RCS Verification Authority API

Version 1.0

25 November 2021

This is a Non-binding Permanent Reference Document of the GSMA

Table of Contents

1 Introduction 4
1.1 Overview 4
1.2 Scope 4
1.3 Definitions 5
1.4 Abbreviations 5
1.5 References 6
1.6 Conventions 6
1.7 Assumptions 6
2 Object Models 8
2.1 Objects 9
2.1.1 Partner 9
2.1.2 Brand/Organization Entity 10
2.1.3 Chatbot 11
2.1.4 Documents 13
2.1.5 Chatbot verification signature 13
2.2 Events 13
2.2.1 Verification request 13
2.2.2 Autonomous notification 13
2.2.3 Verification signature (and associated documentation) retrieval 13
2.2.4 Verification Signature Certificate Retrieval 14
2.3 Actors 14
2.3.1 Verification Authority 14
2.3.2 RCS Service Provider MaaP or Chatbot lifecycle management function 14
2.3.3 Chatbot Provider platform (Partner and Brand when applicable) 14
2.4 Notifications 14
3 APIs 16
3.1 Partners 16
3.1.1 Resource: Partners 16
3.1.2 POST 16
3.1.3 GET 19
3.1.4 PATCH 23
3.1.5 DELETE 25
3.2 Brand APIs 27
3.2.1 Resource: Brands 27
3.2.2 POST /Brands 27
3.2.3 GET /Brands 29
3.2.4 PATCH /Brands/{id} 35
3.2.5 DELETE /Brands/{id} 37
3.3 Chatbots APIs 38
3.3.1 Resource: Chatbots 38
3.3.2 POST /chatbots 38
3.3.3 GET /chatbots 41
3.3.4 PATCH /chatbots/{id} 48

3.3.5 DELETE /chatbots/{id} 51
3.4 Notification API 52
3.4.1 Resource: Notification 52
3.4.2 PUT /notification 52
3.4.3 GET /notification 54
3.4.4 DELETE /notification 55
3.5 Autonomous Notifications 56
3.5.1 Resource: Notification 56
3.5.2 Push Interface 56
3.6 RCSVA Public Certificate API 57
3.6.1 Resource: Public Certificate 57
3.6.2 GET/Certificate URI 58
3.7 Utility APIs 59
3.7.1 Resources: Network Providers 59
3.7.2 GET / Network_Providers 60
4 Process Flows and Diagrams 62
4.1 Authentication 62
4.2 Notification Channel 63
4.3 Partner Verification via RCS SP 63
4.4 Brand Verification via RCS SP 65
4.5 Chatbot Verification via RCS SP 66
4.6 Partner Verification directly to VA 67
4.7 Brand Verification directly to VA 68
4.8 Chatbot Verification directly to VA 69
Annex A State Flow Diagram 70
Annex B Error Responses and Message Codes 71
Annex C Document Management 90
C.1 Document History 90
C.2 Other Information 90

1 Introduction

1.1 Overview

The RCS-based Rich Business Messaging (RBM) Verification Authority (VA) service provides independent, neutral, and secure processes for Company/Brand verification and digital signatures for verified business Chatbots. This VA API is intended to give stakeholders at every level—from Service Providers, to Aggregators, to Content Providers and Application Developers—the ability to enhance the trust in their communications, while at the same time streamlining back-end processes to enable industry-wide scale.

This Interface Specification PRD identifies the following information as relates to use of the RCS VA Interface Specification:

- Various actors in the RBM ecosystem interacting with the VA function
- The objects (and their data) involved in the interaction with the VA function
- The entity relationship and state diagrams for these objects
- API Methods indicating what operations the VA supports on these objects
- Business rules indicating when verification is triggered if object data changes
- Message flows for the use cases in the business Chatbot lifecycle

The information listed above is provided to enable automated interaction between the VA and Chatbot applicants (i.e., the Company/Brand through their Partner or a network provider’s management platform) to request verification and collect the Chatbot digital signature as well as interaction between the VA and the RBM Platform (a.k.a. MaaP) relying upon the signature.

1.2 Scope

This specification describes the REST API interface provided for an RBM VA platform to automate interaction between the VA and Chatbot applicants (i.e., the Company/Brand or their Partner) requesting verification and the Chatbot digital signature as well as between the VA and the RBM Platform (a.k.a. MaaP) relying upon said signature. The RBM Platform may use a system provided as an industry-facing front end.

1.3 Definitions

Term	Description
RCS Business Messaging Platform	The RCS Business Messaging Platform enables Brands and aggregators to connect to operator RCS services and exchange rich messages with users. The RCS Business Messaging Platform function and role in the verification process, is to add the signature that it receives from the Verification Authority to the Chatbot information, which is transferred to the MNO network.
Partner, or Messaging Partner	A Partner (or Messaging Partner) provides connectivity and aggregation services to Brands that want to use RCS Business Messaging to communicate with consumers. The Partner is responsible for collecting information about a Chatbot that needs to be verified from the Brand that they represent and present to the Verification Authority and / or MNO.
Brand	A business or entity that uses messaging to communicate with consumers, using a Chatbot, and thus wants to be verified as a business message sender
Chatbot	An RCS-based service provided to the users whose output is presented in a conversational form. Often a piece of software interfacing with one or more users aiming to simulate intelligent human conversation. A Chatbot is operated by a Brand, and therefore it is the responsibility of the Brand to provide the information necessary to verify a Chatbot.
Chatbot Profile Information	Information on the Chatbot provided by the RCS Service Provider to the consumer that allows the consumer to identify the Chatbot and the Brand operating the Chatbot, contact the Brand operating the Chatbot over other channels, or better understand what the purpose of the Chatbot is or what the Brand operating the Chatbot does.
RCS Service Provider (SP)	A company providing Rich Communications Services (RCS) to end customers. RCS Service Providers are typically Mobile Network Operators (MNOs). An RCS Service Provider grants the right to one or more Verification Authorities to verify Chatbots on their behalf. The RCS Service Provider also confirms whether the signature(s) provided by the Verification Authorities matches the Chatbot Information in the RCC specifications about verification signatures in the Chatbot info Function and Chatbot Directory and if it does, the RCS Service Provider provides an indication to the RCS client that the Chatbot has been verified.
Verification Authority	A provider of verification for Chatbots. A Verification Authority could be, but not limited to, a commercial trusted business (e.g. verification companies from the internet world), an MNO or a governmental function.
Verified Chatbot	A Chatbot that was verified to represent the identity (e.g. name, Brand or institution) that their name, Brand Icon and Chatbot Profile Information suggest.

1.4 Abbreviations

Term	Description
PRD	Permanent Reference Document

1.5 References

Ref	Doc Number	Title
[1]	[GSMA-RCS-Verified-Sender-Guideline]	RCS Verified Sender Product Feature Implementation Guideline, version 5, March 2019
[2]	[GSMA PRD-RCC.07]	Rich Communication Suite Advanced Communications Services and Client Specification Version 12.0, 16 October 2020, including GSG CR005 “iat” in the Chatbot Verification Signature
[3]	[RFC5789]	IETF RFC 5789: Patch Method For HTTP
[4]	[RFC6902]	IETF RFC 6902: JavaScript Object Notation (JSON) Patch
[5]	[RFC4122]	A Universally Unique IDentifier (UUID) URN Namespace
[6]	[ISO8601]	Data elements and interchange formats — Information interchange - Representation of dates and times
[7]	[RFC8174]	IETF RFC 8174: Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words

1.6 Conventions

This document does not duplicate object attribute definitions and other attribute information. Attributes are defined once at their first use and the reader is expected to refer backwards for its full information.

All sections and appendixes, except “Introduction”, “Process Flows and Diagrams”, and “State Flow Diagram” are normative, unless they are explicitly indicated to be informative. The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC8174] when, and only when, they appear in all capitals, as shown here.

HTTP is used throughout the document, the use of HTTPS is, of course, mandated.

1.7 Assumptions

- A Brand needs to provide all of the information necessary for the verification process. This information shall be shared with a Verification Authority that is responsible for conducting the verification either directly or through a Partner (or Messaging Partner).
- The VA shall be able to discern that a Chatbot previously verified for one RCS Service Provider, requires limited verification effort for a second or subsequent network. It is understood that the VA shall supply a different signature based on a differing ServiceID, but as long as this is the only difference in the submitted Chatbot Object properties, the verification effort shall be limited to managing the signature and its differing expiry time, and incrementally meeting any verification criteria/policy that the second or subsequent network may have specified.
- It is assumed that the NetworkProviderId representing the RCS Service Provider in the Chatbot object shall be unique for each VA and this namespace is identified when onboarding to a VA and further, that these identifiers shall be discoverable within the API.

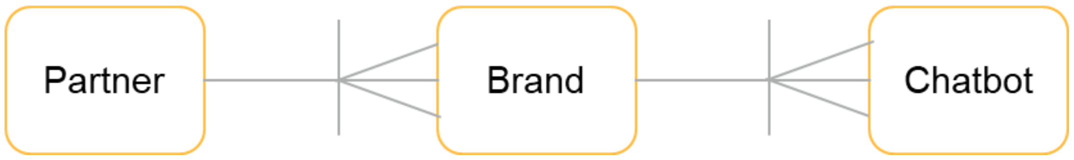
- For API efficiency, the HTTP PATCH method [RFC5789 and RFC6902] shall be used to modify specific attributes of an existing object rather than using the PUT method which requires all object attributes be provided.

2 Object Models

This API provides a solution for up to 3 objects that require verification, either in unison or independently, to prove that the Chatbot is authorized and legally allowed to communicate with a mobile user on behalf of the Brand that the Chatbot represents.

- Partner Object: Where the Partner is the legally responsible entity for conducting the messaging activities on behalf of the Brand.
- Brand Object: Captures information of the Brand, and the Brand owner.
- Chatbot Object: The ultimate object that is being Verified for the purposes of the mobile user consumption.

The Object model allows for each entity to only be verified once, but for the result of the verification process to be available to the MaaP platform RCS Service Provider.



In some cases, a Brand may register a Chatbot directly with the RCS Service Provider without a third party Partner. In this case, the Brand is also performing the role of the Partner. Therefore, the Brand shall first register as a Partner and get verified. Once it is verified as a Partner, it will submit the Brand details for verification. In the case a Brand may use multiple Partners for different Chatbots it shall be considered as a new object with distinct BrandId(s). In addition, an RCS Service Provider may register internal Chatbots with the VA without requiring verification.

Lists of registered objects can be retrieved using GET methods without specifying unique object IDs. The entire tree and parent-child relationships can be discovered by using successive GET methods while traversing the object hierarchy. For efficiency, optional filters may be added to the GET methods per object in future versions of this API. For example, a Partner shall retrieve all Chatbots associated with a given BrandId or an RCS Service Provider shall retrieve all Chatbots associated with a given PartnerId.

In order to provide security for submitted documents and details, a VA shall only respond to a GET method with the objects that are within the span of control of the requestor. For example, an RCS Service Provider can only retrieve objects and their attributes for the Chatbots registered on their network, and similarly, can retrieve only the Brands and Partners associated with those Chatbots. This shall also apply for the PATCH method.

An object shall only be deleted if all associations that depend on it have been deleted previously. For example, a Partner cannot be deleted if there is still a Brand or Brand and Chatbot associated with that Partner and a Brand cannot be deleted if there is still a Chatbot associated with that Brand.

The RCS Service Provider on boarding the Chatbot shall be identified by a NetworkProvider attribute in the Chatbot object. This attribute shall align with the Host Name of the Chatbot ServiceId attribute which refers to a connection point on a specific RCS Service Provider's

RBM platform. This may also be a connection point on an RBM platform that is hosted by a third party or hosted by a consortium of RCS Service Providers.

2.1 Objects

The following objects and their attributes are available for use by the API.

It is noted that the Object attributes shall be flexible, in terms of whether they are mandatory and if new attributes are needed, the attribute list shall be extensible for all Objects. It shall be necessary that the VA and RCS SP agrees the list of Object attributes that must be verified as part of the business process prior to offering the verification service.

Each object is uniquely referenced via an identifier; BrandId for the Brand object, PartnerId for the Partner object, ChatbotId for the Chatbot object.

2.1.1 Partner

All Partners shall be successfully verified before they can be associated with a Brand and their Chatbot(s). When a Partner is first POSTed, a unique PartnerId attribute shall be returned for future reference to that specific Partner object – such as for changing Partner info. The attributes to register and verify a Partner shall include:

- PartnerName - legal entity name
- Website – internet domain owned by the Partner company
- RefNumber – government reference number of the Partner company (e.g., EIN, etc.)
- RefNumberType – category identifying the nature of the RefNumber
- CountryOfIncorp – country where the entity is formally established as noted in ISO-3166-1 using Alpha-2 scheme
- StateOfIncorp – state (or province) where the entity is formally established (if applicable) as noted in ISO-3166-2
- RegNumber – additional company registration number for verification of entity details
- RegNumberType – the type of registration number provided for the Brand
- PrimaryBusinessDomain – primary business domain reference number
- PrimaryBusinessDomainType – category identifying primary business domain reference number (e.g. SIC, NIC, NACE, NAF)
- PartnerAddress - legal address (street, city, state/district, zip/postal code, country)
- MainBusinessTN – official business telephone number for the legal entity
- PartnerContactInfo – first&last name, email, title, business telephone number
- Verify – whether the VA should verify the Partner object or merely save it for future reference

Additional Partner object attributes shall be supported as extensions where required for local regulation or policy reasons. The Partner object shall include a GET /partners method to retrieve attributes describing all Partners registered in the VA platform and their state. Furthermore, this method shall provide a filter attribute to return verified or unverified Partners if a refined list is desired. This method shall return the following attributes:

- PartnerId – the unique identifier for a Partner object
- PartnerName – the legal entity name of the Partner company
- PartnerVerified – indicates whether verification was successfully completed
- PartnerStatus – indicates whether the Partner is active in the VA platform.

The VA shall provide further detail on a specific Partner object via a GET /partners/{id} method which requires a valid PartnerId be included. This shall return all the attributes in the POST /partners method above as well as the following:

- PartnerVerified – indicates whether verification was successfully completed.
- PartnerStatus – indicates whether the Partner is active in the VA platform.

The object shall support a PATCH /partners method to be used with a specific PartnerId when changing any of the previously POSTed Partner attributes noted above. The PATCH method is referenced in [3] and [4].

Lastly, there shall be a DELETE /partners/{id} method to be used with a specific PartnerId for removing a Partner from the VA platform.

2.1.2 Brand/Organization Entity

The VA Brand object shall support a POST /brands method which enables registration with the VA in order for verification of the Brand to occur. All Brands shall be successfully verified before any Chatbot(s) under the Brand can be verified, though these may be done in direct sequence. When a Brand is first POSTed, a VA shall return a unique BrandId attribute for future reference to that specific Brand object – such as for registering Chatbots or changing Brand info. The attributes to register and verify a Brand shall include:

- BrandName - legal entity name
- PartnerId – the unique identifier of the Partner who is submitting the Brand
- InformationWebsite – internet domain owned by the Brand. This services as a default value for website.
- DefaultIcon – logo to be used with Chatbots if no other logo is in the Chatbot object
- ServiceIconSN – the registration number for the registered service mark, if applicable
- SNJurisdiction – the country where the service mark is registered, if applicable
- ServiceIconOwner - Full name of trademark owner (if different than Brand name)
- RefNumberType – category identifying the nature of the RefNumber
- RefNumber – reference number of the Brand entity (e.g., EIN, etc.)
- CountryOfIncorp – country where the entity is formally established as noted in ISO-3166-1
- StateOfIncorp – state (or province) where the entity is formally established (if applicable) as noted in ISO-3166-2
- RegNumber – company registration number for verification of entity details
- RegNumberType – the type of registration number provided for the Brand
- PrimaryBusinessDomain – primary business domain reference number
- PrimaryBusinessDomainType – category identifying primary business domain reference number (e.g. SIC , NIC, NACE, NAF, UNSPSC)
- BrandAddress - legal address (street, city, state/district, zip/postal code, country)
- MainBusinessTN – official business telephone number for the legal entity
- Verify – whether the VA should verify the Brand object or merely save it for future reference

Additional Brand object attributes shall be supported as extensions where required for local regulation or policy reasons.

