API DOCUMENT

Dotgo RBM APIs

Table of Contents

1. Introduction

1.1. Overview

RCS APIs platform provides GSMA Chatbot MaaP APIs as per GSMA MaaP Chatbot API specifications and Google specific APIs as per Google API specifications, to extend any bot's services to multiple MaaP platforms with ease. Register a bot on the RCS-APIs Platform and send messages to a phone on any supported network via the RCS-APIs Platform.

If you are already familiar with Google RBM APIs, then you can skip Section 2 (API).

In addition to RCS APIs, Dotgo API platform provides APIs for brands to send promotional and transactional RBM messages to a target base belonging multiple MaaP platforms. These will be sent in pre-defined templates that you (as a brand) need to configure after you sign up in to the Dotgo developer portal.
Dotgo API platform also provides Google Business Messages (GBM) API for brands to develop a conversational experience with users who discover them from Google Maps or organic search results or some other channel. For more information on GBM, please go through the below link: https://developers.google.com/business-communications/business-messages/guides/learn

All Dotgo APIs are REST-style HTTP APIs and, receive and return JSON data. A JSON request is sent to an HTTP API endpoint.

An API call includes the below attributes

- **Method**: One of the HTTP Methods like POST, GET, DELETE, and PUT.
- **URL**: The API endpoint starting with the server_root
- **HTTP Headers**: Content-Type and authorization headers. Content-Type is application/json and sees Authorization header having access token.
- **Request Body:**JSON request data

In response to an HTTP API call, the response returned will have an HTTP Status Code such as 200, 202, 404, 500, etc. and a JSON Response.

The Chatbot needs to register it's webhook with the RCS APIs platform. The webhook should be secured with HTTPS. User messages, responses, and status notifications will be posted on the webhook.

Dotgo APIs ServerRoot:

ServerRoot: https://api.dotgo.com/rcs

1.2. Sign up for free

Before you get started, sign up for a Dotgo Developer account for free. Once you have submitted your RCS agent details with Dotgo, you will be shared a client Id and client secret and a botID to invoke the RCS APIs via an access token.

The details are shared in the subsequent sections.

1.3. RBM Agent Registration

Step 1

To use the Dotgo RCS APIs, the brand needs to submit the agent details (the form with these details will be shared manually to you) to Dotgo. Along with the agent details, this also includes the webhook endpoint of the brand for receiving message events and notifications from the Dotgo RCS API platform.

Step 2

Once the agent is created and verified at the backend by Dotgo, the agent needs to be registered with the Dotgo RCS APIs.

This will generate the necessary client ID and secret in the Dotgo SSO authentication server.

Step 3

The client Id and secret and the botID will be shared with the brand over email.

The webhook URL to call back the brand will also be registered along with other details.

1.4. Authentication and Authorization

The APIs are secured by the Dotgo Auth 2 SSO service. To access GSMA APIs of the RCS APIs platform, an access token with Chatbot-message scope needs to be provided.

Access token can be obtained from these APIs :

https://auth.dotgo.com/auth/oauth/token?grant_type=client_credentials

To get the token, the clientId and client_secret should be sent in the Authorization header as Basic authentication (base64 encoded).

The above endpoint is rate limited in terms of the number of requests allowed per minute.
By default the number of transactions per minute (TPM) will be 60 per client (A client will typically be mapped to a single bot created)
In case you wish to have a higher value, please get in touch with rbm-support@dotgo.com

2. GSMA Style RBM APIS

2.1. Communicate with user

Resource endpoint: {serverRoot}/bot/v1/{botId}/messages/async

This is the API used to send messages and isTyping indications to users.

Request format:

URL: {serverRoot}/bot/v1/OsOsQ0GwNvUdLTV9Bd/messages/async

Authorization: Access Token obtained from Auth2 SSO service as Bearer token

Request:

```
{
   "RCSMessage":{
      "textMessage":"hello world"
   },
   "messageContact":{
      "userContact":"+914253136789"
   }
}
```

Request Parameters:

Name	Type	Description	Remarks
botId	Path variable	botId registered with the RCS APIs platform	
userContact	A field in the Request Body	User MSISDN in canonical form	Ex: +914253136789

Response Format:

```
{
    "RCSMessage": {
        "msgId": "6cd095cd-62f6-4338-bba2-4b14b98b0537",
        "status": "pending"
    }
}
```

Response Parameters:

Name	Type	Description	Remarks
msgId	A field in the Response body	Message Identifier for the message sent	
status	A field in the response body	Status of the message	The values are as defined in the Message Status response 2.11 of GSMA MaaP Chatbot API specifications

HTTP Response Codes

Code	Description
202	The request of sending message or isTyping indication is accepted by the Platform and ready to send to the user
400	This is a bad request with invalid input, invalid object, etc
401	This request is unauthorized.
403	Invalid clientId or Invalid IP address
5XX	Server error.

2.1.1. Send a message to the user

Various types of messages that can be sent to users, including text message, file, audio message, geolocation push, rich card, and suggested chip list are as per GSMA MaaP Chatbot API specifications.

In addition to the RCS messages, any template created on the Developer portal can be sent on this endpoint. Please refer to RCS Template section for more details.

1. POST {serverRoot}/bot/v1/OsOsQ0GwNvUdLTV9Bd/messages/async :- This is the GSMA

styled messaging API. Request submitted on this endpoint will receive only Pending event and validation errors (invalid client, invalid template, rate limit etc...) as API response, any other errors (charging failure, number not found, etc...) will be sent as a failure event on the agent's callback URL.

Authorization: Access Token obtained from Auth2SSO service as Bearer token

Request Parameters:

Name	Type	Description	Remarks
botId	Path variable	botId registered with the RCS APIs platform	
enableFallback	Query parameter	This is an optional parameter. GIP fallback will be disabled by default and will be enabled only if this flag is set to true.	Values: true/false
sendGipLink	Query parameter	This is an optional parameter and applicable only If bot has chosen GIP as a fallback. When this is set to true, an SMS will be sent to user along with GIP link.	Values: true/false

2.1.1.1. Example

Request:

Ex1: Template Message

```
{
 "RCSMessage": {
   "templateMessage": {
     "templateCode":"template_123",
     "customParams":"{\"name\":\"user\"}"
   }
 },
 "messageContact": {
  "userContact": "+14251234567"
 }
}
```

Ex1. Text Message

```
{
  "RCSMessage":{
    "textMessage":"hello world"
  },
  "messageContact":{
    "userContact":"+914253136789"
  }
}
```

Ex2. File Message

```
{
  "RCSMessage": {
 "fileMessage": {
    "fileUrl" : "https://konnect.kirusa.com/uploads/rcsTemplates/InstaVoice.png"
    }
 },
 "messageContact": {
    "userContact": "+14251234567"
 }
}
```

Ex3. File Message with suggestions

```
{
   "RCSMessage": {
      "fileMessage": {
         "fileUrl" : "https://storage.googleapis.com/kitchen-sink-sample-images/elephant.jpg"
      },
      "suggestedChipList": {
         "suggestions": [
         {
            "reply": {
               "displayText": "suggestion#1",
               "postback": {
               "data": "set_by_chatbot_reply_1"
               }
            }
         },
         {
            "action": {
               "displayText": "Call",
               "postback": {
                  "data" : "postback_data_1234"
               },
               "dialerAction": {
                  "dialPhoneNumber":{
                  "phoneNumber": "+15556667777"
                  }
               }
            }
         },
         {
            "action": {
               "urlAction":{
                  "openUrl":{
                     "url":"https://www.google.com"
                  }
               },
               "displayText":"Open website or deep link",
               "postback":{
                  "data":"set_by_chatbot_open_url"
               }
            }
         },
         {
            "action":{
               "calendarAction":{
                  "createCalendarEvent":{
                     "startTime":"2017-03-14T00:00:00Z",
                     "endTime":"2017-03-14T23:59:59Z",
                     "title":"Meeting",
                     "description":"GSG review meeting"
                  }
               },
               "displayText":"Schedule Meeting",
               "postback":{
                  "data":"set_by_chatbot_create_calendar_event"
               }
            }
         },
         {
            "action":{
               "mapAction":{
                  "showLocation":{
                     "location":{
                     "latitude":37.4220041,
                     "longitude":-122.0862515,
                     "label":"Googleplex"
                     },
                     "fallbackUrl":"https://www.google.com/maps/@37.4219162,-122.078063,15z"
                  }
               },
               "displayText":"Show location on a map",
               "postback":{
                  "data":"set_by_chatbot_open_map"
               }
            }
         },
         {
            "action":{
               "mapAction":{
                  "requestLocationPush":{}
               },
               "displayText":"Share location on a map",
               "postback":{
                  "data":"set_by_chatbot_open_map"
                }
            }
         }
         ]
      }
   },
   "messageContact": {
      "userContact": "+14251234567"
   }
}
```

The following table provides the file types allowed:
The max size mentioned below for images and videos applies to sending direct file messages (and not for rich cards/carousels).
For rich cards and carousels, please click below for file size recommendations and limits:
https://developers.google.com/business-communications/rcs-business-messaging/guides/learn/best-practices

Media Type	Document Type	Extension	Max Size	Works with rich cards
application/pdf	PDF	.pdf	100 MB	No
image/jpeg	JPEG	.jpeg, .jpg	100 MB	Yes
image/gif	GIF	.gif	100 MB	Yes
image/png	PNG	.png	100 MB	Yes
video/h263	H263 video	.h263	100 MB	Yes
video/m4v	M4V video	.m4v	100 MB	Yes
video/mp4	MP4 video	.mp4	100 MB	Yes
video/mpeg4	MPEG-4 video	.mp4, .m4p	100 MB	Yes
video/mpeg	MPEG video	.mpeg	100 MB	Yes

video/webm	WEBM video	.webm	100 MB	Yes

Ex4. Rich Card with Suggested chiplist and actions

```
{
"RCSMessage": {
  "trafficType": "advertisement",
  "richcardMessage": {
    "message": {
      "generalPurposeCard": {
        "layout": {
          "cardOrientation": "HORIZONTAL",
          "imageAlignment": "LEFT"
        },
        "content": {
          "media": {
            "mediaUrl": "https://cdn.server/path/media.mp4",
            "mediaContentType": "video/mp4",
            "mediaFileSize": 2718288,
            "thumbnailUrl": "https://cdn.server/path/media.png",
            "thumbnailContentType": "image/png",
            "thumbnailFileSize": 314159,
            "height": "MEDIUM_HEIGHT",
            "contentDescription": "Textual description of media content, e. g. for use with screen readers."
          },
          "title": "This is a single rich card.",
          "description": "This is the description of the rich card. It's the first field that will be truncated if it exceeds the maximum width or height of a card."
        }
      }
    }
  },
  "suggestedChipList": {
    "suggestions": [
      {
        "reply": {
          "displayText": "Yes",
          "postback": {
            "data": "set_by_chatbot_reply_yes"
          }
        }
      },
      {
        "reply": {
          "displayText": "No",
          "postback": {
            "data": "set_by_chatbot_reply_no"
          }
        }
      },
      {
        "action": {
          "urlAction": {
            "openUrl": {
              "url": "https://www.gsma.com"
            }
          },
          "displayText": "Open website or deep link",
          "postback": {
            "data": "set_by_chatbot_open_url"
          }
        }
      }
    ]
  }
},
"messageContact": {
  "userContact": "+14251234567"
}
}
```

