Company Field	Value
Name	The Company
Description	The Company

The ACD was created as follows:



Agent Groups



The first agent group defines a single address, 64711, which is, in this example, supposed to be a valid VDN configured in the ACD system.

The second also contains a single address, 64790, which is invalid, inexisting in the AACM used in this example. It is used to trigger a re-queue scenario. Since the VDN used to place the virtual call is invalid, a re-queue is necessary, and the ICR Core Extension module is invoked again to determine the next destination to be used to queue the call.

Next section: Analyzing the Flows from Log Files

The next and last page for ICR integration with Dynamic Routing is dedicated to the log files generated by the many applications involved: SSA, ICR Core and Dynamic Routing.

These log files contain all the information one would need to fully understand the flow and investigate any suspect behavior at runtime.

4. Dynamic Routing - ICR Integration: Log Analysis

- Dynamic Routing log
- ICR Core
- CXI
- SSA log
    - Enabling Log Recording
    - Log Analysis
- Further documentation

During a call flow, some log files are fed with messages that can be used both to understand the behavior of the many components in the architecture and to troubleshoot/debug.

In this page, the most relevant files will be introduced, as well as tips to go through the text looking for meaningful information.

Dynamic Routing log

Dynamic Routing itself does not create any file specifically used to register decision making. Instead, it prints out all the information in Gigaspaces XAP log files, all located at *<DR_HOME>/dr/logs/application*. The one we need is called *dr-core.log*, and the box below illustrates how the messages we want look like.

```
...

<14>Jul 12 14:43:52 MGMT1 DR[27464]: -04:00 2016 93 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:52.093|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT -
**********************************************************************
**********
<14>Jul 12 14:43:52 MGMT1 DR[27464]: -04:00 2016 93 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:52.093|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - ***** Simple script to evaluate the
integration between Dynamic Routing and ICR *****
<14>Jul 12 14:43:52 MGMT1 DR[27464]: -04:00 2016 96 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:52.096|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request comes from an application
named ICRSSA
<14>Jul 12 14:43:52 MGMT1 DR[27464]: -04:00 2016 97 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:52.097|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Segmentation Result: LABEL=Premium-01
<14>Jul 12 14:43:52 MGMT1 DR[27464]: -04:00 2016 97 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:52.097|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Single destination has been selected:
AG-01 / AGENT_GROUP
<14>Jul 12 14:43:52 MGMT1 DR[27464]: -04:00 2016 101 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:52.101|GS-LRMI Custom
Pool-pool-4-thread-2|INFO|BestDestination PROCESSED, decision result:
{"@type":"com.avaya.ept.dr.services.routing.model.dto.DecisionDTO","timest
amp":1468349032092,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189164eca6"
,"selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","selectedDest
Name":"AG-01","selectedDestAddress":"64711","extendedResponse":{"@type":"j
ava.util.HashMap","sip.domain":"edp.avaya.com","previousDestination":"6471
1@edp.avaya.com","refer":"false"},"resultCode":10,"debugString":"SGds:S","
interactionId":"10000003671468348818","countDecision":true,"locationName":
"S?o Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
```

```
e":null}
<14>Jul 12 14:43:54 MGMT1 DR[27464]: -04:00 2016 816 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:54.816|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT -
**********************************************************************
**********
<14>Jul 12 14:43:54 MGMT1 DR[27464]: -04:00 2016 816 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:54.816|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - ***** Simple script to evaluate the
integration between Dynamic Routing and ICR *****
<14>Jul 12 14:43:54 MGMT1 DR[27464]: -04:00 2016 817 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:54.817|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request is coming from a re-queue
context, from ICR CCA
<14>Jul 12 14:43:54 MGMT1 DR[27464]: -04:00 2016 817 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:54.817|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Re-Queue detected! Destination used
before was 64711@edp.avaya.com
<14>Jul 12 14:43:54 MGMT1 DR[27464]: -04:00 2016 817 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:54.817|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request comes from an application
named ICRSSA
<14>Jul 12 14:43:54 MGMT1 DR[27464]: -04:00 2016 818 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:54.818|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Segmentation Result: LABEL=Premium-01
<14>Jul 12 14:43:54 MGMT1 DR[27464]: -04:00 2016 818 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:54.818|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Single destination has been selected:
AG-01 / AGENT_GROUP
<14>Jul 12 14:43:54 MGMT1 DR[27464]: -04:00 2016 820 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:54.820|GS-LRMI Custom
Pool-pool-4-thread-2|INFO|BestDestination PROCESSED, decision result:
{"@type":"com.avaya.ept.dr.services.routing.model.dto.DecisionDTO","timest
amp":1468349034815,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189164eca6"
,"selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","selectedDest
Name":"AG-01","selectedDestAddress":"64711","extendedResponse":{"@type":"j
ava.util.HashMap","complement":"extended_response_on_the_fly","sip.domain"
:"edp.avaya.com","previousDestination":"64711@edp.avaya.com","refer":"fals
e"},"resultCode":10,"debugString":"SGds:S","interactionId":"10000003671468
348818","countDecision":true,"locationName":"S?o Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
<14>Jul 12 14:43:57 MGMT1 DR[27464]: -04:00 2016 361 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:57.361|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT -
**********************************************************************
**********
<14>Jul 12 14:43:57 MGMT1 DR[27464]: -04:00 2016 361 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:57.361|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - ***** Simple script to evaluate the
integration between Dynamic Routing and ICR *****
<14>Jul 12 14:43:57 MGMT1 DR[27464]: -04:00 2016 362 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:57.362|GS-LRMI Custom
```

```
Pool-pool-4-thread-2|INFO| SCRIPT - This request is coming from a re-queue
context, from ICR CCA
<14>Jul 12 14:43:57 MGMT1 DR[27464]: -04:00 2016 362 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:57.362|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Re-Queue detected! Destination used
before was 64711@edp.avaya.com
<14>Jul 12 14:43:57 MGMT1 DR[27464]: -04:00 2016 362 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:57.362|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request comes from an application
named ICRSSA
<14>Jul 12 14:43:57 MGMT1 DR[27464]: -04:00 2016 362 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:57.362|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Segmentation Result: LABEL=Premium-01
<14>Jul 12 14:43:57 MGMT1 DR[27464]: -04:00 2016 363 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:57.363|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Single destination has been selected:
AG-01 / AGENT_GROUP
<14>Jul 12 14:43:57 MGMT1 DR[27464]: -04:00 2016 364 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:57.364|GS-LRMI Custom
Pool-pool-4-thread-2|INFO|BestDestination PROCESSED, decision result:
{"@type":"com.avaya.ept.dr.services.routing.model.dto.DecisionDTO","timest
amp":1468349037360,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189164eca6"
,"selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","selectedDest
Name":"AG-01","selectedDestAddress":"64711","extendedResponse":{"@type":"j
ava.util.HashMap","complement":"extended_response_on_the_fly","sip.domain"
:"edp.avaya.com","previousDestination":"64711@edp.avaya.com","refer":"fals
e"},"resultCode":10,"debugString":"SGds:S","interactionId":"10000003671468
348818","countDecision":true,"locationName":"S?o Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
<14>Jul 12 14:43:59 MGMT1 DR[27464]: -04:00 2016 909 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:59.909|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT -
**********************************************************************
**********
<14>Jul 12 14:43:59 MGMT1 DR[27464]: -04:00 2016 909 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:59.909|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - ***** Simple script to evaluate the
integration between Dynamic Routing and ICR *****
<14>Jul 12 14:43:59 MGMT1 DR[27464]: -04:00 2016 909 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:59.909|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request is coming from a re-queue
context, from ICR CCA
<14>Jul 12 14:43:59 MGMT1 DR[27464]: -04:00 2016 909 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:59.909|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Re-Queue detected! Destination used
before was 64711@edp.avaya.com
<14>Jul 12 14:43:59 MGMT1 DR[27464]: -04:00 2016 910 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:59.910|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request comes from an application
named ICRSSA
<14>Jul 12 14:43:59 MGMT1 DR[27464]: -04:00 2016 910 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:59.910|GS-LRMI Custom
```

```
Pool-pool-4-thread-2|INFO| SCRIPT - Segmentation Result: LABEL=Premium-01
<14>Jul 12 14:43:59 MGMT1 DR[27464]: -04:00 2016 910 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:59.910|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Single destination has been selected:
AG-01 / AGENT_GROUP
<14>Jul 12 14:43:59 MGMT1 DR[27464]: -04:00 2016 912 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:43:59.912|GS-LRMI Custom
Pool-pool-4-thread-2|INFO|BestDestination PROCESSED, decision result:
{"@type":"com.avaya.ept.dr.services.routing.model.dto.DecisionDTO","timest
amp":1468349039908,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189164eca6"
,"selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","selectedDest
Name":"AG-01","selectedDestAddress":"64711","extendedResponse":{"@type":"j
ava.util.HashMap","complement":"extended_response_on_the_fly","sip.domain"
:"edp.avaya.com","previousDestination":"64711@edp.avaya.com","refer":"fals
e"},"resultCode":10,"debugString":"SGds:S","interactionId":"10000003671468
348818","countDecision":true,"locationName":"S?o Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
<14>Jul 12 14:44:02 MGMT1 DR[27464]: -04:00 2016 441 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:02.441|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT -
**********************************************************************
**********
<14>Jul 12 14:44:02 MGMT1 DR[27464]: -04:00 2016 442 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:02.442|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - ***** Simple script to evaluate the
integration between Dynamic Routing and ICR *****
<14>Jul 12 14:44:02 MGMT1 DR[27464]: -04:00 2016 443 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:02.443|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request is coming from a re-queue
context, from ICR CCA
<14>Jul 12 14:44:02 MGMT1 DR[27464]: -04:00 2016 443 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:02.443|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Re-Queue detected! Destination used
before was 64711@edp.avaya.com
<14>Jul 12 14:44:02 MGMT1 DR[27464]: -04:00 2016 443 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:02.443|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request comes from an application
named ICRSSA
<14>Jul 12 14:44:02 MGMT1 DR[27464]: -04:00 2016 443 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:02.443|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Segmentation Result: LABEL=Premium-01
<14>Jul 12 14:44:02 MGMT1 DR[27464]: -04:00 2016 443 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:02.443|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Single destination has been selected:
AG-01 / AGENT_GROUP
<14>Jul 12 14:44:02 MGMT1 DR[27464]: -04:00 2016 445 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:02.445|GS-LRMI Custom
Pool-pool-4-thread-2|INFO|BestDestination PROCESSED, decision result:
{"@type":"com.avaya.ept.dr.services.routing.model.dto.DecisionDTO","timest
amp":1468349042441,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189164eca6"
,"selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","selectedDest
Name":"AG-01","selectedDestAddress":"64711","extendedResponse":{"@type":"j
```

ava.util.HashMap","complement":"extended_response_on_the_fly","sip.domain"
:"edp.avaya.com","previousDestination":"64711@edp.avaya.com","refer":"fals
e"},"resultCode":10,"debugString":"SGds:S","interactionId":"10000003671468
348818","countDecision":true,"locationName":"S?o Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
<14>Jul 12 14:44:04 MGMT1 DR[27464]: -04:00 2016 987 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:04.987|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT -
****
<14>Jul 12 14:44:04 MGMT1 DR[27464]: -04:00 2016 987 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:04.987|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - ***** Simple script to evaluate the
integration between Dynamic Routing and ICR *****
<14>Jul 12 14:44:04 MGMT1 DR[27464]: -04:00 2016 988 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:04.988|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request is coming from a re-queue
context, from ICR CCA
<14>Jul 12 14:44:04 MGMT1 DR[27464]: -04:00 2016 988 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:04.988|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Re-Queue detected! Destination used
before was 64711@edp.avaya.com
<14>Jul 12 14:44:04 MGMT1 DR[27464]: -04:00 2016 988 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:04.988|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request comes from an application
named ICRSSA
<14>Jul 12 14:44:04 MGMT1 DR[27464]: -04:00 2016 988 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:04.988|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Segmentation Result: LABEL=Premium-01
<14>Jul 12 14:44:04 MGMT1 DR[27464]: -04:00 2016 989 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:04.989|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Single destination has been selected:
AG-01 / AGENT_GROUP
<14>Jul 12 14:44:04 MGMT1 DR[27464]: -04:00 2016 990 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:04.990|GS-LRMI Custom
Pool-pool-4-thread-2|INFO|BestDestination PROCESSED, decision result:
{"@type":"com.avaya.ept.dr.services.routing.model.dto.DecisionDTO","timest
amp":1468349044986,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189164eca6"
,"selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","selectedDest
Name":"AG-01","selectedDestAddress":"64711","extendedResponse":{"@type":"j
ava.util.HashMap","complement":"extended_response_on_the_fly","sip.domain"
:"edp.avaya.com","previousDestination":"64711@edp.avaya.com","refer":"fals
e"},"resultCode":10,"debugString":"SGds:S","interactionId":"10000003671468
348818","countDecision":true,"locationName":"S?o Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
<14>Jul 12 14:44:07 MGMT1 DR[27464]: -04:00 2016 524 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:07.524|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT -
****
<14>Jul 12 14:44:07 MGMT1 DR[27464]: -04:00 2016 525 1

```
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:07.525|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - ***** Simple script to evaluate the
integration between Dynamic Routing and ICR *****
<14>Jul 12 14:44:07 MGMT1 DR[27464]: -04:00 2016 525 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:07.525|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request is coming from a re-queue
context, from ICR CCA
<14>Jul 12 14:44:07 MGMT1 DR[27464]: -04:00 2016 525 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:07.525|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Re-Queue detected! Destination used
before was 64711@edp.avaya.com
<14>Jul 12 14:44:07 MGMT1 DR[27464]: -04:00 2016 526 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:07.526|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - This request comes from an application
named ICRSSA
<14>Jul 12 14:44:07 MGMT1 DR[27464]: -04:00 2016 526 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:07.526|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Segmentation Result: LABEL=Premium-01
<14>Jul 12 14:44:07 MGMT1 DR[27464]: -04:00 2016 526 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:07.526|GS-LRMI Custom
Pool-pool-4-thread-2|INFO| SCRIPT - Single destination has been selected:
AG-01 / AGENT_GROUP
<14>Jul 12 14:44:07 MGMT1 DR[27464]: -04:00 2016 528 1
com.avaya.ept.dynamicrouting | 0 |@2016-07-12/14:44:07.528|GS-LRMI Custom
Pool-pool-4-thread-2|INFO|BestDestination PROCESSED, decision result:
{"@type":"com.avaya.ept.dr.services.routing.model.dto.DecisionDTO","timest
amp":1468349047524,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189164eca6"
,"selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","selectedDest
Name":"AG-01","selectedDestAddress":"64711","extendedResponse":{"@type":"j
ava.util.HashMap","complement":"extended_response_on_the_fly","sip.domain"
:"edp.avaya.com","previousDestination":"64711@edp.avaya.com","refer":"fals
e"},"resultCode":10,"debugString":"SGds:S","interactionId":"10000003671468
348818","countDecision":true,"locationName":"S?o Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
<14>Jul 12 14:45:34 MGMT1 DR[27464]: -04:00 2016 588 1
com.avaya.ept.dynamicrouting | 0
|@2016-07-12/14:45:34.588|pool-46-thread-1|INFO|INVOKING WEBLM CONNECTOR
<14>Jul 12 14:45:34 MGMT1 DR[27464]: -04:00 2016 602 1
com.avaya.ept.dynamicrouting | 0
|@2016-07-12/14:45:34.602|pool-46-thread-1|INFO|License internal state
```

```
enabled=true

...
```

This is an example where many re-queue cycles happened in sequence. In such scenarios, notice that Dynamic Routing receives not only the information to compose a brand new decision, but some historical data related to the previous request. This information is key to a script writer, for example, who can apply some additional filters when building the experience to be delivered on the current request.

ICR Core

The most important log file in ICR Core is called **AvayaICRDebug.log**, found at *<ICR_HOME>/ICRCore/tomcat/logs*. AvayaICRDebug.log. The box below shows a log snippet that contains the most relevant messages one could use to analyze a call flow from the ICR Core's standpoint.

