



A MITEL
PRODUCT
GUIDE

Unify OpenScape Contact Center V11

OpenMedia Framework

Programming Guide

10/2024

Notices

The information contained in this document is believed to be accurate in all respects but is not warranted by Mitel Europe Limited. The information is subject to change without notice and should not be construed in any way as a commitment by Mitel or any of its affiliates or subsidiaries. Mitel and its affiliates and subsidiaries assume no responsibility for any errors or omissions in this document. Revisions of this document or new editions of it may be issued to incorporate such changes. No part of this document can be reproduced or transmitted in any form or by any means - electronic or mechanical - for any purpose without written permission from Mitel Networks Corporation.

Trademarks

The trademarks, service marks, logos, and graphics (collectively “Trademarks”) appearing on Mitel’s Internet sites or in its publications are registered and unregistered trademarks of Mitel Networks Corporation (MNC) or its subsidiaries (collectively “Mitel), Unify Software and Solutions GmbH & Co. KG or its affiliates (collectively “Unify”) or others. Use of the Trademarks is prohibited without the express consent from Mitel and/or Unify. Please contact our legal department at iplegal@mitel.com for additional information. For a list of the worldwide Mitel and Unify registered trademarks, please refer to the website: <http://www.mitel.com/trademarks>.

© Copyright 2024, Mitel Networks Corporation

All rights reserved

Contents

1. About the OpenMedia Framework.....	3
OpenMedia Framework solution overview.....	3
Simple OpenMedia flow example	3
Simple OpenMedia External Media flow example	5
2. OpenMedia Framework prerequisites.....	6
3. OpenMedia commands.....	7
Connector Registration	7
OpenMedia Object.....	7
Connector Registration Response.....	8
UserCredentials Object	8
New Contact.....	9
Source Object.....	9
PublishedObject Object.....	10
Destination Object	10
AdditionalInfos Object.....	11
InReplyTo Object.....	11
ContactData Object.....	11
Attachment Object	11
Tag Object.....	11
New Contact Response	13
Keep Alive	13
Keep Alive Response.....	14
Listen For Events	14
Listen For Events Response.....	14
Listen For Events Response Object	14
4. OpenMedia Requests from OpenScape Contact Center to Connector	16
Outgoing Publication	16
Source Object.....	16
ObjectToBePublished Object.....	17
Destination Object	17
InReplyTo Object.....	17
Attachment Object	17
Outgoing Publication Response.....	18
Stream Request	19

Stream Response	20
OrderedItem Object	20
InReplyTo Object	21
Source Object	21
AdditionalInfo Object	21
5. OpenMedia Events	24
DeliveredOpenMediaEvent	24
Source Object	24
PublishedObject Object	25
Destination Object	26
EstablishedEvent	27
HeldOpenMediaEvent	28
Source Object	28
PublishedObject Object	28
Destination Object	29
RetrievedOpenMediaEvent.....	31
Source Object	31
PublishedObject Object	31
Destination Object	32
TransferredOpenMediaEvent.....	34
TransferParty Object	34
DisconnectedEvent	36
DisconnectedParty Object	36
6. Error Code	37
7. Appendix	37
JavaScript and NodeJS code example	37

1. About the OpenMedia Framework

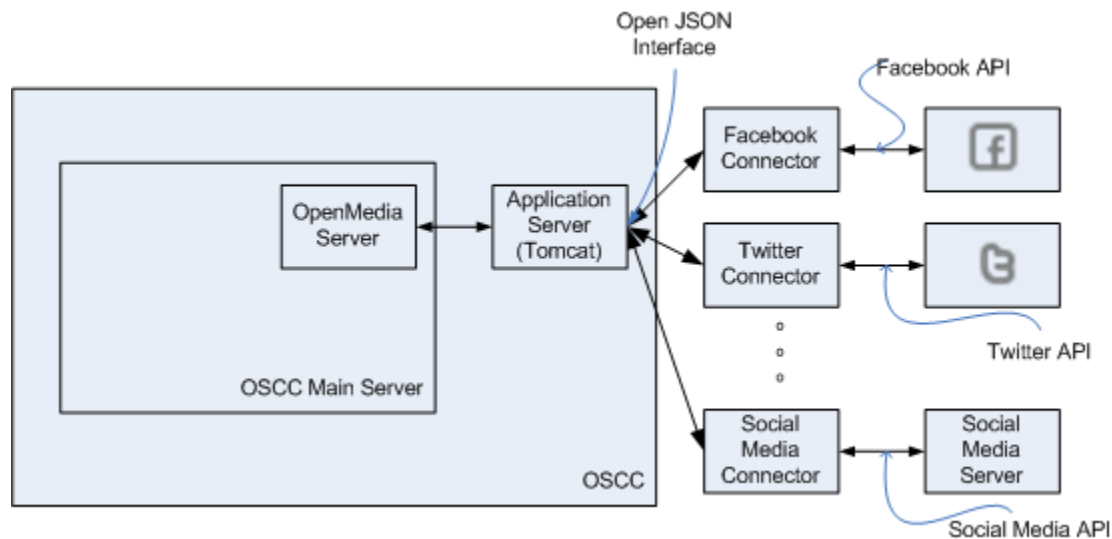
The OpenMedia Framework allows the creation of Connectors, which perform the integration of Corporate Systems and Social Media to the OpenScape Contact Center.

The framework consists of a REST interface, which allows receiving contacts in the OpenScape Contact Center, which will then route the contact to the most appropriate agent.

The framework consists of commands sent from the Corporate System or Social Medium to the OpenScape Contact Center and the other way round.

OpenMedia Framework solution overview

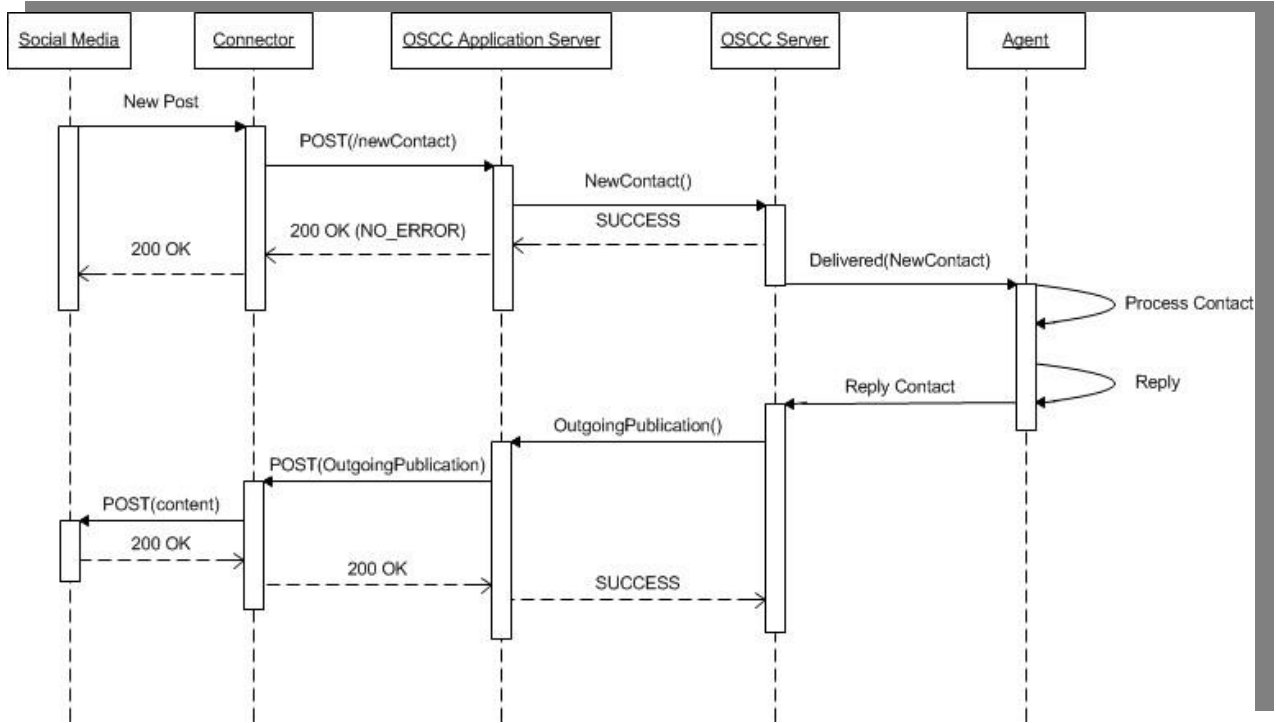
The figure below shows a high level overview of the OpenMedia solution.