Response: 202 Accepted

```
{
    "RCSMessage": {
        "msgId": "63c30986-3222-4369-a8e8-16c09b36f34f",
        "status": "pending"
    }
}
```

Response:403 Forbidden

```
{
    "RCSMessage": {
        "msgId": "937796c9-46d3-432f-b958-5d9215874f8f",
        "status": "failed",
        "timestamp": "2023-02-01T04:43:48.361Z"
    },
    "reason": {
        "text": "Invalid client id!",
        "code": 403
    }
}
```

Error Code Details:

error_type	**error_code** (Within response, not http status code)	**error_message**
insufficient_balance	402	insufficient balance. Please recharge to send message
invalid_template	409	user is not in a conversation and provided message template is not approved
opted_out	410	opted_out
dnd_enabled	423	dnd_enabled
curfew_hrs	403	curfew_hrs
message_limit	429	monthly message limit exceeded
rcs_disabled	404	Number is rcs disabled
rate_limit	429	429 TOO_MANY_REQUESTS - You have reached the maximum number of times that this request can be made in a given time period
charge_failure	502	Unable to charge for this message.
internal_server_error	500	specific internal server error
4xx_maap_errors	400	maap specific error
invalid_client	403	Invalid client id
Invalid_template_data	400	‘Template with the code [temp01] and bot [TestAgent] received with insufficient custom params’ or Template with the code [temp01] and bot [TestAgent] is not approved, hence cannot be used for sending message.
google_rate_limit	400	429 TOO_MANY_REQUESTS - You have reached the allowed traffic limit for this agent on the user, your message will not be delivered.

2.1.2. Send isTyping indication to the user

The Chatbot can send isTyping indication to users as defined in 2.4 of GSMA MaaP Chatbot API specifications.

2.1.2.1. Example:

POST {serverRoot}/bot/v1/OsOsQ0GwNvUdLTV9Bd/messages/async

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request Parameters:

Name	Type	Description	Remarks
botId	Path variable	botId registered with the RCS APIs platform	

Request:

```
{
   "RCSMessage":{
      "isTyping":"active"
   },
   "messageContact":{
      "userContact":"+914253136789"
   }
}
```

Response:202 Accepted

```
{
    "RCSMessage": {
        "msgId": "e125527c-1af3-4158-8db9-9cc4f16d4733",
        "status": "pending"
    }
}
```

2.2. Message Status

Resource endpoint: {serverRoot}/bot/v1/{botId}/messages/{msgId}/status

Request format:

URL: {serverRoot}/bot/v1/OsOsQ0GwNvUdLTV9Bd/messages/2bbedeb3-b071-473c-878b-dac4252d149b/status

Authorization: Access Token obtained from Auth2 SSO service as Bearer token

Request Parameters:

Name	Type	Description	Remarks
botId	Path variable	botId registered with the RCS APIs platform	

2.2.3. Query status of a message

Although the message status is received on the webhook, this API provides an alternative optional way to check the message status. Possible message status includes 'pending', 'sent', 'delivered', 'displayed', 'cancelled', 'revoked', and 'failed'.

2.2.3.1. Example:

GET{serverRoot}/bot/v1/OsOsQ0GwNvUdLTV9Bd/messages/ddc24c2-cff5-48ac-baaa-4f286bc28061/status

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

No request body

Request Parameters:

Name	Type	Description	Remarks
msgId	Path variable	Reference message-id (received from the RCS APIs platform)	

Response: 200 OK

```
{
    "RCSMessage": {
        "msgId": "ddc24c2-cff5-48ac-baaa-4f286bc28061",
        "status": "sent",
        "timestamp": "2020-11-10T11:06:26"
    }
}
```

Response Parameters:

Name	Type	Description	Remarks
msgId	A field in the Response body	Message Identifier for the message	
status	A field in the response body	Status of the message	The values are as defined in the Message Status response 2.11 of GSMA MaaP Chatbot API specifications

HTTP Response Codes

Code	Description
200	The status of the message will be returned in the response body.
404	The message ID cannot be found
401	This request is unauthorized.
5XX	Server error.

2.2.4. Send 'read' notification to users

A chatbot can send a 'read' notification to users for any received user message.

2.2.4.1. Example

PUT{serverRoot}/bot/v1/OsOsQ0GwNvUdLTV9Bd/messages/MsYMv92L4HS5GJ54rW5Xm2JA/status

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

```
{
  "RCSMessage": {
    "status": "displayed"
  }
}
```

Request Parameters:

Name	Type	Description	Remarks

msgId	Path variable	Message-Id of the received user message	

Response:204 No Content

No response body

HTTP Response Codes

Code	Description
204	The status of the message has been updated and a read notification will be sent to the user
400	This is a bad request with invalid input, invalid object, etc.
404	The message ID cannot be found
401	This request is unauthorized.
5XX	Server error.

2.2.5. Revoke a sent message

Sent message can be revoked only if it has not been delivered to the user. Hence the response from the platform is not the confirmation of revocation. It only indicates that the RCS APIs platform shall try to revoke the message.

2.2.5.1. RIchExample:

PUT{serverRoot}/bot/v1/OsOsQ0GwNvUdLTV9Bd/messages/MsW6S-iK49TuCRSwG=mfNP7Q/status

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

```
{
  "RCSMessage": {
    "status": "cancelled"
  }
}
```

Request Parameters:

Name	Type	Description	Remarks

msgId	Path variable	Reference message-id (received from the RCS APIs platform)	

Response:204 No Content

No response body

HTTP Response Codes

Code	Description
204	The RCS APIs Platform shall try to revoke the message if it has not been delivered to the user.
400	This is a bad request with invalid input, invalid object, etc.
404	The message ID cannot be found
401	This request is unauthorized.
5XX	Server error.

2.3. Check RCS capability of the contact/user

Resource endpoint: {serverRoot}/bot/v1/{botId}/contactCapabilities

This is an RCS based communication service, the Chatbot shall only communicate with the user using an RCS capable device. So the Chatbot shall conduct the RCS capability discovery to learn about whether the given user's device is RCS capable or not.

Possible capabilities include 'chat', 'fileTransfer', 'videoCall', 'geolocationPush', 'callComposer' and 'chatBotCommunication'.

Request format:

URL: {serverRoot}/bot/v1/OsQ0GwNvUdLTV9Bd/contactCapabilities

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

No request body

Request Parameters:

Name	Type	Description	Remarks

botId	Path variable	botId registered with the RCS APIs platform	
userContact	Query parameter	User phone number in canonical form	Ex: +914253136789 (Please note the country code prefix along with a '+')

Response Format:

```
{
    "capabilities": [
        "chatBotCommunication",
        "chat",
        "fileTransfer"
    ]
}
```

Response Parameters:

Name	Type	Description	Remarks
capabilities	A field in the Response body	capabilities supported by contact/user	These capabilities need to be interpreted as specified in 3.3 of GSMA MaaP Chatbot API specifications.

HTTP Response Codes

Code	Description
200	Supported capabilities will be returned if the user's device is RCS capable.
401	This request is unauthorized.
404	User's device is not RCS capable.
403	Invalid client Id
5XX	Server error.

2.3.6. Example:

GET {serverRoot}/bot/v1/OsQ0GwNvUdLTV9Bd/contactCapabilities? userContact=+914253136788

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

No request body

Response:404 NOT FOUND

```
{
  "status": "Error",
  "message": "Not found : Number is rcs disabled and URCS is not enabled for the bot"
}
```

2.4. Files

Resource endpoints: {serverRoot}/bot/v1/{botId}/files
{serverRoot}/bot/v1/{botId}/files/{fileId}

This is not for file transfer, but uploading files from Chatbot to the RCS APIs platform, to check the status of the uploaded file and to delete the uploaded file.

Request Parameters:

Name	Type	Description	Remarks
botId	Path variable	botId registered with the RCS APIs platform	

2.4.7. Upload a file to the platform

Resource endpoints: {serverRoot}/bot/v1/{botId}/files

There are two ways to upload files to the RCS APIs platform. Files can be uploaded directly to the RCS APIs platform or by sharing a file URL.

Request format:

URL: {serverRoot}/bot/v1/OsQ0GwNvUdLTV9Bd/files

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:contentType - multipart/formdata form data fields : -

‘fileType‘

‘until ‘

‘fileContent’ or ‘fileUrl’

Request Parameters:

Name	Type	Description	Remarks
fileType	A field in the form	Format of the file	Ex: audio/mp4 or image/png etc..
until	A field in the form	The expiry for the file	
fileContent	A field in the form	The file that needs to be uploaded	
fileUrl	A field in the form	Valid file URL from where the file needs to be read and uploaded	

Response Format :

```
{
    "File": {
        "fileId": "GiYosOsQ0GwNvUdLTV9Bd2naXGkuz8bmDA",
        "fileUrl": "https:/stagingrcsapi.kirusa.com/OsQ0GwNvUdLTV9Bd/someFile.png",
        "fileSize": 15022,
        "status": "ready",
        "validity": "2020-11-12T11:06:26"
    }
}
```

Response Parameters:

Name	Type	Description	Remarks
fileId	A field in the Response body	The identifier for the uploaded file	
fileUrl	A field in the response body	The file URL, that can be used later as media content in the rich card or other message types such as file transfer or audio message	

status	A field in the response body	Status, if the file is ready for use	

HTTP Response Codes

Code	Description
202	The file is uploaded to the platform.
400	This is a bad request with invalid input, invalid object, etc
401	This request is unauthorized.
5XX	Server error.

2.4.7.1. Upload a raw file

2.4.7.1.1. Example:

POST {serverRoot}/bot/v1/OsQ0GwNvUdLTV9Bd/files

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:contentType - multipart/formdata

fileType	Image/png
until	2020-11-12T11:06:26
fileContent	picture1.png

Response: 202 Accepted

```
{
    "File": {
        "fileId": "YB5tV2prGiY2eAyIuFZ6AiEJ1zPnaXGk",
        "fileUrl": "https:/stagingrcsapi.kirusa.com/OsQ0GwNvUdLTV9Bd/picture1.png",
        "fileSize": 15022,
        "status": "ready",
        "validity": "2020-11-12T11:06:26"
    }
}
```

2.4.7.2. Upload a file by URL

2.4.7.2.1. Example:

POST {serverRoot}/bot/v1/OsQ0GwNvUdLTV9Bd/files

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:contentType - multipart/formdata

fileType	Image/png
until	2020-11-12T11:06:26
fileUrl	https://somedomain.com/pictures/picture2.png

Response: 202 Accepted

```
{
    "File": {
        "fileId": "wPT9VpDqEfW98EnYDpSiuMC74yn7N7tX",
        "fileUrl":"https:/stagingrcsapi.kirusa.com/OsQ0GwNvUdLTV9Bd/picture2.png",
        "fileSize": 14052,
        "status": "ready",
        "validity": "2020-11-12T11:06:26"
    }
}
```

2.4.8. Delete the uploaded file

Resource endpoints: {serverRoot}/bot/v1/{botId}/files/{fileId}

A chatbot can delete the file uploaded to the platform earlier, by providing the file Id.