AvayaICRDebug.log

```
...

2016-07-12 14:40:19,524|INFO
|com.avaya.icr.core.service.SkillLookUpService|http-20443-1|[CCXML-Session
-ID: aaepicr-2016194184018-18] -- WebService request recevied from host = [
10.130.125.112 ] for operation = [ getDestinationEx ] and skill id = [ 1 ]
2016-07-12
14:40:19,524|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20443-1|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Checking for license
availability...
2016-07-12
14:40:19,525|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20443-1|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Validating icrcore service
status.
2016-07-12 14:40:19,525|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20443-1|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Validating ICR skill Id = 1
2016-07-12 14:40:19,527|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20443-1|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Performing best destination selection for
skill = [ 1 ] using custom extension module - ICRCORE-DR3-EXT
2016-07-12
14:40:19,527|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20443-1|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Following are the caller context
information received from the application : [Context] key = ani, value =
64500
[Context] key = appName, value = ICRSSA
[Context] key = customParams, value =
Custom1=CustomValue;Custom2=CustomValue2
[Context] key = decisionFunction, value = ICRDRDF
[Context] key = dnis, value = 64600
[Context] key = requeue, value = false
[Context] key = segAttributes, value = Package=PREMIUM;Tenure=14
[Context] key = segTableName, value = ICRDRST
[Context] key = ucid, value = 10000003671468348818
```

```
2016-07-12 14:40:19,528|INFO
|com.avaya.icr.sdk.util.ICRLogManager|http-20443-1|Creating Default logger
for com.avaya.ept.dr.icr.extensions.icr.util.LogManager
2016-07-12
14:40:19,528|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20443
-1|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:19,528|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20443
-1|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:19,528|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20443
-1|Getting Data from the Shared ExternalCache
2016-07-12
14:40:19,532|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Cache is still empty. No previous execution for Session ID
aaepicr-2016194184018-18
2016-07-12
14:40:19,532|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20443
-1|Entry StoreData(aaepicr-2016194184018-18_count)
2016-07-12
14:40:19,710|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20443
-1|Exit StoreData(aaepicr-2016194184018-18_count)
2016-07-12 14:40:19,765|INFO
|com.avaya.icr.sdk.util.ICRLogManager|http-20443-1|Creating Default logger
for com.avaya.icr.sdk.util.ICRUtils
2016-07-12
14:40:19,790|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
ICRCORE-DR3-EXTDR3 POST message body:
{"decisionFunctionName":"ICRDRDF","interactionId":"10000003671468348818","
from":"64500","to":"64600","customParameters":{"appName":"ICRSSA","segTabl
eName":"ICRDRST","requeue":"false","Custom1":"CustomValue","Custom2":"Cust
omValue2"},"segAttributes":{"Tenure":"14","Package":"PREMIUM"}}
2016-07-12
14:40:19,791|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Dynamic Routing REST URL:
http://10.130.125.107:9090/dr_routing_rest_pu/rest/getdestination
2016-07-12
14:40:19,812|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
DR3 HTTP Response Code: 200
2016-07-12
14:40:19,813|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
DR3 Response:
{"timestamp":1468349032092,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189
164eca6","selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","sele
ctedDestName":"AG-01","selectedDestAddress":"64711","extendedResponse":{"s
ip.domain":"edp.avaya.com","refer":"false","previousDestination":"64711@ed
p.avaya.com"},"resultCode":10,"debugString":"SGds:S","interactionId":"1000
```

```
0003671468348818","countDecision":true,"locationName":"São
Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
2016-07-12
14:40:19,824|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20443
-1|Entry StoreData(aaepicr-2016194184018-18)
2016-07-12
14:40:19,826|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20443
-1|Exit StoreData(aaepicr-2016194184018-18)
2016-07-12
14:40:19,826|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
ICR Skill ID: 1
2016-07-12
14:40:19,826|ERROR|com.avaya.icr.sdk.util.ICRUtils|http-20443-1|[CCXML-Ses
sion-ID: aaepicr-2016194184018-18] -- Skill record is null for destination
[Destination VO [ ICRSkill = { 1 }, QueueVDN = null, PollingVDN = null, Acd
= { null }, AcdSkill = { null }, BusinessGroup = { null },HolidayGroup = {
null }, identifier = { null},skillRecord = {null} ]]
2016-07-12
14:40:19,826|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Destination to be used: 64711@edp.avaya.com
2016-07-12
14:40:19,826|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Refer property value: false
2016-07-12
14:40:19,828|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
DR3 extended response:
{"sip.domain":"edp.avaya.com","refer":"false","previousDestination":"64711
@edp.avaya.com"}
2016-07-12
14:40:19,828|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20443-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Destination Detail custom data:
{"sip.domain":"edp.avaya.com","refer":"false","previousDestination":"64711
@edp.avaya.com"}
2016-07-12 14:40:19,828|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20443-1|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Returning routing results : DestinationDetail
[identifier=null, destination=64711@edp.avaya.com, acd=null,
acdType=Avaya-CM, defaultDest=64710@edp.avaya.com, defaultACD=LABACD,
result=AGENT_AVAILABLE, ewt=-1, refer=false,
customData={"sip.domain":"edp.avaya.com","refer":"false","previousDestinat
ion":"64711@edp.avaya.com"}, elapsedTime=0, ccxmlAppUrl=, ewtTop=-1,
ewtHigh=-1, ewtMedium=-1, ewtLow=-1, agentsStaffed=-1, agentsAvailable=-1,
estimatedQpos=-1, cmskillnumber=-1, priority=-1, percAnsSL=]
2016-07-12 14:40:22,512|INFO
|com.avaya.icr.core.service.SkillLookUpService|http-20080-1|[CCXML-Session
-ID: aaepicr-2016194184018-18] -- WebService request recevied from host = [
```

```
135.122.99.97 ] for operation = [ getDestinationEx ] and skill id = [ 1 ]
2016-07-12
14:40:22,512|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20080-1|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Checking for license
availability...
2016-07-12
14:40:22,512|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20080-1|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Validating icrcore service
status.
2016-07-12 14:40:22,512|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20080-1|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Validating ICR skill Id = 1
2016-07-12 14:40:22,513|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20080-1|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Performing best destination selection for
skill = [ 1 ] using custom extension module - ICRCORE-DR3-EXT
2016-07-12
14:40:22,513|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20080-1|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Following are the caller context
information received from the application : [Context] key = customParams,
value = Custom1=CustomValue;Custom2=CustomValue2
[Context] key = appName, value = ICRSSA
[Context] key = previousICRExecInfo, value =
{"sip.domain":"edp.avaya.com","refer":"false","previousDestination":"64711
@edp.avaya.com"}
[Context] key = segAttributes, value = Package=PREMIUM;Tenure=14
[Context] key = segTableName, value = ICRDRST
[Context] key = dnis, value = 64600
[Context] key = ucid, value = 10000003671468348818
[Context] key = ani, value = 64500
[Context] key = decisionFunction, value = ICRDRDF
[Context] key = requeue, value = true

2016-07-12
14:40:22,513|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-1|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:22,513|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-1|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:22,513|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-1|Getting Data from the Shared ExternalCache
2016-07-12
14:40:22,514|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-1|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:22,514|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-1|Getting Data from the Shared ExternalCache
2016-07-12
14:40:22,515|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous extension executions for Session ID aaepicr-2016194184018-18: 1
2016-07-12
```

```
14:40:22,515|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-1|Entry StoreData(aaepicr-2016194184018-18_count)
2016-07-12
14:40:22,516|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-1|Exit StoreData(aaepicr-2016194184018-18_count)
2016-07-12
14:40:22,518|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous ICR Execution has a parameter 'sip.domain' with value
'edp.avaya.com'
2016-07-12
14:40:22,518|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous ICR Execution has a parameter 'refer' with value 'false'
2016-07-12
14:40:22,518|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous ICR Execution has a parameter 'previousDestination' with value
'64711@edp.avaya.com'
2016-07-12
14:40:22,518|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
ICRCORE-DR3-EXTDR3 POST message body:
{"decisionFunctionName":"ICRDRDF","interactionId":"10000003671468348818","
from":"64500","to":"64600","customParameters":{"appName":"ICRSSA","sip.dom
ain":"edp.avaya.com","segTableName":"ICRDRST","requeue":"true","previousDe
stination":"64711@edp.avaya.com","refer":"false","Custom1":"CustomValue","
Custom2":"CustomValue2"},"segAttributes":{"Tenure":"14","Package":"PREMIUM
"}}
2016-07-12
14:40:22,518|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Dynamic Routing REST URL:
http://10.130.125.107:9090/dr_routing_rest_pu/rest/getdestination
2016-07-12
14:40:22,530|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
DR3 HTTP Response Code: 200
2016-07-12
14:40:22,531|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
DR3 Response:
{"timestamp":1468349034815,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189
164eca6","selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","sele
ctedDestName":"AG-01","selectedDestAddress":"64711","extendedResponse":{"c
omplement":"extended_response_on_the_fly","sip.domain":"edp.avaya.com","re
fer":"false","previousDestination":"64711@edp.avaya.com"},"resultCode":10,
"debugString":"SGds:S","interactionId":"10000003671468348818","countDecisi
on":true,"locationName":"São Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
2016-07-12
14:40:22,532|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
```

```
-1|Entry StoreData(aaepicr-2016194184018-18)
2016-07-12
14:40:22,533|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-1|Exit StoreData(aaepicr-2016194184018-18)
2016-07-12
14:40:22,533|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
ICR Skill ID: 1
2016-07-12
14:40:22,533|ERROR|com.avaya.icr.sdk.util.ICRUtils|http-20080-1|[CCXML-Ses
sion-ID: aaepicr-2016194184018-18] -- Skill record is null for destination
[Destination VO [ ICRSkill = { 1 }, QueueVDN = null, PollingVDN = null, Acd
= { null }, AcdSkill = { null }, BusinessGroup = { null },HolidayGroup = {
null }, identifier = { null},skillRecord = {null} ]]
2016-07-12
14:40:22,533|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Destination to be used: 64711@edp.avaya.com
2016-07-12
14:40:22,533|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Refer property value: false
2016-07-12
14:40:22,533|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
DR3 extended response:
{"complement":"extended_response_on_the_fly","sip.domain":"edp.avaya.com",
"refer":"false","previousDestination":"64711@edp.avaya.com"}
2016-07-12
14:40:22,534|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-1|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Destination Detail custom data:
{"complement":"extended_response_on_the_fly","sip.domain":"edp.avaya.com",
"refer":"false","previousDestination":"64711@edp.avaya.com"}
2016-07-12 14:40:22,534|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20080-1|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Returning routing results : DestinationDetail
[identifier=null, destination=64711@edp.avaya.com, acd=null,
acdType=Avaya-CM, defaultDest=64710@edp.avaya.com, defaultACD=LABACD,
result=AGENT_AVAILABLE, ewt=-1, refer=false,
customData={"complement":"extended_response_on_the_fly","sip.domain":"edp.
avaya.com","refer":"false","previousDestination":"64711@edp.avaya.com"},
elapsedTime=0, ccxmlAppUrl=, ewtTop=-1, ewtHigh=-1, ewtMedium=-1,
ewtLow=-1, agentsStaffed=-1, agentsAvailable=-1, estimatedQpos=-1,
cmskillnumber=-1, priority=-1, percAnsSL=]
2016-07-12 14:40:25,049|INFO
|com.avaya.icr.core.service.SkillLookUpService|http-20080-2|[CCXML-Session
-ID: aaepicr-2016194184018-18] -- WebService request recevied from host = [
135.122.99.97 ] for operation = [ getDestinationEx ] and skill id = [ 1 ]
2016-07-12
14:40:25,050|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20080-2|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Checking for license
availability...
```

```
2016-07-12
14:40:25,050|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20080-2|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Validating icrcore service
status.
2016-07-12 14:40:25,050|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20080-2|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Validating ICR skill Id = 1
2016-07-12 14:40:25,050|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20080-2|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Performing best destination selection for
skill = [ 1 ] using custom extension module - ICRCORE-DR3-EXT
2016-07-12
14:40:25,050|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20080-2|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Following are the caller context
information received from the application : [Context] key = customParams,
value = Custom1=CustomValue;Custom2=CustomValue2
[Context] key = appName, value = ICRSSA
[Context] key = previousICRExecInfo, value =
{"sip.domain":"edp.avaya.com","refer":"false","previousDestination":"64711
@edp.avaya.com"}
[Context] key = segAttributes, value = Package=PREMIUM;Tenure=14
[Context] key = segTableName, value = ICRDRST
[Context] key = dnis, value = 64600
[Context] key = ucid, value = 10000003671468348818
[Context] key = ani, value = 64500
[Context] key = decisionFunction, value = ICRDRDF
[Context] key = requeue, value = true

2016-07-12
14:40:25,050|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:25,050|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:25,050|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Getting Data from the Shared ExternalCache
2016-07-12
14:40:25,056|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:25,056|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Getting Data from the Shared ExternalCache
2016-07-12
14:40:25,056|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous extension executions for Session ID aaepicr-2016194184018-18: 2
2016-07-12
14:40:25,056|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Entry StoreData(aaepicr-2016194184018-18_count)
2016-07-12
14:40:25,058|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Exit StoreData(aaepicr-2016194184018-18_count)
```

```
2016-07-12
14:40:25,058|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Checking if Shared ExternalCache is available: true
2016-07-12
14:40:25,058|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Getting Data from the Shared ExternalCache
2016-07-12
14:40:25,059|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Caller context updated with previous ICR execution info:
{"complement":"extended_response_on_the_fly","sip.domain":"edp.avaya.com",
"refer":"false","previousDestination":"64711@edp.avaya.com"}
2016-07-12
14:40:25,059|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous ICR Execution has a parameter 'complement' with value
'extended_response_on_the_fly'
2016-07-12
14:40:25,060|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous ICR Execution has a parameter 'sip.domain' with value
'edp.avaya.com'
2016-07-12
14:40:25,060|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous ICR Execution has a parameter 'refer' with value 'false'
2016-07-12
14:40:25,060|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Previous ICR Execution has a parameter 'previousDestination' with value
'64711@edp.avaya.com'
2016-07-12
14:40:25,061|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
ICRCORE-DR3-EXTDR3 POST message body:
{"decisionFunctionName":"ICRDRDF","interactionId":"10000003671468348818","
from":"64500","to":"64600","customParameters":{"complement":"extended_resp
onse_on_the_fly","appName":"ICRSSA","sip.domain":"edp.avaya.com","segTable
Name":"ICRDRST","requeue":"true","previousDestination":"64711@edp.avaya.co
m","refer":"false","Custom1":"CustomValue","Custom2":"CustomValue2"},"segA
ttributes":{"Tenure":"14","Package":"PREMIUM"}}
2016-07-12
14:40:25,061|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Dynamic Routing REST URL:
http://10.130.125.107:9090/dr_routing_rest_pu/rest/getdestination
2016-07-12
14:40:25,075|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
DR3 HTTP Response Code: 200
2016-07-12
14:40:25,076|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
```

```
DR3 Response:
{"timestamp":1468349037360,"selectedDestId":"7ecb127d-1b0f-4c9a-966e-52189
164eca6","selectedDestNativeId":"1","selectedDestType":"AGENT_GROUP","sele
ctedDestName":"AG-01","selectedDestAddress":"64711","extendedResponse":{"c
omplement":"extended_response_on_the_fly","sip.domain":"edp.avaya.com","re
fer":"false","previousDestination":"64711@edp.avaya.com"},"resultCode":10,
"debugString":"SGds:S","interactionId":"10000003671468348818","countDecisi
on":true,"locationName":"São Paulo","companyName":"The
Company","segLabel":"Premium-01","segTable":"ICRDRST","strategySettingsNam
e":null}
2016-07-12
14:40:25,077|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Entry StoreData(aaepicr-2016194184018-18)
2016-07-12
14:40:25,078|DEBUG|com.avaya.icr.sdk.cache.ExternalCacheManager|http-20080
-2|Exit StoreData(aaepicr-2016194184018-18)
2016-07-12
14:40:25,078|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
ICR Skill ID: 1
2016-07-12
14:40:25,078|ERROR|com.avaya.icr.sdk.util.ICRUtils|http-20080-2|[CCXML-Ses
sion-ID: aaepicr-2016194184018-18] -- Skill record is null for destination
[Destination VO [ ICRSkill = { 1 }, QueueVDN = null, PollingVDN = null, Acd
= { null }, AcdSkill = { null }, BusinessGroup = { null },HolidayGroup = {
null }, identifier = { null},skillRecord = {null} ]]
2016-07-12
14:40:25,079|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Destination to be used: 64711@edp.avaya.com
2016-07-12
14:40:25,079|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Refer property value: false
2016-07-12
14:40:25,079|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
DR3 extended response:
{"complement":"extended_response_on_the_fly","sip.domain":"edp.avaya.com",
"refer":"false","previousDestination":"64711@edp.avaya.com"}
2016-07-12
14:40:25,079|DEBUG|com.avaya.ept.dr.icr.extensions.icr.util.LogManager|htt
p-20080-2|[CCXML-Session-ID: aaepicr-2016194184018-18] -- [DR3 EXTENSION]
Destination Detail custom data:
{"complement":"extended_response_on_the_fly","sip.domain":"edp.avaya.com",
"refer":"false","previousDestination":"64711@edp.avaya.com"}
2016-07-12 14:40:25,079|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20080-2|[CCXML-Session-ID:
aaepicr-2016194184018-18] -- Returning routing results : DestinationDetail
[identifier=null, destination=64711@edp.avaya.com, acd=null,
acdType=Avaya-CM, defaultDest=64710@edp.avaya.com, defaultACD=LABACD,
result=AGENT_AVAILABLE, ewt=-1, refer=false,
customData={"complement":"extended_response_on_the_fly","sip.domain":"edp.
```

```
avaya.com","refer":"false","previousDestination":"64711@edp.avaya.com"},
elapsedTime=0, ccxmlAppUrl=, ewtTop=-1, ewtHigh=-1, ewtMedium=-1,
ewtLow=-1, agentsStaffed=-1, agentsAvailable=-1, estimatedQpos=-1,
cmskillnumber=-1, priority=-1, percAnsSL=]
2016-07-12 14:40:27,600|INFO
|com.avaya.icr.core.service.SkillLookUpService|http-20080-3|[CCXML-Session
-ID: aaepicr-2016194184018-18] -- WebService request recevied from host = [
135.122.99.97 ] for operation = [ getDestinationEx ] and skill id = [ 1 ]
2016-07-12
14:40:27,600|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20080-3|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Checking for license
availability...
2016-07-12
14:40:27,600|DEBUG|com.avaya.icr.core.ssi.RouteSelector|http-20080-3|[CCXM
L-Session-ID: aaepicr-2016194184018-18] -- Validating icrcore service
status.
2016-07-12 14:40:27,600|INFO
|com.avaya.icr.core.ssi.RouteSelector|http-20080-3|[CCXML-Session-ID:
```

```
aaepicr-2016194184018-18] -- Validating ICR skill Id = 1

...
```

From line 8 to 16, the extension prints the caller context received in the request. This is useful to see all the parameters sent to the extension, through the caller context object.

Next, in line 22, there is information about the ICR cache. As it is the first time the extension is being invoked by the SSA, the cache is still empty. This cache is used to store the latest extended response received from the Dynamic Routing, sent back to Dynamic Routing each time the extension is invoked again, in a requeue condition. In lines 60 and 100, another message indicates how many times the extension was executed within the same MPP session, withinformation stored each time the extension is called.

In line 26, it is possible to see the full Dynamic Routing request body, formatted in JSON, which contains all the information Dynamic Routing needs to execute and make a decision.

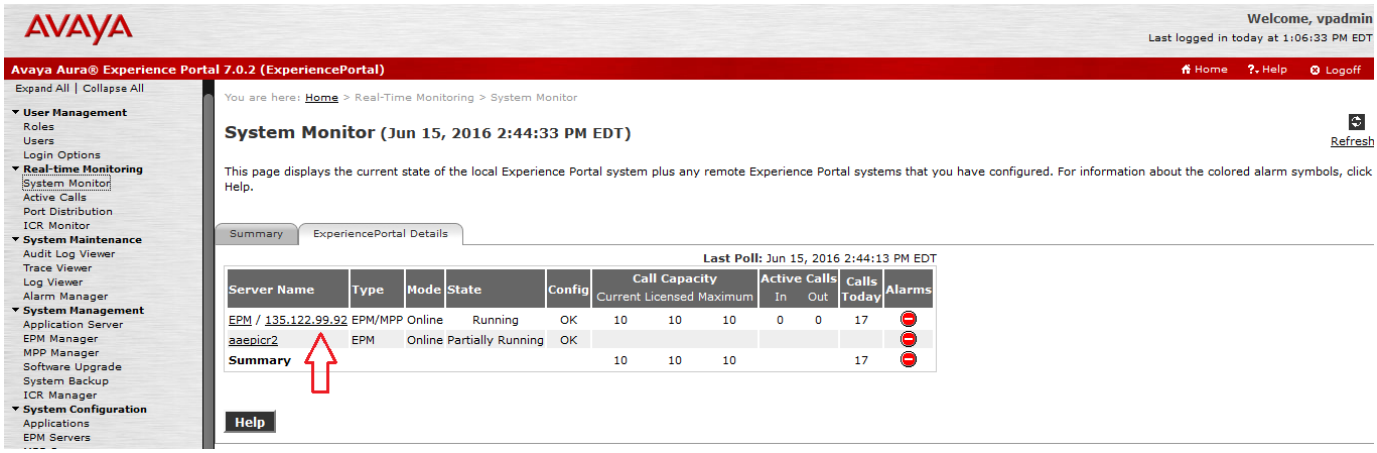
Line 29 contains the associated Dynamic Routing response, also in JSON format. There it is possible to see what was returned by Dynamic Routing, from the selected destination to all the fields of the extended response.

Finally, in lines 66 and 110, new calls to Dynamic Routing can be seen. This is a log snippet extract from a requeue exercise, when the destination (a.k.a. VDN) returned by Dynamic Routing was, on purpose, set to AUX, forcing the requeue to happen and the extension to be invoked again, three more times.