The VA shall also support a GET /brands method to retrieve an array of attributes describing all Brands registered in the VA platform and their state. For Partners, this shall be limited to the Brands that they have provisioned. For RCS Service Providers, this shall be limited to the Brands that they have provisioned or that are associated with their network through a

Chatbot. Furthermore, this method provides a filter attribute to return verified or unverified Brands if a refined list is desired. This method shall return the following attributes:

- BrandId – the unique identifier for a Brand object
- BrandName – the name of the Brand entity
- Verified – the aggregate status of the Brand entity and default logo verification
- BrandVerified – indicates whether verification of the entity was successfully completed
- IconVerified – indicates whether the default service icon, if provided, was successfully verified
- BrandStatus – indicates whether the Brand is active in the VA platform.

Data on a specific Brand object shall be retrieved with a GET /brands/{id} method which requires a valid BrandID be included. This shall return all the attributes in the POST /brands method above as well as the following:

- Verified – the aggregate status of the Brand entity and default logo verification
- BrandVerified – indicates whether verification of the entity was successfully completed
- IconVerified – indicates whether the default logo, if provided, was successfully verified
- BrandStatus – indicates whether the Brand is active in the VA platform

The VA shall support a PATCH /brands/{id} method to be used with a specific BrandId when changing any of the previously POSTed Brand attributes noted above. The PATCH method is referenced in [3] and [4].

Lastly, the VA shall support a DELETE /brands/{id} method to be used with a specific BrandId for removing a Brand from the VA platform.

2.1.3 Chatbot

The Chatbot object shall support a POST /chatbots method containing the attributes which enable verification of the Chatbot. When a Chatbot is first POSTed, a unique ChatbotID attribute shall be returned for future reference to that specific Chatbot object – such as for updating Chatbot info including the associated Partner, service icon, etc. The attributes to register and verify a Chatbot shall include:

- ServiceName – the common name used by the Chatbot in the Chatbot directory
- ServiceId – the internal URI where this Chatbot will connect to an RCS Service Provider
- PartnerId – the Partner entity approved to send content for this Chatbot
- Description – a textual description characterizing the purpose of the Chatbot
- SMSFallbackNo – the number used as the sender address (short, long or alphanumeric) when the Chatbot has to fallback to SMS
- Category – a list of categories of the Chatbot as referenced in [2]
- Website – the URL of the website for the products/services being represented by the Chatbot which should match the Chatbot InfoCard in the MaaP
- ServiceIcon – base64 encoding of the fingerprint of the Chatbot icon
- ServiceIconSN – the registration serial number for the registered service mark, if applicable
- SNJurisdiction – the country where the service mark is registered, if applicable
- ServiceIconOwner - Full name of trademark owner (if different than Brand name)
- CapacityProfile – indicates a particular traffic capacity level for the NetworkProvider
- ChatbotType – indicates the Chatbot is internal (RCS Service Provider supplied) or external
- BrandId – the unique ID assigned to the Brand who owns the Chatbot

- BrandContactInfo – first&last name, email, title, business telephone number of the person who will authorize a Partner to register Chatbots with the Brand name and logo
- NetworkProviderId – the unique ID of the RCS Service Provider this Chatbot will connect to
- Verify – indicates whether to proceed with Chatbot verification or simply to save the Chatbot object such as for internal Chatbots or a Chatbot to be verified at a later time

Additional Chatbot object attributes shall be supported as extensions where required for local regulation or policy reasons.

The Chatbot object shall support a GET /chatbots method to retrieve attributes describing all Chatbots, within the requestor's span of control, registered in the VA platform and their state. For Partners, this shall be limited to the Chatbots that they have provisioned. For RCS Service Providers, this shall be limited to the Chatbots that they have provisioned or that are associated with their NetworkProvider attribute in each Chatbot. Furthermore, this method provides a filter attribute to return verified or unverified Chatbots if a refined list is desired. The attributes returned shall include:

- ChatbotId – the unique identifier for a Chatbot object
- ServiceName - the common name used by the Chatbot in the Chatbot directory
- BrandId – the unique identifier for a Brand object
- PartnerId – the Partner entity approved to send content for this Chatbot
- PartnerName – the legal entity name of the Partner company
- Status – indicates whether the Chatbot is active in the VA platform
- Verified – the aggregate status of the Chatbot 2FA and service icon
- 2FACompleted – indicates if the Brand completed the 2FA confirming Chatbot ownership derived the BrandContactInfo provided in the object. IconVerified – indicates whether the logo, if provided, was successfully verified

The VA shall provider further detail on specific Chatbot objects via a GET /chatbots/{id} method which requires a valid ChatbotID be included. This shall return all the attributes in the POST /chatbots method above as well as the following:

- Status – indicates whether the Chatbot is active in the VA platform
- Verified – the aggregate status of the Chatbot 2FA and service iconverification.
- 2FACompleted – indicates if the Brand completed the 2FA confirming Chatbot ownership
- IconVerified – indicates whether the logo, if provided, was successfully verified
- In order to retrieve the Chatbot signature the Chatbot object shall support a GET /chatbots/id/documents method which shall return:
- JWT – the Chatbot signature token as specified in the GSMA RCC.07 RCS standard [2]
- Additional documents maybe added in future versions of the API or as proprietary extensions for a given VA.

The Chatbot object shall also include a PATCH /chatbots/{id} method to be used with a specific ChatbotId when changing any of the previously POSTed Chatbot attributes noted above except for the ChatbotType attribute (i.e., internal, external) which cannot be changed. Such a change shall require a new Chatbot be registered. [reword to bullet all attributes that cannot be patched]. The PATCH method is referenced in [3] and [4].

Lastly, there shall be a DELETE /chatbots/{id} method to be used with a specific ChatbotId for removing a Chatbot from the VA platform.

This is a description of States of Chatbot verification:

- STATE = NOT-STARTED: an object will have this state until verification is requested
- STATE = PENDING: once verification is triggered, the state is Pending until complete or Failed.
- STATE = COMPLETE: completed state is only changed if the verification approaches expiry
- STATE = FAILED: a failed verification may trigger another attempt, or be over-ridden
- STATE = EXPIRED: the verification has expired and requires re-submission
- STATE = OVERRIDDEN: the RCS Service Provider shall be able to bypass verification or override a failed verification. Such Chatbots shall also support an expiry time.

For an illustrative diagram of the States see Annex A.

2.1.4 Documents

Each Object shall support the addition of sub-documents, submitted for object registration and verification. These documents are base 64 encoded for transmission to the VA. The maximum size of attached documents shall not exceed 10MB. The documents shall exist for correlation between the MaaP Platform, the Partner and the VA to ensure submitted legal documents are accurate between the parties.

Documents will vary by market or local RCS Service Provider policy therefore each document type shall be supported as proprietary extensions by each VA based on the customers and market they serve.

2.1.5 Chatbot verification signature

The successful result of the Verification exercise shall be the availability of the Chatbot Signature as per [1] and [2] section 3.6.3.2.1.

2.2 Events

The following subsections describe the key events that shall be supported by the VA. Additional events may be supported via proprietary extensions or in future versions of this PRD.

2.2.1 Verification request

Verification of objects (Partner, Brand, and/or Chatbot) shall be requested by POSTing (or PATCHing) an object with the Verify attribute set accordingly.

2.2.2 Autonomous notification

The PUT/notification API shall initiate a notification message to the CallbackURI to notify the VA user of a CRL (i.e., certificate revocation) change, a new or revised JWT verification signature, or an update for any of the Partner, Brand or Chatbot objects managed by the VA.

When Partners and RCS Service providers do not register a CallbackURI via this API method, the VA shall send notifications by an alternate mechanism that is mutually agreed.

2.2.3 Verification signature (and associated documentation) retrieval

The GET /chatbots/{id}/documents API shall initiate a request to retrieve Chatbot verification documents provided by the VA platform. This API shall be used for periodic polling after Chatbot verification is requested or shall be used after receipt of an autonomous notification that the Chatbot verification is complete for a given ChatbotId.

At present, JWT is the only query parameter required in the API. It shall be used to retrieve the Chatbot verification signature. If the query parameter is omitted then all documents will be returned in the response body, including any documents handled via proprietary extensions. On success, the response will be a 200 OK.

For security reasons, the JWT object containing the Chatbot verification signature shall only be retrieved via a GET method.

2.2.4 Verification Signature Certificate Retrieval

In order to add increased security and scalability for Chatbot verification signatures, in addition to using the “kid” in the digital signature to decode the payload, the VA shall support a GET /certificate API to retrieve the public certificate based on a JWS x5u (i.e., X.509 URI) optional header parameter in the digital signature. The RCSVA public key within that certificate shall be used to decode the JWS digital signature and validate the certificate path terminates at a trusted root CA. This API shall return the public certificate in the form of a Base64 PEM encoded string.

In addition to this API GET method, the VA public certificate may be exchanged out of band, outside the API. In this case, a means to keep up to date shall be implemented as certificates are renewed or revoked.

Caching of the Public Certificate shall be supported as long as the Chatbot signature has not expired and the Certificate Revocation List has been checked. Refer to the Notifications section below for subscribing to CRL updates.

2.3 Actors

2.3.1 Verification Authority

The VA shall support the APIs to communicate with other actors in the RCS Business Messaging ecosystem order to increase the security and scalability of Chatbot verification signatures throughout the Chatbot lifecycle.

2.3.2 RCS Service Provider MaaP or Chatbot lifecycle management function

The APIs shall provide the RCS Service Provider with the ability to fully manage the lifecycle of a Chatbot from initial on boarding, through verification, to launch and into production. An example state diagram is shown in Annex A for illustraive purposes depicting the full lifecycle that maybe supported.

2.3.3 Chatbot Provider platform (Partner and Brand when applicable)

A Chatbot Provider (Partner or Aggregator) shall use these APIs to request verification services for Brands and Chatbots from VAs that are selected by the RCS Service Providers. The Chatbot provider shall then present the signature to the RCS Service Provider as part of completing the onboarding process for each Chatbot.

2.4 Notifications

The Notification API shall be supported as an option for any system interfacing to the VA platform with this suite of APIs. A Callback URI is supported as an attribute to identity where the VA shall send notifications. Partners and RCS Service Providers can both choose to register individual callbacks with this API. This URI only needs to be posted once for each 3rd party system to subscribe to notifications.

When a Partner registers for the callback, it shall receive notifications for its own Partner Entity, and for all Brands and Chatbots that it has submitted to the VA.

When a RCS Service Provider registers for the callback, it shall receive notifications for all Partners, Brands and Chatbots entities that it has submitted to VA, and for all Chatbots that have been submitted by any Partner for that RCS Service Provider, as well as all the Brands to which these Chatbots belong, and the Partners submitting these Chatbots.

The VA shall support a PUT /notification method for entities to subscribe to available VA notifications. This method shall support the AuthURI address, the CallbackURI address, as well as an array-based attribute which identifies the desired events to trigger notification:

- CRL – registers for notifications regarding changes to the Certificate Revocation List
- JWT – registers for notifications regarding changes to any Chatbot JWTs
- Partner – registers for notifications regarding changes to Partner object states
- Brand – registers for notifications regarding changes to Brand object states
- Chatbot – registers for notifications regarding changes to Chatbot object states

The VA shall respond to a GET /notification method with the info provided in the PUT method above. This shall identify the VA notifications which the requesting entity has subscribed to. The VA shall cease notifications to that entity upon receipt of a DELETE /notification.

When an AuthURI and CallbackURI are registered via the PUT method above, the VA shall POST the requested information to the CallbackURI using the attributes above. Items POSTed shall include the following information:

- CRL – URI where the updated CRL can be retrieved and the timestamp when it was issued
- JWT – ChatbotId for the updated JWT and the timestamp when it was issued
- Partner – the PartnerId, and the reason (Create, Modify or Delete)
- Brand – the BrandId, and the reason (Create, Modify or Delete)
- Chatbot – the ChatbotId, and the reason (Create, Modify or Delete).

A Partner submitting a Create, Modify or Delete request to the VA shall get back notification for these requests if previously subscribed to the notifications API. Similary, an RCS Service Provider submitting a Create, Modify or Delete request to the VA shall get back notification for these requests if previously subscribed to the notifications API. An RCS Service Provider shall receive notifications for Create, Modify and Delete requests initiated by any Partners, or Brands serving the Partner role, for all Chatbots registered for the RCS Service Provider's Network. Furthermore, notifications with reason Modify shall be sent to the Partner and RCS Service Provider for all changes in the VA status of a Partner, Brand or Chatbot.

3 APIs

3.1 Partners

These APIs shall be used to register and maintain the Partner object with the VA.

3.1.1 Resource: Partners

Request

The resource used is

```
https://{ServerRoot}/rcsva/{apiVersion}/partners/{PartnerId}{filter}
```

Name	Type	Description
ServerRoot	Path	The Server Hosting the API
ServiceRoot	Path	The Root specifically for the Service is RCSVA
apiVersion	Path	The version of API being used
apiMethod	Path	The APi method being used, such as Partners

Example (Informative)

Sample Partners resource address:

```
https://intel-rcsapi.iconectiv.com/rcsva/v1/partners
```

3.1.2 POST

Request

The API shall initiate a request to store registration data associated with a Partner in the VA platform.   On success, the response shall contain a unique identifier for the Partner (i.e., PartnerId) that shall be used on subsequent API calls to retrieve and update partner data. The PartnerId object is later used to explicitly associate a Brand or a Chatbot with said Partner.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response using OAuth2 as an example.

Request Example (Informative)

```
curl -X POST\
-H "Accept: application/json"\
-H "Content-Type: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/partners"
```

Body Schema

Attribute	Type	Required v Optional	Maximum Size
PartnerInfo	object	R	0
PartnerName	string	R	80
Website	string	R	128
RefNumberType	string	O	20
RefNumber	string	O	20
CountryOfIncorp	string	R	2
StateOfIncorp	string	O	3
RegNumber	string	R	20
RegNumberType	String	R	20
PrimaryBusinessDomain	string	O	10
PrimaryBusinessDomainType	string	O	20
PartnerAddress	Object	R	0
StreetAddress1	string	R	80
StreetAddress2	string	O	80
City	string	R	50
GoverningDistrict	string	O	50
PostalCode	string	R	10
Country	string	R	2
MainBusinessTN	string	R	15
PartnerContactInfo	object	R	0
FirstName	string	R	32
LastName	string	R	32
EmailAddress	string	R	254
Title	string	R	128
TelephoneNumber	string	R	15
Verify	string	O	20

Example (Informative)

```
{
    "PartnerInfo": {
        "PartnerName": "Partner100",
        "Website": "www.partner100.com",
        "RefNumberType": "FEIN",
        "RefNumber": "876543210",
        "CountryOfIncorp": "US",
        "StateOfIncorp": "NJ",
        "RegNumber": "54932938ICRETNJ5VZ41",
        “RegNumberType”: “GLEIF”
    },
    "PrimaryBusinessDomain": "1000",
    "PrimaryBusinessDomainType": "SIC",
    "PartnerAddress": {
        "StreetAddress1": "100 Corporate Blvd",
        "StreetAddress2": Suite 800",
        "City": "Anytown",
        "GoverningDistrict": "NJ",
        "PostalCode": "08807",
        "Country": "US"
    },
    "MainBusinessTN": "7325551212",
    "PartnerContactInfo": {
        "FirstName": "Robert",
        "LastName": "Smith",
        "EmailAddress": "Bob.Smith@Partner100.com",
        "Title": "Director",
        "TelephoneNumber": "7325552121"
    },
    "Verify": "not-started"
}
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

Attribute	Type
PartnerId	string

Example (Informative)

```
{"PartnerId":"9942eaf3-4d67-46e6-89a7-6ca311513dd4"}
```

3.1.3 GET

The API shall support a combination of PartnerId and Filter parameters.

Example 1: No Parameters or Filters

Request

This API initiates a query request to retrieve registration data associated with all Partners that are contained in the VA platform, and which are associated with the RCS entity making the request. On success, the response shall contain an array of Partner data. The Id value returned for each Partner shall be used in GET/partners/{Id} requests to retrieve data associated with the specific Partner.

Note that the value for the "Authorization" header tag is the `access_token` returned in the /auth response.

Filters