Request format:

URL:{serverRoot}//bot/v1/OsQ0GwNvUdLTV9Bd/files/GiYosQ0GwNvUdLTV9Bd2naXGkuz8bmDA

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

No request body

Request Parameters:

Name	Type	Description	Remarks
fileId	Path variable	Reference file Id (received from the RCS APIs platform)	

Response Format: 204 No Content

No response body

HTTP Response Codes

Code	Description
204	The file has been deleted.
404	The file cannot be found
401	This request is unauthorized.
5XX	Server error.

2.4.8.1. Example:

DELETE
{serverRoot}/bot/v1/OsQ0GwNvUdLTV9Bd/files/DqVpWEf8E7N7tnpXwPSiuMT99YDC74yn

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:
No request body

Response:204 No Content

2.4.9. Get status of the uploaded file

Resource endpoint: {serverRoot}/bot/v1/{botId}/files/{fileId}
A chatbot can retrieve the uploaded file's information. This API provides a way to check the file status along with other information. Possible file status includes 'ready' and 'expired'.

Request format:

URL: {serverRoot}/bot/v1/OsQ0GwNvUdLTV9Bd/files/YB5tV2prGiY2eAyIuFZ6AiEJ1zPnaXGk

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:
No request body

Request Parameters:

Name	Type	Description	Remarks
fileId	Path variable	Reference file Id (received from the RCS APIs platform)	

Response Format:

```
{
    "File": {
        "fileId": "YB5tV2prGiY2eAyIuFZ6AiEJ1zPnaXGk",
        "fileUrl": "https:/stagingrcsapi.kirusa.com/OsQ0GwNvUdLTV9Bd/picture1.png",
        "fileSize": 15022,
        "status": "ready",
        "validity": "2020-11-12T11:06:26"
    }
}
```

Response Parameters:

Name	Type	Description	Remarks
fileId	A field in the Response body	The identifier for the uploaded file	
fileUrl	A field in the response body	The file URL, that can be used later as media content in the rich card or other message types such as file transfer or audio message	
status	A field in the response body	Status, if the file is ready for use	Values: ready, expired

HTTP Response Codes

Code	Description
200	File information will be returned
404	The file cannot be found
401	This request is unauthorized.
5XX	Server error.

2.4.9.1. Example:

GET{serverRoot}/bot/v1/OsQ0GwNvUdLTV9Bd/files/YB5tV2prGiY2eAyIuFZ6AiEJ1zPnaXGk

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

No request body

Response:200 OK

```
{
    "File": {
        "fileId": "YB5tV2prGiY2eAyIuFZ6AiEJ1zPnaXGk",
        "fileUrl": "https:/stagingrcsapi.kirusa.com/OsQ0GwNvUdLTV9Bd/picture1.png",
        "fileSize": 15022,
        "status": "expired",
        "validity": "2020-11-12T11:06:26"
    }
}
```

2.5. Webhook

Webhook is the callback API provided by the Chatbot. RCS APIs Platform uses the webhook exposed by Chatbot to send an HTTP POST payload when certain RCS events occur.

A chatbot may receive the following events from the RCS-APIs platform:

1. message from a user (text, file, location, or audio message)
2. isTyping notification from a user
3. message status update
4. response to suggested reply or action

The Chatbot shall always return a 200 OK HTTP response to the HTTP POST from the RCS APIs Platform.

2.5.10. Webhook payload from RCS-APIs platform :

The payload that can be sent from the RCS APIs platform to Chatbot's webhook is as per the WebhookPayload model specified in 2.15 of GSMA MaaP Chatbot API specifications.

2.5.10.1. Example

2.5.10.1.1. Message from a user

```
{
   "RCSMessage": {
      "msgId":"MsWHg8WU3cTjK7scyQ1bv-aA",
      "textMessage":"hello world",
      "timestamp":"2020-11-12T09:16:29.527061Z"
   },
   "messageContact": {
      "userContact":"+914253136789"
   },
   "event":"message"
}
```

2.5.10.1.2. isTyping message from a user

```
{
   "RCSMessage":{
      "isTyping":"active",
      "timestamp":"2020-11-12T09:34:44.876908Z"
   },
   "messageContact":{
      "userContact":"+914253136789"
   },
   "event":"isTyping"
}
```

2.5.10.1.3. Message status update

```
{
   "RCSMessage":{
      "msgId":"334a5db9-b19d-46c4-80e1-ff662ba2a982",
      "status":"delivered",
      "timestamp":"2020-11-12T09:55:49.444158Z"
   },
   "messageContact":{
      "userContact":"+914253136789"
   },
   "event":"messageStatus"
}
```

2.5.10.1.4. Response to suggested reply/action

```
{
   "RCSMessage":{
      "msgId":"Ms4xAa25HvQfGpX8dWXdNXDw",
      "timestamp":"2020-11-12T09:34:38.553753Z",
      "suggestedResponse":{
         "response":{
            "reply":{
               "displayText":"Visit a website",
               "postback":{
                  "data":"set_by_chatbot_reply_yes"
               }
            }
         }
      }
   },
   "messageContact":{
      "userContact":"+914253136789"
   },
   "event":"response"

}
```

2.5.10.1.5. Response to suggested reply/action inside templates

(Template description has been provided in Templates and Rich Notification APIs)

```
{
   "RCSMessage":{
      "msgId":"MxNsfgtg86T-6-=o2i7P3pGw",
      "timestamp":"2023-06-20T15:39:30.475341Z",
      "suggestedResponse":{
         "response":{
            "action":{
               "displayText":"Reach Us",
               "postback":{
                  "data":"user_clicked_Reach_Us",

"metadata":"{\"suggestionType\":\"url_action\",\"templateType\":\"carousel\",\"cardIndex
\":1,\"suggestionIndex\":2,\"msgId\":\"1ed15c61-7fd1-436a-a127-
0baa8d91cdb2\",\"a2pMsgDate\":\"2023-07-03T15:22:00.356\",\"templateCode\":\"test
template\"}"
               }
            }
         }
      }
   },
   "messageContact":{
      "userContact":"+919976878767"
   },
   "event":"response"
}
```

The metadata fields will be additionally sent as part of the payload to the agent webhook in case the user responses are within A2P message templates. This will include the following:

SuggestionType: reply, dialer_action, url_action

templateType: text_message, rich_card, carousel

cardIndex: in case of a carousel, the index of the card for which the user event is generated (index starts from 0)

suggestionIndex: index of the suggestion starting from 0

msgId: the message Id of the original A2P message

a2pMsgDate: date in which the original A2P message was sent

templateCode: name of the template

2.6. Bulk Capability Check

This is the API which returns RBM enabled users from the given list of users.

To get the number of RBM-reachable users, do a bulk capability check. Bulk checks include whether a phone number is reachable but not which capabilities or features a phone number supports (Please refer to the single capability check API in Section 2.3 to understand what is meant by capabilities). You can specify up to 10,000 phone numbers per bulk capability check. To check more numbers, perform multiple checks.

Bulk capability checks respond with the specified numbers that are reachable by the agent.

Note: This is a common API for both Google and GSMA and This API is applicable for only **launched**

2.6.0. Example

POST {serverRoot}/bot/v1/OsOsQ0GwNvUdLTV9Bd/rcsEnabledContacts

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

```
{
 "users": [
   "+919687895543",
   "+919686960876",
   "+919688757768"
 ]
}
```

Request Parameters:

Name	Type	Description	Remarks
botId	Path variable	botId registered with the RCS APIs platform	

Response:

Ex1. 200 OK

```
{
 "rcsEnabledContacts": [
   "+919686960876"
 ]
}
```

Ex2. 400 Bad Request

```
{
 "Error": "Requested users shouldn't be more than 10000 in a request"
}
```

HTTP Response Codes

Code	Description
200	RCS enabled users from the provided list of users
400	This is a bad request with invalid input, invalid object, etc.
401	This request is unauthorized.
5XX	Server error.

3. Google Style RBM APIs

Dotgo MaaP also provides Google specific APIs as per Google RBM standard specifications for the ease of use for developers and brands.
This comes handy for a developer or brand who has already integrated with Google RBM APIs earlier, and now wants to integrate with a non-Google MaaP. With these APIs, they can seamlessly integrate with Dotgo's Google Specific API's (even for non-Google MaaPs) with minor changes since the API request and response payload will be same as per Google standards.

3.1. Authorization

To access Google APIs of the Dotgo RCS platform, an access token with 'Chatbot-message' and 'google' scope needs to be provided.
Register a ServiceClient with scope 'Chatbot-message' and 'google' to receive OAuth2client_credentials. Using the credentials obtained from SSO, an access token can be generated which in turn should be used as an Authorization Bearer token, to access Google spec APIs of RCS platform.

Access token can be obtained from these APIs :

https://auth.dotgo.com/auth/oauth/token?grant_type=client_credentials

To get the token, the clientId and client_secret should be sent in the Authorization header as Basic authentication (base64 encoded).

3.2. Send a Message to the user

This is the API used to send messages to users. Various types of messages can be sent to users, including text message, file, rich card, and carousel as per Google API specifications. Suggestions can be sent along with any of the mentioned message types.

In addition to the RCS messages, any template created on the Developer portal can be sent on this endpoint. Please refer to Rich Template section for more details.

1. POST
{serverRoot}/v1/phones/{phone_number}/agentMessages/async?botId={bot_id}&messageId={message_id}

This is the Google styled messaging API. Request submitted on this endpoint will receive only Pending event and validation errors (invalid client, invalid template, rate limit etc...) as API response, any other errors (charging failure, number not found, etc...) will be sent as a failure event on the agent's callback URL.

Request Parameters:

Name	Type	Description	Remarks
phone_number	Path variable	User MSISDN in canonical form	Ex: +914253136789
botId	Query parameter	botId registered with the RCS APIs platform	
messageId	Query parameter	Unique id for the message (not mandatory)	Optional. In case of duplicate messageId, request will be rejected. If messageId is not provided, Platform will assign one and include it in response.
enableFallback	Query parameter	This is an optional parameter. GIP fallback will be disabled by default and will be enabled only if this flag is set to true.	Boolean : true/false
sendGipLink	Query parameter	This is an optional parameter and applicable only If bot has choosen GIP as a fallback. When this is set to true, an SMS will be sent to user along with GIP link.	Boolean : true/false

<u>Message expiry:</u>

An Additional field available in all agent messages to support message expiry feature. To help ensure timely and relevant messages, set a message expiration. This can prevent offline users from receiving stale content when they come back online. Expiration is also a good cue to invoke your fallback messaging strategy, so users get the info they need on time.

To set a message expiration, specify one of the following fields in the agent message Json :

expireTime: the exact time in UTC when the message expires. A timestamp in RFC3339 UTC "Zulu" format, with nanosecond resolution and up to nine fractional digits.
example: - "expireTime": "2024-10-02T15:01:23Z"

ttl(time to live): the amount of time before the message expires. A duration in seconds
with up to nine fractional digits, ending with 's'.
example:- "ttl": "3.5s"

These fields are optional and if any older dates or invalid formats are used such messages will be rejected. Once the message expires, the platform stops trying to deliver the message, and it's automatically revoked. Based on revoke status, 'ttl_expiration_revoke' or ttl_expiration_revoke_failed event will be sent on agent webhook. Please refer to section 3.8.1.1.6 and 3.8.1.1.7 for details.