Simple OpenMedia flow example

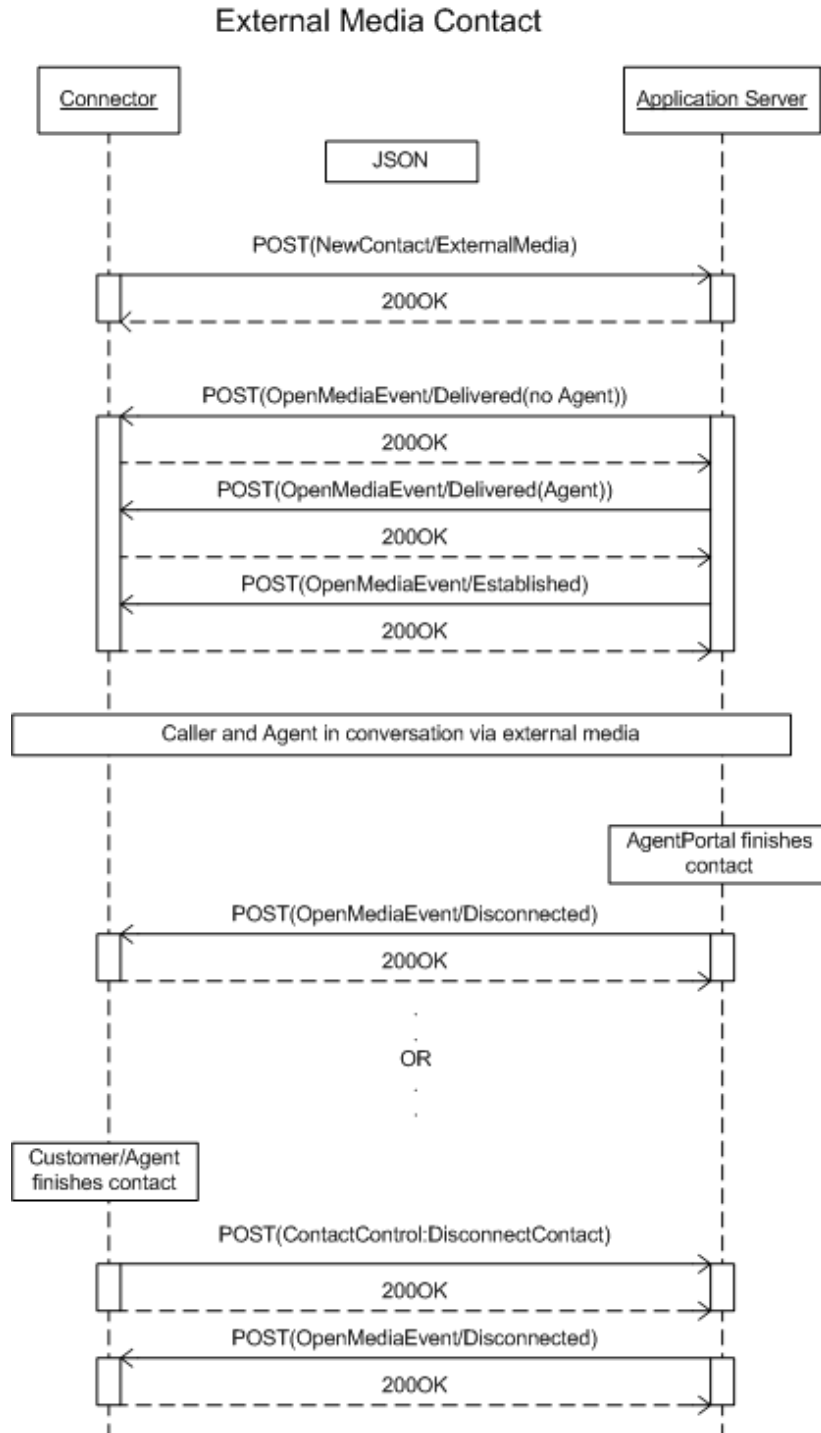
Below we can see a simple sequence flow that demonstrates how the entire system will integrate with each system processes.

Simple New Contact request from the Connector to OSCC system.



Simple OpenMedia External Media flow example

When a new contact is received from a Connector via the OpenMedia interface, with the type ExternalMedia, the contact shall be handled by the OSCC. When the agent is selected, the Agent Portal will get the Contact Details. The external application which is negotiating the payload needs to get status messages from OSCC informing that the agent is selected, the agent started being alerted, the agent has answered to the contact, the agent has finished the contact. If Listen for Events is enabled, the events will be generated for any type of OpenMedia incoming contact.



2. OpenMedia Framework prerequisites

For OpenMedia to work there are some steps that must be performed before using the API.

- Knowledge on REST (Representational state transfer) web services.
- Licenses for OpenMedia connector stored on the server.
- Enable the OpenMedia feature from the Manager Application.
- Create a new **connector** using the Manager Application and generate the connector token.
- Install the **OpenScape Contact Center Application Server** (it can be collocated into OSCC server or into another machine).
- Logon an Agent to the connector by using Agent Portal to be able to handle the contacts.

Note: For more details about the configuration, see *OpenScape Contact Center Manager and Administration Guide*.

Note: For more details about installation, see *OpenScape Contact Center Installation Guide*.

3. OpenMedia commands

OpenMedia commands are requests sent from the **Connector** to the OpenScape Contact Center. The requests are listed below:

Command Name	HTTP Command Type	REST URL and Description
Connector Registration	POST	https://oscchostaddress/openmedia/webapi/main/registerConnector Command used to send the request to register the connector on OpenScape Contact Center.
New Contact	POST	https://oscchostaddress/openmedia/webapi/main/newContact Command used to create new contacts to OpenScape Contact Center.
Keep Alive	POST	https://oscchostaddress/openmedia/webapi/main/keepalive Command used to keep the session between the Connector and the OpenScape Contact Center alive.

Connector Registration

Object used to register the Connector to OpenMedia Server.

The Connector Registration process will enable the connectivity to OpenMedia service. Also will return the authorization **session token** to be used on the other REST requests to OSCC.

Connector Registration Request Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of the request. The string value supported by the OpenScape Contact Center is: "Registration" .
webhookURL	String	The webhookURL is used to send requests from the OpenScape Contact Center to the Connector.
openMedia	Object	OpenMedia object is contains the credentials which are configured in the OpenScape Contact Center. Note: The configuration must be aligned on what is configured on the OpenScape Contact Center server. See the configuration on Manager application. See the openMedia object definition below.

OpenMedia Object		
Attribute name	Attribute type	Description
openMediaTitle	String	Consists of the name of the connector and must match the name configured in Manager. Note: Containing up to 50 characters.
token	String	Generated at the Manager and must be copied to this attribute. Note: Containing up to 50 characters.

Connector Registration Request JSON body example:

This is a full example of the connectorRegistration object that can be sent to OpenScape Contact Center.

```
{
  "type": "Registration",
```

```

    "webhookURL": "https://connectorhostaddress/connectorpath",
    "openMedia": {
        "openMediaTitle": "facebook",
        "token": "HWJVUAzT1BGx3vBmnwUVUMAMZHkGWX"
    }
}

```

Connector Registration Response

The response after the registration request. The response is a JSON object returned **synchronously** by OSCC with the following data:

Connector Registration Response Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of the response. The string value returned by the OpenScape Contact Center is: Registration .
errorCode	Integer	Indicates the returned error code number. See error code enum definition below.
errorText	String	Indicates the returned error code text. See error code enum definition below.
sessionToken	String	Session token for a registration request. The connector must use this session token for other commands. Important: all the commands sent to OpenScape Contact Center must have this session token on the HTTP header as Authorization . The HTTP Header must have for example: Content-Type = application/json Authorization = "session token returned by OpenScape Contact Center"
userCredentials	Object	Contains the credentials which allow the Connector to authenticate against the Corporate System / Social Media. See the userCredentials object definition below.

UserCredentials Object		
Attribute name	Attribute type	Description
userName	String	User name part of the credentials.
password	String	Password part of the credentials.

Connector Registration Response JSON body example:

```

{
    "type": "Registration",
    "errorCode": 0,
    "errorText": "NO_ERROR",
    "sessionToken":
"a5e272bcfa964c6dea958b583da4e721703bb2a74bff8533a5818ccaedd71669",
    "userCredentials": {
        "userName": "user name for social media",
        "password": "string"
    }
}

```

New Contact

Object used to create a new contact to OpenScape Contact Center.