```
Object
Not Applicable
```

Body Schema

```
Object
Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/partners"
```

Response

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

Attribute	Type	Maximum Length	Comments
Partners	object		
PartnerId	string		
PartnerName	string		

Attribute	Type	Maximum Length	Comments
PartnerVerified	string	15	not-started, pending, complete, failed
PartnerStatus	string	15	pending, active, suspended, inactive
UpdateDateTime	string	25	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.

Example (Informative)

```
{"Partners":[{"PartnerId":"bbe6015c-c1f7-11ea-a905-
6e374e423552","PartnerName":"Partner Company Name
1","PartnerVerified":"not-
started","PartnerStatus":"active","UpdateDateTime":"2020-09-30T21:16:28-
04:00"}]}
```

Example 2: With Filters

Request

This API initiates a query request to retrieve registration data associated with select Partners that are contained in the VA platform, and which are associated with the RCS entity making the request. The request shall accept a query parameter to filter Partners based on whether they have been verified or not. Also note that multiple filters may be supported. On success, the response shall contain an array of Partner data. The Id value returned for each Partner shall be used in GET/partners/{Id} requests to retrieve data associated with the specific Partner.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Filter Parameters

```
verified               complete, not-started, pending, failed
```

Body Schema

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
"https://intel-
rcsapi.iconectiv.com/rcsva/v1/partners?verified=complete&verified=pending"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

Attribute	Type	Comments
Partners	object	
PartnerId	string	
PartnerName	string	
PartnerVerified	string	not-started, pending, complete, failed
PartnerStatus	string	pending, active, suspended, inactive
UpdateDateTime	string	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.

Example (Informative)

```
{"Partners":[{"PartnerId":"bbe6015c-c1f7-11ea-a905-
6e374e423552","PartnerName":"Partner Company Name
1","PartnerVerified":"verified","PartnerStatus":"active","UpdateDateTime":"
2020-12-30T21:16:28Z"}]}
```

Example 3: With PartnerId Parameter

Request

This API initiates a query request to retrieve registration data associated with a Partner identified by {Id} in the VA platform. On success, the response shall contain the data associated with the Partner.   The Id value used on the URI of the GET request shall be found by sending a Partners list GET request to retrieve a collection of PartnerIds.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Query Parameters

Id	Id of the Partner

Body Schema

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X GET\

-H "Accept: application/json"\

"https://intel-rcsapi.iconectiv.com/rcsva/v1/partners/bbe6015c-c1f7-11ea-
a905-6e374e423552"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

Attribute	Type	Comments
ParnerInfo	object	
PartnerName	string	
Website	string	
RefNumberType	string	
RefNumber	string	
CountryOfIncorp	string	
StateOfIncorp	string	
RegNumber	string	
RegNumberType	string	
PrimaryBusinessDomain	string	
PrimaryBusinessDomainType	string	
PartnerAddress	object	
StreetAddress1	string	
StreetAddress2	string	
City	string	

Attribute	Type	Comments
GoverningDistrict	string	
PostalCode	string	
Country	string	
MainBusinessTN	string	
PartnerContactInfo	object	
FirstName	string	
LastName	string	
EmailAddress	string	
Title	string	
TelephoneNumber	string	
PartnerVerifed	string	complete, not-started, pending, failed
PartnerStatus	string	pending, active, suspended, inactive
UpdateDateTime	string	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.

Example (Informative)

```
{"PartnerInfo":{"PartnerName":"Generic
Partner","Website":"www.uscompany.com","RefNumberType":"FEIN","RefNumber":"
999999999","CountryOfIncorp":"US","StateOfIncorp":"NJ","RegNumber":"5493293
8ICRETNJ5VZ41",”RegNumberType”:”GLEIF”},"PrimaryBusinessDomain":"5600","Pri
maryBusinessDomainType":"SIC","PartnerAddress":{"StreetAddress1":"100
Corporate Blvd","StreetAddress2":"Building
200","City":"Bridgewater","GoverningDistrict":"NJ","PostalCode":"08807","Co
untry":"US"},"MainBusinessTN":"1-
7326999999","PartnerContactInfo":{"FirstName":"John","LastName":"Smith","Em
ailAddress":"jsmith@genericpartner.com","Title":"QA","TelephoneNumber":"173
26999998"},"PartnerVerifed":"verified","PartnerStatus":"active","UpdateDate
Time":"2020-09-30T21:16:28-04:00"}
```

3.1.4 PATCH

Request

This API initiates a request to update registration data associated with a Partner in the VA platform. Only those attributes being updated need to be included in the request body.

Note that in order to remove/delete a previously provided value for a field (through a POST or PATCH), the field value shall be set to JSON null. The use of “” or blanks/spaces between “” is invalid and will return a syntax error. Additionally, fields which are not supported by a message or are not applicable for a country will be ignored, and no error response code will be triggered).

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Call Example (Informative)

```
curl -X PATCH\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/partners/9942eaf3-4d67-46e6-
89a7-6ca311513dd4"
```

Path Parameters

Id	Id of the Partner

Body Schema

Attribute	Type	Required v Optional
PartnerInfo	object	O
PartnerName	string	O
Website	string	O
RefNumberType	string	O
RefNumber	string	O
CountryOfIncorp	string	O
StateOfIncorp	string	O
RegNumber	String	O
RegNumberType	String	O
PrimaryBusinessDomain	string	O
PrimaryBusinessDomainType	string	O
PartnerAddress	Object	O
StreetAddress1	string	O
StreetAddress2	string	O
City	string	O
GoverningDistrict	string	O
PostalCode	string	O
Country	string	O
MainBusinessTN	string	O
PartnerContactInfo	object	O
FirstName	string	O
LastName	string	O
EmailAddress	string	O
Title	string	O
TelephoneNumber	string	O

Attribute	Type	Required v Optional
`Verify`	`String`	`O`

Example (Informative)

```
{"PartnerInfo":{"PartnerName":"Generic
Partner","Website":"www.uscompany.com","RefNumberType":"FEIN","RefNumber":"
999999999","CountryOfIncorp":"US","StateOfIncorp":"NJ","RegNumber":"5493293
8ICRETNJ5VZ41”,”RegNumberType”:”GLEIF”}
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Attribute	Type
PartnerId	string

Example (Informative)

```
{"PartnerId":"9942eaf3-4d67-46e6-89a7-6ca311513dd4"}
```

3.1.5 DELETE

This method shall support removal of a Partner from the VA platform once all associated Brands and Chatbots have been deleted.

Request

The API initiates a request to delete a partner object from the VA platform for the partner identified by {Id}. On success, the response will be a 200 OK. Note that a partner cannot be deleted from the system if it is associated with an active chatbot. Also, only the account that created the partner object can delete it.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Path Parameters

Id	Id of the Partner

Body Schema

```
Object
   Not Applicable
```

Example (Instructive)

```
curl -X DELETE\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/partners/ d14244a0-3dd0-42ca-
bb94-7342037a99ff"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

```
Object
PartnerId                     string
```

Example (Instructive)

```
{"PartnerId":"d14244a0-3dd0-42ca-bb94-7342037a99ff"}
```

3.2 Brand APIs

3.2.1 Resource: Brands

Request

The resource used is

```
https://{ServerRoot}/rcsva/{apiVersion}/brands/{BrandId}{filter}
```

Name	Type	Description
ServerRoot	Path	The Server Hosting the API
ServiceRoot	Path	The Root specifically for the Service is RCSVA
apiVersion	Path	The version of API being used
apiMethod	Path	The API method being used, such as Brands

Example (Informative)

Sample Brands resource address:

```
https://intel-rcsapi.iconectiv.com/rcsva/v1/brands
```

3.2.2 POST /Brands

Request

These APIs are used to register and maintain a Brand with the VA platform. The API refers to the unique Brand identifier as the BrandId.

Request Example (Informative)

```
url -X POST\
-H "Accept: application/json"\
-H "Content-Type: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/brands"
```

Body Schema

Attribute	Type	Required v Optional	Maximum Size	Comments
BrandInfo	object	R	0	
BrandName	string	R	80	
Website	string	R	128	
DefaultIcon	String	O		Base64 Encoded

Attribute	Type	Required v Optional	Maximum Size	Comments
ServiceIconSN	string	O	8	If there is a default icon then all Serial Number related fields shall be populated with a default of “Null” if not a registered mark.
SNJurisdiction	string	O	2	If there is a default icon then all Serial Number related fields shall be populated with a default of “Null” if not a registered mark.
ServiceIconOwner	string	O	128	
RefNumberType	string	O	20	
RefNumber	string	O	20	
CountryOfIncorp	string	R	2	
StateOfIncorp	string	O	3	
RegNumber	string	R	20	
RegNumberType	String	R	20	
PrimaryBusinessDomain	string	O	10	
PrimaryBusinessDomainType	string	O	20	
BrandAddress	Object	R	0	
StreetAddress1	string	R	80	
StreetAddress2	string	O	80	
City	string	R	50	
GoverningDistrict	string	O	50	
PostalCode	string	R	10	
Country	string	R	2	
MainBusinessTN	string	R	15	
Verify	string	O	11	

Example (Informative)

```
{
    "BrandInfo": {
      "BrandName": "ABC",
      "Website": "www.ABC.com",
      "DefaultIcon": "[insert icon encoding base64]",
      "ServiceIconSN": "123456578",
      "SNJurisdiction": "US",
      "ServiceIconOwner": "ABC"
      "RefNumberType": "FEIN",
      "RefNumber": "123456789",
      "CountryOfIncorp": "US",
      "StateOfIncorp": "WA",
      "RegNumber": "54932938ICRETNJ5VZ41",
      "RegNumberType": "GLEIF"
    },
    "PrimaryBusinessDomain": "1000",
    "PrimaryBusinessDomainType": "SIC",
    "BrandAddress": {
      "StreetAddress1": "1313 Mockingbird Lane",
      "StreetAddress2": "XXXSuite 100",
      "City": "Anytown",
      "GoverningDistrict": "DC",
      "PostalCode": "10011",
      "Country": "US"
    },
    "Verify": "complete"
}
```

Responses

Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

Attribute	Type
BrandId	string

Example (Informative)

```
{"BrandId":"7f99985f-74ee-4c86-bfff-b15adb1811cb"}
```

3.2.3 GET /Brands

Example 1: No Filters or Parameters

Request

This API shall initiate a query request to retrieve registration data associated with all Brands that are contained in the VA platform, and which are associated with the RCS entity making the request. On success, the response will contain an array of Brand data. The BrandId value returned for each Brand can be used in GET/brands/{Id} requests to retrieve data associated with the specific Brand.

Filters

```
Object
Not Applicable
```

Body Schema

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X POST\
-H "Accept: application/json"\
-H "Content-Type: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/brands"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

Attribute	Type	Max	Comments
Brands	object		
BrandId	string		
BrandName	string		
Verified	string	15	Aggregate verification status - complete, not-started, pending, failed
BrandVerified	string	15	complete, not-started, pending, failed
IconVerified	string	15	complete, not-started, pending, failed
BrandStatus	string	15	pending, active, suspended, inactive
UpdateDateTime	string	25	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.

Example (Informative)

```
{
            "BrandId": "5de23b55-992f-4557-b212-79d58292efab",
            "BrandName": "AMERICAN EXPRESS COMPANY",
            "Verified": "failed",
            "BrandVerified": "failed",
            "IconVerified": "not-started",
            "BrandStatus": "active",
            "UpdateDateTime": "2021-05-05T19:32:17+05:30"
        }
```

Example 2: With Filters

Request

This API shall initiate a query request to retrieve registration data associated with select Brands that are contained in the VA platform, and which are associated with the RCS entity making the request. The request accepts an optional query parameter to filter Brands based on whether they have been verified or not. Also note that multiple filters may be supported. On success, the response will contain an array of Brand data.   The BrandId value returned for each Brand can be used in GET/brands/{Id} requests to retrieve data associated with the specific Brand.

Filter Parameters

verified	complete, not-started, pending, failed

Body Schema

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
"https://intel-
rcsapi.iconectiv.com/rcsva/v1/brands?verified=complete&verified=pending "
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Attribute	Type	Comments
Brands	object	
BrandId	string	
BrandName	string	
Verified	string	Aggregate verification status – complete, not-started, pending, failed
BrandVerified	string	complete, not-started, pending, failed
IconVerified	string	complete, not-started, pending, failed
BrandStatus	string	pending, active, suspended, inactive
UpdateDateTime	string	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.

Example (Informative)

```
{
            "BrandId": "5de23b55-992f-4557-b212-79d58292efab",
            "BrandName": "AMERICAN EXPRESS COMPANY",
            "Verified": "pending",
            "BrandVerified": "pending",
            "IconVerified": "pending",
            "BrandStatus": "pending",
            "UpdateDateTime": "2021-05-05T19:32:17+05:30"
        }
```

Example 3: With BrandId Parameter

Request

The API initiates a query request to retrieve registration data associated with a brand identified by {Id} in the VA platform. On success, the response will contain the data associated with the brand.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Query Parameters

Id	Id of the Brand

Body Schema

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/brands/7f99985f-74ee-4c86-
bfff-b15adb1811cb"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

Attribute	Type	Comments
BrandInfo	object	
BrandName	string	
Website	string	
DefaultIcon	string	
ServiceIconSN	string	
SNJurisdiction	string	
ServiceIconOwner	string	
RefNumberType	string	
RefNumber	string	
CountryOfIncorp	string	
StateOfIncorp	string	
RegNumber	string	
RegNumberType	string	

Attribute	Type	Comments
PrimaryBusinessDomain	string	
PrimaryBusinessDomainType	string	
BrandAddress	Object	
StreetAddress1	string	
StreetAddress2	string	
City	string	
GoverningDistrict	string	
PostalCode	string	
Country	string	
MainBusinessTN	string	
VerificationInfo	object	
Verified	string	complete, not-started, pending, failed
BrandVerified	string	complete, not-started, pending, failed
IconVerified	string	complete, not-started, pending, failed
BrandStatus	string	pending, active, suspended, inactive
UpdateDateTime	string	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.

Example (Informative)

```
{"BrandInfo":{"BrandName":"Generic
Brand","Website":"www.genericbrand.com","DefaultIcon":"base64 encoded
string goes
here","ServiceIconSN":"85827444","SNJurisdiction":"US","ServiceIconOwner":"
Generic Brand’s Trademarked Logo Owner
Name","RefNumberType":"FEIN","RefNumber":"999999999","CountryOfIncorp":"US"
,"StateOfIncorp":"NJ","RegNumber":"54932938ICRETNJ5VZ41",”RegNumberType”:”G
LEIF”},"PrimaryBusinessDomain":"5600","PrimaryBusinessDomainType":"SIC","Br
andAddress":{"StreetAddress1":"59th Street and Lexington Avenue, 1000 Third
Avenue ","StreetAddress2":"Building A ","City":"New
York","GoverningDistrict":"NY","PostalCode":"10022","Country":"US"},"MainBu
sinessTN":"1-
2127059999”,"VerificationInfo":{"Verified":"verified","BrandVerified":"veri
fied","IconVerified":"not-
started"},"BrandStatus":"active","UpdateDateTime":"2020-11-15T21:16:28Z"}
```

3.2.4 PATCH /Brands/{id}

Request

This API initiates a request to update registration data associated with a Brand in the VA platform. Only those attributes being updated need to be included in the request body.

Note that in order to remove/delete a previously provided value for a field (through a POST or PATCH), the field value shall be set to JSON null. The use of “” or blanks/spaces between “” is invalid and will return a syntax error. Additionally, fields which are not supported by a message or are not applicable for a country will be ignored, and no error response code will be triggered).

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Call Example (Informative)

```
curl -X PATCH\
-H "Accept: application/json"\
-H "Content-Type: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/brands/bbe6015c-c1f7-11ea-
a905-6e374e423122"
```

Path Parameters

Id	Id of the Brand

Body

Attribute	Type	Required v Optional
BrandInfo	object	O
BrandName	string	O
Website	string	O
DefaultIcon	String	O
ServiceIconSN	string	O
SNJurisdiction	string	O
ServiceIconOwner	string	O
CountryOfIncorp	string	O
StateOfIncorp	string	O
PrimaryBusinessDomain	string	O
PrimaryBusinessDomainType	string	O
BrandAddress	Object	O

Attribute	Type	Required v Optional
StreetAddress1	string	O
StreetAddress2	string	O
City	string	O
GoverningDistrict	string	O
PostalCode	string	O
Country	string	O
MainBusinessTN	string	O
Verify	string	O

Example (Informative)

```
{
    "BrandInfo": {
      "BrandName": "ABC",
      "Website": "www.ABC.com",
      "DefaultIcon": "[insert icon encoding base64]",
      "ServiceIconSN": "123456578",
      "SNJurisdiction": "US",
      "ServiceIconOwner": "ABC"
      "RefNumberType": "FEIN",
      "RefNumber": "123456789",
      "CountryOfIncorp": "US",
      "StateOfIncorp": "WA",
      "RegNumber": "54932938ICRETNJ5VZ41",
      "RegNumberType": "GLEIF"
    },
    "Verify": "complete"
}
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Attribute	Type
BrandId	string

Example (Informative)

```
{"BrandId":"bbe6015c-c1f7-11ea-a905-6e374e423122"}
```

3.2.5 DELETE /Brands/{id}

This method shall support removal of a Brand from the VA platform once all associated Chatbots have been deleted.