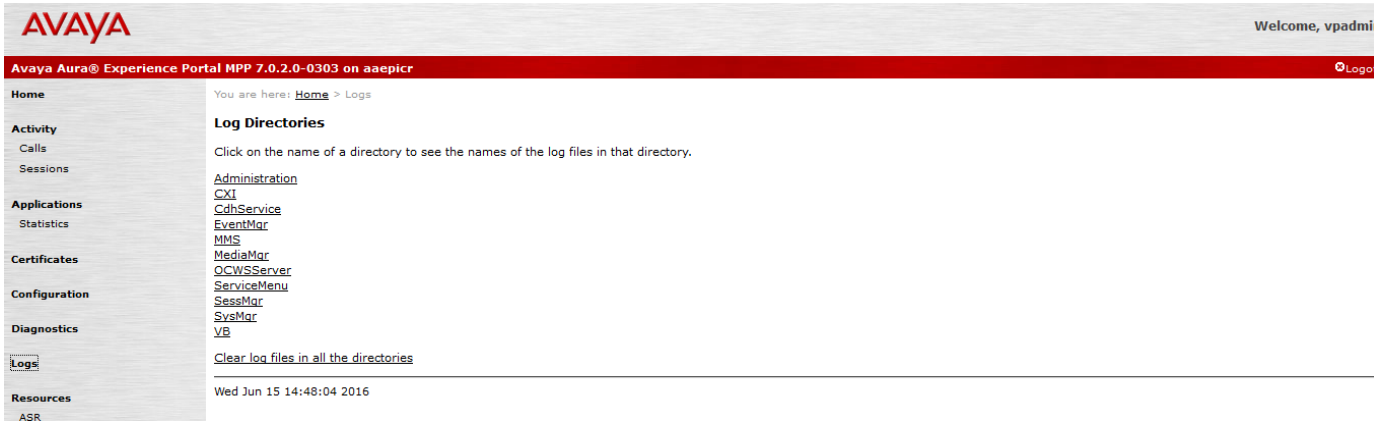
All the messages related to the extension, in this log file, are identified by the tag [DR3 EXTENSION], making it easier to identify meaningful messages when debugging a call flow.

CXI

CXI, available in AAEPMS, can also be used to understand the whole call flow and, also, how and when dialogs are created, finished, and how CCA communicates to other components such as the ICR Core and MPP. The CXI logs can be found under AAEPMS » Real-time Monitoring » System Monitor. In this page, select the AAEP server you want to see the logs, like in the picture below:



In the page opened when you click the link indicated by the red arrow above, click on the link called **Service Menu**. In the new page displayed, select the item **Logs** in the side menu and, finally, **CXI**.



The log files we are interested on usually follow the pattern '*CCXML-SessionSlot-<session_number>.log*' in their name. The block below shows some messages from one of these files.