Important: For New Contact requests, the HTTP must have the following **headers**:

- Content-Type = application/json
- Authorization = "session token returned by OSCC"

New Contact Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of publication. The string values supported by the OpenScape Contact Center are: " Post ", " DirectMessage ", " Article ", " Ticket ", " ExternalMedia ".
dateTime	String	Indicates the date and time of the creation of the published post/message. Note: Use UTC Date/Time format to create the dateTime string before sending to OpenScape Contact Center. This field is shown at the Agent Portal. JavaScript example: Date().toUTCString();
source	Object	Contains the information about the person or organization that posted the message to the corporate system or the social media. See the source object definition below.
publishedObject	Object	Contains all the information about the content posted to the corporate system or to the social media. See the publishedObject definition below.
destination	Object	Defines the place on which the message was published. One example is a page on which the source has published the message. See the destination object definition below.
isRealTimeHandling	Boolean	This flag defines if the message has a real-time contact characteristic or not. Note: This flag changes the way on how the Agent user will handle the contact. If the value is true, the handling will be like a chat contact, that means it will be a continuous conversation session and the contact will remain active until one of the sides decides to finish it, if the value is false, the contact will be finished when the Agent replies to the message.

Source Object		
Attribute name	Attribute type	Description
id	String	Identifies of the source (From) of the contact. Note: This field can be used for routing at the Manager, through Design Center in Source/Destination component. Also is used to match the source for the 360° identification of the customer. Containing up to 256 characters.
type	String	Defines if the source is a Person or Organization. The string values supported by the OpenScape Contact Center are " Person ", " Organization ".
name	String	Defines the name of the source that means the name of the person or organization. Note: The name is shown at the Agent Portal. Containing up to 256 characters.

location	String	(Optional). Contains information about the location of the source who sent the message. Note: Containing up to 256 characters.
language	String	(Optional). Contains the language used by the source. Example: "en". Note: Containing up to 256 characters.
additionalInfos	Array of Objects	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector. Note: the key/values are not used by OpenScape Contact Center, but will be returned to the connector when a reply is created for the message. The connector can use the values for its own purposes. The objects in the list must have the attributes "key" and "value". See the additionalInfos object definition below.

PublishedObject Object		
Attribute name	Attribute type	Description
id	String	Contains the published message on the corporate system or social media. Note: Containing up to 256 characters.
fatherId	String	Identifies the parent of the post or comment on the media. Example: On Facebook, one post in the timeline can have comments. The father ID is the Original post identification and the "id" is the comment identification. Note: This object is very important for the stream and the realtime contact handling. Containing up to 256 characters.
inReplyTo	Object	Contains the identification value if it is a reply to another post or comment. See the inReplyTo object definition below.
title	String	Contains the title of the comment/post. This data is shown on the Contact Detail view for the Agent. Note: Containing up to 256 characters.
content	String	Contains the text content of the post/comment sent by the source. Note: Containing up to 5000 characters.
contactData	Array of Objects	List of contactData key/value pairs which may carry customized data about the contact. Note: This list can be used by OpenScape Contact Center to define routing strategies according to the input key/values. This field is also shown at the Agent Portal. See the contactData object definition below.
attachments	Array of Objects	List of attachment objects that contain information about files attached to the message on the corporate system/social media. Note: The Agent Portal will show only if there's any attachment. See the attachments object definition below.
tagList	Array of Objects	List of tag objects that contain tag data depending on the social media. Tags are keywords used by social media to classify a post or mark some subject. See the tag object definition below.

Destination Object		
Attribute name	Attribute type	Description
id	String	Identifies the destination (To) of the contact. Note: This field can be used for routing at the Manager, through Design Center in Source/Destination component. Containing up to

		256 characters.
type	String	Defines if the destination is a User or Page. The string values supported by the OpenScape Contact Center are "User" , "Page" .
name	String	Define the name of the destination. The name of the user or page. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
URL	String	(Optional). Contain information about the URL of the destination where the message were sent.
additionalInfos	Array of Objects	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector. Note: the key/values are not used by OpenScape Contact Center, but will be returned to the connector when a reply is created for the message. The connector can use the values for its own purposes. The objects in the list must have the attributes "key" and "value". See the additionalInfos object definition below.

AdditionalInfos Object		
Attribute name	Attribute type	Description
key	String	Key for connector purposes.
value	String	Value for connector purposes.

InReplyTo Object		
Attribute name	Attribute type	Description
id	String	Identification of the reply from a comment. Note: Containing up to 256 characters.

ContactData Object		
Attribute name	Attribute type	Description
key	String	Key for the contact data.
value	String	Value for the contact data.

Attachment Object		
Attribute name	Attribute type	Description
type	String	Indicates the media type of the file. The string values supported by the OpenScape Contact Center are: "Image" , "Video" , "Audio" , "Document" .
content	String	(Optional) Description of the attachment.
url	String	URL of the file. Example: http://www.example.com/cat.jpeg Note: The URL will be used by the OpenScape Contact Center to download the file during a contact handling.

Tag Object		
Attribute name	Attribute type	Description
tag	String	Example: "#hashtag" .

New Contact JSON body Example:

Here there is a full example of the NewContact object to send to OpenScape Contact Center.

```

{
  "type": "Post",
  "dateTime": "2016-09-10T15:04:55Z",
  "source": {
    "id": "source id",
    "type": "Person",
    "name": "name surname",
    "location": "Town, Country",
    "language": "en",
    "additionalInfo": [
      {
        "key": "info1",
        "value": "value1"
      },
      {
        "key": "info2",
        "value": "value2"
      }
    ]
  },
  "publishedObject" : {
    "id": "Object id",
    "fatherId": "Object id",
    "inReplyTo": {
      "id": "Id of post for which this object is a reply or a
comment."
    },
    "title": "Title of the document",
    "content": "This field contains the content of the post",
    "contactData": [
      {
        "key": "key1",
        "value": "value1"
      },
      {
        "key": "key2",
        "value": "value2"
      }
    ],
    "attachments": [
      {
        "type": "Image",
        "content": "Description of the attachment",
        "url": "http://www.example.com/cat.jpeg"
      }
    ],
    "tagList": [
      {
        "tag": "#hashtag"
      },
      {
        "tag": "@mention"
      }
    ]
  },
  "destination" : {
    "id": "Page id",
    "type": "Page",
    "name": "Page Name",
    "URL": "Page address",

```

```

        "additionalInfo": [
            {
                "key": "info1",
                "value": "value1"
            },
            {
                "key": "info2",
                "value": "value2"
            }
        ],
        "isRealTimeHandling": false
    }
}

```

New Contact Response

Response to newContact. The response is a JSON object returned **synchronously** by OpenScape Contact Center with the following data:

New Contact Response Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of the response. The string values supported by the OpenScape Contact Center are: “Post” , “DirectMessage” , “Article” , “Ticket” , “ExternalMedia” .
errorCode	Integer	Indicates the returned error code number. See error code enum definition below.
errorText	String	Indicates the returned error code text. See error code enum definition below.

New Contact Response JSON body example:

```

{
    "type": "Post",
    "errorCode": 0,
    "errorText": "NO_ERROR"
}

```

Keep Alive

The HTTP REST command is used to keep the connection between the Connector and the OpenScape Contact Center alive.

Important: In the Keep Alive request the HTTP request must be sent with the following headers:

- Content-Type = application/json
- Authorization = “session token returned by OSCC”

Note: send an empty JSON object.

Keep Alive JSON body example:

```

{}

```

Keep Alive Response

Response after every request. The response is a JSON object returned **synchronously** by OpenScape Contact Center with the following data:

Keep Alive Response Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of the response. The string value supported by the OpenScape Contact Center is: "keepAlive" .
errorCode	Integer	Indicates the returned error code number. See error code enum definition below.
errorText	String	Indicates the returned error code text. See error code enum definition below.

Keep Alive Response JSON body example:

```
{
  "type": "keepAlive",
  "errorCode": 0,
  "errorText": "NO_ERROR"
}
```

Listen For Events

If Listen for Events is enabled, the events will be generated for any type of OpenMedia incoming contact.

Listen For Events Request Object		
Attribute name	Attribute type	Description
enabledListenForEvent	Boolean	Indicates the state function "ListenForEvent"

Listen For Events Object Request JSON body example:

This is a full example of the connectorRegistration object that can be sent to OpenScape Contact Center.

```
{
  "enabledListenForEvent": true
}
```

Listen For Events Response

Response after every request. The response is a JSON object returned **synchronously** by OpenScape Contact Center with the following data

Listen For Events Response Object		
Attribute name	Attribute type	Description
errorCode	Integer	Indicates the returned error code number. See error code enum definition below.

errorText	String	Indicates the returned error code text. See error code enum definition below.
------------------	--------	--

Connector Registration Response JSON body example:

```
{  
  "errorCode": 0,  
  "errorText": "NO_ERROR"  
}
```

4. OpenMedia Requests from OpenScape Contact Center to Connector

Objects sent by OpenScape Contact Center to the Connector on the “Web Hook URL” sent by the Connector during the registration.