Request

The API initiates a request to delete a brand object from the VA platform for the brand identified by {id}. Note that a brand cannot be deleted from the system if it is associated with an active chatbot. Also, only the account that created the brand object can delete it.

Note that the value for the "Authorization" header tag is the `access_token` returned in the /auth response.

Path Parameters

Id	Id of the Brand

Body Schema

```
Object

  Not Applicable
```

Example (Instructive)

```
curl -X DELETE\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/brands/ c9c80609-a50c-4219-
bb1e-b4763163bb31"
```

Responses

Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

```
Object
    BrandId                       string
```

Example (Instructive)

```
{"BrandId":"c9c80609-a50c-4219-bb1e-b4763163bb31"}
```

3.3 Chatbots APIs

These APIs are used to register and maintain a Chatbot with the VA platform.

3.3.1 Resource: Chatbots

Request

The resource used is

```
https://{ServerRoot}/rcsva/{apiVersion}/chatbots/{ChatbotId}{filter}
```

Name	Type	Description
ServerRoot	Path	The Server Hosting the API
ServiceRoot	Path	The Root specifically for the Service is RCSVA
apiVersion	Path	The version of the API being used
apiMethod	Path	The API method being used, such as Chatbots

Example (Informative)

Sample Chatbots resource address:

```
https://intel-rcsapi.iconectiv.com/rcsva/v1/chatbots
```

3.3.2 POST /chatbots

Request

This API shall initiate a request to store Chatbot data associated with a Brand in the VA platform. The value of BrandId found in the ChatbotBrandInfo object shall match an existing BrandId associated with a Brand that was previously registered. On success, the response shall contain a unique ChatbotId identifier that shall be used on subsequent API calls to retrieve and update Chatbot data.

The list of valid RCS Service Providers that shall be used to populate the NetworkProviderId attribute shall be obtained by using the GET /util/network_provider's API.

Note that the value for the "Authorization" header tag is the access_token returned in the /auth response.

Request Example (Informative)

```
curl -X POST\
-H "Accept: application/json"\
-H "Content-Type: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/chatbots"
```

Body Schema

Attribute	Type	Required v Optional	Maximum Size	Comments
ChatbotInfo	object	R	0	
ServiceName	string	R	100	
ServiceId	string	R	128	
PartnerId	string	O	128	
Website	string	R	128	
Description	string	O	500	
SMSFallbackNo	string	O	15	
Category	array	O	Up to 15 strings, each with maximum length of 50 characters	
ServiceIcon	string	O	Base64 encoded	
ServiceIconSN	string	O	8	If there is an icon then all Serial Number related fields shall be populated with a default of "Null" if not a registered mark.
SNJurisdiction	string	O	2	If there is an icon then all Serial Number related fields shall be populated with a default of "Null" if not a registered mark.
ServiceIconOwner	string	O	83	
Capacity Profile	String	O	128	
ChatbotType	string	O	8	
BrandId	string	R	Uuid	Per reference [5]
BrandContactInfo	object	O	0	

Attribute	Type	Required v Optional	Maximum Size	Comments
FirstName	string	O	32	
LastName	string	O	32	
EmailAddress	string	O	254	
Title	string	O	128	
TelephoneNumber	string	O	15	
NetworkProviderId	string	R	Uuid	
Verify	string	O	11	

Example (Informative)

```
{
    "ChatbotInfo": {
        "ServiceName": "TestChatbot",
        "ServiceId": "https://www.chatbot.com",
        "PartnerId": "ddc49f77-56a9-430b-ac99-e5e4c6c42ef9",
        "Website": "www.productabc.com",
  "Description": "Test Chat",
        "SMSFallbackNo": "2025553333",
        "Category": ["testQAcategory",” Category2”,”Category3”],
        "ServiceIcon": "{{serviceIcon base64}}",
        "ServiceIconSN": "12345678",
        "SNJurisdiction": "US",
        "ServiceIconOwner": "ABC",
        "CapacityProfile": "TestCapacity",
        "ChatbotType": "external"
    },
    "BrandId": "311b779e-381b-4cd8-b21d-1ce42e2d7c26",
    “BrandContactInfo”: {
      “FirstName”: ”John”,
      “LastName”: ”Doe”,
      “EmailAddress”: “john.doe@anycompany.com”,
      “Title”: ”Brand Manager”,
      “TelephoneNumber”: ”12124449876”
    },
    "NetworkProviderId": "df15cef4-a9ac-4adc-a5e0-4f5b7b9c30fd",
    “Verify”: “complete”
}
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

Attribute	Type
ChatbotId	string

Example (Informative)

```
{"ChatbotId":"9942eaf3-4d67-46e6-89a7-6ca311513ee5"}
```

3.3.3 GET /chatbots

Example 1: No Parameters or Filters

Request

This API shall initiate a query request to retrieve Chatbot data associated with Brands that are registered in the VA platform and which are associated with the RCS entity making the request. On success, the response shall contain an array of Chatbot data. The ChatbotId values returned shall be used on GET/chatbots/{Id} requests to retrieve data associated with a specific Chatbot.

Note that the value for the "Authorization" header tag is the `access_token` returned in the /auth response.

Filters

```
Object
Not Applicable
```

Body

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/chatbots"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Attribute	Type	Max	Comments
Chatbots	object		
ChatbotId	string		
ServiceName	string		
BrandId	string		
BrandName	string		
PartnerId	string		
PartnerName	string		
Status	string	15	active, suspended, inactive
Verified	string	15	Aggregate of the individual verifications. Values are complete, not-started, pending, failed, expired or expiry-warning, or Overridden.
IconVerified	string	15	complete, not-started, pending, failed
2FACompleted	string	15	complete, not-started, pending, failed
UpdateDateTime	string	25	`ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.`

Example (Informative)

```
{"Chatbots":[{"ChatbotId":"4ff3d5f6-fba9-46d6-82c9-
010083593dd5","ServiceName":"Generic Brand Chatbot","BrandId":"b1124a12-
6392-4907-b8cc-78b4e6fa0950","BrandName":"Generic
Brand","PartnerId":"046eee8a-ddd6-4475-a9b8-
e3d6db4a5a77","PartnerName":"Generic
Partner","Status":"active","Verified":"pending
","IconVerified":"complete”,”2FACompleted”:”complete”,"UpdateDateTime":"202
0-09-29T21:16:28-04:00"},{"ChatbotId":"46b6b3ad-c2cf-4036-851b-
bfd97478d3a3","ServiceName":"Generic Brand Chatbot 2","BrandId":"b1124a12-
6392-4907-b8cc-78b4e6fa0950","BrandName":"Generic Brand
2","PartnerId":"046eee8a-ddd6-4475-a9b8-
e3d6db4a5a77","PartnerName":"Generic
Partner","Status":"active","Verified":"pending ","IconVerified":"complete,
”2FACompleted”:”complete”,"UpdateDateTime":"2020-09-28T21:16:28-
04:00"},{"ChatbotId":"7673fb23-90fe-4358-994a-
eb6e2210f625","ServiceName":"Generic Brand Chatbot 3","BrandId":"b1124a12-
6392-4907-b8cc-78b4e6fa0950","BrandName":"Generic Brand
3","PartnerId":"046eee8a-ddd6-4475-a9b8-
e3d6db4a5a77","PartnerName":"Generic
Partner","Status":"active","Verified":"complete","IconVerified":"complete",
”2FACompleted”:”complete”,"UpdateDateTime":"2020-09-27T21:16:28-04:00"}]}
```

Example 2: With Filters

This API shall initiate a query request to retrieve Chatbot data associated with Brands that are registered in the VA platform and which are associated with the RCS entity making the request. The request shall accept an optional query parameter to filter the Chatbots returned based on whether they have been verified or not. Also note that multiple filters may be supported. On success, the response shall contain an array of Chatbot data. The ChatbotId values returned shall be used on GET/chatbots/{Id} requests to retrieve data associated with a specific Chatbot.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Filter Parameters

verified	complete, not-started, pending, failed

Body Schema

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
"https://intel-
rcsapi.iconectiv.com/rcsva/v1/chatbots?verified=complete&verified=pending"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Attribute	Type	Comments
Chatbots	object	
ChatbotId	string	
ServiceName	string	
BrandId	string	
BrandName	string	
PartnerId	string	
PartnerName	string	
Status	string	active, suspended, inactive
Verified	string	Aggregate of the individual verifications. Values are complete, not-started, pending, failed, expired or expiry-warning, or Overridden.
IconVerified	string	complete, not-started, pending, failed
2FACompleted	string	complete, not-started, pending, failed
UpdateDateTime	string	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.

Example (Informative)

```
{"Chatbots":[{"ChatbotId":"4ff3d5f6-fba9-46d6-82c9-
010083593dd5","ServiceName":"Generic Brand Chatbot","BrandId":"b1124a12-
6392-4907-b8cc-78b4e6fa0950","BrandName":"Generic
Brand","PartnerId":"046eee8a-ddd6-4475-a9b8-
e3d6db4a5a77","PartnerName":"Generic
Partner","Status":"active","Verified":"pending","IconVerified":"complete",”
2FACompleted”:”complete”,"UpdateDateTime":"2020-09-30T21:16:28-
04:00"},{"ChatbotId":"46b6b3ad-c2cf-4036-851b-
bfd97478d3a3","ServiceName":"Generic Brand Chatbot 2","BrandId":"b1124a12-
6392-4907-b8cc-78b4e6fa0950","BrandName":"Generic Brand
2","PartnerId":"046eee8a-ddd6-4475-a9b8-
e3d6db4a5a77","PartnerName":"Generic
Partner","Status":"active","Verified":"pending","IconVerified":"complete",”
2FACompleted”:”complete”,"UpdateDateTime":"2020-09-29T21:16:28-
04:00"},{"ChatbotId":"7673fb23-90fe-4358-994a-
eb6e2210f625","ServiceName":"Generic Brand Chatbot 3","BrandId":"b1124a12-
6392-4907-b8cc-78b4e6fa0950","BrandName":"Generic Brand
3","PartnerId":"046eee8a-ddd6-4475-a9b8-
e3d6db4a5a77","PartnerName":"Generic
Partner","Status":"active","Verified":"complete","IconVerified":"complete",
”2FACompleted”:”complete”,"UpdateDateTime":"2020-09-28T21:16:28-04:00"}]}
```

Example 3: With ChatbotId Parameter

This API shall initiate a query request to retrieve registration data associated with a Chatbot identified by {id} in the VA platform.   On success, the response shall contain the data associated with the Chatbot.   The Id value used on the URI of the GET request shall be discoverable by sending a GET/chatbots request to retrieve a collection of ChatbotIds.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Request

Query Parameters

Id	Id of the Chatbot

Body Schema

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/chatbots/4ff3d5f6-fba9-46d6-
82c9-010083593dd5"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Attribute	Type	Comments
ChatbotInfo	object	
ServiceName	string	
ServiceId	string	
PartnerId	string	
Website	string	
Description	string	
SMSFallbackNo	string	
Category	array	
ServiceIcon	string	
ServiceIconSN	string	
SNJurisdiction		
ServiceIconOwner	string	
VerificationLevel	Integer	
CapacityProfile	string	
ChatbotType	string	
BrandId	string	
BrandContactInfo	object	
FirstName	string	
LastName	string	
EmailAddress	string	
Title	string	
TelephoneNumber	string	
NetworkProviderId	string	
VerificationInfo	object	
Verified	string	Aggregate of the individual verifications. Values include complete, not-started, pending, failed, expired, expiry-warning.
IconVerified	string	complete, not-started, pending, failed
2FACompleted	string	complete, not-started, pending, failed

Attribute	Type	Comments
ServiceNameVerified	string	complete, not-started, pending, failed
Status	string	active, suspended, inactive
UpdateDateTime	string	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDThh:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.

Example (Informative)

```
{"Status":"active","BrandId":"8b9f4097-2f5f-4722-87d3-
da6844815208","NetworkProviderId":"721420d4-d9c8-11ea-b296-
82bc83aa48e0","ChatbotInfo":{"ServiceName":"RBMTestQAChatbot002","ServiceId
":"sip:user@botplatform.platdomain","PartnerId":"ae2fbd5a-649f-4da4-ba83-
2579d670ff08","Website": "www.productabc.com","Description":"Generic Brand
Sales Offering Chatbot","SMSFallbackNo":"A23456","Category":["GSMA Standard
Category",”Category2”],"ServiceIconSN":"77876213","SNJurisdiction":"US","Se
rviceIconOwner":"Generic Brand’s Trademarked Logo Owner
Name","CapacityProfile":"Capacity
Profile","ChatbotType":"external","ServiceIcon":"base64 encoded string goes
here"},“BrandContactInfo”:{“FirstName”:”John”,“LastName”:”Doe”,“EmailAddres
s”:“john.doe@anycompany.com”,“Title”:”Brand
Manager”,“TelephoneNumber”:”12124449876”},”VerificationInfo":{"Verified":"p
ending","IconVerified":"complete",”2FACompleted”:”complete”,"ServiceNameVer
ified":"complete"},"UpdateDateTime":"2020-09-25T21:16:28-07:00"}
```

Example 4: ChatbotId Documents

Request

This API shall initiate a request to retrieve Chatbot verification documents registered in the VA platform. Also note that multiple filters may be supported. If the query parameter is omitted, then all documents shall be returned in the response body.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Query Parameters

```
Id                      Id of the Chatbot
type                    Comma separated list (no spaces) of one or
                        more document types to get (JWT)
```

Body

```
Object
   Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
https://intel-rcsapi.iconectiv.com/rcsva/v1/chatbots/4ff3d5f6-fba9-46d6-
82c9-010083593dd5/documents?=JWT}"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Attribute	Type	Comments
JWT	String	Base64 encoded string of the Chatbot JWT for the Chatbot represented by {id}. The JWT conforms to the GSMA RCC.07 specifications. Refer to Appendix A of this interface spec for the complete format.

Example (Informative)

```
{"Chatbot":{"JWT":"<base64>"}}
```

3.3.4 PATCH /chatbots/{id}

```
https://<API_Host> /chatbots/{id}
```

Request

This API shall initiate a request to update registration data associated with a Chatbot registered in the VA platform. Only those attributes being updated shall need to be included in the request body. The VA platform shall initiate Chatbot renewal when the ‘Verified’ attribute in the request body is set to ‘complete’ and the Chatbot verification is approaching expiry. Verified aggregate status can be obtained by performing a GET /chatbots request).