CCXML-SessionSlot-016.log

```
@2016-07-12
14:40:18,701||FINE|CXI|18730|Session=aaepicr-2016194184018-18|########
(Thread: -190842000) Constructing Session: 16786252|aaepicr####
@2016-07-12
14:40:18,714||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
SessionStarted message to SessionManager.|aaepicr####
@2016-07-12
14:40:18,714||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: SessionStarted & size: 71|aaepicr####
@2016-07-12
14:40:18,714||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::Run) Fetching Url :
http://135.122.99.97:6080/ICRCCApp/jsp/start.jsp?%5f%5fVPapploggingurl=htt
ps%3a%2f%2f135%2e122%2e99%2e92%2faxis%2fservices%2fVPReport4&%5f%5fVPapplo
ggingurl%5f2=https%3a%2f%2f135%2e122%2e99%2e97%2faxis%2fservices%2fVPRepor
t4&%5f%5fVPapplog=%2faxis2%2fservices%2fVPAppLogService&%5f%5fVPvpms=135%2
e122%2e99%2e92%7c135%2e122%2e99%2e97&%5f%5fVPappvars=%2faxis2%2fservices%2
fVPAppVarsService&%5f%5fVPVarAppDate=0&%5f%5fVPVarAppURL=https%3a%2f%2f135
%2e122%2e99%2e92%2faxis%2fservices%2fVPAppRuntimeVars&%5f%5fVPVarAppURL2=h
ttps%3a%2f%2f135%2e122%2e99%2e97%2faxis%2fservices%2fVPReport4&%5f%5fVPVar
GlobalDate=1468348554038&%5f%5fVPbreadcrumb=disabled&%5f%5fVPmaxbackuplogf
iles=10&%5f%5fVPlogname=%25default%256BwWYXmzTKa2OIei&%5f%5fVPlogpassword=
CTeYQuZOnPsJrlgBfgUHJyha%2fwvAJ%2bor%2fHdbYCTuCtKIoIesHMsJuszkk0a%2f1BtX&%
5f%5fVPappname=0%3aCCA&session%5f%5f%5fsessionid=aaepicr%2d2016194184018%2
d18|aaepicr####
@2016-07-12
14:40:18,715||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::script_element)
<uri='../jscript/functions.js'>|aaepicr####
@2016-07-12
14:40:18,715||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::script_element) Expanded relative uri to
'http://135.122.99.97:6080/ICRCCApp/jsp/../jscript/functions.js'.|aaepicr#
###
@2016-07-12
```

```
14:40:18,716||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
CallAlertingAck message to SessionManager.|aaepicr####
@2016-07-12
14:40:18,717||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: CallAlertingAck & size: 48|aaepicr####
@2016-07-12
14:40:18,721||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.alerting message to CXI.|aaepicr####
@2016-07-12
14:40:18,721||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.signal message to CXI.|aaepicr####
@2016-07-12
14:40:18,724||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.sip', expr='new
Object()'>|aaepicr####
@2016-07-12
14:40:18,724||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.sip.unknownhdr',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.connectionid',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.callerid',
expr=''unknown''>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ucid',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.refer',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.skillid',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa', expr='new
Object()'>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa.appname',
expr='ssaObject.appName'>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa.dialogid',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,725||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
```

```
Diagnostic: (Interpreter::assign_element) <name='client.ssa.result',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa.starttime',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa.endtime',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.vdn',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.vdnuri',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.acd',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.acdtype',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='agent.ccxmlapplicationurl', expr='0'>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.ccxmlfetchid',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,726||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.connectionid',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,727||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.prov_timeout',
expr='provtimeout'>|aaepicr####
@2016-07-12
14:40:18,727||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.timeout',
expr='agenttimeout'>|aaepicr####
@2016-07-12
14:40:18,727||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.AAI',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,727||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.failurereason',
```

```
expr=''''>|aaepicr####
@2016-07-12
14:40:18,727||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.sendid',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,727||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.skilldefaultvdn',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,727||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.mergeonconnect',
expr='mergeonconnect'>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.ringingReceived',
expr=''false''>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.siprespcode',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta', expr='new
Object()'>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.appname',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.interrupt',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.announcement',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.dialogid',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.qs',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,728||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.optout',
expr=''false''>|aaepicr####
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.optoutvdn',
expr=''''>|aaepicr####
```

```
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.starttime',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.endtime',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.sendid',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wtadefault',
expr='new Object()'>|aaepicr####
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.wtadefault.appname', expr=''''>|aaepicr####
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.wtadefault.isinterrupt', expr=''''>|aaepicr####
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.wtadefault.isannouncement', expr=''''>|aaepicr####
@2016-07-12
14:40:18,729||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.eha', expr='new
Object'>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.eha.generic',
expr='new Object'>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.eha.generic.appname',
expr='genericEhaObject.appName'>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.eha.generic.dialogid', expr=''''>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.eha.generic.defaultvdn',
expr='genericEhaObject.defaultvdn'>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
```

```
<name='client.eha.generic.isexplicit', expr=''false''>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.eha.nonbusi',
expr='new Object'>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.eha.nonbusi.appname',
expr='nonBusEhaObject.appName'>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.eha.nonbusi.dialogid', expr=''''>|aaepicr####
@2016-07-12
14:40:18,730||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.eha.nonbusi.defaultvdn',
expr='nonBusEhaObject.defaultvdn'>|aaepicr####
@2016-07-12
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.eha.qs',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.cause0',
expr=''Avaya-cc-reason; cause=0; text=\u0022Normal\u0022''>|aaepicr####
@2016-07-12
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.cause1',
expr=''Avaya-cc-reason; cause=1; text=\u0022Resources not
available\u0022''>|aaepicr####
@2016-07-12
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.cause2',
expr=''Avaya-cc-reason; cause=2; text=\u0022Wait time increased
drastically\u0022''>|aaepicr####
@2016-07-12
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.cause3',
expr=''Avaya-cc-reason; cause=3; text=\u0022Caller receiving
treatment\u0022''>|aaepicr####
@2016-07-12
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.cause4',
expr=''Avaya-cc-reason; cause=4; text=\u0022Network failure
recovery\u0022''>|aaepicr####
@2016-07-12
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.cause5',
expr=''Avaya-cc-reason; cause=5; text=\u0022VP failure
recovery\u0022''>|aaepicr####
@2016-07-12
```

```
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.cause6',
expr=''Avaya-cc-reason; cause=6; text=\u0022Cancelled due to caller
interaction\u0022''>|aaepicr####
@2016-07-12
14:40:18,731||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.value',
expr='reasoncodes.cause0'>|aaepicr####
@2016-07-12
14:40:18,732||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.text0',
expr=''%22Normal%22''>|aaepicr####
@2016-07-12
14:40:18,732||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.text1',
expr=''%22Resources%20not%20available%22''>|aaepicr####
@2016-07-12
14:40:18,732||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.text2',
expr=''%22Wait%20time%20increased%20drastically%22''>|aaepicr####
@2016-07-12
14:40:18,732||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.text3',
expr=''%22Caller%20receiving%20treatment%22''>|aaepicr####
@2016-07-12
14:40:18,732||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.text4',
expr=''%22Network%20failure%20recovery%22''>|aaepicr####
@2016-07-12
14:40:18,732||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.text5',
expr=''%22VP%20failure%20recovery%22''>|aaepicr####
@2016-07-12
14:40:18,732||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.text6',
expr=''%22Cancelled%20due%20to%20caller%20interaction%22''>|aaepicr####
@2016-07-12
14:40:18,732||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.value',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.prefix',
expr=''Reason=SIP%3Bcause%3D480%3Btext%3D%22Temporarily%20Unavailable%22&R
eason=Redirection%3Bcause%3DNORMAL%3Bavaya-cm-reason%3D''>|aaepicr####
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.sessionid',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.ucid',
expr=''''>|aaepicr####
```

```
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.callingno',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.appname',
expr='encodeURI(appObject.appName)'>|aaepicr####
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.skillid',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='queuedCallRTD.destination', expr=''''>|aaepicr####
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='queuedCallRTD.callcentername', expr=''''>|aaepicr####
@2016-07-12
14:40:18,733||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.ewt',
expr='-1'>|aaepicr####
@2016-07-12
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.qpos',
expr='-1'>|aaepicr####
@2016-07-12
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.sessionid',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.ucid',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.callingno',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.calledno',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.appname',
expr='encodeURI(appObject.appName)'>|aaepicr####
@2016-07-12
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.apptype',
expr=''1''>|aaepicr####
@2016-07-12
```

```
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.ssaappname',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,734||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.wtappname',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,735||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.ehappname',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,735||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.insert',
expr=''true''>|aaepicr####
@2016-07-12
14:40:18,735||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.exitreason',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,735||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.apptype',
expr=''ICRCCA''>|aaepicr####
@2016-07-12
14:40:18,735||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.destination',
expr=''''>|aaepicr####
@2016-07-12
14:40:18,735||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='SDRData.requeuereasonlist', expr=''''>|aaepicr####
@2016-07-12
14:40:18,735||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='SDRData.maxewtdestination', expr=''''>|aaepicr####
@2016-07-12
14:40:18,735||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='SDRData.maxqposdestination', expr=''''>|aaepicr####
@2016-07-12
14:40:18,736||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.maxewt',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,736||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.maxqpos',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,736||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.ssaduration',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,736||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
```

```
Diagnostic: (Interpreter::assign_element) <name='SDRData.wtaduration',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,736||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.ccaduration',
expr='(new Date()).getTime()'>|aaepicr####
@2016-07-12
14:40:18,736||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.wtadurationtemp',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,736||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.requeuecount',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,736||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='SDRData.requeueseparatorcount', expr='3'>|aaepicr####
@2016-07-12
14:40:18,737||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 23
milliseconds|aaepicr####
@2016-07-12
14:40:18,737||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::!!!!!!!!!!!!!!!!Loop DOUBTED!!!!!!!!!!!!) loop
count is 1|aaepicr####
@2016-07-12
14:40:18,737||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::missed) event 'ccxml.loaded'|aaepicr####
@2016-07-12
14:40:18,737||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 16
milliseconds|aaepicr####
@2016-07-12
14:40:18,738||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.alerting] State:[init] Inside connection alerting
event.'>|aaepicr####
@2016-07-12
14:40:18,738||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 125|aaepicr####
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa.dialogid',
expr='0'>|aaepicr####
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.connectionid',
expr='event$.connectionid'>|aaepicr####
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.callerid',
expr='event$.connection.remote'>|aaepicr####
```

```
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ucid',
expr='ucid_prefix_service_provider + HexEncodeUCID(
event$.connection.avaya.ucid )'>|aaepicr####
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='mode', expr=''service
provider''>|aaepicr####
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.AAI',
expr='client.ucid'>|aaepicr####
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.AAI',
expr='uui_prefix + agent.AAI'>|aaepicr####
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.AAI',
expr='agent.AAI + uui_sufix'>|aaepicr####
@2016-07-12
14:40:18,739||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.ucid',
expr='event$.connection.avaya.ucid'>|aaepicr####
@2016-07-12
14:40:18,740||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.ucid',
expr='event$.connection.avaya.ucid'>|aaepicr####
@2016-07-12
14:40:18,740||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.sessionid',
expr='event$.eventsource'>|aaepicr####
@2016-07-12
14:40:18,740||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.callingno',
expr='event$.connection.remote'>|aaepicr####
@2016-07-12
14:40:18,740||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.callingno',
expr='encodeURI(event$.connection.protocol.sip.from.uri)'>|aaepicr####
@2016-07-12
14:40:18,740||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.sessionid',
expr='event$.eventsource'>|aaepicr####
@2016-07-12
14:40:18,740||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.callingno',
expr='event$.connection.remote'>|aaepicr####
@2016-07-12
14:40:18,740||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.callingno',
expr='encodeURI(event$.connection.protocol.sip.from.uri)'>|aaepicr####
```

```
@2016-07-12
14:40:18,740||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.calledno',
expr='event$.connection.local'>|aaepicr####
@2016-07-12
14:40:18,741||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.calledno',
expr='encodeURI(event$.connection.protocol.sip.to.uri)'>|aaepicr####
@2016-07-12
14:40:18,741||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='state',
expr=''fetching''>|aaepicr####
@2016-07-12
14:40:18,741||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.alerting] State:[fetching] Fetching encoded CCXML.
URL is
:|aaepicr####'>http://135.122.99.97:6080/ICRCCApp/jsp/eccxml.jsp'>|aaepicr
####
@2016-07-12
14:40:18,741||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 177|aaepicr####
@2016-07-12
14:40:18,741||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::fetch_element)
<next='http://135.122.99.97:6080/ICRCCApp/jsp/eccxml.jsp',
type='application/ccxml+xml', method='GET', timeout='15s', namelist='',
fetchid='', synch='', maxage='0' , maxstale='0' ,
enctype='application/x-www-form-urlencoded'>|aaepicr####
@2016-07-12
14:40:18,741||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 20
milliseconds|aaepicr####
@2016-07-12
14:40:18,742||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::missed) event 'connection.signal'|aaepicr####
@2016-07-12
14:40:18,768||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 22
milliseconds|aaepicr####
@2016-07-12
14:40:18,796||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[fetch.done]
State:[fetching] Fetched encoded CCXML successfully.'>|aaepicr####
@2016-07-12
14:40:18,796||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 122|aaepicr####
@2016-07-12
14:40:18,796||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='state',
expr=''init''>|aaepicr####
@2016-07-12
14:40:18,796||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
```

```
Diagnostic: (Interpreter::accept_element) <connectionid='33563469'
hints=''>|aaepicr####
@2016-07-12
14:40:18,796||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
CallAccept message to SessionManager.|aaepicr####
@2016-07-12
14:40:18,796||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: CallAccept & size: 58|aaepicr####
@2016-07-12
14:40:18,797||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::goto_element)
<fetchid='83895120aaepicr-2016194184018-18'>|aaepicr####
@2016-07-12
14:40:18,797||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:18,797||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[ccxml.loaded] State:[init] Missed event. '>|aaepicr####
@2016-07-12
14:40:18,797||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 99|aaepicr####
@2016-07-12
14:40:19,269||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Received
CallConnected message for a connection; No cpa results, setting
classification to empty|aaepicr####
@2016-07-12
14:40:19,270||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.connected message to CXI.|aaepicr####
@2016-07-12
14:40:19,270||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:19,270||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.connected] State:[init] Inside encoded
CCXML.'>|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 114|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appStage',
expr=''ssa''>|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='state',
expr=''ssa''>|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa.dialogid',
```

```
expr='event$.dialogid'>|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='dialog_id',
expr='client.ssa.dialogid'>|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.connected] State:[ssa] Launching SSA. App name is :
SSA'>|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.ssaappname',
expr='encodeURI(client.ssa.appname)'>|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::dialogstart_element) <connectionid='33563469'
conferenceid='' prepareddialogid='' src='app://0:CCA...SSA'
type='application/voicexml+xml' namelist=''
parameters='',dialogid='dialog_id'maxage='0' maxstale='0'
enctype='application/x-www-form-urlencoded' mediadirection='both'
method='get' hints=''>|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 124|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
DialogPrepare message to SessionManager.|aaepicr####
@2016-07-12
14:40:19,271||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: DialogPrepare & size: 158|aaepicr####
@2016-07-12
14:40:19,395||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
DialogStart message to SessionManager.|aaepicr####
@2016-07-12
14:40:19,395||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: DialogStart & size: 56|aaepicr####
@2016-07-12
14:40:19,396||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
dialog.started message to CXI.|aaepicr####
@2016-07-12
14:40:19,396||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:19,397||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[dialog.started] State:[ssa] Inside
dialog.started.'>|aaepicr####
@2016-07-12
14:40:19,397||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 108|aaepicr####
@2016-07-12
```

```
14:40:19,397||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.ssaduration',
expr='(new Date()).getTime()'>|aaepicr####
@2016-07-12
14:40:19,397||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.apptype',
expr=''1''>|aaepicr####
@2016-07-12
14:40:19,397||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::fetch_element)
<next='http://localhost/mpp/config/addAppCallDetails.php?sessionid=aaepicr
-2016194184018-18&ucid=10000003671468348818&appname=0:CCA&apptype=1&ssaapp
name=SSA&wtaappname=&ehaappname=&callingno=sip:64500@edp.avaya.com&calledn
o=sip:64600@edp.avaya.com&insert=true', type='text/ecmascript',
method='GET', timeout='15s', namelist='', fetchid='', synch='', maxage='0'
, maxstale='0' , enctype='application/x-www-form-urlencoded'>|aaepicr####
@2016-07-12
14:40:19,397||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.insert',
expr=''false''>|aaepicr####
@2016-07-12
14:40:19,403||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:19,403||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::script_element)
<fetchid='83895125aaepicr-2016194184018-18'>|aaepicr####
@2016-07-12
14:40:19,403||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[fetch.done]
State:[ssa] Reporting data action [addAppCallDetails] performed
successfully.'>|aaepicr####
@2016-07-12
14:40:19,403||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 147|aaepicr####
@2016-07-12
14:40:19,545||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.signal message to CXI.|aaepicr####
@2016-07-12
14:40:19,545||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.signal message to CXI.|aaepicr####
@2016-07-12
14:40:19,545||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:19,546||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.signal] State:[ssa] Missed event. '>|aaepicr####
@2016-07-12
14:40:19,546||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 103|aaepicr####
```

```
@2016-07-12
14:40:19,546||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:19,547||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::!!!!!!!!!!!!!!!!Loop DOUBTED!!!!!!!!!!!!) loop
count is 1|aaepicr####
@2016-07-12
14:40:19,547||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.signal] State:[ssa] Missed event. '>|aaepicr####
@2016-07-12
14:40:19,547||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 103|aaepicr####
@2016-07-12
14:40:19,793||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.signal message to CXI.|aaepicr####
@2016-07-12
14:40:19,793||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:19,794||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::!!!!!!!!!!!!!!!!Loop DOUBTED!!!!!!!!!!!!) loop
count is 2|aaepicr####
@2016-07-12
14:40:19,794||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.signal] State:[ssa] Missed event. '>|aaepicr####
@2016-07-12
14:40:19,794||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 103|aaepicr####
@2016-07-12
14:40:19,975||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
dialog.exit message to CXI.|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.ssaduration',
expr='Math.floor((((new Date()).getTime()) -
SDRData.ssaduration)/1000)'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] SSA exited.'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='dialog_id',
expr='0'>|aaepicr####
```

```
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] Result returned by SSA is : AGENT_AVAILABLE'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa.result',
expr='(event$.values.ssaResult).toLowerCase()'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] VDN returned by SSA is : 64711@edp.avaya.com'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.vdn',
expr='event$.values.vdn'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.vdnuri',
expr='event$.values.vdn'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] Default VDN returned by SSA is :
64710@edp.avaya.com'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.skilldefaultvdn',
expr='event$.values.skilldefaultvdn'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] ACD type returned by SSA is : Avaya-CM'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.acdtype',
expr='event$.values.acdtype'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.appname',
expr='client.wtadefault.appname'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.isinterrupt',
expr='client.wtadefault.isinterrupt'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='client.wta.isannouncement',
expr='client.wtadefault.isannouncement'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.AAI',
```

```
expr='event$.values.AAI'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='AAI',
expr='agent.AAI'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] UUI in mode [service provider] is : '>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.refer',
expr='event$.values.refer'>|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 94|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 126|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 127|aaepicr####
@2016-07-12
14:40:19,978||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 135|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 121|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 119|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] Enable External Message returned by SSA is :
false'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 133|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] EWT returned by SSA is : -1'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 110|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='ewt', expr='1 *
event$.values.ewt'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='ssa_ret_ewt', expr='1 *
```

```
event$.values.ewt'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='prev_ewt', expr='1 *
event$.values.ewt'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.skillid',
expr='event$.values.skillId'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='skillId',
expr='event$.values.skillId'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] On Requeue Use Extension returned by SSA is :
true'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 133|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='onRequeueUseExtension',
expr='true'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[ssa] Caller context is returned by SSA.'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='callercontext',
expr='event$.values.callercontext'>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 117|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='state',
expr=''calling_agent''>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::send_element) <target='aaepicr-2016194184018-18',
targetype='ccxml', sendid='100672347', delay='0s', name='start_wta',
namelist='', hints=''>|aaepicr####
@2016-07-12
14:40:19,979||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
start_wta message to CXI.|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
```

```
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::DefaultEventProcessor) putting send.successful
with sendid (100672347) to queue|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[start_wta]
State:[calling_agent] Launching WTA. App name is : WTA'>|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appStage',
expr=''wta''>|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.dialogid',
expr='0'>|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='dialog_id',
expr='client.wta.dialogid'>|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.wtadurationtemp',
expr='(new Date()).getTime()'>|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.wtappname',
expr='encodeURI(client.wta.appname)'>|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::dialogstart_element) <connectionid='33563469'
conferenceid='' prepareddialogid='' src='app://0:CCA...WTA'
type='application/voicexml+xml' namelist='ewt queueposition qs AAI skillId'
parameters='ewt queueposition qs AAI
skillId',dialogid='dialog_id'maxage='0' maxstale='0'
enctype='application/x-www-form-urlencoded' mediadirection='both'
method='get' hints=''>|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:19,980||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[send.successful] State:[calling_agent] Missed event.
'>|aaepicr####
@2016-07-12
14:40:19,981||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 123|aaepicr####
@2016-07-12
14:40:19,981||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
DialogPrepare message to SessionManager.|aaepicr####
@2016-07-12
14:40:19,981||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
```

```
message of type: DialogPrepare & size: 307|aaepicr####
@2016-07-12
14:40:19,981||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 111|aaepicr####
@2016-07-12
14:40:20,001||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
DialogStart message to SessionManager.|aaepicr####
@2016-07-12
14:40:20,001||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: DialogStart & size: 56|aaepicr####
@2016-07-12
14:40:20,002||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
dialog.started message to CXI.|aaepicr####
@2016-07-12
14:40:20,002||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:20,002||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[dialog.started] State:[calling_agent] Inside
dialog.started.'>|aaepicr####
@2016-07-12
14:40:20,003||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 118|aaepicr####
@2016-07-12
14:40:20,003||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[dialog.started] State:[calling_agent] Initiating no media call
to destination : 64711@edp.avaya.com'>|aaepicr####
@2016-07-12
14:40:20,003||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 157|aaepicr####
@2016-07-12
14:40:20,003||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::createcall_element)
<connectionid='agent.connectionid', dest='sip:64711@edp.avaya.com',
timeout='', aai='', callerid='64500', hints='{MergeTimeout:agent.timeout,
DestinationURI:agent.vdnuri, contact:agent.contact, AAI:agent.AAI,
sip:hints.sip}' ,joinid='', joindirection='both' >|aaepicr####
@2016-07-12
14:40:20,004||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
CallCreate message to SessionManager.|aaepicr####
@2016-07-12
14:40:20,004||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: CallCreate & size: 246|aaepicr####
@2016-07-12
14:40:20,004||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.skillid',
expr='client.skillid'>|aaepicr####
@2016-07-12
14:40:20,004||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
```

```
<name='queuedCallRTD.destination', expr='agent.vdn'>|aaepicr####
@2016-07-12
14:40:20,004||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='queuedCallRTD.callcentername',
expr='encodeURI(agent.acd)'>|aaepicr####
@2016-07-12
14:40:20,005||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::fetch_element)
<next='http://localhost/mpp/config/addQueuedCallDetails.php?sessionid=aaep
icr-2016194184018-18&ucid=10000003671468348818&skillid=1&appname=0:CCA&des
tination=64711@edp.avaya.com&callcentername=&qpos=-1&ewt=-1&callingno=sip:
64500@edp.avaya.com', type='text/ecmascript', method='GET', timeout='15s',
namelist='', fetchid='', synch='', maxage='0' , maxstale='0' ,
enctype='application/x-www-form-urlencoded'>|aaepicr####
@2016-07-12
14:40:20,005||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.apptype',
expr=''2''>|aaepicr####
@2016-07-12
14:40:20,006||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::fetch_element)
<next='http://localhost/mpp/config/addAppCallDetails.php?sessionid=aaepicr
-2016194184018-18&ucid=10000003671468348818&appname=0:CCA&apptype=2&ssaapp
name=SSA&wtaappname=WTA&ehaappname=&callingno=sip:64500@edp.avaya.com&call
edno=sip:64600@edp.avaya.com&insert=false', type='text/ecmascript',
method='GET', timeout='15s', namelist='', fetchid='', synch='', maxage='0'
, maxstale='0' , enctype='application/x-www-form-urlencoded'>|aaepicr####
@2016-07-12
14:40:20,006||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Received
CallConnected message for a connection; No cpa results, setting
classification to empty|aaepicr####
@2016-07-12
14:40:20,006||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.progressing message to CXI.|aaepicr####
@2016-07-12
14:40:20,007||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.insert',
expr=''false''>|aaepicr####
@2016-07-12
14:40:20,007||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 2
milliseconds|aaepicr####
@2016-07-12
14:40:20,010||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:20,010||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::script_element)
<fetchid='83895135aaepicr-2016194184018-18'>|aaepicr####
@2016-07-12
14:40:20,010||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
```

```
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[fetch.done]
State:[calling_agent] Reporting data action [addQueuedCallDetails]
performed successfully.'>|aaepicr####
@2016-07-12
14:40:20,010||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 160|aaepicr####
@2016-07-12
14:40:20,010||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:20,011||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::!!!!!!!!!!!!!!!!Loop DOUBTED!!!!!!!!!!!!) loop
count is 1|aaepicr####
@2016-07-12
14:40:20,011||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::script_element)
<fetchid='83895136aaepicr-2016194184018-18'>|aaepicr####
@2016-07-12
14:40:20,011||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[fetch.done]
State:[calling_agent] Reporting data action [addAppCallDetails] performed
successfully.'>|aaepicr####
@2016-07-12
14:40:20,011||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 157|aaepicr####
@2016-07-12
14:40:20,080||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.signal message to CXI.|aaepicr####
@2016-07-12
14:40:20,081||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:20,081||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.signal] State:[calling_agent] Missed event.
'>|aaepicr####
@2016-07-12
14:40:20,081||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 113|aaepicr####
@2016-07-12
14:40:20,244||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Received
CallConnected message for a connection; No cpa results, setting
classification to empty|aaepicr####
@2016-07-12
14:40:20,245||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.progressing message to CXI.|aaepicr####
@2016-07-12
14:40:20,246||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
```

```
14:40:20,246||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] Received
connection.progressing event for response code 100'>|aaepicr####
@2016-07-12
14:40:20,246||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 163|aaepicr####
@2016-07-12
14:40:20,246||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::send_element) <target='aaepicr-2016194184018-18',
targetype='ccxml', sendid='100672358', delay='90s',
name='provisional_resp_timeout', namelist='', hints=''>|aaepicr####
@2016-07-12
14:40:20,246||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
provisional_resp_timeout message to CXI.|aaepicr####
@2016-07-12
14:40:20,476||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Received
CallConnected message for a connection; No cpa results, setting
classification to empty|aaepicr####
@2016-07-12
14:40:20,477||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
connection.progressing message to CXI.|aaepicr####
@2016-07-12
14:40:20,478||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:20,478||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::!!!!!!!!!!!!!!!!Loop DOUBTED!!!!!!!!!!!!!) loop
count is 1|aaepicr####
@2016-07-12
14:40:20,478||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] Received 182
response on the no media call.'>|aaepicr####
@2016-07-12
14:40:20,478||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 147|aaepicr####
@2016-07-12
14:40:20,478||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] Cancelling
provisional timer.'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 133|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::cancel_element) <sendid =
'100672358'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Attempting to
cancel event with the id: <100672358>|aaepicr####
```

```
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
cancel.successful message to CXI.|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.sendid',
expr=''0''>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::send_element) <target='aaepicr-2016194184018-18',
targetype='ccxml', sendid='100672360', delay='90s',
name='provisional_resp_timeout', namelist='', hints=''>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
provisional_resp_timeout message to CXI.|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] History Info
header is not present in 182 response.'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 155|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] Sip Response
Text (avaya-cm-data) is : Queued,
avaya-cm-data=0001FFFEFFFE008C'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 181|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queueposition',
expr='returnQueuePos(event$.connection.protocol.sip.resptext)'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='ewt',
expr='returnEWT(event$.connection.protocol.sip.resptext)'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.maxewt',
expr='ewt'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='SDRData.maxewtdestination', expr='agent.vdn'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.maxqpos',
expr='queueposition'>|aaepicr####
@2016-07-12
```

```
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='SDRData.maxqposdestination', expr='agent.vdn'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='prev_queueposition',
expr='queueposition'>|aaepicr####
@2016-07-12
14:40:20,479||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.ewt',
expr='ewt'>|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.qpos',
expr='queueposition'>|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::fetch_element)
<next='http://localhost/mpp/config/updateQueuedCallDetails.php?sessionid=a
aepicr-2016194184018-18&ucid=10000003671468348818&skillid=1&appname=0:CCA&
destination=64711@edp.avaya.com&callcentername=&qpos=1&ewt=65534&callingno
=sip:64500@edp.avaya.com', type='text/ecmascript', method='GET',
timeout='15s', namelist='', fetchid='', synch='', maxage='0' , maxstale='0'
, enctype='application/x-www-form-urlencoded'>|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] Previous EWT is
: -1. Updated EWT is : 65534'>|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 148|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] Previous Queue
Position is : 1. Updated Queue Position is : 1'>|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 165|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] EWT or Queue
Position has spiked.'>|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='is_spiked',
expr='true'>|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='wtaIsNotPresent',
expr='true'>|aaepicr####
```

```
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.appname',
expr='client.wtadefault.appname'>|aaepicr####
@2016-07-12
14:40:20,481||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 137|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.interrupt',
expr='client.wtadefault.isinterrupt'>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.announcement',
expr='client.wtadefault.isannouncement'>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[calling_agent] WTA is not
configured for EWT : 65534.Default WTA will be launched. WTA name is :
WTA'>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 189|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='reasoncodes.value',
expr='reasoncodes.cause2'>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.value',
expr='historyinfotext.text2'>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='SDRData.requeuereasonlist', expr='SDRData.requeuereasonlist +
'EWT_SPIKE#''>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.requeuecount',
expr='SDRData.requeuecount + 1 '>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='terminatereasonvalue',
expr=''EWT_SPIKE''>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::send_element) <target='aaepicr-2016194184018-18',
targetype='ccxml', sendid='100672362', delay='2s', name='request_icrcore',
namelist='', hints=''>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
request_icrcore message to CXI.|aaepicr####
```

```
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='prev_ewt',
expr='ewt'>|aaepicr####
@2016-07-12
14:40:20,482||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='prev_queueposition',
expr='queueposition'>|aaepicr####
@2016-07-12
14:40:20,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='iswta_terminated',
expr='true'>|aaepicr####
@2016-07-12
14:40:20,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='state',
expr=''call_queued''>|aaepicr####
@2016-07-12
14:40:20,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[connection.progressing] State:[call_queued] Terminating
running WTA due to increase in EWT or Queue Position. Reason :
EWT_SPIKE'>|aaepicr####
@2016-07-12
14:40:20,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 186|aaepicr####
@2016-07-12
14:40:20,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::dialogterminate_element) <dialogid='67117916',
immediate='false',hints='{TerminateReason:terminatereasonvalue}'>|aaepicr#
###
@2016-07-12
14:40:20,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
DialogTerminate message to SessionManager.|aaepicr####
@2016-07-12
14:40:20,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: DialogTerminate & size: 58|aaepicr####
@2016-07-12
14:40:20,484||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 2
milliseconds|aaepicr####
@2016-07-12
14:40:20,484||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[cancel.successful] State:[call_queued] Missed event.
'>|aaepicr####
@2016-07-12
14:40:20,484||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 111|aaepicr####
@2016-07-12
14:40:20,485||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
```

```
14:40:20,486||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::script_element)
<fetchid='83895145aaepicr-2016194184018-18'>|aaepicr####
@2016-07-12
14:40:20,486||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[fetch.done]
State:[call_queued] Reporting data action [updateQueuedCallDetails]
performed successfully.'>|aaepicr####
@2016-07-12
14:40:20,486||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 161|aaepicr####
@2016-07-12
14:40:20,594||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
dialog.exit message to CXI.|aaepicr####
@2016-07-12
14:40:20,594||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:20,594||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.wtadurationtemp',
expr='Math.floor((((new Date()).getTime()) -
SDRData.wtadurationtemp)/1000)'>|aaepicr####
@2016-07-12
14:40:20,594||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='SDRData.wtaduration',
expr='SDRData.wtaduration + SDRData.wtadurationtemp'>|aaepicr####
@2016-07-12
14:40:20,594||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[call_queued] WTA exited.'>|aaepicr####
@2016-07-12
14:40:20,595||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 102|aaepicr####
@2016-07-12
14:40:20,595||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[call_queued] AAI is not updated by WTA.'>|aaepicr####
@2016-07-12
14:40:20,595||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 117|aaepicr####
@2016-07-12
14:40:20,595||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='updatedAAI',
expr='false'>|aaepicr####
@2016-07-12
14:40:20,595||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.qs',
expr='event$.values.querystring'>|aaepicr####
@2016-07-12
14:40:20,595||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='qs',
expr='event$.values.querystring'>|aaepicr####
```

```
@2016-07-12
14:40:20,595||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='enableExternalMessage',
expr='false'>|aaepicr####
@2016-07-12
14:40:20,596||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[dialog.exit]
State:[call_queued] Launching WTA with updated queue position and
EWT.'>|aaepicr####
@2016-07-12
14:40:20,596||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 141|aaepicr####
@2016-07-12
14:40:20,596||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::send_element) <target='aaepicr-2016194184018-18',
targetype='ccxml', sendid='100672365', delay='0s', name='start_wta',
namelist='', hints=''>|aaepicr####
@2016-07-12
14:40:20,596||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
start_wta message to CXI.|aaepicr####
@2016-07-12
14:40:20,596||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='iswta_terminated',
expr='false'>|aaepicr####
@2016-07-12
14:40:20,596||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 1
milliseconds|aaepicr####
@2016-07-12
14:40:20,596||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::DefaultEventProcessor) putting send.successful
with sendid (100672365) to queue|aaepicr####
@2016-07-12
14:40:20,596||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[start_wta]
State:[call_queued] Launching WTA. App name is : WTA'>|aaepicr####
@2016-07-12
14:40:20,597||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 121|aaepicr####
@2016-07-12
14:40:20,597||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appStage',
expr=''wta''>|aaepicr####
@2016-07-12
14:40:20,597||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.wta.dialogid',
expr='0'>|aaepicr####
@2016-07-12
14:40:20,597||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='dialog_id',
expr='client.wta.dialogid'>|aaepicr####
@2016-07-12
14:40:20,597||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
```

```
Diagnostic: (Interpreter::assign_element) <name='SDRData.wtadurationtemp',
expr='(new Date()).getTime()'>|aaepicr####
@2016-07-12
14:40:20,597||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.wtappname',
expr='encodeURI(client.wta.appname)'>|aaepicr####
@2016-07-12
14:40:20,597||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::dialogstart_element) <connectionid='33563469'
conferenceid='' prepareddialogid='' src='app://0:CCA...WTA'
type='application/voicexml+xml' namelist='ewt queueposition qs AAI skillId'
parameters='ewt queueposition qs AAI
skillId',dialogid='dialog_id'maxage='0' maxstale='0'
enctype='application/x-www-form-urlencoded' mediadirection='both'
method='get' hints=''>|aaepicr####
@2016-07-12
14:40:20,598||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
DialogPrepare message to SessionManager.|aaepicr####
@2016-07-12
14:40:20,598||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: DialogPrepare & size: 313|aaepicr####
@2016-07-12
14:40:20,598||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 1
milliseconds|aaepicr####
@2016-07-12
14:40:20,598||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[send.successful] State:[call_queued] Missed event.
'>|aaepicr####
@2016-07-12
14:40:20,598||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 109|aaepicr####
@2016-07-12
14:40:20,616||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
DialogStart message to SessionManager.|aaepicr####
@2016-07-12
14:40:20,617||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: DialogStart & size: 56|aaepicr####
@2016-07-12
14:40:20,618||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
dialog.started message to CXI.|aaepicr####
@2016-07-12
14:40:20,618||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:20,619||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[dialog.started] State:[call_queued] Inside
dialog.started.'>|aaepicr####
@2016-07-12
14:40:20,619||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
```

```
Diagnostic: (Interpreter::assign_element) <name='appRTD.apptype',
expr=''2''>|aaepicr####
@2016-07-12
14:40:20,619||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::fetch_element)
<next='http://localhost/mpp/config/addAppCallDetails.php?sessionid=aaepicr
-2016194184018-18&ucid=10000003671468348818&appname=0:CCA&apptype=2&ssaapp
name=SSA&wtaappname=WTA&ehaappname=&callingno=sip:64500@edp.avaya.com&call
edno=sip:64600@edp.avaya.com&insert=false', type='text/ecmascript',
method='GET', timeout='15s', namelist='', fetchid='', synch='', maxage='0'
, maxstale='0' , enctype='application/x-www-form-urlencoded'>|aaepicr####
@2016-07-12
14:40:20,619||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='appRTD.insert',
expr=''false''>|aaepicr####
@2016-07-12
14:40:20,619||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 116|aaepicr####
@2016-07-12
14:40:20,623||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:20,623||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::script_element)
<fetchid='83895152aaepicr-2016194184018-18'>|aaepicr####
@2016-07-12
14:40:20,623||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[fetch.done]
State:[call_queued] Reporting data action [addAppCallDetails] performed
successfully.'>|aaepicr####
@2016-07-12
14:40:20,623||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 155|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 2 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::DefaultEventProcessor) putting send.successful
with sendid (100672362) to queue|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[request_icrcore] State:[call_queued] Fetching information for
re-queuing the call.'>|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='icrcore_sendid',
expr='0'>|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
```

```
Diagnostic: (Interpreter::assign_element) <name='icrcore_vdn',
expr=''''>|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='icrcore_defaultvdn',
expr=''''>|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='icrcore_ewt',
expr='0'>|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 140|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='icrcore_result',
expr=''''>|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[request_icrcore] State:[call_queued] EWT is INFINITY.
Enforcing information from ICRCore.'>|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 147|aaepicr####
@2016-07-12
14:40:22,483||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::fetch_element)
<next='http://135.122.99.97:6080/ICRCCApp/jsp/requesticrcore.jsp?skillid=1
&sessionid=aaepicr-2016194184018-18&enforce=true&appname=0:CCA&onRequeueUs
eExtension=true', type='text/ecmascript', method='POST', timeout='30000',
namelist='callercontext', fetchid='', synch='', maxage='0' , maxstale='0' ,
enctype='application/x-www-form-urlencoded'>|aaepicr####
@2016-07-12
14:40:22,484||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 1
milliseconds|aaepicr####
@2016-07-12
14:40:22,484||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[send.successful] State:[call_queued] Missed event.
'>|aaepicr####
@2016-07-12
14:40:22,484||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 109|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::script_element)
<fetchid='83895154aaepicr-2016194184018-18'>|aaepicr####
```

```
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='', expr='Event:[fetch.done]
State:[call_queued] Successfully feteched information for re-queuing the
call. Destination is : 64711@edp.avaya.com. Default Destination is :
64710@edp.avaya.com. EWT is : -1. Estimated Queue Position is : -1. Result
is : AGENT_AVAILABLE. Refer is : false.'>|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 326|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.ssa.result',
expr='icrcore_result.toLowerCase()'>|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='client.refer',
expr='icrcore_refer'>|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.skilldefaultvdn',
expr='icrcore_defaultvdn'>|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.acd',
expr='icrcore_acd'>|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='ewt',
expr='icrcore_ewt'>|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::send_element) <target='aaepicr-2016194184018-18',
targetype='ccxml', sendid='100672372', delay='0s',
name='create_nomedia_call', namelist='', hints=''>|aaepicr####
@2016-07-12
14:40:22,541||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
create_nomedia_call message to CXI.|aaepicr####
@2016-07-12
14:40:22,542||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 0
milliseconds|aaepicr####
@2016-07-12
14:40:22,542||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::DefaultEventProcessor) putting send.successful
with sendid (100672372) to queue|aaepicr####
@2016-07-12
14:40:22,542||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[create_nomedia_call] State:[call_queued] Cancelling no-media
call before re-queue.'>|aaepicr####
@2016-07-12
14:40:22,542||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
```

```
message of type: AppLog & size: 140|aaepicr####
@2016-07-12
14:40:22,542||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::disconnect_element) <connectionid='33563486',
reason='Disconnecting no-media call before re-queue.',
hints='{sip:hints.sip}'>|aaepicr####
@2016-07-12
14:40:22,543||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
CallDisconnect message to SessionManager.|aaepicr####
@2016-07-12
14:40:22,543||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: CallDisconnect & size: 299|aaepicr####
@2016-07-12
14:40:22,543||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.connectionid',
expr='0'>|aaepicr####
@2016-07-12
14:40:22,543||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.vdn',
expr='icrcore_vdn'>|aaepicr####
@2016-07-12
14:40:22,543||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.vdnuri',
expr='icrcore_vdn'>|aaepicr####
@2016-07-12
14:40:22,543||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
expr='Event:[create_nomedia_call] State:[call_queued] Re-queuing the
no-media call to destination : 64711@edp.avaya.com'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: AppLog & size: 164|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::createcall_element)
<connectionid='agent.connectionid', dest='sip:64711@edp.avaya.com',
timeout='', aai='', callerid='64500', hints='{MergeTimeout:agent.timeout,
DestinationURI:agent.vdnuri, contact:agent.contact, AAI:agent.AAI,
sip:hints.sip}' ,joinid='', joindirection='both' >|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.skillid',
expr='client.skillid'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='queuedCallRTD.destination', expr='agent.vdn'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element)
<name='queuedCallRTD.callcentername',
expr='encodeURI(agent.acd)'>|aaepicr####
@2016-07-12
```

```
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
CallCreate message to SessionManager.|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|Sending
message of type: CallCreate & size: 1025|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.ewt',
expr='-1'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='queuedCallRTD.qpos',
expr='-1'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::fetch_element)
<next='http://localhost/mpp/config/updateQueuedCallDetails.php?sessionid=a
aepicr-2016194184018-18&ucid=10000003671468348818&skillid=1&appname=0:CCA&
destination=64711@edp.avaya.com&callcentername=&qpos=-1&ewt=-1&callingno=s
ip:64500@edp.avaya.com', type='text/ecmascript', method='GET',
timeout='15s', namelist='', fetchid='', synch='', maxage='0' , maxstale='0'
, enctype='application/x-www-form-urlencoded'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='historyinfotext.value',
expr=''''>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='prev_queueposition',
expr='0'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='handle_merge_fail',
expr='false'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='handle_sip_error',
expr='false'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.siprespcode',
expr='0'>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::assign_element) <name='agent.ringingReceived',
expr=''false''>|aaepicr####
@2016-07-12
14:40:22,544||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (EventProcessor) Time spent by event is 0 sec and 2
milliseconds|aaepicr####
@2016-07-12
14:40:22,545||FINE|CXI|24848|Session=aaepicr-2016194184018-18|CXI
Diagnostic: (Interpreter::log_element) <label='',
```

```
expr='Event:[send.successful] State:[call_queued] Missed event.
```

```
'>|aaepicr####

...
```