Object Name	REST URL example / Description
Outgoing Publication	webhookURL sent to OpenScape Contact Center during registration. Defines a Reply or an Outgoing publication sent by the OpenScape Contact Center to be posted on the external media by the Connector.
Stream Request	webhookURL sent to OpenScape Contact Center during registration. Request from OpenScape Contact Center to get the stream or the history of the contact handled by an Agent.

Outgoing Publication

Object used to publish (from the OpenScape Contact Center) the object to the destination (Connector).

Outgoing Publication Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of publication. The string values supported by the OpenScape Contact Center are: “ Post ”, “ DirectMessage ”, “ Article ”, “ Ticket ”, “ ExternalMedia ”. Note: the type is defined by the New Contact sent by the connector.
source	Object	Contains the information about the person or organization that is posting the message to the corporate system or social media. See the source object definition below.
objectToBePublished	Object	Contains all the information about the content to be posted to the social media or other media. See the objectToBePublished object definition below.
destination	Object	Defines the place on which the message shall be published. One example is a page on which the message shall be published. See the destination object definition below.

Source Object		
Attribute name	Attribute type	Description
id	String	Identifies of the source (From) of the contact.
type	String	Defines if the source is a Person or Organization. The string values supported by the OpenScape Contact Center are “ Person ”, “ Organization ”.
name	String	Defines the name of the source that means the name of the person or organization. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
location	String	(Optional). Contains information about the location of the source who sent the message. Note: Containing up to 256 characters.
language	String	(Optional). Contains the language used by the source. Example:

		“en”. Note: Containing up to 256 characters.
additionalInfos	Array of Objects	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector. Note: the key/values are not used by OpenScape Contact Center, but will be returned to the connector when a reply is created for the message. The connector can use the values for its own purposes. The objects in the list must have the attributes “key” and “value”. See the additionalInfos object definition below.

ObjectToBePublished Object		
Attribute name	Attribute type	Description
id	String	Identifies the post or comment on the media.
inReplyTo	Object	Contains the identification of the message for which this outgoing message is a replay. See the inReplyTo object definition below.
title	String	Contains the title of the post/message.
content	String	Contain the text content of the post/message sent by OpenScape Contact Center.
attachments	Array of Objects	Contains information about files attached to the post/message. See the attachments object definition below.

Destination Object		
Attribute name	Attribute type	Description
id	String	Identifies the destination (To) of the contact. Note: This field can be used for routing at the Manager, through Design Center in Source/Destination component. Containing up to 256 characters.
type	String	Defines if the destination is a User or Page. The string values supported by the OpenScape Contact Center are “ User ”, “ Page ”.
name	String	Define the name of the destination. The name of the user or page. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
URL	String	(Optional). Contain information about the URL of the destination where the message were sent.
additionalInfos	Array of Objects	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector. Note: the key/values are not used by OpenScape Contact Center, but will be returned to the connector when a reply is created for the message. The connector can use the values for its own purposes. The objects in the list must have the attributes “key” and “value”. See the additionalInfos object definition below.

InReplyTo Object		
Attribute name	Attribute type	Description
id	String	Identifies the reply from a comment.

Attachment Object		
Attribute name	Attribute type	Description
type	String	Indicates the media type of the file. The string values supported by the OpenScape Contact Center are: “ Image ”, “ Video ”, “ Audio ”, “ Document ”.
content	String	(Optional) Contains the description of the attachment.

url	String	Note: attachment from OpenScape Contact Center not supported.
------------	--------	--

Outgoing Publication JSON body example:

Here there is a full example of the outgoingPublication object to send to the OpenScape Contact Center.

```
{
  "type": "Post",
  "dateTime": "2016-09-10T15:04:55Z",
  "source": {
    "id": "source id",
    "type": "Person",
    "name": "name surname",
    "location": "Town, Country",
    "language": "en",
    "additionalInfo": []
  },
  "objectToBePublished" : {
    "id": "Object id",
    "inReplyTo": {
      "id": "Id of post for which this object is a reply or a
comment."
    },
    "title": "Title of the document",
    "content": "This field contains the content of the publication",
    "attachments": []
  },
  "destination" : {
    "id": "Page id",
    "type": "Page",
    "name": "Page Name",
    "URL": "Page address",
    "additionalInfo": [
      {
        "key": "info1",
        "value": "value1"
      }
    ]
  }
}
```

Outgoing Publication Response

Response after Outgoing Publication is received by the connector. The response is a JSON object returned **synchronously** by the Connector to the OpenScape Contact Center with the following data:

Outgoing Publication Response Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of the response. The string values supported by the OpenScape Contact Center are: “Post” , “DirectMessage” , “Article” , “Ticket” .
errorCode	Integer	Indicates the returned error code number. See error code enum definition below.
errorText	String	Indicates the returned error code text. See error code enum definition below.

Outgoing Publication Response JSON body example:

Here there is a full example of the Outgoing Publication Response object to be sent to the OpenScope Contact Center.

```
{  
  "type": "Post",  
  "errorCode": 0,  
  "errorText": "NO_ERROR"  
}
```

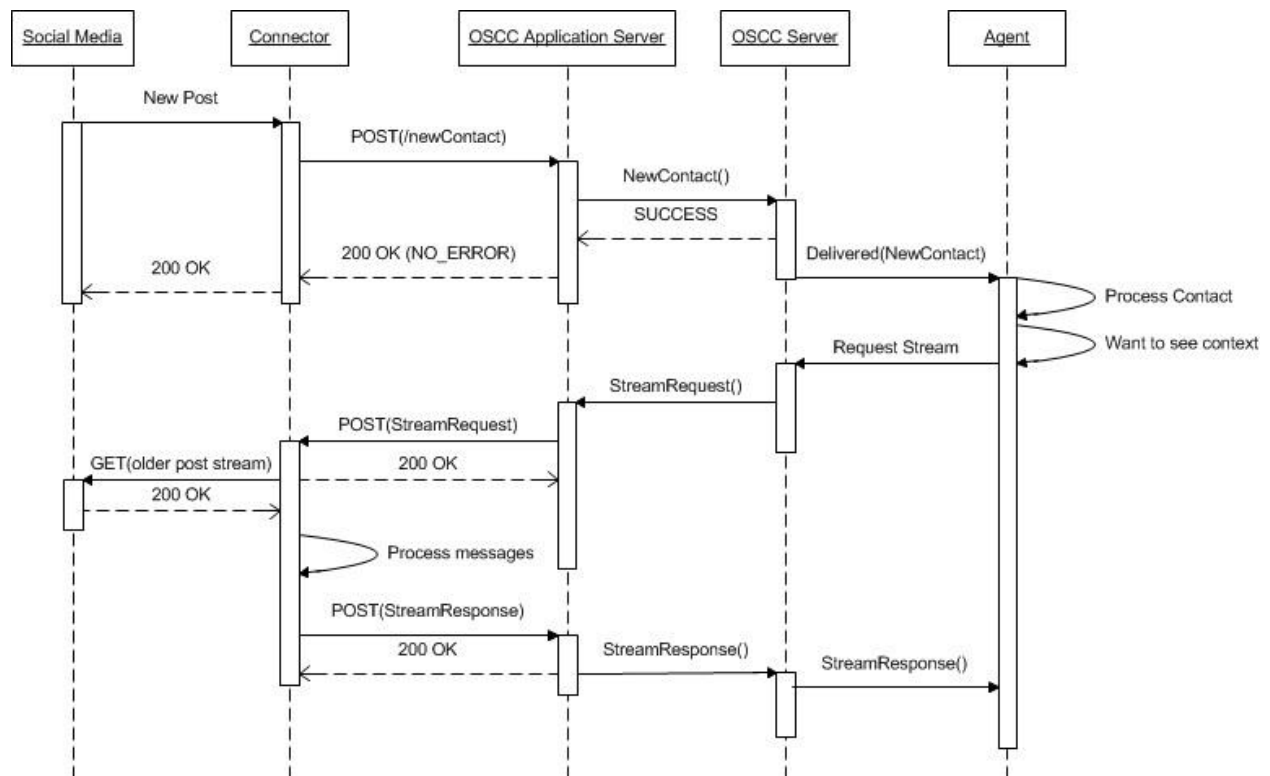
Stream Request

The Stream Request is used to request from the corporate system or from the social media for the messages which are part of the stream.

A useful use case for this request from OpenScope Contact Center is when an Agent receives a contact and needs to know the context in which the message was posted. Then the agent is able to request the stream of which the contact is part.