Request format:

URL:
POST https://api.dotgo.com/rcs/v1/phones/+914253136789/agentMessages/async?botId= OsOsQ0GwNvUdLTV9Bd&messageId=d2f81052-37c7-469c-997b-9576023b4220

Authorization: Access Token obtained from Auth2 SSO service as Bearer token

Request:

Ex 1: TEMPLATE MESSAGE

```
{
  "contentMessage":{
    "templateMessage":{
       "templateCode":"template_123",
       "customParams":"{\"name\":\"user\"}"
    }
  },
  "ttl": "10s"
}
```

Ex2: TEXT

```
{
 "contentMessage" : {
   "text" : "Welcome to RCS chat!"
 },
 "ttl": "10s"
}
```

Ex3: TEXT with SUGGESTIONS

```
{
   "contentMessage": {
      "text": "Welcome to RCS chat!",
      "suggestions": [
         {
            "reply": {
               "text": "what is RCS?",
               "postbackData": "user_reply_what_is_rcs"
```

```
        }
      },
      {
        "action": {
          "text": "visit our website",
          "postbackData": "user_action_open_url",
          "openUrlAction": {
            "url": "https://www.dotgo.com"
          }
        }
      },
      {
        "action": {
          "text": "visit our website",
          "postbackData": "user_action_open_url",
          "openUrlAction": {
            "url": "https://www.dotgo.com"
          }
        }
      },
      {
        "action": {
          "text": "Show location on a map",
          "postbackData": "set_by_chatbot_open_map",
          "viewLocationAction": {
            "latLong": {
              "latitude": 37.4220041,
              "longitude": -122.0862515
            },
            "label": "Googleplex"
          }
        }
      },
      {
        "action": {
          "text": "Share location on a map",
          "postbackData": "set_by_chatbot_open_map",
          "shareLocationAction": {}
        }
      }
    ]
  }
}
```

Ex4: FILE WITH URL

```
{
 "contentMessage" : {
  "contentInfo" : {
      "fileUrl" : "https://konnect.kirusa.com/uploads/rcsTemplates/InstaVoice.png"
    }
  }
}
```

Ex5: FILE WITH SUGGESTIONS

```
{
  "contentMessage" : {
    "contentInfo" : {
      "fileUrl" : "https://konnect.kirusa.com/uploads/rcsTemplates/InstaVoice.png"
    },
    "suggestions" : [ {
      "reply" : {
        "text" : "what is RCS?",
        "postbackData" : "user_reply_what_is_rcs"
      }
    },
    {
      "action" : {
        "text" : "visit our website",
        "postbackData" : "user_action_open_url",
        "openUrlAction" : {
          "url" : "https://www.dotgo.com"
        }
      }
    }]
  },
  "expireTime": "2014-10-02T15:01:23Z"
}
```

Ex6: FILE WITH ID (uploaded file)

```
{
  "contentMessage" : {
    "fileName":"VLXtA7s35cGmyq6g9TcqCSEn2Uqi9QLR"
  }
}
```

Ex7: RICHCARD

```
{
  "contentMessage" : {
    "richCard" : {
      "standaloneCard" : {
        "cardContent" : {
          "media" : {
            "contentInfo" : {
              "fileUrl" :
"https://stagingchannels.kirusa.com/vobolo/blogs/9827/11914487_1609907755880.mp4",
              "forceRefresh" : false,
              "thumbnailUrl" :
"https://stagingchannels.kirusa.com/vobolo/blogs/9827/11914487_1609907755880_preview_500x500.pn
g"
            },
            "height" : "MEDIUM"
          },
          "title" : "Celebrity",
          "suggestions" : [ {
            "reply" : {
              "text" : "Like",
              "postbackData" : "user_like"
            }
          } ]
        },
        "thumbnailImageAlignment" : "LEFT",
        "cardOrientation" : "VERTICAL"
      }
    },
    "suggestions" : [ {
      "reply" : {
        "text" : "Know more",
        "postbackData" : "user_query"
      }
    } ]
  }
}
```

Ex7: RICH CARD CAROUSEL

```
{
  "contentMessage" : {
    "richCard" : {
      "carouselCard" : {
        "cardWidth" : "MEDIUM",
        "cardContents" : [ {
          "title" : "This is the first rich card in a carousel.",
          "description" : "This is the description of the rich card.",
          "media" : {
            "height" : "SHORT",
            "contentInfo" : {
              "fileUrl" : "https://konnect.kirusa.com/uploads/rcsTemplates/ReachMe.png",
              "forceRefresh" : false
            }
          }
        }, {
          "title" : "This is the second rich card in a carousel.",
          "media" : {
            "height" : "SHORT",
            "contentInfo" : {
              "fileUrl" : "https://www.google.com/logos/doodles/2015/googles-new-logo-5078286822539264.3-hp2x.gif",
              "forceRefresh" : false
            }
          }
        } ]
      }
    }
  }
}
```

Response :

Ex1:

200 OK

```
{
   "name": "phones/+919686960276/agentMessages/19e538c9-69ad-4c76-ac20-3e092a87f97a",
   "sendTime": "2023-01-16T07:37:29.787Z",
   "contentMessage": {
      "text": "Welcome to RCS chat!"
   }
}
```

Ex 2:

```
{
   "senderPhoneNumber": "+919686960276",
   "eventType": "FAILED",
   "reason": "Invalid client id!",
   "sendTime": "2023-02-01T04:45:44.206Z",
   "messageId": "00b53bde-9760-41f5-8f11-7a2f25b6b2ac",
   "code": 403
}
```

Error Code Details :

error_type	**error_code** (Within response, not http status code)	**error_message**
insufficient_balance	402	insufficient balance. Please recharge to send message
invalid_template	409	user is not in a conversation and provided message template is not approved

opted_out	410	opted_out
dnd_enabled	423	dnd_enabled
curfew_hrs	403	curfew_hrs
message_limit	429	monthly message limit exceeded
rcs_disabled	404	Number is rcs disabled
rate_limit	429	429 TOO_MANY_REQUESTS - You have reached the maximum number of times that this request can be made in a given time period
charge_failure	502	Unable to charge for this message.
internal_server_error	500	specific internal server error
4xx_maap_errors	400	maap specific error
invalid_client	403	Invalid client id
Invalid_template_data	400	‘Template with the code [temp01] and bot [TestAgent] received with insufficient custom params’ or Template with the code [temp01] and bot [TestAgent] is not approved, hence cannot be used for sending message.
google_rate_limit	400	429 TOO_MANY_REQUESTS - You have reached the allowed traffic limit for this agent on the user, your message will not be delivered.

Response Parameters:

Name	Type	Description	Remarks
messageId	A field in the Response body	Message Identifier for the message sent	
eventType	A field in the response body	Status of the message	The values are as defined in Google user event types. Additional to that there could be “PENDING” and “FAILED” events.

senderPhoneNumber	A field in the response body	User MSISDN in canonical form	Ex: +914253136789
eventId	A field in the response body	Unique Identifier for the event	
message	A field in the response body	Error message if request fails	

HTTP Response Codes

Code	Description
200	The request of sending message or isTyping indication is accepted by the Platform and ready to send to the user
400	This is a bad request with invalid input, invalid object, etc
403	Invalid client Id or Invalid IP address
401	This request is unauthorized.
5XX	Server error.

3.3. Message Status

Agent generated events for user messages can be sent through this API. Agents can send read event for a user message or typing indication to users.

Resource endpoint:
{serverRoot}/rcs/v1/phones/{phone_number}/agentEvents?botId={bot_id}&eventId={event_id}

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request Parameters:

Name	Type	Description	Remarks
phone_number	Path variable	User MSISDN in canonical form	Ex: +914253136789
botId	Query parameter	botId registered with the RCS APIs platform	
eventId	Query parameter	Unique identifier for event	Optional.

3.3.1. Send 'read' notification to users

A chatbot can send a 'read' notification to the end user for any received user message. This event indicates that a message has been opened or acknowledged. To users, this event appears as a read receipt for a specific message.

Request format:
URL :
POST
https://api.dotgo.com/rcs/v1/phones/+914253136789/agentEvents?botId=OsOsQ0GwNvUdLTV9Bd&eventId=6924cc84-dc0b-4679-a4ab-50c0875b309f

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

```
{
 "eventType": "READ",
 "messageId": "6924cc84-dc0b-4679-a4ab-50c0875b309f "
}
```

Response: 200 OK

No response body

HTTP Response Codes

Code	Description

200	The status of the message has been updated and a read notification will be sent to the user
400	This is a bad request with invalid input, invalid object, etc.
404	The message ID cannot be found
401	This request is unauthorized.
5XX	Server error.

3.3.2. Send Typing Event

The Chatbot can send typing indication to users. To a user, this event appears as a typing indicator and lets him know that agent is composing a message. The typing indicator expires after a short time (approximately 20 seconds) or when the user's device receives a new message from your agent. Your agent can send multiple IS_TYPING events to reset the typing indicator's expiration timer.

Request format:
URL :
POST
https://api.dotgo.com/rcs/v1/phones/+914253136789/agentEvents?botId=OsOsQ0GwNvUdLTV9Bd&eventId=6924cc84-dc0b-4679-a4ab-50c0875b309f
Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:

```
{
 "eventType" : "IS_TYPING"
}
```

Response: 200 OK
No response body

HTTP Response Codes

Code	Description
200	The isTyping event has been sent to user
400	This is a bad request with invalid input, invalid object, etc.
404	The message ID cannot be found
401	This request is unauthorized.
5XX	Server error.

3.4. Revoke a Message

Agent can revoke messages that it has sent but that have not been delivered to the user. There is a small chance that your bot may initiate a revocation while the MaaP is in the process of delivering the message. In these rare cases, your bot receives a DELIVERED event for the message shortly after initiating the revocation request.

Resource endpoint:
{serverRoot}/rcs/v1/phones/{phone_number}/agentMessages/{message_id}?botId={bot_id}

Request parameter:

Name	Type	Description	Remarks
botId	Query parameter	botId registered with the RCS APIs platform	
phone_number	Path variable	User msisdn in canonical form	Ex: +914253136789
message_id	Path variable	Identifier for the message to be revoked.	

Request format:

URL:
DELETE
https://api.dotgo.com/rcs/v1/phones/+914253136789/agentMessages/{message_id}?botId=
OsOsQ0GwNvUdLTV9Bd

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request: request body is optional. Include only if conversationId is available.

```
{
 "conversationId" : "a5a96c284d90ff352ac415e5a8ec213a"
}
```

Response Parameters:

Name	Type	Description	Remarks
conversationId	A field in the Response body	Identifier for the conversation. If request is sent through Vodafone MaaP, platform includes a conversationId in response. Agent must record this and send it back during revoke.	Optional. Include only if conversationId is available to agent(in case of Vodafone MaaP it will be available.)

Response :

Ex1 :
200 OK
No response body
Ex2: if messageId Not found.
404 NOT FOUND

```
{
   "status": "Error",
   "message": "Not found : Message Id"
}
```

Response Parameters:

Name	Type	Description	Remarks
Status	A field in the Response body	Status of the request	
Message	A field in the Response body	Reason for failure	

HTTP Response Codes

Code	Description
200	Revoke request has been accepted and platform will try to revoke if it's not delivered to user yet.
404	Message referred by the provided messageId not found.
401	This request is unauthorized.
5XX	Server error.