Pay attention especially to the **line 347**, where CCA invokes the ICR Core directly. Because of the configuration made on the reference SSA, already explained in a previous section, ICR Core identifies that the extension module has to be invoked. From the extension module, we get Dynamic Routing invoked once more, to handle the re-queue scenario, which is reflect in the Dynamic Routing logs already described.

SSA log

Last but not least, the OD applications also have a file where information about the call flow is registered.

However, in order to get these messages recorded, the application must be configured in such a way that debugging is enabled. In this section, both the configuration and log analysis will be provided.

Enabling Log Recording

Every OD application has a file called ddrt.properties, where debugging/reporting must be enabled in order to get messages written to the log file about to be analyzed. This file is found *[od_application_home_dir]/data/log*. at and the block below shows its content, with the debugging/reporting properties, at the bottom, already enabled.

ddrt.properties

```
 # localddtrace
# Orchestration Designer builtin tracing in the trace.log
[enabled|disabled]

# localapptrace
# Application tracing in the trace.log [enabled|disabled]

# showxml
# Enables display of the generated XML output in the trace.log.
[enabled|disabled]

# showxmlroot=once
# Enables display of the generated XML output for the AppRoot in the
trace.log. [once|always|never]
# The default is once if this property is not specified.

# xmlrootinline
# Enables generation of the approot inline  [enabled|disabled]
# The default is enabled if this property is not specified.
# This results in 1/2 as many page fetches.

# localreportlog
# Enables the local report.log on platforms (Voice Portal) when the local
log
# is normally disabled. [enabled|disabled]

# frameworkreporting
# Enables application framework reporting in the report.log.  This is
reporting that tracks the
# path a caller takes through an application. [enabled|disabled]

# localsoaptrace
# Enables the WSOP to print the soap request and soap response to the
console
# is normally disabled. [enabled|disabled]

# showcaughtexceptions
# Enables display of stack trace on caught exceptions in the runtime
# is normally disabled. [enabled|disabled]

localddtrace=enabled
localapptrace=enabled
showxml=enabled
showxmlroot=once
localreportlog=enabled
frameworkreporting=enabled
localsoaptrace=enabled
xmlrootinline=enabled
showcaughtexceptions=enabled
```

Log Analysis

In this log file, the content is displayed in the format of a CCXML document, and the information is exactly the same as in the log files described in previous sections.The log file, called *trace.log*, can be found at.

trace.log

```
 12/07/2016 14:43:59:747  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
SCESession bound to HttpSession 59AB840660936AC65099F768A5DFDC67
12/07/2016 14:43:59:748  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Browser Version 2.1.4, terminationURL enabled : true

...

12/07/2016 14:43:59:872  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Using SCESession 59AB840660936AC65099F768A5DFDC67:/ICRSSA servlet : CallDR3
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :_sipcallid to simple: _sipcallid  as
[7abc92c485e41e6a4db0505683f3f]
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :redirectinfo___presentationinfo to complex:
redirectinfo:presentationinfo  as []
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :redirectinfo___reason to complex: redirectinfo:reason  as []
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :redirectinfo___screeninginfo to complex:
redirectinfo:screeninginfo  as []
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :redirectinfo___uri to complex: redirectinfo:uri  as []
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___aai to complex: session:aai  as []
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___ani to complex: session:ani  as [64500]
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___calltag to complex: session:calltag  as
[aaepicr-135.20.215.56-1-2016194184018]
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___channel to complex: session:channel  as []
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___dnis to complex: session:dnis  as [64600]
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___mediatype to complex: session:mediatype  as [video]
12/07/2016 14:43:59:873  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___protocolname to complex: session:protocolname  as [sip]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___protocolversion to complex: session:protocolversion  as
[2.0]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___sessionlabel to complex: session:sessionlabel  as
[10000036714683488 18]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
```

```
Storing :session___sharedmode to complex: session:sharedmode  as
[serviceprovider]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___ucid to complex: session:ucid  as [10000036714683488l8]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___uui to complex: session:uui  as []
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videobitrate to complex: session:videobitrate  as
[unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videocodec to complex: session:videocodec  as [unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videoenabled to complex: session:videoenabled  as
[unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videofarfmtp to complex: session:videofarfmtp  as
[unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videoformat to complex: session:videoformat  as
[unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videofps to complex: session:videofps  as [unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videoheight to complex: session:videoheight  as
[unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videonearfmtp to complex: session:videonearfmtp  as
[unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___videowidth to complex: session:videowidth  as [unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___vpcalledextension to complex: session:vpcalledextension
as []
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___vpcoveragereason to complex: session:vpcoveragereason
as []
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___vpcoveragetype to complex: session:vpcoveragetype  as []
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :session___vprdnis to complex: session:vprdnis  as []
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :shareduui___id to complex: shareduui:id  as [unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Storing :shareduui___value to complex: shareduui:value  as [unknown]
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [skillToSend] to [skill:value]
12/07/2016 14:43:59:874 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
skillToSend
12/07/2016 14:43:59:874 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin VariableField:getStringValue(): return string value for skillToSend
12/07/2016 14:43:59:874 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
```

```
12/07/2016 14:43:59:874  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [applicationName] to [callerContext:appName]
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [decisionFunction] to [callerContext:decisionFunction]
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
decisionFunction
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin VariableField:getStringValue(): return string value for
decisionFunction
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [segAttributes] to [callerContext:segAttributes]
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
segAttributes
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin VariableField:getStringValue(): return string value for segAttributes
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [segTableName] to [callerContext:segTableName]
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
segTableName
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin VariableField:getStringValue(): return string value for segTableName
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [customParams] to [callerContext:customParams]
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
customParams
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin VariableField:getStringValue(): return string value for customParams
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [session:dnis] to [callerContext:dnis]
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [session:ucid] to [callerContext:ucid]
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [session:ani] to [callerContext:ani]
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [applicationName] to [requeueCallerContext:appName]
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [decisionFunction] to [requeueCallerContext:decisionFunction]
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
decisionFunction
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
```

```
Admin VariableField:getStringValue(): return string value for
decisionFunction
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [segAttributes] to [requeueCallerContext:segAttributes]
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
segAttributes
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin VariableField:getStringValue(): return string value for segAttributes
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [segTableName] to [requeueCallerContext:segTableName]
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
segTableName
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin VariableField:getStringValue(): return string value for segTableName
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [customParams] to [requeueCallerContext:customParams]
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin SimpleVariable:getSimpleVariable(): return admin simple variable -
customParams
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Admin VariableField:getStringValue(): return string value for customParams
12/07/2016 14:43:59:875 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
get param for app: 0:CCA...SSA
12/07/2016 14:43:59:875  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [session:dnis] to [requeueCallerContext:dnis]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [session:ucid] to [requeueCallerContext:ucid]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [session:ani] to [requeueCallerContext:ani]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Executing IPluggableExecutable com.avaya.icr.pdc : Retrieving Destination
Information using extended APIs
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Skill ID : 1
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Preferred ICRCore List is : [[ICRConfigEntry] id:0, ip:135.122.99.97,
name:135.122.99.97, isPreferred:true, isFailover:] and Failover ICRCore
List is : []
12/07/2016 14:43:59:876 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Parsing global variable, ICRCore identified is : [ICRCore] id:0,
ip:135.122.99.97, protocol:null, name:null, state:1, mode:0, zone:null
12/07/2016 14:43:59:876 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Running Preferred ICR Core list is : [[ICRCore] id:0, ip:135.122.99.97,
protocol:null, name:null, state:1, mode:0, zone:null]
12/07/2016 14:43:59:876 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
```

```
Running Failover ICR Core list is : []
12/07/2016 14:43:59:876 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Stopped Preferred ICR Core list is : []
12/07/2016 14:43:59:876 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Stopped Failover ICR Core list is : []
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [ani] is [64500]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [appName] is [ICRSSA]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [customParams] is
[Custom1=CustomValue;Custom2=CustomValue2]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [decisionFunction] is [ICRDRDF]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [dnis] is [64600]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [requeue] is [false]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [segAttributes] is [Package=PREMIUM;Tenure=14]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [segTableName] is [ICRDRST]
12/07/2016 14:43:59:876  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
the value of field [ucid] is [10000003671468348818]
12/07/2016 14:43:59:876 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Randomly selected ICRCore is : [ICRCore] id:0, ip:135.122.99.97,
protocol:null, name:null, state:1, mode:0, zone:null
12/07/2016 14:44:00:320 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
getDestinationEx -- Retrieved destination information  [ DestinationDetail
[identifier=null, destination=64711@edp.avaya.com, acd=null,
acdType=Avaya-CM, defaultDest=64710@edp.avaya.com, defaultACD=LABACD,
result=AGENT_AVAILABLE, ewt=-1, refer=false,
customData={"sip.domain":"edp.avaya.com","refer":"false","previousDestinat
ion":"64711@edp.avaya.com"}, elapsedTime=0, ccxmlAppUrl=, ewtTop=-1,
ewtHigh=-1, ewtMedium=-1, ewtLow=-1, agentsStaffed=-1, agentsAvailable=-1,
estimatedQpos=-1, cmskillnumber=-1, priority=-1, percAnsSL=] ]
12/07/2016 14:44:00:320  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Result retrieved is : DestinationDetail [identifier=null,
destination=64711@edp.avaya.com, acd=null, acdType=Avaya-CM,
defaultDest=64710@edp.avaya.com, defaultACD=LABACD, result=AGENT_AVAILABLE,
ewt=-1, refer=false,
customData={"sip.domain":"edp.avaya.com","refer":"false","previousDestinat
ion":"64711@edp.avaya.com"}, elapsedTime=0, ccxmlAppUrl=, ewtTop=-1,
ewtHigh=-1, ewtMedium=-1, ewtLow=-1, agentsStaffed=-1, agentsAvailable=-1,
estimatedQpos=-1, cmskillnumber=-1, priority=-1, percAnsSL=]
12/07/2016 14:44:00:320  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Completed IPluggableExecutable com.avaya.icr.pdc
12/07/2016 14:44:00:320  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Assigning [DestinationInformation1:extra_info] to
[requeueCallerContext:previousICRExecInfo]
12/07/2016 14:44:00:321  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Using SCESession 59AB840660936AC65099F768A5DFDC67:/ICRSSA servlet : Exit
12/07/2016 14:44:00:321 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
```

```
*** Reply for [/ICRSSA/Exit] ***
12/07/2016 14:44:00:321 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
0:<?xml version="1.0" encoding="UTF-8"?>
1:<vxml version="2.1" xmlns="http://www.w3.org/2001/vxml" xml:lang="en-us">
2:<meta name="author" content="Avaya Aura Orchestration Designer"/>
3:<meta name="runtime-version" content="07.01.08.04"/>
4:<meta name="runtimecommon-version" content="07.01.08.04"/>
5:<meta name="copyright" content="Copyright (c) 2002-2011, Avaya"/>
6:<var name="_avayaExitReason" expr="''"/>
7:<var name="_avayaExitInfo1" expr="''"/>
8:<var name="_avayaExitInfo2" expr="''"/>
9:<var name="_avayaExitCustomerId" expr="''"/>
10:<var name="_avayaExitPreferredPath" expr="'1'"/>
11:<var name="_avayaExitTopic" expr="''"/>
12:<form id="Exit">
13:<block>
14:<script>
15:function getcallercontext() {
16:    var temp = new Object();
17:    temp.ani = '64500';
18:    temp.appName = 'ICRSSA';
19:    temp.customParams = 'Custom1=CustomValue;Custom2=CustomValue2';
20:    temp.decisionFunction = 'ICRDRDF';
21:    temp.dnis = '64600';
22:    temp.previousICRExecInfo =
'{&quot;sip.domain&quot;:&quot;edp.avaya.com&quot;,&quot;refer&quot;:&quot
;false&quot;,&quot;previousDestination&quot;:&quot;64711@edp.avaya.com&quo
t;}';
23:    temp.requeue = 'true';
24:    temp.segAttributes = 'Package=PREMIUM;Tenure=14';
25:    temp.segTableName = 'ICRDRST';
26:    temp.ucid = '10000003671468348818';
27:return(temp);
28:}
29:</script>
30:<var name="ssaResult" expr="'AGENT_AVAILABLE'"/>
31:<var name="skillId" expr="'1'"/>
32:<var name="vdn" expr="'64711@edp.avaya.com'"/>
33:<var name="acd" expr="''"/>
34:<var name="acdtype" expr="'Avaya-CM'"/>
35:<var name="skilldefaultvdn" expr="'64710@edp.avaya.com'"/>
36:<var name="ewt" expr="'-1'"/>
37:<var name="refer" expr="'false'"/>
38:<var name="ccxmlapplicationurl" expr="''"/>
39:<var name="wta" expr="''"/>
40:<var name="eha" expr="''"/>
41:<var name="AAI" expr="''"/>
42:<var name="querystring" expr="''"/>
43:<var name="_avayaExitInfo1" expr="'ICRSSA'"/>
44:<var name="UUI" expr="''"/>
45:<var name="enableExternalMessage" expr="false"/>
46:<var name="onRequeueUseExtension" expr="true"/>
47:<var name="callercontext" expr="getcallercontext()"/>
```

```
48:<exit namelist="ssaResult skillId vdn acd acdtype skilldefaultvdn ewt
refer ccxmlapplicationurl wta eha AAI querystring _avayaExitInfo1 UUI
enableExternalMessage onRequeueUseExtension callercontext _avayaExitReason
_avayaExitInfo1 _avayaExitInfo2 _avayaExitCustomerId
_avayaExitPreferredPath _avayaExitTopic"/>
49:</block>
50:</form>
51:</vxml>
52:

12/07/2016 14:44:00:326  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Removing session from AvayaSessionTermination Servlet
12/07/2016 14:44:00:326  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
Termination URL reply : <?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.1" xmlns="http://www.w3.org/2001/vxml" xml:lang="en-us">
</vxml>

12/07/2016 14:44:00:326  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
HTTP Session lost removing SCESession 59AB840660936AC65099F768A5DFDC67
12/07/2016 14:44:00:326  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
SCESession Removed
12/07/2016 14:44:00:326 DEBUG - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
```

```
Removing 1 session stack frames.
12/07/2016 14:44:00:326  INFO - 59AB840660936AC65099F768A5DFDC67:/ICRSSA :
** Popped Stack Frame [/ICRSSA]
```