Basically OpenScope Contact Center will send a POST request with StreamRequest object and the connector will process the request. If there are older messages the connector will send a POST back with StreamResponse object fulfilled.

See the example sequence below:



Important: After receiving this request, the connector must handle it asynchronously. The Stream Response object needs to be sent to URL:

<https://oscchostaddress/openmedia/webapi/main/streamResponse>

Stream Request Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of the request. The string value supported by the OpenScape Contact Center is: "Stream" .
destinationId	String	Identifies the destination (To) of the contact.
id	String	Identifies the post or message on the corporate system or social media.
numberOfMessages	String	Contains the number of messages the object shall retrieve.

Stream Request JSON body example:

Here there is a full example of the streamRequest object to send to OpenScape Contact Center.

```
{
  "type": "Stream",
  "destinationId": "Page id",
  "id": "205056409590639",
  "numberOfMessages": "10"
}
```

Stream Response

Object to be sent to OpenScape Contact Center with the stream response definition. The HTTP request must be a POST to URL: <https://oscchostaddress/openmedia/webapi/main/streamResponse>

Response after stream request. The response is a JSON object with the following data:

Stream Response Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of the response. The string value supported by the OpenScape Contact Center is: "Stream" .
errorCode	Integer	Indicates the returned error code number. See error code enum definition below.
errorText	String	Indicates the returned error code text. See error code enum definition below.
id	String	Identifies the source (From) of the contact.
page	Integer	Indicates how many pages the stream will retrieve.
lastPage	String	Indicates if the page requested is the last page.
itemsinpage	Integer	Indicates how many replies/comments are in the page.
totalItems	Integer	Contains the number of replies/comments that will be retrieved in total.
orderedItems	Array of Objects	Contains all items retrieved. See the orderedItem object definition below.

OrderedItem Object		
Attribute name	Attribute type	Description
id	String	Identifies the post or comment on the media.

fatherId	String	Identifies the parent of the post or comment on the media. Example: On Facebook, one post in the timeline can have comments. The father ID is the Original post identification and the "id" is the comment identification. Note: This object is very important for the stream and the realtime contact handling.
inReplyTo	Object	Contains identification value if it is a reply to another post or comment. See the inReplyTo object definition below.
type	String	Indicates the type of the response. The string values supported by the OpenScape Contact Center are: "Post" , "DirectMessage" .
dateTime	String	Indicates the date and time creation of the published post/message. Note: Use UTC Date/Time format to create the dateTime string before send to OpenScape Contact Center. JavaScript example: Date().toUTCString();
source	Object	Contains the information about the person or organization that sends the message to the media. See the source object definition below.
content	String	Contains the text content of the post/comment sent by the source.

InReplyTo Object		
Attribute name	Attribute type	Description
id	String	Identifies the reply from a comment.

Source Object		
Attribute name	Attribute type	Description
type	String	Defines whether the source is a Person or Organization. The string values supported by the OpenScape Contact Center are "Person" , "Organization" .
name	String	Defines the name of the source. The name of the person or organization. Note: The name is shown at the Agent Portal.

AdditionalInfo Object		
Attribute name	Attribute type	Description
key	String	Key for connector purposes.
value	String	Value for connector purposes.

Stream Response JSON body example:

```
{
  "type": "Stream",
  "errorCode": 0,
  "errorText": "NO_ERROR",
  "id": "205056409590639_124328589355467",
  "page": 1,
  "lastPage": "true",
```

```
"itemsinpage": 3,
"totalItems": 3,
"orderedItems": [
  {
    "id": "205056409590639_537658564310000",
    "fatherId": "205056409590639_124328589355467",
    "inReplyTo": {
      "id": "205056409590639_124328589355467"
    },
    "type": "Post",
    "dateTime": "2016-09-10T13:15:00Z",
    "source": {
      "type": "Person",
      "name": "John Doe"
    },
    "content": "Please provide more detail about your problem."
  },
  {
    "id": "205056409590639_124328589310001",
    "fatherId": "205056409590639_124328589355467",
    "inReplyTo": {
      "id": "205056409590639_124328589355467"
    },
    "type": "Post",
    "dateTime": "2016-09-10T13:04:55Z",
    "source": {
      "type": "Person",
      "name": "Martin Smith"
    },
    "content": "This was the 1st comment to the complaint."
  },
  {
    "id": "205056409590639_740548589310002",
    "fatherId": "205056409590639_124328589355467",
```

```
"inReplyTo": {
  "id": "205056409590639_124328589355467"
},
"type": "Post",
"dateTime": "2016-09-10T12:04:55Z",
"source": {
  "type": "Person",
  "name": "Brian Jameson"
},
"content": "This is the original complaint."
}
]
}
```

5. OpenMedia Events

Objects sent from OpenScape Contact Center to the Connector on the “Web Hook URL” sent by the Connector after function listenForEvents is enable.

Object Name	Description
DeliveredOpenMediaEvent	The DeliveredEvent object is sent when a contact is delivered to a monitored device in the system.
EstablishedEvent	The EstablishedEvent object is sent whenever a contact that is in the system is connected with a user.
HeldOpenMediaEvent	The HeldEvent object is sent whenever a contact that is in the system is placed on hold.
RetrievedOpenMediaEvent	The RetrievedEvent object is sent whenever a contact that has been placed on hold is retrieved from hold.
TransferredOpenMediaEvent	The TransferredEvent is sent whenever a contact is transferred from one device (for example, a user) to another device.
DisconnectedEvent	The DisconnectedEvent object is sent whenever a party that is on a contact in the system disconnects (hangs-up).
DivertedOpenMediaEvent	

DeliveredOpenMediaEvent

DeliveredOpenMediaEvent Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of Event. To all Independent Media Events the type is OpenMediaEvent
openMediaEventType	String	Indicate type of Event. DeliveredOpenMediaEvent
dateTime	String	Indicates the date and time of the creation of the published post/message. Note: Use UTC Date/Time format to create the dateTime string before sending to OpenScape Contact Center. This field is shown at the Agent Portal. JavaScript example: <code>Date().toUTCString();</code>
contactID	String	Identify the contact who will be routed by OSCC. This is unique in all events to same user.
agentID	String	Identify the agent who receive the request.
source	Object	Contains the information about the person or organization that is posting the message to the corporate system or social media. See the source object definition below.
publishedObject	Object	Contains all the information about the content to be posted to the social media or other media. See the objectToBePublished object definition below.
destination	Object	Defines the place on which the message shall be published. One example is a page on which the message shall be published. See the destination object definition below.

Source Object		
Attribute name	Attribute type	Description
id	String	Identifies of the source (From) of the contact.

type	String	Defines if the source is a Person or Organization. The string values supported by the OpenScape Contact Center are “Person” , “Organization” .
name	String	Defines the name of the source that means the name of the person or organization. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
location	String	(Optional). Contains information about the location of the source who sent the message. Note: Containing up to 256 characters.
language	String	(Optional). Contains the language used by the source. Example: “en”. Note: Containing up to 256 characters.
additionalInfos	Array of Strings	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector.

PublishedObject Object		
Attribute name	Attribute type	Description
id	String	Contains the published message on the corporate system or social media. Note: Containing up to 256 characters.
fatherId	String	Identifies the parent of the post or comment on the media. Example: On Facebook, one post in the timeline can have comments. The father ID is the Original post identification and the “id” is the comment identification. Note: This object is very important for the stream and the realtime contact handling. Containing up to 256 characters.
inReplyTo	Object	Contains the identification value if it is a reply to another post or comment. See the inReplyTo object definition below.
title	String	Contains the title of the comment/post. This data is shown on the Contact Detail view for the Agent. Note: Containing up to 256 characters.
content	String	Contains the text content of the post/comment sent by the source. Note: Containing up to 5000 characters.
contactData	Array of Objects	List of contactData key/value pairs which may carry customized data about the contact. Note: This list can be used by OpenScape Contact Center to define routing strategies according to the input key/values. This field is also shown at the Agent Portal. See the contactData object definition below.
attachments	Array of Objects	List of attachment objects that contain information about files attached to the message on the corporate system/social media. Note: The Agent Portal will show only if there’s any attachment. See the attachments object definition below.
tagList	Array of Objects	List of tag objects that contain tag data depending on the social media. Tags are keywords used by social media to classify a post or mark some subject. See the tag object definition below.

Destination Object		
Attribute name	Attribute type	Description
id	String	Identifies the destination (To) of the contact. Note: This field can be used for routing at the Manager, through Design Center in Source/Destination component. Containing up to 256 characters.
type	String	Defines if the destination is a User or Page. The string values supported by the OpenScape Contact Center are "User" , "Page" .
name	String	Define the name of the destination. The name of the user or page. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
URL	String	(Optional). Contain information about the URL of the destination where the message were sent.
additionalInfos	Array of String	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector.