Note that in order to remove/delete a previously provided value for a field (through a POST or PATCH), the field value shall be set to JSON null. The use of “” or blanks/spaces between “” is invalid and will return a syntax error. Additionally, fields which are not supported by a message or are not applicable for a country will be ignored, and no error will be triggered.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Call Example (Informative)

```
curl -X PATCH\
-H "Accept: application/json"\
-H "Content-Type: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/chatbots/9942eaf3-4d67-46e6-
89a7-6ca311513ee5"
```

Path Parameters

Id	Id of the Chatbot

Body

Attribute	Type	Required v Optional
ChatbotInfo	object	O
ServiceName	string	O
ServiceId	string	O
PartnerId	string	O
Website	string	O
Description	string	O
SMSFallbackNo	string	O
Category	array	O
ServiceIcon	string	O
ServiceIconSN	string	O
SNJurisdiction	string	O
ServiceIconOwner	string	O
CapacityProfile	String	O
ChatbotType	string	O
BrandId	string	O
BrandContactInfo	object	O

FirstName	string	O
LastName	string	O
EmailAddress	string	O
Title	string	O
TelephoneNumber	string	O
NetworkProviderId	string	O
Verify	string	O

Example (Informative)

```
{
    "ChatbotInfo": {
        "ServiceName": "TestChatbot",
        "ServiceId": "https://www.chatbot.com",
        "PartnerId": "de00c136-9e55-4ea3-8ce8-4ca0c5cb6346",
        "Website": "www.productabc.com",
  "Description": "Test Chat",
        "SMSFallbackNo": "2025553333",
        "Category": ["testQAcategory",”Category2”],
        "ServiceIcon": "{{serviceIcon base64}}",
        "ServiceIconSN": "12345678",
        "SNJurisdiction": "US",
        "ServiceIconOwner": "ABC",
        "CapacityProfile": "TestCapacity",
        "ChatbotType": "external"
    },
    “Verify”: “complete”
}
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Attribute	Type
ChatbotId	string

Example (Informative)

```
{"ChatbotId":"9942eaf3-4d67-46e6-89a7-6ca311513ee5"}
```

3.3.5 DELETE /chatbots/{id}

This method shall support removal of a Chatbot from the VA platform.

Request

The API initiates a request to delete a chatbot object from the VA platform for the chatbot identified by {id}.   Only the account that created the chatbot object can delete it.

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response.

Path Parameters

Id	Id of the Chatbot

Body Schema

```
Object

  Not Applicable
```

Example (Informative)

```
curl -X DELETE\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/chatbots/6ab19bc8-3304-4ff6-
aa67-36450cfd3c95"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

```
Object
   ChatbotId                  string
```

Example (Informative)

```
{"ChatbotId":"6ab19bc8-3304-4ff6-aa67-36450cfd3c95"}
```

3.4 Notification API

3.4.1 Resource: Notification

Request

The resource used is

```
https://{ServerRoot}/rcsva/{apiVersion}/notification
```

Name	Type	Description
ServerRoot	Path	The Server Hosting the API
ServiceRoot	Path	The Root specifically for the Service is RCSVA
apiVersion	Path	The version of API being used
apiMethod	Path	The API method being used, such as Notification

Example (Informative)

Sample Chatbots resource address:

```
https://intel-rcsapi.iconectiv.com/rcsva/v1/notification
```

3.4.2 PUT /notification

Request

This API shall initiate a request to set the callback URI to an endpoint in the VA user's domain that is able to receive notifications from the VA system. The request body shall contain both CallbackURI and AuthURI.

Note that the value for the "Authorization" header tag is the `access_token` returned in the /auth response.

Request Example

```
curl -X PUT\
-H "Accept: application/json"\
-H "Content-Type: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/notification"
```

Body Schema

Attribute	Type	Required v Optional	Maximum Length	Comments
CallbackURI	string	R	2048	URI where notifications are sent
AuthURI	string	R	2048	URI used to authenticate and return access token
Filter	array(string)	O	50	Array of notifications to receive. Possible values include CRL, JWT, Partner, Brand and Chatbot. If no filters are provided, all notifications will be received.

Example (Informative)

```
{
 "CallbackURI": "https://rcsvatest.customera.com/dev/notification",
 "AuthURI": "https://rcsvatest.customera.com/oauth2/token",
 "Filter": ["JWT"]
}
```

Responses

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

```
Object
  Not Applicable
```

Example (Informative)

```
200OK
```

3.4.3 GET /notification

Request

This API shall initiate a request to get the value of the CallbackURI and notification filter previously set by the VA user via the PUT method /notification.

Note that the value for the "Authorization" header tag is the access_token returned in the /auth response.

Request Example

```
curl -X GET\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/notification"
```

Body Schema

```
Object
  Not Applicable
```

Responses

Body Schema

Attribute	Type	Required v Optional	Comments
CallbackURI	string	R	URI where notifications are sent
AuthURI	string	R	URI used to authenticate and return access token
Filter	array	O	Array of notifications to receive. Possible values include CRL, JWT, Partner, Brand and Chatbot. If no filters are provided, all notifications will be received.

Table 1Body Schema

Example (Informative)

```
{
 "CallbackURI": "https://rcsvatest.customera.com/dev/notification",
 "AuthURI": "https://rcsvatest.customera.com/oauth2/token",
 "Filter": ["JWT"]
}
```

3.4.4 DELETE /notification

Request

This API shall initiate a request to delete the callbackURI previously set by the VA user via the PUT /notification API. The DELETE method shall be used to stop the VA system from sending notifications to the callbackURI

Note that the value for the “Authorization” header tag is the `access_token` returned in the /auth response

Request Example

```
curl -X DELETE\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/notification"
```

Body Schema

```
Object
  Not Applicable
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body Schema

```
Object
  Not Applicable
```

Example (Informative)

```
200OK
```

3.5 Autonomous Notifications

3.5.1 Resource: Notification

Request

The resource used is

```
{CallbackURI provided to the VA}
```

Example (Informative)

Sample Chatbots resource address:

```
https://rcsvatest.customera.com/dev/notification
```

3.5.2 Push Interface

The VA shall supply notifications to the Partner or Service Provider as follows. Note that this is a PUSH to the party who registers the data.

This API shall initiate a notification message to the CallbackURI to notify the VA user of a CRL change, reissue of a JWT signature or an entity attribute update.

Attribute	Type	Required v Optional	Maximum Length	Comments
CRL	object	O		Container for CRL attributes
CRL_URI	string		2048	URI where updated CRL can be retrieved
CreatedTS	integer		25	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDTHH:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.
JWT	object	O		Container for JWT attributes
ChatbotId	string		Uuid	ChatbotId associated with the resissued JWT
IssuedTS	integer		25	ISO 8601 Timestamp when the record was last updated in the form YYYY-MM-DDTHH:mm:ssTZD, where TZD is replaced by ±hh:mm for local time or by Z for UTC.
Entity		Required v Optional		Container for entity attributes
EntityType	string	O	10	Partner, Brand or Chatbot
EntityId	String		Uuid	Id of entity that was updated
NotifyReason	String		10	Create, Modify or Delete
		O		

Table 2 Body Schema

Example Notification

```
{
"JWT": { "ChatbotId": "41caba6e-9d67-4742-8f16-4826aa4b56dd", "IssuedTS":
"2021-07-28T20:05:04Z" }
}
```

3.6 RCSVA Public Certificate API

3.6.1 Resource: Public Certificate

The resource used is

```
https://{ServerRoot}/rcsva/{apiVersion}/certificate?{ChatbotId}&{a
lgorithm}
```

Name	Type	Description
ServerRoot	Path	The Server Hosting the API
ServiceRoot	Path	The Root specifically for the Service is RCSVA
apiVersion	Path	The version of the API being used
apiMethod	Path	The API method being used, such as certificate

Example (Informative)

Sample resource address:

```
https://intel-rcsapi.iconectiv.com/rcsva/v1/certificate
```

3.6.2 GET/Certificate URI

Per agreement between the RCS SP and the VA will define the specific query parameters. The API initiates a request to retrieve the certificate based on the JWS x5u to obtain the RCSVA public key that shall be used to validate the JWS digital signatures. Additional parameters may be agreed to with the VA as part of the JWT and as optional extensions for this URI.

GET/certificate

Attribute	Type	Required	Max	Comments
ChatbotId	String	R	Uuid	
Algorithm	String	O	10	Default algorithm is ES256 if not specified

Table 3 Query Parameters

Body

```
Object
  Not Applicable
```

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
https://rcsapi.iconectiv.com/rcsva/v1/certificate?ChatbotId=ece5c252-5b1c-
45d2-851c-09d9da6fada5&algorithm=ES256
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Object			
1	Certificate	string	Base64 PEM encoded Certificate

Example (Informative)

```
{
{"payload":"eyJpYXQiOjE2MTMyNDMwOTcsImljb25maW5nZXJwcmludCI6IjEzZWI5ZDE2YjY
zNTcwZjU0MGQ1OGQ0ZTI1Yzk1NDJlZWZjNmY0MTNkNWVlZjkwYWY1NGI3ZTYwMjVhYjEyNzUiLC
JpZCI6Imh0dHBzOi8vd3d3LmNoYXRib3QuY29tIiwibmFtZSI6IkRFTU9fTW9udEJsYW5jXzAxI
n0","signatures":[{"protected":"eyJhbGciOiJFUzI1NiIsImJvdHZmZXhwaXJlcyI6IjI
wMjItMDItMTNUMTk6MDQ6NTZaIiwiY3JpdCI6WyJib3R2ZmV4cGlyZXMiXSwieDV1IjoiaHR0cH
M6Ly9pbnRlbC1yY3NhcGkuaWNvbmVjdGl2LmNvbS9yY3N2YS92MS9jZXJ0aWZpY2F0ZT9jZXJ0S
WQ9Q2VydEluZm9fQ2hhdGJvdF9lY2U1YzI1Mi01YjFjLTQ1ZDItODUxYy0wOWQ5ZGE2ZmFkYTUm
YWxnb3JpdGhtPUVTMjU2JmFjY291bnRJZD1iODI3ZWZmMC02ZDc2LTExZWItYThkZS0yNjMyOTE
0OTQ1NGMifQ","header":{"kid":"a793d8be89b1395e752ec8b6628adc8d5396782643970
a61e8364f307604b2ea"},"signature":"gkxo_I8zQHDQGYooHctPvk19mXoMVLJBe6tw_wzy
0Dooq6uldVJ-C61Gb6WXlaYceeB8ZYS-
tpdu5Rm0Ec5xjw"},{"protected":"eyJhbGciOiJSUzI1NiIsImJvdHZmZXhwaXJlcyI6IjIw
MjItMDItMTNUMTk6MDQ6NTZaIiwiY3JpdCI6WyJib3R2ZmV4cGlyZXMiXSwieDV1IjoiaHR0cHM
6Ly9pbnRlbC1yY3NhcGkuaWNvbmVjdGl2LmNvbS9yY3N2YS92MS9jZXJ0aWZpY2F0ZT9jZXJ0SW
Q9Q2VydEluZm9fQ2hhdGJvdF9lY2U1YzI1Mi01YjFjLTQ1ZDItODUxYy0wOWQ5ZGE2ZmFkYTUmY
Wxnb3JpdGhtPVJTMjU2JmFjY291bnRJZD1iODI3ZWZmMC02ZDc2LTExZWItYThkZS0yNjMyOTE0
OTQ1NGMifQ","header":{"kid":"4ce058246c9ec7eb5f20232c5969874b01f5a5c351cdf8
e09ce07e1dac1a30be"},"signature":"KIigIN7-UrTP0iv4LsfohlC34PXuIqo-
ogXcIV_MEqVWnUUxeHaxrDYOtrVsI-Etn_NpjblClKQkBJziZ1y4sJ1EPJf89Hn4-
uDrAL_VFPmAdEVsfjTPbIW3wc74p6PPAMu9nsjG5-
0Qs5zd0tTNxFTyJOHPka_18Syt9OG24YA"}]}
```

3.7 Utility APIs

These APIs provide utility functions for the service.

3.7.1 Resources: Network Providers

The Network Providers resource shall support a request to get the list of network providers that are associated with the VA. This method is anticipated to be used by the Partner role in order to retrieve the NetworkProviderId for use in the Chatbot POST and PATCH methods.

Requests

The resource used is

```
https://{ServerRoot}/rcsva/{apiVersion}/util
```

Name	Type	Description
ServerRoot	Path	The Server Hosting the API
ServiceRoot	Path	The Root specifically for the Service is RCSVA
apiVersion	Path	The version of the API being used
apiMethod	Path	The API method being used, such as util

Example (informative)

Sample resource address:

```
curl -X GET\
-H "Accept: application/json"\
https://intel-rcsapi.iconectiv.com/rcsva/v1/util/network_providers
```

3.7.2 GET / Network_Providers

The API initiates a request to get the list of network providers that are associated with the VA.

Note that the value for the “Authorization” header tag is the access_token returned in the /auth response.

Request

Filters

```
  Not Applicable
```

Body

Object	
Not Applicable	

Table 4 Body

Example (Informative)

```
curl -X GET\
-H "Accept: application/json"\
"https://intel-rcsapi.iconectiv.com/rcsva/v1/util/network_providers"
```

Responses

● Success 200

Please refer to Annex B for a list of applicable error codes and messages. In the event of an Error Code the body shall contain the Error Code Number and explanatory message.

Body

Object		
NetworkProviders	**array[object]**	**Array of network providers**
NetworkProviderId	string	
NetworkProviderName	string	
NetworkProviderVerified	string	complete, not-started, pending, failed
NetworkProviderStatus	string	pending, active

Example (Informative)