Further documentation

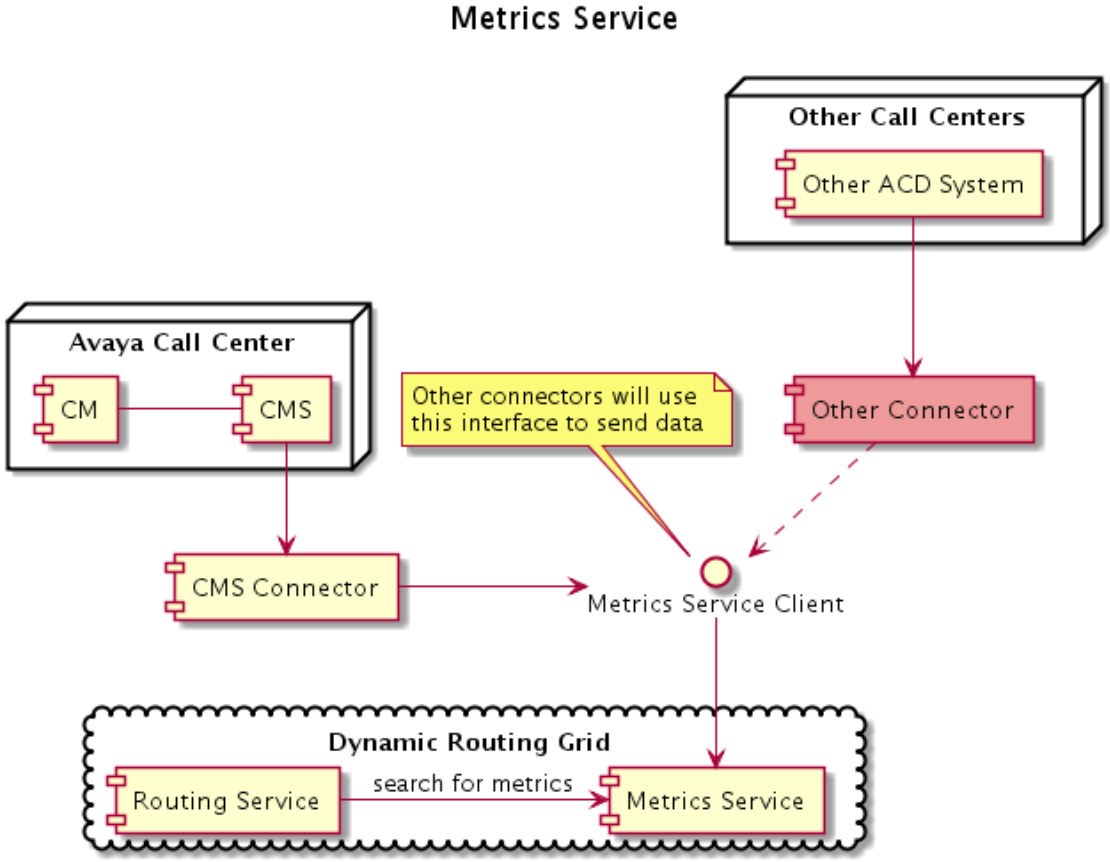
For additional information on the ICR integration with Dynamic Routing, please access this page.

DR 3.2 Metrics Connector SDK (for Metrics Connectors)

- Overview
- Dependencies
    - Dynamic Routing Libraries
    - Other Libraries
- Configuration
- Sample Code

Overview

Dynamic Routing exposes APIs to its various components, as we see in the DR3 Integration Guides. The following image shows how the *Metrics* component allows connectors to interact with it, and send metrics (e.g., EWT, STAFFED, etc) to the Dynamic Routing grid (memory space).



DR3 provides one metrics connector for CMS, out of the box, a.k.a the *CMS Connector or CMSC*.

New metrics connectors can be created to send metrics from other ACD systems (such as: Cisco, and Aspect) to Dynamic Routing. In other words, it is possible to create the red box in the image above.

The sections below describe how to create a new metrics connector (as this is just an example, fake metric values will be used).

Dependencies

Dynamic Routing Libraries

- dr_metrics_comm_client
- dr_metrics_comm_messages
- dr_metrics_comm_listeners

- dr_metrics_space_client
- commons-logger.jar

Other Libraries

- spring-context-4.1.1.RELEASE.jar
- spring-core-4.1.1.RELEASE.jar
- spring-beans-4.1.1.RELEASE.jar
- spring-expression-4.1.1.RELEASE.jar
- spring-tx-4.1.1.RELEASE.jar
- commons-logging-1.1.3.jar
- gs-openspaces-10.1.1-12800-RELEASE.jar
- gs-runtime-10.1.1-12800-RELEASE.jar
- spring-aop-4.1.1.RELEASE.jar
- aopalliance-1.0.jar
- slf4j-api-1.7.7.jar
- kryo-3.0.0.jar
- minlog-1.3.0.jar
- objenesis-2.1.jar
- snappy-java-1.1.1.6.jar

Configuration

The Metrics Client SDK only requires one property file called ***`metrics_connector_communication.properties`*** and must be located in the application´s `classpath`. There is one mandatory modification to be done within this file:

- set Metrics Space and Lookup Locators, through the property *`metricsSpace.name`*

This property requires the configuration of the lookup services used to find the data grids. These lookup service addresses are usually the same ones defined at installation time (*LOOKUP LOCATORS*) and can be configured using the suffix *?locators=<locators addresses>*. Taking the Lookup Locators *172.28.128.3:4174* and *172.28.128.4:4174* as example, the properties would be configured as follows:

metrics_connector_communication.properties

```
metricsSpace.name=dr_routing_core_space?locators=172.28.128.3:4174,172.28.
128.4:4174
```

As we can see in the example above, the metrics space name is `dr_routing_core_space` which is the default and the lookup locators have to be set (manually), separated by commas.

Sample Code

Main.java

```
import
com.avaya.ept.dr.services.metrics.comm.client.IDestinationMetricsClient;
import
com.avaya.ept.dr.services.metrics.comm.client.factory.DestinationMetricsCl
ientServiceFactory;
import
com.avaya.ept.dr.services.metrics.comm.messages.DestinationMetricsUpdateMe
ssage;
import
com.avaya.ept.dr.services.metrics.comm.messages.DestinationResolutionMode;
```

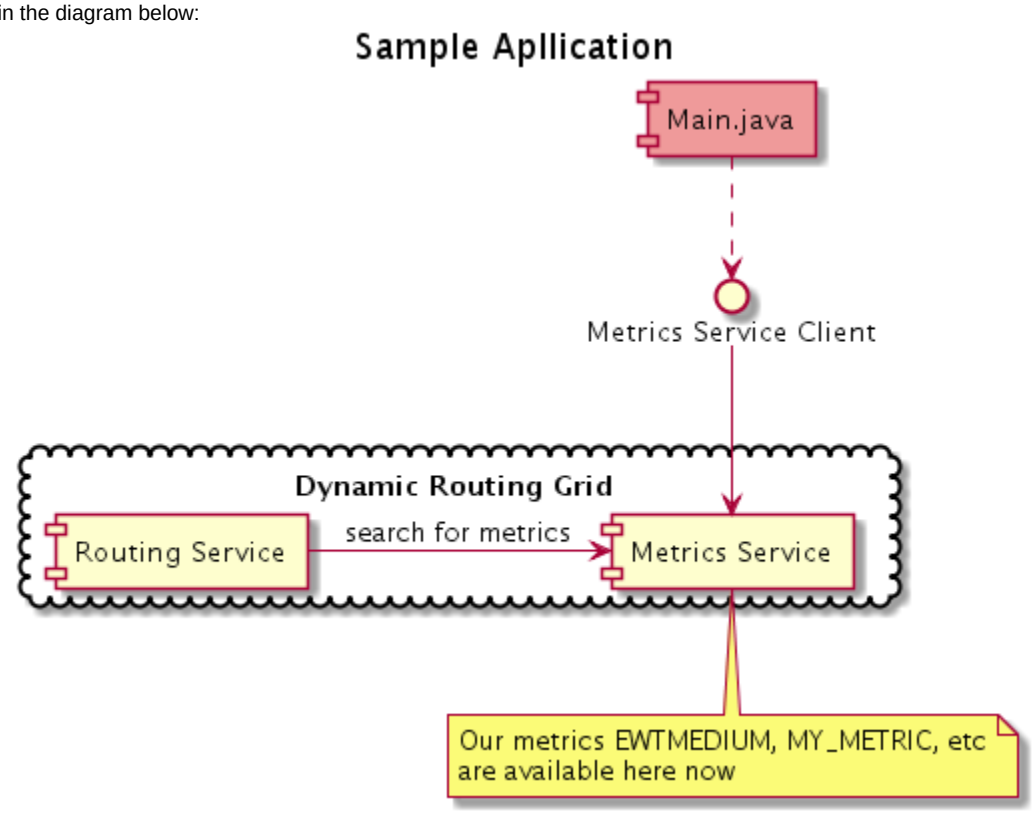
```
import java.util.HashMap;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;

public class Main {

    public static void main(String[] args) {
        IDestinationMetricsClient metricsClient =
DestinationMetricsClientServiceFactory.getDestinationMetricsGSPollingServi
ce();
        metricsClient.setSingleSourceHint(true);

        // Bulk of messages
        List<DestinationMetricsUpdateMessage> messages = new
LinkedList<>();

        // Message per Agent Group
        DestinationMetricsUpdateMessage message = new
DestinationMetricsUpdateMessage();

        // IMPORTANT: This is Dynamic Routing ACD ID. Please look into
Dynamic Routing RA > ACDs to find out.
        String acdId = "f15ba477-9216-4bf5-968f-8444d21cdbc4";
        message.setSystemId(acdId);

        // Our fake metric values
        Map<String, Object> metricsMap = new HashMap<>();
        metricsMap.put("EWTMEDIUM", "3");
        metricsMap.put("STAFFED", "20");
        metricsMap.put("AVAILABLE", "9");
        metricsMap.put("CALLSOFFERED", "150");
        metricsMap.put("PERCENTINSL", "80");
        metricsMap.put("MY_METRIC", "9999"); // I can send other metrics to
the DR too!
        message.addAll(metricsMap);

        // Agent Group Native ID (on ACD side)
        message.setNativeId("1011");

        message.setDestinationName("AgentGroup_Name");

        // In the future might be used to identify tenant and help on
partitioning. Now it´s OPTIONAL.
        message.setTenantId("1");

        // It determines how metrics will be identified in DR. The only
supported value now is SYSTEM_ID_NATIVE_DESTINATION_ID

 message.setResolutionMode(DestinationResolutionMode.SYSTEM_ID_NATIVE_DEST
INATION_ID);
```

```
        // Zero is the only supported value for now, which means
DestinationType.AGENT_GROUP
        message.setDestinationType((short) 0);

        // Add message to the bulk
        messages.add(message);

        // Call DR remotely
        try {
            metricsClient.updateDestinations(acdId, messages);
        } catch (Exception e) {
            // Log the error here...
        }
    }
```

```
}
```

> The method `metricsClient.updateDestinations(acdId, messages)` does not return any value (`void`) but might throw `IllegalStateException` if something goes wrong. There are error messages in the log too.

If everything is setup properly when you execute your Java application, your metrics will be sent to Dynamic Routing and available to script writers, as we see in the diagram below:



For further information take a look at DR Java Docs available with the DR installation bundle.

DR 3.2 RDR Reference

- About RDRs
- RDRs Location
    - File System (Flat File)
    - Dynamic Routing Internal Database (Internal DB)
    - Customer Provided External Database (External DB)
- RDR Space Provisioning
    - Database Space Estimation
    - Data Purge
    - RDR DB Persistence Error Handler Configuration
- RDR Content
    - Diagnostic Information Available in an RDR
    - RDR Columns
- External Access to the Internal PostgreSQL RDR DB
    - Exploring RDR DB with PostgreSQL Administration Tool
- Sample RDR Queries
    - Retrieve all RDRs After a Specific Date and Time
    - Retrieve all RDRs Using Current Date
    - Retrieve Number of Decisions to Each Destination
- Configuring RDRs to External Database
    - Configuration for PostgreSQL
    - Configuring Databases other than PostgreSQL
    - Configuration for MySQL
        - CMT Configuration
        - Table Creation Command
    - Configuration for Oracle
        - CMT Configuration
        - Table Creation Command
    - Configuration for Microsoft SQL Server
        - CMT Configuration
        - Table Creation Command

About RDRs

A Routing Decision Record contains the detail of each request/response received/answered by Dynamic Routing. These records can be saved in various formats and targets, depending on customer needs, possible option is:

- Flat File - this is the simplest way to save RDRs, there is no Database, and the files (which are in JSON format) are saved in the nodes that have Management Services
- Internal Database - RDRs are saved in a Dynamic Routing internal database located at the Management nodes (low volume)
- External Database - RDRs are saved in an external database provided by the customer (high volume)

RDRs Location

File System (Flat File)

Dynamic Routing will write the RDRs in files, in JSON format. The files should be extracted from the Management nodes to be post processed. This RDR option is recommended to small systems that have no Messaging Services (small footprint) and low traffic.

Dynamic Routing Internal Database (Internal DB)

Dynamic Routing has internal databases on the nodes that run Management Services (PostgreSQL) and the RDRs are saved on these database. There is an internal process that purges the old data. This RDR option is recommended for troubleshooting or quick review of traffic details.

Customer Provided External Database (External DB)

Saving the RDRs in an external database, the user can store the data as long as it is needed (and supported by the provided database). To know

more about the supported database technologies and how to configure the connection to external databases, please review the section below **Configuring RDRs to External Database**. This is the most recommended option for storing RDRs,

RDR Space Provisioning

Database Space Estimation

Space requirements are proportional to the amount of data stored for each record and the call traffic. A rough guideline is provided below for various traffic levels, but should be confirmed for each implementation:

- 25,000 BHCC Requests - 600MB/Day
- 250,000 BHCC Requests - 5.2GB/Day
- 500,000 BHCC Requests - 10.4GB/Day

- 150,000 RDRs - 1 GB

Data Purge

Depending on the destination of the RDRs, the data purge is different:

- File System: No data purge
- Internal Database: Dynamic Routing retains up to 100,000 RDR records. After reaching this limit, the internal table is truncated
- External Database: The purge should be configured by the customer. There are no Dynamic Routing restrictions.

RDR DB Persistence Error Handler Configuration

Dynamic Routing might run into errors while trying to store RDR data into the database, in such cases, after 3 attempts, an error handling mechanism is triggered in order to prevent infinite retries. The error handler will raise an alarm, log an error and export the RDR to the file system.

The non-persisted RDRs are written to '*<DR_HOME>//dr/data/application/decisions_rejected.json*'. There is a scheduled process i.e. CRON job (configurable using the Cluster Management Tool) that runs and re-injects these decision records into the database.

RDR Content

The following list describes the main fields, grouped by purpose:

- **RDR identification fields**:
  - **`id`**: Primary Key and **unique** identifier of each record in the RDR DB; it includes the unique Decision object identifier from the Dynamic Routing Data Grid.
  - **`client_interaction_id`**: unique identifier from the application invoking Dynamic Routing. For example, an IVR platform (such as AAEP) can generate a UCID for each voice call. In some cases, this value may also be called UCID but be generated by the SBC or ASM.
  - **`dr_timestamp:`** specific time when Dynamic Routing received the request.
- **Input data in the Decision Request.**
  - Some of these fields contain data specific to each request: such as **`segmentation_attributes`** or the business identifier of the caller: **`from_contact_id`** (e.g.: customer Id in a CRM system or customer account number).
  - Other fields contain data coming from the application invoking Dynamic Routing requests (e.g.: **`to_address`** also known as DNIS in a telephony environment, **`client_from`** i.e. ANI, **`channel`**, client application invoking Dynamic Routing info such as **`from_client_ip`**, **`from_client_app`**)
- **Decision processing details**:
  - **`dr_server`** provides the IP Address of the Dynamic Routing node that processed the request
  - Some fields refer to Dynamic Routing configuration elements that affect the decision process, like the Decision Function being executed i.e. **`dr_script`**, or **`segmentation_table`**, **`segmentation_label`**, **`strategy_settings_name`**, and **`strategy_script_name`**
  - **`debug_string`** is a long string containing information described in a section below

    - Response time in milliseconds: **`response_time`**
- **Output data in the Decision Response**:
    - **`result_code`** described in a section below
    - Some fields relate to the selected Destination and associated data: **`selected_dest_name`**, **`selected_dest_id`**, **`selected_dest_address`**, **`selected_dest_alias`**
    - Some fields are only populated when the selected destination is an Agent Group: **`company_name`**, **`location_name`**
- **User defined data**:
    - The fields **`biz_data1`** to **5** are free for users to populate any value from the decision Scripts. They can be used to augment the reporting information already available.

Other fields not mentioned in this list are only relevant to the Dynamic Routing system, but do not provide useful information to users or external systems. For this reason they can be ignored by any process working with the RDR DB.

Diagnostic Information Available in an RDR

The **RESULT CODE** is the most important data in a Dynamic Routing response. It explains what happened in the decision logic.