DeliveredOpenMediaEvent JSON body example:

```
{
  "type": "OpenMediaEvent",
  "openMediaEventType": "DeliveredOpenMediaEvent",
  "dateTime": "2018-02-20 15:08:40.994",
  "contactID": "O553A8C5A0300",
  "agentID": "100",
  "source": {
    "id": "703545976517482_703546043184142",
    "type": "Organization",
    "name": "John Smith",
    "location": "Brasilia, Brasil",
    "language": "pt",
    "additionalInfo": [
    ]
  },
  "publishedObject": {
    "id": "703545976517482_703546026517477",
    "fatherId": "205056409590639_124328589355467",
    "inReplyTo": null,
    "title": "Sales Testing",
  }
}
```

```

    "content": "",
    "contactData": null,
    "attachments": null,
    "tagList": null
  },
  "destination": {
    "id": "Node",
    "type": null,
    "name": "",
    "url": "",
    "additionalInfo": [
    ]
  }
}

```

EstablishedEvent

EstablishedEvent Object		
Attribute name	Attribute type	Description
type	String	Indicates the type of Event. To all Independent Media Events the type is OpenMediaEvent
openMediaEventType	String	Indicate type of Event. EstablishedEvent
dateTime	String	Indicates the date and time of the creation of the published post/message. Note: Use UTC Date/Time format to create the dateTime string before sending to OpenScape Contact Center. This field is shown at the Agent Portal. JavaScript example: Date().toUTCString();
contactID	String	
agentID	String	

EstablishedEvent JSON body example:

```

{
  "type": "OpenMediaEvent",
  "openMediaEventType": "EstablishedEvent",
  "dateTime": "2018-02-20 15:08:41.043",
  "contactID": "O553A8C5A0300",
  "agentID": "100"
}

```

HeldOpenMediaEvent

HeldOpenMediaEventObject		
Attribute name	Attribute type	Description
type	String	Indicates the type of Event. To all Independent Media Events the type is OpenMediaEvent
openMediaEventType	String	Indicate type of Event. HeldOpenMediaEvent
dateTime	String	Indicates the date and time of the creation of the published post/message. Note: Use UTC Date/Time format to create the dateTime string before sending to OpenScape Contact Center. This field is shown at the Agent Portal. JavaScript example: Date().toUTCString();
contactID	String	
agentID	String	
source	Object	Contains the information about the person or organization that is posting the message to the corporate system or social media. See the source object definition below.
publishedObject	Object	Contains all the information about the content to be posted to the social media or other media. See the objectToBePublished object definition below.
destination	Object	Defines the place on which the message shall be published. One example is a page on which the message shall be published. See the destination object definition below.
heldReason	String	

Source Object		
Attribute name	Attribute type	Description
id	String	Identifies of the source (From) of the contact.
type	String	Defines if the source is a Person or Organization. The string values supported by the OpenScape Contact Center are " Person ", " Organization ".
name	String	Defines the name of the source that means the name of the person or organization. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
location	String	(Optional). Contains information about the location of the source who sent the message. Note: Containing up to 256 characters.
language	String	(Optional). Contains the language used by the source. Example: "en". Note: Containing up to 256 characters.
additionalInfos	Array of Strings	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector.

PublishedObject Object		
Attribute name	Attribute type	Description
id	String	Contains the published message on the corporate system or social media. Note: Containing up to 256 characters.
fatherId	String	Identifies the parent of the post or comment on the media. Example: On Facebook, one post in the timeline can have comments. The

		father ID is the Original post identification and the “id” is the comment identification. Note: This object is very important for the stream and the realtime contact handling. Containing up to 256 characters.
inReplyTo	Object	Contains the identification value if it is a reply to another post or comment. See the inReplyTo object definition below.
title	String	Contains the title of the comment/post. This data is shown on the Contact Detail view for the Agent. Note: Containing up to 256 characters.
content	String	Contains the text content of the post/comment sent by the source. Note: Containing up to 5000 characters.
contactData	Array of Objects	List of contactData key/value pairs which may carry customized data about the contact. Note: This list can be used by OpenScape Contact Center to define routing strategies according to the input key/values. This field is also shown at the Agent Portal. See the contactData object definition below.
attachments	Array of Objects	List of attachment objects that contain information about files attached to the message on the corporate system/social media. Note: The Agent Portal will show only if there’s any attachment. See the attachments object definition below.
tagList	Array of Objects	List of tag objects that contain tag data depending on the social media. Tags are keywords used by social media to classify a post or mark some subject. See the tag object definition below.

Destination Object		
Attribute name	Attribute type	Description
id	String	Identifies the destination (To) of the contact. Note: This field can be used for routing at the Manager, through Design Center in Source/Destination component. Containing up to 256 characters.
type	String	Defines if the destination is a User or Page. The string values supported by the OpenScape Contact Center are “User”, “Page”.
name	String	Define the name of the destination. The name of the user or page. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
URL	String	(Optional). Contain information about the URL of the destination where the message were sent.
additionalInfos	Array of String	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector.

HeldOpenMediaEvent JSON body example:

```
{
  "type": "OpenMediaEvent",
  "openMediaEventType": "HeldOpenMediaEvent",
```

```
"dateTime":"2018-02-20 15:09:14.054",
"contactID":"O553A8C5A0300",
"agentID":"100",
"source":{
  "id": "",
  "type": "",
  "name": "",
  "location": "",
  "language": "",
  "additionalInfo": [
  ]
},
"publishedObject":{
  "id": "",
  "fatherId": "",
  "inReplyTo": null,
  "title": "",
  "content": "",
  "contactData": null,
  "attachments": null,
  "tagList": null
},
"destination":{
  "id": "",
  "type": null,
  "name": "",
  "url": "",
  "additionalInfo": [
  ]
},
"heldReason": "0"
}
```

RetrievedOpenMediaEvent

HeldOpenMediaEventObject		
Attribute name	Attribute type	Description
type	String	Indicates the type of Event. To all Independent Media Events the type is OpenMediaEvent
openMediaEventType	String	Indicate type of Event. RetrievedOpenMediaEvent
dateTime	String	Indicates the date and time of the creation of the published post/message. Note: Use UTC Date/Time format to create the dateTime string before sending to OpenScape Contact Center. This field is shown at the Agent Portal. JavaScript example: Date().toUTCString();
contactID	String	
agentID	String	
source	Object	Contains the information about the person or organization that is posting the message to the corporate system or social media. See the source object definition below.
publishedObject	Object	Contains all the information about the content to be posted to the social media or other media. See the objectToBePublished object definition below.
destination	Object	Defines the place on which the message shall be published. One example is a page on which the message shall be published. See the destination object definition below.

Source Object		
Attribute name	Attribute type	Description
id	String	Identifies of the source (From) of the contact.
type	String	Defines if the source is a Person or Organization. The string values supported by the OpenScape Contact Center are "Person" , "Organization" .
name	String	Defines the name of the source that means the name of the person or organization. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
location	String	(Optional). Contains information about the location of the source who sent the message. Note: Containing up to 256 characters.
language	String	(Optional). Contains the language used by the source. Example: "en". Note: Containing up to 256 characters.
additionalInfos	Array of Strings	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector.

PublishedObject Object		
Attribute name	Attribute type	Description
id	String	Contains the published message on the corporate system or social media. Note: Containing up to 256 characters.
fatherId	String	Identifies the parent of the post or comment on the media. Example: On Facebook, one post in the timeline can have comments. The father ID is the Original post identification and the "id" is the comment

		identification. Note: This object is very important for the stream and the realtime contact handling. Containing up to 256 characters.
inReplyTo	Object	Contains the identification value if it is a reply to another post or comment. See the inReplyTo object definition below.
title	String	Contains the title of the comment/post. This data is shown on the Contact Detail view for the Agent. Note: Containing up to 256 characters.
content	String	Contains the text content of the post/comment sent by the source. Note: Containing up to 5000 characters.
contactData	Array of Objects	List of contactData key/value pairs which may carry customized data about the contact. Note: This list can be used by OpenScape Contact Center to define routing strategies according to the input key/values. This field is also shown at the Agent Portal. See the contactData object definition below.
attachments	Array of Objects	List of attachment objects that contain information about files attached to the message on the corporate system/social media. Note: The Agent Portal will show only if there's any attachment. See the attachments object definition below.
tagList	Array of Objects	List of tag objects that contain tag data depending on the social media. Tags are keywords used by social media to classify a post or mark some subject. See the tag object definition below.