3.5. Check RCS capability of the contact/user

In RCS based communication service, the Chatbot shall only communicate with the user using an RCS capable device. So the Chatbot shall conduct the RCS capability discovery to learn about whether the given user's device is RCS capable or not.

Possible capabilities include "REVOCATION","RICHCARD_STANDALONE","RICHCARD_CAROUSEL","ACTION_CREATE_CALENDAR_EVENT","ACTION_DIAL","ACTION_OPEN_URL","ACTION_SHARE_LOCATION","ACTION_VIEW_LOCATION".

Resource endpoint:
{serverRoot}/rcs/v1/phones/{phone_number}/capabilities?botId={bot_id}&requestId={request_id}

Request Parameters:

Name	Type	Description	Remarks
botId	Query parameter	botId registered with the RCS APIs platform	
phone_number	Path variable	User msisdn in canonical form	Ex: +914253136789

requestId	Query parameter	Unique identifier for request	Optional.

Request format:

URL:
GET
https://api.dotgo.com/rcs/v1/phones/+914253136789/capabilities?botId=
OsOsQ0GwNvUdLTV9Bd&requestId= 2918644c-9b58-4cf9-a598-3eec750ad0e3

Authorization: Access Token obtained from Auth2 SSO service as Bearer token
Request:

No request body

Response :

Ex1: 200 - Device capabilities will be returned

```
{
 "features": [
   "REVOCATION",
   "RICHCARD_STANDALONE",
   "RICHCARD_CAROUSEL",
   "ACTION_CREATE_CALENDAR_EVENT",
   "ACTION_DIAL",
   "ACTION_OPEN_URL",
   "ACTION_SHARE_LOCATION",
   "ACTION_VIEW_LOCATION",
 ]
}
```

Ex2: 404 - if RCS disabled, No capabilities.

```
{
   "error": {
        "code": 404,
        "message": "Requested entity was not found.",
        "status": "NOT_FOUND"
    }
}
```

Response Parameters:

Name	Type	Description	Remarks
features	A field in the Response body	capabilities supported by contact/user	These capabilities need to be interpreted as specified in Google API specifications.

HTTP Response Codes

Code	Description
200	User is RCS enabled and supported capabilities will be returned
404	user's device is not RCS capable.
401	This request is unauthorized.
5XX	Server error.

3.6. Upload a file to the platform

When agent sends a message with an image or video, agent must provide a publicly accessible URL for the content. If not, agent can upload a media file binary to Dotgo CDN and then send the media in a message. The Dotgo RCS platform keeps files for 60 days, and the API returns a file Id that your agent can include in messages to users. After 60 days, the file will be removed from CDN.

Resource endpoint:

{serverRoot}/rcs/upload/v1/files?botId={botId}

Name	Type	Description	Remarks
botId	Query parameter	botId registered with the RCS APIs platform	

Request format:

URL:
POST
https://api.dotgo.com/rcs/upload/v1/files?botId={botId}

Authorization: Access Token obtained from Auth2 SSO service as Bearer token

Request:
binary
contentType – content type of the file (image/jpeg, video/mp4, image/png, etc..)
select the file

Response :

Ex1 : returns Identifier for the uploaded file.
200 OK

```
{
 "name": "9mEFZnVTBsycGwtVTH7BovQwuz0Msr9p"
}
```

Ex2: Failed to save the file.
500 INTERNAL SERVER ERROR

```
{
   "status": "Error",
   "message": "Processing failed : Unable to store the file."
}
```

Response Parameters:

Name	Type	Description	Remarks

name	A field in the Response body	Unique identifier for the uploaded file	This value can be used in file messages, instead or file url.
status	A field in the Response body	Status of the request	
message	A field in the Response body	Reason for failure	

HTTP Response Codes

Code	Description
200	Revoke request has been accepted and platform will try to revoke if it's not delivered to user yet.
401	This request is unauthorized.
5XX	Server error.

3.7. Add Tester

Chatbots can use this feature to register a tester device for Google MaaP.

Resource endpoint:
{serverRoot}/rcs/v1/phones/{phone_number}/testers?botId={bot_id}

Request Parameters:

Name	Type	Description	Remarks
botId	Query parameter	botId registered with the RCS APIs platform	
phone_number	Path variable	User msisdn in canonical form	Ex: +914253136789

Request format:

URL:
POST
https://api.dotgo.com/rcs/v1/phones/{phone_number}/testers?botId=OsOsQ0GwNvUdLTV9Bd

Authorization: Access Token obtained from Auth2 SSO service as Bearer token

Request:

No request body

Response :

```
{
   "statusCode": "Success",
   "response": "Submitted, Tester Invite Sent"
}
```

HTTP Response Codes

Code	Description
200	If tester added, success message in the response body. If failed to add tester, a failure message along with the reason will be returned.
401	This request is unauthorized.
404	Number not found.
5XX	Server error.

3.8. Webhook

Webhook is the callback API provided by the Chatbot. Dotgo RCS APIs Platform uses the webhook exposed by Chatbot to send an HTTP POST payload when certain RCS events occur.

A chatbot may receive the following events from the RCS-APIs platform:

1. message from a user (text, file, location, or audio message)
2. isTyping notification from a user
3. message status update
4. response to suggested reply or action

The Chatbot shall always return a 200 OK HTTP response to the HTTP POST from the RCS APIs Platform.

The payload that can be sent from the **Vi** RCS APIs platform to the developer Chatbot's webhook is as defined in Google API specifications.

Every request sent on webhook will be base64 encoded in case of google format and set to 'data' field as below.

```
{
        "subscription": "projects/rbm-stagingtestagent-ngu7dtz/subscriptions/rbm-agent-subscription",
        "message": {
         "data":
        "ewogICJzZW5kZXJQaG9uZU51bWJlciI6ICIrOTE5Njg2OTYwMjc2IiwKICAibWVzc2FnZUlk
        IjogIk14SWdULTRwV0NRdTJ0SDloTWtTVGxqUSIsCiAgInNlbmRUaW1lIjogIjIwMjItMTEtM
        TdUMDU6MTU6MTQuMzg0OTYzWiIsCiAgInRleHQiOiAiVGVzdCIsCiAgImFnZW50SWQi
        OiAic3RhZ2luZ3Rlc3RhZ2VudEByYm0uZ29vZyIKfQ==",
         "attributes": {
            "project_number": "129459109761",
            "product": "RBM",
            "message_type": "TEXT",
            "business_id": "stagingtestagent@rbm.goog",
            "type": "message"
         },
         "messageId": "6267774379637729",
         "publishTime": "2022-11-17T05:15:14.538Z"
         }
}
```

3.8.2.1. Example

3.8.2.1.1. Message from a user

```
{
  "senderPhoneNumber": "PHONE_NUMBER",
  "messageId": "0a99d150-aae7-4247-aa07-a92cdaaf8ed3",
  "sendTime": "2018-12-31T15:01:23.045123456Z",
  "text": "Hello, world!",
  "agentId": " stagingtestagent@rbm.goog"
}
```

3.8.2.1.2. isTyping message from a user

```
{
  "senderPhoneNumber": "+914253136789",
  "eventType": "IS_TYPING",
  "eventId": "47ace754-3191-4101-b494-658cfb314881",
  "sendTime": "2018-12-31T15:01:23.045123456Z"
  "agentId": " stagingtestagent@rbm.goog"
}
```

3.8.2.1.3. Message status update

```
{
  "senderPhoneNumber": "+914253136789",
  "eventType": "DELIVERED",
  "eventId": "fa2fe5a2-d9a9-4d83-87d3-302ae1014610",
  "messageId": "57bed79e-55ba-46fe-b88a-2755aaee77fc",
  "agentId": " stagingtestagent@rbm.goog"
}
```

3.8.2.1.4. Response to suggested reply/action

```
{
  "senderPhoneNumber": "+914253136789",
  "messageId": "4a1d7c74-2b3c-4ec7-b6be-ed7205a15aa3",
  "sendTime": "2018-12-31T15:01:23.045123456Z",
  "agentId": " stagingtestagent@rbm.goog",
  "suggestionResponse": {
    "postbackData": "suggestion_1",
    "text": "Suggestion #1" }
}
```

3.8.2.1.5. Response to suggested reply/action of templates (Template description has been provided in the next section)

```
{
   "suggestionResponse":{
      "postbackData":"user_clicked_Reach_Us",
      "text":"Reach Us",
      "type":"ACTION",

"metadata":"{\"suggestionType\":\"url_action\",\"templateType\":\"carousel\",\"cardIndex
\":1,\"suggestionIndex\":2,\"msgId\":\"1ed15c61-7fd1-436a-a127-
0baa8d91cdb2\",\"a2pMsgDate\":\"2023-07-03T15:22:00.356\",\"templateCode\":\"test
template\"}"
   },
   "senderPhoneNumber":"+919986473361",
   "messageId":"MxNsfgtg86T-6-=o2i7P3pGw",
   "sendTime":"2023-06-20T15:39:30.475341Z",
   "agentId":"dotgo-gupshup_r0mhjgkx_agent@rbm.goog"
}
```

The metadata fields will be additionally sent as part of the payload to the agent webhook in case the user responses are within A2P message templates. This will include the following:

SuggestionType: reply, dialer_action, url_action

templateType: text_message, rich_card, carousel

cardIndex: in case of a carousel, the index of the card for which the user event is generated (index starts from 0)

suggestionIndex: index of the suggestion starting from 0

msgId: the message Id of the original A2P message

a2pMsgDate: date in which the original A2P message was sent

templateCode: name of the template

3.8.2.1.6. ttl_expiration_revoke event (The message has expired and was successfully revoked. In this case the money will be refunded for such messages.)

```
{
  "phoneNumber": "+919986473361" ,
  "messageId": ”a26d6809-9e9a-4c77-ad82-e55b35b80ef1”,
  "agentId": "dotgo-gupshup_r0mhjgkx_agent@rbm.goog",
  "eventType": "TTL_EXPIRATION_REVOKED",
  "eventId": "MxNsfgtg86T-6-=o2i7tryGw",
  "sendTime": "2023-06-20T15:39:30.475341Z"
}
```

3.8.2.1.7. ttl_expiration_revoke_failed (The message has expired, but it was not revoked.)

```
{
  "phoneNumber": "+919986473361",
  "messageId": ”b6586809-9e9a-4c77-ad82-e55b35b80ef1”,
  "agentId": "dotgo-gupshup_r0mhjgkx_agent@rbm.goog",
  "eventType": "TTL_EXPIRATION_REVOKE_FAILED",
  "eventId": "MxNsfgtg8sjfjk6=o2i7P3pGw",
  "sendTime": "2023-06-20T15:39:30.475341Z"
}
```

4. Templates

Dotgo RBM platform lets you add rich media such as images, carousels, and videos, along with customized branding, action buttons, and suggested replies into the notifications and promotional messages.