Selected Destination & **Selected Dest Address** are the key fields when a **Destination** is selected as a decision Response:

- Selected Destination can be an Application or an Agent Group, when it is an:
    - Application, the Address can be anything configured by the user e.g. a Voice dialog module name, or a URL for a completely different application
    - Agent Group (e.g. for the IVR App to transfer to) the Skill/Queue Name and its Address are provided.
- Selected Destination fields (Name & Address) can also be "null" (empty) in the case where a decision is only focused on caller experience variables (audio messages, self service optional treatment, etc). This scenario is typically found when Self Service Apps are using the Dynamic Routing engine for experience configuration flexibility, in addition to destination selection (a.k.a. *routing*).

The **extendedResponse** adds more details to the selected Destination, for example, which **metric** was used for this selection, and what **value** th e metric had.

Example: 32 seconds EWT for the selected XYZc Destination

extendedResponse: Selected=XYZc|EWT|32

Finally the **DEBUG STRING** should explain **HOW** and **WHY** the strategy selected that specific Destination.

The debugString is a collection of consecutive "tags", and there are two different *types* of tags:

- **Support tags**: they have a 5 letter format (example: "AAbb:X"). This low level information can be ignored as it is intended for Avaya Support only. Each one of these represents an invocation to one of the available primitive functions in the Dynamic Routing decision engine.
- **Business tags**: they should be human-readable messages, starting with #. These are printed by the Strategy Script, and are generated with `decisionBuilder.appendStep()`. The goal for these tags is to indicate each part of the decision logic flow or "business criteria" being executed.

Example:

SGds:S;CFtt:S;#chkClosed;#chkRAVA.layer1;CFld:S;CFci:S;#XYZa=RAVA.0.Discarded;

MTam:N;MTam:N;#XYZb=RAVA.Unavailable;#chkRAVA.layer2;CFld:S;CFci:S;

#XYZc=RAVA.0.Discarded;#chkEWT;MTam:N;#XYZb=EWT.Unavailable;

#XYZa=EWT.34.Discarded;DSbd:S

The relevant fields here, start with # (hashtag), so everything else can be ignored. Comments for the above example:

NOTICE: anything after the #, which does not start with “chk*” is a Destination **Name** (Agent Group configured in Dynamic Routing RA)

#chkClosed; // Checking Timetable for hours of operation

#chkRAVA.layer1; // Checking Relative Availability (AVAILABLE/STAFFED) for Destinations marked as “layer1”

#XYZa=RAVA.0.Discarded; // Discarding XYZa destination, RAVA value 0 means no available agents

#XYZb=RAVA.Unavailable; // Discarding XYZb destination, some of the metrics are not loaded in DR3 (either AVAILABLE and/or STAFFED)

#chkRAVA.layer2; // Checking Rel. Availability for Destinations marked as “layer2”

#XYZc=RAVA.0.Discarded; // Discarding XYZc destination, no agents available

#chkEWT; // Checking EWT

#XYZb=EWT.Unavailable; // EWT metric is not loaded on DR3 for XYZb so it is ignored

#XYZa.34.Discarded; // EWT value for XYZa is higher than the selected destination so it is discarded.

In the end, XYZc destination was selected because its EWT=32 seconds and was the lowest.

This is just an EXAMPLE, each script can have its own flow and tags, but this provides an idea of the “thought process” on *how to structure* such information.

RDR Columns

Field Name	Type	Size	Information Source	Content
CHANNEL	String	255	WS REQUEST - CUSTOMER PROVIDED	Requestor's channel (i.e. VOICE, SMS, etc.)
COUNT_DECISION	Boolean (String)	true / false	WS REQUEST - CUSTOMER PROVIDED	Indicates if the request will affect the internal counters (charts)
FROM_CONTACT_ADDRESS	String	255	WS REQUEST - CUSTOMER PROVIDED	Customer endpoint address. (ex. IP Address for Chat or Email)
CLIENT_FROM	String	255	WS REQUEST - CUSTOMER PROVIDED	Customer endpoint identifier. (ex. ANI for Voice, Customer Email, etc.)
FROM_CONTACT_ID	String	255	WS REQUEST - CUSTOMER PROVIDED	Customer Identifier e.g. customer Id in a CRM system or customer account number
DR_SCRIPT	String	255	WS REQUEST - CUSTOMER PROVIDED	Decision Function name used for the request
FROM_CLIENT_IP	String	255	WS REQUEST - CUSTOMER PROVIDED	Requestor's Application Name (i.e. IVR script name)
FROM_CLIENT_APP_RELEASE	String	255	WS REQUEST - CUSTOMER PROVIDED	Requestor's Application Version
FROM_CLIENT_APP	String	255	WS REQUEST - CUSTOMER PROVIDED	Requestor's IP Address
CLIENT_INTERACTION_ID	String	255	WS REQUEST - CUSTOMER PROVIDED	Universal Contact Identifier
SEGMENTATION_ATTRIBUTES	String	255	WS REQUEST - CUSTOMER PROVIDED	Contact Attributes
TO_ADDRESS	String	255	WS REQUEST - CUSTOMER PROVIDED	Customer's entry point (i.e. DNIS)
SELECTED_DEST_ADDRESS	String	255	WS REQUEST - CUSTOMER PROVIDED	Destination route (i.e. SIP number)
BIZ_DATA1	String	255	DECISION FUNCTION PROVIDED INFORMATION	Available to be filled with custom information.
BIZ_DATA2	String	255	DECISION FUNCTION PROVIDED INFORMATION	Available to be filled with custom information.
BIZ_DATA3	String	255	DECISION FUNCTION PROVIDED INFORMATION	Available to be filled with custom information.
BIZ_DATA4	String	255	DECISION FUNCTION PROVIDED INFORMATION	Available to be filled with custom information.
BIZ_DATA5	String	255	DECISION FUNCTION PROVIDED INFORMATION	Available to be filled with custom information.
RESULT_CODE	Int		WS OUTPUT - DR PROVIDED	Numeric value to show success/failure. 1 identifies success, other values are customized in DF to represent different processing results

SEGMENTATION_LABEL	String	255	WS OUTPUT - DR PROVIDED	Segmentation Table selected row identifier (Label column)
COMPANY_NAME	String	255	WS OUTPUT - DR PROVIDED	Selected Company
LOCATION_NAME	String	255	WS OUTPUT - DR PROVIDED	Selected Location
SELECTED_DEST_ALIAS	String	255	INTERNAL PROCESSING INFORMATION	Selected Agent Group Alias
SELECTED_DEST_ID	String	255	INTERNAL PROCESSING INFORMATION	Selected Agent Group Internal ID
SELECTED_DEST_NAME	String	255	INTERNAL PROCESSING INFORMATION	Selected Agent Group Name
ID	String	255	INTERNAL PROCESSING INFORMATION	Dynamic Routing transaction internal identifier
DEBUG_STRING	String	255	INTERNAL PROCESSING INFORMATION	Executed steps to get the decision.
SEGMENTATION_TABLE	String	255	INTERNAL PROCESSING INFORMATION	Last Segmentation Table name used in the decision
DR_TIMESTAMP	Timestamp	N/A	INTERNAL PROCESSING INFORMATION	Request's Data and Time
DR_SERVER	String	255	INTERNAL PROCESSING INFORMATION	Server used to process the request
STRATEGY_SCRIPT_NAME	String	255	INTERNAL PROCESSING INFORMATION	Strategy Script name used in the decision
STRATEGY_SETTINGS_NAME	String	255	INTERNAL PROCESSING INFORMATION	Strategy Settings name used in the decision
RESPONSE_TIME	String	255	INTERNAL PROCESSING INFORMATION	Time to answer the request (in msecs)

External Access to the Internal PostgreSQL RDR DB

The PostgreSQL database that is included with Dynamic Routing is closed to external access. It is updated automatically by Dynamic Routing when a backup management node starts, so the standby node can connect to the database.

To access the RDRs from an external tool, you need to configure PostgreSQL to accept connections from other servers.

PROCEDURE:

1. Open the *<DR_HOME>/dr/data/pgsql/pg_hba.conf*
2. Edit the file and add the following line to the bottom of the file:

```
host    dynamicrouting_rdr_db    all    192.168.0.1/24    md5
```

> **Note on PostgreSQL Configuration**
> In the example above, all servers in the sub-network 192.168.0.X will be allowed to reach this database. The number '24' in 192.168.0.1/**24** indicates that the static (fixed) part of the addresses is the 3 first octects (so the last one forms a range of machines to be considered and allowed).
>
> If only a specific IP address is allowed to reach the database, change from 24 to 32 and indicate the IP address to be mapped. Then, no ranges would be considered and only the specific address you provided will be allowed to communicate with PostgreSQL.
>
> When finding any communication problem with the database, you may also change the permission mode from md5 (the one recommended above) to '**trust**'.

3. Save the file.

4. Restart PostgreSQL by running the command: **service dr-postgres stop** and then run the command **service dr-postgres start**

Remember to use Port 5555 when accessing the DB.

Exploring RDR DB with PostgreSQL Administration Tool

Using the **pgAdmin3** client, you can connect to the **RDR DB** and execute arbitrary SQL queries to retrieve data for every Decision made by Dynamic Routing.

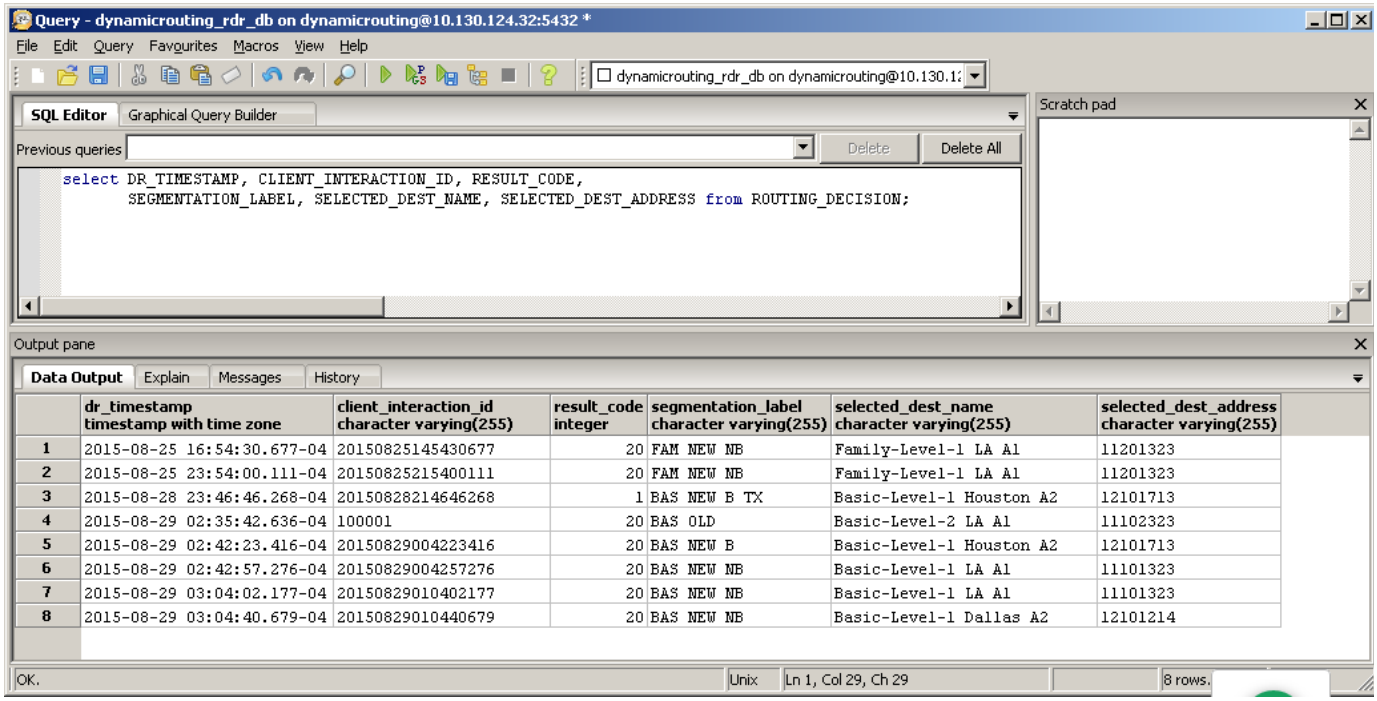
The "Add Connection to a server" button (1st toolbar button) allows to connect to the RDR DB, using the following parameters:

- Name: any arbitrary name you want to use for this connection, ex: DR3_RDR-DB
- Host: the IP address of your RDR Database Server
- Port: <port> (5555 default)
- Username: <provided_username> (default: postgres)
- Password: <provided_password> (default: 123456)
- Maintenance DB: dynamicrouting_rdr_db

Then use the SQL button to bring up an editor window, and execute a query like this:

```
select DR_TIMESTAMP, CLIENT_INTERACTION_ID, RESULT_CODE, SEGMENTATION_LABEL, SELECTED_DEST_NAME,
SELECTED_DEST_ADDRESS from ROUTING_DECISION;
```

The resulting output should contain your most recent test requests, like in the screen capture below:



When running this query with **pgAdmin3**, the tool auto-limits the resulting record set to the first 100 records to avoid loading too much data.

WARNING: It is NOT recommended to run this query in a Production setup (without any "where" filter to narrow down the result set). See suggested queries in later section.

Sample RDR Queries

The following queries can help monitor Dynamic Routing operation, and diagnose issues at run-time.

Retrieve all RDRs After a Specific Date and Time

The 4 columns/fields being printed are the most frequently used, but you could choose to print other fields, such as: debug_string, to_address, company_name, location_name, etc.

```
SELECT to_char(dr_timestamp, 'MM-DD HH24:MI:SS.MS') AS Timestamp,
      result_code, segmentation_label, selected_dest_name, selected_dest_address
      FROM routing_decision WHERE dr_timestamp >= '2015-08-29 02:36:00';
```

Retrieve all RDRs Using Current Date

This query is identical to the previous one, except the timestamp filter is using a built-in PostgreSQL function to generate *current date* and avoid the need to adjust the specific date every time you run.

```
SELECT to_char(dr_timestamp, 'MM-DD HH24:MI:SS.MS') AS Timestamp,
      result_code, segmentation_label, selected_dest_name, selected_dest_address
      FROM routing_decision WHERE dr_timestamp >= CURRENT_DATE;
```

WARNING: The previous queries can return a **large** number of **RDRs** (many thousands), so be careful with date filtering.

Retrieve Number of Decisions to Each Destination

This query **counts** how many decisions were routed to each Destination:

```
SELECT selected_dest_name, COUNT(id) FROM routing_decision
      WHERE dr_timestamp >= '2015-08-01 02:00:00' AND dr_timestamp < '2015-08-01 03:00:00'
      GROUP BY selected_dest_name;
```

By replacing, “selected_dest_name” field with other column names, other interesting aggregations can be obtained. For example: result_code, company_name or segmentation_label are different counting criteria which will likely be useful.

Configuring RDRs to External Database

Dynamic Routing has the option of using external databases for storing the RDR records. The user must configure:

- DB User
- DB Password
- JDBC Connection URL
- Hibernate Dialect
- DB vendor specific JDBC Driver Class
- DB connection test statement

The configuration can be done using the Cluster Management Tool, see more details in the Node Properties section.

Configuration for PostgreSQL

Example of CMT node properties that could be configured are:

server.kafkaconsumer.groupid = dr-decisions-group-001

```
server.kafkaconsumer.db.user = dynamicrouting
server.kafkaconsumer.db.pwd = 123456
server.kafkaconsumer.db.connection.url = jdbc:postgresql://127.0.0.1:5555/dynamicrouting_rdr_db
server.kafkaconsumer.db.hibernate.dialect = org.hibernate.dialect.PostgreSQL82Dialect
server.kafkaconsumer.db.driver.class = org.postgresql.Driver
server.kafkaconsumer.db.connection.testquery=select 1
```

For PostgreSQL, the user does not need to provide the JDBC driver, as it is included in the Dynamic Routing load. Neither does he need to configure the dialect, and class, as they will be set correctly by default.

Configuring Databases other than PostgreSQL

For databases other the PostgreSQL, the user must also:

- place the JDBC driver jar in the folder *[DR_HOME]/lib/shared*
- execute an SQL command, to create the RDR DB tables. Samples commands are shown below for the following DB vendors:
    - MySQL
    - Oracle
    - Microsoft SQL Server

For any other DB, a DB administrator should follow the examples below to determine the correct command for their specific DB.

IMPORTANT: For external RDR DBs, users must implement their own mechanism to **purge RDR records**. It is up to the user to determine what is the appropriate mechanism to purge RDR records on a regular basis, and schedule such purges so that the RDR database does not grow indefinitely.