Destination Object		
Attribute name	Attribute type	Description
id	String	Identifies the destination (To) of the contact. Note: This field can be used for routing at the Manager, through Design Center in Source/Destination component. Containing up to 256 characters.
type	String	Defines if the destination is a User or Page. The string values supported by the OpenScape Contact Center are "User", "Page".
name	String	Define the name of the destination. The name of the user or page. Note: The name is shown at the Agent Portal. Containing up to 256 characters.
URL	String	(Optional). Contain information about the URL of the destination where the message were sent.
additionalInfos	Array of String	(Optional) List of additionalInfos objects that contain a key/value pair that can be used by the connector.

RetrievedOpenMediaEvent JSON body example:

```
{
  "type": "OpenMediaEvent",
  "openMediaEventType": "RetrievedOpenMediaEvent",
  "dateTime": "2018-02-20 15:09:19.867",
  "contactID": "O553A8C5A0300",
```

```
"agentID":"100",
"source":{
  "id":"703545976517482_703546043184142",
  "type":"Organization",
  "name":"John Smith",
  "location":"Brasilia, Brasil",
  "language":"pt",
  "additionalInfo":[

  ]
},
"publishedObject":{
  "id":"703545976517482_703546026517477",
  "fatherId":"205056409590639_124328589355467",
  "inReplyTo":null,
  "title":"Sales Testing",
  "content":"",
  "contactData":null,
  "attachments":null,
  "tagList":null
},
"destination":{
  "id":"Node",
  "type":null,
  "name":"",
  "url":"",
  "additionalInfo":[

  ]
}
}
```

TransferredOpenMediaEvent

TransferredOpenMediaEventObject		
Attribute name	Attribute type	Description
type	String	Indicates the type of Event. To all Independent Media Events the type is OpenMediaEvent
openMediaEventType	String	Indicate type of Event. TransferredOpenMediaEvent
dateTime	String	Indicates the date and time of the creation of the published post/message. Note: Use UTC Date/Time format to create the dateTime string before sending to OpenScape Contact Center. This field is shown at the Agent Portal. JavaScript example: <code>Date().toUTCString();</code>
contactID	String	
agentID	String	
m_TransferReason	String	
m_TransferTargetType	String	
m_TransferringParty	Object type of TransferParty	
m_TransferTargetParty	Object type of TransferParty	

TransferParty Object		
Attribute name	Attribute type	Description
type	String	
called	String	
ISiteKey	String	
deviceld	String	
agentld	String	
IAgentKey	String	
ITeamKey	String	
teamName	String	
state	String	
m_isCtiEventReceived	Boolean	
m_isCtiEventExpected	Boolean	
m_connectionState	String	

TransferredOpenMediaEvent JSON body example:

```
{
  "type": "OpenMediaEvent",
  "openMediaEventType": "TransferredOpenMediaEvent",
  "dateTime": "2018-02-20 15:09:23.470",
  "contactID": "O553A8C5A0300",
  "agentID": "100",
```

```
"m_TransferReason":1,
"m_TransferTargetType":1,
"m_TransferringParty":{
  "type":0,
  "callId":"O553A8C5A0300O553A8C5A0300",
  "ISiteKey":354091819,
  "deviceId":"",
  "agentId":"100",
  "IAgentKey":7,
  "ITeamKey":0,
  "teamName":"",
  "state":2,
  "m_isCtiEventReceived":false,
  "m_isCtiEventExpected":false,
  "m_connectionState":7
},
"m_TransferTargetParty":{
  "type":0,
  "callId":"O553A8C5A0300O553A8C5A0300",
  "ISiteKey":354091819,
  "deviceId":"",
  "agentId":"100",
  "IAgentKey":7,
  "ITeamKey":0,
  "teamName":"",
  "state":2,
  "m_isCtiEventReceived":false,
  "m_isCtiEventExpected":false,
  "m_connectionState":7
}
}
```

DisconnectedEvent

DisconnectedEventObject		
Attribute name	Attribute type	Description
type	String	Indicates the type of Event. To all Independent Media Events the type is OpenMediaEvent
openMediaEventType	String	Indicate type of Event. DisconnectedEvent
dateTime	String	Indicates the date and time of the creation of the published post/message. Note: Use UTC Date/Time format to create the dateTime string before sending to OpenScape Contact Center. This field is shown at the Agent Portal. JavaScript example: Date().toUTCString();
contactID	String	The contactID property is an internal number used to uniquely identify contacts.
agentID	String	Use the AgentID property to store the ID that the user will use to log on.
discardReason	String	This DiscardReason property returns a Discard reason key as defined in the database.
disconnectReason	String	The enDisconnectReasons enumeration represents the reasons that a disconnect
disconnectedParty	Object type of DisconnectedParty	The DisconnectedParty property is the party that is disconnecting from the contact.

DisconnectedParty Object		
Attribute name	Attribute type	Description
agentKey	String	The AgentKey property is the key for the user in the database. This key represents the user who is associated with this call
device	String	The device property specifies the device of a user
partyType	String	The PartyType property describes the type of party object you have. This may also provide more information about the Device, depending on what the party object represents

DisconnectedEvent JSON body example:

```
{
  "type": "OpenMediaEvent",
  "openMediaEventType": "DisconnectedEvent",
  "dateTime": "2018-02-20 15:09:37.253",
  "contactID": "O553A8C5A0300",
  "agentID": "100",
  "discardReason": "11",
  "disconnectReason": "DISCARDED",
  "disconnectedParty": {
    "agentKey": "7",
```

```
"device": "",  
"partyType": "0"  
}  
}
```

6. Error Code

All the error codes and error text are set as follow:

Error Codes	
Number	Text
0	NO_ERROR
1	GENERAL_ERROR
2	ALREADY_REGISTERED_ON
4	WRONG_TITLE_OR_TOKEN_NOT_FOUND
6	AUTH_STATEMENT_NOT_VALID
7	NOT_READY
8	NOT_ENABLED
9	CONNECTOR_BLOCKED
10	CONNECTOR_NOT_REGISTERED
11	WRONG_API_REQUEST
12	STRING_OUT_OF_BOUND
13	WRONG_DATE_FORMAT

7. Appendix

JavaScript and NodeJS code example

In this session we have an example of code written in JavaScript containing functions and objects that can be used to integrate with the OpenScape Contact Center. If there is an intention to use any other programming language, the objects below can guide you as an example.

How to:

Step 1) To execute the code it is mandatory to install the Node JS (<https://nodejs.org/en/>).

Step 2) Create a .js file at any folder and copy the code below.

Step 3) After the file has been created, change some attributes values to enable the example to work. Attributes to be changed:

- myaddress: the ip address of the local machine
- myport: the port opened to receive events from OpenScape Contact Center
- omserveraddress: the ip address of the OpenScape Contact Center
- _omtoken: the token generated for the connector in Manager Application
- _omtitle: the title of the connector created in Manager Application

Step 4) Install the node module called **express** in the folder containing the .js file. Command: npm install express

Step 5) Install the node module called **body-parser** in the folder containing the .js file. Command: npm install body-parser

Step 6) Run the example using node in the folder containing the .js file. Command: node filename.js

Code example:

```
var express = require('express');
var bodyParser = require('body-parser');
var app = express();
var http = require('http');
var https = require('https');

var myaddress = '192.168.214.127';
var myport = '8080';

//Application Server specifics
var omserveraddress = '192.168.215.177';
var omserverport = '443';

// OpenMedia specifics
var _myconnhook = 'http://' + myaddress + ':8080/connectorhook';
var _omsession;
var _omtoken = "0xS9GJ7LAUaNSRGeyNqPUXmBkgPAjo";
var _omtitle = "ConnectorTitle";
var _omuserettings = null;
var _omIsRegistered = false;
var _ompooling = null;

initialize();
```

```

// REST Interface Configuration
app.use(express.static('public'));
app.use(bodyParser.json());

app.post('/connectorhook', function (req, res) {
  console.log(req.body);
  var data = req.body;

  processReceivedDataFromOSCC(data);

  response = {
    code: "200",
    data: "OK"
  };

  res.send(JSON.stringify(response));
});

// Starts the Server to receive the requests/response from OpenMedia web service
var server = app.listen(myport, function () {

  var host = server.address().address;
  var port = server.address().port;

});

function initialize(){
  registerConnectorToOSCC();
}

//Register connector to OpenScape Contact Center
function registerConnectorToOSCC(){

  var data = {
    "type": "Registration",
    "webhookURL": _myconnhook,
    "openMedia" : {