```
{"NetworkProviders":[{"NetworkProviderId":"487e2b46-1476-11eb-804a-
3e16735c7110","NetworkProviderName":"IC QA Test
MNO1","NetworkProviderVerified":"complete","NetworkProviderStatus":"active”
}]}
```

4 Process Flows and Diagrams

4.1 Authentication

API should NOT be specific on which flavour of Auth is to be used but should adhere to industry best practices for the obtaining of, and expiry of tokens.

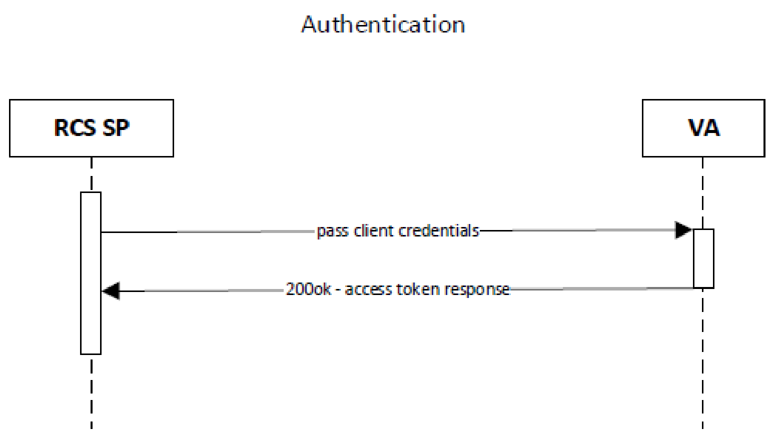


Figure 1 Authentication Flow

4.2 Notification Channel

A Client may wish to receive notifications on the status changes of submitted object. By creating a notification channel, they may listen for events relative to the MaaP platform, Partner, Brand or Chatbot Objects.

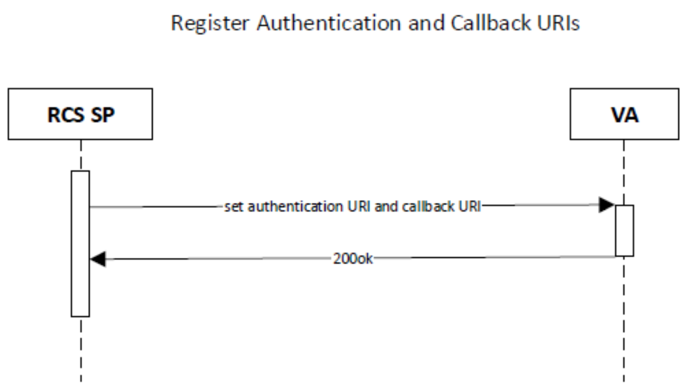

Register Authentication and Callback URIs
RCS SP
VA
set authentication URI and callback URI
200ok

Figure 2 Register Uthentication Flow

4.3 Partner Verification via RCS SP

- The Partner Object is submitted.
- The Partner ID is returned to the client.
- The Verification of the Partner is a policy decision and happens according to agreement between the MaaP provider and the VA.
- The Verification status is made available on request.

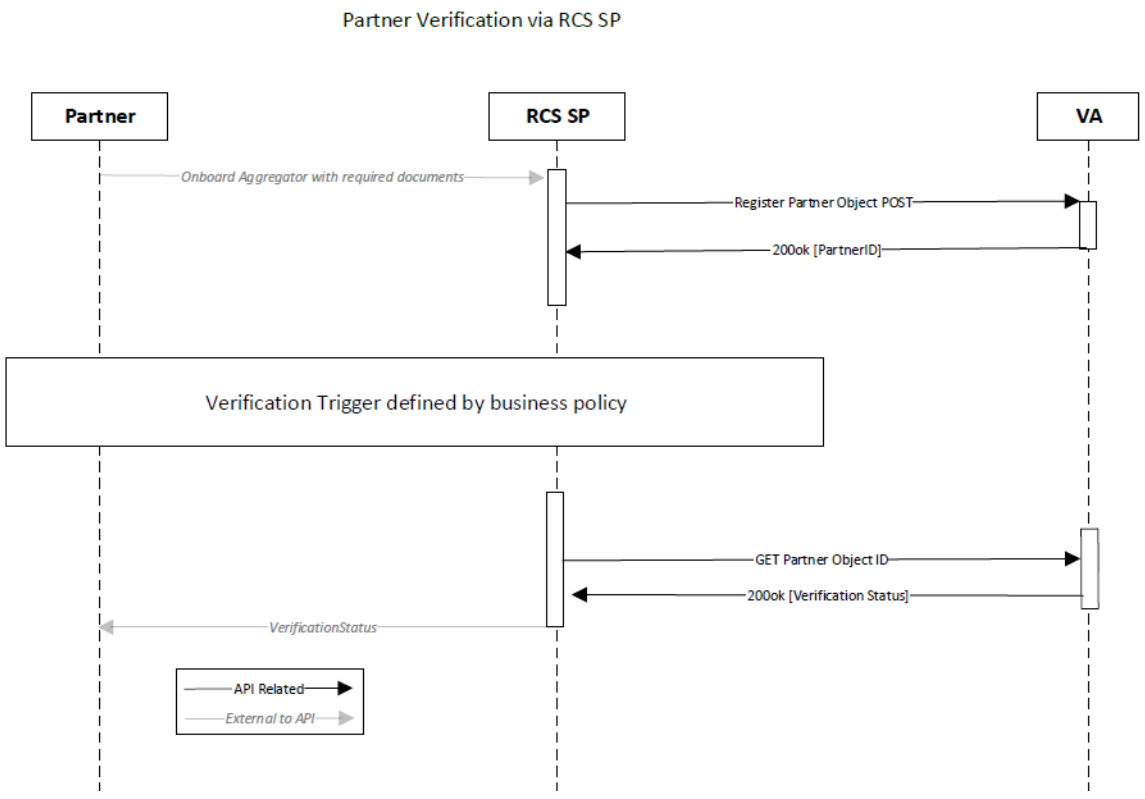

Partner Verification via RCS SP
Partner
RCS SP
VA
Onboard Aggregator with required documents
Register Partner Object POST
200ok [PartnerID]
Verification Trigger defined by business policy
GET Partner Object ID
200ok [Verification Status]
VerificationStatus
API Related
External to API

Figure 3 Partner Verification Flow

4.4 Brand Verification via RCS SP

- The Brand Object is submitted.
- The Brand ID is returned to the client.
- The Verification of the Brand is a policy decision and happens according to agreement between the MaaP provider and the VA.
- The Verification status is made available on request.

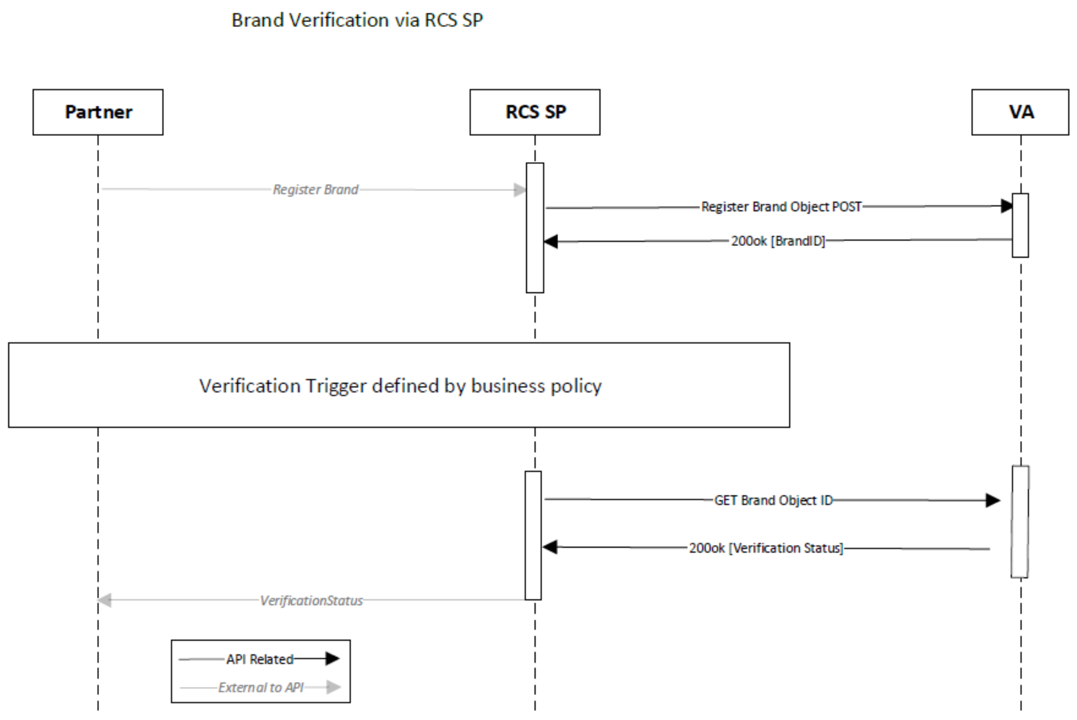

Brand Verification via RCS SP
Partner
RCS SP
VA
Register Brand
Register Brand Object POST
200ok [BrandID]
Verification Trigger defined by business policy
GET Brand Object ID
200ok [Verification Status]
VerificationStatus
API Related
External to API

Figure 4 Brand Verification Flow

4.5 Chatbot Verification via RCS SP

- The Chatbot Object is submitted.
- The Chatbot ID is returned to the client.
- The Verification process is triggered at the VA, and may start verification of the Brand and Partner (depending on policy).
- The Verification status is made available on request.
- The Client downloads the signature
- VA signature is checked against the documents uploaded and against the public key provisioned on the MaaP platform.

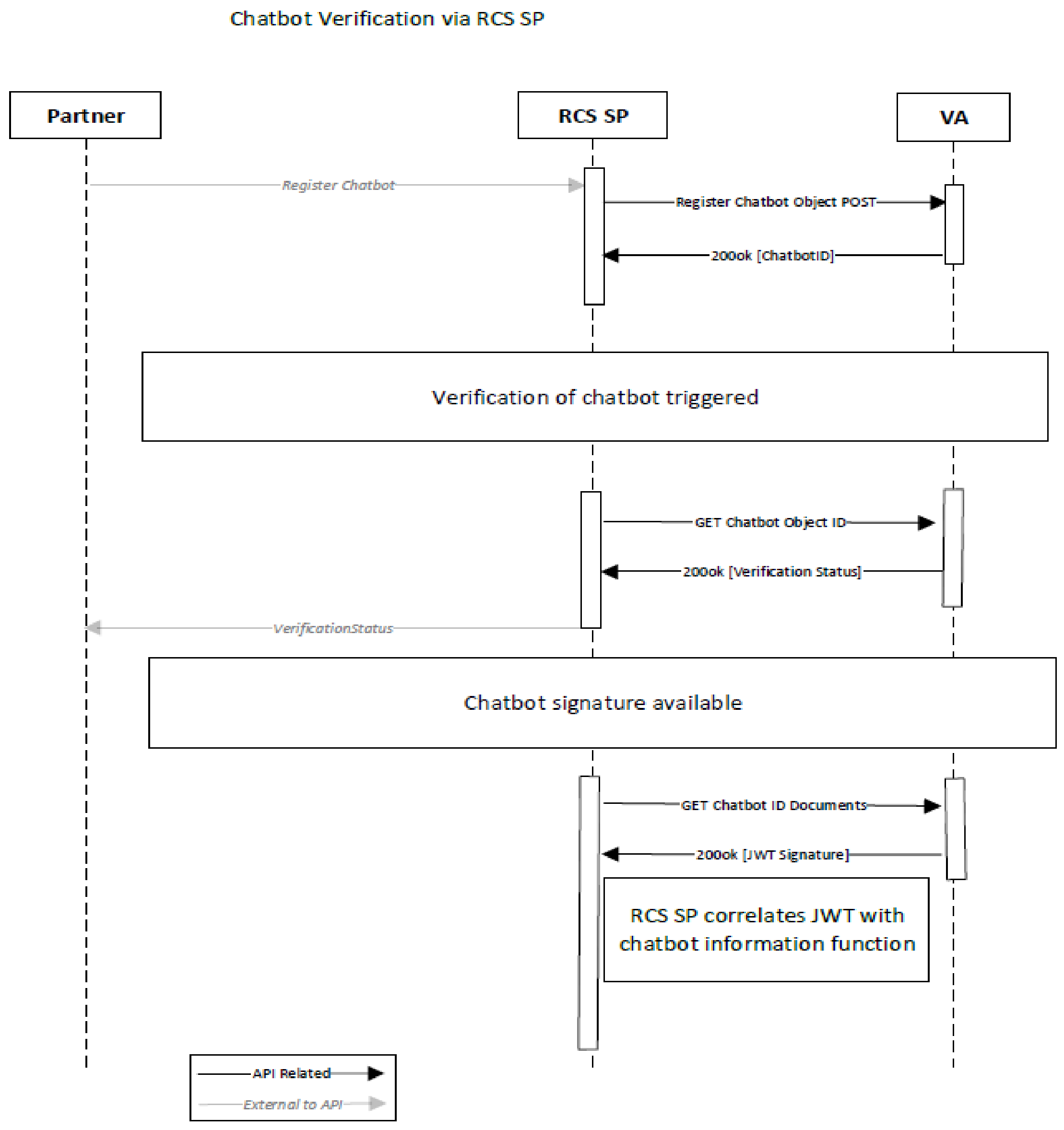

Chatbot Verification via RCS SP
Partner
RCS SP
VA
Register Chatbot
Register Chatbot Object POST
200ok [ChatbotID]
Verification of chatbot triggered