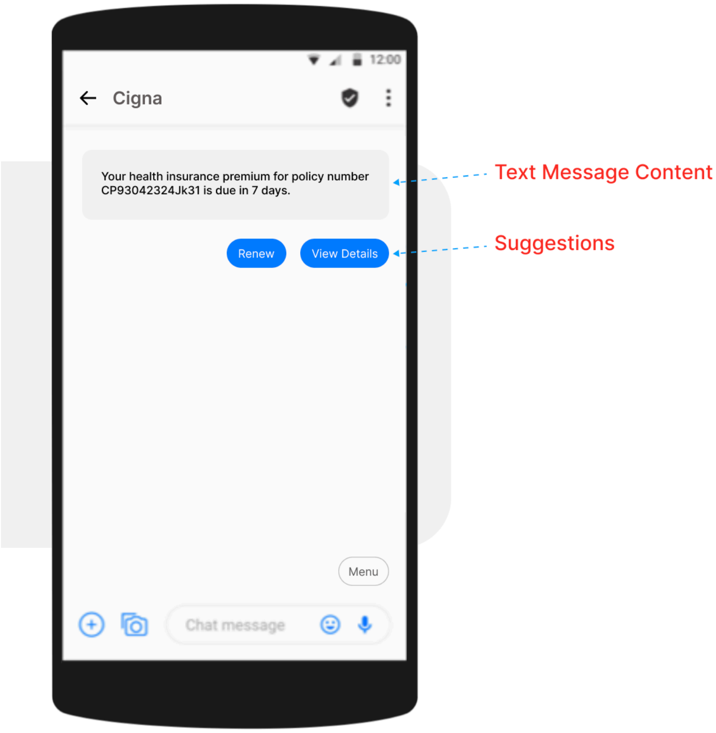
These template messages are delivered using the underlying RCS APIs.

4.1. What is an RCS Template

An RCS Template is a predefined set of RBM UI elements for e.g., a Rich Card with suggestions which can be used as a base for formulating messages in your campaign to the target audience. Once a bot is created, the aggregator can create multiple templates against the bot and use them for running campaigns. You can add images, videos, and suggested actions for your template.
Dotgo RBM portal supports 3 types of templates:

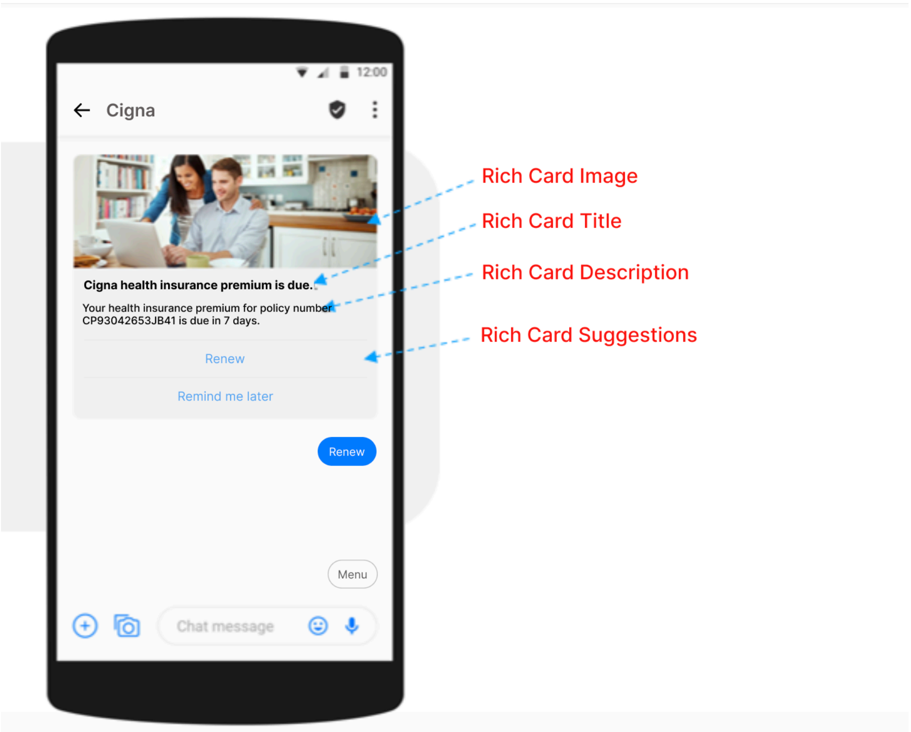
- Text Message
- Rich Card Stand-alone
- Rich Card Carousel

See figures below for examples of these template messages.



Your health insurance premium for policy number CP93042324Jk31 is due in 7 days.

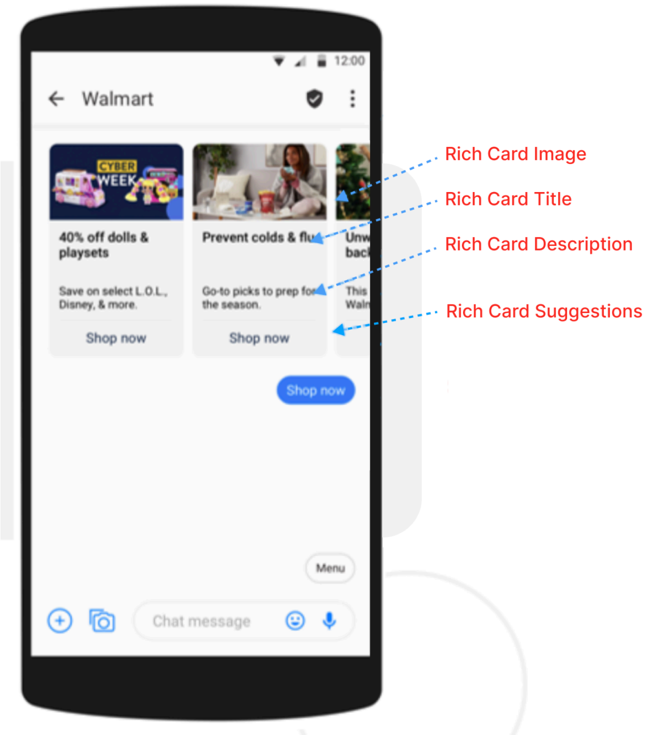
Text Message Template



Cigna health insurance premium is due.

Your health insurance premium for policy number CP93042653JB41 is due in 7 days.

Rich Card Stand-alone Template



40% off dolls & playsets

Save on select L.O.L., Disney, & more.

Prevent colds & flu

Go-to picks to prep for the season.

Rich Card Carousel Template

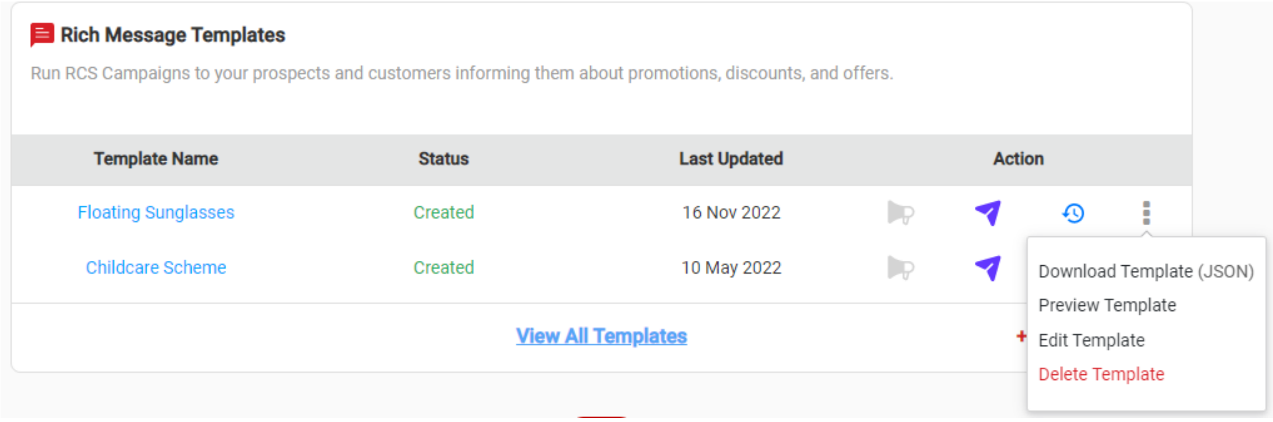
4.2. Template Approval Process

If the template approval feature is enabled at the country level, then any template created needs to be approved by Dotgo and(or) Carrier Admin.

If this configuration is not enabled,
then no approval is required prior to sending RCS traffic using the templates.

The text below assumes that template approval is required.
Once it is approved, you can send a message either to a tester (by adding the end user's phone number as a test device) if the bot is not yet launched, or directly send a message to a set of users if the bot is launched in the selected carrier(s).
You can send a message in either of the following ways:

1. Using the promotions/notifications API and passing the template code (name). This is the recommended approach.
2. Using the native messaging RCS API (GSMA or Google styled). To use this, you need to strictly adhere to the **mediaURL** parameter as received in the request format downloaded via clicking on the option 'Download Template JSON' available against each template created. For eg., https://dotgo.com/rcsbotdirectory/rcs_message/media/UB7ONL1264NIX5YPST3E.jpeg



Rich Message Templates

Run RCS Campaigns to your prospects and customers informing them about promotions, discounts, and offers.

Template Name	Status	Last Updated	Action
Floating Sunglasses	Created	16 Nov 2022	
Childcare Scheme	Created	10 May 2022	

The above URL is created by the Dotgo platform when the user uploads the media during template creation. The URL denotes the Dotgo RBM content server path and will be used for template matching in the incoming request. Hence, the **same URL** needs to be passed (for the

same media) while calling the RCS send message API. If any other **mediaURL** is passed, the request will be rejected because it will not match the approved template details.
For fixed or personalized content, please ensure that the requested message matches with the approved template, else it will be rejected. The values of the variables can only change for a personalized template, not the message itself.

For eg., if your approved template is "***The fastest LTE service around. Get you connected anytime***", then the incoming message "***The best LTE service around. Get you connected anytime***" will be rejected, since the latter does not match the approved template.

Another example (with **personalization**):
If the approved template is the below:
Get [DISCOUNT] discount for 6 months on all Vivo's High Speed fiber optic broadband internet plans + Wi-Fi and free installation + Access to digital services such as Paramount+, Band News, TNT & Sports Stadium. Use code [CODE] when ordering.

Then the following message will be accepted:
Get 20% discount for 6 months on all Vivo's High Speed fiber optic broadband internet plans + Wi-Fi and free installation + Access to digital services such as Paramount+, Band News, TNT & Sports Stadium. Use code DISC when ordering.

But the following message will be rejected (since it does not match the approved template):
Get discount of 20% for 6 months on all Vivo's High Speed fiber optic broadband internet plans + Wi-Fi and free installation + Access to digital services such as Paramount+, Band News, TNT & Sports Stadium. Use the DISC code when ordering.

5. API Throttling

The RCS API Platform defines configurations for the throttling limits for the API described in the previous sections. This means that there is an upper limit to the rate of API requests that the RCS API platform will accept from each account.