Configuration for MySQL

CMT Configuration

Example of CMT node properties that could be configured are:

```
server.kafkaconsumer.groupid =  dr-decisions-group-001
server.kafkaconsumer.db.user = root
server.kafkaconsumer.db.pwd = admin
server.kafkaconsumer.db.connection.url = jdbc:mysql://192.168.0.104:3306/rdr
server.kafkaconsumer.db.hibernate.dialect = org.hibernate.dialect.MySQLDialect
server.kafkaconsumer.db.driver.class = com.mysql.jdbc.Driver
server.kafkaconsumer.db.connection.testquery=select 1
```

Table Creation Command

```
CREATE TABLE `routing_decision` (
  `id` varchar(255) NOT NULL DEFAULT '',
  `biz_data1` varchar(255) DEFAULT NULL,
  `biz_data2` varchar(255) DEFAULT NULL,
  `biz_data3` varchar(255) DEFAULT NULL,
  `biz_data4` varchar(255) DEFAULT NULL,
  `biz_data5` varchar(255) DEFAULT NULL,
  `channel` varchar(255) DEFAULT NULL,
  `company_name` varchar(255) DEFAULT NULL,
  `debug_string` text,
  `dr_script` varchar(255) DEFAULT NULL,
```

```
  `dr_server` varchar(255) DEFAULT NULL,
  `client_from` varchar(255) DEFAULT NULL,
  `from_client_app` varchar(255) DEFAULT NULL,
  `from_client_ip` varchar(255) DEFAULT NULL,
  `from_contact_address` varchar(255) DEFAULT NULL,
  `from_contact_id` varchar(255) DEFAULT NULL,
  `client_interaction_id` varchar(255) DEFAULT NULL,
  `location_name` varchar(255) DEFAULT NULL,
  `response_time` bigint(20) DEFAULT NULL,
  `segmentation_attributes` varchar(5000) DEFAULT NULL,
  `segmentation_label` varchar(255) DEFAULT NULL,
  `segmentation_table` varchar(255) DEFAULT NULL,
  `selected_dest_address` varchar(255) DEFAULT NULL,
  `selected_dest_id` varchar(255) DEFAULT NULL,
  `selected_dest_name` varchar(255) DEFAULT NULL,
  `selected_dest_alias` varchar(255) DEFAULT NULL,
  `strategy_script_name` varchar(255) DEFAULT NULL,
  `strategy_settings_name` varchar(255) DEFAULT NULL,
  `dr_timestamp` timestamp NULL DEFAULT NULL,
  `to_address` varchar(255) DEFAULT NULL,
  `count_decision` tinyint(1) DEFAULT NULL,
  `monitoring_metrics_processed` tinyint(1) DEFAULT NULL,
  `result_code` int(11) DEFAULT NULL,
  `version_id` int(11) DEFAULT NULL,
  `update_timestamp` timestamp NULL DEFAULT NULL,
  `custom_parameters` text,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

Configuration for Oracle

CMT Configuration

Example of CMT node properties that could be configured are:

```
server.kafkaconsumer.groupid =  dr-decisions-group-001
server.kafkaconsumer.db.user = SYSTEM
server.kafkaconsumer.db.pwd = 123456
server.kafkaconsumer.db.connection.url = jdbc:oracle:thin:SYSTEM/123456@192.168.0.104:1521:xe
server.kafkaconsumer.db.hibernate.dialect = org.hibernate.dialect.OracleDialect
server.kafkaconsumer.db.driver.class = oracle.jdbc.OracleDriver
server.kafkaconsumer.db.connection.testquery=select 1 from dual
```

Table Creation Command

```
CREATE TABLE SYSTEM.ROUTING_DECISION
(
  ID                          VARCHAR2(255 BYTE),
  BIZ_DATA1                   VARCHAR2(255 BYTE),
  BIZ_DATA2                   VARCHAR2(255 BYTE),
  BIZ_DATA3                   VARCHAR2(255 BYTE),
  BIZ_DATA4                   VARCHAR2(255 BYTE),
  BIZ_DATA5                   VARCHAR2(255 BYTE),
  CHANNEL                     VARCHAR2(255 BYTE),
  COMPANY_NAME                VARCHAR2(255 BYTE),
  DEBUG_STRING                CLOB,
  DR_SCRIPT                   VARCHAR2(255 BYTE),
  DR_SERVER                   VARCHAR2(255 BYTE),
  CLIENT_FROM                 VARCHAR2(255 BYTE),
  FROM_CLIENT_APP             VARCHAR2(255 BYTE),
  FROM_CLIENT_IP              VARCHAR2(255 BYTE),
  FROM_CONTACT_ADDRESS        VARCHAR2(255 BYTE),
  FROM_CONTACT_ID             VARCHAR2(255 BYTE),
  CLIENT_INTERACTION_ID       VARCHAR2(255 BYTE),
  LOCATION_NAME               VARCHAR2(255 BYTE),
  RESPONSE_TIME               LONG,
  SEGMENTATION_ATTRIBUTES     CLOB,
  SEGMENTATION_LABEL          VARCHAR2(255 BYTE),
  SEGMENTATION_TABLE          VARCHAR2(255 BYTE),
  SELECTED_DEST_ADDRESS       VARCHAR2(255 BYTE),
  SELECTED_DEST_ID            VARCHAR2(255 BYTE),
  SELECTED_DEST_NAME          VARCHAR2(255 BYTE),
  SELECTED_DEST_ALIAS         VARCHAR2(255 BYTE),
  STRATEGY_SCRIPT_NAME        VARCHAR2(255 BYTE),
  STRATEGY_SETTINGS_NAME      VARCHAR2(255 BYTE),
  DR_TIMESTAMP                TIMESTAMP(6),
  TO_ADDRESS                  VARCHAR2(255 BYTE),
  COUNT_DECISION              NUMBER,
  MONITORING_METRICS_PROCESSED  NUMBER,
  RESULT_CODE                 INTEGER,
  VERSION_ID                  INTEGER,
  UPDATE_TIMESTAMP            TIMESTAMP(6),
  CUSTOM_PARAMETERS           CLOB
)
ALTER TABLE SYSTEM.ROUTING_DECISION ADD (
  CONSTRAINT RDR_PK
```

PRIMARY KEY

(ID)

USING INDEX SYSTEM.RDR_PK

ENABLE VALIDATE);

Configuration for Microsoft SQL Server

CMT Configuration

Example of CMT node properties that could be configured are:

```
server.kafkaconsumer.groupid =  dr-decisions-group-001
server.kafkaconsumer.db.user = sa
server.kafkaconsumer.db.pwd = 123456
server.kafkaconsumer.db.connection.url = jdbc:jtds:sqlserver://192.168.0.100:1433/rdr
server.kafkaconsumer.db.hibernate.dialect = org.hibernate.dialect.SQLServerDialect
server.kafkaconsumer.db.driver.class = net.sourceforge.jtds.jdbc.Driver
server.kafkaconsumer.db.connection.testquery=select 1
```

Table Creation Command

CREATE TABLE [dbo].[routing_decision](

[id] [varchar](255) NOT NULL,

[biz_data1] [varchar](255) NULL,

[biz_data2] [varchar](255) NULL,

[biz_data3] [varchar](255) NULL,

[biz_data4] [varchar](255) NULL,

[biz_data5] [varchar](255) NULL,

[channel] [varchar](255) NULL,

[company_name] [varchar](255) NULL,

[debug_string] [text] NULL,

[dr_script] [varchar](255) NULL,

[dr_server] [varchar](255) NULL,

[client_from] [varchar](255) NULL,

[from_client_app] [varchar](255) NULL,

[from_client_ip] [varchar](255) NULL,

[from_contact_address] [varchar](255) NULL,

[from_contact_id] [varchar](255) NULL,

[client_interaction_id] [varchar](255) NULL,

[location_name] [varchar](255) NULL,

[response_time] [bigint] NULL,

[segmentation_attributes] [varchar](5000) NULL,

[segmentation_label] [varchar](255) NULL,

[segmentation_table] [varchar](255) NULL,

[selected_dest_address] [varchar](255) NULL,

```
    [selected_dest_id] [varchar](255) NULL,
    [selected_dest_name] [varchar](255) NULL,
    [selected_dest_alias] [varchar](255) NULL,
    [strategy_script_name] [varchar](255) NULL,
    [strategy_settings_name] [varchar](255) NULL,
    [dr_timestamp] [datetime] NULL,
    [to_address] [varchar](255) NULL,
    [count_decision] [bit] NULL,
    [monitoring_metrics_processed] [bit] NULL,
    [result_code] [int] NULL,
    [version_id] [int] NULL,
    [update_timestamp] [datetime] NULL,
    [custom_parameters] [text] NULL,
 CONSTRAINT [PK_routing_decision] PRIMARY KEY CLUSTERED
(
    [id] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON,
ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
SET ANSI_PADDING OFF
GO
```

DR 3.2 Routing Remoting Client SDK

- Description
- Dependencies
    - Dynamic Routing Libraries
    - Other libraries
- Configuration
- Sample Code

Description

This SDK allows users to send requests to Dynamic Routing from a custom Java application (ex: Orchestration Designer, or Web Application).

It wraps the use of the Gigaspaces XAP remote method invocation protocol (LRMI, TCP/IP based), isolating the client code from any specific Gigaspaces details.

The resulting client code is very simple and only coupled to the "Dynamic Routing" specific interface and semantics.

> As it uses Spring, Apache Commons, SLF4J, please be careful to **NOT override** the required libraries with different versions.

Dependencies

The following Java dependencies are required for the Routing Remoting Client to work.

Dynamic Routing Libraries

- dr_global_commons-<DR_VERSION>.jar e.g. dr_global_commons-**3.2.8**.jar
- dr_routing_remoting_client-<DR_VERSION>.jar
- dr_routing_comm_listeners-<DR_VERSION>.jar
- commons-logger-<DR_VERSION>.jar

Other libraries

- spring-context-4.1.1.RELEASE.jar
- spring-core-4.1.1.RELEASE.jar
- spring-beans-4.1.1.RELEASE.jar
- spring-expression-4.1.1.RELEASE.jar
- spring-tx-4.1.1.RELEASE.jar
- spring-aop-4.1.1.RELEASE.jar
- commons-logging-1.1.1.jar
- gs-openspaces.jar
- gs-runtime.jar
- aopalliance-1.0.jar
- slf4j-api-1.7.7.jar
- json-io-3.0.2.jar

IMPORTANT: If you require any of these libraries, you can find them within the Dynamic Routing Installation in file: */opt/Avaya/dr/lib/processing-units/dr_routing_rest_pu-<dr_version>.war.* Unzip this file, and look for the jar files here: */dr_routing_rest_pu-<dr-version>/WEB-INF/lib.*

Additionally, libraries can also be found at:

> */opt/Avaya/dr/lib*
>
> */opt/Avaya/third-party/gigaspaces/gigaspaces-xap-premium-10.2.1-ga/lib/required*

Configuration

The Routing Remoting Client relies on two properties used to define the (relative) path to the two data grids used by the application:

- Configuration Store space, through the property ***configStore_space.name***
- Routing Core space, through the property ***routingcore_space.name***

Both relative paths require the configuration of the **lookup services** used to find the data grids. These lookup service addresses are usually the same ones defined at installation time (*LOOKUP LOCATORS*) and can be configured using the suffix *?locators=<locators addresses>*. Taking the Lookup Locators *172.28.128.3:4174* and *172.28.128.4:4174* as example, the properties would be configured as follows:

rclient_communication.properties

```
configStore_space.name=dr_configurationstore_space?locators=172.28.128.3:4
174,172.28.128.4:4174

routingcore_space.name=dr_routing_core_space?locators=172.28.128.3:4174,17
2.28.128.4:4174
```

As we can see in the example above, the lookup locators have to be (manually) set in the *rclient_communication.properties* file, separated by commas. Finally, you have to make sure this file is placed into the application's classpath.

Sample Code

Example of a simple request

```
import java.util.HashMap;
import java.util.Map;

import
com.avaya.ept.dr.services.global.remoting.client.ServiceUnavailableExcepti
on;
import com.avaya.ept.dr.services.routing.model.RoutingServiceRequest;
import com.avaya.ept.dr.services.routing.model.dto.DecisionDTO;
import
com.avaya.ept.dr.services.routing.remoting.client.factory.RoutingRemotingS
erviceFactory;

/**
 * An example of a Dynamic Routing Client Application
 *
 * @author spintos
 */
public class Example {

    public static void main(String[] args) {
        String decisionFunction = "Standard_DF_Custom"; // The name of the
Decision Function
        String interactionID = "123456789"; // An unique identifier for
this request
        String from = "5132288888"; // ANI
        String to = "1868527"; // DNIS
        String channel = "VOICE"; // Possible values: VOICE, CHAT, EMAIL,
```

```
VIDEO, SMS, FACEBOOK, TWITTER, WEB

        // This following Segmentation Attributes should match the
Dimensions of the Segmentation Table
        Map<String, String> segmentationAttributes = new HashMap<String,
String>();
        segmentationAttributes.put("Package", "BASIC");
        segmentationAttributes.put("State", "TX");
        segmentationAttributes.put("Tenure", "10");
        segmentationAttributes.put("Bundle", "NO");

        // Extra Parameters for our request that won't be mapped to the
Dimensions of the Segmentation Table
        Map<String, String> customParameters = new HashMap<String,
String>();
        // In this example I am sending the name of the Segmentation Table
to use
        // The Standard_DF will look for the parameter segTable
        customParameters.put("segTable", "CableTV");

        // The RoutingServiceRequest class is needed for our request
        RoutingServiceRequest request = new
RoutingServiceRequest(decisionFunction, interactionID, from, to,
                customParameters, new HashMap<String, String>(), channel);

        try {
            // This is the request itself
            DecisionDTO decision =
RoutingRemotingServiceFactory.getRoutingService().getBestDestination(reque
st);

            System.out.println(decision); // toString method will give us a
JSON representation

        } catch (ServiceUnavailableException e) {
            e.printStackTrace();
```

```
        }
    }
}
```

RESPONSE SAMPLE


```
{
   "@type":"com.avaya.ept.dr.services.routing.model.dto.DecisionDTO",
   "timestamp":1471891444237,
   "selectedDestId":"4ac6d78d-54c1-4b1a-aa8d-203654b95879",
   "selectedDestNativeId":"1013",
   "selectedDestType":"AGENT_GROUP",
   "selectedDestName":"Basic-Level-1_Dallas_A2",
   "selectedDestAddress":"12101214",
   "extendedResponse":{
     "defaultDest": "21101504",
     "defaultApp": "10000004",
     "playPrompt":"legalAnn-eng.wav"
   },
   "resultCode":10,
   "debugString":"SGds:S",
   "interactionId":"123456789",
   "countDecision":true,
   "locationName":"Dallas",
   "companyName":"Alive",
   "segLabel":"BAS_NEW_NB_TX",
   "segTable":"CableTV",
   "strategySettingsName":null
}
```


This example was based on the Demo Data provided by the Dynamic Routing Installer. Possible result codes are:

Result Code	Description
10	Destination Found
-1	The segmentation rule has no Destination
-2	The segmentation rule has multiple Destinations available but no Strategy Script was provided
-3	Destination was not selected
-4	Segmentation Rule was not found or there was an error while processing the Strategy Script
-13	Address was not selected

DR 3.2 Routing REST Interface

- Description
- Characteristics of the REST Request
- URI Examples
- Example Code Using Apache HTTP Client

Description

This interface allows developers to send **decision requests** to Dynamic Routing from **any Client application**, through the use of a **REST** interface (over the HTTP protocol).

Most IVR and Web application frameworks support building REST clients in an easy way. Examples include: any Java, Ruby or .Net web application, AAEP/OD applications, as well as the majority of IVR application frameworks (both on premise and cloud).

The REST interface requires customers to provide a **Load Balancer** between the client applications and Dynamic Routing nodes.

Characteristics of the REST Request

URI Structure	http://host:port/dr-decision-api/rest/getdestination
HTTP Method	POST
Authentication	N/A
Body content format	JSON
Request headers	"content-type": "application/json"
Query string parameters	

parameter	type	required	default value	description
trackDecision	boolean	no	true	values allowed: true or false, otherwise server will return error code 400 (bad request) if false, Dynamic Routing will not store the decision in the RDR DB and counters will not be updated.

URI Examples

1. http://10.130.124.31/dr-decision-api/rest/getdestination?trackDecision=true
2. http://10.130.124.31/dr-decision-api/rest/getdestination
3. http://10.130.124.31/dr-decision-api/rest/getdestination?trackDecision=false

Note: examples 1 and 2 are the same, since the default value is true.

REQUEST BODY MODEL SCHEMA

```
RoutingServiceRequest {
 decisionFunctionName (string, optional),
 decisionFunctionVersion (string, optional),
 interactionId (string, optional),
 from (string, optional),
 to (string, optional),
 customParameters (object, optional),
 segAttributes (object, optional),
 channel (string, optional)
}
```

Even though none of the above JSON attributes are mandatory to hit Dynamic Routing, the customParameters
and segAttributes objects must be populated at least in the form:

```
{
  "customParameters": {},
  "segAttributes": {}
}
```

This empty JSON object to Dynamic Routing means that it will execute the Decision Function marked as "Default",
with LIVE Status and without any custom parameters nor segmentation attributes.

RESPONSE BODY MODEL SCHEMA

```
DecisionDTO {
    countDecision (boolean, optional),
    selectedDestAddress (string, optional),
    interactionId (string, optional),
    debugString (string, optional),
    locationId (string, optional),
    selectedDestNativeId (string, optional),
    selectedDestName (string, optional),
    locationName (string, optional),
    companyName (string, optional),
    segLabel (string, optional),
    timestamp (string, optional),
    segTable (string, optional),
    selectedDestId (string, optional),
    strategySettingsName (string, optional),
    resultCode (integer, optional),
    extendedResponse (object, optional),
    selectedDestType (string, optional)
}
```

Example Code Using Apache HTTP Client

Libraries used:

- httpclient-4.5.2.jar
- httpmime-4.5.2.jar
- httpclient-cache-4.5.2.jar
- httpclient-win-4.5.2.jar
- httpcore-4.4.4.jar
- commons-logging-1.2.jar
- fluent-hc-4.5.2.jar

Example of a simple request

```
import java.io.IOException;

import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.fluent.Executor;
import org.apache.http.client.fluent.Request;
import org.apache.http.entity.ContentType;

/**
 * This example was done using Apache HTTP Client.
 * Feel free to use any java library that implements the HTTP protocol
 *
 * @author spintos
 *
 */
public class RestExample {

    public static void main(String[] args) throws ClientProtocolException,
IOException {
        Executor executor = Executor.newInstance();

        // Remember to create a POST request, since any other method will
fail. DR would  respond:
        // 405 Request method not supported
        Request request =
Request.Post("http://10.130.125.110:8081/dr_routing_rest_pu/rest/getdestin
ation");

        // I won't wait more than 1 second for Dynamic Routing to respond
        request.connectTimeout(1000);
        // Also, setting the timeout for the TCP/IP connection
        request.socketTimeout(1000);

        // the payload of this request is a json object which I'm
hardcoding.
        String decisionFunction = "\"decisionFunctionName\":
\"Standard_DF_Custom\",";
        String interactionId = "\"interactionId\": \"12345\",";
        String from = "\"from\": \"5132288888\",";
```

```
        String to = "\"to\": \"1868527\",";
        String customParameters = "\"customParameters\":
{\"segTable\":\"CableTV\"},";
        String segAttributes = "\"segAttributes\": {"
                  + "\"Package\":\"BASIC\","
                  + "\"State\":\"TX\","
                  + "\"Tenure\":\"10\","
                  + "\"Bundle\":\"NO\""
                    + "},";
        String channel = "\"channel\": \"VOICE\"";

        // put the JSON object in the request body and set the content-type
to application/json
        request.bodyString("{"
                + decisionFunction
                + interactionId
                + from
                + to
                + customParameters
                + segAttributes
                + channel
              + "}", ContentType.APPLICATION_JSON);

        // The response of this operation will return a JSON object as well
        String response =
executor.execute(request).returnContent().asString();
        System.out.println(response);
```

```
    }
}
```

This is the JSON object that is hard-coded in the example above.

REQUEST BODY

```
{
  "decisionFunctionName": "Standard_DF",
  "interactionId": "12345",
  "from": "5132288888",
  "to": "1868527",
  "customParameters": {"segTable":"CableTV"},
  "segAttributes": {
      "Package":"BASIC",
      "State":"TX",
      "Tenure":"10",
      "Bundle":"NO"
  },
  "channel": "VOICE"
}
```

And this is the JSON object that Dynamic Routing returns as reponse

RESPONSE SAMPLE


```
{
  "timestamp": 1472136804602,
  "selectedDestId": "4ac6d78d-54c1-4b1a-aa8d-203654b95879",
  "selectedDestNativeId": "1013",
  "selectedDestType": "AGENT_GROUP",
  "selectedDestName": "Basic-Level-1_Dallas_A2",
  "selectedDestAddress": "12101214",
  "extendedResponse": {
    "defaultDest": "21101504",
    "defaultApp": "10000004",
    "playPrompt":"legalAnn-eng.wav"
  },
  "resultCode": 10,
  "debugString": "SGds:S",
  "interactionId": "12345",
  "countDecision": true,
  "locationName": "Dallas",
  "companyName": "Alive",
  "segLabel": "BAS_NEW_NB_TX",
  "segTable": "CableTV",
  "strategySettingsName": null
}
```


This example was based on the Demo data provided with Dynamic Routing Installer.

Possible result codes are:

Result Code	Description
10	Destination Found
-1	The segmentation rule has no Destination
-2	The segmentation rule has multiple Destinations available but no Strategy Script was provided
-3	Destination was not selected
-4	Segmentation Rule was not found or there was an error while processing the Strategy Script
-13	Address was not selected