GET Chatbot Object ID
200ok [Verification Status]
VerificationStatus
Chatbot signature available
GET Chatbot ID Documents
200ok [JWT Signature]
RCS SP correlates JWT with chatbot information function
API Related
External to API

Figure 5 Chatbox Verification Flow

4.6 Partner Verification directly to VA

- The Partner Object is submitted directly by the Partner/aggregator
- The Partner ID is returned.
- The Verification of the Partner is a policy decision and happens according to agreement between the MaaP provider and the VA.
- The Verification status is made available on request.

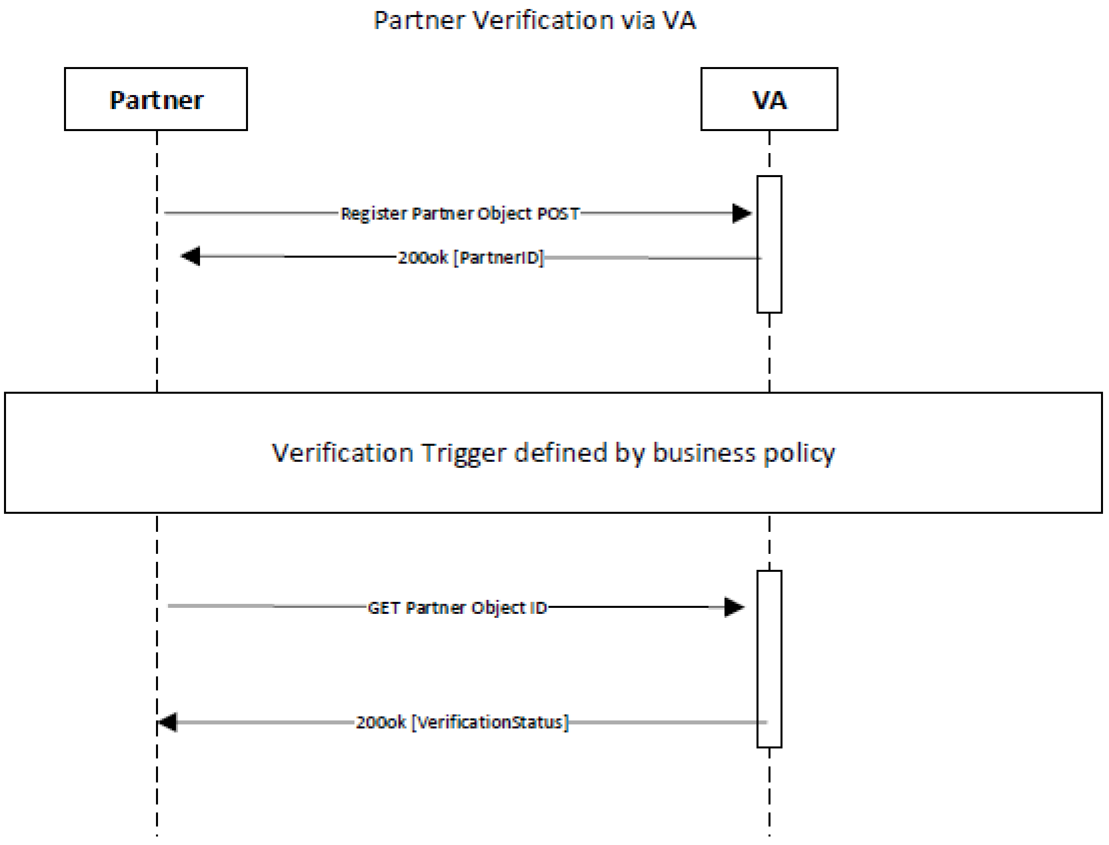

Partner Verification via VA
Partner
VA
Register Partner Object POST
200ok [PartnerID]
Verification Trigger defined by business policy
GET Partner Object ID
200ok [VerificationStatus]

Figure 6 Partner Verification Flow

4.7 Brand Verification directly to VA

- The Brand Object is submitted.
- The Brand ID is returned to the client.
- The Verification of the Brand is a policy decision and happens according to agreement between the MaaP provider and the VA.
- The Verification status is made available on request.

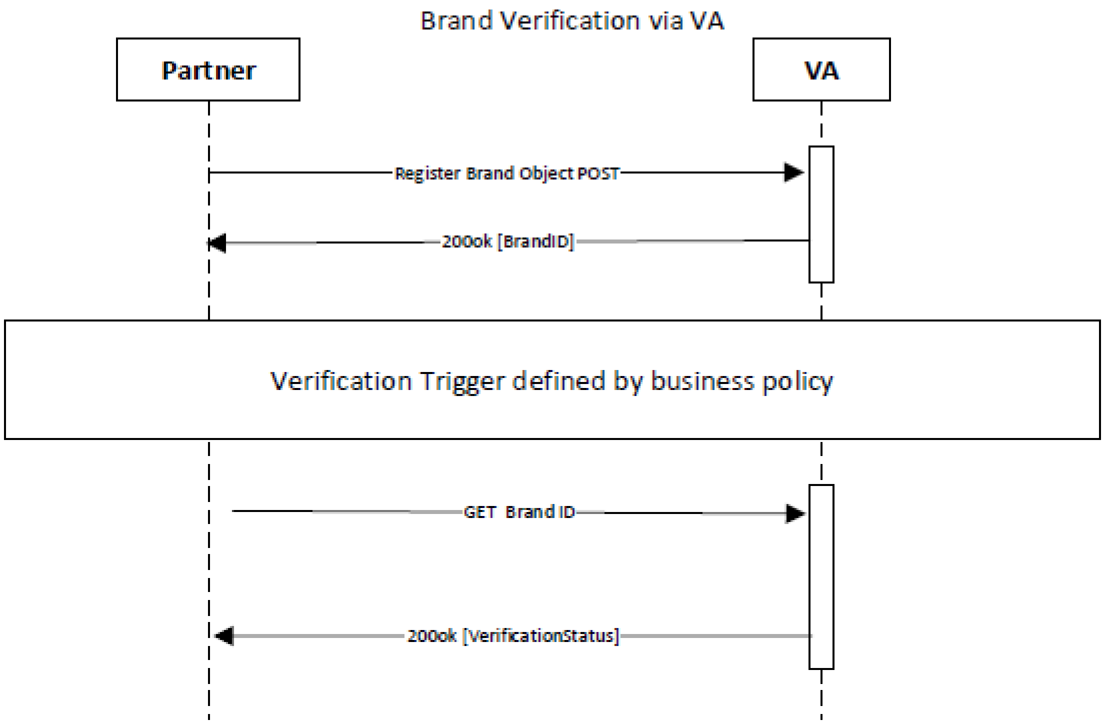

Brand Verification via VA
Partner
VA
Register Brand Object POST
200ok [BrandID]
Verification Trigger defined by business policy
GET Brand ID
200ok [VerificationStatus]

Figure 7 Brand Verification Flow

4.8 Chatbot Verification directly to VA

- The Chatbot Object is submitted by Partner/Aggregator.
- The Chatbot ID is returned to the client.
- The Verification process is triggered at the VA and may start verification of the Brand and Partner (depending on policy).
- The Partner can request Verification status - is made available on request.
- The Client downloads the signature and uploads to the RCS SPMaaP Platform.
- VA signature is checked against the documents uploaded (decoded using the public key provisioned on the MaaP platform).
- RCS SP may also check provisioned details of the Chatbot, Brand and Partner using GET APIs shown in previous slides.

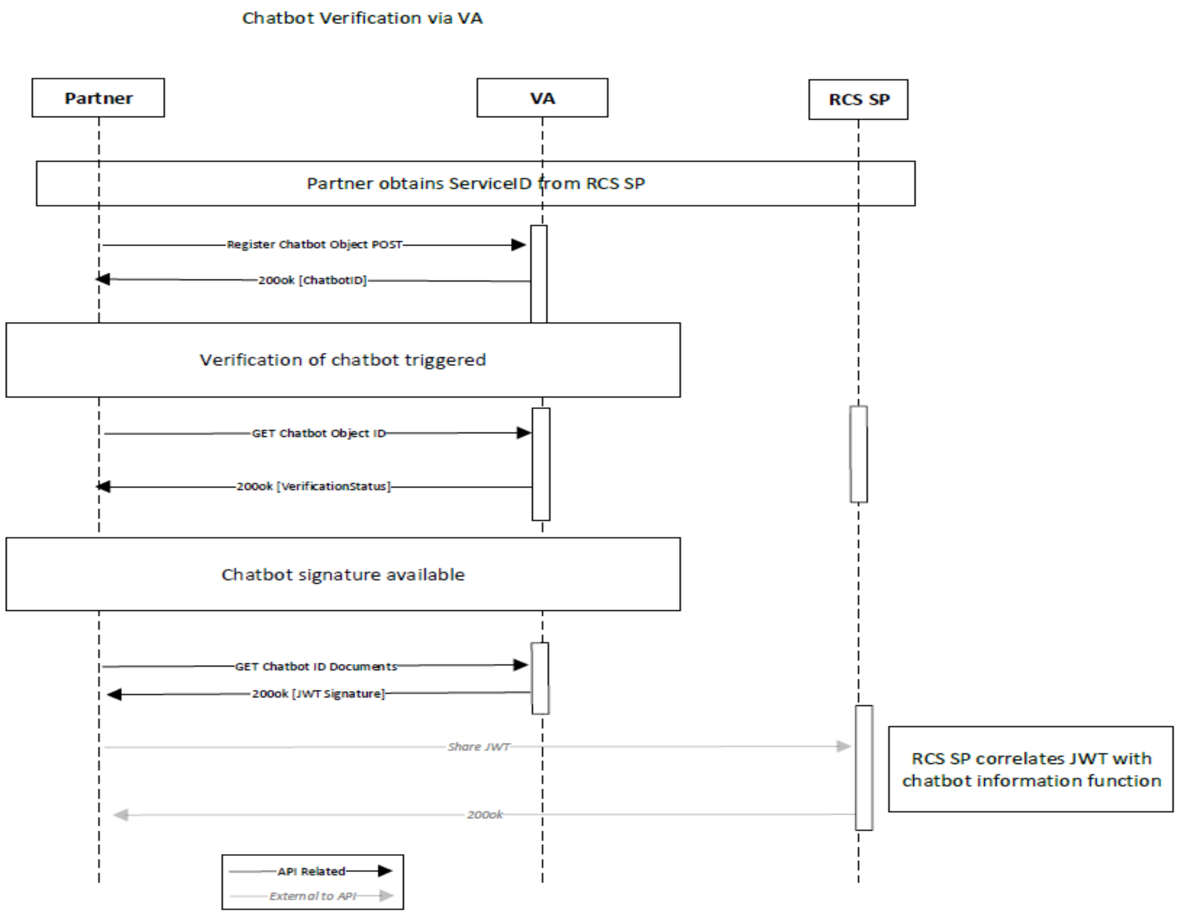

Chatbot Verification via VA
Partner
VA
RCS SP
Partner obtains ServiceID from RCS SP
Register Chatbot Object POST
200ok [ChatbotID]
Verification of chatbot triggered
GET Chatbot Object ID
200ok [VerificationStatus]
Chatbot signature available
GET Chatbot ID Documents
200ok [JWT Signature]
Share JWT
200ok
RCS SP correlates JWT with chatbot information function
API Related
External to API

Figure 8 Chatbot Verification Flow

Annex A State Flow Diagram

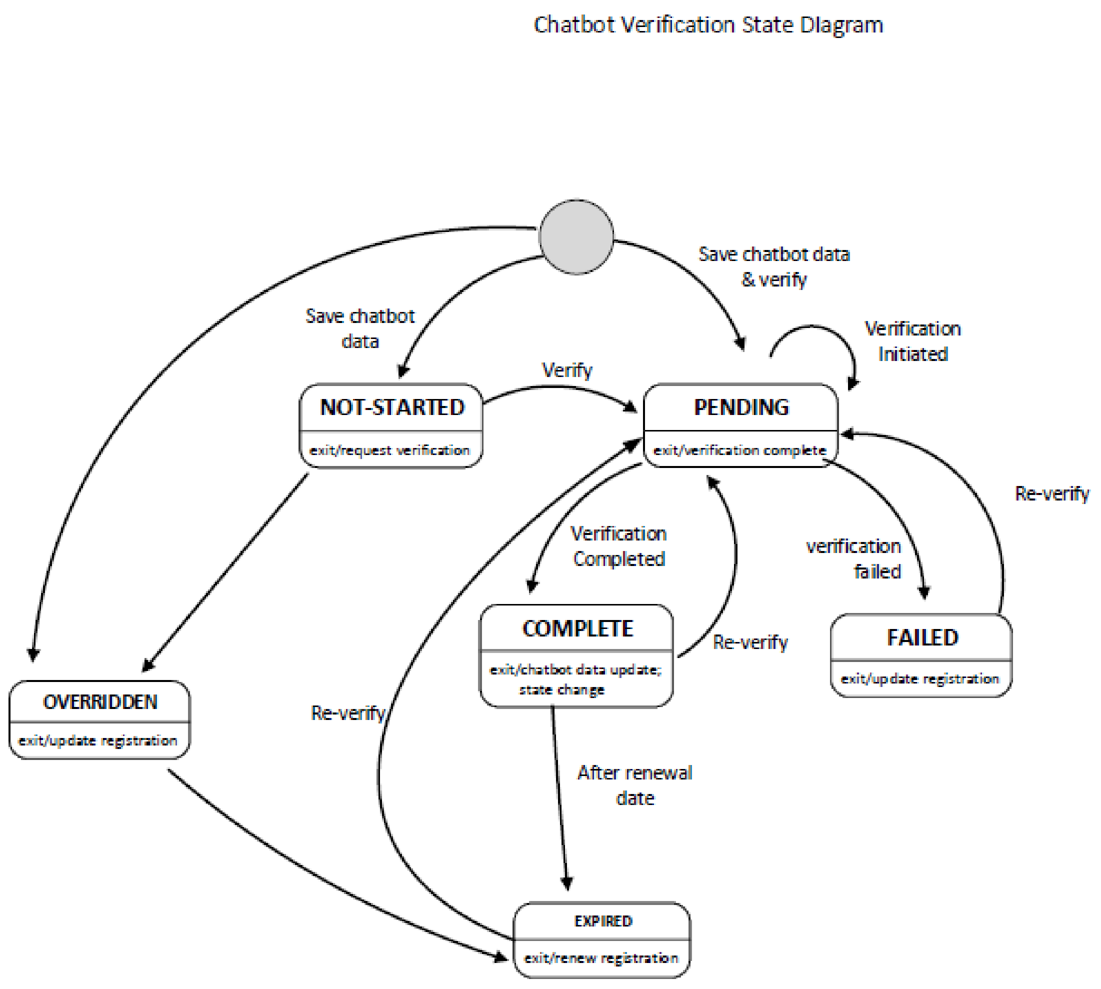


Figure 9 Chatbot Verification State Flow Diagram

Annex B Error Responses and Message Codes

In the event of a failure, the RCS VA APIs will respond with a common error response structure, which is applicable to all APIs.

Error response structure:

```
      {
"messages":       [
{
"code": “XXXXX”
"message": "message text",
}
],
"status": "failure"
}
```

The structure consists of the following components:

- messages: Depending on the error condition, there may be one or multiple error conditions. This is an array of messages that allows the application to convey all applicable errors
- code: 5-digit Error Code that uniquely identifies each error condition
- message: Description associated with the applicable Error Code
- status: The status is set to failure

The table below contains a list of recommended Error Codes and corresponding Messages that could be returned in the external APIs.

API	Operation	HTTP	code	message
Brands	DELETE	400	11025	Path parameter <PathFieldName> has an invalid format [Note: replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Brands	DELETE	400	13204	The entity has an active Chatbot therefore request for deletion can’t be completed
Brands	DELETE	400	21115	Path parameter id does not belong to a {Partner, Brand} entity [Note: where "Partner"is used for Partner API, and "Brand" for Brand API]
Brands	DELETE	400	21122	Non-active entities cannot be updated

API	Operation	HTTP	code	message
Brands	DELETE	400	21123	Entity is currently going through the verification process. Please try again later
Brands	DELETE	400	24309	Requestor must be an verified entity
Brands	DELETE	400	24312	Requestor must be an active entity
Brands	DELETE	401	no error structure	Note: Message text in the body: Unauthorized
Brands	DELETE	401	no error structure	Note: Message text in the body: The incoming token has expire d
Brands	DELETE	403	24302	The request failed because the requestor did not create the entity
Brands	DELETE	404	11028	URL requested was not found
Brands	DELETE	404	24301	The requestor entity was not found
Brands	DELETE	404	24400	The entity requested was not found
Brands	DELETE	500	10500	Internal Server Error occurred. Please retry later or contact your RCSVA Administrator
Brands	DELETE	500	25100	Internal Server Error occurred. Please retry later or contact your RCSVA Administrator
Brands	GET	400	11024	{Field Name} value is invalid
Brands	GET	400	24309	Requestor must be an verified entity
Brands	GET	401	no error structure	Note: Message text in the body: Unauthorized
Brands	GET	401	no error structure	Note: Message text in the body: The incoming token has expire d
Brands	GET	403	24305	The requested entity can't be retrieved by an entity with the given requestor entity type or the entity was not authorized to request this resource
Brands	GET	404	11028	URL requested was not found
Brands	GET	404	24301	The requestor entity was not found

API	Operation	HTTP	code	message
Brands	GET	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Brands	GET	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Brands	GET{id}	400	11025	Path parameter <PathFieldName> has an invalid format [Note: replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Brands	GET{id}	400	21115	Path parameter id does not belong to a {Partner, Brand} entity [Note: where "Partner"is used for Partner API, and "Brand" for Brand API]
Brands	GET{id}	400	24309	Requestor must be an verified entity
Brands	GET{id}	401	no error structure	Note: Message text in the body: Unauthorized
Brands	GET{id}	401	no error structure	Note: Message text in the body: The incoming token has expire d
Brands	GET{id}	403	24302	The request failed because the requestor did not create the entity
Brands	GET{id}	404	11028	URL requested was not found
Brands	GET{id}	404	24301	The requestor entity was not found
Brands	GET{id}	404	24400	The entity requested was not found
Brands	GET{id}	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Brands	GET{id}	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Brands	PATCH	400	11000	{Field Name} requires a non-blank value
Brands	PATCH	400	11001	{Field Name} length must be minimum <n> and maximum <n>
Brands	PATCH	400	11002	{Field Name} has an invalid format

API	Operation	HTTP	code	message
Brands	PATCH	400	11003	{Field Name} length must be maximum <n>
Brands	PATCH	400	11004	Invalid syntax present in the request
Brands	PATCH	400	11008	When specified, {Field Name} must be a non-blank value
Brands	PATCH	400	11009	CountryOfIncorp & StateOfIncorp can only be updated in a pair
Brands	PATCH	400	11014	When any of the DefaultIcon fields (DefaultIcon, ServiceIconSN, SNJurisdiction & ServiceIconOwner) are specified, then all of them must be specified
Brands	PATCH	400	11017	SNJurisdiction is required and must be US when ServiceIconSN is specified
Brands	PATCH	400	11018	SNJurisdiction, ServiceIconSN and ServiceIconOwner should be null when DefaultIcon is null
Brands	PATCH	400	11019	When any of the BrandAddress fields (StreetAddress1, City, GoverningDistrict, PostalCode, Country) are missing but required for a given Country
Brands	PATCH	400	11020	When any of the BrandMgmtContactInfo fields (FirstName, LastName, EmailAddress, Title, TelephoneNumber) are specified, then all of them must be specified as a non-blank value
Brands	PATCH	400	11025	Path parameter <PathFieldName> has an invalid format [replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Brands	PATCH	400	11029	{Field Name} can't be changed on a completed entity [NOTE: replacing {Field Name} with Country]
Brands	PATCH	400	11030	Verified must be set to complete when field updates will require reverification

API	Operation	HTTP	code	message
Brands	PATCH	400	11031	The calculated converted image size of the {fieldname} base64 encoded string exceeds the maximum 2 MB limit [Note: Replacing fieldname with DefaultIcon or ServiceIcon]
Brands	PATCH	400	11033	{Field Name} can't be changed on a verified entity [NOTE: replacing {Field Name} with "RegNumber" or "RefNumber"]
Brands	PATCH	400	21101	The entity can't be updated when the status is complete
Brands	PATCH	400	21103	{Field Name} value is invalid [Note: where {Field Name} is CountryOfIncorp or Country]
Brands	PATCH	400	21104	StateOfIncorp value is invalid
Brands	PATCH	400	21105	GoverningDistrict value is invalid
Brands	PATCH	400	21115	Path parameter id does not belong to a {Partner, Brand} entity [Note: where "Partner" is used for Partner API, and "Brand" for Brand API]
Brands	PATCH	400	21118	Verify must be set to complete when field updates will require reverification of the entity or service icon
Brands	PATCH	400	21122	Non-active entities cannot be updated
Brands	PATCH	400	21123	Entity is currently going through the verification process. Please try again later
Brands	PATCH	400	21300	An entity with the same RegNumber exists. Therefore, the entity creation request can't be honored
Brands	PATCH	400	24309	Requestor must be an verified entity
Brands	PATCH	400	24312	Requestor must be an active entity
Brands	PATCH	401	no error structure	Note: Message text in the body: Unauthorized
Brands	PATCH	401	no error structure	Note: Message text in the body:

API	Operation	HTTP	code	message
				The incoming token has expire d
Brands	PATCH	403	24302	The request failed because the requestor did not create the entity
Brands	PATCH	404	11028	URL requested was not found
Brands	PATCH	404	24301	The requestor entity was not found
Brands	PATCH	404	24400	The entity requested was not found
Brands	PATCH	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Brands	PATCH	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Brands	POST	400	11000	{Field Name} requires a non-blank value
Brands	POST	400	11001	{Field Name} length must be minimum <n> and maximum <n>
Brands	POST	400	11002	{Field Name} has an invalid format
Brands	POST	400	11003	{Field Name} length must be maximum <n>
Brands	POST	400	11004	Invalid syntax present in the request
Brands	POST	400	11015	DefaultIcon is required when <Field Name > is specified [Note: where {Field Name} may be ServiceIconSN, SNJurisdiction or ServiceIconOwner]
Brands	POST	400	11017	SNJurisdiction is required when ServiceIconSN is specified
Brands	POST	400	21103	{Field Name} value is invalid [Note: where {Field Name} is CountryOfIncorp or Country]
Brands	POST	400	21104	StateOfIncorp value is invalid
Brands	POST	400	21105	GoverningDistrict value is invalid
Brands	POST	400	21300	An entity with the same RegNumber exists. Therefore,

API	Operation	HTTP	code	message
				the entity creation request can't be honored
Brands	POST	400	24304	Requestor must be a verified entity in order to create a Brand
Brands	POST	400	24312	Requestor must be an active entity
Brands	POST	401	no error structure	Note: Message text in the body: Unauthorized
Brands	POST	401	no error structure	Note: Message text in the body: The incoming token has expire d
Brands	POST	404	11028	URL requested was not found
Brands	POST	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Brands	POST	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Certificate	GET	400	11000	{Field Name} requires a non-blank value
Certificate	GET	400	11002	{Field Name} has an invalid format
Certificate	GET	400	11024	{Field Name} value is invalid
Certificate	GET	401	no error structure	Note: Message text in the body: Unauthorized
Certificate	GET	401	no error structure	Note: Message text in the body: The incoming token has expire d
Certificate	GET	404	11011	The Id was not found
Certificate	GET	404	11028	URL requested was not found
Certificate	GET	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	DELETE	400	11025	Path parameter <PathFieldName> has an invalid format [Note: replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Chatbots	DELETE	400	13206	The Chatbot can't be updated or deleted by a requestor who is not associated with it

API	Operation	HTTP	code	message
Chatbots	DELETE	400	13207	The Chatbot can't be updated or deleted when the Verified field is pending
Chatbots	DELETE	401	no error structure	Note: Message text in the body: Unauthorized
Chatbots	DELETE	401	no error structure	Note: Message text in the body: The incoming token has expire d
Chatbots	DELETE	404	11028	URL requested was not found
Chatbots	DELETE	404	13212	The Chatbot requested was not found
Chatbots	DELETE	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	DELETE	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	GET	400	11024	{Field Name} value is invalid
Chatbots	GET	401	no error structure	Note: Message text in the body: Unauthorized
Chatbots	GET	401	no error structure	Note: Message text in the body: The incoming token has expire d
Chatbots	GET	404	11028	URL requested was not found
Chatbots	GET	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	GET	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	GET{id}	400	11024	{Field Name} value is invalid
Chatbots	GET{id}	400	11025	Path parameter <PathFieldName> has an invalid format [Note: replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Chatbots	GET{id}	401	no error structure	Note: Message text in the body: Unauthorized
Chatbots	GET{id}	401	no error structure	Note: Message text in the body: The incoming token has expire d
Chatbots	GET{id}	404	11028	URL requested was not found

API	Operation	HTTP	code	message
Chatbots	GET{id}	404	13212	The Chatbot requested was not found
Chatbots	GET{id}	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	GET{id}	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	GET{id}documents	400	11024	{Field Name} value is invalid
Chatbots	GET{id}documents	400	11025	Path parameter <PathFieldName> has an invalid format [NOTE: replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Chatbots	GET{id}documents	401	no error structure	Note: Message text in the body: Unauthorized
Chatbots	GET{id}documents	401	no error structure	Note: Message text in the body: The incoming token has expired
Chatbots	GET{id}documents	404	11028	URL requested was not found
Chatbots	GET{id}documents	404	13212	The Chatbot requested was not found
Chatbots	GET{id}documents	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	GET{id}documents	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	PATCH	400	11000	{Field Name} requires a non-blank value
Chatbots	PATCH	400	11001	{Field Name} length must be minimum <n> and maximum <n>
Chatbots	PATCH	400	11002	{Field Name} has an invalid format
Chatbots	PATCH	400	11003	{Field Name} length must be maximum <n>
Chatbots	PATCH	400	11004	Invalid syntax present in the request
Chatbots	PATCH	400	11006	PartnerId should not be specified for internal Chatbots

API	Operation	HTTP	code	message
Chatbots	PATCH	400	11016	ServiceIcon is required when <Field Name > is specified [Note: where {Field Name} may be ServiceIconSN, SNJurisdiction or ServiceIconOwner]
Chatbots	PATCH	400	11017	SNJurisdiction is required when ServiceIconSN is specified
Chatbots	PATCH	400	11021	When any of the BrandContactInfo fields (FirstName, LastName, EmailAddress, Title, TelephoneNumber) are specified, then all of them must be specified as a non-blank value
Chatbots	PATCH	400	11025	Path parameter <PathFieldName> has an invalid format [Note: replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Chatbots	PATCH	400	11026	SNJurisdiction, ServiceIconSN and ServiceIconOwner should be null when ServiceIcon is null
Chatbots	PATCH	400	13200	The PartnerId was not found
Chatbots	PATCH	400	13206	The Chatbot can't be updated or deleted by a requestor who is not associated with it
Chatbots	PATCH	400	13207	The Chatbot can't be updated or deleted when the Verified field is pending
Chatbots	PATCH	400	13208	No action can be taken on a suspended, deleted or inactive chatbot
Chatbots	PATCH	400	13209	The Verify field value ‘not-started’ can’t be specified when the Chatbot verification status is ‘complete’
Chatbots	PATCH	400	13216	When any of the ServiceIcon serial number fields (ServiceIconSN, SNJurisdiction & ServiceIconOwner) are present, then all of them must be present

API	Operation	HTTP	code	message
Chatbots	PATCH	400	13221	${accountType} ${accountId} is NOT active [where accountType= brand, partner or RCS SP, and accountId=BrandId, PartnerId or NetworkProviderId]
Chatbots	PATCH	401	no error structure	Note: Message text in the body: Unauthorized
Chatbots	PATCH	401	no error structure	Note: Message text in the body: The incoming token has expire d
Chatbots	PATCH	404	11028	URL requested was not found
Chatbots	PATCH	404	13212	The Chatbot requested was not found
Chatbots	PATCH	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	PATCH	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	POST	400	11000	{Field Name} requires a non-blank value
Chatbots	POST	400	11001	{Field Name} length must be minimum <n> and maximum <n>
Chatbots	POST	400	11002	{Field Name} has an invalid format
Chatbots	POST	400	11003	{Field Name} length must be maximum <n>
Chatbots	POST	400	11004	Invalid syntax present in the request
Chatbots	POST	400	11006	PartnerId should not be specified for internal Chatbots
Chatbots	POST	400	11007	ServiceIcon is required when Brand does not have a verified DefaultIcon
Chatbots	POST	400	11016	ServiceIcon is required when <Field Name > is specified [Note: where {Field Name} may be ServiceIconSN, SNJurisdiction or ServiceIconOwner]

API	Operation	HTTP	code	message
Chatbots	POST	400	11017	SNJurisdiction is required when ServiceIconSN is specified
Chatbots	POST	400	13200	The PartnerId was not found
Chatbots	POST	400	13201	The BrandId was not found
Chatbots	POST	400	13202	The NetworkProvider was not found
Chatbots	POST	400	13217	Partner account does not match request
Chatbots	POST	400	13218	NetworkProvider does not match request
Chatbots	POST	400	13221	${accountType} ${accountId} is NOT active [Note: where accountType= brand, Partner or mno, and accountId=BrandId, PartnerId or NetworkProviderId]
Chatbots	POST	401	no error structure	Note: Message text in the body: Unauthorized
Chatbots	POST	401	no error structure	Note: Message text in the body: The incoming token has expire d
Chatbots	POST	403	13210	Partner entities are not permitted to create Internal Chatbots
Chatbots	POST	403	13219	Only a Partner or Mobile Network Operator is permitted to create a Chatbot
Chatbots	POST	404	11028	URL requested was not found
Chatbots	POST	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Chatbots	POST	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Network Providers	GET	401	no error structure	Note: Message text in the body: Unauthorized
Network Providers	GET	401	no error structure	Note: Message text in the body: The incoming token has expire d
Network Providers	GET	403	11027	The request should be from a verified Partner

API	Operation	HTTP	code	message
Network Providers	GET	404	11028	URL requested was not found
Network Providers	GET	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Network Providers	GET	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Notification	DELETE	401	no error structure	Note: Message text in the body: Unauthorized
Notification	DELETE	401	no error structure	Note: Message text in the body: The incoming token has expired
Notification	DELETE	404	11028	URL requested was not found
Notification	DELETE	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Notification	GET	401	no error structure	Note: Message text in the body: Unauthorized
Notification	GET	401	no error structure	Note: Message text in the body: The incoming token has expired
Notification	GET	404	11028	URL requested was not found
Notification	GET	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Notification	PUT	400	11000	{Field Name} requires a non-blank value
Notification	PUT	400	11002	{Field Name} has an invalid format
Notification	PUT	400	11024	{Field Name} value is invalid
Notification	PUT	401	no error structure	Note: Message text in the body: Unauthorized
Notification	PUT	401	no error structure	Note: Message text in the body: The incoming token has expired
Notification	PUT	404	11028	URL requested was not found

API	Operation	HTTP	code	message
Notification	PUT	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	DELETE	400	11025	Path parameter <PathFieldName> has an invalid format [Note: replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Partners	DELETE	400	13204	The entity has an active Chatbot therefore request for deletion can't be completed
Partners	DELETE	400	21115	Path parameter id does not belong to a {Partner , Brand} entity [Note: where "Partner" is used for Partner API, and "Brand" for Brand API]
Partners	DELETE	400	21122	Non-active entities cannot be updated
Partners	DELETE	400	21123	Entity is currently going through the verification process. Please try again later
Partners	DELETE	400	24309	Requestor must be an verified entity
Partners	DELETE	400	24312	Requestor must be an active entity
Partners	DELETE	401	no error structure	Note: Message text in the body: Unauthorized
Partners	DELETE	401	no error structure	Note: Message text in the body: The incoming token has expired
Partners	DELETE	403	13205	The requestor can't delete itself
Partners	DELETE	403	24302	The request failed because the requestor did not create the entity
Partners	DELETE	404	11028	URL requested was not found
Partners	DELETE	404	24301	The requestor entity was not found
Partners	DELETE	404	24400	The entity requested was not found
Partners	DELETE	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator

API	Operation	HTTP	code	message
Partners	DELETE	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	GET	400	11024	{Field Name} value is invalid
Partners	GET	400	24309	Requestor must be an verified entity
Partners	GET	401	no error structure	Note: Message text in the body: Unauthorized
Partners	GET	401	no error structure	Note: Message text in the body: The incoming token has expire d
Partners	GET	403	24305	The requested entity can't be retrieved by an entity with the given requestor entity type or the entity was not authorized to request this resource
Partners	GET	404	11028	URL requested was not found
Partners	GET	404	24301	The requestor entity was not found
Partners	GET	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	GET	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	GET{id}	400	11025	Path parameter <PathFieldName> has an invalid format [Note: replacing <PathFieldName> with ParterId, BrandId or ChatbotId]
Partners	GET{id}	400	21115	Path parameter id does not belong to a {Partner, Brand} entity [Note: where "Partner"is used for Partner API, and "Brand" for Brand API]
Partners	GET{id}	400	24309	Requestor must be an verified entity
Partners	GET{id}	401	no error structure	Note: Message text in the body: Unauthorized
Partners	GET{id}	401	no error structure	Note: Message text in the body: The incoming token has expire d

API	Operation	HTTP	code	message
Partners	GET{id}	403	24302	The request failed because the requestor did not create the entity
Partners	GET{id}	404	11028	URL requested was not found
Partners	GET{id}	404	24301	The requestor entity was not found
Partners	GET{id}	404	24400	The entity requested was not found
Partners	GET{id}	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	GET{id}	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	PATCH	400	11001	{Field Name} length must be minimum <n> and maximum <n>
Partners	PATCH	400	11002	{Field Name} has an invalid format
Partners	PATCH	400	11003	{Field Name} length must be maximum <n>
Partners	PATCH	400	11004	Invalid syntax present in the request
Partners	PATCH	400	11008	When specified, {Field Name} must be a non-blank value
Partners	PATCH	400	11009	CountryOfIncorp & StateOfIncorp can only be updated in a pair
Partners	PATCH	400	11022	When any of the PartnerAddress fields (StreetAddress1, City, GoverningDistrict, PostalCode, Country) are specified, then are missing and must be specified as a non-blank value.
Partners	PATCH	400	11023	When any of the PartnerContactInfo fields (FirstName, LastName, EmailAddress, Title, TelephoneNumber) are specified, then all of them must be specified as a non-blank value
Partners	PATCH	400	11025	Path parameter <PathFieldName> has an invalid format [Note: replacing

API	Operation	HTTP	code	message
				<PathFieldName> with ParterId, BrandId or ChatbotId]
Partners	PATCH	400	11029	{Field Name} can't be changed on a completed entity [Note: replacing Field Name with Country
Partners	PATCH	400	11030	Verified must be set to complete when field updates will require reverification
Partners	PATCH	400	11031	The calculated converted image size of the {fieldname} base64 encoded string exceeds the maximum 2 MB limit [Note: Replacing fieldname with DefaultIcon or ServiceIcon]
Partners	PATCH	400	11033	{Field Name} can't be changed on a verified entity [Note: replacing {Field Name} with either "RegNumber" or "RefNumber"]
Partners	PATCH	400	21101	The entity can't be updated when the status is complete
Partners	PATCH	400	21103	{Field Name} value is invalid [Note: where {Field Name} is CountryOfIncorp or Country]
Partners	PATCH	400	21104	StateOfIncorp value is invalid
Partners	PATCH	400	21105	GoverningDistrict value is invalid
Partners	PATCH	400	21115	Path parameter id does not belong to a {Partner, Brand} entity [Note: where "Partner"is used for Partner API, and "Brand" for Brand API]
Partners	PATCH	400	21118	Verify must be set to complete when field updates will require reverification of the entity or logo
Partners	PATCH	400	21122	Non-active entities cannot be updated
Partners	PATCH	400	21123	Entity is currently going through the verification process. Please try again later
Partners	PATCH	400	21300	An entity with the same RegNumber exists. Therefore,

API	Operation	HTTP	code	message
				the entity creation request can't be honored
Partners	PATCH	400	24309	Requestor must be an verified entity
Partners	PATCH	400	24312	Requestor must be an active entity
Partners	PATCH	401	no error structure	Note: Message text in the body: Unauthorized
Partners	PATCH	401	no error structure	Note: Message text in the body: The incoming token has expire d
Partners	PATCH	403	11010	In order to update a Partner, the requestor must be an RCS Service Provider
Partners	PATCH	403	24302	The request failed because the requestor did not create the entity
Partners	PATCH	404	11028	URL requested was not found
Partners	PATCH	404	24301	The requestor entity was not found
Partners	PATCH	404	24400	The entity requested was not found
Partners	PATCH	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	PATCH	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	POST	400	11000	{Field Name} requires a non-blank value
Partners	POST	400	11001	{Field Name} length must be minimum <n> and maximum <n>
Partners	POST	400	11002	{Field Name} has an invalid format
Partners	POST	400	11003	{Field Name} length must be maximum <n>
Partners	POST	400	11004	Invalid syntax present in the request
Partners	POST	400	21103	{Field Name} value is invalid [Note: where {Field Name} is CountryOfIncorp or Country]
Partners	POST	400	21104	StateOfIncorp value is invalid

API	Operation	HTTP	code	message
Partners	POST	400	21105	GoverningDistrict value is invalid
Partners	POST	400	21300	An entity with the same RegNumber exists. Therefore, the entity creation request can't be honored
Partners	POST	400	24308	Where Requestor must be a registeredRCS Service Provider in order to create a Partner
Partners	POST	400	24312	Requestor must be an active entity
Partners	POST	401	no error structure	Note: Message text in the body: Unauthorized
Partners	POST	401	no error structure	Note: Message text in the body: The incoming token has expire d
Partners	POST	404	11028	URL requested was not found
Partners	POST	500	10500	Internal Server Error occurred. Please retry later or contact your VA Administrator
Partners	POST	500	25100	Internal Server Error occurred. Please retry later or contact your VA Administrator

Annex C Document Management

C.1 Document History

Version	Date	Brief Description of Change	Type	Approval Authority	Editor / Company
0.12	9 September 2021	Final Baseline from subgroup	Baseline	GSG Subgroup	Haraburda / iconectiv
1.0	25 November 2021	CR1001 First Version		ISAG	Haraburda / iconectiv

C.2 Other Information

Type	Description
Document Owner	GSG RCS VA API Working Group
Editor(s) / Company	John Haraburda – iconectiv Surinder Anand - DotGo