The API Platform has two separate configurations for defining the throttling limits:

Group 1 Rate Limit

- This is defined in terms of API requests per second (TPS).
- Group 1 Rate Limit applies to following requests:

  - Individual Capability check
  - Sending a Message
  - Revoking a Message
- The TPS defined above is the average rate accepted by the API Platform. In addition to that, the platform allows short time spikes of 20% of the configured TPS, as long as the load is below the configured TPS over a 30 second period.
- For example, if the TPS is set at 50, developer can send up to 60 TPS, as long as the total number of requests is no more than 1,500 in a 30 second window (50 TPS x 30 seconds = 1,500).

Group 2 Rate Limit

- This is defined in terms of API requests per minute (TPM).
- Group 2 Rate Limit applies to Bulk Capability Check

5.1. API Request Limits Exceeded

In the case that the number of requests for any given account or a given agent exceeds the throttling limit, the response object will be as below:

5.1.1. GSMA API

```
 HTTP status - 429 Too Many Requests
{
   "RCSMessage": {
      "msgId": "b1ef710e-cc77-4b73-a3dc-d149dcae3aad",
      "status": "failed",
      "timestamp": "2023-03-07T10:42:50.615Z"
   },
   "reason":

{
      "text": " Number of API requests allowed per second exceeded. Please retry "
```

```
  }
}
```

5.1.2. Google API

HTTP status - 429 Too Many Requests

```
{
   "senderPhoneNumber": "+919686960276",
   "eventType": "FAILED",
   "reason": " Number of API requests allowed per second exceeded. Please retry"",

   "sendTime": "2023-03-07T10:45:37.959Z",
   "messageId": "e4d4529f-0336-4292-8a39-34964cad2bd3"
}
```

5.2. Rate Limits for your Account

Both Group 1 and Group 2 Rate Limits are configured and enforced at account level. Please contact rbm-support@dotgo.com for the value of TPS configured for your account.

6. Google Business Messages APIs

6.1. Authorization

To access GBM APIs of the Dotgo API platform, an access token with 'Chatbot-message' and 'google' scope needs to be provided.
Register a ServiceClient with scope 'Chatbot-message' and 'google' to receive OAuth2client_credentials. Using the credentials obtained from SSO, an access token can be generated which in turn should be used as an Authorization Bearer token, to access GBM APIs of Dotgo platform.

Access token can be obtained from these APIs :

https://auth.dotgo.com/auth/oauth/token?grant_type=client_credentials

To get the token, the clientId and client_secret should be sent in the Authorization header as Basic authentication (base64 encoded).

6.2. Communicate with user

Resource endpoint: {serverRoot}/gbmapi/{botId}/{conversationId}/messages
This is the API used to send messages to users.

conversationId is the user level mapping in Google Business Messages(GBM). It is generated by GBM when the user initiates a conversation with the GBM bot.

Request format:
URL: {serverRoot}/gbmapi/gbm-bot/123qwe4r5t/messages
Authorization: Access Token obtained from Auth2 SSO service as Bearer token
Request:

```
{
    "representative": {
        "avatarImage": "https://konnect.kirusa.com/uploads/gbm_bot_logo.png",
        "displayName": "GBM Bot",
        "representativeType": "BOT"
    },
    "text": "Hello from GBM"
}
```

Request Parameters:

Name	Type	Description	Remarks
botId	Path variable	botId registered with the RCS APIs platform	
conversationId	Path variable	The unique identifier of the conversation	Ex: 278bab79a-bf6e-493e-82d2-910f3d08da79

Response Format:

```
{
  "conversationId": "123qwe4r5t",
  "customAgentId": "gbm-bot",
  "requestId": "e059df24-c470-4f9b-9187-557becc63148",
  "receipts": {
    "receipts": [
      {
        "message": "conversations/278ba79a-bf6e-493e-82d2-
910f3d08da79/messages/31374a36-5840-4a32-af70-a9820e6ca216",
        "receiptType": "PENDING"
      }
    ],
    "createTime": "2021-10-04T10:43:50.493"
  }
}
```

Response Parameters:

Name	Type	Description	Remarks
Receipts.receipts.message	A field in the Response body	Format : conversations/{conversationId}/messages/{messageId} {messageId} Identifier for the message sent	
Receipts.receipts.receiptType	A field in the response body	Status of the message	The values are as defined in the Receipt Type of GBM MaaP Chatbot API specifications

HTTP Response Codes

Code	Description
202	The request of sending message or is accepted by the Platform and ready to send to the user
400	This is a bad request with invalid input, invalid object, etc
401	This request is unauthorized.
5XX	Server error.

6.2.1. Send a Message to the User

Various types of messages that can be sent to users, including text message, file, rich card, and suggested chip list are as per GBM MaaP Chatbot API specifications.

6.2.1.1. Example

POST {serverRoot}/gbmapi/gbm-bot/123qwe4r5t/messages
Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:
Ex1. Text Message

```
{
    "representative": {
        "avatarImage": "https://konnect.kirusa.com/uploads/gbm_bot_logo.png",
        "displayName": "GBM Bot",
        "representativeType": "BOT"
    },
    "text": "Hello from GBM"
}
```

Ex2. File Message

```
{
    "representative": {
        "avatarImage": "https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png ",
        "displayName": "Sylvan Marlton",
        "representativeType": "BOT"
    },
    "image": {
        "contentInfo": {
            "fileUrl": "https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png",
            "thumbnailUrl": "https://konnect.kirusa.com/uploads/rcsTemplates/sylvan_marlton/sylvan_home_logo.png",
            "forceRefresh": false,
            "altText": "Hi"
        }
    }
}
```

Ex3. File Message with suggestions

```
{
    "messageId": "c6613b66-91fc-479c-9301-eb249b40eoiko",
    "representative": {
        "avatarImage": "https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png ",
        "displayName": "Sylvan Marlton",
        "representativeType": "BOT"
    },
    "suggestions": [{
        "reply": {
            "postbackData": "{\"data\":\"office_info\"}",
            "text": "Office Info"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"learning_options\"}",
            "text": "Learning Options"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"special_offers\"}",
            "text": "Special Offers"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"how_we_compare\"}",
            "text": "How We Compare?"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"safe_screening\"}",
            "text": "Safe Screening"
        }
    }, {
        "action": {
            "openUrlAction": {
                "url": "https://www.facebook.com/sylvanofmarltonnj/"
            },
            "postbackData": "{\"data\":\"open_url\"}",
            "text": "Our Facebook page"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"live_chat\"}",
            "text": "Live Chat"
        }
    }],
    "image": {
        "contentInfo": {
            "fileUrl": "https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png ",
            "thumbnailUrl": "https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.p
ng",
            "forceRefresh": false,
            "altText": "GBM BOT"
        }
    }
}
```

The following table provides the file types allowed:
The max size mentioned below for images and videos applies to sending direct file messages (and **not** for rich cards/carousels).
For rich cards and carousels, please click below for file size recommendations and limits:
https://developers.google.com/business-communications/rcs-business-messaging/guides/learn/best-practices

Media Type	**Document Type**	**Extension**	**Max Size**	**Works with rich cards**
application/pdf	PDF	.pdf	100 MB	No
image/jpeg	JPEG	.jpeg, .jpg	100 MB	Yes

image/gif	GIF	.gif	100 MB	Yes
image/png	PNG	.png	100 MB	Yes
video/h263	H263 video	.h263	100 MB	Yes
video/m4v	M4V video	.m4v	100 MB	Yes
video/mp4	MP4 video	.mp4	100 MB	Yes
video/mpeg4	MPEG-4 video	.mp4, .m4p	100 MB	Yes
video/mpeg	MPEG video	.mpeg	100 MB	Yes
video/webm	WEBM video	.webm	100 MB	Yes

Ex4. Rich Card with Suggested chiplist

```
{
    "representative": {
        "avatarImage": "https://konnect.kirusa.com/uploads/ gbmTemplates/gbm_bot_home_logo.png
",
        "displayName": "Sylvan Marlton",
        "representativeType": "BOT"
    },
    "suggestions": [{
        "reply": {
            "postbackData": "{\"data\":\"office_info\"}",
            "text": "Office Info"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"learning_options\"}",
            "text": "Learning Options"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"special_offers\"}",
            "text": "Special Offers"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"how_we_compare\"}",
            "text": "How We Compare?"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"safe_screening\"}",
            "text": "Safe Screening"
        }
    }, {
        "action": {
            "openUrlAction": {
                "url": "https://www.facebook.com/sylvanofmarltonnj/"
            },
            "postbackData": "{\"data\":\"open_url\"}",
            "text": "Our Facebook page"
        }
    }, {
        "reply": {
            "postbackData": "{\"data\":\"live_chat\"}",
            "text": "Live Chat"
        }
    }],
    "richCard": {
        "standaloneCard": {
            "cardContent": {
                "title": "it's a richcard",
                "description": "richcard with suggestions",
                "media": {
                    "height": "MEDIUM",
                    "contentInfo": {
                        "fileUrl": "https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home
_logo.png ",
                        "thumbnailUrl": "https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot
_home_logo.png ",
                        "forceRefresh": false,
                        "altText": "Hi"
                    }
                },
                "suggestions": [{
                    "reply": {
                        "postbackData": "{\"data\":\"office_info\"}",
                        "text": "Office Info"
                    }
                    }, {
                    "reply": {
                        "postbackData": "{\"data\":\"learning_options\"}",
                        "text": "Learning Options"
                    }
                }]
            }
        }
    }
```

```
}
```

Response: 202 Accepted

```
{
  "conversationId": "123qwe4r5t",
  "customAgentId": "gbm-bot",
  "requestId": "e059df24-c470-4f9b-9187-557becc63148",
  "receipts": {
    "receipts": [
      {
        "message": "conversations/278ba79a-bf6e-493e-82d2-
910f3d08da79/messages/31374a36-5840-4a32-af70-a9820e6ca216",
        "receiptType": "PENDING"
      }
    ],
    "createTime": "2021-10-04T10:43:50.493"
  }
}
```

6.2.2. Send Typing and Human Representative indication to the user

Resource endpoint: {serverRoot}/gbmapi/{botId}/{conversationId}/events
This is the API used to send events to the user.

conversationId is the user level mapping in Google Business Messages(GBM). It is generated by GBM when the user initiates a conversation with the GBM bot.