```

```

        "openMediaTitle" : _omtitle,
        "token" : _omtoken
    }
};

var options = {
    host: omserveraddress,
    port: omserverport,
    path: '/openmedia/webapi/main/registerConnector',
    method: 'POST',
    headers: {
        "Content-Type": "application/json",
    },
    payload: JSON.stringify(data)
};

sendRequestToOSCC(options);

}

//Creating and sending a NewContact
function createAndSendNewContactToOSCC(){

    var date = new Date(); // Get the current Date and Time
    var dateTime = date.toUTCString();
    var inReplyTo = new InReplyTo("111111111");

    // Add some Contact Data. These values can be used by OSCC.
    var contactDatas = [];
    var contactData1 = new ContactData("KEY1", "VALUE1");
    var contactData2 = new ContactData("KEY2", "VALUE2");
    contactDatas.push(contactData1);
    contactDatas.push(contactData2);

    // Add some additional info that can be useful only for the Connector.
    var addtSourceInfos = [];
    var addInfo1 = new AdditionalInfo("KEY1", "VALUE1");

```

```

    var addInfo2 = new AdditionalInfo("KEY2", "VALUE2");
    addtSourceInfos.push(addInfo1);
    addtSourceInfos.push(addInfo2);

    // Create the Source object that represents the from person that is posting a
message.
    var source = new Source("abigail.flores@emailtest.com", "PERSON", "Abgail
Flores", "New York, NY", "en", addtSourceInfos);

    // Create the Destination of the contact. This defines where the source is
posting a new message.
    var addtDestInfos = [];
    var destination = new Destination("3333444555", "PAGE", "My Page", "",
addtDestInfos);

    // Array of tags
    var tags = [];

    //Array of Attachments
    var attachment = [];

    // Main object that represents the publication for OSCC. This object contains
the
    var publishedObject = new PublishedObject("22222222", "111111111", inReplyTo,
"Test message", "This is a test message content. DateTime = " + dateTime,
contactDatas, attachment, tags);

    var type = "POST"; // Defines the type of the new contact.
    var newContact = new NewContact(type, dateTime, source, publishedObject,
destination, "true");
    sendNewContactDataToOSCC(JSON.stringify(newContact));
}

//Sending the information of the new contact data to the OpenScape Contact Center
function sendNewContactDataToOSCC(data) {

    var options = {
        host: omserveraddress,
        port: omserverport,
        path: '/openmedia/webapi/main/newContact',

```

```

method: 'POST',
headers: {
  "Content-Type": "application/json",
  "Authorization": _omsession
},
payload: data
};
console.log('New contact created.');
```

```

sendRequestToOSCC(options);
}

//Send the request to the OpenScape Contact Center
function sendRequestToOSCC(options) {
  var req;
  console.log('options.host: ' + options.host);
  console.log('options.port: ' + options.port);
  console.log('options.payload: ' + options.payload);

  process.env.NODE_TLS_REJECT_UNAUTHORIZED = "0";
  req = https.request(options, function (res) {
    res.setEncoding('utf8');

    res.on('data', function (chunk) {
      console.log('BODY: ' + chunk);
      processOSCCReturn(chunk);
    });
  });

  req.write(options.payload);

  req.on('response', function (e) {
    console.log('Response: ' + e.message);
    req.end();
  });

  req.on('error', function (e) {
    console.log('problem with request: ' + e.message);
  });
}

```

```

        req.end();
    });
}

//OpenScape Contact Center return
function processOSCCReturn(data) {
    var obj = JSON.parse(data);
    if (obj.errorCode === 0) {
        if(obj.type === "Registration") {
            _omsession = obj.sessionToken;
            _omuserSettings = obj.userCredentials;

            _omIsRegistered = true;
            openMediaServerPooling();
        }
    }
}

//Received Data from OpenScape Contact Center
function processReceivedDataFromOSCC(data) {
    var obj = data;

    console.log('processReceivedDataFromOSCC: ' + JSON.stringify(data));
    if (obj.type === "POST" ){
        console.log(obj.objectToBePublished.inReplyTo.id);
        console.log(obj.objectToBePublished.content);
    }
    else if (obj.type == "Stream") {
        var inReplyTo = new InReplyTo("4444444444");

        var sourcePerson = new SourceInfo("Person", "John Doe");
        var sourceOrganization = new SourceInfo("Organization", "Organization Name");

        var item1 = new OrderedItem("5555555", "1111_555555", inReplyTo, 'Post', '2016-09-10T13:15:00Z', sourceOrganization, 'Please provide more detail about your problem.');
```

```

        var item2 = new OrderedItem("6666666", "1111_555555", inReplyTo, 'Post', '2016-09-10T13:04:55Z', sourcePerson, 'This was the 1st comment to the complaint.');
```

```
    var item3 = new OrderedItem("7777777", "1111_55555", inReplyTo, 'Post', '2016-09-10T12:04:55Z', sourcePerson, 'This is the original complaint.');
```

```
    var items = [];  
    items.push(item1);  
    items.push(item2);  
    items.push(item3);
```

```
    var fakeStreamResponse = new StreamResponse("Stream", 0, 'NO_ERROR',  
"111111111", 1, 'true', "3", "3", items);
```

```
    sendStreamResponseToOSCC(fakeStreamResponse);
```

```
    }  
}
```

```
function sendStreamResponseToOSCC(fakeStreamResponse) {
```

```
    var options = {  
        host: omserveraddress,  
        port: omserverport,  
        path: '/openmedia/webapi/main/streamResponse',  
        method: 'POST',  
        headers: {  
            "Content-Type": "application/json",  
            "Authorization": _omsession  
        },  
        payload: JSON.stringify(fakeStreamResponse)  
    };
```

```
    sendRequestToOSCC(options);  
}
```

```
function openMediaServerPooling(){
```

```
    if (_omIsRegistered) {  
  
        var data = {};  
        var options = {
```

```

    host: omserveraddress,
    port: omserverport,
    path: '/openmedia/webapi/main/keepalive',
    method: 'POST',
    headers: {
      "Content-Type": "application/json",
      "Authorization": _omsession
    },
    payload: JSON.stringify(data)
  };

  process.env.NODE_TLS_REJECT_UNAUTHORIZED = "0";
  var req = https.request(options, function (res) {
    res.setEncoding('utf8');
    res.on('data', function (chunk) {
      console.log('KeepAlive response: ' + chunk);
      processOSCCReturn(chunk);
    });
  });
  req.end();
  req.on('error', function (e) {
    console.log('problem with request: ' + e.message);
  });
  createAndSendNewContactToOSCC();
  // Restart pooling
  pollMWSServer();
}

function pollMWSServer() {
  setTimeout(function() {
    openMediaServerPooling();
  }, 60000);
}

//Objects model
function AdditionalInfo(key, value) {

```

```

    this.key = key;
    this.value = value;
}

function Source(id, type, name, location, language, additionalInfos) {
    this.id = id;
    this.type = type;
    this.name = name;
    this.location = location;
    this.language = language;
    this.additionalInfos = additionalInfos;
}

function InReplyTo(id) {
    this.id = id;
}

function ContactData(key, value) {
    this.key = key;
    this.value = value;
}

function Attachment(type, content, url) {
    this.type = type;
    this.content = content;
    this.url = url;
}

function Tag(tag) {
    this.tag = tag;
}

function PublishedObject(id, fatherId, inReplyTo, title, content, contactData,
attachments, tagList) {
    this.id = id;
    this.fatherId = fatherId;
    this.inReplyTo = inReplyTo;
}

```

```

    this.title = title;
    this.content = content;
    this.contactData = contactData;
    this.attachments = attachments;
    this.tagList = tagList;
}

function Destination(id, type, name, URL, additionalInfos) {
    this.id = id;
    this.type = type;
    this.name = name;
    this.URL = URL;
    this.additionalInfos = additionalInfos;
}

function NewContact(type, dateTime, source, publishedObject, destination,
isRealTimeHandling) {
    this.type = type;
    this.dateTime = dateTime;
    this.source = source;
    this.publishedObject = publishedObject;
    this.destination = destination;
    this.isRealTimeHandling = isRealTimeHandling;
}

function OrderedItem(id, fatherId, inReplyTo, type, dateTime, source, content) {
    this.id = id;
    this.fatherId = fatherId;
    this.inReplyTo = inReplyTo;
    this.type = type;
    this.dateTime = dateTime;
    this.source = source;
    this.content = content;
}

function StreamResponse(type, errorCode, errorText, id, page, lastPage, itemsinpage,
totalItems, orderedItems) {

```

```
    this.type = type;
    this.errorCode = errorCode;
    this.errorText = errorText;
    this.id = id;
    this.page = page;
    this.lastPage = lastPage;
    this.itemsinpage = itemsinpage;
    this.totalItems = totalItems;
    this.orderedItems = orderedItems;
}
```

```
function SourceInfo(type, name) {
    this.type = type;
    this.name = name;
}
```