Request format:
URL: {serverRoot}/gbmapi/gbm-bot/123qwe4r5t/events
Authorization: Access Token obtained from Auth2 SSO service as Bearer token
Request:

```
{
  "eventType":"TYPING_STARTED",
  "representative":{
     "displayName":"GBM BOT",
     "avatarImage":"https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png",
     "representativeType":"BOT"
  }
}
```

Request Parameters:

Name	Type	Description	Remarks
eventType	A field in the Request body	Type of an agent event	typing & human representative is communicating indication to users as defined in Event Type

			of GBM MaaP Chatbot API specifications
conversationId	Path variable	The unique identifier of the conversation	Ex: 278bab79a-bf6e-493e-82d2-910f3d08da79

Response Format:

```
{ "name" :"conversations/278ba79a-bf6e-493e-82d2-910f3d08da79/events/f6ec77c1-b1c2-4759-
a793-68c773c72dcb"}
```

Response Parameters:

Name	Type	Description	Remarks
name	A field in the Response body	Format : conversations/{conversationId}/events/{eventId} {eventId} Identifier for the event sent	

6.2.2.1. Example

POST {serverRoot}/gbmapi/gbm-bot/123qwe4r5t/events
Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Request:
Ex1. Typing Started

```
{
   "eventType":"TYPING_STARTED",
   "representative":{
      "displayName":"GBM BOT",
      "avatarImage":"https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png",
      "representativeType":"BOT"
   }
}
```

Ex1. Typing Stopped

```
{
   "eventType":"TYPING_STOPPED",
   "representative":{
      "displayName":"GBM BOT",
      "avatarImage":"https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png",
      "representativeType":"BOT"
   }
}
```

Ex1. Representative Joined

```
{
   "eventType":"REPRESENTATIVE_JOINED",
   "representative":{
      "displayName":"GBM BOT",
      "avatarImage":"https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png",
      "representativeType":"BOT"
   }
}
```

Ex1. Representative Left

```
{
   "eventType":"REPRESENTATIVE_LEFT",
   "representative":{
      "displayName":"GBM BOT",
      "avatarImage":"https://konnect.kirusa.com/uploads/gbmTemplates/gbm_bot_home_logo.png",
      "representativeType":"BOT"
   }
}
```

Response: 202 Accepted

```
{
  "name" :"conversations/278ba79a-bf6e-493e-82d2-910f3d08da79/events/f6ec77c1-b1c2-4759-
a793-68c773c72dcb"
       }
```

6.2.3. Send survey to the user

Resource endpoint: {serverRoot}/gbmapi/{botId}/{conversationId}/survey

This is the API used to send a survey to the user.
A survey can be initiated only once in 24 hours by a bot with a **conversationId**.

conversationId is the user level mapping in Google Business Messages(GBM). It is generated by GBM when the user initiates a conversation with the GBM bot.

Request format:
URL: {serverRoot}/gbmapi/gbm-bot/123qwe4r5t/survey
Authorization: Access Token obtained from Auth2 SSO service as Bearer token
Request:

```
 No request body is sent for survey
```

Request Parameters:

Name	Type	Description	Remarks
conversationId	Path variable	The unique identifier of the conversation	Ex: 278bab79a-bf6e-493e-82d2-910f3d08da79

Response Format:

```
{
  "name": "conversations/fd9e891d-26fc-4d6c-ba8c-e71eb1f1e7b3/surveys/27477560-f7d6-
41f4-bf20-6727c38cc8be"
}
```

Response Parameters:

Name	Type	Description	Remarks
name	A field in the Response body	Format : conversations/{conversationId}/surveys/{surveyId} {surveytId} Identifier for the survey sent	

HTTP Response Codes

Code	Description
200	The request of sending survey or is accepted by the Platform and ready to send to the user
429	If more than one survey is sent within 24 hours
401	This request is unauthorized.
5XX	Server error.

6.2.3.1. Example

POST {serverRoot}/gbmapi/gbm-bot/123qwe4r5t/events

Authorization: Access Token obtained from Auth2 SSO service as Bearer token.

Response: 202 Accepted

```
{
  "name" :"conversations/278ba79a-bf6e-493e-82d2-910f3d08da79/events/f6ec77c1-b1c2-4759-
a793-68c773c72dcb"
}
```

Response: 429 Too Many Requests

```
{
  "error": {
    "code": 429,
    "message": "This conversation received a survey in the past 24 hours.",
    "status": "RESOURCE_EXHAUSTED"
  }
}
```

6.3. Webhook

Webhook is the callback API provided by the Chatbot. Dotgo APIs Platform uses the webhook exposed by Chatbot to send an HTTP POST payload when certain GBM events occur.

A chatbot may receive the following events from the Dotgo-APIs platform:

1. message from a user (text, file, location, or audio message)
2. response to the survey
3. Delivered Receipt
4. response to suggested reply or action
5. Typing started event
6. Typing Stopped event
7. Request for a live agent (human) to converse with

The Chatbot shall always return a 200 OK HTTP response to the HTTP POST from the Dotgo APIs Platform.

6.3.4. Webhook Payload from Dotgo-APIs platform

The payload that can be sent from the Dotgo APIs platform to Chatbot’s webhook is as per the WebhookPayload model specified in GBM MaaP Chatbot API specifications.

6.3.4.1. Example

6.3.4.1.1. Message From User

```
{
    "message": {
        "name": "conversations/278ba79a-bf6e-493e-82d2-910f3d08da79/messages/caa03cfb-70f2-4b2b-b24a-c9fae1a699c1-161024167220191602701",
        "text": "menu",
        "createTime": "2021-02-24T11:47:01.252113Z",
        "messageId": "caa03cfb-70f2-4b2b-b24a-c9fae1a699c1-161024167220191602701"
    },
    "context": {
        "placeId": "",
        "entryPoint": "URL",
        "userInfo": {
            "displayName": "User 1",
            "userDeviceLocale": "en-IN"
        },
        "resolvedLocale": "en",
        "customContext": "parking_space"
    },
    "sendTime": "2021-02-24T11:47:01.970546Z",
    "conversationId": "278ba79a-bf6e-493e-82d2-910f3d08da79",
    "customAgentId": "gbm-bot",
    "requestId": "caa03cfb-70f2-4b2b-b24a-c9fae1a699c1-16141601272209026022027O1",
    "agent": "brands/0c018bb7-67cb-4133-abbf-bd7f351f4ee0/agents/73a769da-46b0-4e87-82ab-
eac6da6ed540"
}
```

6.3.4.1.2. Response to Survey

```
{
  "context": {
    "placeId": "",
    "userInfo": {
      "displayName": "User 1"
    }
  },
  "sendTime": "2021-05-17T09:53:54.910533Z",
  "conversationId": "278ba79a-bf6e-493e-82d2-910f3d08da79",
  "customAgentId": "gbm-bot",
  "requestId": "62efb854-abc3-4f96-863a-1b508a92c2f7",
  "surveyResponse": {
    "survey": "conversations/278ba79a-bf6e-493e-82d2-910f3d08da79/surveys/1b8c5989-555c-4487-
b08c-5039eee663be",
    "rating": "SOMEWHAT_SATISFIED",
    "createTime": "2021-05-17T09:53:54.235503Z"
  },
  "agent": "brands/0c018bb7-67cb-4133-abbf-bd7f351f4ee0/agents/73a769da-46b0-4e87-82ab-
eac6da6ed540"
}
```

6.3.4.1.3. Message Status Update

```
{
  "context": {
    "placeId": "",
    "userInfo": {
      "displayName": "User 1"
    }
  },
  "sendTime": "2021-05-17T10:00:57.408482Z",
  "conversationId": "278ba79a-bf6e-493e-82d2-910f3d08da79",
  "customAgentId": "gbm-bot",
  "requestId": "1b845c29-07a0-434f-a392-1ae226c3779299",
  "receipts": {
    "receipts": [{
      "message": "conversations/278ba79a-bf6e-493e-82d2-910f3d08da79/messages/7210d9d1-8795-
4617-af24-b1027aa5e077",
      "receiptType": "DELIVERED"
    }],
    "createTime": "2021-05-17T10:00:56.648631Z"
  },
  "agent": "brands/0c018bb7-67cb-4133-abbf-bd7f351f4ee0/agents/73a769da-46b0-4e87-82ab-
eac6da6ed540"
}
```

6.3.4.1.4. Response to Suggested reply/action

```
{
    "suggestionResponse": {
        "message": "conversations/82b09a5d-7b3d-44d7-b95e-7bb53d2d0e37/messages/8bd39e25-7e57-
4678-98cc-89c98672002a91",
        "postbackData": "{\"data\":\"office_info\"}",
        "createTime": "2021-02-24T11:38:11.360950Z",
        "text": "Office Info",
        "type": "REPLY"
    },
    "context": {
        "placeId": "",
        "entryPoint": "URL",
        "userInfo": {
            "displayName": "User 1",
            "userDeviceLocale": "en-IN"
        },
        "resolvedLocale": "en",
        "customContext": "parking_space"
    },
    "sendTime": "2021-02-24T11:38:12.152721Z",
    "conversationId": "82b09a5d-7b3d-44d7-b95e-7bb53d2d0e37",
    "customAgentId": "gbm-bot",
    "requestId": "80e1f1b0-214b-43c3-a444-b25e72ad18d5-161416669100541",
    "agent": "brands/0c018bb7-67cb-4133-abbf-bd7f351f4ee0/agents/73a769da-46b0-4e87-82ab-
eac6da6ed540",
    "msgType": "suggestedResponse",
    "textMessage": false,
    "suggestedResponse": true,
    "receipt": false,
    "authResponse": false
}
```

6.3.4.1.5. Typing started event

```
{
  "context": {
    "placeId": "ChIJh3I4p5E0wYkRP6iGd6j1RU8",
    "entryPoint": "MAPS",
    "userInfo": {
      "displayName": "User 1",
      "userDeviceLocale": "en-GB"
    },
    "resolvedLocale": "en"
  },
  "sendTime": "2021-05-17T10:17:36.652229Z",
  "conversationId": "82b09a5d-7b3d-44d7-b95e-7bb53d2d0e37",
  "customAgentId": "gbm-bot",
  "requestId": "2bcbabf7-747f-4574-a02a-1643ee018401021ee",
  "userStatus": {
    "isTyping": true,
    "createTime": "2021-05-17T10:17:35.300310Z"
  },
  "agent": "brands/0c018bb7-67cb-4133-abbf-bd7f351f4ee0/agents/73a769da-46b0-4e87-82ab-
eac6da6ed540"
}
```

6.3.4.1.6. Typing stopped event

```
{
  "context": {
    "placeId": "ChIJh3I4p5E0wYkRP6iGd6j1RU8",
    "entryPoint": "MAPS",
    "userInfo": {
      "displayName": "User 1",
      "userDeviceLocale": "en-GB"
    },
    "resolvedLocale": "en"
  },
  "sendTime": "2021-05-17T10:17:36.652229Z",
  "conversationId": "278ba79a-bf6e-493e-82d2-910f3d08da79",
  "customAgentId": "gbm-bot",
  "requestId": "2bcbabf7-747f-4574-a02a-1643ee018402ee",
  "userStatus": {
    "isTyping": false,
    "createTime": "2021-05-17T10:17:35.300310Z"
  },
  "agent": "brands/0c018bb7-67cb-4133-abbf-bd7f351f4ee0/agents/73a769da-46b0-4e87-82ab-
eac6da6ed540"
}
```

6.3.4.1.7. Request for a Live Agent

```
{
  "context": {
    "placeId": "ChIJh3I4p5E0wYkRP6iGd6j1RU8",
    "entryPoint": "MAPS",
    "userInfo": {
      "displayName": "User 1",
      "userDeviceLocale": "en-GB"
    },
    "resolvedLocale": "en"
  },
  "sendTime": "2021-05-17T10:15:05.875146Z",
  "conversationId": "278ba79a-bf6e-493e-82d2-910f3d08da79",
  "customAgentId": "gbm-bot",
  "requestId": "248a09e3-7ebe-4531-be57-af5b2f6cbe06",
  "userStatus": {
    "createTime": "2021-05-17T10:15:04.918700Z",
    "requestedLiveAgent": true,
    "statusType": "LIVE_AGENT"
  },
  "agent": "brands/0c018bb7-67cb-4133-abbf-bd7f351f4ee0/agents/73a769da-46b0-4e87-82ab-eac6da6ed540"
}
```